



---

The following document contains information on Cypress products. Although the document is marked with the name "Spansion", the company that originally developed the specification, Cypress will continue to offer these products to new and existing customers.

**Continuity of Specifications**

There is no change to this document as a result of offering the device as a Cypress product. Any changes that have been made are the result of normal document improvements and are noted in the document history page, where supported. Future revisions will occur when appropriate, and changes will be noted in a document history page.

**Continuity of Ordering Part Numbers**

Cypress continues to support existing part numbers. To order these products, please use only the Ordering Part Numbers listed in this document.

**For More Information**

Please contact your local sales office for additional information about Cypress products and solutions.

**About Cypress**

Cypress (NASDAQ: CY) delivers high-performance, high-quality solutions at the heart of today's most advanced embedded systems, from automotive, industrial and networking platforms to highly interactive consumer and mobile devices. With a broad, differentiated product portfolio that includes NOR flash memories, F-RAM™ and SRAM, Traveo™ microcontrollers, the industry's only PSoC® programmable system-on-chip solutions, analog and PMIC Power Management ICs, CapSense® capacitive touch-sensing controllers, and Wireless BLE Bluetooth® Low-Energy and USB connectivity solutions, Cypress is committed to providing its customers worldwide with consistent innovation, best-in-class support and exceptional system value.

# MB9D560 Series

32-bit Microcontroller  
Spansion® Traveo™ Family  
MB9DF564/F565/F566

*Hardware Manual*

---



**ARM®**



## Preface

Thank you for your continued use of Spansion semiconductor products.

Read this manual and "Data Sheet" thoroughly before using MB9D560 series.

### **Purpose of this manual and intended readers**

This series has 32-bit microcontrollers for automotive motor control.

This manual explains the functions and operations of this family and describes how it is used. The manual is intended for engineers engaged in the actual development of products using this family.

### **Microcontroller support information:**

**<http://www.spansion.com/support/microcontrollers/>**





## H A R D W A R E   M A N U A L

---

# Table of Contents

CHAPTER 1:	Overview .....	27
1.	Overview .....	28
2.	Product Lineup .....	34
3.	Configuration .....	35
4.	Pin Assignment .....	37
5.	Pin Description .....	41
6.	I/O Circuit Type .....	59
7.	Memory Map .....	62
8.	I/O Map .....	65
9.	Legend .....	70
9.1.	Register Attribute .....	70
10.	Abbreviations .....	72
CHAPTER 2:	CPU .....	75
1.	Overview .....	76
2.	Notes .....	77
CHAPTER 3:	Operation Mode .....	79
1.	Overview .....	80
2.	Configuration .....	81
3.	Operation .....	82
3.1.	Pin Settings .....	82
3.2.	Fetching the Operation Mode .....	82
4.	Register .....	83
4.1.	Mode Register (MODEC_MODER) .....	84
CHAPTER 4:	Reset .....	87
1.	Overview .....	88
2.	Configuration .....	89
3.	Operation .....	93
3.1.	Reset Factor .....	94
3.2.	Internal Reset of Device .....	112
3.3.	Reset Sequence .....	117
3.4.	Operations after Reset Release .....	118
4.	Registers .....	126
4.1.	Reset Control Register 0 (SYSC_RSTCNTR0) .....	127
4.2.	Reset Control Register 1 (SYSC_RSTCNTR1) .....	129
4.3.	User Reset Factor Register (SYSC_RSTCAUSEUR) .....	131
4.4.	User Extended CSV Reset Factor Register (SYSC_EXCSVRSTCAUSEUR) .....	135
4.5.	BootROM Reset Factor Register (SYSC_RSTCAUSEBT) .....	136
4.6.	BootROM Extended CSV Reset Factor Register (SYSC_EXCSVRSTCAUSEBT) .....	140
4.7.	Watchdog Reset Boot CPU Selection Register (SYSC_WRBOOTCPUSEL) .....	141
CHAPTER 5:	Clock System .....	147
1.	Overview .....	148
2.	Configuration .....	150
2.1.	Overall Configuration and Block Diagrams of Clock System .....	150
2.2.	Configuration and Block Diagram of Clock Generator .....	152
2.3.	Configuration and Block Diagram of Source Clock Selection Section .....	154
2.4.	Configuration and Block Diagram of Clock Distributor/Divider .....	156
3.	Operation .....	159
3.1.	Source Clock Generation Control .....	159

3.2.	PLL Clock Control.....	160
3.3.	Source Clock Selection Control.....	162
3.4.	Clock Distribution/Division Control .....	163
3.5.	Clock Gear Operation.....	165
3.6.	Oscillation Stabilization Wait Time.....	169
3.7.	Interrupt Factor.....	169
4.	Setting Procedure Example.....	170
4.1.	RUN Operation Setting.....	171
4.2.	PSS Operation Setting .....	172
5.	Registers.....	173
5.1.	Common Configuration Registers .....	174
5.2.	Clock Output Function Register.....	180
6.	Usage Precautions .....	182
CHAPTER 6:	Low-power Consumption.....	183
1.	Overview .....	184
2.	Configuration.....	185
2.1.	Block Diagram .....	185
3.	Operations.....	186
3.1.	Low-power Consumption States.....	186
3.2.	CPU Operation States.....	190
3.3.	Profile Setting Items .....	191
3.4.	Profiles .....	193
3.5.	Interrupts .....	199
3.6.	Bus Error Response .....	200
4.	Operation Procedure .....	201
5.	Registers.....	212
5.1.	Protection Register Group .....	216
5.2.	RUN Profile Register Group .....	217
5.3.	PSS Profile Register Group.....	238
5.4.	APPLIED Profile Register Group.....	259
5.5.	Status Profile Register Group.....	279
5.6.	System Register Group .....	301
5.7.	Special Setting Register Group .....	321
5.8.	Debug Register Group.....	324
6.	Other .....	327
6.1.	Restrictions on Transitioning to the PSS .....	327
CHAPTER 7:	Low-voltage Detection.....	329
1.	Overview .....	330
2.	Configuration.....	331
3.	Operation .....	332
4.	Setting Procedure Examples.....	334
5.	Operation Examples.....	336
6.	Precautions for Using .....	339
CHAPTER 8:	Clock Supervisor .....	341
1.	Overview .....	342
2.	Configuration.....	343
3.	Operation .....	347
4.	Setting Procedure Example.....	349
5.	Registers.....	351
5.1.	Main Clock Supervisor Configuration Register 00 (SYSC_CSVMOFCGR00) .....	352
5.2.	Main Clock Supervisor Configuration Register 01 (SYSC_CSVMOFCGR01) .....	354

5.3.	Main Clock Supervisor Configuration Register 10 (SYSC_CSMOCFGR10)	356
5.4.	Main Clock Supervisor Configuration Register 11 (SYSC_CSMOCFGR11)	358
5.5.	PLL Clock Supervisor Configuration Register 0 (SYSC_CSVPLLCFGR0)	360
5.6.	PLL Clock Supervisor Configuration Register 1 (SYSC_CSVPLLCFGR1)	362
5.7.	Sub-system PLL Clock Supervisor Configuration Register 0 (SYSC_CSVSSCFGR0)	363
5.8.	Sub-system PLL Clock Supervisor Configuration Register 1 (SYSC_CSVSSCFGR1)	365
5.9.	Clock Supervisor Output Enable Register (SYSC_CSVOUTER)	366
5.10.	Clock Supervisor Test Register (SYSC_CSVTESTR)	367
6.	Precautions for Using	368
CHAPTER 9:	Source Clock Timer	369
1.	Overview	370
2.	Configuration	371
3.	Operation	373
4.	Setting Procedure Example	374
5.	Registers	375
5.1.	Fast-CR Clock Timer Trigger Register (SYSC_FCRCTTRGR)	376
5.2.	Fast-CR Clock Timer Control Register (SYSC_FCRCTCNTR)	377
5.3.	Fast-CR Clock Timer Compare Prescaler Register (SYSC_FCRCTCPR)	378
5.4.	Fast-CR Clock Timer Status Register (SYSC_FCRCTSTR)	380
5.5.	Fast-CR Clock Timer Interrupt Enable Register (SYSC_FCRCTINTER)	382
5.6.	Fast-CR Clock Timer Interrupt Clear Register (SYSC_FCRCTICLR)	383
5.7.	Slow-CR Clock Timer Trigger Register (SYSC_SCRCTTRGR)	384
5.8.	Slow-CR Clock Timer Control Register (SYSC_SCRCTCNTR)	385
5.9.	Slow-CR Clock Timer Compare Prescaler Register (SYSC_SCRCTCPR)	386
5.10.	Slow-CR Clock Timer Status Register (SYSC_SCRCTSTR)	388
5.11.	Slow-CR Clock Timer Interrupt Enable Register (SYSC_SCRCTINTER)	390
5.12.	Slow-CR Clock Timer Interrupt Clear Register (SYSC_SCRCTICLR)	391
5.13.	Main Clock Timer Trigger Register (SYSC_MOCTTRGR)	392
5.14.	Main Clock Timer Control Register (SYSC_MOCTCNTR)	393
5.15.	Main Clock Timer Compare Prescaler Register (SYSC_MOCTCPR)	394
5.16.	Main Clock Timer Status Register (SYSC_MOCTSTR)	396
5.17.	Main Clock Timer Interrupt Enable Register (SYSC_MOCTINTER)	398
5.18.	Main Clock Timer Interrupt Clear Register (SYSC_MOCTICLR)	399
6.	Precautions for Using	400
CHAPTER 10:	Interrupt Controller	401
1.	Overview	402
2.	Configuration	403
3.	Operation	404
4.	Setting Procedure Examples	418
5.	Registers	419
5.1.	IRC NMI Vector Address Status Register (IRCn_NMIVAS)	420
5.2.	IRC NMI Status Register (IRCn_NMIST)	421
5.3.	IRC IRQ Vector Address Status Register (IRCn_IRQVAS)	422
5.4.	IRC IRQ Status Register (IRCn_IRQST)	423
5.5.	IRC NMI Vector Address Register (IRCn_NMIVAr)	425
5.6.	IRC IRQ Vector Address Register (IRCn_IRQVAr)	426
5.7.	IRC NMI Priority Level Register (IRCn_NMIPL0)	427
5.8.	IRC NMI Priority Level Register (IRCn_NMIPL1 to IRCn_NMIPL7)	429
5.9.	IRC IRQ Priority Level Register (IRCn_IRQPL0 to IRCn_IRQPL127)	431

5.10.	IRC NMI Software Interrupt Set Register (IRCN_NMIS) .....	433
5.11.	IRC NMI Software Interrupt Reset Register (IRCN_NMIR) .....	434
5.12.	IRC NMI Software Interrupt Status Register (IRCN_NMISIS) .....	435
5.13.	IRC IRQ Software Interrupt Set Register (IRCN_IRQS0 to IRCN_IRQS15)....	436
5.14.	IRC IRQ Software Interrupt Reset Register (IRCN_IRQR0 to IRCN_IRQR15)	437
5.15.	IRC IRQ Software Interrupt Status Register (IRCN_IRQSIS0 to IRCN_IRQSIS15) .....	438
5.16.	IRC IRQ Channel Enable Set Register (IRCN_IRQCES0 to IRCN_IRQCES15)...	439
5.17.	IRC IRQ Channel Enable Clear Register (IRCN_IRQCEC0 to IRCN_IRQCEC15) .....	440
5.18.	IRC IRQ Channel Enable Setting Register (IRCN_IRQCE0 to IRCN_IRQCE15).. .....	441
5.19.	IRC NMI Hold Clear Register (IRCN_NMIHC) .....	442
5.20.	IRC NMI Hold Status Register (IRCN_NMIHS).....	443
5.21.	IRC IRQ Hold Clear Register (IRCN_IRQHC).....	444
5.22.	IRC IRQ Hold Status Register (IRCN_IRQHS0 to IRCN_IRQHS15).....	445
5.23.	IRC IRQ Priority Level Mask Register (IRCN_IRQPLM) .....	446
5.24.	IRC Control/Status Register (IRCN_CSR) .....	447
5.25.	IRC NMI RAW Status Register (IRCN_NMIRS) .....	449
5.26.	IRC NMI Preprocessed Status Register (IRCN_NMIPS) .....	450
5.27.	IRC IRQ RAW Status Register (IRCN_IRQRS0 to IRCN_IRQRS15) .....	451
5.28.	IRC IRQ Preprocessed Status Register (IRCN_IRQPS0 to IRCN_IRQPS15).	452
5.29.	IRC Unlock Register (IRCN_UNLOCK).....	453
5.30.	IRC ECC Error Interrupt Register (IRCN_EEI).....	454
5.31.	IRC ECC Address Number Register (IRCN_EAN).....	456
5.32.	IRC ECC Test Register (IRCN_ET).....	457
5.33.	IRC ECC Error Bit Register (IRCN_EEB0 to IRCN_EEB1) .....	458
5.34.	IRC ECC Error Bit Register (IRCN_EEB2).....	459
5.35.	IRC NMI Vector Address Status Register (IRCN_NMIVASBR) .....	460
5.36.	IRC NMI Vector Address Status Mirror Register (IRC_NMIVASBR).....	461
5.37.	IRC ECC Error Vector Address Register (IRCN_IRQEEVA) .....	462
6.	Others .....	463
CHAPTER 11:	BootROM Hardware Interface .....	465
1.	Overview .....	466
2.	Configuration .....	467
3.	Operation .....	468
4.	Setting Procedure Example.....	469
5.	Registers.....	470
5.1.	EXCFG Lock Release Register (EXCFG_UNLOCK) .....	471
5.2.	EXCFG Setting Register (EXCFG_CNFG).....	472
5.3.	EXCFG Inactive Set - Undefined Instruction Vector Register (EXCFG_UNDEFINACT).....	473
5.4.	EXCFG Inactive Set - Supervisor Call Vector Register (EXCFG_SVCINACT)	474
5.5.	EXCFG Inactive Set - Prefetch Abort Vector Register (EXCFG_PABORTINACT) .....	475
5.6.	EXCFG Inactive Set - Data Abort Vector Register (EXCFG_DABORTINACT)	476
5.7.	EXCFG Active Set - Undefined Instruction Vector Register (EXCFG_UNDEFACT) .....	477
5.8.	EXCFG Active Set - Supervisor Call Vector Register (EXCFG_SVCACT) .....	478
5.9.	EXCFG Active Set - Prefetch Abort Vector Register (EXCFG_PABORTACT)	479

5.10.	EXCFG Active Set - Data Abort Vector Register (EXCFG_DABORTACT) .....	480
6.	Others .....	481
CHAPTER 12:	BootROM Software Interface.....	483
1.	Overview .....	484
2.	BootROM Markers.....	485
2.1.	Overview of BootROM Markers.....	486
2.2.	Maker List.....	487
2.3.	Flash Security Marker (SDR_FSECM) .....	489
2.4.	Debugger Connection Enable Marker (SDR_DSM) .....	490
2.5.	Debugger Security Key Marker 0 (SDR_DSKM0) .....	491
2.6.	Debugger Security Key Marker 1 (SDR_DSKM1) .....	492
2.7.	Debugger Security Key Marker 2 (SDR_DSKM2) .....	493
2.8.	Debugger Security Key Marker 3 (SDR_DSKM3) .....	494
2.9.	Debugger Connection Wait Enable Marker (BDR_DWEM) .....	495
2.10.	Alternative Boot Vector Marker (BDR_ABVM).....	496
2.11.	Alternative Boot Vector Enable Marker (BDR_ABVEM) .....	497
2.12.	Hardware Watchdog Interrupt Configuration Marker (WDR_INTM).....	498
2.13.	Hardware Watchdog Trigger 0 Configuration Marker (WDR_TRG0CFGM) ...	499
2.14.	Hardware Watchdog Trigger 1 Configuration Marker (WDR_TRG1CFGM) ...	500
2.15.	Hardware Watchdog Lower Limit RUN Setting Marker (WDR_RUNLLM) .....	501
2.16.	Hardware Watchdog Upper Limit RUN Setting Marker (WDR_RUNULM) .....	502
2.17.	Hardware Watchdog Lower Limit PSS Setting Marker (WDR_PSSLLM) .....	503
2.18.	Hardware Watchdog Upper Limit PSS Setting Marker (WDR_PSSULM).....	504
2.19.	Hardware Watchdog Reset Delay Counter Marker (WDR_RSTDLYM).....	505
2.20.	Hardware Watchdog Configuration Marker (WDR_CFGM) .....	506
2.21.	Hardware Watchdog Configuration Enable Marker (WDR_CEM).....	507
3.	BootROM Operation.....	508
4.	Notes.....	513
CHAPTER 13:	NMI Distribution.....	515
1.	Overview .....	516
2.	Configuration.....	517
3.	Operation .....	518
4.	Setting Procedure Examples.....	519
5.	Registers .....	520
5.1.	NMID Lock Release Register (NMID_UNLOCK) .....	521
5.2.	NMID Lock Status Register (NMID_LST) .....	522
5.3.	NMID NMIm Distribution Enable Register (NMID_DISTm).....	523
6.	Others .....	524
CHAPTER 14:	External Interrupt.....	525
1.	Overview .....	526
2.	Configuration.....	527
2.1.	Block Diagrams .....	527
3.	Operation .....	528
4.	Setting Procedure Example.....	530
5.	Registers.....	531
5.1.	External Interrupt Enable Register (EICxx_ENIR) .....	532
5.2.	External Interrupt Enable Set Register (EICxx_ENISR) .....	533
5.3.	External Interrupt Enable Clear Register (EICxx_ENICR) .....	534
5.4.	External Interrupt Factor Register (EICxx_EIRR) .....	535
5.5.	External Interrupt Factor Clear Register (EICxx_EIRCR) .....	536
5.6.	Noise Filter Enable Register (EICxx_NFER) .....	537

5.7.	Noise Filter Enable Set Register (EICxx_NFESR) .....	538
5.8.	Noise Filter Enable Clear Register (EICxx_NFECR) .....	539
5.9.	External Interrupt Level Registers (EICxx_ELVR0 to 3) .....	540
5.10.	Non-maskable Interrupt Register (EICxx_NMIR) .....	542
5.11.	DMA Request Enable Register (EICxx_DRER) .....	543
5.12.	DMA Request Enable Set Register (EICxx_DRESR) .....	544
5.13.	DMA Request Enable Clear Register (EICxx_DRECR) .....	545
5.14.	DMA Request Flag Register (EICxx_DRFR) .....	546
CHAPTER 15:	Security .....	547
1.	Overview .....	548
2.	Access Restriction with Each Security .....	549
3.	Notes on Security .....	550
CHAPTER 16:	TCFLASH .....	551
1.	Overview .....	552
2.	Configuration .....	553
2.1.	Block Diagrams .....	553
2.2.	Address/Sector Map of TCFLASH .....	555
3.	Operation .....	557
3.1.	Operation Mode of TCFLASH .....	557
3.2.	Programming and Erasing .....	557
3.3.	Interleaved Access .....	558
3.4.	TCM Region and AXI Region .....	559
3.5.	Automatic Algorithm .....	560
3.6.	Automatic Algorithm Execution State .....	562
3.7.	Insert Wait Cycle .....	568
3.8.	ECC Generation and Check .....	569
3.9.	Interrupt .....	572
3.10.	Bus Error Response .....	573
3.11.	Flash Security .....	574
4.	Setting Procedure Example .....	575
4.1.	Setting Wait Cycle Count .....	575
4.2.	Reading Flash Memory State/Transition to Reset State .....	575
4.3.	Programming Procedure .....	576
4.4.	Macro Erasure Procedure .....	579
4.5.	Sector Erasure Procedure .....	579
4.6.	Flash Security Setting Method .....	586
4.7.	Flash Security Unlocking Method .....	586
5.	Registers .....	587
5.1.	TCFLASH0/1/2/3 Configuration Protection Key Register (TCFCFGn_FCPROTKEY) .....	588
5.2.	TCFLASH0/1/2/3 Configuration Register (TCFCFGn_FCFGR) .....	589
5.3.	TCFLASH0/1/2/3 ECC Control Register (TCFCFGn_FECCTRL) .....	591
5.4.	TCFLASH0/1/2/3 Data Bit Error Injection Register (TCFCFGn_FDATEIR) ....	592
5.5.	TCFLASH0/1/2/3 ECC Bit Error Injection Register (TCFCFGn_FECCEIR) ...	593
5.6.	TCFLASH0/1/2/3 Interrupt Control Register (TCFCFGn_FICTRL0/1) .....	595
5.7.	TCFLASH0/1/2/3 Status Register (TCFCFGn_FSTAT0/1) .....	597
5.8.	TCFLASH0/1/2/3 SEC Interrupt Register (TCFCFGn_FSECIR) .....	599
5.9.	TCFLASH0/1/2/3 ECC Error Address Register (TCFCFGn_FECCEAR) .....	601
5.10.	TCFLASH0/1/2/3 Module Identification Register (TCFCFGn_FMIDR) .....	602
5.11.	TCFLASH0/1/2/3 Uncorrectable Error Detection Interrupt Register (TCFCFGn_FUCEDIR) .....	603

5.12.	TCFLASH0/1/2/3 Uncorrectable Error Address Register (TRCFGn_FUCEAR) .	605
6.	Others .....	606
CHAPTER 17:	TCRAM Interface.....	607
1.	Overview .....	608
2.	Configuration.....	609
3.	Operation .....	611
4.	Setting Procedure Examples.....	615
5.	Registers.....	617
5.1.	TCRAM IF Configuration Register 0 (TRCFGn_TCMCFG0) .....	618
5.2.	TCRAM IF Configuration Register 1 (TRCFGn_TCMCFG1) .....	620
5.3.	TCRAM IF Unlock Register (TRCFGn_TCMUNLOCK).....	621
5.4.	TCRAM IF Test Error Address Register 0 (TRCFGn_TEAR0).....	622
5.5.	TCRAM IF Test Error Address Register 1 (TRCFGn_TEAR1).....	624
5.6.	TCRAM IF Test Error Address Register 2 (TRCFGn_TEAR2).....	626
5.7.	TCRAM IF Test End Address Register (TRCFGn_TAEAR) .....	628
5.8.	TCRAM IF Test Start Address Register (TRCFGn_TASAR).....	629
5.9.	TCRAM IF Test Pseudo Error Generation Control Register (TRCFGn_TFECR) ..	630
5.10.	TCRAM IF Test Initialization Function Register (TRCFGn_TICR) .....	631
5.11.	TCRAM IF Test Diagnosis Function Register (TRCFGn_TTCR) .....	632
5.12.	TCRAM IF Test Soft Reset Generation Control Register (TRCFGn_TSRCR) .....	634
5.13.	TCRAM IF Test Key Code Control Register (TRCFGn_TKCCR) .....	635
6.	Notes.....	636
CHAPTER 18:	WorkFLASH .....	637
1.	Overview .....	638
2.	Configuration.....	639
2.1.	Block Diagrams .....	639
2.2.	Address/Sector Maps in User Mode .....	640
3.	Operation .....	643
3.1.	Read.....	643
3.2.	Write and Erase.....	643
3.3.	Transfer of Write Data with DMA .....	644
3.4.	Insert Wait Cycle .....	644
3.5.	ECC Generation and Check .....	645
3.6.	Interrupt.....	648
3.7.	Bus Error Response .....	649
3.8.	WorkFLASH Access Restrictions.....	649
4.	Setting Procedure.....	650
4.1.	Setting the Number of Wait Cycles.....	650
4.2.	Write Procedure (Example) .....	650
4.3.	Sector Erase Procedure .....	652
5.	Registers.....	653
5.1.	WorkFLASH00/01/02/03 Configuration Protection Key Register (WFCFGxx_CPR).....	654
5.2.	WorkFLASH00/01/02/03 Configuration Register (WFCFGxx_CR).....	655
5.3.	WorkFLASH00/01/02/03 ECC Control Register (WFCFGxx_ECR).....	657
5.4.	WorkFLASH00/01/02/03 Write Command Sequencer Configuration Register (WFCFGxx_WCR).....	658
5.5.	WorkFLASH00/01/02/03 Write Command Sequencer Status Register (WFCFGxx_WSR).....	659



5.6.	WorkFLASH00/01/02/03 Data Bit Error Injection Register (WFCFGxx_DBEIR) ..	660
5.7.	WorkFLASH00/01/02/03 ECC Bit Error Injection Register (WFCFGxx_EEIR)	661
5.8.	WorkFLASH00/01/02/03 Interrupt Control Register (WFCFGxx_ICR) .....	662
5.9.	WorkFLASH00/01/02/03 Status Register (WFCFGxx_SR) .....	663
5.10.	WorkFLASH00/01/02/03 SEC Interrupt Register (WFCFGxx_SECIR) .....	665
5.11.	WorkFLASH00/01/02/03 ECC Error Address Register (WFCFGxx_EEAR) ...	667
5.12.	WorkFLASH00/01/02/03 Module Identification Register (WFCFGxx_MIR) ....	668
5.13.	WorkFLASH00/01/02/03 Sequencer Command Register (WFCFGxx_SEQCM) .	669
5.14.	WorkFLASH00/01/02/03 Bus Error Response Factor Register (WFCFGxx_BERR) .....	671
5.15.	WorkFLASH00/01/02/03 Bus Error Response Factor Clear Register (WFCFGxx_BERRCLR) .....	674
5.16.	WorkFLASH00/01/02/03 Uncorrectable Error Status Register (WFCFGxx_UCESR) .....	676
5.17.	WorkFLASH00/01/02/03 Uncorrectable Error Address Register (WFCFGxx_UCEAR) .....	677
6.	Others .....	678
CHAPTER 19:	Hardware Watchdog Timer .....	679
1.	Overview .....	680
2.	Configuration .....	681
3.	Operation .....	682
4.	Setting Procedure Example .....	699
5.	Operation Examples .....	700
6.	Registers .....	703
6.1.	Hardware Watchdog Protection Register (HWDG_PROT) .....	704
6.2.	Hardware Watchdog Counter Register (HWDG_CNT) .....	705
6.3.	Hardware Watchdog Reset Factor Register (HWDG_RSTCAUSE) .....	706
6.4.	Hardware Watchdog Trigger 0 Register (HWDG_TRG0) .....	708
6.5.	Hardware Watchdog Trigger 1 Register (HWDG_TRG1) .....	709
6.6.	Hardware Watchdog Interrupt Configuration Register (HWDG_INT) .....	710
6.7.	Hardware Watchdog Interrupt Clear Register (HWDG_INTCLR) .....	712
6.8.	Hardware Watchdog Trigger 0 Configuration Register (HWDG_TRG0CFG) .	713
6.9.	Hardware Watchdog Trigger 1 Configuration Register (HWDG_TRG1CFG) .	714
6.10.	Hardware Watchdog Lower Limit RUN Setting Register (HWDG_RUNLLS) ..	715
6.11.	Hardware Watchdog Upper Limit RUN Setting Register (HWDG_RUNULS) .	716
6.12.	Hardware Watchdog Lower Limit PSS Setting Register (HWDG_PSSLLS) ...	717
6.13.	Hardware Watchdog Upper Limit PSS Setting Register (HWDG_PSSULS) ..	718
6.14.	Hardware Watchdog Reset Delay Counter Register (HWDG_RSTDLY) .....	719
6.15.	Hardware Watchdog Configuration Register (HWDG_CFG) .....	720
6.16.	Hardware Watchdog Lower Limit RUN Current Register (HWDG_RUNLLC) .	723
6.17.	Hardware Watchdog Upper Limit RUN Current Register (HWDG_RUNULC) .	724
6.18.	Hardware Watchdog Lower Limit PSS Current Register (HWDG_PSSLLC) ..	725
6.19.	Hardware Watchdog Upper Limit PSS Current Register (HWDG_PSSULC) .	726
7.	Precautions for Using .....	727
CHAPTER 20:	Software Watchdog Timer .....	729
1.	Overview .....	730
2.	Configuration .....	731
3.	Operation .....	732
4.	Setting Procedure Example .....	748

5.	Operation Examples .....	749
6.	Registers .....	752
6.1.	Software Watchdog Protection Register (SWDGn_PROT) .....	753
6.2.	Software Watchdog Counter Register (SWDGn_CNT) .....	754
6.3.	Software Watchdog Reset Factor Register (SWDGn_RSTCAUSE).....	755
6.4.	Software Watchdog Trigger 0 Register (SWDGn_TRG0) .....	757
6.5.	Software Watchdog Trigger 1 Register (SWDGn_TRG1) .....	758
6.6.	Software Watchdog Interrupt Configuration Register (SWDGn_INT) .....	759
6.7.	Software Watchdog Interrupt Clear Register (SWDGn_INTCLR) .....	761
6.8.	Software Watchdog Trigger 0 Configuration Register (SWDGn_TRG0CFG) .	762
6.9.	Software Watchdog Trigger 1 Configuration Register (SWDGn_TRG1CFG) .	763
6.10.	Software Watchdog Lower Limit RUN Setting Register (SWDGn_RUNLLS) .	764
6.11.	Software Watchdog Upper Limit RUN Setting Register (SWDGn_RUNULS) .	765
6.12.	Software Watchdog Lower Limit PSS Setting Register (SWDGn_PSSLLS) ..	766
6.13.	Software Watchdog Upper Limit PSS Setting Register (SWDGn_PSSULS) ..	767
6.14.	Software Watchdog Reset Delay Counter Register (SWDGn_RSTDLY) .....	768
6.15.	Software Watchdog Configuration Register (SWDGn_CFG).....	769
6.16.	Software Watchdog Lower Limit RUN Current Register (SWDGn_RUNLLC) .	772
6.17.	Software Watchdog Upper Limit RUN Current Register (SWDGn_RUNULC) .	773
6.18.	Software Watchdog Lower Limit PSS Current Register (SWDGn_PSSLLC) .	774
6.19.	Software Watchdog Upper Limit PSS Current Register (SWDGn_PSSULC) .	775
7.	Precautions for Using .....	776
CHAPTER 21:	Bit-band Unit.....	779
1.	Overview .....	780
2.	Operation .....	781
CHAPTER 22:	DMA Controller .....	843
1.	Overview .....	844
2.	Configuration .....	845
3.	Operation .....	846
3.1.	DMA Channels .....	848
3.2.	DMA Client Matrix.....	856
3.3.	DMA Arbiter .....	858
3.4.	DMA AHB Slave Interface.....	861
3.5.	Additional information .....	862
4.	Registers .....	866
4.1.	DMA Controller Global Configuration Register (DMAi_R).....	867
4.2.	DMA Controller Global Completion Interrupt 1 Register (DMAi_DIRQ1) .....	870
4.3.	DMA Controller Global Error Interrupt 1 Register (DMAi_EDIRQ1) .....	871
4.4.	DMA Controller Global Error Interrupt 2 Register (DMAi_EDIRQ2) .....	872
4.5.	DMA Controller Global Completion Interrupt 2 Register (DMAi_DIRQ2) .....	873
4.6.	DMA Controller ID Register (DMAi_ID).....	874
4.7.	DMA Controller Channel Configuration A Register Channel "n" (DMAi_An) ...	875
4.8.	DMA Controller Channel Configuration B Register Channel "n" (DMAi_Bn)...	878
4.9.	DMA Controller Channel Configuration Source Address Register Channel n (DMAi_SAn) .....	882
4.10.	DMA Controller Channel Configuration Destination Address Register Channel n (DMAi_DAn) .....	883
4.11.	DMA Controller Channel Configuration C Register Channel n (DMAi_Cn) .....	884
4.12.	DMA Controller Channel Configuration D Register Channel n (DMAi_Dn) .....	885
4.13.	DMA Controller Channel Configuration E Register Channel n (DMAi_En) .....	888
4.14.	DMA Controller Channel Configuration Source Address Shadow Register	

	Channel "n" (DMAi_SASHDWn).....	889
4.15.	DMA Controller Channel Configuration Destination Address Shadow Register Channel "n" (DMAi_DASHDWn) .....	890
4.16.	DMA Controller Client Matrix Internal Client Interface Configuration Register m (DMAi_CMICICm) .....	891
4.17.	DMA Controller Client Matrix Channel Interface Configuration Register "n" (DMAi_CMCHICn) .....	893
CHAPTER 23:	Memory Protection (AHB).....	895
1.	Overview .....	896
2.	Configuration .....	897
3.	Operation .....	898
4.	Registers .....	903
4.1.	MPU AHB Control Register (MPUHm_CTRL0) .....	904
4.2.	MPU AHB NMI Enable Register (MPUHm_NMIEN) .....	907
4.3.	MPU AHB Memory Error Control Register (MPUHm_MERRC).....	908
4.4.	MPU AHB Memory Error Address Register (MPUHm_MERRA) .....	909
4.5.	MPU AHB Region Control Registers (MPUHm_CTRL1 to 8) .....	910
4.6.	MPU AHB Start Address Registers (MPUHm_SADDR1 to 8) .....	912
4.7.	MPU AHB End Address Registers (MPUHm_EADDR1 to 8).....	913
4.8.	MPU AHB Unlock Register (MPUHm_UNLOCK) .....	914
4.9.	MPU AHB Module ID Register (MPUHm_MID) .....	915
5.	Precautions for Using .....	916
CHAPTER 24:	Time Protection .....	919
1.	Overview .....	920
2.	Configuration .....	921
3.	Operation .....	922
4.	Setting Procedure Examples.....	924
5.	Registers .....	926
5.1.	TPU Lock Release Register (TPUn_UNLOCK).....	927
5.2.	TPU Lock Status Register (TPUn_LST) .....	928
5.3.	TPU Configuration Register (TPUn_CFG).....	929
5.4.	TPU Timer Interrupt Request Register (TPUn_TIR) .....	931
5.5.	TPU Timer Status Register (TPUn_TST).....	932
5.6.	TPU Timer Interrupt Enable Register (TPUn_TIE) .....	933
5.7.	TPU Timer m Control Register 0 (TPUn_TCN0m).....	934
5.8.	TPU Timer m Control Register 1 (TPUn_TCN1m).....	936
5.9.	TPU Timer m Current Count Value Register (TPUn_TCCm) .....	938
6.	Others .....	939
CHAPTER 25:	Inter-processor Communication .....	941
1.	Overview .....	942
2.	Configuration .....	943
3.	Operation .....	944
4.	Setting Procedure Example.....	945
5.	Registers .....	949
5.1.	IPCU Interrupt Status Register (IPCU_ISTRn) .....	950
5.2.	IPCU Mailbox Address Register (IPCU_MARn) .....	951
5.3.	IPCU Mailbox m Request Sender Setting Register (IPCU_MBmSRCR).....	953
5.4.	IPCU Mailbox m Operation Mode Setting Register (IPCU_MBmMR).....	955
5.5.	IPCU Mailbox m Request Send Register (IPCU_MBmSR) .....	956
5.6.	IPCU Mailbox m Request Recipient Set Register (IPCU_MBmDSR) .....	957
5.7.	IPCU Mailbox m Request Recipient Clear Register (IPCU_MBmDCR) .....	958

5.8.	IPCU Mailbox m Request Recipient Status Register (IPCU_MBmDSTR) .....	959
5.9.	IPCU Mailbox m Request Transmission Mask Set Register (IPCU_MBmMSR) ...	960
5.10.	IPCU Mailbox m Request Transmission Mask Clear Register (IPCU_MBmMCR)	962
5.11.	IPCU Mailbox m Request Transmission Mask Status Register (IPCU_MBmMSTR) .....	964
5.12.	IPCU Mailbox m Acknowledgment Set Register (IPCU_MBmASR) .....	965
5.13.	IPCU Mailbox m Acknowledgment Clear Register (IPCU_MBmACR) .....	967
5.14.	IPCU Mailbox m Acknowledgment Status Register (IPCU_MBmASTR) .....	969
5.15.	IPCU Mailbox m Acknowledgment Sender Status Register (IPCU_MBmASRCR)	970
5.16.	IPCU Mailbox m Data Registers 0 to 8 (IPCU_MBmDR0 to 8) .....	971
5.17.	IPCU Mailbox Status Register (IPCU_MBSTR) .....	972
6.	Other .....	973
CHAPTER 26:	Exclusive Access Memory (EAM) .....	975
1.	Overview .....	976
2.	Configuration .....	977
2.1.	Block Diagram .....	977
3.	Operation .....	978
3.1.	General Aspects of Transfer .....	979
3.2.	Normal Access .....	980
3.3.	Exclusive Access .....	981
4.	Operation Examples .....	987
5.	Memory Area .....	994
5.1.	Exclusive Access Memory .....	995
CHAPTER 27:	I/O Port .....	997
1.	Overview .....	998
2.	Configuration .....	999
3.	Setting Procedure Examples .....	1000
4.	Registers .....	1007
4.1.	Data Direction Register (GPIO_DDRI) (i=0 to 4) .....	1008
4.2.	Data Direction Set Register (GPIO_DDSRI) (i=0 to 4) .....	1009
4.3.	Data Direction Clear Register (GPIO_DDCRI) (i=0 to 4) .....	1010
4.4.	Port Output Data Register (GPIO_PODRI) (i=0 to 4) .....	1011
4.5.	Port Output Set Register (GPIO_POSRI) (i=0 to 4) .....	1012
4.6.	Port Output Clear Register (GPIO_POCRi) (i=0 to 4) .....	1013
4.7.	Port Input Data Register (GPIO_PIDRI) (i=0 to 4) .....	1014
4.8.	Port Input Enable Register (GPIO_PORTEN) .....	1015
4.9.	GPIO Key Code Register (GPIO_KEYCDR) .....	1016
4.10.	Port Configuration Register (PPC_PCFGRIj) (i=0 to 4, j=00 to 31) .....	1018
4.11.	PPC Key Code Register (PPC_KEYCDR) .....	1021
4.12.	Resource Input Setting Register (RIC_RESINn) (n = 0 to 11) .....	1023
5.	Precautions for Using .....	1028
CHAPTER 28:	CR Calibration .....	1029
1.	Overview .....	1030
2.	Configuration .....	1031
3.	Operation .....	1032
3.1.	CR Clock Frequency Calculation .....	1032
3.2.	CR Clock Frequency Correction .....	1033
4.	Registers .....	1034

4.1.	Correction Unit Control Register 1 (CU_CUCR1).....	1035
4.2.	CR Clock Timer Data Register 1 (CU_CUTD1).....	1037
4.3.	Main Oscillation Timer Data Register 1 (CU_CUTR1).....	1038
4.4.	Correction Unit Control Clear Register 1 (CU_CUCRC1).....	1039
CHAPTER 29:	CRC .....	1041
1.	Overview .....	1042
2.	Configuration.....	1043
3.	Operation .....	1044
3.1.	CRC Computing Sequence .....	1045
3.2.	CRC Usage Examples.....	1046
4.	Registers.....	1050
4.1.	CRC Control Register (CRCCR) .....	1051
4.2.	Initial Value Register (CRCINIT).....	1053
4.3.	Input Data Register (CRCIN).....	1054
4.4.	CRC Register (CRCR).....	1055
CHAPTER 30:	CAN.....	1057
1.	Overview .....	1058
2.	Configuration.....	1059
3.	Operation .....	1060
3.1.	CAN Interrupt Request Batch Read .....	1061
4.	Registers.....	1062
4.1.	CAN Interrupt Request Batch Read Register (CANxx_CIRRR) .....	1063
CHAPTER 31:	CAN Controller .....	1065
1.	Overview .....	1066
2.	Configuration.....	1067
3.	Operation .....	1068
3.1.	Message Objects.....	1069
3.2.	Message Transmission.....	1071
3.3.	Message Reception.....	1073
3.4.	FIFO Buffer Function.....	1075
3.5.	Interrupt Function .....	1077
3.6.	Bit Timing.....	1078
3.7.	Test Mode.....	1080
3.8.	Software Initialization .....	1084
3.9.	CAN Wakeup Function .....	1085
4.	Registers.....	1086
4.1.	Total Control Registers .....	1087
4.2.	Message Interface Registers.....	1099
4.3.	Message Object .....	1111
4.4.	Message Handler Registers .....	1116
CHAPTER 32:	CAN Message RAM ECC.....	1129
1.	Overview .....	1130
2.	Configuration.....	1131
3.	Interrupt.....	1133
4.	Operation .....	1134
4.1.	Message RAM ECC Generation.....	1135
4.2.	Message RAM ECC Test.....	1136
4.3.	ECC Error Insertion Function of the Message RAM .....	1138
5.	ECC Error Processing .....	1139
6.	Registers.....	1140
6.1.	CAN ECC Error Control Register (CANEECR).....	1141

6.2.	CAN ECC Error Status Register (CANEESR) .....	1144
6.3.	CAN ECC Error Status Clear Register (CANEESCR) .....	1145
6.4.	CAN ECC Double-bit Error Address Register (CANDEEAR).....	1146
6.5.	CAN ECC Single-bit Error Address Register (CANSEEAR) .....	1147
6.6.	CAN ECC Error Insertion Control Register (CANEFECR).....	1148
7.	Precautions for Using .....	1150
CHAPTER 33:	CAN Prescaler.....	1151
1.	Overview .....	1152
2.	Configuration .....	1153
3.	Operation .....	1154
4.	Registers .....	1155
4.1.	CAN Prescaler Control Register (CANP_CANPRE).....	1156
4.2.	CAN PLL Clock Control Register (CANP_CANPCK).....	1157
CHAPTER 34:	Multi-function Serial Interface .....	1159
1.	Overview .....	1160
2.	Configuration .....	1161
CHAPTER 35:	UART (Asynchronous Serial Interface).....	1165
1.	Overview .....	1166
2.	Interrupts .....	1167
2.1.	Occurrence of Reception Interrupts and Flag Set Timing .....	1169
2.2.	Occurrence of Interrupts and Flag Set Timing When the Reception FIFO is Used .....	1171
2.3.	Occurrence of Transmission Interrupts and Flag Set Timing .....	1173
2.4.	Occurrence of Interrupts and Flag Set Timing When the Transmission FIFO is Used .....	1175
2.5.	Timing of Timer Interrupt Generation and Flag Setting .....	1176
3.	Operation .....	1177
4.	Serial Timer Operation .....	1183
5.	Test Mode .....	1189
6.	Dedicated Baud Rate Generator .....	1190
6.1.	Setting the Baud Rate .....	1191
7.	Setup Procedure and Program Flow for Operation Mode 0 (Asynchronous Normal Mode) ..	1196
8.	Setup Procedure and Program Flow for Operation Mode 1 (Asynchronous Multi-processor Mode) .....	1198
9.	Registers .....	1202
9.1.	Serial Control Register (SCR) .....	1204
9.2.	Serial Mode Register (SMR).....	1207
9.3.	Serial Status Register (SSR).....	1209
9.4.	Extended Communication Control Register (ESCR) .....	1212
9.5.	Reception Data Register/Transmission Data Register (RDR/TDR) .....	1214
9.6.	Serial Auxiliary Control Status Register (SACSR) .....	1218
9.7.	Serial Timer Register (STMR) .....	1221
9.8.	Serial Timer Comparison Register (STMCR).....	1222
9.9.	Transfer Byte Register (TBYTE0).....	1223
9.10.	Baud Rate Generator Register 1, 0 (BGR1, BGR0) .....	1224
9.11.	FIFO Control Register 1 (FCR1).....	1226
9.12.	FIFO Control Register 0 (FCR0).....	1229
9.13.	FIFO Byte Register (FBYTE).....	1233
9.14.	Transmission FIFO Interrupt Control Register (FTICR) .....	1235
9.15.	Serial Auxiliary Control Status Clear Register (SACSRC) .....	1237

9.16.	FIFO Control Clear Register 1 (FCR1C).....	1239
9.17.	FIFO Control Clear Register 0 (FCR0C).....	1240
9.18.	Serial Auxiliary Control Status Set Register (SACSRS) .....	1241
9.19.	FIFO Control Set Register 1 (FCR1S).....	1242
9.20.	FIFO Control Set Register 0 (FCR0S).....	1243
10.	Precautions for Using .....	1244
CHAPTER 36:	CSIO (Clock Synchronous Serial Interface) .....	1245
1.	Overview .....	1246
2.	Interrupts of the CSIO (Clock Synchronous Serial Interface) .....	1247
2.1.	Reception Interrupt Generation and Flag Set Timing .....	1250
2.2.	Interrupt Generation and Flag Set Timing When the Reception FIFO is Used .....	1252
2.3.	Transmission Interrupt Generation and Flag Set Timing .....	1254
2.4.	Interrupt Generation and Flag Set Timing When the Transmission FIFO is Used .....	1256
2.5.	Timer Interrupt Generation and Flag Set Timing.....	1257
2.6.	Chip Select Error Occurrence and Flag Set Timing .....	1258
3.	Operation .....	1260
3.1.	Normal Transfer (I) .....	1260
3.2.	Normal Transfer (II) .....	1270
3.3.	SPI Transfer (I) .....	1280
3.4.	SPI Transfer (II) .....	1290
4.	Serial Timer Operation .....	1300
5.	Operation of Serial Chip Select .....	1306
6.	Test Mode .....	1318
7.	Dedicated Baud Rate Generator .....	1319
7.1.	Setting the Baud Rate .....	1320
7.2.	Setup Procedure and Program Flow for the CSIO (Clock Synchronous Serial Interface) .....	1324
8.	Registers .....	1326
8.1.	Serial Control Register (SCR) .....	1327
8.2.	Serial Mode Register (SMR).....	1331
8.3.	Serial Status Register (SSR).....	1334
8.4.	Extended Communication Control Register (ESCR) .....	1337
8.5.	Reception Data Register/Transmission Data Register (RDR/TDR) .....	1340
8.6.	Serial Auxiliary Control Status Register (SACSR) .....	1345
8.7.	Serial Timer Register (STMR) .....	1349
8.8.	Serial Timer Comparison Register (STMCR).....	1350
8.9.	Serial Chip Select Control Status Register (SCSCR) .....	1351
8.10.	Serial Chip Select Timing Registers (SCSTR3 to SCSTR0).....	1356
8.11.	Serial Chip Select Format Registers (SCSFR2 to SCSFR0) .....	1359
8.12.	Transfer Byte Registers (TBYTE3 to TBYTE0).....	1369
8.13.	Baud Rate Generator Registers 1, 0 (BGR1, BGR0) .....	1371
8.14.	FIFO Control Register 1 (FCR1).....	1372
8.15.	FIFO Control Register 0 (FCR0).....	1375
8.16.	FIFO Byte Register (FBYTE).....	1379
8.17.	Transmission FIFO Interrupt Control Register (FTICR) .....	1381
8.18.	Serial Auxiliary Control Status Clear Register (SACSRC) .....	1383
8.19.	FIFO Control Clear Register 1 (FCR1C).....	1385
8.20.	FIFO Control Clear Register 0 (FCR0C).....	1386
8.21.	Serial Auxiliary Control Status Set Register (SACSRS) .....	1387



8.22.	FIFO Control Set Register 1 (FCR1S).....	1389
8.23.	FIFO Control Set Register 0 (FCR0S).....	1390
9.	Precautions for Using .....	1391
CHAPTER 37:	LIN Interface (v2.1) (LIN Communication Control Interface (v2.1)) .....	1393
1.	Overview .....	1394
1.1.	Manual Mode.....	1394
1.2.	Assist Mode.....	1395
2.	Interrupts.....	1397
2.1.	Manual Mode.....	1397
2.2.	Assist Mode.....	1408
3.	Serial Timer Operation .....	1423
4.	Test Mode .....	1428
4.1.	Manual Mode.....	1428
4.2.	Assist Mode.....	1429
5.	Dedicated Baud Rate Generator .....	1434
5.1.	Setting Baud Rate .....	1435
6.	Operation .....	1440
6.1.	Manual Mode.....	1440
6.2.	Assist Mode.....	1453
7.	Setup Procedure and Program Flow for Operation Mode 3 (LIN Communication Mode).....	1472
7.1.	Manual Mode.....	1473
7.2.	Assist Mode.....	1477
8.	Registers.....	1482
8.1.	Serial Control Register (SCR) .....	1484
8.2.	Serial Mode Register (SMR).....	1488
8.3.	Serial Status Register (SSR).....	1490
8.4.	Extended Communication Control Register (ESCR) .....	1494
8.5.	Reception Data Register/Transmission Data Register (RDR/TDR) .....	1496
8.6.	Serial Auxiliary Control Status Register (SACSR) .....	1498
8.7.	Serial Timer Register (STMR) .....	1502
8.8.	Serial Timer Comparison Register (STMCR).....	1503
8.9.	Sync Field Upper Limit Register (SFUR) .....	1504
8.10.	Sync Field Lower Limit Register (SFLR).....	1505
8.11.	Baud Rate Generator Register 1, 0 (BGR1 to 0) .....	1506
8.12.	LIN Assist Mode Status Register (LAMSR) .....	1508
8.13.	LIN Assist Mode Control Register (LAMCR).....	1511
8.14.	LIN Assist Mode Interrupt Enable Register (LAMIER) .....	1514
8.15.	LIN Assist Mode Transmission/Reception ID Register (LAMTID/LAMRID).....	1516
8.16.	LIN Assist Mode Error Status Register (LAMESR) .....	1518
8.17.	LIN Assist Mode Error Test Register (LAMERT) .....	1521
8.18.	FIFO Control Register 1 (FCR1).....	1524
8.19.	FIFO Control Register 0 (FCR0).....	1527
8.20.	FIFO Byte Register (FBYTE).....	1532
8.21.	Transmission FIFO Interrupt Control Register (FTICR) .....	1534
8.22.	Serial Control Clear Register (SCRC) .....	1536
8.23.	Serial Mode Clear Register (SMRC) .....	1538
8.24.	Serial Status Clear Register (SSRC).....	1539
8.25.	Extended Communication Control Clear Register (ESCRC) .....	1540
8.26.	Serial Auxiliary Control Status Clear Register (SACSRC) .....	1541
8.27.	LIN Assist Mode Status Clear Register (LAMSRC) .....	1543
8.28.	LIN Assist Mode Control Clear Register (LAMCRC).....	1544



8.29.	LIN Assist Mode Interrupt Enable Clear Register (LAMIERC) .....	1545
8.30.	LIN Assist Mode Error Status Clear Register (LAMESRC) .....	1547
8.31.	FIFO Control Clear Register 1 (FCR1C).....	1548
8.32.	FIFO Control Clear Register 0 (FCR0C).....	1549
8.33.	Serial Control Set Register (SCRS).....	1550
8.34.	Serial Mode Set Register (SMRS).....	1552
8.35.	Serial Status Set Register (SSRS) .....	1553
8.36.	Extended Communication Control Set Register (ESCRS).....	1554
8.37.	Serial Auxiliary Control Status Set Register (SACRS) .....	1555
8.38.	LIN Assist Mode Control Set Register (LAMCRS) .....	1557
8.39.	LIN Assist Mode Interrupt Enable Set Register (LAMIERS) .....	1558
8.40.	FIFO Control Set Register 1 (FCR1S).....	1560
8.41.	FIFO Control Set Register 0 (FCR0S).....	1561
9.	Precautions for Using .....	1562
CHAPTER 38:	Base Timer .....	1563
1.	Overview .....	1564
2.	Configuration .....	1566
3.	Operation .....	1569
4.	32-bit Mode Operation.....	1571
5.	Interrupts .....	1573
6.	Start of DMA Controller (DMAC) .....	1574
7.	Registers .....	1575
8.	Notes on Using.....	1577
9.	Description by Function Mode.....	1579
9.1.	PWM Timer Function.....	1580
9.2.	PPG Timer Function .....	1599
9.3.	Reload Timer Function .....	1618
9.4.	PWC Timer Function .....	1634
CHAPTER 39:	Base Timer I/O Selection Function.....	1653
1.	Overview .....	1654
2.	Configuration.....	1655
3.	Operation .....	1656
4.	Registers.....	1659
4.1.	I/O Selection Registers (BT_BTSEL01, BT_BTSEL23, BT_BTSEL45, BT_BTSEL67, BT_BTSEL_89, BT_BTSEL1011) .....	1660
4.2.	Simultaneous Soft Start Register (BT_BTSSSR) .....	1661
CHAPTER 40:	32-bit Free-run Timer .....	1663
1.	Overview .....	1664
2.	Configuration.....	1665
3.	Operation .....	1666
3.1.	Interrupts of the 32-bit Free-run Timer.....	1672
4.	Registers.....	1673
4.1.	Compare Clear Buffer Register (CPCLRB)/ Compare Clear Register (CPCLR) ..	1674
4.2.	Timer Data Register (TCDT).....	1676
4.3.	Timer State Control Register (TCCS) .....	1677
4.4.	Timer Extended State Control Register (TECCS).....	1682
4.5.	Timer State Control Clear Register (TCCSC).....	1684
4.6.	Timer State Control Set Register (TCCSS) .....	1686
5.	Precautions for Using .....	1688
CHAPTER 41:	32-bit Input Capture.....	1689

1.	Overview .....	1690
2.	Configuration .....	1691
3.	Operation .....	1692
3.1.	Interrupt .....	1694
3.2.	Setting Procedure Example .....	1695
4.	Registers .....	1696
4.1.	Input Capture Data Registers 0, 1 (IPCP0, IPCP1) .....	1697
4.2.	Input Capture State Control Register (ICS) .....	1699
4.3.	Input Capture State Control Clear Register (ICSC) .....	1702
4.4.	Input Capture State Control Set Register (ICSS) .....	1704
5.	Precautions for Using .....	1705
CHAPTER 42:	FlexRay/RDC Dedicated Clock .....	1707
1.	Overview .....	1708
2.	Configuration .....	1709
3.	Operation .....	1710
4.	Setting Procedure Example .....	1711
5.	Registers .....	1714
5.1.	FlexRay/RDC PLL CSV Control Register (ERAYP_CSVR) .....	1715
5.2.	FlexRay/RDC PLL Multiplication Rate (M Division) Selection Register (ERAYP_PLL2DIVM) .....	1716
5.3.	FlexRay/RDC PLL Multiplication Rate (N Division) Selection Register (ERAYP_PLL2DIVN) .....	1717
5.4.	FlexRay/RDC PLL Automatic Gear Multiplication Rate (G Division) Selection Register (ERAYP_PLL2DIVG) .....	1718
5.5.	FlexRay/RDC PLL Step Multiplication Rate (G division) Selection Register (ERAYP_PLL2MULG) .....	1719
5.6.	Automatic Gear Control Register (ERAYP_PLL2CTRL) .....	1720
5.7.	FlexRay/RDC PLL Multiplication Rate (K Division) Selection Register (ERAYP_PLL2DIVK) .....	1721
5.8.	FlexRay/RDC PLL Clock Output Control Register (ERAYP_CLKR2) .....	1722
5.9.	Automatic Gear Control Flag Register (ERAYP_PLL2CTRLF) .....	1724
5.10.	FlexRay/RDC PLL Clock Output Control Flag Register (ERAYP_CLKR2F) .....	1725
5.11.	Automatic Gear Control Clear Register (ERAYP_PLL2CTRLC) .....	1726
5.12.	FlexRay/RDC PLL Clock Output Control Clear Register (ERAYP_CLKR2C) .....	1727
6.	Precautions for Using .....	1728
CHAPTER 43:	Clock Monitor .....	1729
1.	Overview .....	1730
2.	Configuration .....	1731
3.	Operation .....	1732
4.	Registers .....	1733
4.1.	Clock Monitor Control Register (CLKMN_CMCFG) .....	1734
4.2.	Clock Control Register (CLKMN_CSCFG) .....	1736
5.	Notes .....	1737
CHAPTER 44:	FlexRay Controller .....	1739
1.	Overview .....	1740
2.	Configuration .....	1741
3.	Operation .....	1744
3.1.	Communication Cycle .....	1744
3.2.	Communication Modes .....	1747
3.3.	Clock Synchronization .....	1748
3.4.	Error Handling .....	1750

3.5.	Communication Controller States .....	1752
3.6.	Network Management .....	1766
3.7.	Filtering and Masking .....	1767
3.8.	Transmission Procedure.....	1770
3.9.	Reception Procedure.....	1772
3.10.	FIFO Function .....	1774
3.11.	Message Handling.....	1776
3.12.	Message RAM.....	1784
3.13.	Interrupts .....	1791
4.	Registers .....	1793
4.1.	Customer Registers.....	1798
4.2.	Special Register .....	1809
4.3.	Interrupt-related Registers.....	1810
4.4.	Communication Controller (CC) Control Registers.....	1842
4.5.	Communication Controller (CC) Status Registers .....	1872
4.6.	Message Buffer Control Registers.....	1895
4.7.	Message Buffer Status Registers .....	1902
4.8.	Identification Registers .....	1917
4.9.	Input Buffer .....	1919
4.10.	Output Buffer .....	1929
CHAPTER 45:	Up/Down Counter.....	1945
1.	Overview .....	1946
2.	Configuration .....	1948
3.	Operation .....	1949
4.	Registers .....	1961
4.1.	Up/Down Count Register (UDCR) .....	1962
4.2.	Reload Compare Register (RCR).....	1963
4.3.	Counter Status Register (CSRL) .....	1964
4.4.	Counter Control Register (CCR).....	1966
4.5.	Counter Compare Register (CMPR0 to 5).....	1970
4.6.	Counter Compare Buffer Transfer Register (CMPBR0 to 5).....	1971
4.7.	Counter Compare Mask Register (CMPMSKR0 to 5) .....	1972
4.8.	Comparison Result Match Detection Flag Register (CMPFR).....	1973
4.9.	Comparison Result Match Interrupt Enable Flag Register (CITER) .....	1974
4.10.	Buffer Transfer Setting Register (CBTR) .....	1975
4.11.	Counter Control Set Register (CCSR).....	1976
4.12.	Counter Control Clear Register (CCCR).....	1978
4.13.	Counter Status Set Register (CSSRL) .....	1980
4.14.	Counter Status Clear Register (CSCRL) .....	1981
4.15.	Comparison Result Match Detection Flag Clear Register (CMPFCR).....	1983
5.	Precautions for Using This Device .....	1984
CHAPTER 46:	16-bit Free-run Timer .....	1985
1.	Overview .....	1986
2.	Configuration .....	1987
3.	Operation .....	1988
3.1.	Interrupts of the 16-bit Free-run Timer.....	1995
4.	Registers .....	1996
4.1.	Compare Clear Buffer Register (CPCLRB) / Compare Clear Register (CPCLR) . .....	1997
4.2.	Timer Data Register (TCDT).....	1999
4.3.	Timer State Control Register (TCCS) .....	2000

4.4.	Timer State Clear Register (TCCSC) .....	2006
4.5.	Timer State Set Register (TCCSS) .....	2008
5.	Notes .....	2010
CHAPTER 47:	Free-run Timer Selection and Simultaneous Activation .....	2011
1.	Overview .....	2012
2.	Configuration .....	2013
3.	Operation .....	2017
4.	Setting Procedure Example .....	2021
5.	Registers .....	2023
5.1.	Register for Free-run Timer Selection .....	2024
5.2.	Register for Timer Simultaneous Activation .....	2034
5.3.	Register for Indicating Free-run Timer Count Direction .....	2039
CHAPTER 48:	16-bit Input Capture .....	2041
1.	Overview .....	2042
2.	Configuration .....	2043
3.	Operation .....	2044
3.1.	Interrupt of 16-bit Input Capture .....	2046
3.2.	Setting Procedure Example .....	2047
4.	Registers .....	2048
4.1.	Input Capture Data Register 0, 1 (ICPD0, ICPD1) .....	2049
4.2.	Input Capture State Control Register (ICS) .....	2051
4.3.	Input Capture State Clear Register (ICSC) .....	2054
4.4.	Input Capture State Set Register (ICSS) .....	2056
5.	Precautions for Using .....	2057
CHAPTER 49:	16-bit Output Compare .....	2059
1.	Overview .....	2060
2.	Configuration .....	2061
3.	Operation .....	2062
3.1.	Interrupt .....	2073
4.	Registers .....	2074
4.1.	Output Compare Buffer Register 0 (OCCPB0) / Output Compare Register 0 (OCCP0) .....	2075
4.2.	Output Compare Buffer Register 1 (OCCPB1) / Output Compare Register 1 (OCCP1) .....	2077
4.3.	Compare Control Register (OCS) .....	2079
4.4.	Compare Mode Control Register (OCMOD) .....	2084
4.5.	Compare Control Clear Register (OCSC) .....	2085
4.6.	Compare Control Set Register (OCSS) .....	2088
5.	Precautions for Using .....	2091
CHAPTER 50:	12-bit A/D Converter Interface .....	2093
1.	Overview .....	2094
2.	Configuration .....	2095
3.	Operation .....	2096
3.1.	Operation of 12-bit A/D Converter Interface .....	2096
4.	Setting Procedure Example .....	2099
5.	Registers .....	2100
5.1.	A/D Control Status Register (ADCS) .....	2101
5.2.	A/D Channel Status Register (ADCH) .....	2102
5.3.	A/D Mode Setting Register (ADMD) .....	2103
5.4.	A/D Sampling Time Setting Per Channel Register m(ADSTPCSm)(m=0 to 7) .....	2105

6.	Precautions for Using .....	2106
CHAPTER 51:	12-bit A/D converter Activation Compare .....	2107
1.	Overview .....	2108
2.	Configuration .....	2111
3.	Operation .....	2112
3.1.	A/D Activation Compare Interrupts .....	2112
3.2.	A/D Activation Compare Operation .....	2114
4.	Setting Procedure Example .....	2141
4.1.	A/D Conversion Setting Procedure Examples .....	2141
4.2.	Setting Procedure Example for Scan Conversion .....	2150
5.	Registers .....	2158
5.1.	Registers of Analog Input Control .....	2159
5.2.	A/D Start Compare Register .....	2162
6.	Precautions for Using .....	2193
CHAPTER 52:	12-bit A/D Converter Arbitration .....	2197
1.	Overview .....	2198
2.	Configuration .....	2199
3.	Operation .....	2200
3.1.	A/D Activation Arbitration Operation .....	2200
CHAPTER 53:	Waveform Generator .....	2203
1.	Overview .....	2204
2.	Configuration .....	2205
3.	Operation .....	2208
3.1.	Interrupts for Waveform Generator .....	2226
4.	Registers .....	2227
4.1.	16-bit Dead Timer Register n (WFGxx_TMRRn) (n=0 to 2) .....	2228
4.2.	16-bit Dead Timer State Control Register n (WFGxx_DTICRn) (n=0 to 2) ....	2230
4.3.	16-bit Dead Timer Reload Interrupt Register (WFGxx_DTIR) .....	2239
4.4.	16-bit Dead Timer Minus Control Register (WFGxx_DTMNS) .....	2241
4.5.	Waveform Control Register 1/2 (WFGxx_SIGCR1, WFGxx_SIGCR2) .....	2243
4.6.	PPG Output Control Register (WFGxx_PICS) .....	2247
4.7.	16-bit Dead Timer State Control Clear Register (WFGxx_DTCRCn) (n=0 to 2) ...	2249
4.8.	16-bit Dead Timer Reload Interrupt Clear Register (WFGxx_DTIRC) .....	2252
4.9.	Waveform Control Clear Register 1 (WFGxx_SIGCR1C) .....	2253
4.10.	16-bit Dead Timer State Control Set Register (WFGxx_DTCRSn) (n=0 to 2) ....	2254
4.11.	16-bit Dead Timer Reload Interrupt Set Register (WFGxx_DTIRS) .....	2257
4.12.	Waveform Control Set Register 1 (WFGxx_SIGCR1S) .....	2258
4.13.	DTTI Selection Register (WFG02_DTSR, WFG03_DTSR) .....	2259
4.14.	Software DTTI Control Register (SDTCR2) .....	2261
4.15.	External DTTI Input Control Register (EDTCR2) .....	2262
4.16.	RTO Output Level Conversion Register (RTOSELn) (n=0 to 3) .....	2263
5.	Precautions for Using .....	2265
CHAPTER 54:	R/D Converter .....	2267
1.	Overview .....	2268
CHAPTER 55:	D/A Converter .....	2269
1.	Overview .....	2270
2.	Configuration .....	2271
3.	Operation .....	2272
4.	Setting Procedure Example .....	2273

5.	Registers .....	2274
5.1.	D/A Control Register (DACxx_DACR) .....	2275
5.2.	D/A Data Register (DACxx_DADR) .....	2276
5.3.	Analog Output Control Register (DACxx_DAER) .....	2277
5.4.	Key Code Register (DACxx_KEYCDR) .....	2278
6.	Precautions for Using .....	2280
CHAPTER 56:	12-bit 4ch A/D Converter Interface .....	2281
1.	Overview .....	2282
2.	Configuration .....	2283
3.	Operation .....	2284
3.1.	12-bit 4ch A/D Converter Interface Operation .....	2284
4.	Setting Procedure Example .....	2291
5.	Registers .....	2292
5.1.	12-bit 4ch A/D Converter Interface Registers .....	2293
6.	Precautions for Using .....	2302
CHAPTER 57:	12-bit 4ch A/D Converter A/D Activation Compare .....	2303
1.	Overview .....	2304
2.	Configuration .....	2306
3.	Operation .....	2307
3.1.	A/D Activation Compare Interrupts .....	2307
3.2.	A/D Activation Compare Operation (n=0 to 7) .....	2309
3.3.	Operation Timing Example .....	2327
4.	Setting Procedure Examples .....	2329
4.1.	A/D Conversion Setting Procedure Examples .....	2329
5.	Registers .....	2337
5.1.	A/D Activation Compare Registers .....	2338
6.	Precautions for Using .....	2360
CHAPTER 58:	12-bit 4ch A/D Converter Arbitration .....	2365
1.	Overview .....	2366
2.	Configuration .....	2367
3.	Operation .....	2368
3.1.	Operation of 12-bit 4ch A/D Arbitration .....	2368
CHAPTER 59:	Motor Vector Operation Accelerator .....	2375
1.	Overview .....	2376
CHAPTER 60:	Appendix .....	2379
1.	I/O Port Settings .....	2380
1.1.	Input Level Setting .....	2380
1.2.	Output Drive Setting .....	2380
1.3.	Output Resource Selection .....	2381
1.4.	Resource Input Selection .....	2385
2.	Registers Map .....	2387
3.	List of Interrupt Factors and DMA Activation Factors .....	2445
3.1.	List of Interrupt Factors .....	2445
3.2.	List of NMI Factors .....	2453
3.3.	List of DMA Activation Factors .....	2454
4.	Master Access .....	2457
5.	Low-power Mode .....	2458
6.	Major Changes .....	2459



## H A R D W A R E   M A N U A L

---



# CHAPTER 1: Overview

This chapter explains overview of MB9D560 series.

---

1. Overview
2. Product Lineup
3. Configuration
4. Pin Assignment
5. Pin Description
6. I/O Circuit Type
7. Memory Map
8. I/O Map
9. Legend
10. Abbreviations





## 1. Overview

This section gives a brief overview of MB9D560 series.

**Table 1-1 Function Overview**

Function	Description
Technology	<ul style="list-style-type: none"> <li>- CMOS 90nm technology</li> </ul>
CPU	<ul style="list-style-type: none"> <li>- ARM Cortex™-R5F</li> <li>- 32-bit ARM architecture</li> <li>- 2-instruction issuance super scalar</li> <li>- 8-stage pipeline</li> <li>- ARMv7 / Thumb@-2 instruction set</li> <li>- Floating-Point Unit (FPU) Double precision</li> <li>- Memory protection Unit (MPU) 16 area</li> <li>- ECC support for the TCM port 1-bit error correction, 2-bit error detection ECC (SEC-DED)</li> <li>- TCM port 2 TCM ports</li> <li>- ATCM port</li> <li>- BTCM 2ports (B0TCM, B1TCM)</li> <li>- VIC port Low latency interrupt</li> <li>- AXI master interface 64-bit AXI interface (instruction / data access) 32-bit AXI interface (I/O access)</li> <li>- AXI slave interface 64-bit AXI interface (accessible to TCM port)</li> <li>- CPU configuration 2CPUs (AMP operation)</li> <li>- Operating frequency Maximum 200MHz</li> <li>- Trace with ETM-R5</li> </ul>
Debugging	<ul style="list-style-type: none"> <li>- ARM CoreSight™ Technology Each CPU embedded Embedded Trace Macro (ETM), trace support of CPU operation</li> <li>- Debugging interface JTAG (5pin ) Support clock : maximum 20MHz</li> <li>- Debugging security support 128-bit security key (Device security key)</li> <li>- Wakeup function on JTAG</li> </ul>
Operation mode	<ul style="list-style-type: none"> <li>- User mode Normal mode (internal memory activation)</li> <li>- Serial writer mode</li> </ul>

Function	Description
Clock control	<ul style="list-style-type: none"> <li>- Internal clock source Fast-CR oscillation (8MHz) Slow-CR oscillation (100kHz)</li> <li>- External oscillation input Main clock input</li> <li>- Embedded PLL Main PLL (Multiplying clock of main oscillation )</li> <li>- Oscillator stabilized timer Support oscillator stabilized timer for all clock source independently After a lapse of oscillator stabilized time, it is able to use source clock timer (Except PLL for FlexRay/RDC)</li> </ul>
Reset control	<ul style="list-style-type: none"> <li>- Reset level Hardware reset (system initialization) Software reset (programming initialization)</li> <li>- Reset factor (Hardware reset ) Power-on reset (PONR), external reset input (RSTX, NMIX+RSTX), clock stop waiting with time-out reset, low-voltage detection reset (internal low-voltage detection reset, 5V external low-voltage detection reset ), watchdog reset (hardware watchdog reset, software watchdog reset ), clock supervisor reset (main clock monitor, PLL clock monitor), software trigger hardware reset , profile error reset</li> <li>- Reset factor (software reset ) Software reset</li> </ul>
Low-power consumption control	<ul style="list-style-type: none"> <li>- Device state RUN (Run State, CPU is operation status) PSS (Power Saving State, CPU wait event from WFI)</li> <li>- Setting parameter of each device state Clock (clock source enable, clock source selection, clock divider, clock domain enable) Clock monitor Low-voltage detection</li> </ul>
Memory protection unit (MPU)	<ul style="list-style-type: none"> <li>- Memory protection as master except processor</li> <li>- Target master DMA controller</li> <li>- 8 area</li> <li>- NMI generation when violation detection</li> </ul>
Timing protection unit (TPU)	<ul style="list-style-type: none"> <li>- TPU 1 unit as CPU 1 unit</li> <li>- 24-bit timer x 8 channels per unit</li> <li>- Support execution time protection, locking time protection, inter-arrival time protection, deadline protection</li> <li>- Support normal mode and over flow mode</li> <li>- Prescaler of each channels Timer clock divider (1/1 to 1/64)</li> <li>- Independent prescaler of each channels Timer clock divider (1/1, 1/2, 1/4, 1/16)</li> </ul>
Clock supervisor (CSV)	<ul style="list-style-type: none"> <li>- Monitor target clock Main oscillation input , main PLL output</li> <li>- Monitor method Monitor of frequency range</li> <li>- Operation after error detection Reset or NMI</li> </ul>



Function	Description
Watchdog timer (WDT)	<ul style="list-style-type: none"> <li>- Watchdog timer embedded <ul style="list-style-type: none"> <li>Hardware watchdog timer</li> <li>Software watchdog timer</li> </ul> </li> <li>- Hardware watchdog timer <ul style="list-style-type: none"> <li>1 unit per system</li> <li>32-bit watchdog timer with window function</li> <li>Clock source: fast-CR or slow-CR</li> <li>Set by boot program (BootROM maker)</li> <li>Not set by user program</li> </ul> </li> <li>- Software watchdog timer <ul style="list-style-type: none"> <li>1 unit per CPU</li> <li>32-bit watchdog timer with window function</li> <li>Clock source: fast-CR, slow-CR, main clock</li> <li>One time set on user program (not set again)</li> </ul> </li> </ul>
Low-voltage detection (LVD)	<ul style="list-style-type: none"> <li>- Select voltage monitor <ul style="list-style-type: none"> <li>External low-voltage detection (5V power line monitor): 3.9V, 4.1V, 4.3V</li> <li>Internal low-voltage detection (1.2V power line monitor): 0.9V</li> </ul> </li> <li>- Internal low-voltage detection: always valid</li> <li>- External low-voltage detection: valid/invalid set</li> <li>- External low-voltage detection: set threshold voltage independently on RUN / PSS</li> <li>- Output when low-voltage detection <ul style="list-style-type: none"> <li>External low-voltage detection: reset or NMI</li> <li>Internal low-voltage detection: reset</li> </ul> </li> </ul>
Main Flash memory (TCFLASH)	<ul style="list-style-type: none"> <li>- Cortex-R5F ATCM connection <ul style="list-style-type: none"> <li>1 Main Flash memory as CPU 1 unit</li> </ul> </li> <li>- HPM connection with 64-bit AXI</li> <li>- Flash memory configuration <ul style="list-style-type: none"> <li>Interleave with 64-bit Flash 2 units</li> </ul> </li> <li>- 2 address areas <ul style="list-style-type: none"> <li>TCM (read only)</li> <li>AXI (read / write)</li> </ul> </li> <li>- ECC support (SEC-DED)</li> <li>- Parallel programming support</li> <li>- Flash security</li> </ul>
Work Flash memory (WorkFLASH)	<ul style="list-style-type: none"> <li>- 2 Work Flash memories <ul style="list-style-type: none"> <li>1 Work Flash memory as CPU 1 unit</li> </ul> </li> <li>- (8K byte x 8) x 2 units (128K byte)</li> <li>- ECC support (SEC-DED)</li> <li>- Parallel programming support</li> <li>- Flash security</li> </ul>
Main SRAM (TCRAM)	<ul style="list-style-type: none"> <li>- BTCM connection of Cortex-R5F <ul style="list-style-type: none"> <li>1 main SRAM as CPU 1 unit</li> <li>Interleave with 2 ports of B0TCM and B1TCM</li> </ul> </li> <li>- ECC support (SEC-DED)</li> </ul>
BootROM	<ul style="list-style-type: none"> <li>- Size: 16K byte</li> <li>- Boot operation support</li> <li>- Serial writer program support</li> </ul>

Function	Description
DMA controller (DMAC)	<ul style="list-style-type: none"> <li>- 16 channels</li> <li>- Transfer mode Block transfer, Burst transfer</li> <li>- Addressing mode Fixed, increment</li> <li>- Priority between channels Fixed, Dynamic, Round robin</li> </ul>
Interrupt control (IRC)	<ul style="list-style-type: none"> <li>- Support normal interrupt (IRQ) and non-maskable interrupt (NMI)</li> <li>- Normal interrupt (IRQ) Use Interrupt Request (IRQ) of Cortex-R5F 512 channels 32 level for priority</li> <li>- Support low latency interrupt response from VIC port of Cortex-R5F</li> <li>- Non-maskable interrupt (NMI) Use fast interrupt request (FIQ) of Cortex-R5F 32 channels 16 level for priority</li> <li>- Support software interrupt generation</li> </ul>
External interrupt (EXT-IRQ)	<ul style="list-style-type: none"> <li>- Input Normal interrupt (IRQ): 8 input Non-maskable interrupt (NMI): 1 input</li> <li>- Detection method H level , L level , rise edge, fall edge, both edge</li> </ul>
Inter-processor communications unit (IPCU)	<ul style="list-style-type: none"> <li>- Mailbox function Data communication for CPU core communication by 8 Mailbox Support of interrupt between CPU core</li> </ul>
Exclusion access memory (EAM)	<ul style="list-style-type: none"> <li>- Small size memory to support exclusion control on exclusion access instruction</li> <li>- Use for semaphore</li> <li>- Size: 48 byte</li> </ul>
Bit-band function	<ul style="list-style-type: none"> <li>- The bit operation of specified register bit on Bit band area, it is mapping 1bit of bit band area to support bit band alias area for 1 byte. The target of bit band access is specified register bit on I/O area</li> </ul>
CRC	<ul style="list-style-type: none"> <li>- Output to register of CRC code according real time writing to input register</li> </ul>
Base timer	<ul style="list-style-type: none"> <li>- 16-bit timer Any of four PWM/PPG/reload/PWC timer functions can be selected and used. A 32-bit timer can be used in 2 channels of cascade mode as reload/PWC timer.</li> </ul>
16-bit free-run timer	<ul style="list-style-type: none"> <li>- 16bit up/down counter (2 channels for motor control only)</li> </ul>
32-bit free-run timer	<ul style="list-style-type: none"> <li>- 32bit up/down counter</li> </ul>
16-bit input capture	<ul style="list-style-type: none"> <li>- Input capture 16-bit capture register that detects rise edge, Fall edge, both edge Generate interrupt request after latch of counter number of 16 bit Free-run timer with edge detection of pin input</li> </ul>
32-bit input capture	<ul style="list-style-type: none"> <li>- Input capture 32-bit capture register that detects rise edge, fall edge, both edge Generate interrupt request after latch of counter number of 32 bit Free-run timer with edge detection of pin input</li> <li>- LIN sync break/sync field relation is following. Input capture ch.0 → Multi-function serial interface ch.0 Input capture ch.1 → Multi-function serial interface ch.1 Input capture ch.2 → Multi-function serial interface ch.2 Input capture ch.3 → Multi-function serial interface ch.3 Input capture ch.4 → Multi-function serial interface ch.4</li> </ul>



Function	Description
16-bit output compare	<ul style="list-style-type: none"> <li>- Output interrupt signal when compare with 16-bit free-run timer</li> </ul>
Waveform generator (WFG)	<ul style="list-style-type: none"> <li>- Generate variable output</li> <li>Real time output</li> <li>16-bit PPG waveform output</li> <li>PPG uses 16-bit PPG timer of base timer</li> <li>The relation is following</li> <li>WFG(ch.0 to ch.5)</li> <li>Base timer ch.0 → PPG0</li> <li>Base timer ch.2 → PPG2</li> <li>Base timer ch.4 → PPG4</li> <li>WFG(ch.6 to ch.11)</li> <li>Base timer ch.6 → PPG6</li> <li>Base timer ch.8 → PPG8</li> <li>Base timer ch.10 → PPG10</li> <li>Non overlap three-phase waveform output (inverter control)</li> <li>DC chopper waveform output</li> <li>- Dead time timer function</li> <li>- GATE function</li> <li>- DTTI function</li> </ul>
A/D converter (ADC)	<ul style="list-style-type: none"> <li>- 12-bit resolution A/D converter: 1 unit (32 channels)</li> <li>- Sampling analog value from input port of 32 channels</li> <li>- Conversion time: 1 μs</li> <li>- External trigger activation (ADTG)</li> <li>- Activation from internal timer (base timer)</li> </ul>
4ch sample-hold A/D converter	<ul style="list-style-type: none"> <li>- 12 bit resolution A/D converter: 2 units (8 channels )</li> </ul>
Multi-function serial interface (MFS)	<ul style="list-style-type: none"> <li>- UART / CSIO / LIN interface (v2.1) communication support</li> <li>- Transmission FIFO: 64 byte, reception FIFO: 64 Byte</li> <li>- Reception interrupt factor</li> <li>Reception error detection (parity, over run, frame error)</li> <li>Reception to FIFO for data of setting value</li> <li>Reception data under setting value in FIFO, idle term detection of over 8 clocks with baud rate clock</li> <li>- Transmission interrupt element</li> <li>No transmission operation</li> <li>Transmission FIFO empty (contain transmission operation)</li> <li>- SPI (serial peripheral interface) support</li> <li>- LIN protocol revision 2.1 support</li> </ul>
Up/down counter (UDC)	<ul style="list-style-type: none"> <li>- 8/16-bit Up/down counter (2 channels uses for R/D converter)</li> </ul>
CAN interface	<ul style="list-style-type: none"> <li>- The CAN is based on the CAN protocol ver. 2.0A/B</li> <li>- 64 message buffers x 3 channels</li> <li>- An identification mask is applied to each message object</li> <li>- Up to 1Mbps support</li> <li>- Clock support CAN prescaler</li> <li>- CAN wakeup functions</li> </ul>
FlexRay controller	<ul style="list-style-type: none"> <li>- Supports FlexRay protocol specification v2.1</li> <li>- Maximum 128 message buffers</li> <li>- 8K Byte message RAM</li> <li>- Variable length of message buffers</li> <li>- Each message buffer can be allocated as a part of reception buffer, transmission buffer or reception FIFO</li> <li>- Host access to the message buffer via input and output buffers</li> <li>- Filtering for slot counter, cycle counter and channels</li> <li>- Maskable interrupts are supported</li> </ul>

Function	Description
R/D converter (RDC)	- Connect to resolver interface
D/A converter (DAC)	- 10-bit resolution
Motor vector operation accelerator (MVA)	<ul style="list-style-type: none"> <li>- Assist for three-phase current normalizing, three-phase to two-phase DC conversion / two-phase to three-phase AC conversion, angler calculation, PID control calculation.</li> <li>- Error detection in processing (overflow/under flow/non normalizing error of FLOP)</li> <li>- Amplitude diagnosis /angle diagnosis function of R/D converter</li> <li>- Error current diagnosis function</li> </ul>
Key code	<ul style="list-style-type: none"> <li>- Key code supports</li> </ul> <p>A part of General-purpose I/O (GPIO) register  Port pin configuration (PPC) register  Analog input control register (ADER)  4ch ADC analog input control register (ADER4CH_1, ADER4CH_0)  Analog output control register (DAC00_DAER, DAC01_DAER)</p>



## 2. Product Lineup

This section explains product lineup of MB9D560 series.

**Table 2-1 Memory Size**

Parameter	MB9DF564	MB9DF565	MB9DF566
FLASH size (program)	(512KB+128KB)×2	(768KB+128KB)×2	(1024KB+128KB)×2
FLASH size (Work)	64KB×2	64KB×2	64KB×2
RAM size	64KB×2	96KB×2	128KB×2

**Table 2-2 Function**

Pin Number	208 pin	176 pin
System clock	On-chip PLL clock multiplication system Minimum instruction execution time :5ns (200MHz)	
CR oscillator (fast/slow)	Yes	
DMAC	16 channels	
Base timer	12 channels (0 to 11)	6 channels (0 to 3, 6, 7)
32-bit free-run timer	5 channels	
32-bit input capture	3 units (6 channels)	
16-bit free-run timer	20 channels <sup>*1</sup>	
16-bit input capture	8 units (0 to 7) (15 channels (0 to 14))	7 units (0 to 6) (13 channels (0 to 12))
16-bit output compare	12 units (0 to 11) (24 channels (0 to 23))	9 units (0 to 5, 9 to 11) (18 channels (0 to 11, 18 to 23))
Waveform generator	4 units (0 to 3) (24 channels (0 to 23))	3 units (0, 1, 3) (18 channels (0 to 11, 18 to 23))
External interrupt	8 channels (0 to 7)	6 channels (0 to 4, 7)
A/D converter	1 unit (32 channels)	
4ch sample-hold A/D converter (4-SH)	2 units (8 channels)	
R/D converter	2 units <sup>*2</sup>	
D/A converter	2 channels <sup>*2</sup>	
Up/Down counter	4 channels	
Motor vector operation accelerator	2 units	
Multi-function serial interface	5 channels (0 to 4)	3 channels (0, 1, 4)
CAN	3 channels	
FlexRay	128 msb x 1 unit (ch.A / ch.B) <sup>*2</sup>	
Inter-processor communications unit (IPCU)	Yes	
Exclusive access memory (EAM)	Yes	
Software watchdog timer	Yes	
Hardware watchdog timer	Yes	
CRC	2 channels	
Internal power supply low-voltage detection	Yes	
External power supply low-voltage detection	Yes	
Key code	Yes <sup>*2</sup>	
Package	LER208	LEP176
Debugging interface	JTAG interface	

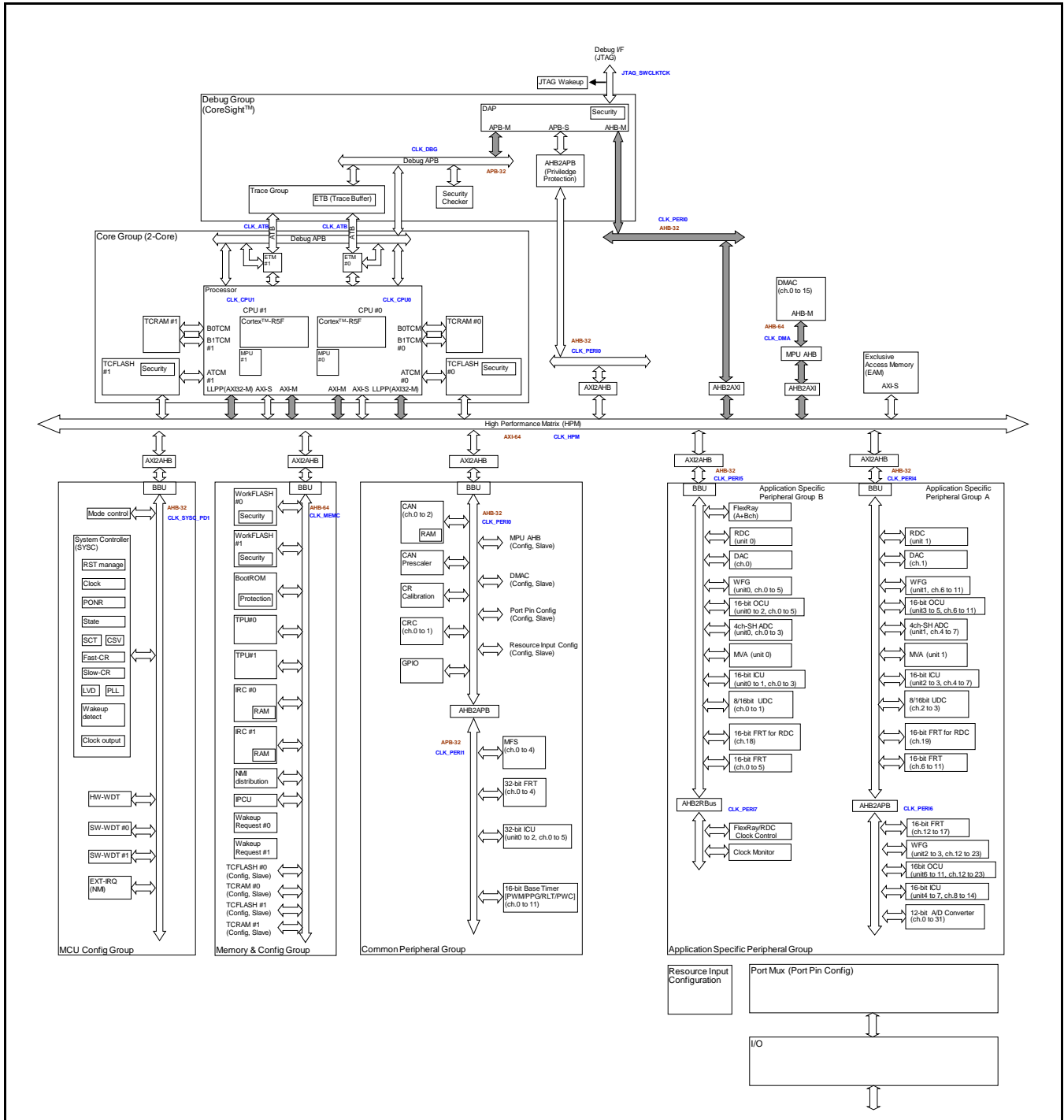
\*1: 2 channels for motor control

\*2: The function is different according to the part number. See Data Sheet.

### 3. Configuration

This section shows configuration of MB9D560 series.

Figure 3-1 Block Diagram



**Note:**

- In the block diagram, block name (Config, Slave) describe bus connection for register setting of control block.





**Table 3-1 Group**

Group Name	Description
Core Group	- CPU and TCM connected memory group
Debug Group	- CoreSight of Debugging group
MCU Config Group	- System control and supervision IP group
Memory & Config Group	- CPU related function and memory group
Common Peripheral Group	- Common peripheral IP group for vehicle application
Application Specific Peripheral Group	- Product specified peripheral group

**Table 3-2 Independent IP**

Name	Description
HPM	- Bus matrix of AXI - Bus bridge (AXI-to-AHB, AHB-to-AXI)
DMAC	- DMA controller
EAM	- Exclusive access memory
Resource input configuration	- Input selection circuit of MCU peripheral
Port MUX	- Port MUX circuit
I/O	- I/O circuit

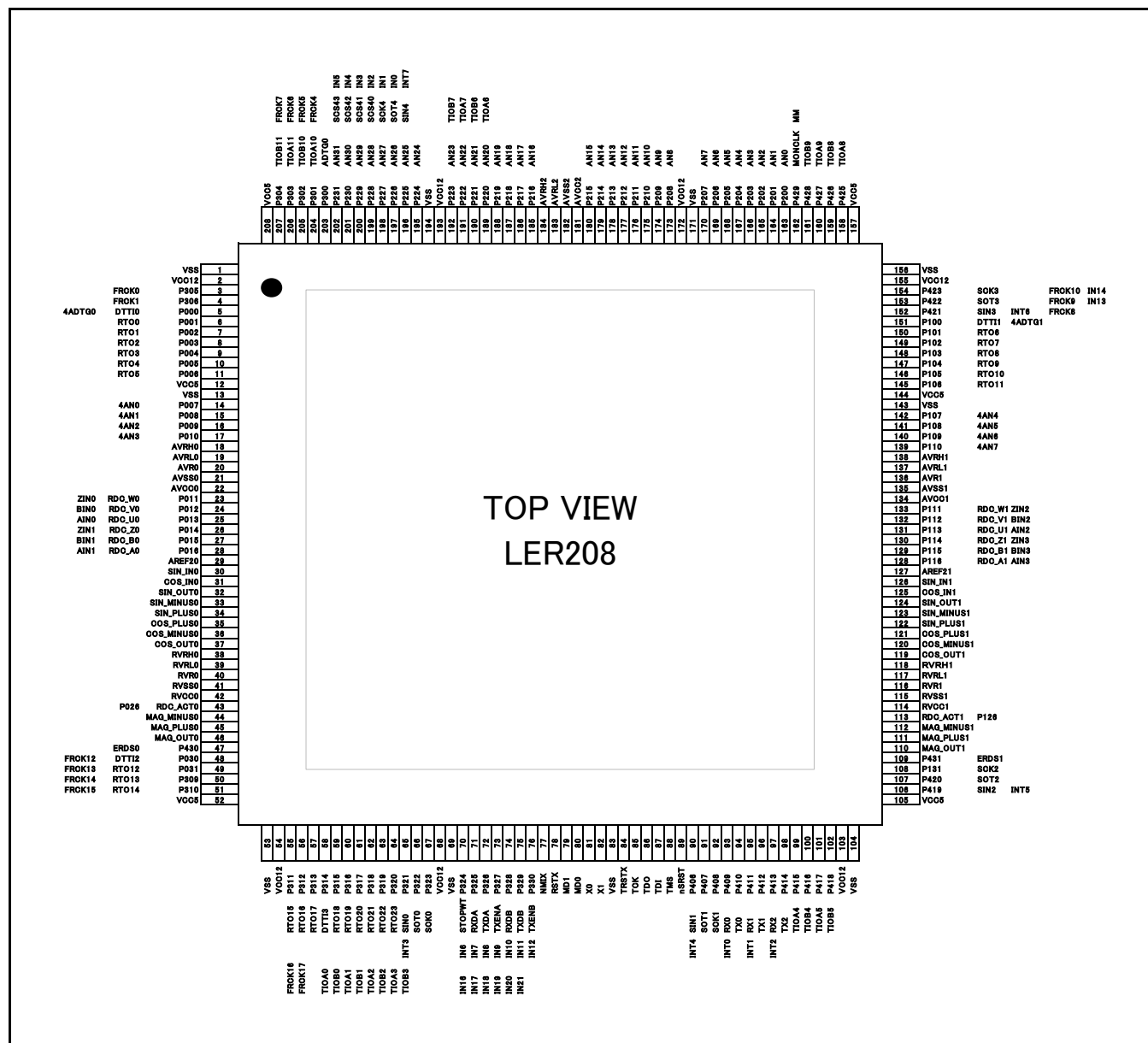
**Notes:**

- Each master connects to HPM. Each master has different transaction ID on AXI, Out-Of-Order for transaction completion.

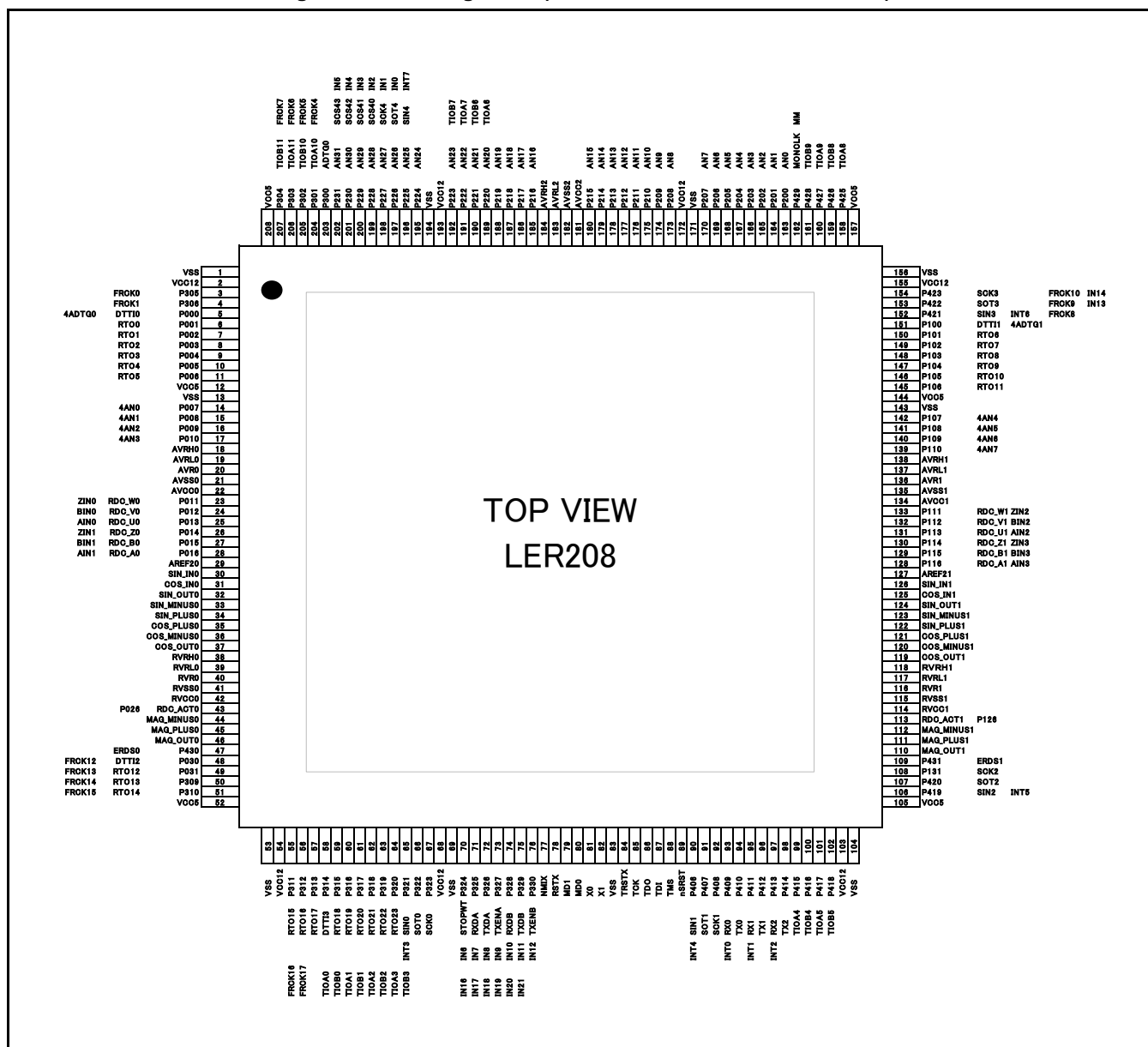
## 4. Pin Assignment

This section shows pin assignment of MB9D560 series.

Figure 4-1 Pin Assignment (208 Pin, Part Number with RDC)



**Figure 4-2 Pin Assignment (208 Pin, Part Number without RDC)**



**Figure 4-3 Pin Assignment (176 Pin, Part Number with RDC)**

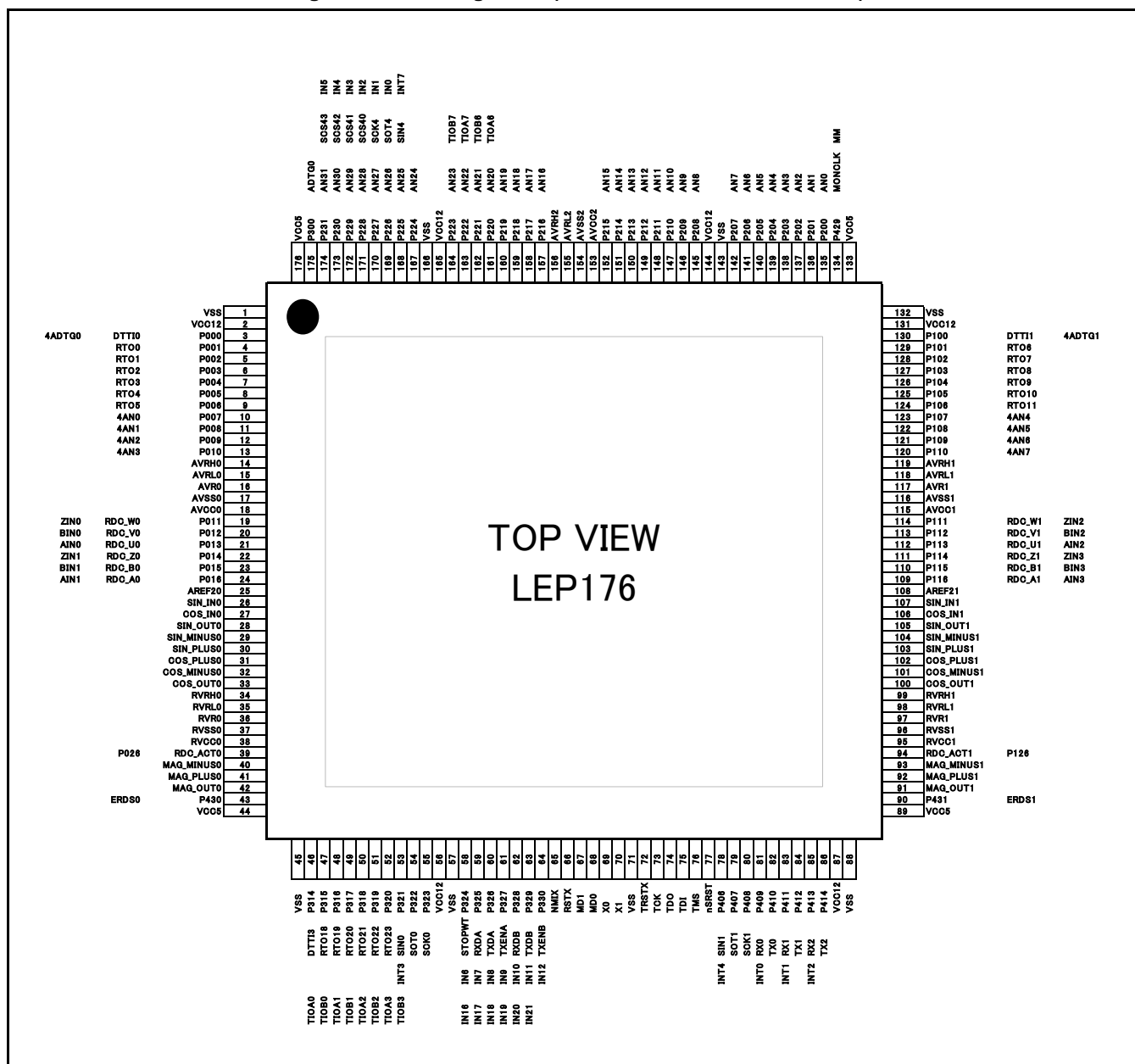
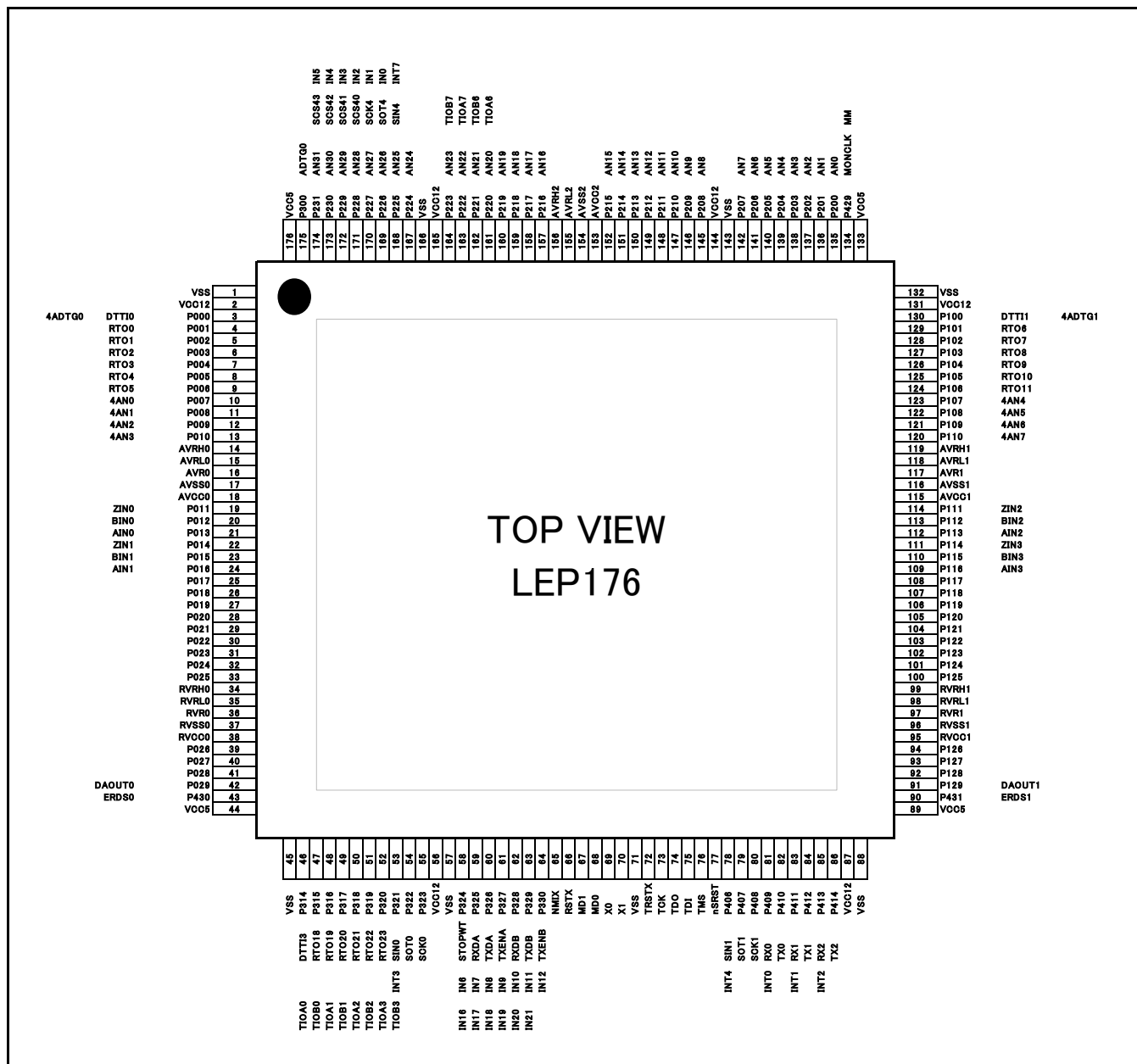




Figure 4-4 Pin Assignment (176 Pin, Part Number without RDC)



## 5. Pin Description

This section explains pin functions.

**Table 5-1 Pin Description (Part Number with RDC)**

Pin Number		Pin Name	I/O Circuit Type	Functions
208 Pin	176 Pin			
3	-	P305 FRCK0	E	General-purpose I/O port 16-bit free-run timer ch.0 external clock input pin
4	-	P306 FRCK1	E	General-purpose I/O port 16-bit free-run timer ch.1 external clock input pin
5	3	P000 DTTI0 4ADTG0	E	General-purpose I/O port Waveform generator output stop signal input pin 0 4ch sample-hold A/D converter unit0 external trigger input pin
6	4	P001 RTO0	E	General-purpose I/O port Waveform generator ch.0 output pin
7	5	P002 RTO1	E	General-purpose I/O port Waveform generator ch.1 output pin
8	6	P003 RTO2	E	General-purpose I/O port Waveform generator ch.2 output pin
9	7	P004 RTO3	E	General-purpose I/O port Waveform generator ch.3 output pin
10	8	P005 RTO4	E	General-purpose I/O port Waveform generator ch.4 output pin
11	9	P006 RTO5	E	General-purpose I/O port Waveform generator ch.5 output pin
14	10	P007 4AN0	F	General-purpose I/O port 4ch sample-hold A/D converter unit0 analog 0 input pin
15	11	P008 4AN1	F	General-purpose I/O port 4ch sample-hold A/D converter unit0 analog 1 input pin
16	12	P009 4AN2	F	General-purpose I/O port 4ch sample-hold A/D converter unit0 analog 2 input pin
17	13	P010 4AN3	F	General-purpose I/O port 4ch sample-hold A/D converter unit0 analog 3 input pin
23	19	P011 RDC_W0 ZIN0	E	General-purpose I/O port R/D converter unit0 W-phase output pin Up/down counter ch.0 ZIN input pin
24	20	P012 RDC_V0 BIN0	E	General-purpose I/O port R/D converter unit0 V-phase output pin Up/down counter ch.0 BIN input pin
25	21	P013 RDC_U0 AIN0	E	General-purpose I/O port R/D converter unit0 U-phase output pin Up/down counter ch.0 AIN input pin
26	22	P014 RDC_Z0 ZIN1	E	General-purpose I/O port R/D converter unit0 Z-phase output pin Up/down counter ch.1 ZIN input pin
27	23	P015 RDC_B0 BIN1	E	General-purpose I/O port R/D converter unit0 B-phase output pin Up/down counter ch.1 BIN input pin
28	24	P016 RDC_A0 AIN1	E	General-purpose I/O port R/D converter unit0 A-phase output pin Up/down counter ch.1 AIN input pin



Pin Number		Pin Name	I/O Circuit Type	Functions
208 Pin	176 Pin			
29	25	AREF20	L	R/D converter unit0 Aref output pin(RVCC0/2)
30	26	SIN_IN0	K	R/D converter unit0 SIN coil earth leakage detection input pin
31	27	COS_IN0	K	R/D converter unit0 COS coil earth leakage detection input pin
32	28	SIN_OUT0	L	R/D converter unit0 SIN output pin
33	29	SIN_MINUS0	K	R/D converter unit0 SIN input pin-
34	30	SIN_PLUS0	K	R/D converter unit0 SIN input pin+
35	31	COS_PLUS0	K	R/D converter unit0 COS input pin+
36	32	COS_MINUS0	K	R/D converter unit0 COS input pin-
37	33	COS_OUT0	L	R/D converter unit0 COS output pin
43	39	RDC_ACT0 P026	E	R/D converter unit0 operation status output pin General-purpose I/O port
44	40	MAG_MINUS0	K	R/D converter unit0 excitation external input pin-
45	41	MAG_PLUS0	K	R/D converter unit0 excitation external input pin+
46	42	MAG_OUT0	L	R/D converter unit0 excitation signal output pin
47	43	P430 ERDS0	E	General-purpose I/O port Error detection output pin ch.0
48	-	P030 DTTI2 FRCK12	E	General-purpose I/O port Waveform generator output stop signal input pin 2 16-bit free-run timer ch.12 external clock input pin
49	-	P031 RTO12 FRCK13	E	General-purpose I/O port Waveform generator ch.12 output pin 16-bit free-run timer ch.13 external clock input pin
50	-	P309 RTO13 FRCK14	E	General-purpose I/O port Waveform generator ch.13 output pin 16-bit free-run timer ch.14 external clock input pin
51	-	P310 RTO14 FRCK15	E	General-purpose I/O port Waveform generator ch.14 output pin 16-bit free-run timer ch.15 external clock input pin
55	-	P311 RTO15 FRCK16	E	General-purpose I/O port Waveform generator ch.15 output pin 16-bit free-run timer ch.16 external clock input pin
56	-	P312 RTO16 FRCK17	E	General-purpose I/O port Waveform generator ch.16 output pin 16-bit free-run timer ch.17 external clock input pin
57	-	P313 RTO17	E	General-purpose I/O port Waveform generator ch.17 output pin
58	46	P314 DTTI3 TIOA0	E	General-purpose I/O port Waveform generator output stop signal input pin 3 Base timer ch.0 TIOA output pin
59	47	P315 RTO18 TIOB0	E	General-purpose I/O port Waveform generator ch.18 output pin Base timer ch.0 TIOB input pin
60	48	P316 RTO19 TIOA1	E	General-purpose I/O port Waveform generator ch.19 output pin Base timer ch.1 TIOA I/O pin
61	49	P317 RTO20 TIOB1	E	General-purpose I/O port Waveform generator ch.20 output pin Base timer ch.1 TIOB input pin

Pin Number		Pin Name	I/O Circuit Type	Functions
208 Pin	176 Pin			
62	50	P318 RTO21 TIOA2	E	General-purpose I/O port Waveform generator ch.21 output pin Base timer ch.2 TIOA output pin
63	51	P319 RTO22 TIOB2	E	General-purpose I/O port Waveform generator ch.22 output pin Base timer ch.2 TIOB input pin
64	52	P320 RTO23 TIOA3	E	General-purpose I/O port Waveform generator ch.23 output pin Base timer ch.3 TIOA I/O pin
65	53	P321 SIN0 INT3 TIOB3	E	General-purpose I/O port Multi-function serial interface ch.0 serial data input pin INT3 external interrupt input pin Base timer ch.3 TIOB input pin
66	54	P322 SOT0	E	General-purpose I/O port Multi-function serial interface ch.0 serial data output pin
67	55	P323 SCK0	E	General-purpose I/O port Multi-function serial interface ch.0 clock I/O pin
70	58	P324 STOPWT IN6 IN16	E	General-purpose I/O port FlexRay stop watch input pin 16-bit input capture ch.6 external pulse input pin 32-bit input capture ch.0 external pulse input pin
71	59	P325 RXDA IN7 IN17	H	General-purpose I/O port FlexRay ch.A data input pin 16-bit input capture ch.7 external pulse input pin 32-bit input capture ch.1 external pulse input pin
72	60	P326 TXDA IN8 IN18	H	General-purpose I/O port FlexRay ch.A data output pin 16-bit input capture ch.8 external pulse input pin 32-bit input capture ch.2 external pulse input pin
73	61	P327 TXENA IN9 IN19	H	General-purpose I/O port FlexRay ch.A operation enable output pin 16-bit input capture ch.9 external pulse input pin 32-bit input capture ch.3 external pulse input pin
74	62	P328 RXDB IN10 IN20	H	General-purpose I/O port FlexRay ch.B data input pin 16-bit input capture ch.10 external pulse input pin 32-bit input capture ch.4 external pulse input pin
75	63	P329 TXDB IN11 IN21	H	General-purpose I/O port FlexRay ch.B data output pin 16-bit input capture ch.11 external pulse input pin 32-bit input capture ch.5 external pulse input pin
76	64	P330 TXENB IN12	H	General-purpose I/O port FlexRay ch.B operation enable output pin 16-bit input capture ch.12 external pulse input pin
77	65	NMIX	B	Non-maskable interrupt input pin
78	66	RSTX	B	External reset input pin
79	67	MD1	C	Mode pin 1 (with high-voltage control)
80	68	MD0	C	Mode pin 0 (with high-voltage control)
81	69	X0	A	Main clock oscillation input pin
82	70	X1		Main clock oscillation output pin





Pin Number		Pin Name	I/O Circuit Type	Functions
208 Pin	176 Pin			
84	72	TRSTX	J	JTAG test reset input
85	73	TCK	J	JTAG test clock input
86	74	TDO	I	JTAG test data output
87	75	TDI	J	JTAG test data input
88	76	TMS	J	JTAG test mode status input
89	77	nSRST	J	System reset input for debugger
90	78	P406 SIN1 INT4	E	General-purpose I/O port Multi-function serial interface ch.1 serial data input pin INT4 external interrupt input pin
91	79	P407 SOT1	E	General-purpose I/O port Multi-function serial interface ch.1 serial data output pin
92	80	P408 SCK1	E	General-purpose I/O port Multi-function serial interface ch.1 clock I/O pin
93	81	P409 RX0 INT0	E	General-purpose I/O port CAN ch.0 reception data input pin INT0 external interrupt input pin
94	82	P410 TX0	E	General-purpose I/O port CAN ch.0 transmission data output pin
95	83	P411 RX1 INT1	E	General-purpose I/O port CAN ch.1 reception data input pin INT1 external interrupt input pin
96	84	P412 TX1	E	General-purpose I/O port CAN ch.1 transmission data output pin
97	85	P413 RX2 INT2	E	General-purpose I/O port CAN ch.2 reception data input pin INT2 external interrupt input pin
98	86	P414 TX2	E	General-purpose I/O port CAN ch.2 transmission data output pin
99	-	P415 TIOA4	E	General-purpose I/O port Base timer ch.4 TIOA output pin
100	-	P416 TIOB4	E	General-purpose I/O port Base timer ch.4 TIOB input pin
101	-	P417 TIOA5	E	General-purpose I/O port Base timer ch.5 TIOA I/O pin
102	-	P418 TIOB5	E	General-purpose I/O port Base timer ch.5 TIOB input pin
106	-	P419 SIN2 INT5	E	General-purpose I/O port Multi-function serial interface ch.2 serial data input pin INT5 external interrupt input pin
107	-	P420 SOT2	E	General-purpose I/O port Multi-function serial interface ch.2 serial data output pin
108	-	P131 SCK2	E	General-purpose I/O port Multi-function serial interface ch.2 clock I/O pin
109	90	P431 ERDS1	E	General-purpose I/O port Error detection output pin ch.1
110	91	MAG_OUT1	L	R/D converter unit1 excitation signal output pin
111	92	MAG_PLUS1	K	R/D converter unit1 excitation external input pin+
112	93	MAG_MINUS1	K	R/D converter unit1 excitation external input pin-

Pin Number		Pin Name	I/O Circuit Type	Functions
208 Pin	176 Pin			
113	94	RDC_ACT1 P126	E	R/D converter unit1 operation status output pin General-purpose I/O port
119	100	COS_OUT1	L	R/D converter unit1 COS output pin
120	101	COS_MINUS1	K	R/D converter unit1 COS input pin-
121	102	COS_PLUS1	K	R/D converter unit1 COS input pin+
122	103	SIN_PLUS1	K	R/D converter unit1 SIN input pin+
123	104	SIN_MINUS1	K	R/D converter unit1 SIN input pin-
124	105	SIN_OUT1	L	R/D converter unit1 SIN output pin
125	106	COS_IN1	K	R/D converter unit1 COS coil earth leakage detection input pin
126	107	SIN_IN1	K	R/D converter unit1 SIN coil earth leakage detection input pin
127	108	AREF21	L	R/D converter unit1 Aref output pin(RVCC1/2)
128	109	P116 RDC_A1 AIN3	E	General-purpose I/O port R/D converter unit1 A phase output pin Up/down counter ch.3 AIN input pin
129	110	P115 RDC_B1 BIN3	E	General-purpose I/O port R/D converter unit1 B phase output pin Up/down counter ch.3 BIN input pin
130	111	P114 RDC_Z1 ZIN3	E	General-purpose I/O port R/D converter unit1 Z phase output pin Up/down counter ch.3 ZIN input pin
131	112	P113 RDC_U1 AIN2	E	General-purpose I/O port R/D converter unit1 U phase output pin Up/down counter ch.2 AIN input pin
132	113	P112 RDC_V1 BIN2	E	General-purpose I/O port R/D converter unit1 V phase output pin Up/down counter ch.2 BIN input pin
133	114	P111 RDC_W1 ZIN2	E	General-purpose I/O port R/D converter unit1 W phase output pin Up/down counter ch.2 ZIN input pin
139	120	P110 4AN7	F	General-purpose I/O port 4ch sample-hold A/D converter unit1 analog 7 input pin
140	121	P109 4AN6	F	General-purpose I/O port 4ch sample-hold A/D converter unit1 analog 6 input pin
141	122	P108 4AN5	F	General-purpose I/O port 4ch sample-hold A/D converter unit1 analog 5 input pin
142	123	P107 4AN4	F	General-purpose I/O port 4ch sample-hold A/D converter unit1 analog 4 input pin
145	124	P106 RTO11	E	General-purpose I/O port Waveform generator ch.11 output pin
146	125	P105 RTO10	E	General-purpose I/O port Waveform generator ch.10 output pin
147	126	P104 RTO9	E	General-purpose I/O port Waveform generator ch.9 output pin
148	127	P103 RTO8	E	General-purpose I/O port Waveform generator ch.8 output pin
149	128	P102 RTO7	E	General-purpose I/O port Waveform generator ch.7 output pin
150	129	P101 RTO6	E	General-purpose I/O port Waveform generator ch.6 output pin



Pin Number		Pin Name	I/O Circuit Type	Functions
208 Pin	176 Pin			
151	130	P100 DTT11 4ADTG1	E	General-purpose I/O port Waveform generator output stop signal input pin1 4ch sample-hold A/D converter unit1 external trigger input pin
152	-	P421 SIN3 INT6 FRCK8	E	General-purpose I/O port Multi-function serial interface ch.3 serial data input pin INT6 external interrupt input pin 16-bit free-run timer ch.8 external clock input pin
153	-	P422 SOT3 FRCK9 IN13	E	General-purpose I/O port Multi -function serial interface ch.3 serial data output pin 16-bit free-run timer ch.9 external clock input pin 16-bit input capture ch.13 external pulse input pin
154	-	P423 SCK3 FRCK10 IN14	E	General-purpose I/O port Multi-function serial interface ch.3 clock I/O pin 16-bit free-run timer ch.10 external clock input pin 16-bit input capture ch.14 external pulse input pin
158	-	P425 TIOA8	E	General-purpose I/O port Base timer ch.8 TIOA output pin
159	-	P426 TIOB8	E	General-purpose I/O port Base timer ch.8 TIOB input pin
160	-	P427 TIOA9	E	General-purpose I/O port Base timer ch.9 TIOA I/O pin
161	-	P428 TIOB9	E	General-purpose I/O port Base timer ch.9 TIOB input pin
162	134	P429 MONCLK MM	E	General-purpose I/O port Clock monitor output pin Clock supervisor main clock error detection output pin
163	135	P200 AN0	F	General-purpose I/O port A/D converter analog 0 input pin
164	136	P201 AN1	F	General-purpose I/O port A/D converter analog 1 input pin
165	137	P202 AN2	F	General-purpose I/O port A/D converter analog 2 input pin
166	138	P203 AN3	F	General-purpose I/O port A/D converter analog 3 input pin
167	139	P204 AN4	F	General-purpose I/O port A/D converter analog 4 input pin
168	140	P205 AN5	F	General-purpose I/O port A/D converter analog 5 input pin
169	141	P206 AN6	F	General-purpose I/O port A/D converter analog 6 input pin
170	142	P207 AN7	F	General-purpose I/O port A/D converter analog 7 input pin
173	145	P208 AN8	F	General-purpose I/O port A/D converter analog 8 input pin
174	146	P209 AN9	F	General-purpose I/O port A/D converter analog 9 input pin
175	147	P210 AN10	F	General-purpose I/O port A/D converter analog 10 input pin
176	148	P211 AN11	F	General-purpose I/O port A/D converter analog 11 input pin

Pin Number		Pin Name	I/O Circuit Type	Functions
208 Pin	176 Pin			
177	149	P212 AN12	F	General-purpose I/O port A/D converter analog 12 input pin
178	150	P213 AN13	F	General-purpose I/O port A/D converter analog 13 input pin
179	151	P214 AN14	F	General-purpose I/O port A/D converter analog 14 input pin
180	152	P215 AN15	F	General-purpose I/O port A/D converter analog 15 input pin
185	157	P216 AN16	F	General-purpose I/O port A/D converter analog 16 input pin
186	158	P217 AN17	F	General-purpose I/O port A/D converter analog 17 input pin
187	159	P218 AN18	F	General-purpose I/O port A/D converter analog 18 input pin
188	160	P219 AN19	F	General-purpose I/O port A/D converter analog 19 input pin
189	161	P220 AN20 TIOA6	F	General-purpose I/O port A/D converter analog 20 input pin Base timer ch.6 TIOA output pin
190	162	P221 AN21 TIOB6	F	General-purpose I/O port A/D converter analog 21 input pin Base timer ch.6 TIOB input pin
191	163	P222 AN22 TIOA7	F	General-purpose I/O port A/D converter analog 22 input pin Base timer ch.7 TIOA I/O pin
192	164	P223 AN23 TIOB7	F	General-purpose I/O port A/D converter analog 23 input pin Base timer ch.7 TIOB input pin
195	167	P224 AN24	F	General-purpose I/O port A/D converter Analog 24 input pin
196	168	P225 AN25 SIN4 INT7	F	General-purpose I/O port A/D converter analog 25 input pin Multi-function serial interface ch.4 serial data input pin INT7 external interrupt input pin
197	169	P226 AN26 SOT4 IN0	F	General-purpose I/O port A/D converter analog 26 input pin Multi-function serial interface ch.4 serial data output pin 16-bit input capture ch.0 external pulse input pin
198	170	P227 AN27 SCK4 IN1	F	General-purpose I/O port A/D converter analog 27 input pin Multi-function serial interface ch.4 clock I/O pin 16-bit input capture ch.1 external pulse input pin
199	171	P228 AN28 SCS40 IN2	F	General-purpose I/O port A/D converter analog 28 input pin Multi-function serial interface ch.4 serial chip select 0 I/O pin 16-bit input capture ch.2 external pulse input pin
200	172	P229 AN29 SCS41 IN3	F	General-purpose I/O port A/D converter analog 29 input pin Multi-function serial interface ch.4 serial chip select 1 I/O pin 16-bit input capture ch.3 external pulse input pin



Pin Number		Pin Name	I/O Circuit Type	Functions
208 Pin	176 Pin			
201	173	P230 AN30 SCS42 IN4	F	General-purpose I/O port A/D converter analog 30 input pin Multi-function serial interface ch.4 serial chip select 2 I/O pin 16-bit input capture ch.4 external pulse input pin
202	174	P231 AN31 SCS43 IN5	F	General-purpose I/O port A/D converter analog 31 input pin Multi-function serial interface ch.4 serial chip select 3 I/O pin 16-bit input capture ch.5 external pulse input pin
203	175	P300 ADTG0	E	General-purpose I/O port A/D converter external trigger input pin
204	-	P301 TIOA10 FRCK4	E	General-purpose I/O port Base timer ch.10 TIOA output pin 16-bit free-run timer ch.4 external clock input pin
205	-	P302 TIOB10 FRCK5	E	General-purpose I/O port Base timer ch.10 TIOB input pin 16-bit free-run timer ch.5 external clock input pin
206	-	P303 TIOA11 FRCK6	E	General-purpose I/O port Base timer ch.11 TIOA I/O pin 16-bit free-run timer ch.6 external clock input pin
207	-	P304 TIOB11 FRCK7	E	General-purpose I/O port Base timer ch.11 TIOB input pin 16-bit free-run timer ch.7 external clock input pin
18	14	AVRH0	-	4ch sample-hold A/D converter unit0 upper limit reference voltage
19	15	AVRL0	-	4ch sample-hold A/D converter unit0 lower limit reference voltage
20	16	AVR0	-	4ch sample-hold A/D converter unit0 reference voltage
21	17	AVSS0	-	4ch sample-hold A/D converter unit0 analog GND
22	18	AVCC0	-	4ch sample-hold A/D converter unit0 analog power supply
134	115	AVCC1	-	4ch sample-hold A/D converter unit1 analog power supply
135	116	AVSS1	-	4ch sample-hold A/D converter unit1 analog GND
136	117	AVR1	-	4ch sample-hold A/D converter unit1 reference voltage
137	118	AVRL1	-	4ch sample-hold A/D converter unit1 lower limit reference voltage
138	119	AVRH1	-	4ch sample-hold A/D converter unit1 upper limit reference voltage
38	34	RVRH0	-	R/D converter unit0 upper limit reference voltage
39	35	RVRL0	-	R/D converter unit0 lower limit reference voltage
40	36	RVR0	-	R/D converter unit0 reference voltage
41	37	RVSS0	-	R/D converter unit0 analog GND
42	38	RVCC0	-	R/D converter unit0 analog power supply
114	95	RVCC1	-	R/D converter unit1 analog power supply
115	96	RVSS1	-	R/D converter unit1 analog GND
116	97	RVR1	-	R/D converter unit1 reference voltage
117	98	RVRL1	-	R/D converter unit1 lower limit reference voltage
118	99	RVRH1	-	R/D converter unit1 upper limit reference voltage
181	153	AVCC2	-	A/D converter analog power supply
182	154	AVSS2	-	A/D converter analog GND
183	155	AVRL2	-	A/D converter lower limit reference voltage
184	156	AVRH2	-	A/D converter upper limit reference voltage

Pin Number		Pin Name	I/O Circuit Type	Functions
208 Pin	176 Pin			
2 54 68 103 155 172 193	2 56 87 131 144 165	VCC12	-	1.2V power supply
12 52 105 144 157 208	44 89 133 176	VCC5	-	5.0V power supply
1 13 53 69 83 104 143 156 171 194	1 45 57 71 88 132 143 166	VSS	-	GND



Table 5-2 Pin Description (Part Number without RDC)

Pin Number		Pin Name	I/O Circuit Type	Functions
208 Pin	176 Pin			
3	-	P305 FRCK0	E	General-purpose I/O port 16-bit free-run timer ch.0 external clock input pin
4	-	P306 FRCK1	E	General-purpose I/O port 16-bit free-run timer ch.1 external clock input pin
5	3	P000 DTTI0 4ADTG0	E	General-purpose I/O port Waveform generator output stop signal input pin 0 4ch sample-hold A/D converter unit0 external trigger input pin
6	4	P001 RTO0	E	General-purpose I/O port Waveform generator ch.0 output pin
7	5	P002 RTO1	E	General-purpose I/O port Waveform generator ch.1 output pin
8	6	P003 RTO2	E	General-purpose I/O port Waveform generator ch.2 output pin
9	7	P004 RTO3	E	General-purpose I/O port Waveform generator ch.3 output pin
10	8	P005 RTO4	E	General-purpose I/O port Waveform generator ch.4 output pin
11	9	P006 RTO5	E	General-purpose I/O port Waveform generator ch.5 output pin
14	10	P007 4AN0	F	General-purpose I/O port 4ch sample-hold A/D converter unit0 analog 0 input pin
15	11	P008 4AN1	F	General-purpose I/O port 4ch sample-hold A/D converter unit0 analog 1 input pin
16	12	P009 4AN2	F	General-purpose I/O port 4ch sample-hold A/D converter unit0 analog 2 input pin
17	13	P010 4AN3	F	General-purpose I/O port 4ch sample-hold A/D converter unit0 analog 3 input pin
23	19	P011 ZIN0	E	General-purpose I/O port Up/down counter ch.0 ZIN input pin
24	20	P012 BIN0	E	General-purpose I/O port Up/down counter ch.0 BIN input pin
25	21	P013 AIN0	E	General-purpose I/O port Up/down counter ch.0 AIN input pin
26	22	P014 ZIN1	E	General-purpose I/O port Up/down counter ch.1 ZIN input pin
27	23	P015 BIN1	E	General-purpose I/O port Up/down counter ch.1 BIN input pin
28	24	P016 AIN1	E	General-purpose I/O port Up/down counter ch.1 AIN input pin
29	25	P017	E	General-purpose I/O port
30	26	P018	E	General-purpose I/O port
31	27	P019	E	General-purpose I/O port
32	28	P020	E	General-purpose I/O port
33	29	P021	E	General-purpose I/O port
34	30	P022	E	General-purpose I/O port
35	31	P023	E	General-purpose I/O port
36	32	P024	E	General-purpose I/O port
37	33	P025	E	General-purpose I/O port

Pin Number		Pin Name	I/O Circuit Type	Functions
208 Pin	176 Pin			
43	39	P026	E	General-purpose I/O port
44	40	P027	E	General-purpose I/O port
45	41	P028	E	General-purpose I/O port
46	42	P029 DAOUT0	G	General-purpose I/O port D/A converter ch.0 analog output pin
47	43	P430 ERDS0	E	General-purpose I/O port Error detection output pin ch.0
48	-	P030 DTT12 FRCK12	E	General-purpose I/O port Waveform generator output stop signal input pin 2 16-bit free-run timer ch.12 external clock input pin
49	-	P031 RTO12 FRCK13	E	General-purpose I/O port Waveform generator ch.12 output pin 16-bit free-run timer ch.13 external clock input pin
50	-	P309 RTO13 FRCK14	E	General-purpose I/O port Waveform generator ch.13 output pin 16-bit free-run timer ch.14 external clock input pin
51	-	P310 RTO14 FRCK15	E	General-purpose I/O port Waveform generator ch.14 output pin 16-bit free-run timer ch.15 external clock input pin
55	-	P311 RTO15 FRCK16	E	General-purpose I/O port Waveform generator ch.15 output pin 16-bit free-run timer ch.16 external clock input pin
56	-	P312 RTO16 FRCK17	E	General-purpose I/O port Waveform generator ch.16 output pin 16-bit free-run timer ch.17 external clock input pin
57	-	P313 RTO17	E	General-purpose I/O port Waveform generator ch.17 output pin
58	46	P314 DTT13 TIOA0	E	General-purpose I/O port Waveform generator output stop signal input pin 3 Base timer ch.0 TIOA output pin
59	47	P315 RTO18 TIOB0	E	General-purpose I/O port Waveform generator ch.18 output pin Base timer ch.0 TIOB input pin
60	48	P316 RTO19 TIOA1	E	General-purpose I/O port Waveform generator ch.19 output pin Base timer ch.1 TIOA I/O pin
61	49	P317 RTO20 TIOB1	E	General-purpose I/O port Waveform generator ch.20 output pin Base timer ch.1 TIOB input pin
62	50	P318 RTO21 TIOA2	E	General-purpose I/O port Waveform generator ch.21 output pin Base timer ch.2 TIOA output pin
63	51	P319 RTO22 TIOB2	E	General-purpose I/O port Waveform generator ch.22 output pin Base timer ch.2 TIOB input pin
64	52	P320 RTO23 TIOA3	E	General-purpose I/O port Waveform generator ch.23 output pin Base timer ch.3 TIOA I/O pin





Pin Number		Pin Name	I/O Circuit Type	Functions
208 Pin	176 Pin			
65	53	P321 SIN0 INT3 TIOB3	E	General-purpose I/O port Multi-function serial interface ch.0 serial data input pin INT3 external interrupt input pin Base timer ch.3 TIOB input pin
66	54	P322 SOT0	E	General-purpose I/O port Multi-function serial interface ch.0 serial data output pin
67	55	P323 SCK0	E	General-purpose I/O port Multi-function serial interface ch.0 clock I/O pin
70	58	P324 STOPWT IN6 IN16	E	General-purpose I/O port FlexRay stop watch input pin 16-bit input capture ch.6 external pulse input pin 32-bit input capture ch.0 external pulse input pin
71	59	P325 RXDA IN7 IN17	H	General-purpose I/O port FlexRay ch.A data input pin 16-bit input capture ch.7 external pulse input pin 32-bit input capture ch.1 external pulse input pin
72	60	P326 TXDA IN8 IN18	H	General-purpose I/O port FlexRay ch.A data output pin 16-bit input capture ch.8 external pulse input pin 32-bit input capture ch.2 external pulse input pin
73	61	P327 TXENA IN9 IN19	H	General-purpose I/O port FlexRay ch.A operation enable output pin 16-bit input capture ch.9 external pulse input pin 32-bit input capture ch.3 external pulse input pin
74	62	P328 RXDB IN10 IN20	H	General-purpose I/O port FlexRay ch.B data input pin 16-bit input capture ch.10 external pulse input pin 32-bit input capture ch.4 external pulse input pin
75	63	P329 TXDB IN11 IN21	H	General-purpose I/O port FlexRay ch.B data output pin 16-bit input capture ch.11 external pulse input pin 32-bit input capture ch.5 external pulse input pin
76	64	P330 TXENB IN12	H	General-purpose I/O port FlexRay ch.B operation enable output pin 16-bit input capture ch.12 external pulse input pin
77	65	NMIX	B	Non-maskable interrupt input pin
78	66	RSTX	B	External reset input pin
79	67	MD1	C	Mode pin 1 (with high-voltage control)
80	68	MD0	C	Mode pin 0 (with high-voltage control)
81	69	X0	A	Main clock oscillation input pin
82	70	X1		Main clock oscillation output pin
84	72	TRSTX	J	JTAG test reset input
85	73	TCK	J	JTAG test clock input
86	74	TDO	I	JTAG test data output
87	75	TDI	J	JTAG test data input
88	76	TMS	J	JTAG test mode status input
89	77	nSRST	J	System reset input for debugger
90	78	P406 SIN1 INT4	E	General-purpose I/O port Multi-function serial interface ch.1 serial data input pin INT4 external interrupt input pin

Pin Number		Pin Name	I/O Circuit Type	Functions
208 Pin	176 Pin			
91	79	P407 SOT1	E	General-purpose I/O port Multi-function serial interface ch.1 serial data output pin
92	80	P408 SCK1	E	General-purpose I/O port Multi-function serial interface ch.1 clock I/O pin
93	81	P409 RX0 INT0	E	General-purpose I/O port CAN ch.0 reception data input pin INT0 external interrupt input pin
94	82	P410 TX0	E	General-purpose I/O port CAN ch.0 transmission data output pin
95	83	P411 RX1 INT1	E	General-purpose I/O port CAN ch.1 reception data input pin INT1 external interrupt input pin
96	84	P412 TX1	E	General-purpose I/O port CAN ch.1 transmission data output pin
97	85	P413 RX2 INT2	E	General-purpose I/O port CAN ch.2 reception data input pin INT2 external interrupt input pin
98	86	P414 TX2	E	General-purpose I/O port CAN ch.2 transmission data output pin
99	-	P415 TIOA4	E	General-purpose I/O port Base timer ch.4 TIOA output pin
100	-	P416 TIOB4	E	General-purpose I/O port Base timer ch.4 TIOB input pin
101	-	P417 TIOA5	E	General-purpose I/O port Base timer ch.5 TIOA I/O pin
102	-	P418 TIOB5	E	General-purpose I/O port Base timer ch.5 TIOB input pin
106	-	P419 SIN2 INT5	E	General-purpose I/O port Multi-function serial interface ch.2 serial data input pin INT5 external interrupt input pin
107	-	P420 SOT2	E	General-purpose I/O port Multi-function serial interface ch.2 serial data output pin
108	-	P131 SCK2	E	General-purpose I/O port Multi-function serial interface ch.2 clock I/O pin
109	90	P431 ERDS1	E	General-purpose I/O port Error detection output pin ch.1
110	91	P129 DAOUT1	G	General-purpose I/O port D/A converter ch.1 analog output pin
111	92	P128	E	General-purpose I/O port
112	93	P127	E	General-purpose I/O port
113	94	P126	E	General-purpose I/O port
119	100	P125	E	General-purpose I/O port
120	101	P124	E	General-purpose I/O port
121	102	P123	E	General-purpose I/O port
122	103	P122	E	General-purpose I/O port
123	104	P121	E	General-purpose I/O port
124	105	P120	E	General-purpose I/O port
125	106	P119	E	General-purpose I/O port
126	107	P118	E	General-purpose I/O port



Pin Number		Pin Name	I/O Circuit Type	Functions
208 Pin	176 Pin			
127	108	P117	E	General-purpose I/O port
128	109	P116 AIN3	E	General-purpose I/O port Up/down counter ch.3 AIN input pin
129	110	P115 BIN3	E	General-purpose I/O port Up/down counter ch.3 BIN input pin
130	111	P114 ZIN3	E	General-purpose I/O port Up/down counter ch.3 ZIN input pin
131	112	P113 AIN2	E	General-purpose I/O port Up/down counter ch.2 AIN input pin
132	113	P112 BIN2	E	General-purpose I/O port Up/down counter ch.2 BIN input pin
133	114	P111 ZIN2	E	General-purpose I/O port Up/down counter ch.2 ZIN input pin
139	120	P110 4AN7	F	General-purpose I/O port 4ch sample-hold A/D converter unit1 analog 7 input pin
140	121	P109 4AN6	F	General-purpose I/O port 4ch sample-hold A/D converter unit1 analog 6 input pin
141	122	P108 4AN5	F	General-purpose I/O port 4ch sample-hold A/D converter unit1 analog 5 input pin
142	123	P107 4AN4	F	General-purpose I/O port 4ch sample-hold A/D converter unit1 analog 4 input pin
145	124	P106 RTO11	E	General-purpose I/O port Waveform generator ch.11 output pin
146	125	P105 RTO10	E	General-purpose I/O port Waveform generator ch.10 output pin
147	126	P104 RTO9	E	General-purpose I/O port Waveform generator ch.9 output pin
148	127	P103 RTO8	E	General-purpose I/O port Waveform generator ch.8 output pin
149	128	P102 RTO7	E	General-purpose I/O port Waveform generator ch.7 output pin
150	129	P101 RTO6	E	General-purpose I/O port Waveform generator ch.6 output pin
151	130	P100 DTT11 4ADTG1	E	General-purpose I/O port Waveform generator output stop signal input pin 1 4ch sample-hold A/D converter unit1 external trigger input pin
152	-	P421 SIN3 INT6 FRCK8	E	General-purpose I/O port Multi-function serial interface ch.3 serial data input pin INT6 external interrupt input pin 16-bit free-run timer ch.8 external clock input pin
153	-	P422 SOT3 FRCK9 IN13	E	General-purpose I/O port Multi -function serial interface ch.3 serial data output pin 16-bit free-run timer ch.9 external clock input pin 16-bit input capture ch.13 external pulse input pin
154	-	P423 SCK3 FRCK10 IN14	E	General-purpose I/O port Multi-function serial interface ch.3 clock I/O pin 16-bit free-run timer ch.10 external clock input pin 16-bit input capture ch.14 external pulse input pin
158	-	P425 TIOA8	E	General-purpose I/O port Base timer ch.8 TIOA output pin

Pin Number		Pin Name	I/O Circuit Type	Functions
208 Pin	176 Pin			
159	-	P426 TIOB8	E	General-purpose I/O port Base timer ch.8 TIOB input pin
160	-	P427 TIOA9	E	General-purpose I/O port Base timer ch.9 TIOA I/O pin
161	-	P428 TIOB9	E	General-purpose I/O port Base timer ch.9 TIOB input pin
162	134	P429 MONCLK MM	E	General-purpose I/O port Clock monitor output pin Clock supervisor main clock error detection output pin
163	135	P200 AN0	F	General-purpose I/O port A/D converter analog 0 input pin
164	136	P201 AN1	F	General-purpose I/O port A/D converter analog 1 input pin
165	137	P202 AN2	F	General-purpose I/O port A/D converter analog 2 input pin
166	138	P203 AN3	F	General-purpose I/O port A/D converter analog 3 input pin
167	139	P204 AN4	F	General-purpose I/O port A/D converter analog 4 input pin
168	140	P205 AN5	F	General-purpose I/O port A/D converter analog 5 input pin
169	141	P206 AN6	F	General-purpose I/O port A/D converter analog 6 input pin
170	142	P207 AN7	F	General-purpose I/O port A/D converter analog 7 input pin
173	145	P208 AN8	F	General-purpose I/O port A/D converter analog 8 input pin
174	146	P209 AN9	F	General-purpose I/O port A/D converter analog 9 input pin
175	147	P210 AN10	F	General-purpose I/O port A/D converter analog 10 input pin
176	148	P211 AN11	F	General-purpose I/O port A/D converter analog 11 input pin
177	149	P212 AN12	F	General-purpose I/O port A/D converter analog 12 input pin
178	150	P213 AN13	F	General-purpose I/O port A/D converter analog 13 input pin
179	151	P214 AN14	F	General-purpose I/O port A/D converter analog 14 input pin
180	152	P215 AN15	F	General-purpose I/O port A/D converter analog 15 input pin
185	157	P216 AN16	F	General-purpose I/O port A/D converter analog 16 input pin
186	158	P217 AN17	F	General-purpose I/O port A/D converter analog 17 input pin
187	159	P218 AN18	F	General-purpose I/O port A/D converter analog 18 input pin
188	160	P219 AN19	F	General-purpose I/O port A/D converter analog 19 input pin



Pin Number		Pin Name	I/O Circuit Type	Functions
208 Pin	176 Pin			
189	161	P220 AN20 TIOA6	F	General-purpose I/O port A/D converter analog 20 input pin Base timer ch.6 TIOA output pin
190	162	P221 AN21 TIOB6	F	General-purpose I/O port A/D converter analog 21 input pin Base timer ch.6 TIOB input pin
191	163	P222 AN22 TIOA7	F	General-purpose I/O port A/D converter analog 22 input pin Base timer ch.7 TIOA I/O pin
192	164	P223 AN23 TIOB7	F	General-purpose I/O port A/D converter analog 23 input pin Base timer ch.7 TIOB input pin
195	167	P224 AN24	F	General-purpose I/O port A/D converter analog 24 input pin
196	168	P225 AN25 SIN4 INT7	F	General-purpose I/O port A/D converter analog 25 input pin Multi-function serial interface ch.4 serial data input pin INT7 external interrupt input pin
197	169	P226 AN26 SOT4 IN0	F	General-purpose I/O port A/D converter analog 26 input pin Multi-function serial interface ch.4 serial data output pin 16-bit input capture ch.0 external pulse input pin
198	170	P227 AN27 SCK4 IN1	F	General-purpose I/O port A/D converter analog 27 input pin Multi-function serial interface ch.4 clock I/O pin 16-bit input capture ch.1 external pulse input pin
199	171	P228 AN28 SCS40 IN2	F	General-purpose I/O port A/D converter analog 28 input pin Multi-function serial interface ch.4 serial chip select 0 I/O pin 16-bit input capture ch.2 external pulse input pin
200	172	P229 AN29 SCS41 IN3	F	General-purpose I/O port A/D converter analog 29 input pin Multi-function serial interface ch.4 serial chip select 1 I/O pin 16-bit input capture ch.3 external pulse input pin
201	173	P230 AN30 SCS42 IN4	F	General-purpose I/O port A/D converter analog 30 input pin Multi-function serial interface ch.4 serial chip select 2 I/O pin 16-bit input capture ch.4 external pulse input pin
202	174	P231 AN31 SCS43 IN5	F	General-purpose I/O port A/D converter analog 31 input pin Multi-function serial interface ch.4 serial chip select 3 I/O pin 16-bit input capture ch.5 external pulse input pin
203	175	P300 ADTG0	E	General-purpose I/O port A/D converter external trigger input pin
204	-	P301 TIOA10 FRCK4	E	General-purpose I/O port Base timer ch.10 TIOA output pin 16-bit free-run timer ch.4 external clock input pin
205	-	P302 TIOB10 FRCK5	E	General-purpose I/O port Base timer ch.10 TIOB input pin 16-bit free-run timer ch.5 external clock input pin

Pin Number		Pin Name	I/O Circuit Type	Functions
208 Pin	176 Pin			
206	-	P303 TIOA11 FRCK6	E	General-purpose I/O port Base timer ch.11 TIOA I/O pin 16-bit free-run timer ch.6 external clock input pin
207	-	P304 TIOB11 FRCK7	E	General-purpose I/O port Base timer ch.11 TIOB input pin 16-bit free-run timer ch.7 external clock input pin
18	14	AVRH0	-	4ch sample-hold A/D converter unit0 upper limit reference voltage
19	15	AVRL0	-	4ch sample-hold A/D converter unit0 lower limit reference voltage
20	16	AVR0	-	4ch sample-hold A/D converter unit0 reference voltage
21	17	AVSS0	-	4ch sample-hold A/D converter unit0 analog GND
22	18	AVCC0	-	4ch sample-hold A/D converter unit0 analog power supply
134	115	AVCC1	-	4ch sample-hold A/D converter unit1 analog power supply
135	116	AVSS1	-	4ch sample-hold A/D converter unit1 analog GND
136	117	AVR1	-	4ch sample-hold A/D converter unit1 reference voltage
137	118	AVRL1	-	4ch sample-hold A/D converter unit1 lower limit reference voltage
138	119	AVRH1	-	4ch sample-hold A/D converter unit1 upper limit reference voltage
38	34	RVRH0	-	*1
39	35	RVRL0	-	*2
40	36	RVR0	-	*2
41	37	RVSS0	-	*2
42	38	RVCC0	-	*1
114	95	RVCC1	-	*1
115	96	RVSS1	-	*2
116	97	RVR1	-	*2
117	98	RVRL1	-	*2
118	99	RVRH1	-	*1
181	153	AVCC2	-	A/D converter analog power supply
182	154	AVSS2	-	A/D converter analog GND
183	155	AVRL2	-	A/D converter lower limit reference voltage
184	156	AVRH2	-	A/D converter upper limit reference voltage
2 54 68 103 155 172 193	2 56 87 131 144 165	VCC12	-	1.2V power supply
12 52 105 144 157 208	44 89 133 176	VCC5	-	5.0V power supply



Pin Number		Pin Name	I/O Circuit Type	Functions
208 Pin	176 Pin			
1		VSS	-	GND
13	1			
53	45			
69	57			
83	71			
104	88			
143	132			
156	143			
171	166			
194				

\*1: The part number without RDC does not use this pin. Connect it with the VCC5 pin.

\*2: The part number without RDC does not use this pin. Connect it with the VSS pin.

## 6. I/O Circuit Type

Table 6-1 I/O Circuit Type

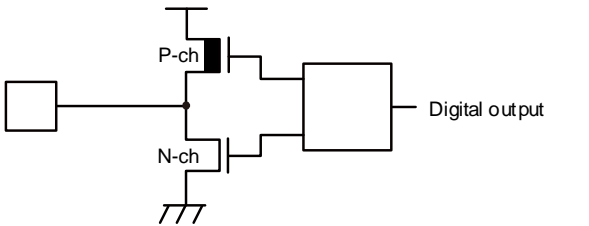
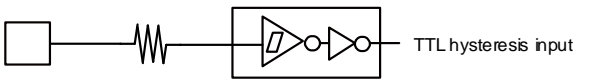
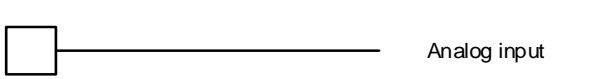
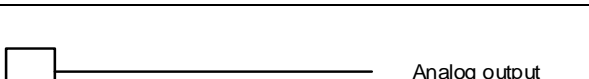
Type	Circuit	Remarks
A		<ul style="list-style-type: none"> <li>- Oscillation feedback resistor: Approx. 1 M<math>\Omega</math></li> </ul>
B		<ul style="list-style-type: none"> <li>- CMOS hysteresis input</li> <li>- With 50 k<math>\Omega</math> pull-up resistor</li> </ul>
C		<ul style="list-style-type: none"> <li>- Schmitt input</li> <li>- With high withstand voltage control</li> </ul>
D		<ul style="list-style-type: none"> <li>- CMOS level output</li> <li><math>I_{OH}=-1/-2\text{mA}</math>, <math>I_{OL}=1/2\text{mA}</math></li> </ul>
E		<ul style="list-style-type: none"> <li>- General-purpose I/O port</li> <li>- CMOS level output</li> <li><math>I_{OH}=-1/-2\text{mA}</math>, <math>I_{OL}=1/2\text{mA}</math></li> <li>- With 50 k<math>\Omega</math> pull-up resistor</li> <li>- CMOS hysteresis input (0.7Vcc/0.3Vcc)</li> <li>- Automotive input (0.8Vcc/0.5Vcc)</li> </ul>





Type	Circuit	Remarks
F		<ul style="list-style-type: none"> <li>- With Analog input, General-purpose I/O port</li> <li>- CMOS level output</li> <li><math>I_{OH}=-1/-2\text{mA}</math>, <math>I_{OL}=1/2\text{mA}</math></li> <li>- With 50 k<math>\Omega</math> pull-up resistor</li> <li>- CMOS hysteresis input (0.7Vcc/0.3Vcc)</li> </ul> <p>During standby, the input value retains the previous value.</p> <ul style="list-style-type: none"> <li>- Automotive input (0.8Vcc/0.5Vcc)</li> </ul> <p>During standby, the input value retains the previous value.</p>
G		<ul style="list-style-type: none"> <li>- With Analog output, General-purpose I/O port</li> <li>- CMOS level output</li> <li><math>I_{OH}=-1/-2\text{mA}</math>, <math>I_{OL}=1/2\text{mA}</math></li> <li>- With 50 k<math>\Omega</math> pull-up resistor</li> <li>- CMOS hysteresis input (0.7Vcc/0.3Vcc)</li> </ul> <p>During standby, the input value retains the previous value.</p> <ul style="list-style-type: none"> <li>- Automotive input (0.8Vcc/0.5Vcc)</li> </ul> <p>During standby, the input value retains the previous value.</p>
H		<ul style="list-style-type: none"> <li>- General-purpose I/O port</li> <li>- CMOS level output</li> <li><math>I_{OH}=-1/-2/-4\text{mA}</math>, <math>I_{OL}=1/2/4\text{mA}</math></li> <li>- With 50 k<math>\Omega</math> pull-up resistor</li> <li>- FlexRay input (0.7Vcc/0.3Vcc)</li> </ul> <p>During standby, the input value retains the previous value.</p> <ul style="list-style-type: none"> <li>- Automotive input (0.8Vcc/0.5Vcc)</li> </ul> <p>During standby, the input value retains the previous value.</p>



Type	Circuit	Remarks
I		<ul style="list-style-type: none"><li>- CMOS level output</li><li><math>I_{OH}=-5mA, I_{OL}=5mA</math></li></ul>
J		<ul style="list-style-type: none"><li>- TTL hysteresis input (2V/0.8V)</li></ul>
K		<ul style="list-style-type: none"><li>- Analog input</li></ul>
L		<ul style="list-style-type: none"><li>- Analog output</li></ul>



## 7. Memory Map

This section shows memory map of MB9D560 series.

**Table 7-1 Memory Map**

Address		Block	
Start	End	Overview	Function
0x0000_0000	64KB: 0x0000_FFFF 96KB: 0x0001_7FFF 128KB: 0x0001_FFFF	Memory (Each CPU exclusive space)	TCRAM
0x0002_0000	0x007F_FFFF		Reserved
0x0080_0000	512KB: 0x0087_FFFF 768KB: 0x008B_FFFF 1024KB: 0x008F_FFFF		TCFLASH large sector area (TCM connection)
0x0090_0000	0x00FD_FFFF		Reserved
0x00FE_0000	0x00FF_FFFF		TCFLASH small sector area (TCM connection)
0x0100_0000	512KB: 0x0107_FFFF 768KB: 0x010B_FFFF 1024KB: 0x010F_FFFF		TCFLASH large sector area (AXI connection)
0x0110_0000	0x01FD_FFFF		Reserved
0x01FE_0000	0x01FF_FFFF		TCFLASH small sector area (AXI connection)
0x0200_0000	0x027F_FFFF	Memory (Common space)	Reserved
0x0280_0000	0x0280_0FFF		EAM
0x0280_1000	0x03FF_FFFF		Reserved
0x0400_0000	64KB: 0x0400_FFFF 96KB: 0x0401_7FFF 128KB: 0x0401_FFFF		CPU0 space TCRAM
0x0402_0000	0x047F_FFFF		Reserved
0x0480_0000	512KB: 0x0487_FFFF 768KB: 0x048B_FFFF 1024KB: 0x048F_FFFF		CPU0 space TCFLASH large sector area (TCM connection)
0x0490_0000	0x04FD_FFFF		Reserved
0x04FE_0000	0x04FF_FFFF		CPU0 space TCFLASH small sector area (TCM connection)
0x0500_0000	512KB: 0x0507_FFFF 768KB: 0x050B_FFFF 1024KB: 0x050F_FFFF		CPU0 space TCFLASH large sector area (AXI connection)
0x0510_0000	0x05FD_FFFF		Reserved
0x05FE_0000	0x05FF_FFFF		CPU0 space TCFLASH small sector area (AXI connection)
0x0600_0000	64KB: 0x0600_FFFF 96KB: 0x0601_7FFF 128KB: 0x0601_FFFF		CPU1 space TCRAM
0x0602_0000	0x067F_FFFF		Reserved

Address		Block	
Start	End	Overview	Function
0x0680_0000	512KB: 0x0687_FFFF 768KB: 0x068B_FFFF 1024KB: 0x068F_FFFF		CPU1 space TCFLASH large sector area (TCM connection)
0x0690_0000	0x06FD_FFFF		Reserved
0x06FE_0000	0x06FF_FFFF		CPU1 space TCFLASH small sector area (TCM connection)
0x0700_0000	512KB: 0x0707_FFFF 768KB: 0x070B_FFFF 1024KB: 0x070F_FFFF	Memory (Common space)	CPU1 space TCFLASH large sector area (AXI connection)
0x0710_0000	0x07FD_FFFF		Reserved
0x07FE_0000	0x07FF_FFFF		CPU1 space TCFLASH small sector area (AXI connection)
0x0800_0000	0x09FF_FFFF		Reserved
0x0A00_0000	0x0BFF_FFFF		Reserved
0x0C00_0000	0x0DFF_FFFF		Reserved
0x0E00_0000	0x0E00_FFFF		WorkFLASH0 mirror area 1
0x0E01_0000	0x0E01_FFFF		WorkFLASH1 mirror area 1
0x0E02_0000	0x0E0F_FFFF		Reserved
0x0E10_0000	0x0E10_FFFF		WorkFLASH0 Reserved mirror area 2
0x0E11_0000	0x0E11_FFFF		WorkFLASH1 Reserved mirror area 2
0x0E12_0000	0x0E1F_FFFF		Reserved
0x0E20_0000	0x0E20_FFFF		WorkFLASH0 mirror area 3
0x0E21_0000	0x0E21_FFFF		WorkFLASH1 mirror area 3
0x0E22_0000	0x0E2F_FFFF		Reserved
0x0E30_0000	0x0FFF_FFFF		Reserved
0x1000_0000	0x1FFF_FFFF	Reserved	Reserved
0x2000_0000	0x2FFF_FFFF	Reserved	Reserved
0x3000_0000	0x3FFF_FFFF	Reserved	Reserved
0x4000_0000	0x7FFF_FFFF	Reserved	Reserved
0x8000_0000	0x9FFF_FFFF	Reserved	Reserved
0xA000_0000	0xA1FF_FFFF	Bit band alias area	Reserved
0xA200_0000	0xA27F_FFFF		Bit band alias area (Memory & Config Group)
0xA280_0000	0xA2FF_FFFF		Reserved
0xA300_0000	0xA37F_FFFF		Bit band alias area (MCU Config Group)
0xA380_0000	0xA47F_FFFF		Bit band alias area (Common Peripheral Group)
0xA480_0000	0xA7FF_FFFF		Reserved



Address		Block	
Start	End	Overview	Function
0xA800_0000	0xA87F_FFFF		Bit band alias area (Application Specific Peripheral Group A)
0xA880_0000	0xA8FF_FFFF		Bit band alias area (Application Specific Peripheral Group B)
0xA900_0000	0xAFFF_FFFF		Reserved
0xB000_0000	0xBFFF_FFFF	I/O area (bit band area)	I/O
0xC000_0000	0xEFFF_FFFF	Reserved	Reserved
0xF000_0000	0xFFFFE_DFFF	BootROM area	Reserved
0xFFFFE_E000	0xFFFFE_FFFF		Error Config
0xFFFF_0000	0xFFFF_FFFF		BootROM

**Notes:**

- Memory (each CPU specified area) define memory area for each CPU specified. Master of exception CPU cannot access. (Reserved area ) If another master access, can access from same area of CPU0/ CPU1.
- Reserved area access cause bus error.
- However, following access of reserved area will be not bus error.
  - 0x0090\_0000 to 0x00FD\_FFFF
  - 0x0110\_0000 to 0x01FD\_FFFF
  - 0x0490\_0000 to 0x04FD\_FFFF
  - 0x0510\_0000 to 0x05FD\_FFFF
  - 0x0690\_0000 to 0x06FD\_FFFF
  - 0x0710\_0000 to 0x07FD\_FFFF
  - 0x1000\_0000 to 0x1FFF\_FFFF
  - 0x2000\_0000 to 0x2FFF\_FFFF
- The following area should be set device attribution or strongly ordered attribution as core access.
  1. I/O area
  2. Bit band alias area
  3. Error Config (BootROM area )
  4. WorkFLASH (when program)
  5. TCFLASH (when program)

About device attribute and Strongly Ordered attribute, see "ARM®Architecture Reference Manual ARM®v7-A and ARM®v7-R edition (ARM DDI 0406B)".
- TCFLASH has a TCM-connected region and an AXI-connected region. AXI-connected region is dedicated for flash memory programming/erasing. When read operation in user mode, use TCM-connected region.

## 8. I/O Map

This section explains I/O map of MB9D560 series.

**Table 8-1 I/O Address Map (HPM, etc.)**

Address		Area	
Start	End	Overview	Function
0xB000_0000	0xB03F_FFFF	Reserved	Reserved

**Table 8-2 I/O Address Map (Memory & Config Group)**

Address		Area	
Start	End	Overview	Function
0xB040_0000	0xB040_0FFF	Memory & Config Group	IRC0
0xB040_1000	0xB040_1FFF		IRC1
0xB040_2000	0xB040_6FFF		Reserved
0xB040_7000	0xB040_73FF		NMI Distributor
0xB040_7400	0xB040_7FFF		Reserved
0xB040_8000	0xB040_83FF		TPU0
0xB040_8400	0xB040_8FFF		Reserved
0xB040_9000	0xB040_93FF		TPU1
0xB040_9400	0xB040_FFFF		Reserved
0xB041_0000	0xB041_03FF		TCRAM0 IF
0xB041_0400	0xB041_07FF		TCRAM1 IF
0xB041_0800	0xB041_0FFF		Reserved
0xB041_1000	0xB041_13FF		TCFLASH0 IF
0xB041_1400	0xB041_17FF		TCFLASH1 IF
0xB041_1800	0xB041_1FFF		Reserved
0xB041_2000	0xB041_23FF		WorkFLASH0 IF
0xB041_2400	0xB041_27FF		WorkFLASH1 IF
0xB041_2800	0xB041_4FFF		Reserved
0xB041_5000	0xB041_5FFF		IPCU
0xB041_6000	0xB04F_7FFF		Reserved



Table 8-3 I/O Address Map (Debug Group)

Address		Area	
Start	End	Overview	Function
0xB050_0000	0xB050_0FFF	Debug Group	DAPROM
0xB050_1000	0xB050_1FFF		ETB
0xB050_2000	0xB050_2FFF		CTI4
0xB050_3000	0xB050_3FFF		TPIU
0xB050_4000	0xB050_4FFF		TRACE_FUNNEL
0xB050_5000	0xB057_FFFF		Reserved
0xB058_0000	0xB058_0FFF		CORTEXROM0
0xB058_1000	0xB058_FFFF		Reserved
0xB059_0000	0xB059_0FFF		CORE0
0xB059_1000	0xB059_1FFF		Reserved
0xB059_2000	0xB059_2FFF		CORE1
0xB059_3000	0xB059_7FFF		Reserved
0xB059_8000	0xB059_8FFF		CTI0
0xB059_9000	0xB059_9FFF		CTI1
0xB059_A000	0xB059_BFFF		Reserved
0xB059_C000	0xB059_CFFF		ETM0
0xB059_D000	0xB059_DFFF		ETM1
0xB059_E000	0xB05F_FFFF		Reserved

Table 8-4 I/O Address Map (MCU Config Group)

Address		Area	
Start	End	Overview	Function
0xB060_0000	0xB060_07FF	MCU Config Group	SYSC
0xB060_0800	0xB060_0FFF		MODEC
0xB060_1000	0xB060_7FFF		Reserved
0xB060_8000	0xB060_83FF		SW-WDT0
0xB060_8400	0xB060_8FFF		Reserved
0xB060_9000	0xB060_93FF		SW-WDT1
0xB060_9400	0xB060_BFFF		Reserved
0xB060_C000	0xB060_C3FF		HW-WDT
0xB060_C400	0xB061_FFFF		Reserved
0xB062_0000	0xB062_03FF		EXT-IRC
0xB062_0400	0xB06F_FFFF		Reserved

**Table 8-5 I/O Address Map (Common Peripheral Group)**

Address		Area	
Start	End	Overview	Function
0xB070_0000	0xB070_3FFF	Common Peripheral Group (AHB32)	DMAC
0xB070_4000	0xB070_FFFF		Reserved
0xB071_0000	0xB071_0FFF		MPU_DMA
0xB071_1000	0xB071_7FFF		Reserved
0xB071_8000	0xB071_87FF		CRC (ch.0,1)
0xB071_8800	0xB071_FFFF		Reserved
0xB072_0000	0xB072_0BFF		CAN (ch.0 to 2)
0xB072_0C00	0xB072_7FFF		Reserved
0xB072_8000	0xB072_83FF		CAN Prescaler
0xB072_8400	0xB072_FFFF		Reserved
0xB073_0000	0xB073_03FF		CR Calibration
0xB073_0400	0xB073_7FFF		Reserved
0xB073_8000	0xB073_8FFF		GPIO
0xB073_9000	0xB073_FFFF		Reserved
0xB074_0000	0xB074_3FFF		PPC
0xB074_4000	0xB074_7FFF		Reserved
0xB074_8000	0xB074_8FFF		RIC
0xB074_9000	0xB07F_FFFF		Reserved
0xB080_0000	0xB080_13FF	Common Peripheral Group (APB)	MFS (ch.0 to 4)
0xB080_1400	0xB080_7FFF		Reserved
0xB080_8000	0xB080_AFFF		Base Timer (ch.0 to 11)
0xB080_B000	0xB081_FFFF		Reserved
0xB082_0000	0xB082_13FF		32bit FRT (ch.0 to 4)
0xB082_1400	0xB082_7FFF		Reserved
0xB082_8000	0xB080_8BFF		32bit ICU (ch.0 to 2)
0xB082_8C00	0xB08F_FFFF		Reserved





**Table 8-6 I/O Address Map (Product Specified Peripheral Bus)**

Address		Area	
Start	End	Overview	Function
0xB090_0000	0xB0FF_FFFF	Reserved	Reserved
0xB100_0000	0xB100_00FF	Application Specific Peripheral Group A (AHB-32)	16-bit FRT (ch.6 to 11)
0xB100_0100	0xB100_01FF		16-bit OCU (ch.6 to 11)
0xB100_0200	0xB100_02FF		16-bit ICU (ch.4 to 7)
0xB100_0300	0xB100_03FF		4ch-SH ADC (unit1)
0xB100_0400	0xB100_04FF		WFG (ch.6 to 11)
0xB100_0500	0xB100_05FF		UDC (ch.2 to 3)
0xB100_0600	0xB100_07FF		Reserved
0xB100_0800	0xB100_09FF		MVA (unit1)
0xB100_0A00	0xB100_0BFF		Reserved
0xB100_0C00	0xB100_0CFF		RDC (unit1)
0xB100_0D00	0xB100_0DFF		DAC (ch.1)
0xB100_0E00	0xB100_0FFF		Reserved
0xB101_0000	0xB101_00FF	Application Specific Peripheral Group A (APB)	16-bit FRT (ch.12 to 17)
0xB101_0100	0xB101_01FF		16-bit OCU (ch.12 to 23)
0xB101_0200	0xB101_02FF		16-bit ICU (ch.7 to 13)
0xB101_0300	0xB101_03FF		Reserved
0xB101_0400	0xB101_05FF		12-bit ADC (ch.0 to 31)
0xB101_0600	0xB101_06FF		WFG(ch.12 to 23)
0xB101_0700	0xB101_0FFF		Reserved
0xB101_1000	0xB101_2FFF		Other (WFG)
0xB101_3000	0xB101_3FFF		Other (ADC, CSV)
0xB101_4000	0xB1FF_FFFF	Reserved	Reserved
0xB200_0000	0xB200_00FF	Application Specific Peripheral Group B (AHB-32)	16-bit FRT (ch.0 to 5)
0xB200_0100	0xB200_01FF		16-bit OCU (ch.0 to 5)
0xB200_0200	0xB200_02FF		16-bit ICU (ch.0 to 3)
0xB200_0300	0xB200_03FF		4ch-SH ADC (unit0)
0xB200_0400	0xB200_04FF		WFG (ch.0 to 5)
0xB200_0500	0xB200_05FF		UDC (ch.0 to 1)
0xB200_0600	0xB200_07FF		Reserved
0xB200_0800	0xB200_09FF		MVA (unit0)
0xB200_0A00	0xB200_0BFF		Reserved
0xB200_0C00	0xB200_0CFF		RDC (unit0)
0xB200_0D00	0xB200_0DFF		DAC (ch.0)
0xB200_0E00	0xB200_0FFF		Reserved
0xB200_1000	0xB200_17FF		FlexRay (ch.A/ch.B)
0xB200_1800	0xB200_FFFF		Reserved
0xB201_0000	0xB201_00FF	Application Specific Peripheral Group B (R-Bus)	FlexRay/RDC clock control
0xB201_0100	0xB201_01FF		Clock monitor
0xB201_0200	0xBFFF_FFFF	Reserved	Reserved

**Table 8-7 I/O Address Map (Error Config)**

Address		Area	
Start	End	Overview	Function
0xFFFE_E000	0xFFFE_E3FF	Error Config	IRC0 (NMIVASBR)
0xFFFE_E400	0xFFFE_E7FF		IRC1 (NMIVASBR)
0xFFFE_E800	0xFFFE_F7FF		Reserved
0xFFFE_F800	0xFFFE_FBFF		IRC (NMIVASBR) mirror*
0xFFFE_FC00	0xFFFE_FFFF		BootROM IF

\*: CPU0 is IRC0, CPU1 is IRC1 able to access this area. The master of excepted CPU is reserved area.

**Notes:**

- I/O address map shows maximum area for possibility. It depends on functions. The detail information, see each address map.
- It causes bus error to access to reserved area. However, following reserved area access is not generation of bus error.
  - 0xB018\_0000 to 0xB018\_03FF
  - 0xB05C\_0000 to 0xB05C\_0FFF
  - 0xB05E\_0000 to 0xB05E\_03FF
  - 0xB05E\_0400 to 0xB05E\_07FF
  - 0xB05E\_0800 to 0xB05E\_0BFF
  - 0xB05E\_0C00 to 0xB05E\_0FFF



## 9. Legend

This section describes the legend used throughout this manual.

### 9.1. Register Attribute

#### 9.1.1. Read/Write Attribute (R/W Attribute)

Table 9-1 R/W Attribute

R/W Attribute	Description
R	Readable
R0	Read value is always 0
R1	Read value is always 1
RX	Read access returns unknown value
W	Writable
W0	Write value must be 0
W1	Write value must be 1
WX	Write invalid (write operation has no influence)
/ (slash mark)	Writable and readable (read value is equal to write value)
, (comma)	Different behavior in read and write (read value is different with write value)

#### Example

R/W : Readable and writable [read value is equal to write value]

R,W0 : Readable, Write value must be 0 [read value is different with write value]

#### Notes:

- In case factors that influences read value are following, R/W attribute is described with / (slash mark).
- Bit set by bit set register operation
- Bit clear by bit clear register operation



9.1.2.    Protection Attribute

Table 9-2 Protection Attribute

Protection Attribute	Description
RP	Readable in privilege mode
WP	Writeable in privilege mode
WS	Writeable, with protected sequence
WPS	Writeable in privilege mode, with protected sequence



## 10. Abbreviations

This section explains abbreviations used in the document

**Table 10-1 List of Abbreviations**

Index	Abbreviations	Meaning
A	AHB	Advanced High-performance Bus
	AMBA™	Advanced Microcontroller Bus Architecture
	APB	Advanced Peripheral Bus
	ATB	AMBA Trace Bus
	ATCM	TCM-A port
	AXI	Advanced eXtensible Interface
B	BBU	Bit Banding Unit
	BDR	Boot Description Record
	B0TCM	TCM-B0 port
	B1TCM	TCM-B1 port
C	CAN	Controller Area Network
	CD	Clock Domain
	CPU	Central Processing Unit
	CR	CR Oscillator
	CRC	Cyclic Redundancy Check
	CSV	Clock Supervisor
D	DAP	Debug Access Port
	DED	Double Error Detection
	DMA	Direct Memory Access
	DMAC	DMA Controller
	EAM	Exclusive Access Memory
E	ECC	Error Checking Code
	ETM	Embedded Trace Macro
	EXT-IRC	External Interrupt Controller
	FIQ	Fast Interrupt Request
F	FPU	Floating Point Unit
	FRT	Free-run Timer
	GPIO	General Purpose Input / Output
H	HPM	High Performance Matrix
	HW-WDT	Hardware Watchdog Timer
I	ICU	Input Capture Unit
	IPCU	Inter-Processor Communication Unit
	IRC	Interrupt Controller
	IRQ	Interrupt Request
	ISR	Interrupt Service Routine
J	JTAG	Joint Test Action Group
L	LLPP	Low Latency Peripheral Port
	LVD	Low-voltage Detector
M	MCU	Microcontroller Unit
	MFS	Multi-function Serial Interface
	MVA	Motor Vector Operation Accelerator
N	NF	Noise Filter
	NMI	Non-maskable Interrupt
O	OSC	Oscillator
	OCU	Output Compare Unit

Index	Abbreviations	Meaning
P	PLL	Phase Locked Loop
	PONR	Power-ON Reset
	PPC	Port Pin Configuration
	PSS	Power Saving State
R	RAM	Random Access Memory
	RDC	Resolver digital Converter
	RIC	Resource Input Configuration
	ROM	Read Only Memory
	RUN	Run State
S	SCT	Source Clock Timer
	SDR	Security Description Record
	SEC	Single Error Correction
	SRAM	Static RAM
	SW-WDT	Software Watchdog Timer
	SYSC	System Controller
T	TCFLASH	TCM Flash Memory
	TCM	Tightly Coupled Memory
	TCRAM	TCM SRAM
	TPU	Timing Protection Unit
U	UDC	Up/down Counter
V	VIC	Vectored Interrupt Controller
W	WDR	Watchdog Description Record
	WDT	Watchdog Timer
	WFG	Waveform Generator
	WorkFLASH	Work Flash Memory





## CHAPTER 2: CPU

This chapter provides an overview of the CPU and related notes.

---

1. Overview
2. Notes





## 1. Overview

This section explains the features of the CPU.

### Features

- The processor is Cortex-R5F™ Split/Lock configuration split mode (2 CPU) processor.
- The CPU is equipped with a memory protection unit (MPU) with 16 regions.
- The CPU is equipped with a double precision floating-point unit (FPU).
- TCFLASHs are connected to the ATCMs of each CPU.
- TCRAMs are connected to the BTCMs of each CPU.
- The ATCM and BTCM support 32-bit ECC.
- B0TCM and B1TCM are used in interleaving.
- Each CPU is equipped with the 64-bit AXI Master and Slave interfaces.
- External master can access to TCM via AXI slave interface.
- Each CPU is equipped with the 32-bit AXI master interface for the low latency peripheral port (LLPP).
- Each CPU is equipped with a VIC port, which enables direct retrieval of interrupt vector addresses.
- For details, see the Cortex™-R5 Revision:r1p2 Technical Reference Manual (ARM DDI 0460D), an ARM manual.

## 2. Notes

This section provides notes on the CPU.

### (1) Cache

The CPU is not equipped with instruction cache and data cache.

### (2) Low Latency Peripheral Port

The low latency peripheral port is used as the port for accessing peripherals.

The internal register Normal AXI Peripheral Interface Region, Virtual AXI Peripheral Interface Region, AHB Peripheral Interface Region: En bit must not be changed.

### (3) WFE Instruction

The WFE instruction must not be used.

When taking synchronization between cores, use the inter-processor communication unit.

### (4) VIC Port

Interrupts (IRQs) are processed through the VIC port.

To enable the VIC port, the System Control Register:VE, which is an internal core register, is set to 1 by the BootROM software.

If the VIC port is disabled, interrupts cannot be processed properly, so the VIC port must not be disabled.





## CHAPTER 3: Operation Mode

This chapter explains the operation modes.

---

1. Overview
2. Configuration
3. Operation
4. Register



## 1. Overview

This section provides an overview of the operation mode.

The mode controller determines the device operation mode at the release of the reset. This device has the following operation modes.

### **(1) User Mode**

User Mode executes the user program from the Flash.

### **(2) Board Mode**

Board Mode has below sub-modes.

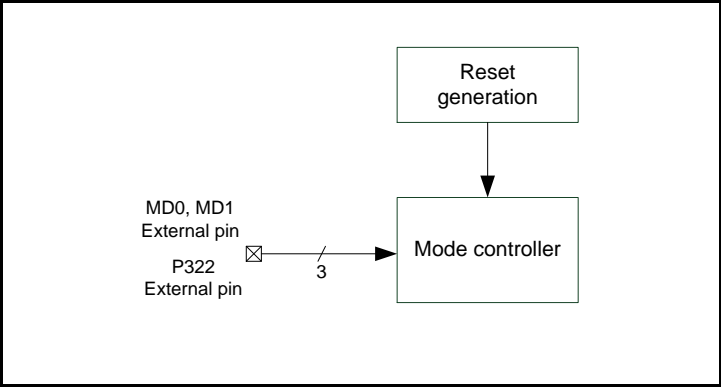
#### **a) Serial Writer Mode**

This mode is used to program the Flash using a serial writer.

2. Configuration

The following figure is a block diagram of the mode controller.

Figure 2-1 Block Diagram of Mode Controller



### 3. Operation

This section explains the operation of the operation mode.

#### 3.1. Pin Settings

This section shows pin settings.

Table 3-1 Pin Settings

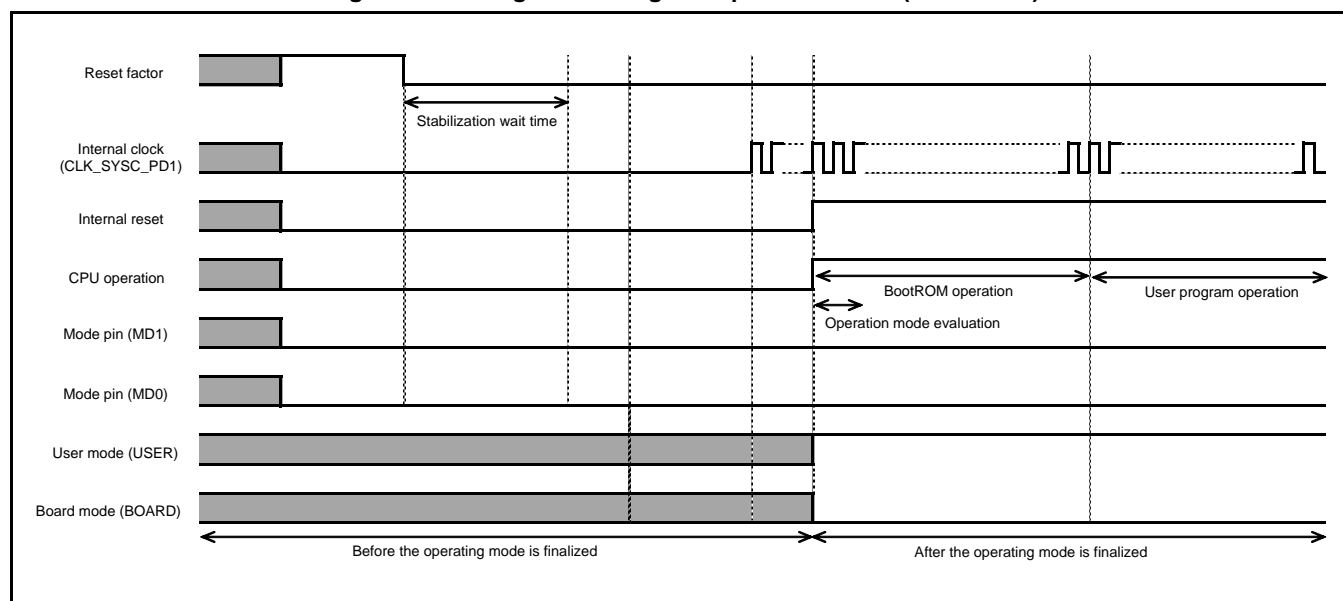
Operation Mode	MD1	MD0	P322
User mode	0	0	-
Serial writer mode	1	0	1

Setting other than above is prohibited.

#### 3.2. Fetching the Operation Mode

This section explains fetching the operation mode.

Figure 3-1 Timing of Fetching the Operation Mode (User Mode)



## 4. Register

This section lists the mode controller registers.

**Table 4-1 List of Mode Controller Registers**

Abbreviated Register Name	Register Name	See
MODEC_MODER	Mode Register	4.1





## 4.1. Mode Register (MODEC\_MODER)

Mode register (MODEC\_MODER) indicates states of the operation mode, the MD pin determined and CPU operation mode during the device start-up.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	USERMODE	BOARDMODE	Reserved					
ACCESS_TYPE	R,WX	R,WX	R0,WX					
PROT_TYPE	-							
INITIAL_VALUE	*	*	000000					

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved		MD1	MD0	Reserved			
ACCESS_TYPE	R0,WX		R,WX	R,WX	R0,WX			
PROT_TYPE	-							
INITIAL_VALUE	00		0	0	0000			

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved					CPUMD		
ACCESS_TYPE	R0,WX					R,WX		
PROT_TYPE	-							
INITIAL_VALUE	00000					*		

\*: Initial value depends on the status during the start-up.

### [bit31] USERMODE: User mode bit

- This bit indicates whether the operation mode is in user mode.
- Writing to this bit is no effect on operation.

Value	Description
0	Reserved
1	User mode

### [bit30] BOARDMODE: Board mode bit

- This bit indicates whether the operation mode is in board mode.
- Writing to this bit is no effect on operation.

Value	Description
0	Not Board mode
1	Board mode

### [bit29:14] Reserved: Reserved bits

**[bit13] MD1: Mode 1 bit**

- The value of current input data to MD1 pin can be read.
- Writing to this bit is no effect on operation.

**[bit12] MD0: Mode 0 bit**

- The value of current input data to MD0 pin can be read.
- Writing to this bit is no effect on operation.

**[bit11:3] Reserved: Reserved bits**

**[bit2:0] CPUMD[2:0]: CPU operation mode**

- These bits indicates CPU operation mode.
- Writing to these bits is no effect on operation.
- When changing CPU operation mode, update CPU operation mode setting bits (SYSC\_SPECPUFCFR:CPUMD[2:0]) and issue reset. After re-start, these bits are updated and device operates specified CPU operation mode.

Value	Description
000	Multi CPU mode (2CPU mode)
001	Reserved
010	Reserved
011	Reserved
100	Single CPU mode (1CPU0 mode)
101	Single CPU mode (1CPU1 mode)
110	Reserved
111	Reserved





## CHAPTER 4: Reset

This chapter explains resets.

---

1. Overview
2. Configuration
3. Operation
4. Registers



## 1. Overview

This product has the following reset factors. Depending on each factor, a reset is issued for initialization within the device.

The following table shows reset factors.

**Table 1-1 Reset Factor**

Reset Category	Reset Factor
Hardware reset	Power-on reset (PONR) RSTX pin input (RSTX) INITX pin input (INITX) Clock stop wait timeout (CKTOR) Hardware watchdog reset (HWDR) Software watchdog reset (SWDR) Internal power supply low-voltage detection (LVD12R) 5V external power supply low-voltage detection (LVD50R) Main clock supervisor reset (CSVMOR) PLL clock supervisor reset (CSVPR) Extended clock supervisor reset (CSVSSR) Profile error reset (PRFERR) Software trigger hardware reset (SHRST) nSRST pin input (SRSTX)
Software reset	Software reset (SRST)
Debugger reset	TRSTX pin input reset (TRSTX) Software debugger reset (SDBGST)

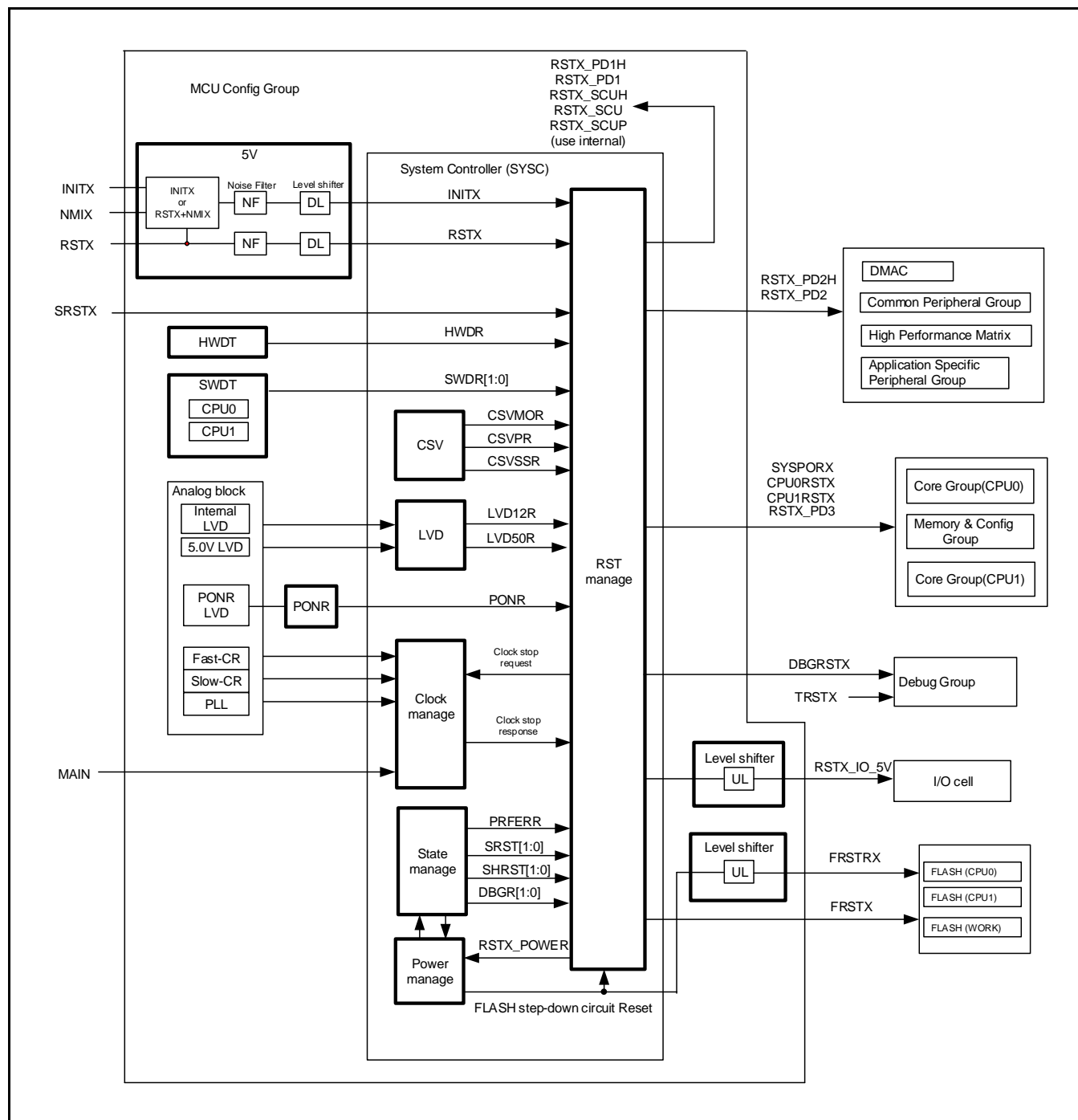
**Notes:**

- In the MB9D560, the INITX pin input (INITX) is replaced by a simultaneous input to the NMIX pin + the RSTX pin.
- In the MB9D560, the extended clock supervisor reset (CSVSSR) is replaced by a PLL clock supervisor reset for FlexRay/RDC.

## 2. Configuration

This section explains the configuration of reset.

Figure 2-1 Block Diagram of Reset System





The block diagram of reset control section shows following.

Figure 2-2 Block Diagram of RST Manage

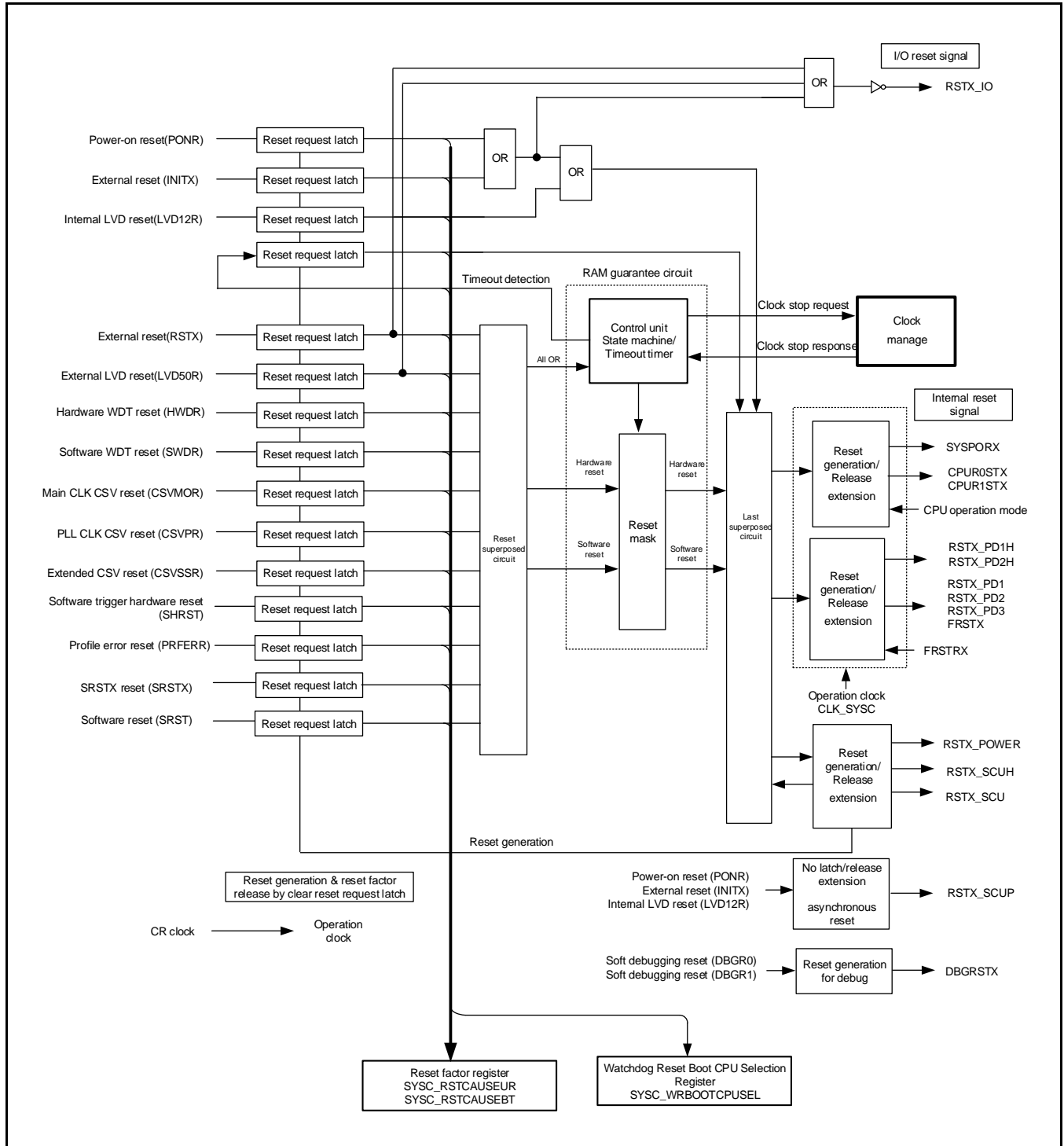


Figure 2-3 Block Diagram of PONR

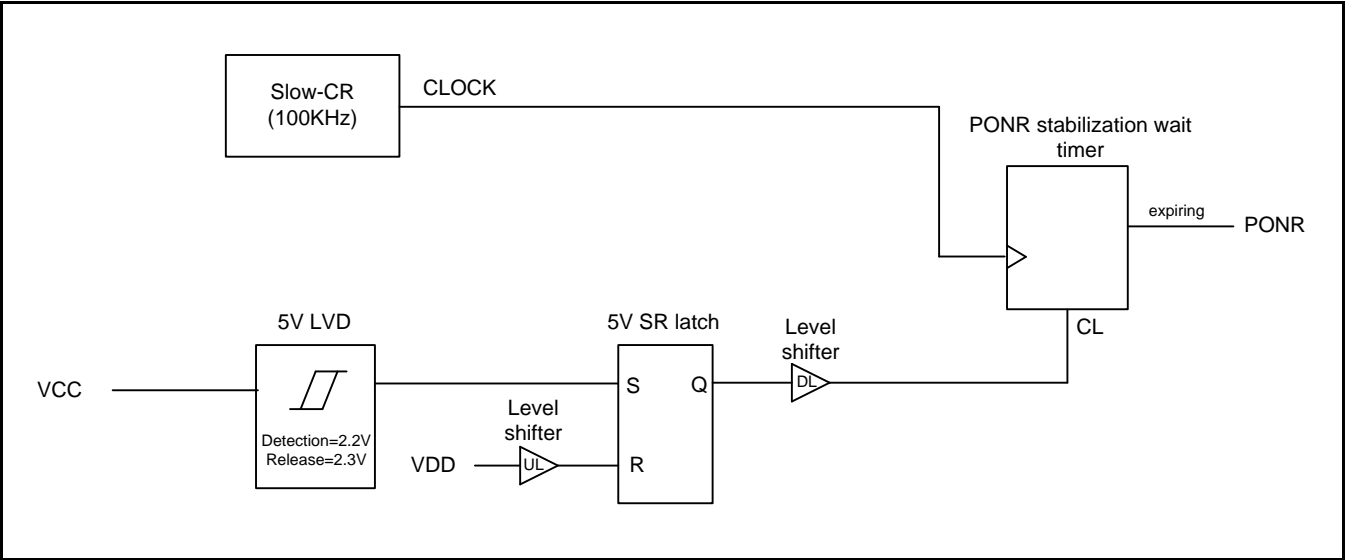
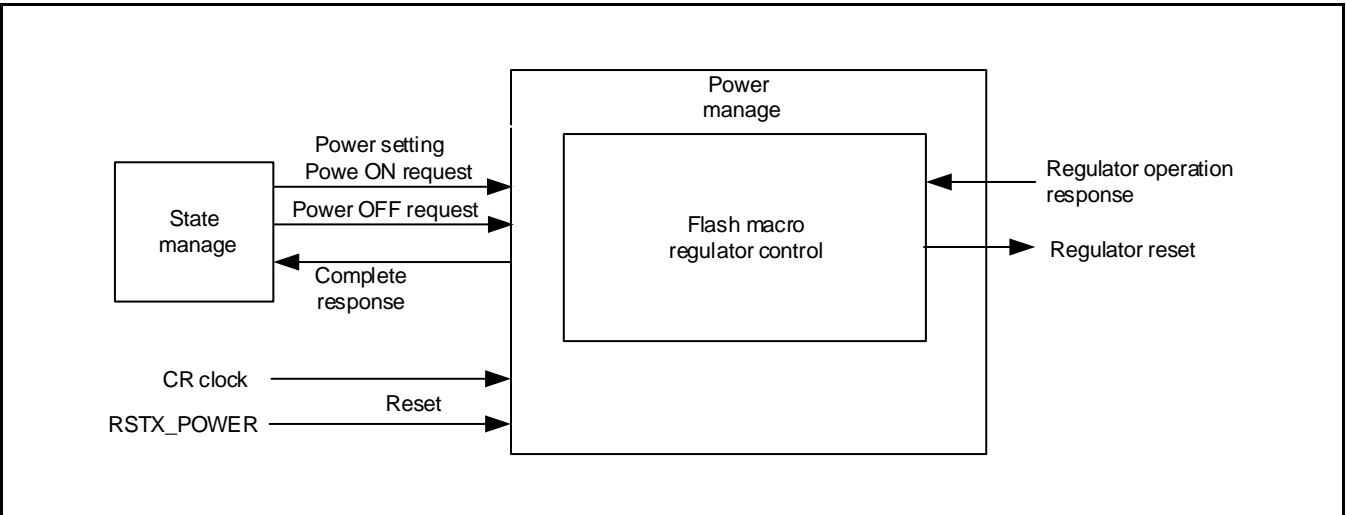


Figure 2-4 Block Diagram of POWER manage







Block diagram of STATE manage

See "CHAPTER: Low-power Consumption".

Block diagram of CSV

See "CHAPTER: Clock Supervisor".

Block diagram of LVD

See "CHAPTER: Low-voltage Detection".

Block diagram of HWDT

See "CHAPTER: Hardware Watchdog Timer".

Block diagram of SWDT

See "CHAPTER: Software Watchdog Timer".

### **3. Operation**

This section explains each operation of resets.

3.1. Reset Factor

3.2. Internal Reset of Device

3.3. Reset Sequence

3.4. Operations after Reset Release



## 3.1. Reset Factor

This section explains details of each reset factors.

- 3.1.1. Power-on Reset
- 3.1.2. Internal Power Supply Low-voltage Detection Reset
- 3.1.3. INITX Pin Input Reset
- 3.1.4. Clock Stop Wait Timeout Reset
- 3.1.5. External Power Supply Low-voltage Detection Reset
- 3.1.6. RSTX Pin Input Reset
- 3.1.7. Hardware Watchdog Reset
- 3.1.8. Software Watchdog Reset
- 3.1.9. Main Clock Supervisor Reset
- 3.1.10. PLL Clock Supervisor Reset
- 3.1.11. Extended Clock Supervisor Reset
- 3.1.12. Profile Error Reset
- 3.1.13. Software Trigger Hardware Reset
- 3.1.14. Software Reset
- 3.1.15. nSRST Pin Input Reset
- 3.1.16. TRSTX Pin Input Reset
- 3.1.17. Software Debugger Reset

### 3.1.1. Power-on Reset

This section explains power-on reset.

**Table 3-1 Power-on Reset**

Occurrence factor	This occurs when the low-voltage detector (LVD) which is monitoring the 5 V power supply status enters the detection state.
Release factor	1.2 V power is applied and the low-voltage detector (LVD) which is monitoring the 5 V power supply status is released. PONR is released after stabilization time by slow-CR clock.
Initialization target	All registers
Corresponding flag	SYSC_RSTCAUSEUR:PONR SYSC_RSTCAUSEBT:PONR (for BootROM)



### 3.1.2. Internal Power Supply Low-voltage Detection Reset

This section explains internal power supply low-voltage detection reset.

**Table 3-2 Internal Power Supply Low-voltage Detection Reset**

Occurrence factor	This occurs when the low-voltage detector (LVD) which is monitoring the 1.2 V power supply status enters the detection state.
Release factor	This is due to the release state of the low-voltage detector (LVD) which is monitoring the 1.2 V power supply status.
Initialization target	All registers except the following. <ul style="list-style-type: none"> <li>- PONR stabilization wait circuit</li> <li>- Reset factor registers</li> </ul>
Corresponding flag	SYSC_RSTCAUSEUR:LVD12R SYSC_RSTCAUSEBT:LVD12R (for BootROM)
Remarks	See Data Sheet for occurrence/release of internal power supply low-voltage detection.

### 3.1.3. INITX Pin Input Reset

This section explains INITX input reset.

**Table 3-3 INITX Pin Input Reset**

Occurrence factor	It is issued by inputting "L" level to INITX pin.
Release factor	It is released by inputting "H" level to INITX pin.
Initialization target	All registers except the following. <ul style="list-style-type: none"><li>- PONR stabilization wait circuit</li><li>- Reset factor registers</li></ul>
Corresponding flag	SYSC_RSTCAUSEUR:INITX SYSC_RSTCAUSEBT:INITX (for BootROM)
Remarks	For the noise filter removal width, see Data Sheet.

**Note:**

- In the MB9D560, the INITX pin input (INITX) is replaced by a simultaneous input to the NMIX pin + the RSTX pin. If an "L" is simultaneously input to both the NMIX pin + the RSTX pin, an INITX factor is generated, and the factor is released if a value other than "L" is simultaneously input.



### 3.1.4. Clock Stop Wait Timeout Reset

This section explains clock stop wait timeout reset.

**Table 3-4 Clock Stop Wait Timeout Reset**

Occurrence factor	This occurs when the clock stop wait timer expiration.
Release factor	This is automatically released after a reset is issued.
Initialization target	<p>All registers except the following.</p> <ul style="list-style-type: none"> <li>- PONR stabilization wait circuit</li> <li>- Fast-CR oscillation stabilization wait circuit</li> <li>- Slow-CR oscillation stabilization wait circuit</li> <li>- Post-reset RAM assurance control circuit</li> <li>- Reset factor registers</li> <li>- Watchdog Reset Boot CPU selection register</li> <li>- CPU control register (CPUMD bits)</li> <li>- Clock supervisor output enable register</li> <li>- Debugging system</li> <li>- POWER Manage</li> </ul>
Corresponding flag	<p>SYSC_RSTCAUSEUR:CKTOR</p> <p>SYSC_RSTCAUSEBT:CKTOR (for BootROM)</p>
Remarks	<p>CR × 1360 cycle (typ=170μs)</p> <p>Slow-CR clock stop response wait time is maximum value.</p> <p>Slow-CR × about 3 cycle = 30μs</p>

### 3.1.5. External Power Supply Low-voltage Detection Reset (5V)

This section explains external power supply low-voltage detection reset (5V).

**Table 3-5 External Power Supply Low-voltage Detection Reset (5V)**

Occurrence factor	This occurs when the low-voltage detector (LVD) which is monitoring the 5.0 V power supply status enters the detection state.
Release factor	This is due to the release state of the low-voltage detector (LVD) which is monitoring the 5.0 V power supply status.
Initialization target	All registers except the following. <ul style="list-style-type: none"><li>- PONR stabilization wait circuit</li><li>- Fast-CR oscillation stabilization wait circuit</li><li>- Slow-CR oscillation stabilization wait circuit</li><li>- Post-reset RAM assurance control circuit</li><li>- Reset factor registers</li><li>- Watchdog Reset Boot CPU selection register</li><li>- CPU control register (CPUMD bits)</li><li>- Clock supervisor output enable register</li><li>- Debugging system</li><li>- POWER Manage</li></ul>
Corresponding flag	SYSC_RSTCAUSEUR:LVD50R SYSC_RSTCAUSEBT:LVD50R (for BootROM)
Remarks	See Data Sheet for occurrence/release of external power supply low-voltage detection.





### 3.1.6. RSTX Pin Input Reset

This section explains RSTX pin input reset.

**Table 3-6 RSTX Pin Input Reset**

Occurrence factor	This occurs because the "L" level is input to the RSTX pin.
Release factor	This is released by input "H" level to the RSTX pin.
Initialization target	<p>All registers except the following.</p> <ul style="list-style-type: none"> <li>- PONR stabilization wait circuit</li> <li>- Fast-CR oscillation stabilization wait circuit</li> <li>- Slow-CR oscillation stabilization wait circuit</li> <li>- Post-reset RAM assurance control circuit</li> <li>- Reset factor registers</li> <li>- Watchdog Reset Boot CPU selection register</li> <li>- CPU control register (CPUMD bits)</li> <li>- Clock supervisor output enable register</li> <li>- Debugging system</li> <li>- POWER Manage</li> </ul>
Corresponding flag	<p>SYSC_RSTCAUSEUR:RSTX</p> <p>SYSC_RSTCAUSEBT:RSTX(for BootROM)</p>
Remarks	For the noise filter removal width, see Data Sheet.

### 3.1.7. Hardware Watchdog Reset

This section explains hardware watchdog reset.

**Table 3-7 Hardware Watchdog Reset**

Occurrence factor	A reset occurs when the CPU does not input a trigger within the specified window period for the hardware watchdog.
Release factor	This is automatically released after the reset is issued.
Initialization target	All registers except the following: <ul style="list-style-type: none"><li>- PONR oscillation stabilization wait circuit</li><li>- Fast-CR oscillation stabilization wait circuit</li><li>- Slow-CR oscillation stabilization wait circuit</li><li>- Post-reset RAM assurance control circuit</li><li>- Watchdog Reset Boot CPU selection register</li><li>- CPU control register (CPUMD bits)</li><li>- Clock supervisor output enable register</li><li>- Reset factor registers</li><li>- Debugging system</li></ul>
Corresponding flag	SYSC_RSTCAUSEUR:HWDR SYSC_RSTCAUSEBT:HWDR (for BootROM)



### 3.1.8. Software Watchdog Reset

This section explains software watchdog reset.

**Table 3-8 Software Watchdog Reset**

Occurrence factor	A reset occurs when the CPU does not input a trigger within the specified window period for the software watchdog.
Release factor	After reset is issued, it automatically releases it.
Initialization target	<p>All registers except the following.</p> <ul style="list-style-type: none"> <li>- PONR oscillation stabilization wait circuit</li> <li>- Fast-CR oscillation stabilization wait circuit</li> <li>- Slow-CR oscillation stabilization wait circuit</li> <li>- Post-reset RAM assurance control circuit</li> <li>- Reset factor registers</li> <li>- Watchdog Reset Boot CPU selection register</li> <li>- CPU control register (CPUMD bits)</li> <li>- Clock supervisor output enable register</li> <li>- Debugging system</li> <li>- POWER Manage</li> </ul>
Corresponding flag	<p>SYSC_RSTCAUSEUR:SWDRn  SYSC_RSTCAUSEBT:SWDRn (for BootROM)  (n=number of Software watchdog timers- 1)</p>
Remarks	A software watchdog is equipped per CPU. The access is restricted per CPU. When one CPU issues reset factor, the target registers is initialized.

### 3.1.9. Main Clock Supervisor Reset

This section explains main clock supervisor reset.

**Table 3-9 Main Clock Supervisor Reset**

Occurrence factor	This occurs when the CSV for monitoring the main clock detects a reset abnormality (reset occurrence condition).
Release factor	This is automatically released after a reset is issued.
Initialization target	All registers except the following. <ul style="list-style-type: none"><li>- PONR stabilization wait circuit</li><li>- Fast-CR oscillation stabilization wait circuit</li><li>- Slow-CR oscillation stabilization wait circuit</li><li>- Post-reset RAM assurance control circuit</li><li>- Reset factor registers</li><li>- Watchdog Reset Boot CPU selection register</li><li>- CPU control register (CPUMD bits)</li><li>- Clock supervisor output enable register</li><li>- Debugging system</li><li>- POWER Manage</li></ul>
Corresponding flag	SYSC_RSTCAUSEUR:CSVMOR SYSC_RSTCAUSEBT:CSVMOR (for BootROM)



### 3.1.10. PLL Clock Supervisor Reset

This section explains PLL clock supervisor reset.

**Table 3-10 PLL Clock Supervisor Reset**

Occurrence factor	This occurs when the CSV for monitoring the PLL clock detects a reset abnormality (reset occurrence condition).
Release factor	This is automatically released after a reset is issued.
Initialization target	<p>All registers except the following</p> <ul style="list-style-type: none"> <li>- PONR stabilization wait circuit</li> <li>- Fast-CR oscillation stabilization wait circuit</li> <li>- Slow-CR oscillation stabilization wait circuit</li> <li>- Post-reset RAM assurance control circuit</li> <li>- Reset factor registers</li> <li>- Watchdog Reset Boot CPU selection register</li> <li>- CPU control register (CPUMD bits)</li> <li>- Clock supervisor output enable register</li> <li>- Debugging system</li> <li>- POWER Manage</li> </ul>
Corresponding flag	<p>SYSC_RSTCAUSEUR:CSVPR</p> <p>SYSC_RSTCAUSEBT:CSVPR (for BootROM)</p>

### 3.1.11. Extended Clock Supervisor Reset

This section explains extended clock supervisor reset.

**Table 3-11 Extended Clock Supervisor Reset**

Occurrence factor	This occurs when the extended CSV for monitoring the PLL clock detects a reset abnormality (reset occurrence condition).
Release factor	This is automatically released after a reset is issued.
Initialization target	All registers except the following. <ul style="list-style-type: none"><li>- PONR stabilization wait circuit</li><li>- Fast-CR oscillation stabilization wait circuit</li><li>- Slow-CR oscillation stabilization wait circuit</li><li>- Post-reset RAM assurance control circuit</li><li>- Reset factor registers</li><li>- Watchdog Reset Boot CPU selection register</li><li>- CPU control register(CPUMD bits)</li><li>- Clock supervisor output enable register</li><li>- Debugging system</li><li>- POWER Manage</li></ul>
Corresponding flag	SYSC_EXCSVRSTCAUSEUR:CSVSSR SYSC_EXCSVRSTCAUSEBT:CSVSSR (for BootROM)



### 3.1.12. Profile Error Reset

This section explains profile error reset.

**Table 3-12 Profile Error Reset**

Occurrence factor	This occurs when an error is detected in the RUN profile in a transition from PSS to RUN
Release factor	This is automatically released after a reset is issued.
Initialization target	<p>All registers except the following:</p> <ul style="list-style-type: none"> <li>- PONR stabilization wait circuit</li> <li>- Fast-CR oscillation stabilization wait circuit</li> <li>- Slow-CR oscillation stabilization wait circuit</li> <li>- Post-reset RAM assurance control circuit</li> <li>- Reset factor registers</li> <li>- Watchdog Reset Boot CPU selection register</li> <li>- CPU control register (CPUMD bits)</li> <li>- Clock supervisor output enable register</li> <li>- Debugging system</li> <li>- POWER Manage</li> </ul>
Corresponding flag	<p>SYSC_RSTCAUSEUR:PRFERR</p> <p>SYSC_RSTCAUSEBT:PRFERR (for BootROM)</p>

### 3.1.13. Software Trigger Hardware Reset

This section explains software trigger hardware reset.

**Table 3-13 Software Trigger Hardware Reset**

Occurrence factor	SYSC_RSTCNTRn:SWHRSTn (n=0 to1) The reset occurs because 0xA5 is written to the control register.( The access is restricted per CPU.)
Release factor	This is automatically released after a reset is issued.
Initialization target	All registers except the following. <ul style="list-style-type: none"> <li>- PONR stabilization wait circuit</li> <li>- Fast-CR oscillation stabilization wait circuit</li> <li>- Slow-CR oscillation stabilization wait circuit</li> <li>- Post-reset RAM assurance control circuit</li> <li>- Reset factor registers</li> <li>- Watchdog Reset Boot CPU selection register</li> <li>- CPU control register (CPUMD bits)</li> <li>- Clock supervisor output enable register</li> <li>- Debugging system</li> <li>- POWER Manage</li> </ul>
Corresponding flag	SYSC_RSTCAUSEUR:SHRSTn SYSC_RSTCAUSEBT:SHRSTn (for BootROM) (n= number of CPUs -1)
Remarks	The control register is equipped per CPU. When one CPU issues reset factor, all initialization target registers are initialized.





### 3.1.14. Software Reset

This section explains software reset.

**Table 3-14 Software Reset**

Occurrence factor	SYSC_RSTCNTRn:SWRSTn (n=0 to 1) This occurs because 0x5A is written to the register. (The access is restricted per CPU.)
Release factor	This is automatically released after a reset is issued.
Initialization target	All registers except the following. <ul style="list-style-type: none"> <li>- PONR stabilization wait circuit</li> <li>- Fast-CR oscillation stabilization wait circuit</li> <li>- Slow-CR oscillation stabilization wait circuit</li> <li>- Post-reset RAM assurance control circuit</li> <li>- Reset factor registers</li> <li>- Watchdog Reset Boot CPU selection register</li> <li>- CPU control register (CPUMD bits)</li> <li>- Clock supervisor output enable register</li> <li>- Debugging system</li> <li>- Software watchdog setting register</li> <li>- Hardware watchdog setting register</li> <li>- Hardware watchdog trigger sequence monitoring logic circuit</li> <li>- Software watchdog trigger sequence monitoring logic circuit</li> <li>- System controller clock setting register (the target is register which can configuration by profile)</li> <li>- POWER Manage</li> </ul>
Corresponding flag	SYSC_RSTCAUSEUR:SRSTn SYSC_RSTCAUSEBT:SRSTn (for BootROM) (n= number of CPUs -1)
Remarks	The control register is equipped per CPU. When one CPU issues reset factor, all initialization target registers are initialized.

Note:

- When generating software reset, clear hardware/software watchdog counter before write to SYSC\_RSTCNTR0/1. After writing to SYSC\_RSTCNTR0/1, do not clear watchdog counter until reset is released.

### 3.1.15. nSRST Pin Input Reset

This section explains nSRST pin input reset.

**Table 3-15 nSRST Pin Input Reset**

Occurrence factor	This occurs because the "L" level is input to the nSRST pin.
Release factor	This is released by input of the "H" level to the nSRST pin.
Initialization target	All registers except the following. <ul style="list-style-type: none"><li>- PONR stabilization wait circuit</li><li>- Fast-CR oscillation stabilization wait circuit</li><li>- Slow-CR oscillation stabilization wait circuit</li><li>- Post-reset RAM assurance control circuit</li><li>- Reset factor registers</li><li>- Watchdog Reset Boot CPU selection register</li><li>- CPU control register (CPUMD bits)</li><li>- Clock supervisor output enable register</li><li>- Debugging system</li><li>- System controller clock setting register (the target is register which can configuration by profile)</li><li>- POWER Manage</li></ul>
Corresponding flag	SYSC_RSTCAUSEUR:SRSTX SYSC_RSTCAUSEBT:SRSTX (for BootROM)
Remarks	nSRST pin input is valid during 128-bit security key (device security key) is released.



### 3.1.16. TRSTX Pin Input Reset

This section explains TRSTX pin input reset.

**Table 3-16 TRSTX Pin Input Reset**

Occurrence factor	This occurs because the "L" level is input to the TRSTX pin.
Release factor	This is released by input of the "H" level to the TRSTX pin.
Initialization target	Debugging system
Corresponding flag	None

### 3.1.17. Software Debugger Reset

This section explains software debugger reset.

**Table 3-17 Software Debugger Reset**

Occurrence factor	SYSC_RSTCNTRn:DBG Rn (n=0 to 1) The reset occurs because 0xDA is written to the control register. ( The access is restricted per CPU.)
Release factor	After reset is issued, it automatically releases it.
Initialization target	Debugging system
Corresponding flag	None
Remarks	The control register is equipped per CPU. The access is restricted per CPU. When one CPU issues reset factor, all initialization target registers are initialized.



## 3.2. Internal Reset of Device

This section explains the internal reset signals of this device.

The reset signals are output corresponding to each domain.

- CPU Reset
- Reset of Peripheral Circuit
- I/O Reset
- FLASH Reset

### 3.2.1. CPU Reset

This section explains the CPU reset.

**Table 3-18 CPU Reset**

Signal Output	Function	Type	Reset Factor
SYSPORX	<ul style="list-style-type: none"> <li>- Connected to nSYSPORESET in Core/Debug Group</li> <li>- It is reset</li> </ul>	Hardware reset	<ul style="list-style-type: none"> <li>- Power-on reset</li> <li>- Internal power supply low-voltage detection</li> <li>- INITX pin input</li> </ul>
CPU0RSTX	<ul style="list-style-type: none"> <li>- Connected to nRESET0 in Core/Debug group</li> <li>- Reset expect debugging logic</li> <li>- Reset is always asserted in 1CPU1 mode.</li> </ul>	Hardware reset	<ul style="list-style-type: none"> <li>- RSTX pin input</li> <li>- Clock stop wait timeout</li> <li>- Hardware watchdog</li> <li>- Software watchdog</li> <li>- 5V external power supply low-voltage detection</li> <li>- Clock supervisor reset</li> <li>- Profile error reset</li> <li>- Software trigger hardware reset</li> <li>- nSRST pin input</li> </ul>
		Software reset	Software reset
CPU1RSTX	<ul style="list-style-type: none"> <li>- Connected to nRESET1 in Core/Debug group</li> <li>- Reset expect debugging logic</li> <li>- Reset is always asserted in 1CPU0 mode.</li> </ul>	Hardware reset	<ul style="list-style-type: none"> <li>- RSTX pin input</li> <li>- Clock stop wait timeout</li> <li>- Hardware watchdog</li> <li>- Software watchdog</li> <li>- 5V external power supply low-voltage detection</li> <li>- Clock supervisor reset</li> <li>- Profile error reset</li> <li>- Software trigger hardware reset</li> <li>- nSRST pin input</li> </ul>
		Software reset	Software reset
DBGRSTX	Debugger reset	Debugger reset	Software debugger reset



### 3.2.2. Reset of Peripheral Circuit

This section explains reset of peripheral circuit.

**Table 3-19 Reset of Peripheral Circuit**

Signal Output	Function	Type	Reset Factor
RSTX_SCUH	<ul style="list-style-type: none"> <li>Reset clock manage in SYSC</li> <li>Reset test control circuit</li> </ul>	Hardware reset	All reset factors except nSRST pin input reset
RSTX_POWER	Reset power manage in SYSC	Hardware reset	<ul style="list-style-type: none"> <li>Power-on reset</li> <li>Internal power supply low-voltage detection</li> <li>INITX pin input</li> <li>Hardware watchdog</li> </ul>
RSTX_SCU	Reset SYSC	Hardware reset	All reset factors
		Software reset	All reset factors
RSTX_SCUP	<ul style="list-style-type: none"> <li>Fast-CR oscillation stabilization wait circuit</li> <li>Slow-CR oscillation stabilization wait circuit</li> <li>Post-reset RAM assurance control circuit</li> </ul> Reset the above	Hardware reset	<ul style="list-style-type: none"> <li>Power-on reset</li> <li>Internal power supply low-voltage detection</li> <li>INITX pin input (latch/release extension are none)</li> </ul>
RSTX_PD1	<ul style="list-style-type: none"> <li>Hardware watchdog</li> <li>Software watchdog</li> <li>External interrupt circuit</li> </ul> Reset the above	Hardware reset	All reset factors
		Software reset	All reset factors
RSTX_PD1H	<ul style="list-style-type: none"> <li>Hardware watchdog</li> <li>Software watchdog</li> </ul> Reset setting register other than the above	Hardware reset	All reset factors
RSTX_PD2	<ul style="list-style-type: none"> <li>DMAC</li> <li>Common Peripheral group</li> <li>High Performance Matrix</li> <li>Application Specific Peripheral group</li> </ul> Reset the above	Hardware reset	All reset factors
		Software reset	All reset factors
RSTX_PD2H	Reset security circuit	Hardware reset	All reset factors except nSRST pin input reset
RSTX_PD3	Reset Memory & Config group	Hardware reset	All reset factors
		Software reset	All reset factors

### 3.2.3. I/O reset

This section explains the I/O reset.

**Table 3-20 I/O Reset**

Signal Output	Function	Type	Reset Factor
RSTX_IO	Initialization of I/O cell	Hardware reset	<ul style="list-style-type: none"><li>- Power-on reset</li><li>- External power supply low-voltage detection</li><li>- RSTX pin input</li><li>- INITX pin input</li></ul>

The I/O cell connected to the external pin of the device is initialized when the I/O reset (RSTX\_IO) is issued. During the initialization, the I/O cell is in the high impedance state.

The I/O cell is asynchronously initialized when a reset factor is received. It stays in this state until an internal reset is issued and the reset factor is released.

Signal Output	Function	Type	Reset Factor
RSTX_PD2	Common Peripheral Group	Hardware reset	All reset factors
		Software reset	All reset factors

If a peripheral circuit reset (RSTX\_PD2) is issued, the I/O port data direction register and the port input enable register are initialized, and the I/O cell connected to the external pin of the device enters the high-impedance state.





### 3.2.4. FLASH Reset

This section explains FLASH reset.

**Table 3-21 FLASH Reset**

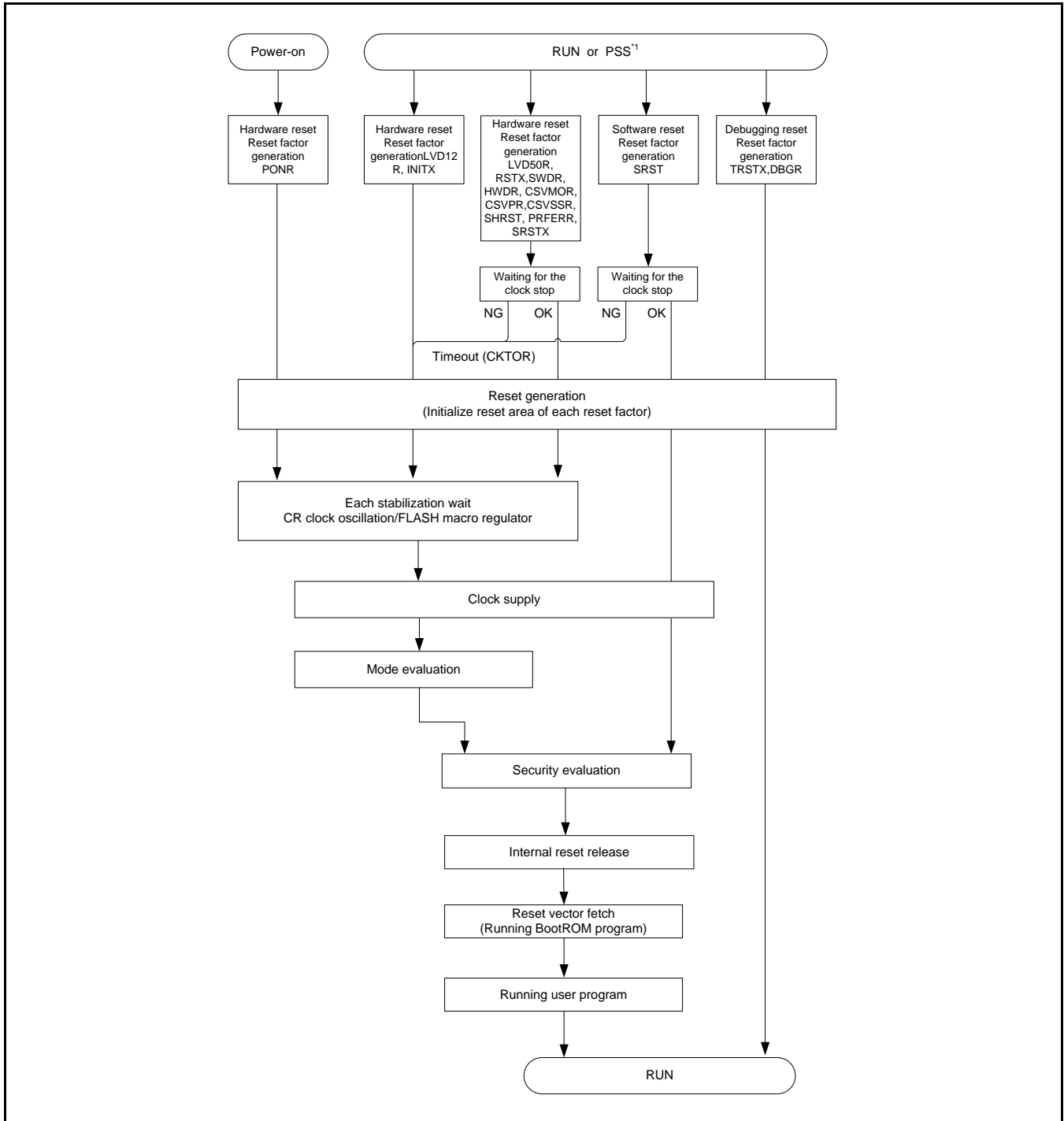
Signal Name	Function	Type	Reset Factor
FRSTRX	TCFLASH/WorkFLASH regulator reset	Hardware reset	<ul style="list-style-type: none"> <li>- Power-on reset</li> <li>- Internal power supply low-voltage detection reset</li> <li>- INITX pin input</li> <li>- Hardware watchdog</li> </ul>
		Low-power consumption control	CPU0 transits to PSS.
FRSTX	TCFLASH/WorkFLASH Reset	Hardware reset	All reset factors
		Software reset	All reset factors
		Low-power consumption control	CPU0 transits to PSS.

FRSTRX initializes the regulator with built-in each FLASH macro. When the reset factor is accepted, the regulator is initialized asynchronously. The initialization release is automatically done in the reset return sequence.

### 3.3. Reset Sequence

This section explains the reset sequence of this device. Program and hardware operation will start from initial state by releasing the reset. The sequence from reset to operation start is called reset sequence.

Figure 3-1 Reset Sequence



**Note:**

- At PSS (It is \*1 in figure), the reset factor that CPU lies (software trigger hardware reset, software reset, and software debugger reset) doesn't generate.



### 3.4. Operations after Reset Release

This section explains the sequence after reset release.

The reset factors and operation after reset is released are the following.

**Table 3-22 Operations after Reset Release**

Reset Category	Reset Factor	Operation after Reset is Released				
		Mode Evaluation	Security Evaluation	Waiting for Stabilization	Operation Clock	RAM Guarantee
Hardware reset	Power-on reset	Yes	Yes	PONR Fast-CR Slow-CR FLASH	Fast-CR	No
	<ul style="list-style-type: none"> <li>Internal power supply low-voltage detection</li> <li>INITX pin input</li> </ul>	Yes	Yes	Fast-CR Slow-CR FLASH	Fast-CR	No
	Clock stop wait timeout	Yes	Yes	No	Fast-CR	No
	Hardware watchdog	Yes	Yes	FLASH	Fast-CR	Yes
	<ul style="list-style-type: none"> <li>RSTX pin input</li> <li>Software watchdog</li> <li>External power supply low-voltage detection</li> <li>Profile error reset</li> <li>Software trigger hardware reset</li> </ul>	Yes	Yes	No	Fast-CR	Yes
	Clock supervisor reset	Yes	Yes	No	Fast-CR	No
	nSRST pin input reset	No	Yes	No	No change.	Yes
Software reset	Software reset	No	Yes	No	No change	Yes
Debugger reset	TRSTX pin input Software debugger reset	No	No	No	No change	Yes

Several hardware reset factors guarantee the RAM data after the reset but the others do not. The reset factors which do not guarantee the RAM data issue the reset regardless RAM access, so the RAM data are not guaranteed after the reset. The reset factors which guarantee the RAM data issue the reset after the clocks have stopped.

However, in a case that a reset factor which can guarantees the RAM data has not completed within a certain time period, another reset is issued (Clock stop wait timeout reset). In this case, the RAM data after recovery from the reset is not guaranteed.

**Note:**

- ETB (Embedded Trace Buffer) RAM is not guaranteed after software debugger reset release.

### 3.4.1. Hardware Reset Operation

Figure 3-2 Operation at Power-on Reset

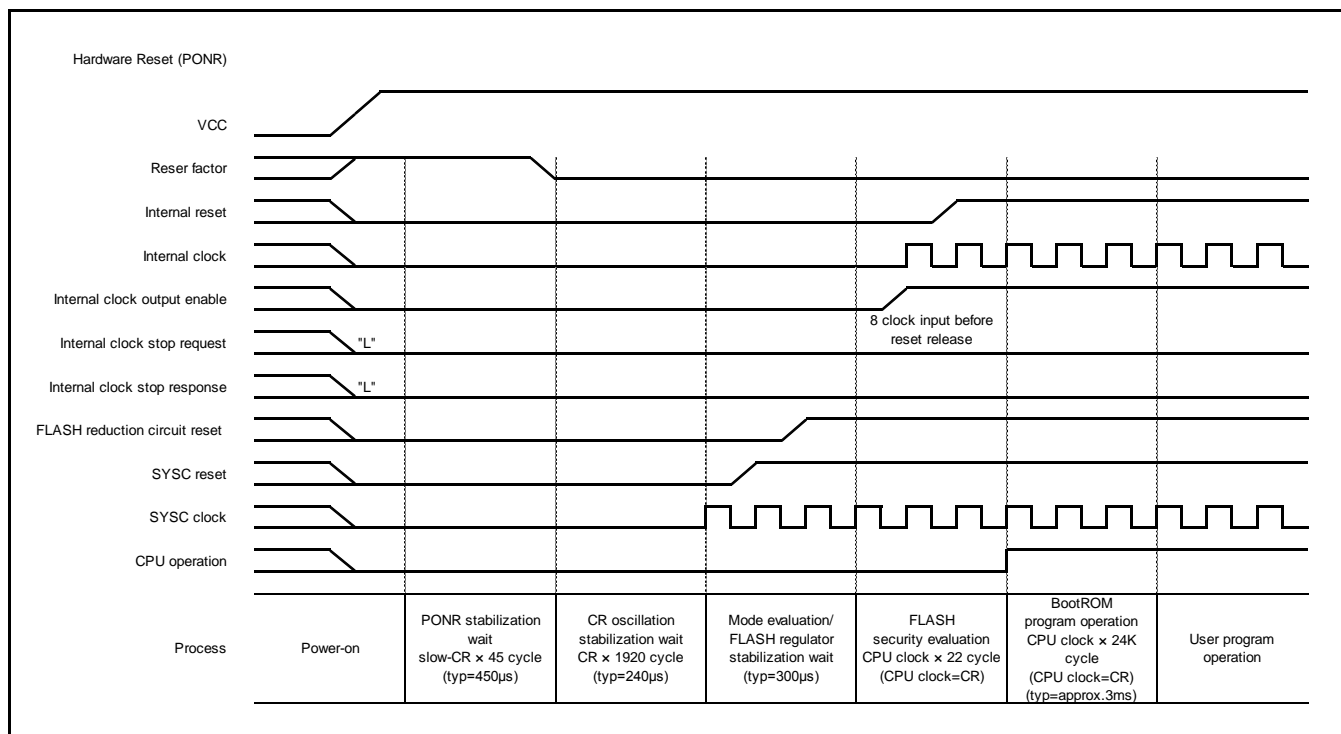


Figure 3-3 Operation at Internal Low-voltage Detection or INITX Pin Input

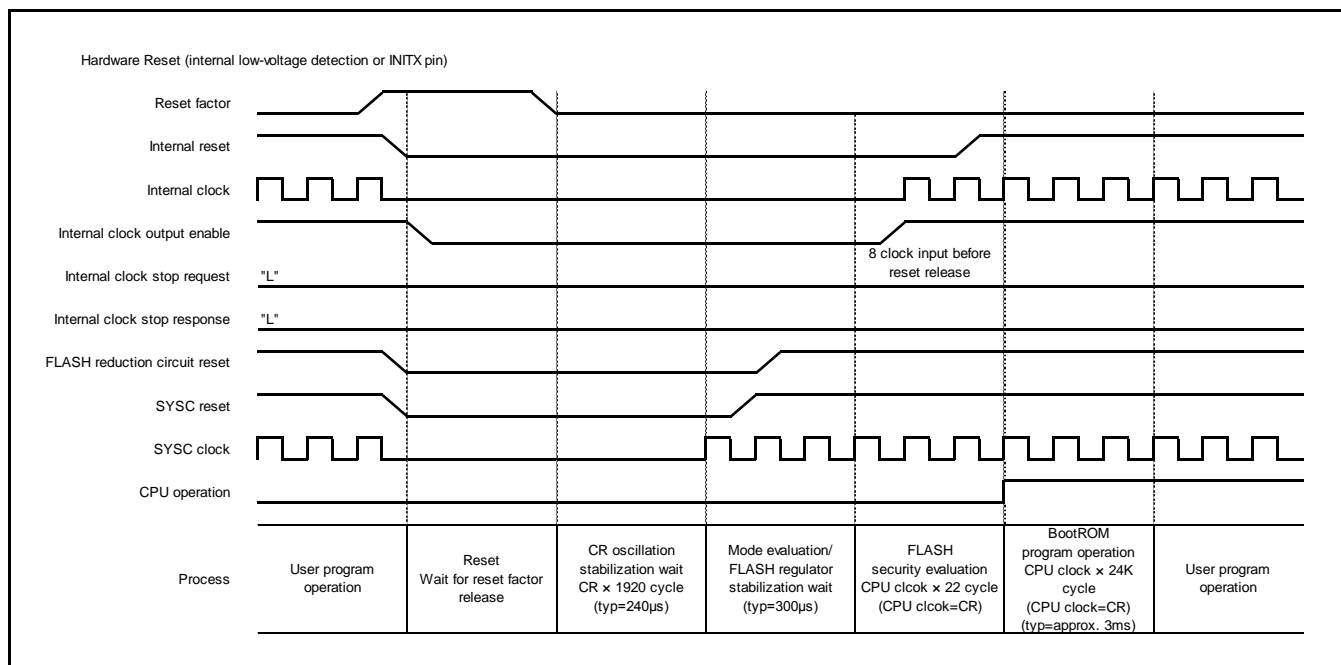




Figure 3-4 Operation at Clock Stop Wait Timeout

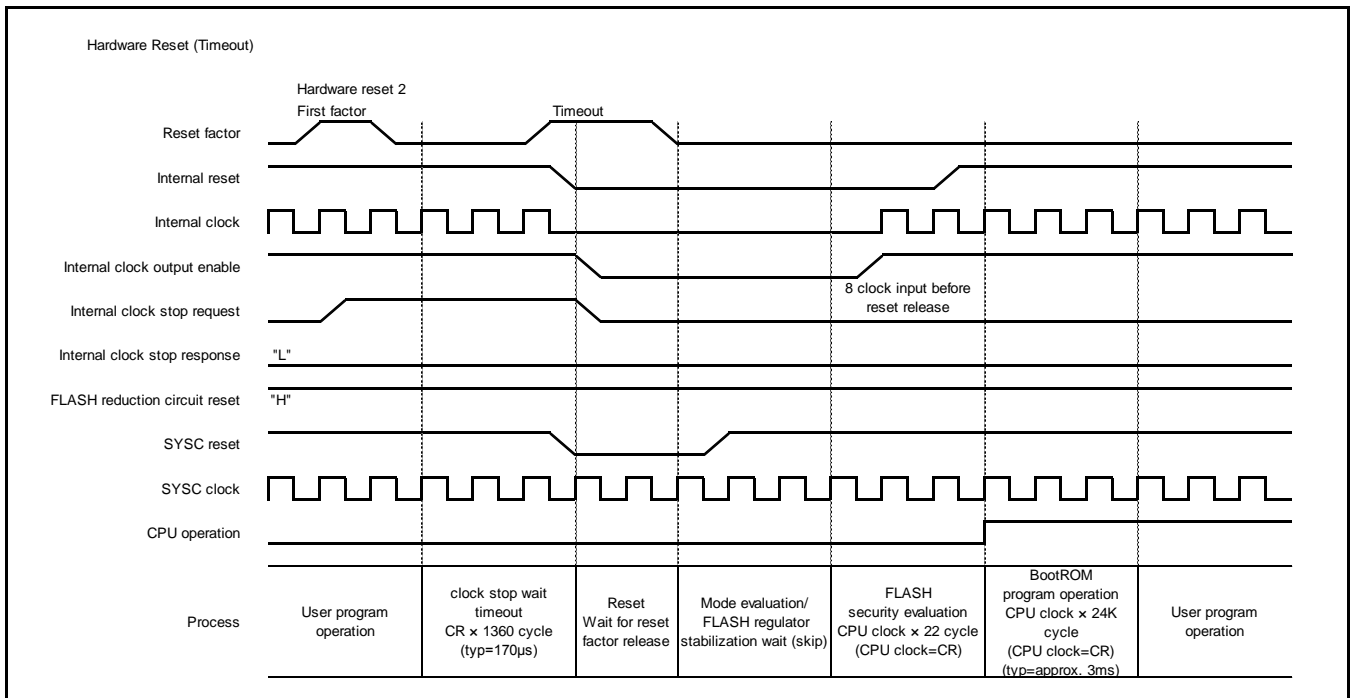


Figure 3-5 Operation at Hardware Watchdog

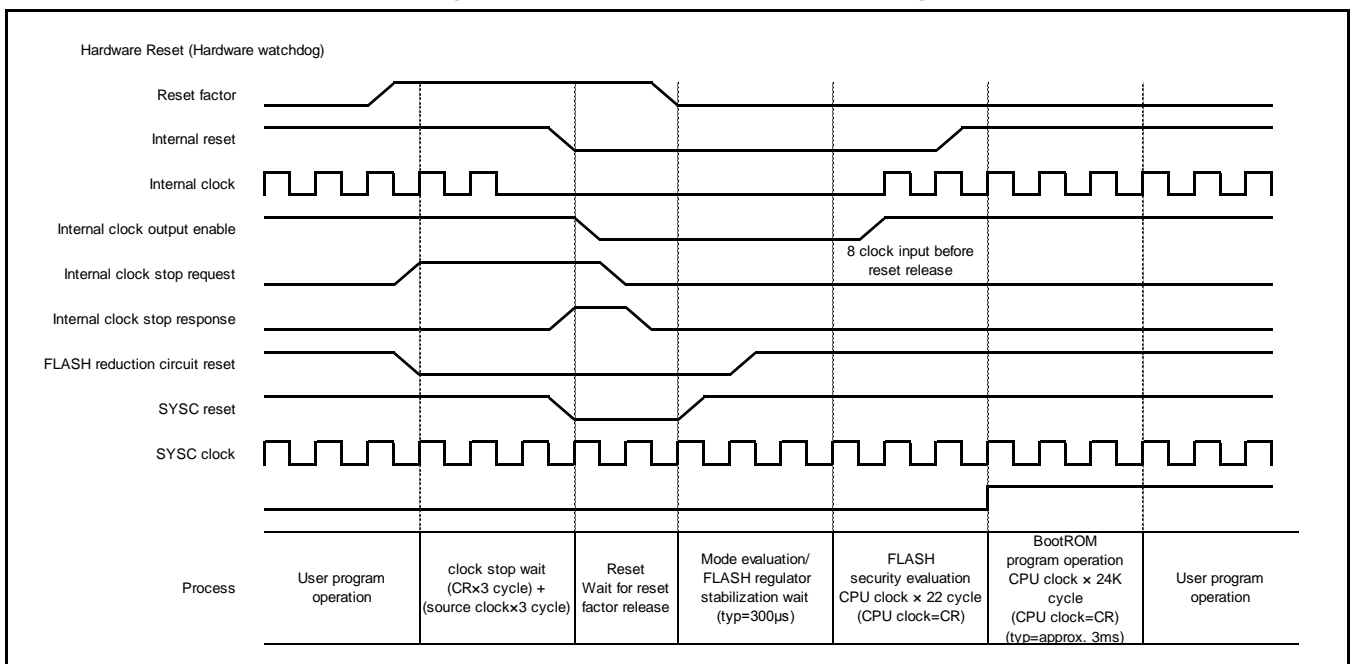
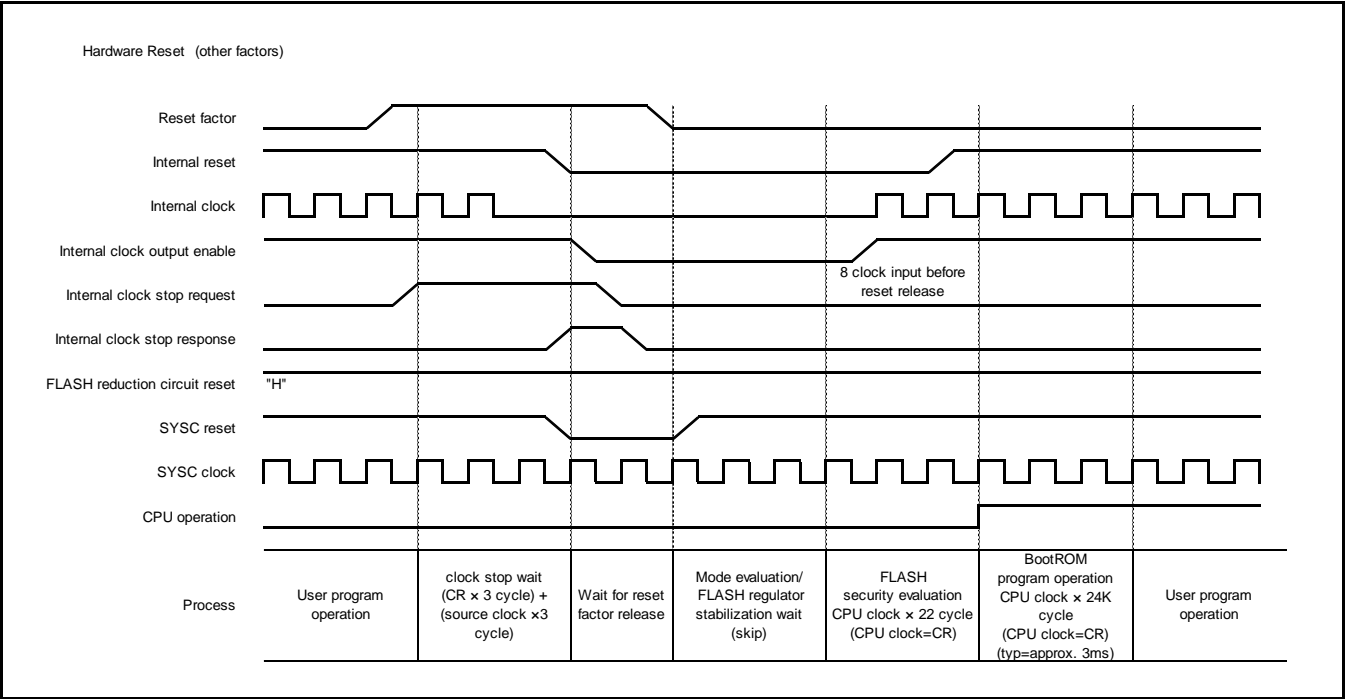


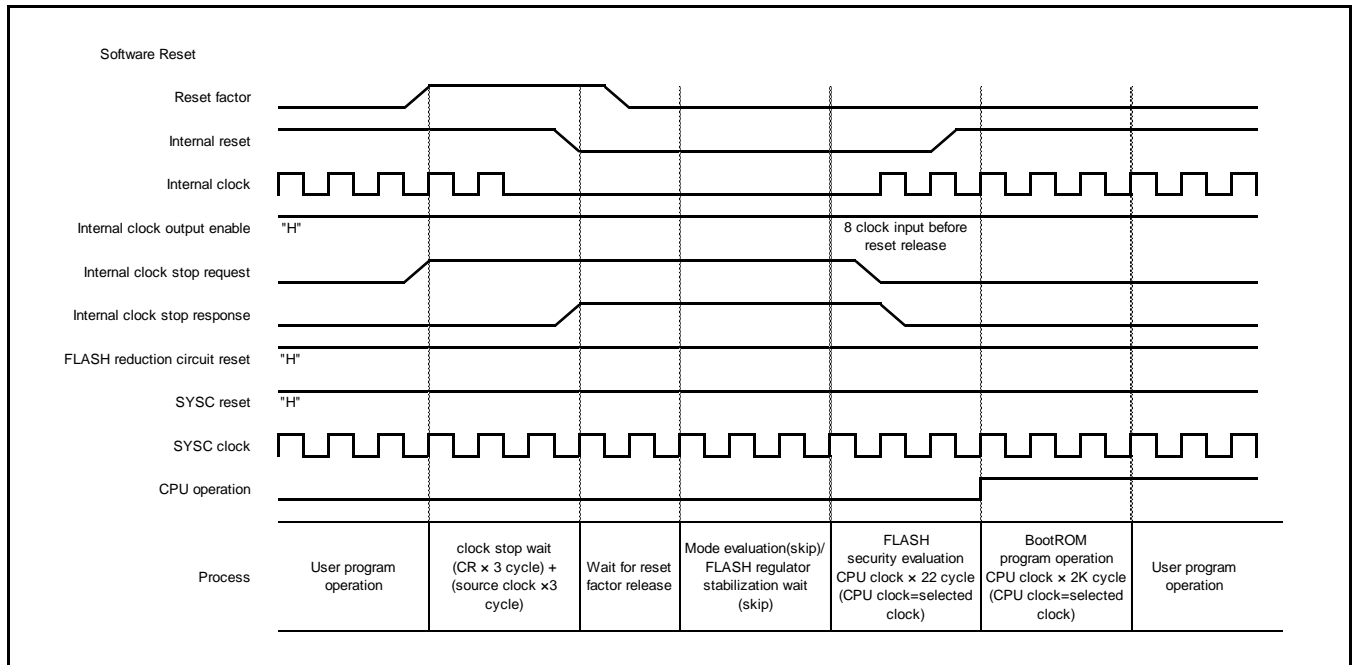
Figure 3-6 Operation at Other Hardware Reset Factors





### 3.4.2. Software Reset Operation

Figure 3-7 Operation at Software Reset



### 3.4.3. Recovery Operation from PSS

Figure 3-8 Recovery after Hardware Reset (Power-on Reset)

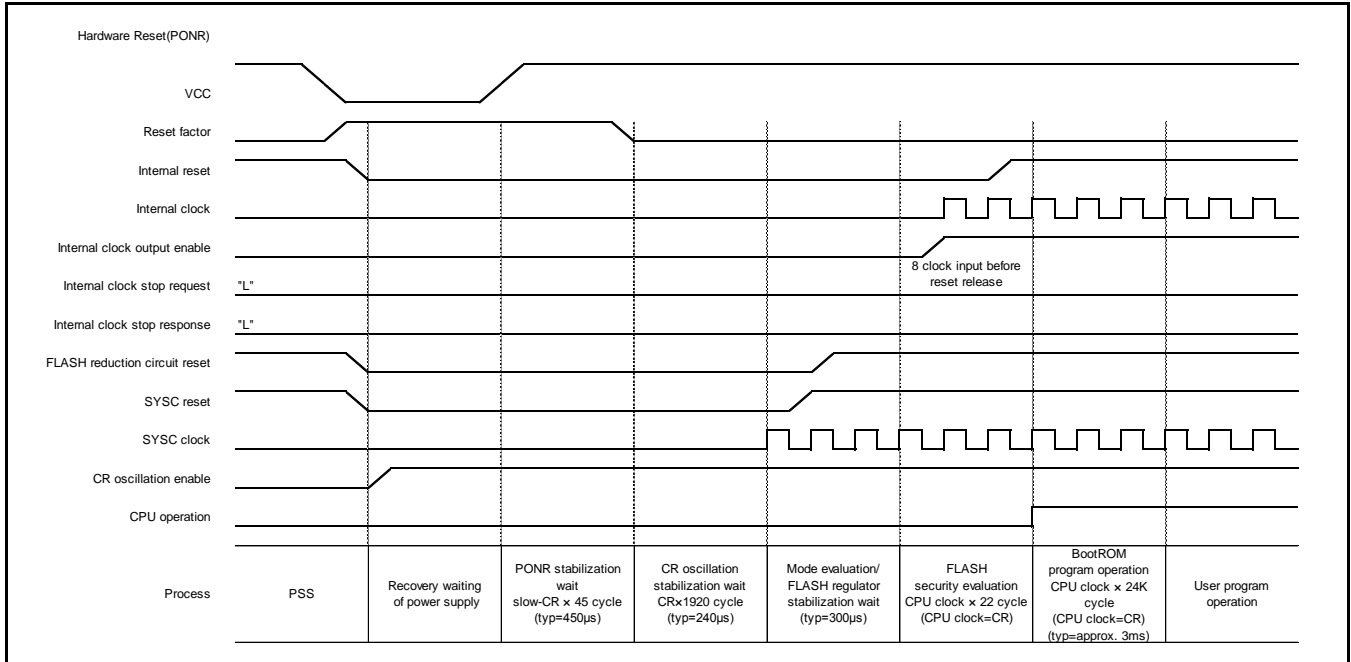


Figure 3-9 Recovery after Hardware Reset (Internal low-voltage Detection and INITX Pin)

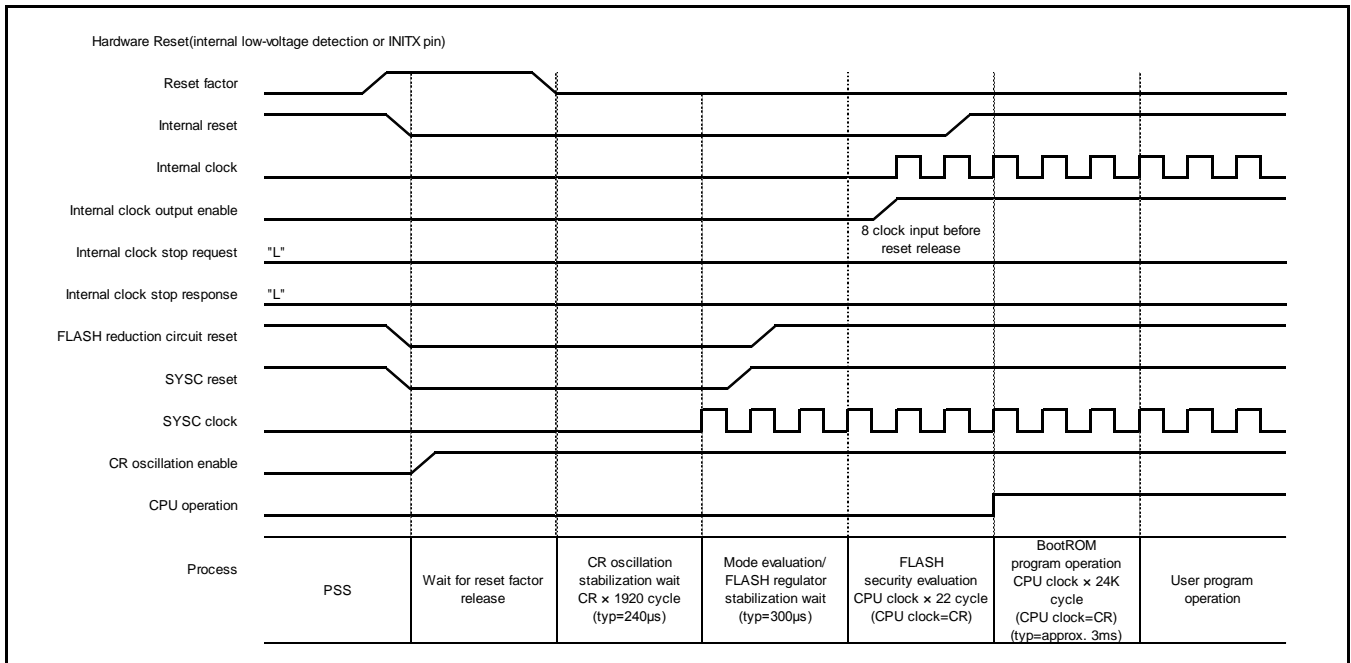






Figure 3-10 Recovery after Hardware Reset (Hardware Watchdog)

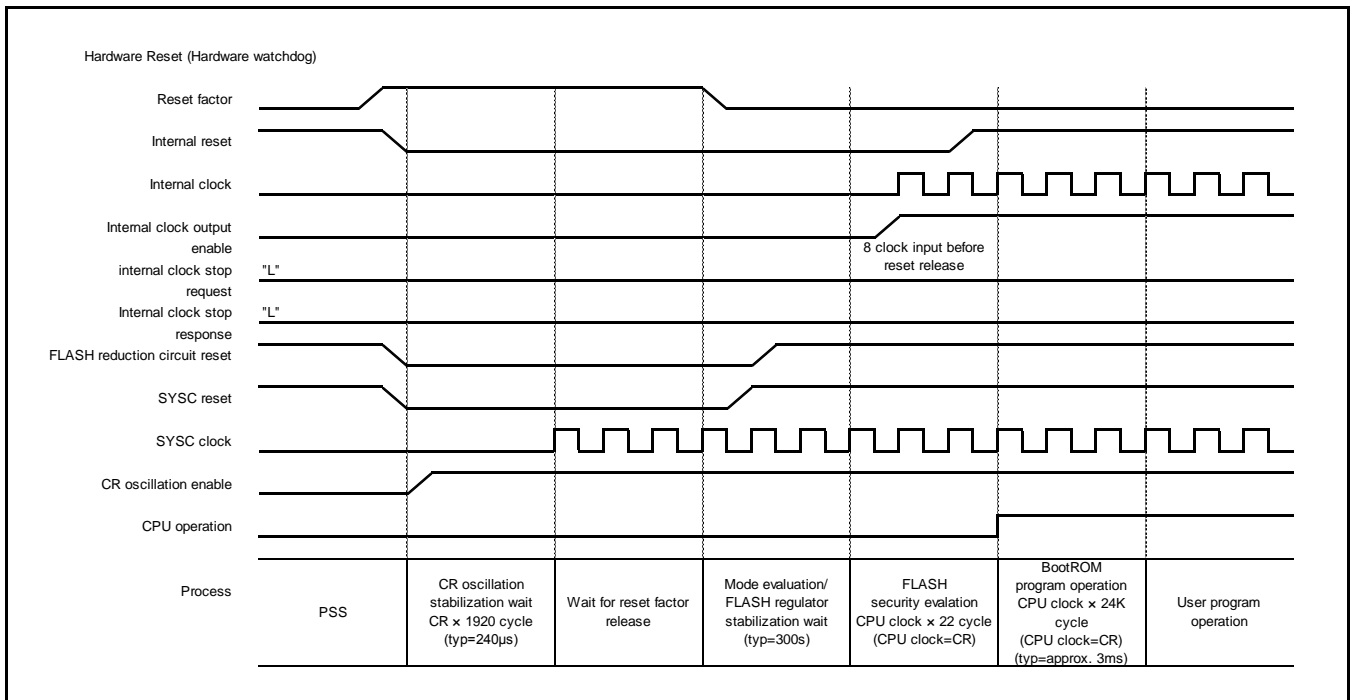


Figure 3-11 Recovery after Hardware Reset (Other Hardware Reset)

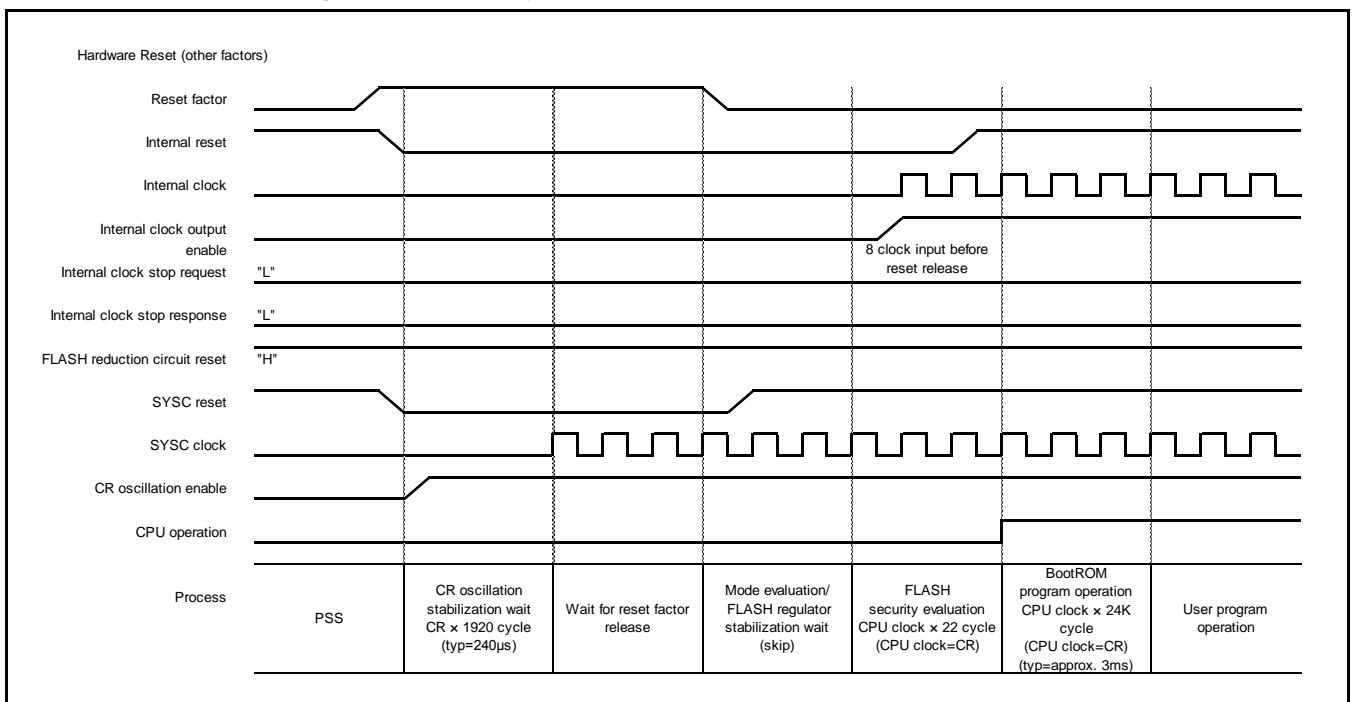
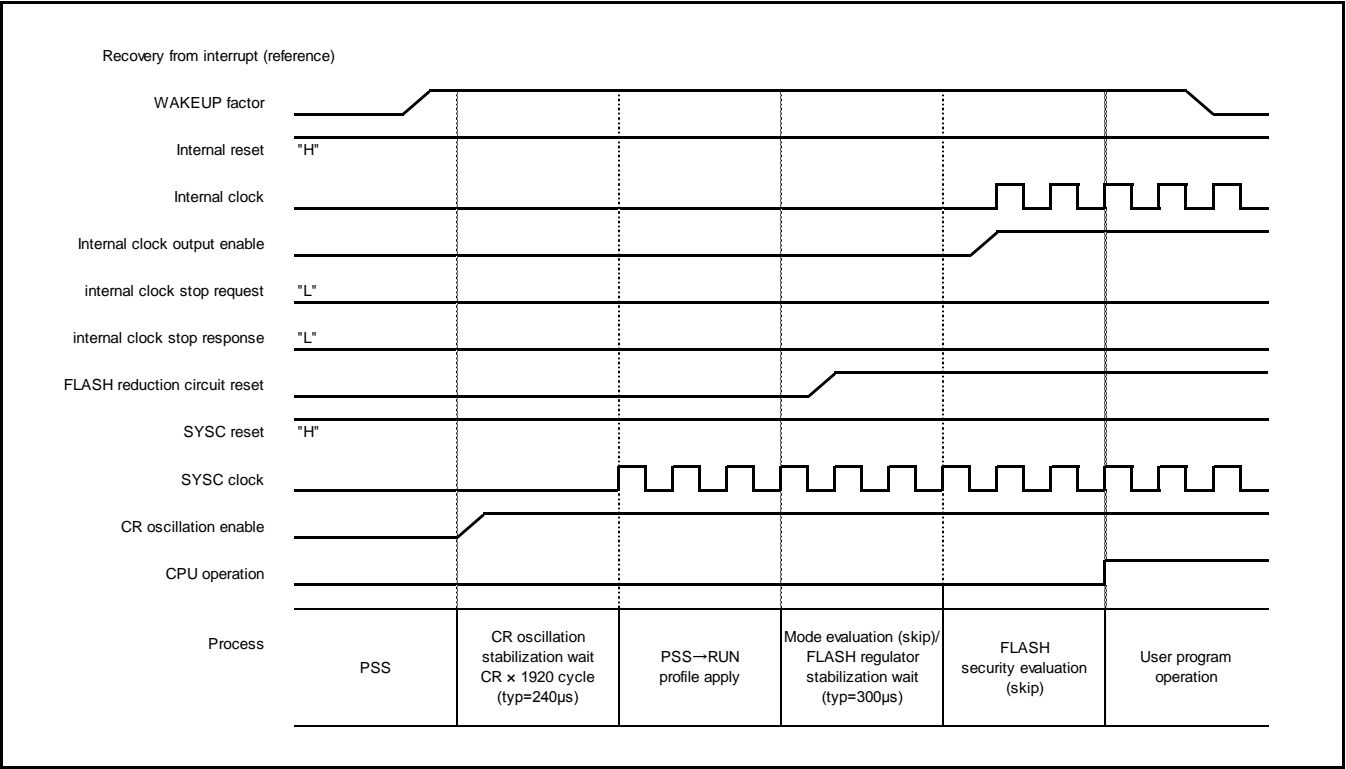




Figure 3-12 Recovery from Interrupt



**Note:**

- The clock in figure is not corresponding to actual cycle number.



## 4. Registers

This section explains a list of reset registers.

**Table 4-1 List of Reset Registers**

Abbreviated Register Name	Register Name	See
SYSC_RSTCNTR0	Reset Control Register 0	4.1
SYSC_RSTCNTR1	Reset Control Register 1	4.2
SYSC_RSTCAUSEUR	User Reset Factor Register	4.3
SYSC_EXCSVRSTCAUSEUR	User Extended CSV Reset Factor Register	4.4
SYSC_RSTCAUSEBT	BootROM Reset Factor Register	4.5
SYSC_EXCSVRSTCAUSEBT	BootROM Extended CSV Reset Factor Register	4.6
SYSC_WRBOOTCPUSEL	Watchdog Reset Boot CPU Selection Register	4.7

## 4.1. Reset Control Register 0 (SYSC\_RSTCNTR0)

This register controls resets of the CPU0. It generates a software reset, software trigger hardware reset, and debugging reset. Protection key code input is required for writing.

BITS	31	30	29	28	27	26	25	24
BIT_OFFSET								
BIT_NAME	DBG0							
ACCESS_TYPE	R0,W							
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							

BITS	23	22	21	20	19	18	17	16
BIT_OFFSET								
BIT_NAME	SWHRST0							
ACCESS_TYPE	R0,W							
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							

BITS	15	14	13	12	11	10	9	8
BIT_OFFSET								
BIT_NAME	Reserved							
ACCESS_TYPE	R0,W0							
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							

BITS	7	6	5	4	3	2	1	0
BIT_OFFSET								
BIT_NAME	SWRST0							
ACCESS_TYPE	R0,W							
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							

### [bit31:24] DBG0[7:0]: Software debugger reset register bits

bit31:24	Description
Write	The bits are used to generate the software debugger reset of CPU0. Write 0xDA in this register to generate it. See Section "3.1. Reset Factor" for details of it.
Read	This register is automatically cleared after writing. The read value is always "0".



**[bit23:16] SWHRST0[7:0]: Software trigger hardware reset register bits**

bit23:16	Description
Write	The bits are used to generate the software trigger hardware reset of CPU0. Write 0xA5 in this register to generate it. See Section "3.1. Reset Factor" for details of it.
Read	This register is automatically cleared after writing. The read value is always "0".

**[bit15:8] Reserved: Reserved bits**

**[bit7:0] SWRST0[7:0]: Software reset register bits**

bit7:0	Description
Write	The bits are used to generate the software reset of CPU0. Write 0x5A in this register to generate it. See Section "3.1. Reset Factor" for details of it.
Read	This register is automatically cleared after writing. The read value is always "0".

**Notes:**

- Protection key code input is required for writing.
- It results bus error response for the violation to above restriction.
- This register can be subject to write access by CPU0/1 only. Read access has no limits.

## 4.2. Reset Control Register 1 (SYSC\_RSTCNTR1)

This register controls resets of the CPU1. It generates a software reset, software trigger hardware reset, and debugging reset. Protection key code input is required for writing.

BITS	31	30	29	28	27	26	25	24
BIT_OFFSET								
BIT_NAME	DBG1							
ACCESS_TYPE	R0,W							
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							

BITS	23	22	21	20	19	18	17	16
BIT_OFFSET								
BIT_NAME	SWHRST1							
ACCESS_TYPE	R0,W							
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							

BITS	15	14	13	12	11	10	9	8
BIT_OFFSET								
BIT_NAME	Reserved							
ACCESS_TYPE	R0,W0							
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							

BITS	7	6	5	4	3	2	1	0
BIT_OFFSET								
BIT_NAME	SWRST1							
ACCESS_TYPE	R0,W							
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							

### [bit31:24] DBG1[7:0]: Software debugger reset register bits

bit31:24	Description
Write	The bits are used to generate the software debugger reset of CPU1. Write 0xDA in this register to generate it. See Section "3.1. Reset Factor" for details of it.
Read	This register is automatically cleared after writing. The read value is always "0".



**[bit23:16] SWHRST1[7:0]: Software trigger hardware reset register bits**

bit23:16	Description
Write	The bits are used to generate the software trigger hardware reset of CPU1. Write 0xA5 in this register to generate it. See Section "3.1. Reset Factor" for details of it.
Read	This register is automatically cleared after writing. The read value is always "0".

**[bit15:8] Reserved: Reserved bits**

**[bit7:0] SWRST1[7:0]: Software reset register bits**

bit7:0	Description
Write	The bits are used to generate the software reset of CPU1. Write 0x5A in this register to generate it. See Section "3.1. Reset Factor" for details of it.
Read	This register is automatically cleared after writing. The read value is always "0".

**Notes:**

- Protection key code input is required for writing.
- It results bus error response for the violation to above restriction.
- This register can be subject to write access by CPU0/1 only. Read access has no limits.

### 4.3. User Reset Factor Register (SYSC\_RSTCAUSEUR)

This register indicates the last reset factor. Clear this reset factor before the next reset. If not cleared, it will be difficult to identify the last reset factors because the register bits are overwritten by new reset factors. The register is initialized by the power-on reset (PONR). For rewriting, protect key code input is necessary for each register access.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved				Reserved	CSVPR	Reserved	CSVMOR
ACCESS_TYPE	R0,W0				R0,W0	R,W	R0,W0	R,W
PROT_TYPE	WPS							
INITIAL_VALUE	0000				0	0	0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved		SHRST1	SHRST0	Reserved		SRST1	SRST0
ACCESS_TYPE	R0,W0		R,W	R,W	R0,W0		R,W	R,W
PROT_TYPE	WPS							
INITIAL_VALUE	00		0	0	00		0	0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved		SWDR1	SWDR0	HWDR	PRFERR	SRSTX	Reserved
ACCESS_TYPE	R0,W0		R,W	R,W	R,W	R,W	R,W	R0,W0
PROT_TYPE	WPS							
INITIAL_VALUE	00		0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved	LVD50R	Reserved	RSTX	CKTOR	INITX	LVD12R	PONR
ACCESS_TYPE	R0,W0	R,W	R0,W0	R,W	R,W	R,W	R,W	R,W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	1

**[bit31:28] Reserved: Reserved bits**

**[bit27] Reserved: Reserved bit**

**[bit26] CSVPR: Clock supervisor reset (PLL clock) detection bit**

This bit is used to confirm detection of a clock supervisor reset (PLL clock). Before the next reset is generated, clear this by writing "0".

Value	Description
0	The reset has not been detected.
1	The reset has been detected.

**[bit25] Reserved: Reserved bit**





**[bit24] CSV MOR: Clock supervisor reset (main clock) detection bit**

This bit is used to confirm detection of a clock supervisor reset (main clock). Before the next reset is generated, clear this by writing "0".

Value	Description
0	The reset has not been detected.
1	The reset has been detected.

**[bit23:22] Reserved: Reserved bits**

**[bit21] SHRST1: Software trigger hardware reset detection bit**

This bit is used to confirm detection of the software trigger hardware reset of CPU1. Before the next reset is generated, clear this by writing "0".

Value	Description
0	The reset has not been detected.
1	The reset has been detected.

**[bit20] SHRST0: Software trigger hardware reset detection bit**

This bit is used to confirm detection of the software trigger hardware reset of CPU0. Before the next reset is generated, clear this by writing "0".

Value	Description
0	The reset has not been detected.
1	The reset has been detected.

**[bit19:18] Reserved: Reserved bits**

**[bit17] SRST1: Software reset detection bit**

This bit is used to confirm the software reset detection of CPU1. Before the next reset is generated, clear this by writing "0".

Value	Description
0	The reset has not been detected.
1	The reset has been detected.

**[bit16] SRST0: Software reset detection bit**

This bit is used to confirm the software reset detection of CPU0. Before the next reset is generated, clear this by writing "0".

Value	Description
0	The reset has not been detected.
1	The reset has been detected.

**[bit15:14] Reserved: Reserved bits**

**[bit13] SWDR1: Software watchdog reset detection bit**

This bit is used to confirm detection of the software watchdog reset of CPU1. Before the next reset is generated, clear this by writing "0".

Value	Description
0	The reset has not been detected.
1	The reset has been detected.

**[bit12] SWDR0: Software watchdog reset detection bit**

This bit is used to confirm detection of the software watchdog reset of CPU0. Before the next reset is generated, clear this by writing "0".

Value	Description
0	The reset has not been detected.
1	The reset has been detected.

**[bit11] HWDR: Hardware watchdog reset detection bit**

This bit is used to confirm detection of the hardware watchdog reset. Before the next reset is generated, clear this by writing "0".

Value	Description
0	The reset has not been detected.
1	The reset has been detected.

**[bit10] PRFERR: Profile error reset detection bit**

This bit is used to confirm detection of the profile error reset. Before the next reset is generated, clear this by writing "0".

Value	Description
0	The reset has not been detected.
1	The reset has been detected.

**[bit9] SRSTX: nSRST pin reset detection bit**

This bit is used to confirm detection of the nSRST pin reset. Before the next reset is generated, clear this by writing "0".

Value	Description
0	The reset has not been detected.
1	The reset has been detected.

**[bit8:7] Reserved: Reserved bits**

**[bit6] LVD50R: External power supply (5.0V) low-voltage detection reset detection bit**

This bit is used to confirm detection of the external power supply (5.0V) low-voltage detection reset. Before the next reset is generated, clear this by writing "0".

Value	Description
0	The reset has not been detected.
1	The reset has been detected.



**[bit5] Reserved: Reserved bit**

**[bit4] RSTX: RSTX pin input reset detection bit**

This bit is used to confirm detection of the RSTX pin input reset. Before the next reset is generated, clear this by writing "0".

Value	Description
0	The reset has not been detected.
1	The reset has been detected.

**[bit3] CKTOR: Clock stop wait timeout reset detection bit**

This bit is used to confirm detection of the clock stop wait timeout reset. Before the next reset is generated, clear this by writing "0".

Value	Description
0	The reset has not been detected.
1	The reset has been detected.

**[bit2] INITX: INITX pin input reset detection bit**

This bit is used to confirm detection of the INITX pin input reset. Before the next reset is generated, clear this by writing "0".

Value	Description
0	The reset has not been detected.
1	The reset has been detected.

**[bit1] LVD12R: Internal power supply low-voltage detection reset detection bit**

This bit is used to confirm detection of the internal power supply low-voltage detection reset. Before the next reset is generated, clear this by writing "0". When reset is generated by LVD12R, the PONR (bit0) detection bit is also set.

Value	Description
0	The reset has not been detected.
1	The reset has been detected.

**[bit0] PONR: Power-on reset detection bit**

This bit is used to confirm detection of the power-on reset. Before the next reset is generated, clear this by writing "0".

Value	Description
0	The reset has not been detected.
1	The reset has been detected.

**Notes:**

- Protection key code input is required for writing.
- If the above is violated, a bus error occurs.
- In multi-CPU mode, this register can be subject to write access by CPU0 only. Read access has no limits.

#### 4.4. User Extended CSV Reset Factor Register (SYSC\_EXCSVRSTCAUSEUR)

This register indicates the last reset factor. Clear reset factor before the next reset. If not cleared, it will be difficult to identify the last reset factors because the register bits are overwritten by new reset factors. The register bits are initialized by the power-on reset (PONR). Protection key code input is required for writing.

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,W0
PROT_TYPE	WPS
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	CSVSSR							
ACCESS_TYPE	R0,W0							R,W
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							

**[bit31:8] Reserved: Reserved bits**

##### **[bit7:0] CSVSSR[7:0]: Clock supervisor reset (sub-system PLL clock) detection bits**

This bit is used to confirm detection of the clock supervisor reset (sub-system PLL clock). Before the next reset is generated, clear this by writing "0".

Value	Description
0	The reset has not been detected.
1	The reset has been detected.

CSVSSR[0] is detection bit of PLL for FlexRay/RDC clock supervisor reset.

CSVSSR[7:1] are reserved bits.

##### **Notes:**

- Protection key code input is required for writing.
- It results bus error response for the violation to above restriction.
- In multi-CPU mode, this register can be subject to write access by CPU0 only. Read access has no limits.



## 4.5. BootROM Reset Factor Register (SYSC\_RSTCAUSEBT)

This register indicates the last reset factor. Clear reset factor before the next reset. If not cleared, it will be difficult to identify the last reset factors because the register bits are overwritten by new reset factors. The register bits are initialized by the power-on reset (PONR). Protection key code input is required for writing.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved				Reserved	CSVPR	Reserved	CSVMOR
ACCESS_TYPE	R0,W0				R0,W0	R,W	R0,W0	R,W
PROT_TYPE	WPS							
INITIAL_VALUE	0000				0	0	0	0

BIT_OFFSET	23	22	21	20	19	18	bit17	bit16
BIT_NAME	Reserved		SHRST1	SHRST0	Reserved		SRST1	SRST0
ACCESS_TYPE	R0,W0		R,W	R,W	R0,W0		R,W	R,W
PROT_TYPE	WPS							
INITIAL_VALUE	00		0	0	00		0	0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved		SWDR1	SWDR0	HWDR	PRFERR	SRSTX	Reserved
ACCESS_TYPE	R0,W0		R,W	R,W	R,W	R,W	R,W	R0,W0
PROT_TYPE	WPS							
INITIAL_VALUE	00		0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved	LVD50R	Reserved	RSTX	CKTOR	INITX	LVD12R	PONR
ACCESS_TYPE	R0,W0	R,W	R0,W0	R,W	R,W	R,W	R,W	R,W
PROT_TYPE	WPS							
INITIAL_VALUE	0	0	0	0	0	0	0	1

[bit31:28] Reserved: Reserved bits

[bit27] Reserved: Reserved bit

[bit26] CSVPR: Clock supervisor reset (PLL clock) detection bit

This bit is used to confirm detection of the PLL clock supervisor reset. Before the next reset is generated, clear this by writing "0".

Value	Description
0	The reset has not been detected.
1	The reset has been detected.

[bit25] Reserved: Reserved bit

**[bit24] CSV MOR: Clock supervisor reset (main clock) detection bit**

This bit is used to confirm detection of the main clock supervisor reset. Before the next reset is generated, clear this by writing "0".

Value	Description
0	The reset has not been detected.
1	The reset has been detected.

**[bit23:22] Reserved: Reserved bits**

**[bit21] SHRST1: Software trigger hardware reset detection bit**

This bit is used to confirm detection of the CPU1 software trigger hardware reset. Before the next reset is generated, clear this by writing "0".

Value	Description
0	The reset has not been detected.
1	The reset has been detected.

**[bit20] SHRST0: Software trigger hardware reset detection bit**

This bit is used to confirm detection of the CPU0 software trigger hardware reset. Before the next reset is generated, clear this by writing "0".

Value	Description
0	The reset has not been detected.
1	The reset has been detected.

**[bit19:18] Reserved: Reserved bits**

**[bit17] SRST1: Software reset detection bit**

This bit is used to confirm detection of the CPU1 software reset. Before the next reset is generated, clear this by writing "0".

Value	Description
0	The reset has not been detected.
1	The reset has been detected.

**[bit16] SRST0: Software reset detection bit**

This bit is used to confirm detection of the CPU0 software reset. Before the next reset is generated, clear this by writing "0".

Value	Description
0	The reset has not been detected.
1	The reset has been detected.

**[bit15:14] Reserved: Reserved bits**



**[bit13] SWDR1: Software watchdog reset detection bit**

This bit is used to confirm detection of the CPU1 software watchdog reset. Before the next reset is generated, clear this by writing "0".

Value	Description
0	The reset has not been detected.
1	The reset has been detected.

**[bit12] SWDR0: Software watchdog reset detection bit**

This bit is used to confirm detection of the CPU0 software watchdog reset. Before the next reset is generated, clear this by writing "0".

Value	Description
0	The reset has not been detected.
1	The reset has been detected.

**[bit11] HWDR: Hardware watchdog reset detection bit**

This bit is used to confirm detection of the hardware watchdog reset. Before the next reset is generated, clear this by writing "0".

bit	Description
0	The reset has not been detected.
1	The reset has been detected.

**[bit10] PRFERR: Profile error reset detection bit**

This bit is used to confirm detection of the profile error reset. Before the next reset is generated, clear this by writing "0".

Value	Description
0	The reset has not been detected.
1	The reset has been detected.

**[bit9] SRSTX: nSRST pin input reset detection bit**

This bit is used to confirm detection of the nSRST pin input reset. Before the next reset is generated, clear this by writing "0".

Value	Description
0	The reset has not been detected.
1	The reset has been detected.

**[bit8:7] Reserved: Reserved bits**

**[bit6] LVD50R: External power supply (5.0V) low-voltage detection reset detection bit**

This bit is used to confirm detection of the external power supply (5.0V) low-voltage detection reset. Before the next reset is generated, clear this by writing "0".

Value	Description
0	The reset has not been detected.
1	The reset has been detected.

**[bit5] Reserved: Reserved bit**

**[bit4] RSTX: RSTX pin input reset detection bit**

This bit is used to confirm detection of the RSTX pin input reset. Before the next reset is generated, clear this by writing "0".

Value	Description
0	The reset has not been detected.
1	The reset has been detected.

**[bit3] CKTOR: Clock stop wait timeout reset detection bit**

This bit is used to confirm detection of the clock stop wait timeout reset. Before the next reset is generated, clear this by writing "0".

Value	Description
0	The reset has not been detected.
1	The reset has been detected.

**[bit2] INITX: INITX input reset detection bit**

This bit is used to confirm detection of the INITX input reset. Before the next reset is generated, clear this by writing "0".

Value	Description
0	The reset has not been detected.
1	The reset has been detected.

**[bit1] LVD12R: Internal power supply low-voltage reset detection bit**

This bit is used to confirm detection of the internal power supply low-voltage detection reset. Before the next reset is generated, clear this by writing "0". When reset is generated by LVD12R, the PONR (bit0) detection bit is also set.

Value	Description
0	The reset has not been detected.
1	The reset has been detected.

**[bit0] PONR: Power-on reset detection bit**

This bit is used to confirm detection of the power-on reset. Before the next reset is generated, clear this by writing "0".

Value	Description
0	The reset has not been detected.
1	The reset has been detected.

**Notes:**

- Protection key code input is required for writing.
- It results bus error response for the violation to above restriction.
- In multi-CPU mode, this register can be subject to write access by CPU0 only. Read access has no limits.





## 4.6. BootROM Extended CSV Reset Factor Register (SYSC\_EXCSVRSTCAUSEBT)

This register indicates the last reset factor. Clear reset factor before the next reset. If not cleared, it will be difficult to identify the last reset factors because the register bits are overwritten by new reset factors. The register bits are initialized by the power-on reset (PONR). Protection key code input is required for writing.

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,W0
PROT_TYPE	WPS
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	CSVSSR							
ACCESS_TYPE	R0,W0							R,W
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							

**[bit31:8] Reserved: Reserved bits**

### **[bit7:0] CSVSSR[7:0]: Clock supervisor reset (sub-system PLL clock) detection bits**

This bit is used to confirm detection of the clock supervisor reset (sub-system PLL clock). Before the next reset is generated, clear this by writing "0".

Value	Description
0	The reset has not been detected.
1	The reset has been detected.

CSVSSR[0] is detection bit of PLL for FlexRay/RDC clock supervisor reset.

CSVSSR[7:1] are reserved bits.

#### **Notes:**

- Protection key code input is required for writing.
- It results bus error response for the violation to above restriction.
- In multi-CPU mode, this register can be subject to write access by CPU0 only. Read access has no limits.

## 4.7. Watchdog Reset Boot CPU Selection Register (SYSC\_WRBOOTCPUSEL)

This register judges whether CPU0 is operating abnormally (or whether it is faulty) by monitoring the number of consecutive resets of the hardware watchdog and software watchdog 0. Set the number of consecutive resets with which CPU0 is judged to be operating abnormally. It is possible to select the CPU to activate when this number is reached. If the condition is satisfied, the selected CPU will be forcibly activated. This register can be initialized with any of the following hardware resets: power-on reset (PONR), internal low-voltage detection (LVD12R), and external pin INITX. Rewriting requires the input of a protection key code for each register access.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,W0							
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved	CPUBS	Reserved				HWRCLR	SWRCLR
ACCESS_TYPE	R0,W0	R/W	R0,W0				R0,W	R0,W
PROT_TYPE	WPS							
INITIAL_VALUE	0	1	0000				0	0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	HWRCC				HWRCS			
ACCESS_TYPE	R/W				R,WX			
PROT_TYPE	WPS							
INITIAL_VALUE	1000				0000			

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	SWRCC				SWRCS			
ACCESS_TYPE	R/W				R,WX			
PROT_TYPE	WPS							
INITIAL_VALUE	1000				0000			

**[bit31:24] Reserved: Reserved bits**

**[bit23] Reserved: Reserved bit**



#### [bit22] CPUBS: CPU activation selection bit

This bit selects activation CPU when consecutive reset count of hardware watchdog or software watchdog match specified value.

Value	Description
0	Activate CPU0
1	Activate CPU1 (initial value)

In addition, relation between CPUMD and HALT registers are the following.

CPU Operation	CPUMD[2:0] Register <sup>*2</sup>	WDG Reset Number of Occurrences	CPUBS Register	HALT[1:0] Register Initial Value <sup>*1</sup>	Remarks
Multi-CPU	0XX	Match specified value	0	10	Activate CPU0
			1	01	Activate CPU1
		Not match or no compare operation	Don't care	10	Activate CPU0
Single-CPU	100	Don't care	Don't care	10	1CPU0 mode
	101			01	1CPU1 mode

X: don't care

\*1: HALT[1:0] = SYSC\_SPECPUFCFR:HEN[1:0]

\*2: CPUMD[2:0] = SYSC\_SPECPUFCFR:CPUMD[2:0]

#### [bit21:18] Reserved: Reserved bits

#### [bit17] HWRCLR: Hardware watchdog reset counter clear bit

Writing "1" to this bit clears hardware watchdog reset counter.

Writing "0" to this bit has no effect. The read value is always "0".

#### [bit16] SWRCLR: Software watchdog reset counter clear bit

Writing "1" to this bit clears software watchdog 0 reset counter.

Writing "0" to this bit has no effect. The read value is always "0".

#### [bit15:12] HWRCC[3:0]: Hardware watchdog reset count compare bits

These bits set the value to compare number of consecutive hardware watchdog reset. These bits can be set 1 to 15.

Value	Description
0000	No compare operation
0001	1 time
0010	2 times
0011	3 times
0100	4 times
0101	5 times
0110	6 times
0111	7 times
1000	8 times
1001	9 times
1010	10 times
1011	11 times
1100	12 times
1101	13 times
1110	14 times
1111	15 times

When setting "0000", compare operation is disabled.

#### [bit11:8] HWRCS[3:0]: Hardware watchdog reset count display bits

These bits indicate the value to compare number of consecutive hardware watchdog reset. These bits can indicate 1 to 15.

Value	Description
0000	No occurrence
0001	1 time
0010	2 times
0011	3 times
0100	4 times
0101	5 times
0110	6 times
0111	7 times
1000	8 times
1001	9 times
1010	10 times
1011	11 times
1100	12 times
1101	13 times
1110	14 times
1111	15 times

- The judgment operation is performed immediately before reset release.
- If hardware watchdog reset detection bit (HWDR) is set, the reset count is incremented.
- If matched the values of HWRCC[3:0] bits, the values of HWRCS[3:0] bits are held.



#### [bit7:4] SWRCC[3:0]: Software watchdog 0 reset count compare bits

These bits set the value to compare number of consecutive software watchdog 0 reset. These bits can be set 1 to 15.

Value	Description
0000	No compare operation
0001	1 time
0010	2 times
0011	3 times
0100	4 times
0101	5 times
0110	6 times
0111	7 times
1000	8 times
1001	9 times
1010	10 times
1011	11 times
1100	12 times
1101	13 times
1110	14 times
1111	15 times

When setting "0000", compare operation is disabled.

#### [bit3:0] SWRCS[3:0]: Software watchdog 0 reset count display bits

These bits indicate the value to compare number of consecutive software watchdog 0 reset. These bits can indicate 1 to 15.

Value	Description
0000	No occurrence
0001	1 times
0010	2 times
0011	3 times
0100	4 times
0101	5 times
0110	6 times
0111	7 times
1000	8 times
1001	9 times
1010	10 times
1011	11 times
1100	12 times
1101	13 times
1110	14 times
1111	15 times

- The judgment operation is performed immediately before reset release.
- If software watchdog reset detection bit (SWDR0) is set, the reset count is incremented.
- If matched the values of SWRCC[3:0] bits, the values of SWRCS[3:0] bits are held.

**Notes:**

- *Protection key code input is required for writing.*
- *It results bus error response for the violation to above restriction.*
- *This register can be accessed by CPU0 and CPU1.*





## CHAPTER 5: Clock System

This chapter explains the clock system.

---

1. Overview
2. Configuration
3. Operation
4. Setting Procedure Example
5. Registers
6. Usage Precautions





## 1. Overview

This section explains the overview of the clock system.

The clock system supplies the clocks for MCU operation.

The external/built-in oscillating clocks of the MCU are generically called source clocks. A source clock is a clock used to generate an internal operating clock for MCU operation.

The clock system generates the following source clocks.

- Fast-CR clock
- Slow-CR clock
- Main clock/Main clock divided by 2
- PLL clock

External/built-in oscillation circuits generate a fast-CR clock, slow-CR clock and main clock. A main clock divided by 2 is generated by the division of the main clock.

And the main clock and the built-in PLL oscillation circuits generate a PLL clock.

An area that uses internal operating clocks generated from the same source clock is called a clock domain. A clock domain contains 1 or more internal operating clocks.

The clock system supplies clocks to the following clock domains.

- Clock domain 0 (system area)
- Clock domain 4 (external output clock area)

The source clock used in each clock domain can be selected. In each clock domain, the selected source clock is divided, and the resulting clocks are distributed within the MCU as internal operating clocks.

### Features

The features of clock system are the following.

- 3 kinds of clock source available (fast-CR oscillation, slow-CR oscillation, and main oscillation)
- Timer used to ensure the oscillation stabilization time of a clock source
- Source clock output mask during the oscillation stabilization wait time
- Timer count (source clock timer) using 3 kinds of clock source
- 4 kinds of source clock generated (fast-CR clock, slow-CR clock, main clock/main clock divided by 2, and PLL clock)
- Stabilization wait counter used to ensure the PLL oscillation stabilization wait time
- PLL clock output mask during the oscillation stabilization wait time
- Main clock and PLL clock monitoring using CSV
- Clocks supplied to 2 kinds of clock domain (CD0 and CD4)
- System start using a fast-CR clock
- Dynamic clock switching without a reset
- Clock gating function that supports low-power consumption
- Generation of a divided clock from a source clock
- Clock stopping response during a reset

## 2. Configuration

This section explains the configuration of the clock system.

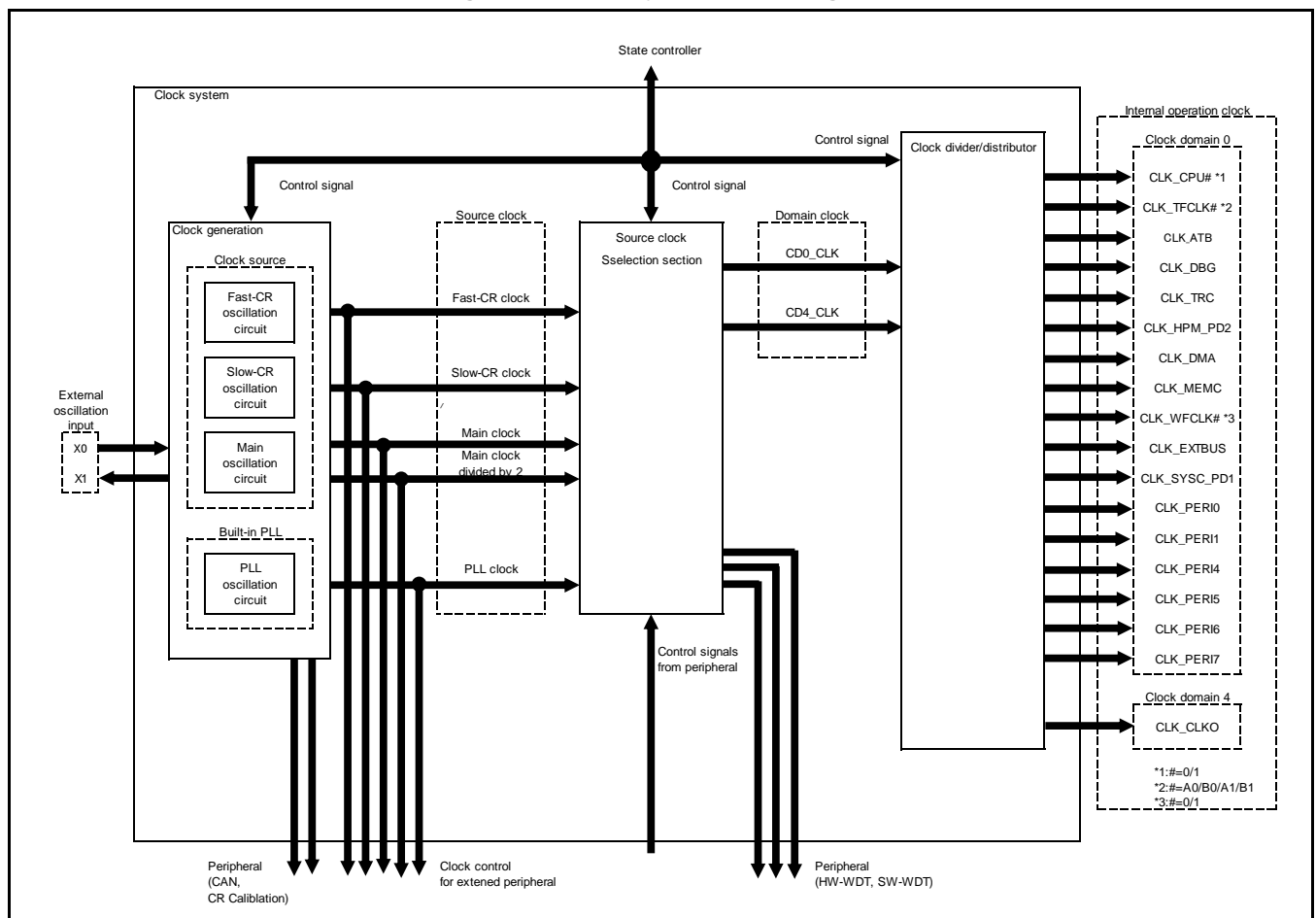
### 2.1. Overall Configuration and Block Diagrams of Clock System

This section explains the overall configuration of the clock system.

#### Block Diagram of Clock System

Figure 2-1 is a block diagram of the clock system.

Figure 2-1 Clock System Block Diagram



**a) Clock Generator**

The clock generator generates a source clock from an external/built-in oscillation circuit.

**b) Source Clock**

The term "source clock" is a generic term for the external/built-in oscillating clocks of the MCU.

- Fast-CR clock

The fast-CR clock is output from the built-in fast-CR oscillator of the MCU.

- Slow-CR clock

The slow-CR clock is output from the built-in slow-CR oscillator of the MCU.

- Main clock/Main clock divided by 2

The main clock is generated with a crystal oscillator connected to the main oscillation pins (X0 and X1). The main clock divided by 2 is the main clock that has been divided by 2.

- PLL clock

The PLL clock is generated by the PLL clock multiplication circuit (PLL oscillation circuit) multiplying the main clock.

**c) Source Clock Selection Section**

The source clock selection section selects the source clock used for each clock domain.

**d) Domain Clock**

An area that uses internal operating clocks generated from the same source clock is called a clock domain. The domain clock is a clock that is selected from the source clocks for each clock domain.

**e) Clock Divider/Distributor**

The clock divider/distributor generates an internal operating clock by dividing a domain clock. It distributes the generated internal operating clock to each resource in the MCU.

**f) Internal Operating Clock**

The internal operating clock is a clock operated by each resource in the MCU.



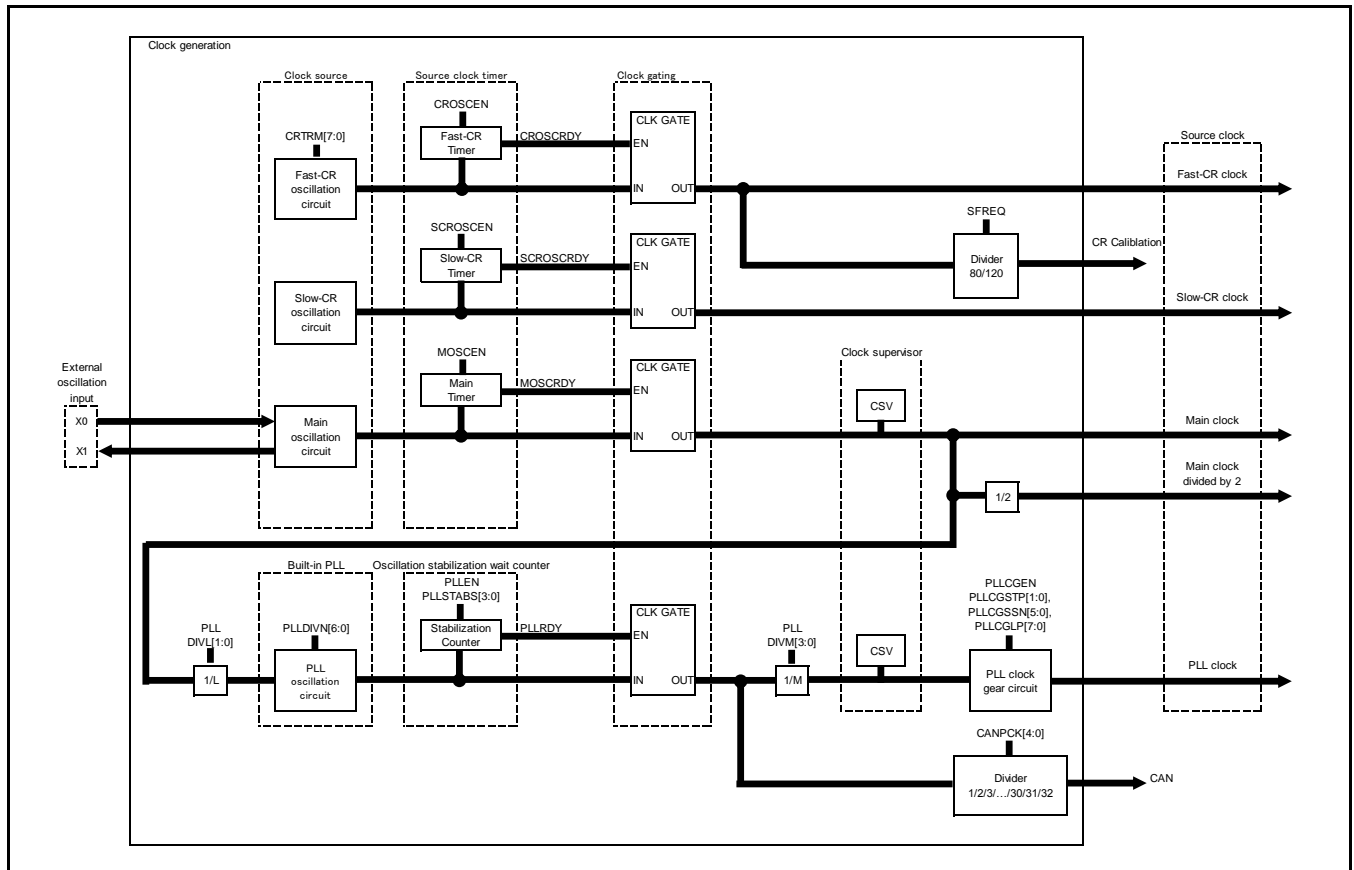
## 2.2. Configuration and Block Diagram of Clock Generator

This section explains the configuration of the clock generator.

### Block Diagram of Clock Generator

Figure 2-2 is a block diagram of the clock generator.

Figure 2-2 Block Diagram of Clock Generator



#### **a) Clock Source**

The clock source is a clock oscillation source derived from an external/built-in oscillation circuit.

- Fast-CR oscillation  
This is output from the built-in fast-CR oscillator of the MCU.
- Slow-CR oscillation  
This is output from the built-in slow-CR oscillator of the MCU.
- Main oscillation  
This is generated with a crystal oscillator connected to the main oscillation pins (X0 and X1).

#### **b) Source Clock Timer**

The source clock timer is a timer that ensures the oscillation stabilization wait time of a clock source. After the oscillation stabilizes, the clock source can use it as a timer for counting.

- Fast-CR clock timer
- Slow-CR clock timer
- Main clock timer

#### **c) Built-in PLL**

- PLL oscillation circuit  
This circuit generates a PLL clock from the main clock.

#### **d) Oscillation Stabilization Wait Counter**

This counter ensures the oscillation stabilization wait time for the built-in PLL oscillation. The counter performs a count operation with the main clock.

#### **e) Source Clock Gating Section**

The source clock gating section controls the generation of source clocks from clock sources and built-in PLL outputs. During the oscillation stabilization wait time, no source clock is output regardless of the oscillation enable setting in the relevant register.

#### **f) Clock Monitor**

This monitors source clocks excluding the fast-CR clock/slow-CR clock.

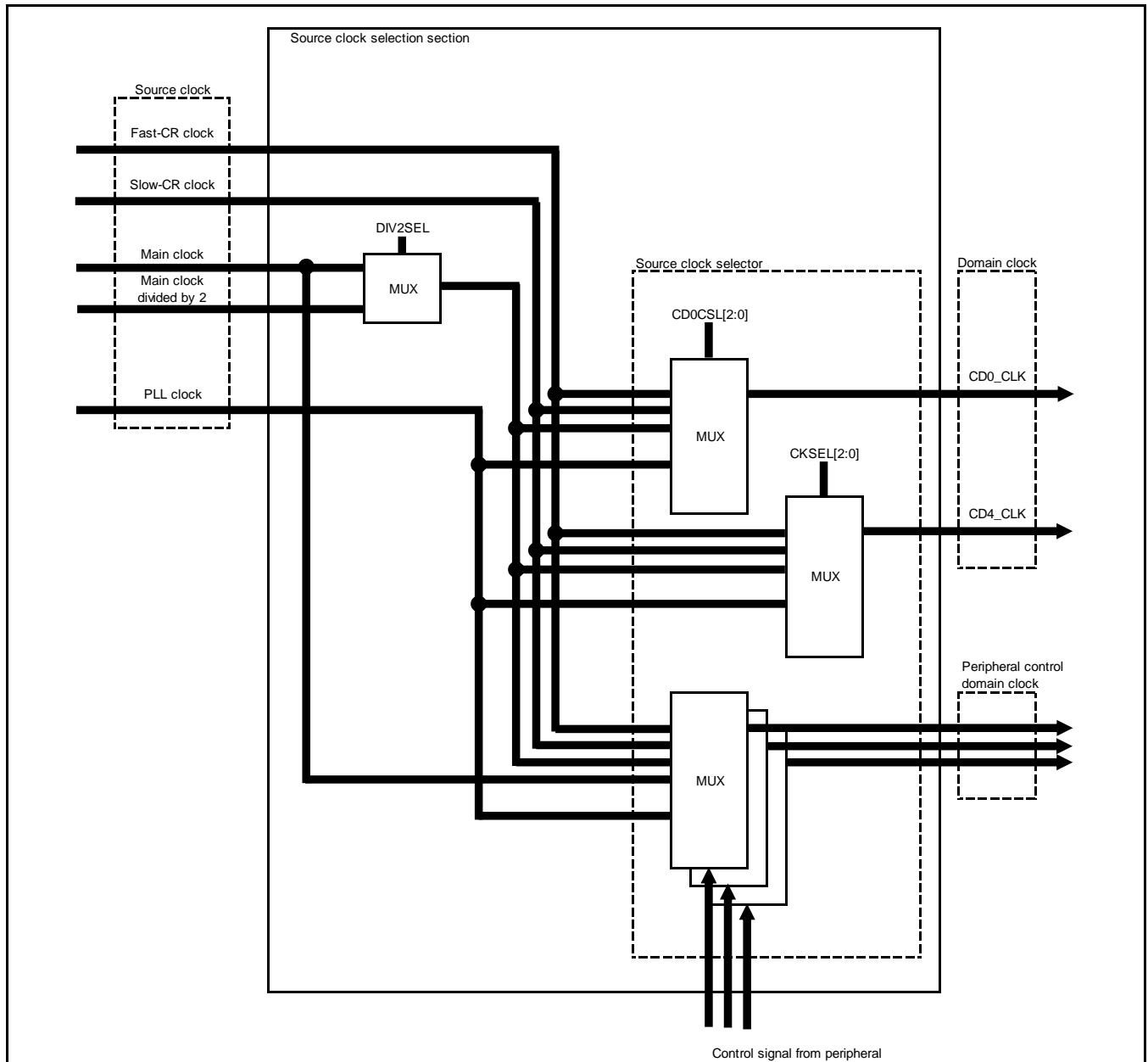
## 2.3. Configuration and Block Diagram of Source Clock Selection Section

This section explains the configuration of the source clock selection section.

### Block Diagram of Source Clock Selection Section

Figure 2-3 is a block diagram of the source clock selection section.

Figure 2-3 Block Diagram of Source Clock Selection Section



**a) Source Clock Selector**

This selects a single source clock for each clock domain from source clocks. The selected clock is output as the domain clock.

**b) Peripheral Control Domain Clock**

This clock is for a domain that is controlled independently of the clock system. For details, see "CHAPTER: Hardware Watchdog Timer", "CHAPTER: Software Watchdog Timer", and "CHAPTER: CAN Prescaler".

- Hardware watchdog timer domain clock
- Software watchdog timer domain clock 0 to 1
- CAN prescaler domain clock





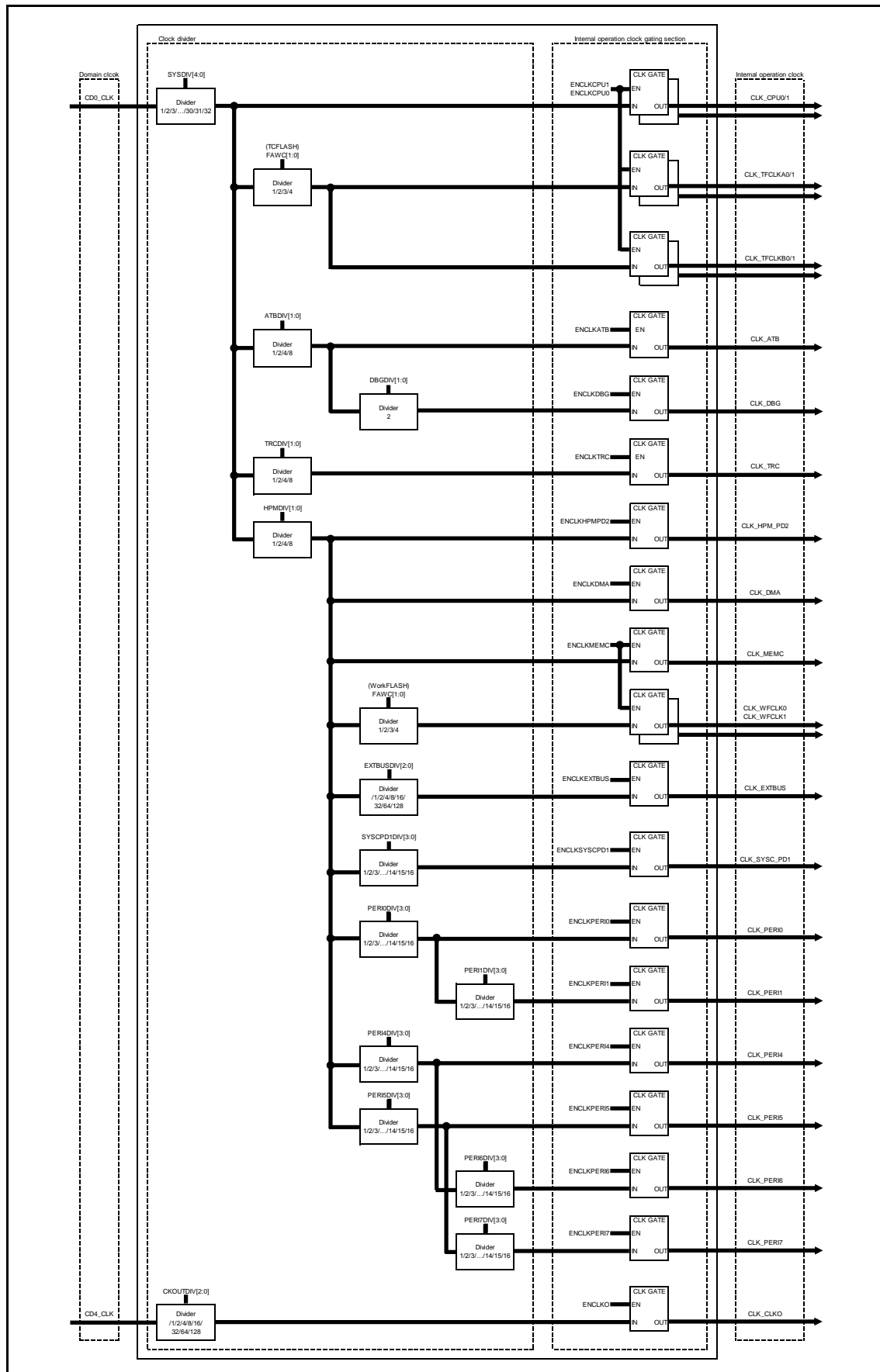
## 2.4. Configuration and Block Diagram of Clock Distributor/Divider

This section explains the configuration of the clock distributor/divider.

### Block Diagram of Clock Distributor/Divider

Figure 2-4 is a block diagram of the clock distributor/divider.

Figure 2-4 Block Diagram of Clock Distributor/Divider





**a) Clock Divider**

This generates an internal operating clock by dividing a domain clock.

**b) Internal Operating Clock Gating Section**

The internal operating clock gating section controls the enabling/disabling of the oscillation of an internal operating clock.

### 3. Operation

This section explains the operation of the clock system.

#### 3.1. Source Clock Generation Control

This section explains source clock generation control.

##### Enabling/Disabling of Source Clock Oscillation

- A hardware reset results in the source clocks from the external/built-in oscillation circuits (fast-CR clock, slow-CR clock and main clock/main clock divided by 2) entering the oscillation-enabled state. Conversely, the PLL clock enters the oscillation-disabled state.
- During system operation, the RUN/PSS profile registers can control whether the oscillation of each source clock is enabled/disabled. Control for enabling/disabling the source clock oscillation is available only when the system is not using the source clock. The fast-CR clock/slow-CR clock oscillation cannot be disabled in RUN.
- To enable PLL oscillation, the oscillation for the main clock must already be enabled. If the oscillation for the main clock has not been enabled, a profile error occurs.

**Table 3-1 Source Clock Oscillation Enabled/Disabled States**

Source Clock	Initial State	RUN	PSS
Fast-CR clock	Oscillation enabled	Oscillation enabled	Oscillation enabled/disabled can be selected
Slow-CR clock	Oscillation enabled	Oscillation enabled	Oscillation enabled/disabled can be selected
Main clock Main clock divided by 2	Oscillation enabled	Oscillation enabled/disabled can be selected	Oscillation enabled/disabled can be selected
PLL clock	Oscillation disabled	Oscillation enabled/disabled can be selected	Oscillation enabled/disabled can be selected



## 3.2. PLL Clock Control

This section explains PLL clock control.

The clock system has a built-in PLL oscillation circuit, and the PLL clock is generated from the main clock. For the PLL oscillation circuit, oscillation can be enabled/disabled, and the stabilization wait time and multiplication can be set.

### (1) Explanation of PLL Operation

This section explains PLL clock operations.

1. The oscillation stabilization wait time control register of the PLL clock sets the oscillation stabilization wait time for the PLL clock.
2. By enabling the oscillation enable bit in the clock source oscillation enable register (SYSC\_CKSRE) of the RUN/PSS profile registers, the PLL circuit starts oscillation.
3. After the oscillation stabilization wait time of the PLL clock elapses, the PLLRDY bit in the status clock source enable register shows a stable state, thereby completing the preparation for a change to the PLL clock.
4. The clock selection register of the RUN/PSS profile registers selects the PLL clock to use.

### (2) Oscillation Stabilization Wait Time Setting of PLL Clock

See Section "5.1.3. PLL Stabilization Time Control Register (SYSC\_PLLSTCNTR)".

### (3) PLL Clock Multiplication Rate Setting

(RUN/PSS) PLLCNTR is used to set the input clock divider, PLL multiplication rate, and PLL output division. The following table shows examples of the settings.

Table 3-2 PLL Clock Setting Examples

Input Clock	Input Clock Divider Setting	PLL <sub>in</sub>	PLL Multiplier Setting	PLL <sub>out</sub>	PLL Output Divider Setting	PLL Clock
4MHz	1	4MHz	100	400MHz	2	200MHz
5MHz	1	5MHz	80	400MHz	2	200MHz
6MHz	1	6MHz	60	360MHz	2	180MHz
8MHz	1	8MHz	50	400MHz	2	200MHz
8MHz	2	4MHz	100	400MHz	2	200MHz
10MHz	1	10MHz	40	400MHz	2	200MHz
10MHz	2	5MHz	80	400MHz	2	200MHz
12MHz	1	12MHz	30	360MHz	2	180MHz
12MHz	2	6MHz	60	360MHz	2	180MHz
15MHz	1	15MHz	24	360MHz	2	180MHz
16MHz	1	16MHz	25	400MHz	2	200MHz
16MHz	2	8MHz	50	400MHz	2	200MHz
16MHz	4	4MHz	100	400MHz	2	200MHz
20MHz	2	10MHz	40	400MHz	2	200MHz
20MHz	4	5MHz	80	400MHz	2	200MHz

**Note:**

- Set the division of the input clock such that the PLL input clock (PLL<sub>in</sub>) is from 4 MHz to 16 MHz. Set the multiplication such that the PLL output clock (PLL<sub>out</sub>) is from 200 MHz to 400 MHz.



### 3.3. Source Clock Selection Control

This section explains source clock selection control.

#### Source Clock Selection

A hardware reset results in the fast-CR clock being selected as the clock used in all clock domains.

During system operation, the source clock used for each clock domain can be selected using the RUN/PSS profile registers.

**Table 3-3 Clock Selection States**

Clock Domain	Domain Clock	Resource	Initial State	RUN	PSS
CD0	CD0_CLK	CPU, TRC, DBG High Performance Matrix MCU Config Group Memory & Config Group Common Peripheral Group Application Specific Peripheral Group (Slave-A, Slave-B)	Fast-CR clock	Clock can be selected	
CD4	CD4_CLK	Clock output function	Fast-CR clock	Clock can be selected	

### 3.4. Clock Distribution/Division Control

This section explains clock distribution/division control.

#### Clock Distribution/Division Control

The clock path of each clock domain has 1 to 4 division circuits. These division circuits distribute a divided clock as an internal operating clock.

During system operation, a divided clock can be generated with the RUN/PSS profile registers.

The divided clock from a source clock is distributed as an internal operating clock.

**Table 3-4 Clock Divider Settings and Maximum Operating Frequencies**

Clock Domain	Internal Operating Clock	Resource	Initial State	RUN/PSS State	Maximum Operating Frequency
CD0	CLK_CPU# (# =0 to 1)	CPU	1	P	200MHz
	CLK_TFCLKA# (# =0 to 1)	TCFLASH	4	P	80MHz
	CLK_TFCLKB# (# =0 to 1)	TCFLASH		P	80MHz
	CLK_ATB	ATB	1	P	100MHz
	CLK_DBG	DBG	2	2	50MHz
	CLK_TRC	TRC	1	P	100MHz
	CLK_HPMPD2	High Performance Matrix	1	P	200MHz
	CLK_DMA	DMA			200MHz
	CLK_MEMC	Memory & Config Group			200MHz
	CLK_WFCLK# (# = 0 to 1)	WorkFLASH	4	P	80MHz
	CLK_SYSCPD1	MCU Config Group	1	P	100MHz
	CLK_PERI0	Common Peripheral Group	1	P	100MHz
	CLK_PERI1	Common Peripheral Group	1	P	50MHz
	CLK_PERI4	Application Specific Peripheral Group (Slave-A)	1	P	100MHz
	CLK_PERI6	Application Specific Peripheral Group (Slave-A)	1	P	50MHz
	CLK_PERI5	Application Specific Peripheral Group (Slave-B)	1	P	100MHz
	CLK_PERI7	Application Specific Peripheral Group (Slave-B)	1	P	50MHz
CD4	CLK_CLKO	Clock output function	1	P	200MHz

P: Setting possible (programmable)





**Notes:**

- *Set the division such that no supplied clock exceeds the maximum operating frequency of any internal operating clock.*
- *The MB9D560 series does not use CLK\_TRC.*

### 3.5. Clock Gear Operation

This section explains the operation of clock gears.

The frequency fluctuates abruptly at the time of switching from the main clock to the PLL clock or from the PLL clock to the main clock, so the power supply current also fluctuates considerably. Make sure to use clock gear to prevent any possible occurrence of overshoot/undershoot of the power supply current at the clock switching time.

#### **Clock Gear Control**

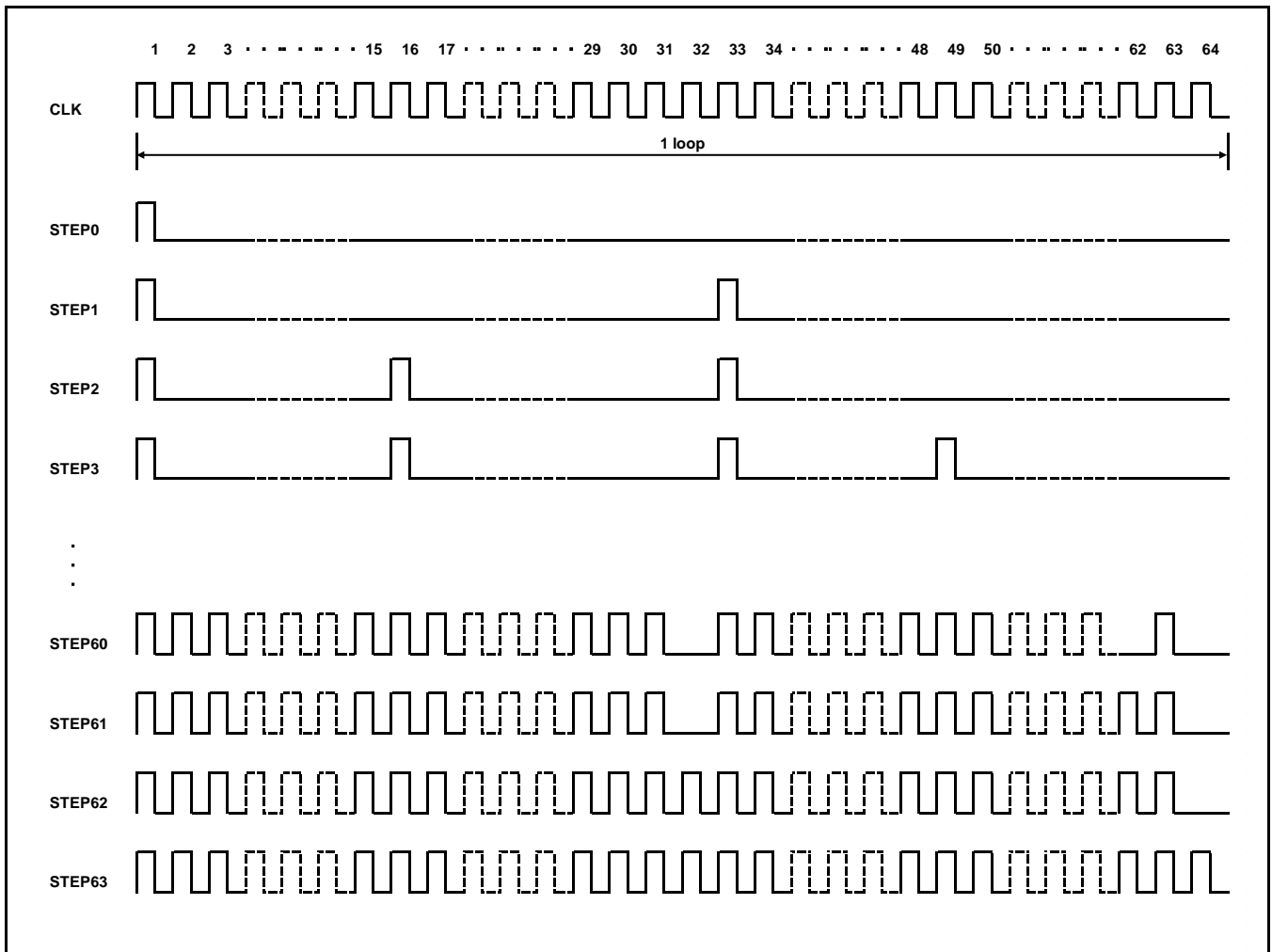
The clock that is output from the clock gear circuit is called a gear clock. The frequency of the gear clock can be gradually changed by step-by-step output of the input clock to the clock gear circuit.

The gear clock can be controlled through various settings, with 64 cycles of the clock gear circuit input clock regarded as a single control unit.

- Start step setting  
This sets the start step at the start time of gear clock output.
- Step loop setting  
This sets the loop count for each step at the gear up/down time.
- Step width setting  
This sets the step width that varies at the gear up/down time.



Figure 3-1 Gear Clock Output



### 3.5.1. Clock Gear Up Procedure

This section explains clock gear up procedure.

1. Set the clock gear operation enable setting (PLLCGCNTR:PLLCGEN) to 1.
2. After the oscillation stabilization wait time of the clock gear input clock elapses, output the clock for the start step that has been set for clock gear start step selection.
3. Select the PLL clock as the domain clock, and then set the clock gear up start (PLLCGCNTR:PLLCGSTR) to 1. The clock gear status flag (PLLCGCNTR:PLLCGSTS[1:0]) changes from "00" to "01" when a rising edge is detected. (Gear-up start)
4. Perform the gear-up operation according to the clock gear and step width settings and the loop count setting. The smaller the step width and the larger the loop count, the more gradually the frequency changes.
5. The clock gear status flag (PLLCGCNTR:PLLCGSTS[1:0]) changes from "01" to "10" when the clock reaches the maximum steps. (Gear-up end and gear stop)  
After that, output the clock with the maximum steps (63 steps).
6. After the gear stop , the clock gear start bit (PLLCGCNTR:PLLCGSTR) is automatically cleared to "0".

**Note:**

- *During the clock gear up/down operation, poll the value of the clock gear status flag, and wait until the clock indicates the low-speed/high-speed stop state.*



### 3.5.2. Clock Gear Down Procedure

This section explains clock gear down procedure.

1. Set the clock gear-down start (PLLCGCNTR:PLLCGSTR) to "1." The clock gear status flag (PLLCGCNTR:PLLCGSTS[1:0]) changes from "10" to "11" when a rising edge is detected. (gear-down start)
2. Perform the gear-down operation according to the clock gear and step width settings and the loop count setting. The smaller the step width and the larger the loop count, the more gradually the frequency changes.
3. The clock gear status flag (PLLCGCNTR:PLLCGSTS[1:0]) changes from "11" to "00" when the clock reaches the minimum steps (Gear-down end and gear stop). After that, output the clock with the step that has been set for the clock gear start step.
4. After the gear stop operation, the clock gear start bit (PLLCGCNTR:PLLCGSTR) is automatically cleared to "0".

**Note:**

- *During the clock gear up/down operation, poll the value of the clock gear status flag, and wait until the clock indicates the low-speed/high-speed stop state.*

### 3.6. Oscillation Stabilization Wait Time

This section explains oscillation stabilization wait time.

Source clock needs to wait for stabilization when source clock is not ready. During the oscillation stabilization wait time (RDY is "0"), only the built-in time counter operates, and the supply of the source clock stops until the oscillation stabilization wait time elapses. After the oscillation stabilization wait time elapses, the corresponding oscillator becomes ready (RDY is "1") and can be used as the source clock for each clock domain.

For more details see "CHAPTER: Source Clock Timer".

### 3.7. Interrupt Factor

This section explains interrupt factor of clock system.

#### Interrupt Factor

The clock system has the following interrupt factor.

- Clock error detection interrupt  
For more details, see "CHAPTER: Clock Supervisor".
- Invalid clock setting detection interrupt  
For more details, see "CHAPTER: Low-power Consumption".



## **4. Setting Procedure Example**

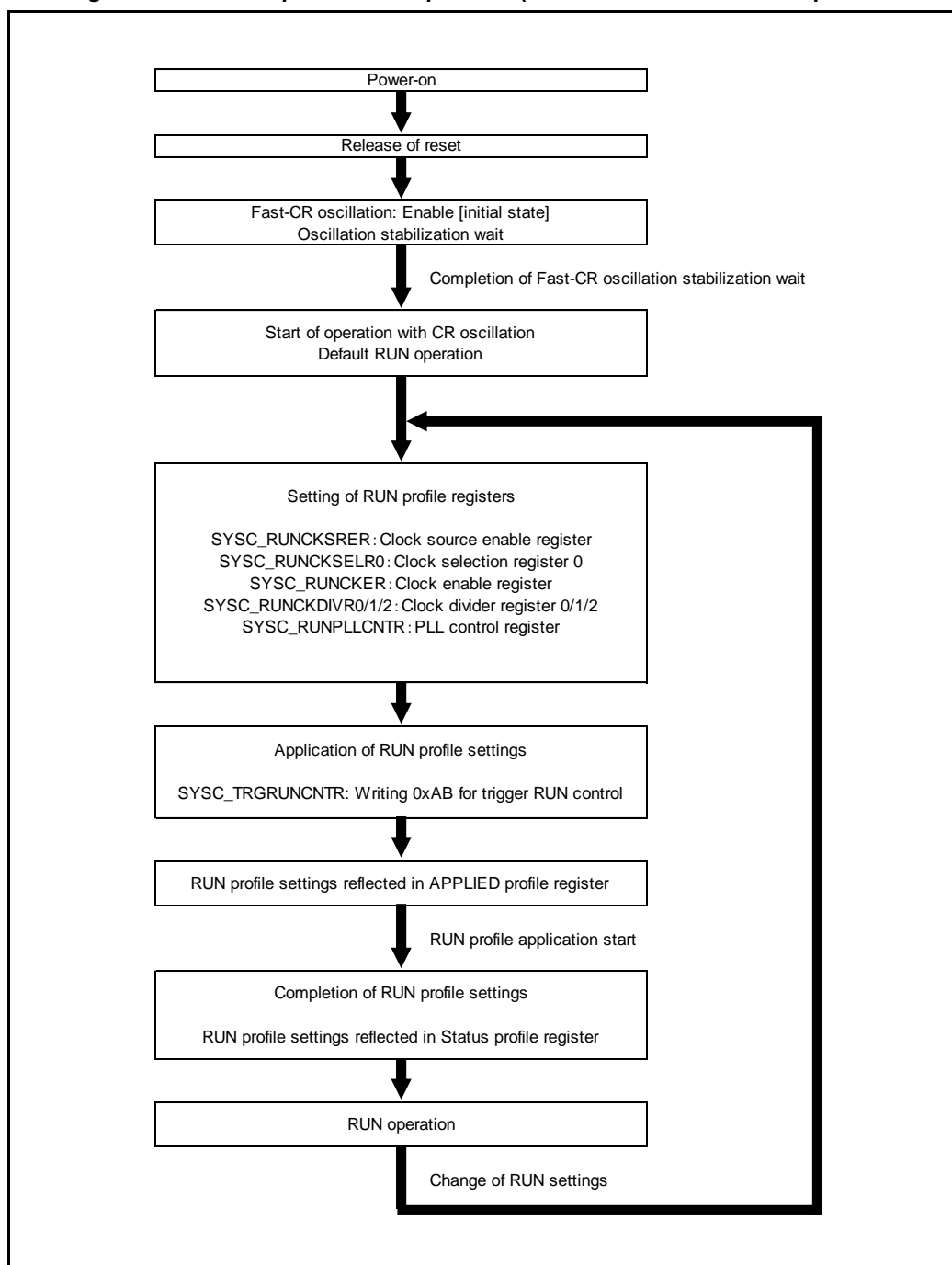
This section explains a setting procedure example of the clock system.

## 4.1. RUN Operation Setting

This section explains RUN operation setting.

### Setting Procedure Example for RUN Operation

Figure 4-1 Setting Procedure Example for RUN Operation (Power-on → Default RUN Operation → RUN Operation)



**Note:**

- To use the main clock and the PLL clock, the clock supervisor must also be set. For details, see "CHAPTER: Clock Supervisor".

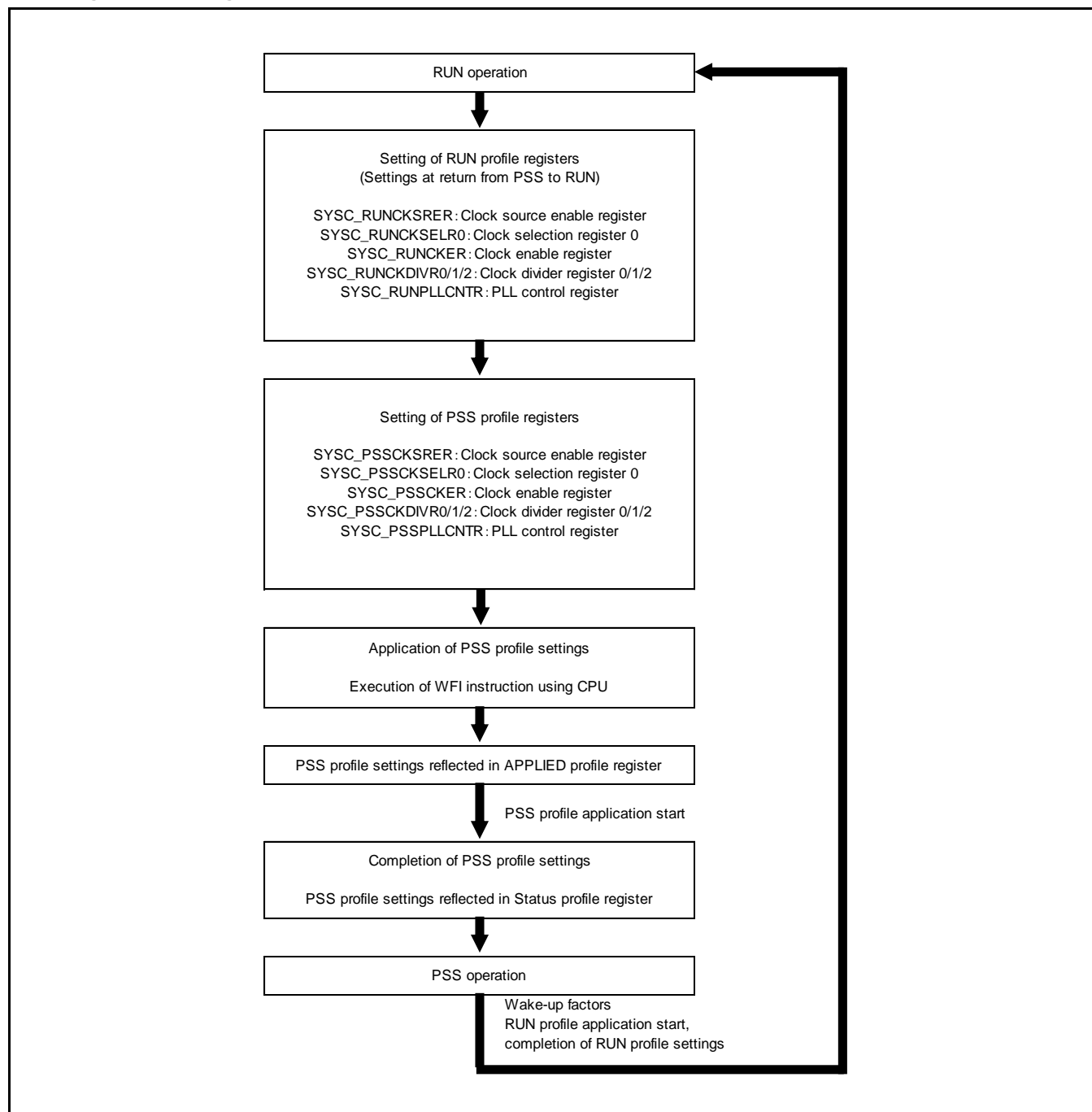


## 4.2. PSS Operation Setting

This section explains PSS operation setting.

### Setting Procedure Example for PSS Operation

Figure 4-2 Setting Procedure Example for PSS Operation (Power-on → PSS Operation → RUN Operation)



**Note:**

- To use the main clock and the PLL clock, the clock supervisor must also be set. For details, see "CHAPTER: Clock Supervisor".

## 5. Registers

This section explains the registers of clock system.

There are 6 register groups in clock system.

- RUN profile registers
- PSS profile registers
- APPLIED profile registers
- Status profile registers
- Common Configuration registers
- Clock Output Function register

For more details about Run profile registers and PSS profile registers, APPLIED profile registers and Status profile registers see "CHAPTER: Low-power Consumption".

### (1) Registers of Clock System (Common Configuration Registers)

**Table 5-1 List of Clock System Registers (Common Configuration Registers)**

Abbreviated Register Name	Register Name	See
SYSC_CRCNTR	Fast-CR Clock Control Register	5.1.1
SYSC_MOSCCNTR	Main Oscillation Control Register	5.1.2
SYSC_PLLSTCNTR	PLL Oscillation Stabilization Time Setting Register	5.1.3
SYSC_PLLGCNTR	PLL Clock Gear Control Register	5.1.4

### (2) Registers of Clock System (Clock Output Function Register)

**Table 5-2 List of Clock System Registers (Clock Output Function Registers)**

Abbreviated Register Name	Register Name	See
SYSC_CKOTCNTR	Clock Output Function Control Register	5.2.1



## 5.1. Common Configuration Registers

The common configuration registers are used for the common settings of RUN/PSS in the MCU clock system.

### 5.1.1. Fast-CR Clock Control Register (SYSC\_CRCNTR)

CR clock control register (SYSC\_CRCNTR) controls CR clock.

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	WPS
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	CRTRM							
ACCESS_TYPE	R/W							
PROT_TYPE	WPS							
INITIAL_VALUE	01111111							

[bit31:8] Reserved: Reserved bits

#### [bit7:0] CRTRM[7:0]: Fast-CR oscillation trimming bits

These bits set the trimming frequency for fast-CR oscillation.

For details on the trimming values, see "CHAPTER: CR Calibration".

### 5.1.2. Main Oscillator Control Register(SYSC\_MOSCCNTR)

The main oscillation control register (SYSC\_MOSCCNTR) controls the main oscillation.

BIT_OFFSET	31-16
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	WPS
INITIAL_VALUE	00000000_00000000

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved							DIV2SEL
ACCESS_TYPE	R0,WX							R/W
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							1

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved							Reserved
ACCESS_TYPE	R0,WX							R/W0
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							0

[bit31:9] Reserved: Reserved bits

#### [bit8] DIV2SEL: Main clock 2 division select bit

This bit selects either no division or 2 division for the main clock used as the source clock.

Value	Description
0	Main clock (no division)
1	Main clock divided by two

[bit7:0] Reserved: Reserved bits



### 5.1.3. PLL Stabilization Time Control Register (SYSC\_PLLSTCNTR)

PLL oscillation stabilization time setting register (SYSC\_PLLSTCNTR) controls the oscillation stabilization wait time of PLL.

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	WPS
INITIAL_VALUE	00000000_00000000 00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved				PLLSTABS			
ACCESS_TYPE	R0,WX				R1,WX	R/W		
PROT_TYPE	WPS							
INITIAL_VALUE	0000				1111			

[bit31:4] Reserved: Reserved bits

#### [bit3:0] PLLSTABS[3:0]: PLL stabilization wait time selection bits

These bits select the PLL stabilization wait time.

Value	Description
1000	Stabilization time : Main clock period[s] $\times 2^9$ [cycle]
1001	Stabilization time : Main clock period[s] $\times 2^{10}$ [cycle]
1010	Stabilization time : Main clock period[s] $\times 2^{11}$ [cycle]
1011	Stabilization time : Main clock period[s] $\times 2^{12}$ [cycle]
1100	Stabilization time : Main clock period[s] $\times 2^{13}$ [cycle]
1101	Stabilization time : Main clock period[s] $\times 2^{14}$ [cycle]
1110	Stabilization time : Main clock period[s] $\times 2^{15}$ [cycle]
1111	Stabilization time : Main clock period[s] $\times 2^{16}$ [cycle]

PLLSTABS[3] bit always returns "1" during reading.

**Note:**

- These bits have to be set before PLL clock oscillation enable setting by the profile update. Do not change setting after PLL clock oscillation is set to enable.

### 5.1.4. PLL Clock Gear Control Register (SYSC\_PLLCGCNTR)

PLL clock gear control register (SYSC\_PLLCGCNTR) controls PLL clock gear.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	PLLCGLP							
ACCESS_TYPE	R/W							
PROT_TYPE	WPS							
INITIAL_VALUE	11111111							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	PLLCGSTP		PLLCGSSN					
ACCESS_TYPE	R/W		R/W					
PROT_TYPE	WPS							
INITIAL_VALUE	00		000000					

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	PLLCGSTS		Reserved				PLLCGSTR	PLLCGEN
ACCESS_TYPE	R,WX		R0,WX				R,W	R/W
PROT_TYPE	WPS							
INITIAL_VALUE	00		0000				0	1

**[bit31:24] Reserved: Reserved bits**

**[bit23:16] PLLCGLP[7:0]: PLL clock gear loop count setting bits**

These bits set the loop count per one step of PLL clock gear operation.

Value	Description
00000000	1 loop
00000001	2 loops
00000010	3 loops
...	
11111101	254 loops
11111110	255 loops
11111111	256 loops

**Note:**

- These bits have to be set before PLL clock oscillation enable setting by the profile update. Do not change setting after PLL clock oscillation is set to enable.



**[bit15:14] PLLCGSTP[1:0]: PLL clock gear step width setting bits**

These bits set the step width at the PLL clock gear-up/down.

Value	Description
00	1
01	2
10	3
11	4

**Note:**

- These bits have to be set before PLL clock oscillation enable setting by the profile update. Do not change setting after PLL clock oscillation is set to enable.

**[bit13:8] PLLCGSSN[5:0]: PLL clock gear start step setting bits**

These bits set the step at the start of PLL clock gear operation.

Value	Description
000000	0
000001	1
000010	2
...	
111101	61
111110	62
111111	63

**Note:**

- These bits have to be set before PLL clock oscillation enable setting by the profile update. Do not change setting after PLL clock oscillation is set to enable.

**[bit7:6] PLLCGSTS[1:0]: PLL clock gear status flag bits**

These bits show the PLL clock gear status.

Value	Description
00	Stopped in the low-speed oscillation state
01	Gear up operation
10	Stopped in the high-speed oscillation state
11	Gear down operation

**[bit5:2] Reserved: Reserved bits**

**[bit1] PLLCGSTR: PLL clock gear operation start bit**

This bit controls the start of a clock gear operation.

Value	Description
0	No operation
1	Start gear operation

**Note:**

- This bit is cleared to "0" after starting clock gear operation.

**[bit0] PLLCGEN: PLL clock gear operation enable bit**

This bit sets enable/disable of clock gear operation.

Value	Description
0	Clock gear operation disabled
1	Clock gear operation enabled

**Note:**

- These bits have to be set before PLL clock oscillation enable setting by the profile update. Do not change setting after PLL clock oscillation is set to enable.





## 5.2. Clock Output Function Register

Clock Output Function register is used for various settings of the clock output function.

### 5.2.1. Clock Output Function Control Register (SYSC\_CKOTCNTR)

Clock output function control register (SYSC\_CKOTCNTR) selects a clock in the clock output function and sets the division.

BITS	31	30	29	28	27	26	25	24
BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							ENCLKO
ACCESS_TYPE	R0,WX							R/W
PROT_TYPE	WPS							
INITIAL_VALUE	0000000							0

BITS	23	22	21	20	19	18	17	16
BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							

BITS	15	14	13	12	11	10	9	8
BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved					CKOUTDIV		
ACCESS_TYPE	R0,WX					R/W		
PROT_TYPE	WPS							
INITIAL_VALUE	00000					000		

BITS	7	6	5	4	3	2	1	0
BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved					CKSEL		
ACCESS_TYPE	R0,WX					R/W		
PROT_TYPE	WPS							
INITIAL_VALUE	00000					000		

[bit31:25] Reserved: Reserved bits

[bit24] ENCLKO: Output enable bit of an external output clock

This bit sets the output enable/disable of an external output clock.

Value	Description
0	Disables output of an external output clock
1	Enables output of an external output clock

[bit23:11] Reserved: Reserved bits

**[bit10:8] CKOUTDIV[2:0]: Division setting bits of an external output clock**

These bits set the division ratio of an external output clock from a source clock.

Value	Description
000	No division
001	2 division
010	4 division
011	8 division
100	16 division
101	32 division
110	64 division
111	128 division

**[bit7:3] Reserved: Reserved bits**

**[bit2:0] CKSEL[2:0]: External output clock select bits**

These bits select the source clock for an external output clock.

Value	Description
000	Fast-CR clock is selected
001	Slow-CR clock is selected
010	Main clock selected/main clock divided by 2 selected
011	Clock fixed at "L"
100	PLL clock is selected
101	Clock fixed at "L"
110	Clock fixed at "L"
111	Clock fixed at "L"



## 6. Usage Precautions

The section explains the precautions when using the clock system.

- To stop the clock in clock domain 0 with a PSS profile, stop the clock with PSS clock selection register 0 (SYSC\_PSSCKSELR0) and simultaneously disable the oscillation of all internal operating clocks with the PSS clock enable register (SYSC\_PSSCKER).
- The change of the PLL setting
  - To change the PLL oscillation settings, stop the PLL oscillation and change the division ratio. After that, enable the oscillation of PLL again.
  - The PLL oscillation stabilization time have to be set before PLL clock oscillation enable setting by the profile update. Do not change setting after PLL clock oscillation is set to enable.
- The change of the clock gear setting
  - The PLL clock gear has to be set before PLL clock oscillation enable setting by the profile update. Do not change setting after PLL clock oscillation is set to enable.
- The operation of the clock gear
  - During the clock gear up/down operation, poll the value of the clock gear status flag, and wait until the clock shows the low-speed/high-speed stop state.
  - If changing to PLL clock or from PLL clock, make sure to use clock gear.
- The main clock cannot be stopped during use of the PLL clock.
- Peripheral group independent of the clock control by the clock system
  - The following peripheral functions operate independently of the clock control by the clock system. For details on the handling of each operating clock, see "CHAPTER: Hardware Watchdog Timer", "CHAPTER: Software Watchdog Timer" and "CHAPTER: CAN Prescaler".
  - Hardware watchdog timer
  - Software watchdog timer
  - CAN prescaler



## CHAPTER 6: Low-power Consumption

This chapter explains the functions and operations related to the low-power consumption.

---

1. Overview
2. Configuration
3. Operations
4. Operation Procedure
5. Registers
6. Other



## 1. Overview

This section provides an overview of the low-power consumption.

This product has a variety of power consumption settings, and can perform power consumption control according to the situation.

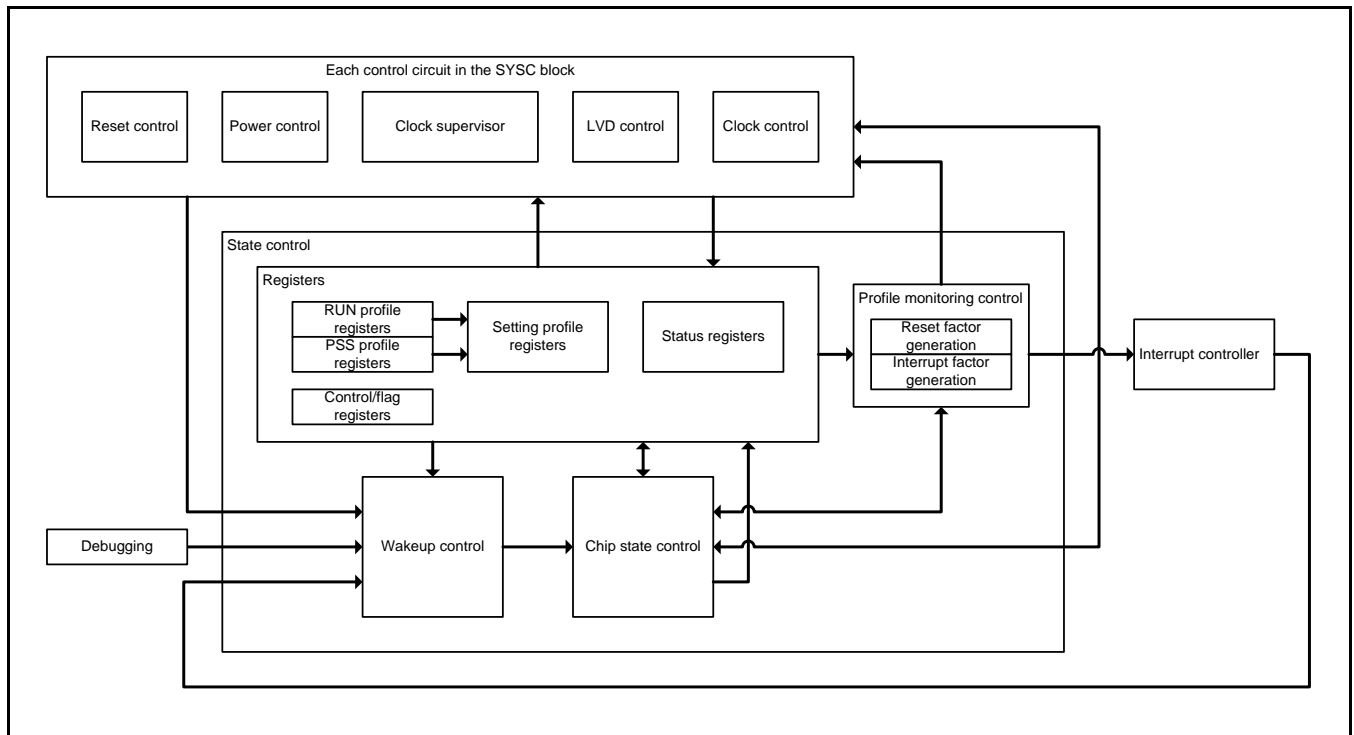
## 2. Configuration

This section explains the configuration of low-power consumption.

## 2.1. Block Diagram

This section shows the configuration of the low-power consumption control.

### Figure 2-1 Low-power Consumption Control Block Diagram



- State control

This block controls each block in the SYSC (System Controller), and controls low-power consumption mode.
- Registers

It is possible to update the RUN/PSS profile setting registers to the setting profile registers and verify the results with the status registers. For details on the registers, see Section "5. Registers".
- Wakeup control

This block controls the factors that will cause the CPU to return from the program stopped state.
- Chip state control

This block manages and controls the state of the entire chip.
- Profile monitoring control

This block monitors the contents of the RUN/PSS profiles.
- Reset/interrupt factor generation

This block generates reset and interrupt factors in the event of a profile error.



### 3. Operations

This section explains the operations related to low-power consumption.

#### 3.1. Low-power Consumption States

This section describes the low-power consumption states.

There are two major low-power consumption states.

- RUN (Normal Operation)
- PSS (Power Saving State)

The control varies depending on the CPU mode.

##### (1) During 2CPU Mode Operation

Low-power consumption control is performed for each CPU, with CPU0 being under main state control and CPU1 being under sub-state control. A group of settings is called a "profile".

The following is a schematic view of the operation modes.

**Figure 3-1 During 2CPU Mode Operation (Main State Control)**

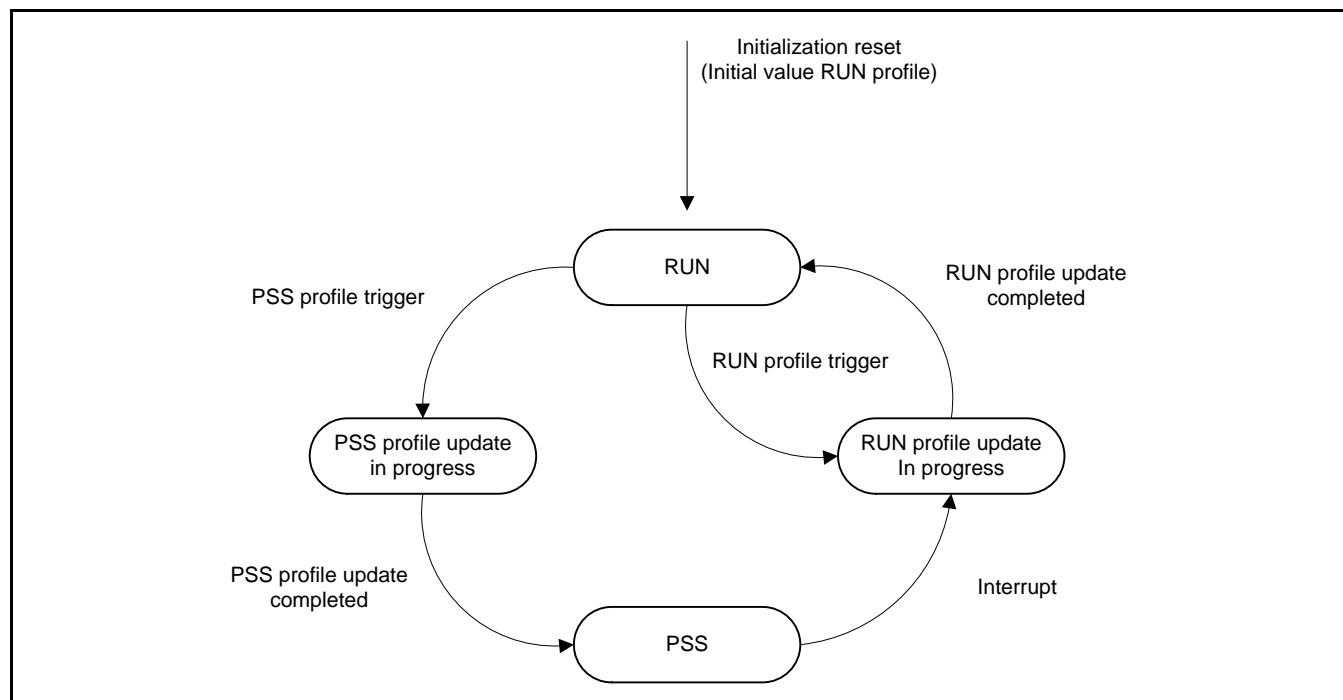
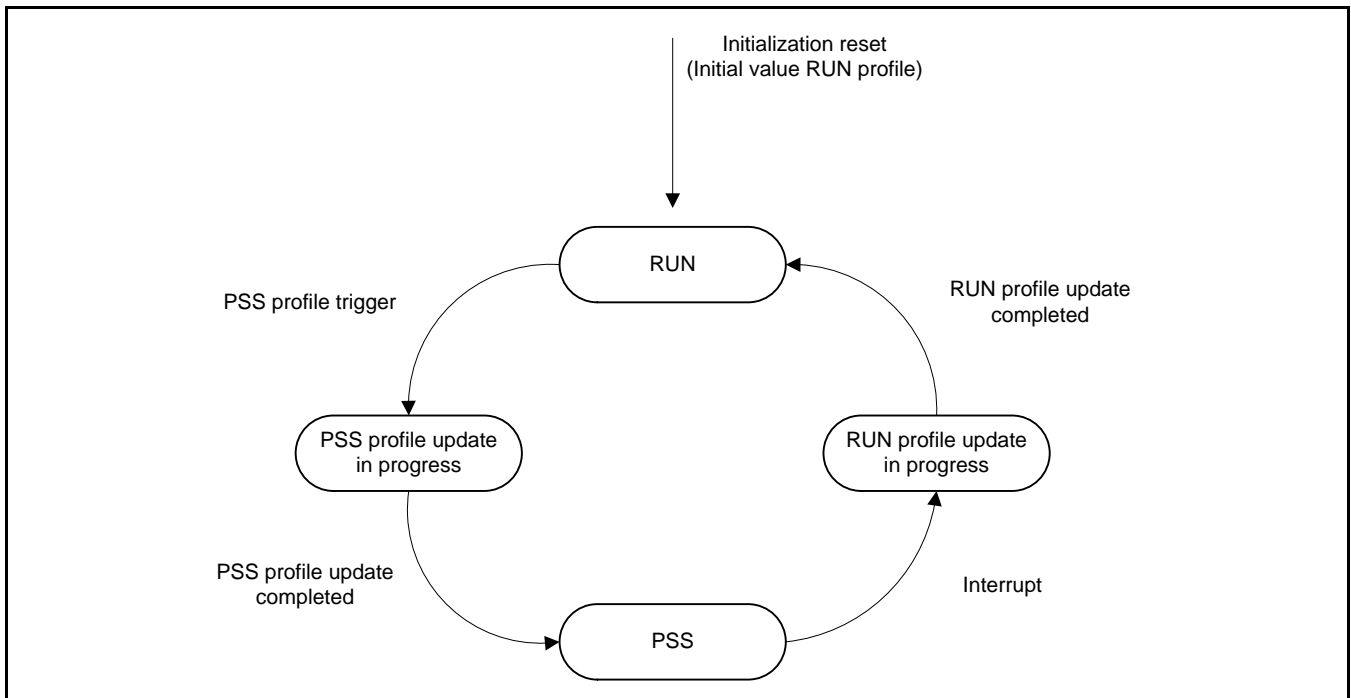


Figure 3-2 During 2CPU Mode Operation (Sub-state Control)



The difference between main state control and sub-state control is as below.

- In main state control, RUN profile update is possible with a register trigger, but in sub-state control, RUN profile update is not possible with a register trigger.
- When main state control changes to PSS, all sub-state control will also change to PSS.
- The items to set differ depending on the PSS profile.

The following items can be set freely in RUN/PSS.

- Clock-related
  - Source clock oscillation enable/stop
  - Clock domain control (source selection, division, and oscillation enable/stop of each domain clock)
- CSV settings
  - Settings (ON/OFF, interrupt or reset control)
- LVD settings

In main state control, all items can be set, but in sub-state control, only the clock operation and stop of the clock of the relevant CPU can be set.

For details on the setting items, see Section "3.3. Profile Setting Items".

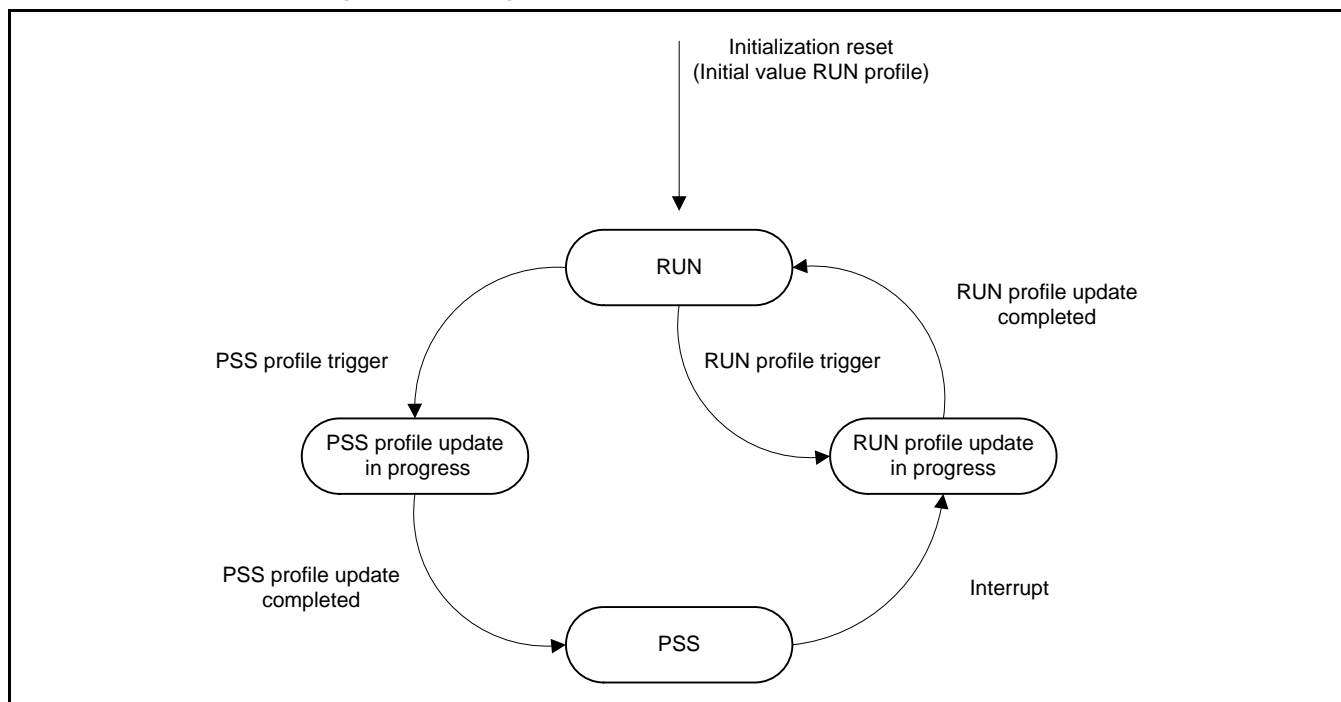


## (2) During 1CPU0/1 Mode Operation

With CPU0 or CPU1, perform main state control. Sub-state control is not performed.

The following is a schematic view of the operation modes.

Figure 3-3 During 1CPU0/1 Mode Operation (Main State Control)



Make the necessary settings in main state control.

The setting and change items are the same as those for 2CPU mode.

For details on the setting items, see Section "3.3. Profile Setting Items".

## (3) RUN (Main State Control)

In this state, the CPU is operating a program or is stopping a program.

The program stopped state is that state in which PSS enable is disabled in RUN and the WFI instruction is executed.

After an initialization reset, operation is performed in this state (program operation).

It is possible to update the RUN profile and switch from this state to PSS.

## (4) PSS (Main State Control)

In this state, the CPU is stopping a program and low-power consumption can be set.

Sub-state control is also in PSS.

It is possible to select whether to return to sub-state control with an interrupt factor when returning from PSS.

#### (5) RUN (Sub-state Control)

In this state, the CPU is operating a program or is stopping a program.

The program stopped state is that state in which PSS enable is disabled in RUN and the WFI instruction is executed.

After an initialization reset, operation is performed in this state (program operation).

It is possible to switch from this state to PSS.

#### (6) PSS (Sub-state Control)

In this state, the CPU is stopping a program and low-power consumption can be set.

Low-power consumption can be set only partially. Only the relevant CPU is in the PSS state.

#### (7) Relationship between CPU Operation Modes and State Controls

Depending on the CPU operation mode, the CPU that is subject to main and sub-state controls varies.

The following describes the relationship between the CPU operation modes and state controls.

**Table 3-1 Relationship between CPU Operation Modes and State Controls**

CPU Operation Mode	Operating Trigger or Enable	CPU Controlled	State Control Performed
2CPU mode	RUN profile update trigger	CPU0	Main state control
		CPU1	Writing disable
1CPU0 mode		CPU0	Main state control
		CPU1	No state control is performed because CPU1 is stopping.
1CPU1 mode		CPU0	No state control is performed because CPU0 is stopping.
	PSS profile enable (Sub-state control 1) (SYSC_PSSSEN:PSSEN1)	CPU1	Main state control
2CPU mode		CPU1	Sub-state control
1CPU0 mode		CPU1	No state control is performed because CPU1 is stopping.
1CPU1 mode	PSS profile enable (Main state control) (SYSC_PSSSEN:PSSEN0)	CPU1	In CPU1, sub-state control 1 is not performed because the main state is controlled.
2CPU mode		CPU0	Main state control
1CPU0 mode		CPU0	Main state control
1CPU1 mode		CPU1	Main state control

For details on CPU operation modes, see "CHAPTER: Operation Mode".

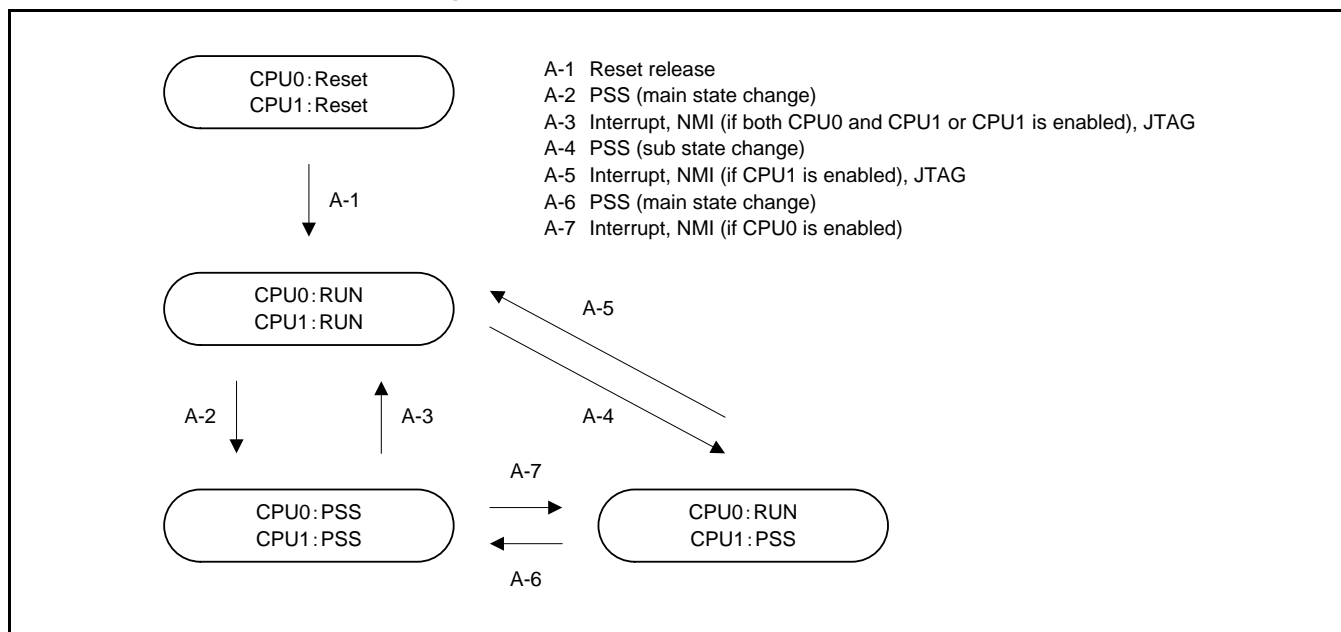


## 3.2. CPU Operation States

This section describes each CPU operation state.

### (1) 2CPU Mode

Figure 3-4 2CPU Mode Operation State Chart

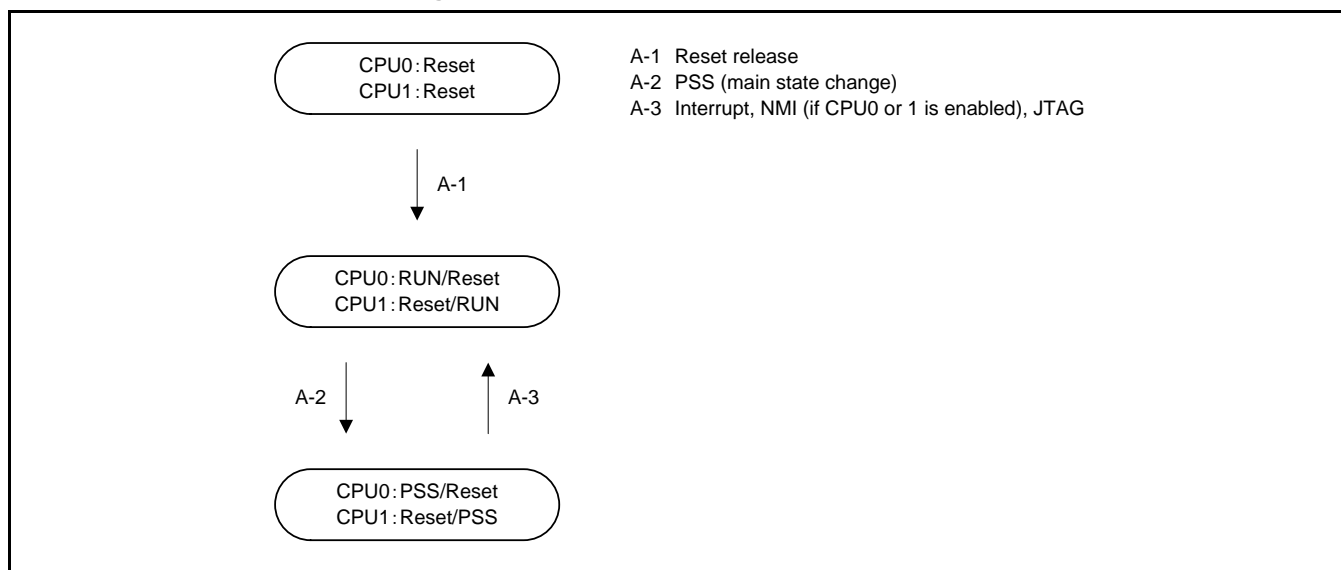


RUN: Program operation state (program operating or program stopped)

PSS: Program stopped state (low-power consumption)

### (2) 1CPU0/1 Mode

Figure 3-5 1CPU0/1 Mode Operation State Chart



RUN: Program operation state (program operating or program stopped)

PSS: Program stopped state (low-power consumption)

### 3.3. Profile Setting Items

The profile setting items are listed below.

In both RUN and PSS, they can be set freely with a program, with some exceptions.

Table 3-2 lists the parameters that can be set with RUN/PSS profiles in each mode.

**Table 3-2 RUN/PSS Profile Setting Parameter List**

Category	Setting Item	RUN Setting (Initial Value)	RUN Setting	PSS Setting	Main State Control Update Enabled Item	Sub state Control Update Enabled Item
Clock enable	PERI0 clock oscillation enable	Enabled	Programmable	Programmable	Yes	No
	PERI1 clock oscillation enable	Enabled	Programmable	Programmable	Yes	No
	PERI4 clock oscillation enable	Enabled	Programmable	Programmable	Yes	No
	PERI5 clock oscillation enable	Enabled	Programmable	Programmable	Yes	No
	PERI6 clock oscillation enable	Enabled	Programmable	Programmable	Yes	No
	PERI7 clock oscillation enable	Enabled	Programmable	Programmable	Yes	No
	SYSCPD1 clock oscillation enable	Enabled	Enabled	Programmable	Yes	No
	EXTBUS clock oscillation enable	Enabled	Programmable	Programmable	Yes	No
	MEMC clock oscillation enable	Enabled	Programmable	Programmable	Yes	No
	DMA clock oscillation enable	Enabled	Programmable	Programmable	Yes	No
	HPMPD2 clock oscillation enable	Enabled	Enabled	Programmable	Yes	No
	TRC clock oscillation enable	Enabled	Programmable	Programmable	Yes	No
	ATB clock oscillation enable	Enabled	Programmable	Programmable	Yes	No
	DBG clock oscillation enable	Enabled	Programmable	Programmable	Yes	No
	CPU1 clock oscillation enable	Enabled	Enabled	Disabled	Yes	Yes
	CPU0 clock oscillation enable	Enabled	Enabled	Disabled	Yes	No
Source clock oscillation	Slow-CR	Enabled	Enabled	Programmable	Yes	No
	Fast-CR	Enabled	Enabled	Programmable	Yes	No
	Main oscillation	Enabled	Programmable	Programmable	Yes	No
	PLL	Prohibited	Programmable	Programmable	Yes	No
Clock selection	Clock domain 0 clock selection	Fast-CR clock	Programmable	Programmable	Yes	No
Clock divider	Trace clock division	1	Programmable	Programmable	Yes	No
	ATB clock division	1	Programmable	Programmable	Yes	No
Clock divider	DBG clock division	2	2	2	Yes	No



Category	Setting Item	RUN Setting (Initial Value)	RUN Setting	PSS Setting	Main State Control Update Enabled Item	Sub state Control Update Enabled Item
	HPM clock division	1	Programmable	Programmable	Yes	No
	EXTBUS clock division	1	Programmable	Programmable	Yes	No
	SYSCPD1 clock division	1	Programmable	Programmable	Yes	No
	PERI0 clock division	1	Programmable	Programmable	Yes	No
	PERI1 clock division	1	Programmable	Programmable	Yes	No
	PERI4 clock division	1	Programmable	Programmable	Yes	No
	PERI5 clock division	1	Programmable	Programmable	Yes	No
	PERI6 clock division	1	Programmable	Programmable	Yes	No
	PERI7 clock division	1	Programmable	Programmable	Yes	No
Low-voltage detection	Setting enable LVD5.0V	Enabled	Programmable	Programmable	Yes	No
	Setting enable LVD1.2V	Enabled	Enabled	Enabled	Yes	No
	Setting voltage LVD5.0V	4.1V	Programmable	Programmable	Yes	No
	Setting voltage LVD1.2V	0.9V	0.9V	0.9V	Yes	No
	Interrupt and reset selection	Reset	Programmable	Programmable	Yes	No
PLL	L division setting	0	Programmable	Programmable	Yes	No
	M division setting	0x01	Programmable	Programmable	Yes	No
	N multiplier setting	0x0C	Programmable	Programmable	Yes	No
CSV	Main CSV enable	Stop	Programmable	Programmable	Yes	No
	PLL and CSV enable	Stop	Programmable	Programmable	Yes	No

Yes: Can be set

No: Cannot be set

### 3.4. Profiles

This section describes profiles.

For details on profiles, see Table 3-2. For profile updates, first start an update with the software and then change the settings with the control circuit. In the profile settings, a combination that presents problems is called a profile error. If there is a profile error, it is not possible to perform a profile update. For details on profile errors, see "(3) Profile Errors". The profile error state can be checked with an appropriate register.

#### (1) RUN Profile

The following RUN profile settings are fixed, and cannot be changed.

- Clock supply to CPU0 to 1
- Clock supply to HPM

#### (2) PSS Profile

The following PSS profile settings are fixed, and cannot be changed.

- Stoppage of the clock to CPU0 to 1

#### (3) Profile Errors

The following lists profile errors.

Table 3-3 Profile Error List

Type	Error Description	Target State (RUN/PSS)	Error Flag
Clock	PLL clock oscillation is enabled, and main clock oscillation is disabled	RUN/PSS	SYSC_SYSRUNPEFR:PEF0 SYSC_SYSPSSPEFR:PEF0
Clock	An oscillation-disabled clock is selected as a clock domain clock	RUN/PSS	SYSC_SYSRUNPEFR:PEF1 SYSC_SYSPSSPEFR:PEF1
Clock	In RUN, the source clock in clock domain 0 is fixed to "L"	RUN	SYSC_SYSRUNPEFR:PEF2
Clock	With PLL oscillation enabled, a PLL setting change (input division setting, multiplier setting, or output division setting) is made	RUN/PSS	SYSC_SYSRUNPEFR:PEF3 SYSC_SYSPSSPEFR:PEF3
Clock supervisor	A monitor clock or a reference clock is oscillation-disabled and is enabled for clock supervisor operation	RUN/PSS	SYSC_SYSRUNPEFR:PEF5 SYSC_SYSPSSPEFR:PEF5
Clock supervisor	When the operation-enabled clock supervisor is disabled, oscillation of the monitor clock is not stopped simultaneously	RUN/PSS	SYSC_SYSRUNPEFR:PEF6 SYSC_SYSPSSPEFR:PEF6
Clock supervisor	When the clock supervisor reset detection bit of the user reset factor register (SYSC_RSTCAUSEUR) is set, the monitor clock is selected for either clock domain	RUN/PSS	SYSC_SYSRUNPEFR:PEF7 SYSC_SYSPSSPEFR:PEF7
Clock supervisor	When the error detection flag of system error interrupt factor register 0 (SYSC_SYSEIRI0) is set, the monitor clock is selected for either clock domain	RUN/PSS	SYSC_SYSRUNPEFR:PEF8 SYSC_SYSPSSPEFR:PEF8
Software watchdog	The clock used with the software watchdog is stopped	RUN/PSS	SYSC_SYSRUNPEFR:PEF9 SYSC_SYSPSSPEFR:PEF9
Hardware watchdog	The clock used with the hardware watchdog is stopped	RUN/PSS	SYSC_SYSPSSPEFR:PEF10



**(4) PLL Clock Oscillation is Enabled, and Main Clock Oscillation is Disabled**

RUN/PSS Clock Source Enable Register (SYSC_RUN (PSS) CKSRER)			
Source Clock Oscillation Enable Bit			
PLLEN	MOSCEN	SCROSCEN	CROSCEN
1	0	X	X

X: Don't care

**(5) An Oscillation-disabled Clock is Selected as a Clock Domain Clock**

RUN/PSS Clock Source Enable Register (SYSC_RUN (PSS) CKSRER)				RUN/PSS Clock Selection Register 0 (SYSC_RUN (PSS) CKSELR0)		
Source Clock Oscillation Enable Bit				Clock Selection Bit		
PLLEN	MOSCEN	SCROSCEN	CROSCEN	CD0CSL[2:0]		
0	X	X	X	1	0	0
X	0	X	X	0	1	0
X	X	0	X	0	0	1
X	X	X	0	0	0	0

X: Don't care

**(6) In RUN, the Source Clock in Clock Domain 0 is Fixed to "L"**

RUN Clock Selection Register (SYSC_RUNCKSELR0)		
Clock Selection Bit		
CD0CSL[2:0]		
1	1	1
1	1	0
1	0	1
0	1	1

**(7) With PLL Oscillation Enabled, a PLL Setting Change (Input Division Setting, Multiplier Setting, or Output Division Setting) is Made**

If any of the following register bits is rewritten

PLL control register (SYSC\_RUN(PSS)PLLCNTR)

[bit22:16] PLLDIVN[6:0]: PLL clock N multiplier setting bits

[bit11:8] PLLDIVM[3:0]: PLL clock M division setting bits

[bit1:0] PLLDIVL[1:0]: PLL input clock division setting bits

**Note:**

- Use caution if changing PLL settings when returning from PSS to RUN. Make these settings so that no profile errors occur.





**(8) A monitor Clock or a Reference Clock is Oscillation-disabled and is Enabled for Clock Supervisor Operation**

**a) RUN profile**

Main Clock Supervisor Setting Register 11 (SYSC_CSVPMOCFGR11)	RUN Clock Source Enable Register (SYSC_RUNCKSRER)				RUN Clock Supervisor Enable Register (SYSC_RUNCSVCFGR)	
Reference Clock Selection Bit						
REFCLKSEL	PLLEN	MOSCEN	SCROSCEN	CROSCEN	PLLCVSVE	MOCSVE
X	1	0	1	1	1	X
X	0	1	1	1	1	X
X	X	0	1	1	X	1

X: Don't care

**b) PSS profile**

Main Clock Supervisor Setting Register 11 (SYSC_CSVPMOCFGR11)	PSS Clock Source Enable Register (SYSC_PSSCKSRER)				PSS Clock Supervisor Enable Register (SYSC_PSSCSVCFGR)	
Reference Clock Selection Bit						
REFCLKSEL	PLLEN	MOSCEN	SCROSCEN	CROSCEN	PLLCVSVE	MOCSVE
X	1	0	1	X	1	X
X	0	1	1	X	1	X
X	X	0	1	X	X	1
0	X	X	1	0	X	1

X: Don't care

**(9) When the Operation-enabled Clock Supervisor is Disabled, the Oscillation of the Monitor Clock is Not Stopped Simultaneously**

Status Clock Supervisor Setting Register (SYSC_STSCSVCFGR)		RUN/PSS Clock Source Enable Register (SYSC_RUN (PSS) CKSRER)				RUN/PSS Clock Supervisor Enable Register (SYSC_RUN (PSS) CSVCFGR)	
PLLCVSVE	MOCSVE	PLLEN	MOSCEN	SCROSCEN	CROSCEN	PLLCVSVE	MOCSVE
1	X	1	1	1	1	0	X
X	1	X	1	1	1	X	0

X: Don't care

**Note:**

- Use caution if turning OFF CSV when returning from PSS to RUN. Make the settings so that no profile errors occur. If an invalid setting is made, a reset will be issued.

**(10) When the Clock Supervisor Reset Detection Bit of the User Reset Factor Register (SYSC\_RSTCAUSEUR), is Set, the Monitor Clock is Selected for Either Clock Domain**

User Reset Source Register (SYSC_RSTCAUSEUR)		RUN/PSS Clock Selection Register 0 (SYSC_RUN (PSS) CKSELR0)		
Clock Supervisor Reset Detection Bit		Clock Selection Bit		
CSVPR	CSVMOR	CD0SL[2:0]		
1	X	1	0	0
X	1	0	1	0

X: Don't care

**(11) When the Error Detection Flag of System Error Interrupt Factor Register 0 (SYSC\_SYSEERRIR0) is Set, the Monitor Clock is Selected for Either Clock Domain**

System Error Interrupt Factor Register 0 (SYSC_SYSEERRIR0)		RUN/PSS Clock Selection Register 0 (SYSC_RUN (PSS) CKSELR0)		
Error Detection Flag Bit		Clock Selection Bit		
PMIF	MOMIF	CD0CSL[2:0]		
1	X	1	0	0
X	1	0	1	0

X: Don't care

**(12) The Clock Used with the Software Watchdog is Stopped**

**a) RUN Profile**

RUN Clock Source Enable Register (SYSC_RUNCKSPER Register)			Software Watchdog Configuration Register (SWDGN_CFG Register)		
MOSCEN (Main)	SCROSCEN (Slow-CR)	CROSCEN (Fast-CR)	WDENRUN (RUN Operation)	ALLOW STOPCLK (Change to PSS Accompanied by Clock Stop)	CLKSEL[1:0] (Clock Selection)
0	1	1	1	X	11

X: Don't care

**b) PSS Profile**

PSS Clock Source Enable Register (SYSC_PSSCKSRER Register)			Software Watchdog Configuration Register (SWDGN_CFG Register)		
MOSCEN (Main)	SCROSCEN (Slow-CR)	CROSCEN (Fast-CR)	WDENPSS (PSS Operation)	ALLOW STOPCLK (Change to PSS Accompanied by Clock Stop)	CLKSEL[1:0] (Clock Selection)
0	X	X	1	0	11
X	0	X	1	0	01
X	X	0	1	0	00

X: Don't care



### (13) The Clock Used with the Hardware Watchdog is Stopped

#### a) RUN Profile

There are no error conditions because the slow-CR clock/fast-CR clock do not stop.

#### b) PSS Profile

PSS Clock Source Enable Register (SYSC_PSSCKSRER Register)			Hardware Watchdog Configuration Register (HWDGn_CFG Register)		
MOSCEN (Main)	SCROSCEN (Slow-CR)	CROSCEN (Fast-CR)	WDENPSS (PSS operation)	ALLOW STOPCLK (Change to PSS Accompanied by Clock Stop)	CLKSEL[0] (Clock selection)
X	0	X	1	0	1
X	X	0	1	0	0

X: Don't care

### 3.5. Interrupts

This section explains interrupts of the base timer.

The following list interrupts that lead to low-power consumption.

**Table 3-4 Interrupt List**

Occurrence Conditions	Factor Register	Enabling Condition	Clearing Condition
A RUN profile update (main state control) is completed	SYSC_SYSTSR: RUNDFO	Writing "1" to the SYSC_SYSINTER:RUN DIE0 bit	Writing "1" to the SYSC_SYSICLR:RUND FCLR0 bit
When RUN profile update triggers are enabled, the contents of the RUN profile register contain an error	SYSC_SYSERRIR1: RUNERRIF0	Always enabled because of the NMI level	Writing "1" to the SYSC_SYSERRICLR1: RUNERRICLR0 bit
During a PSS change (WFI instruction execution), the contents of the RUN profile register contain an error	SYSC_SYSERRIR1: RUNWKERRIF0	Always enabled because of the NMI level	Writing "1" to the SYSC_SYSERRICLR1: RUNWKERRICLR0 bit
During a PSS change (WFI instruction execution), the contents of the PSS profile register contain an error	SYSC_SYSERRIR1: PSSERRIF0	Always enabled because of the NMI level	Writing "1" to the SYSC_SYSERRICLR1: PSSERRICLR0 bit
An invalid value is written to the PSS update enable register (main state control)	SYSC_SYSERRIR1: PSSERRIF0	Always enabled because of the NMI level	Writing "1" to the SYSC_SYSERRICLR1: PSSERRICLR0 bit
An invalid value is written to the PSS update enable register (sub-state control)	SYSC_SYSERRIR1: PSSERRIF1	Always enabled because of the NMI level	Writing "1" to the SYSC_SYSERRICLR1: PSSERRICLR1 bit
During a PSS change (main state control), the change is canceled	SYSC_SYSERRIR1: PSSTRGCIF0	Always enabled because of the NMI level	Writing "1" to the SYSC_SYSERRICLR1: PSSTRGCICLR0 bit
During a PSS change (sub-state control), the change is canceled	SYSC_SYSERRIR1: PSSTRGCIF1	Always enabled because of the NMI level	Writing "1" to the SYSC_SYSERRICLR1: PSSTRGCICLR1 bit
An invalid value is written to the RUN update enable register	SYSC_SYSERRIR1: RUNTRGERRIF	Always enabled because of the NMI level	Writing "1" to the SYSC_SYSERRICLR1: RUNTRGERRICLR bit
During a RUN profile update, a RUN profile update is attempted again	SYSC_SYSERRIR1: TRGERRIF	Always enabled because of the NMI level	Writing "1" to the SYSC_SYSERRICLR1: TRGERRICLR bit



### 3.6. Bus Error Response

This section explains the bus error response.

A bus error response is made if a register access is made under any of the following conditions.

1. An invalid value is written to the protection key setting register.
2. A write access is made to a protected register without releasing the protection. (Each SCT register is protected only while waiting for oscillation stabilization.)
3. A write access is made to the protection key setting register while the protection is being released. For an effective write from another CPU, however, the protection is released as that by another CPU, and no bus error will occur.
4. A write access is made to a protected register from other than the master that has released the protection key.
5. A write access is made to the RUN profile register group during a RUN profile update. The write data will be disabled.
6. A write access is made to the RUN/PSS profile register group after PSSSEN is enabled. The write data will be disabled.

## 4. Operation Procedure

This section explains the operation procedure for low-power consumption.

### (1) RUN Profile Update

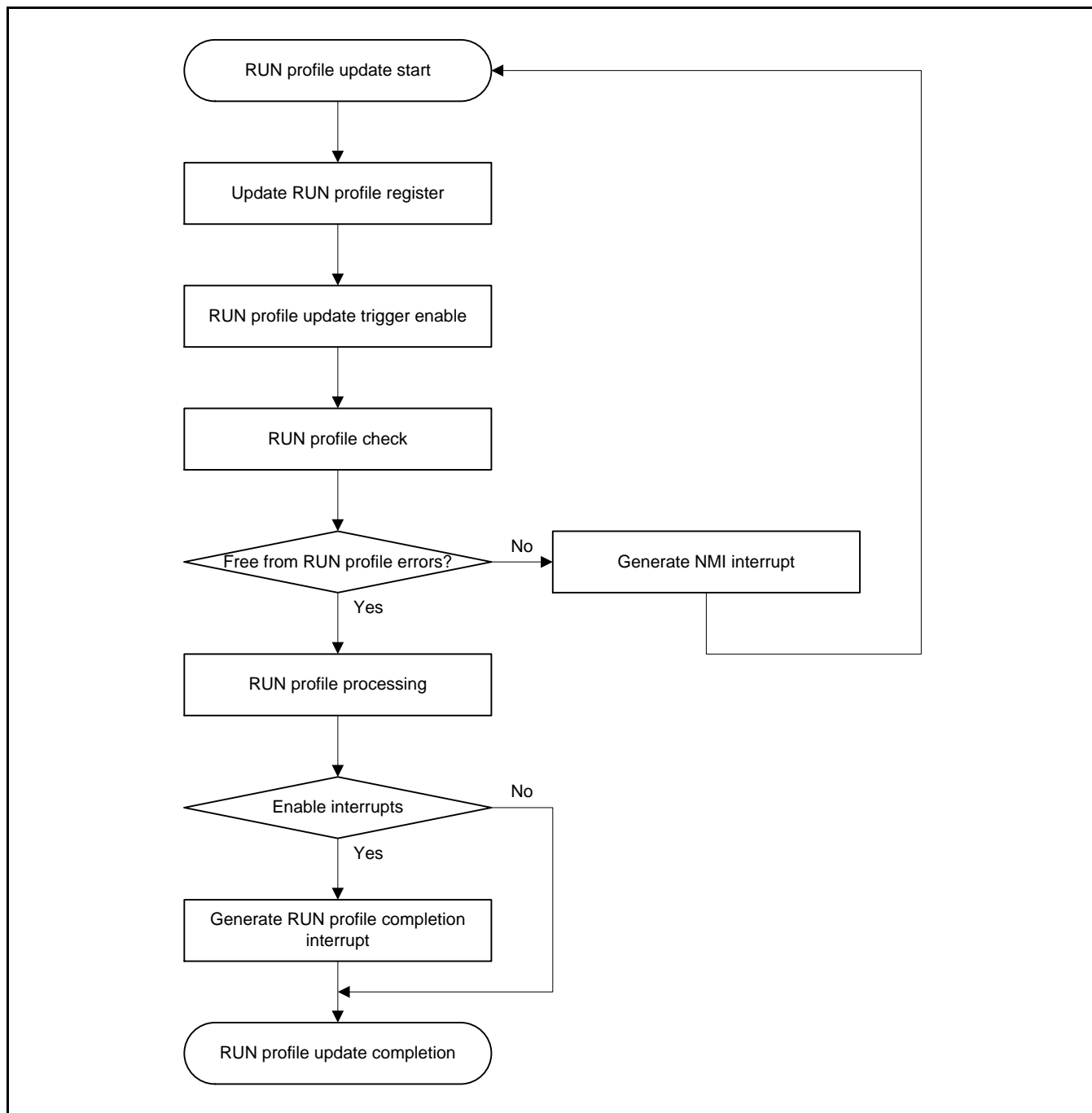
The following explains an example procedure for updating a RUN profile.

1. The user prepares a new RUN profile (sets registers).
2. By writing 0xAB to the RUN profile update trigger register (SYSC\_TRGRUNCNTR:APPLY\_RUN), start a profile update.
3. The control circuit verifies the contents of the new RUN profile. If any settings present problems, a profile error occurs, an error interrupt (NMI) is generated and, at the same time, system error interrupt factor register 1 (SYSC\_SYERRIR1: RUNERRIF0) is set. The contents of the new profile are discarded, and the circuit operates with the contents of the profile currently in use.
4. If the contents of the new RUN profile do not present any problems, the control circuit will reflect the contents of the profile, as follows.
  1. "1" is set in the system status register (SYSC\_SYSSTSR: RUNSTS0).
  2. The contents of the RUN profile are copied to the APPLIED profile.
  3. Clock oscillation enable/stop (including waiting for oscillation stabilization), CSV setting changes, LVD setting changes, clock operation settings (source clock changes, division, and ON/OFF of each clock source), and clock stop settings (source clock stop) are reflected.
  4. When RUN profile update is completed, the system status register (SYSC\_SYSSTSR: RUNSTS0) is cleared to "0", and "1" is set in the system status register (SYSC\_SYSSTSR: RUNDFO). In addition, if the system status interrupt enable register (SYSC\_SYSINTER: RUNDIE0) is enabled, an interrupt is generated.

#### Notes:

- If, while a new profile is being updated, an attempt is made to update the profile again, a system error interrupt (SYSC\_SYERRIR1:RUNTRGERRIF) will be generated, and the profile that is about to be updated again will be disabled.
- Before updating the RUN profile, verify that the flag of the profile status register (SYSC\_SYSPROSTSR: RUNPSTS) does not have a profile error. If an attempt is made to update the RUN profile when a profile error occurs, an NMI interrupt will be generated, and the RUN profile settings will be discarded.
- The following RUN profile settings cannot be changed.
  - CPU0 to 1, HPM clock, SYSC clock

Figure 4-1 RUN Profile Update Operation Flow Chart



## (2) Change from RUN to PSS

The following explains an example procedure for changing each CPU to PSS.

### a) If Main State Control is to Change

1. The user prepares a new PSS profile and a new RUN profile (sets registers). For the RUN profile, make a setting for returning from PSS.
2. Verify whether CPU1 is in a state in which it can change to PSS. Read the value of the following register, and verify each CPU.
  - CPU1 state (SYSC\_SYSSTSR: CPUSTS1)
  - -> CPU1 must be in the WFI state.

- If CPU1 is updating the RUN profile, wait for the update to be completed.
  - If CPU1 is in PSS (WFI instruction executed), steps 3 and 4 need not be performed.
3. Using a resource function (such as inter-core communication), notify CPU1 to change to PSS.
  4. Upon receiving the notification, CPU1 executes the WFI instruction.
  5. CPU0 writes 0xBA to the PSS profile update enable register (SYSC\_PSSSEN0).
  6. Verify the state of CPU1 again. The check items are the same as those in step 2.
  7. Have CPU0 execute the WFI instruction. With this instruction as a trigger, start a profile update.
  8. Verify the contents of the PSS/RUN profiles to update. If the contents of the RUN/PSS profiles present problems, system error interrupt factor register 1 (SYSC\_SYSERRIR1:RUNWKERRIF0/PSSERRIF0) will be enabled. If either profile contains an error, an NMI interrupt will be generated. The interrupt cannot be disabled. Verify the profile error in the flag register, and then correct the location in question.
  9. If the contents of the PSS profile present no problems, the control circuit will reflect the contents of the profile, as below.
    1. The PSS profile setting state of each CPU (SYSC\_SYSS TSR: PSSSTS0-1) is set to "1".
    2. The contents of the PSS profile are copied to the APPLIED profile.
    3. A WAKEUP request is generated, or the state of CPU1 is verified. If a WAKEUP request has been generated or the state of CPU1 has been WFI-released, the change to PSS will be canceled. In this case, an NMI interrupt will be generated, and system error interrupt factor register 1 (SYSC\_SYSERRIR1:PSSTRGCIF0) will be set to "1".
    4. Clock oscillation enable/stop (including waiting for oscillation stabilization), CSV setting changes, LVD setting changes, clock operation settings (source clock changes, division, and ON/OFF of each clock source), and clock stop settings (source clock stop) are performed.
    5. The PSS profile setting state of each CPU (SYSC\_SYSS TSR: PSSSTS0-1) is set to "0", and "1" is set in the PSS profile completion flag of each CPU (SYSC\_SYSS TSR:PSSDF0-1).
  10. The change to PSS is completed.

**Notes:**

- During 1CPU0/1 mode operation, steps 2, 3, 4, and 6 need not be performed. They are necessary for 2CPU mode only.
- When main state control changes to PSS, all CPUs will change to PSS. The state of any CPUs that are already in PSS will not be changed.
- If the WFI instruction is executed without enabling the PSS profile update enable register, the change to PSS will not be made, and the program will stop. If returning to program execution, use an interrupt, a reset, and so on to restore it.
- To perform a PLL start in PSS, make sure that the PLL settings (division or multiplier) in the RUN/PSS profiles are the same. If the settings differ, a profile error reset will be issued at the time of returning to PSS.
- If a profile update is performed with the settings for enabling CSV in PSS and stopping it in RUN, make sure that the settings satisfy the CSV stop conditions. If any invalid settings are made, a profile error reset will be issued at the time of returning to PSS.

The following shows a RUN->PSS (main state control) operation flow chart.



Figure 4-2 2CPU Mode RUN->PSS (Main State Control) Operation Flow Chart

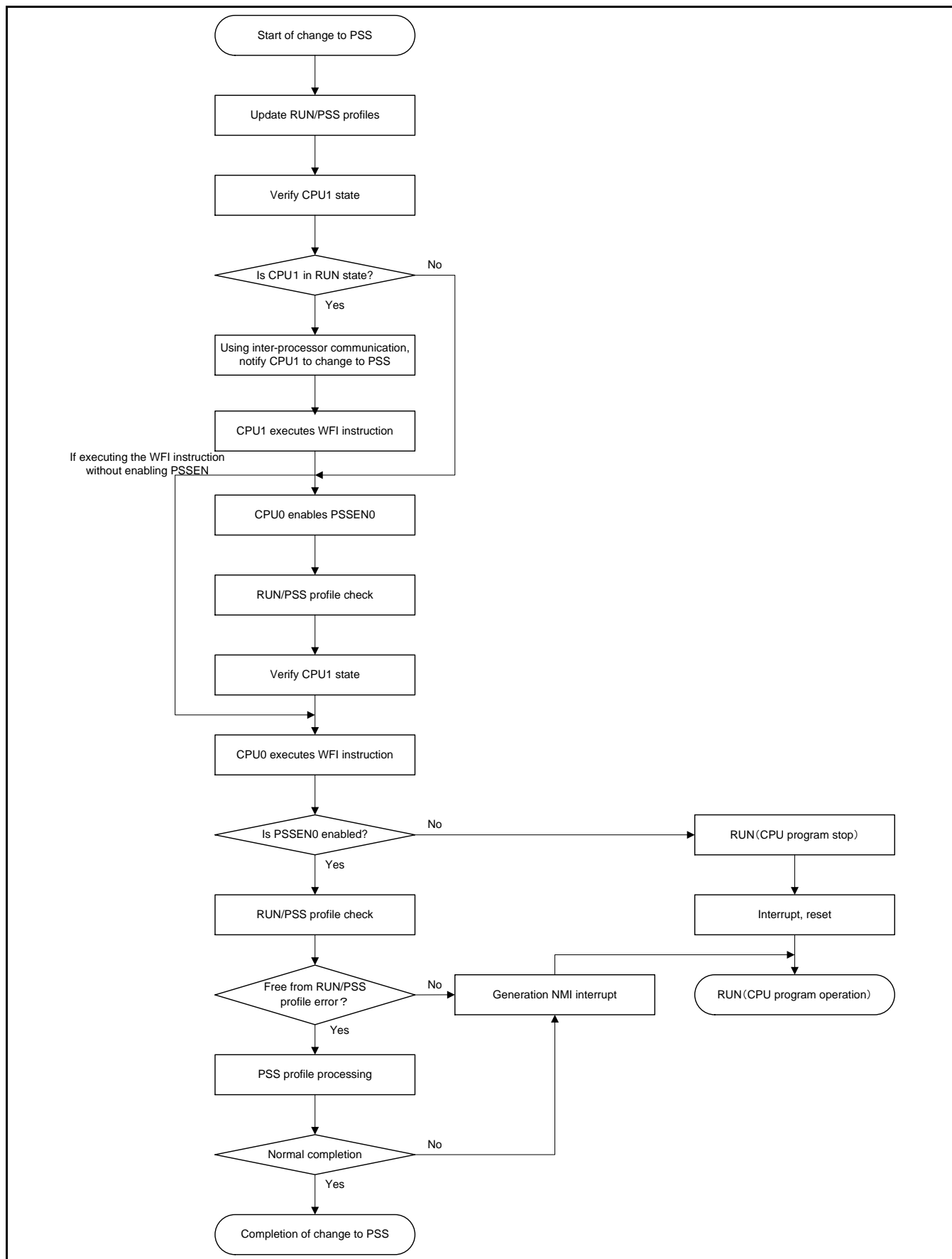
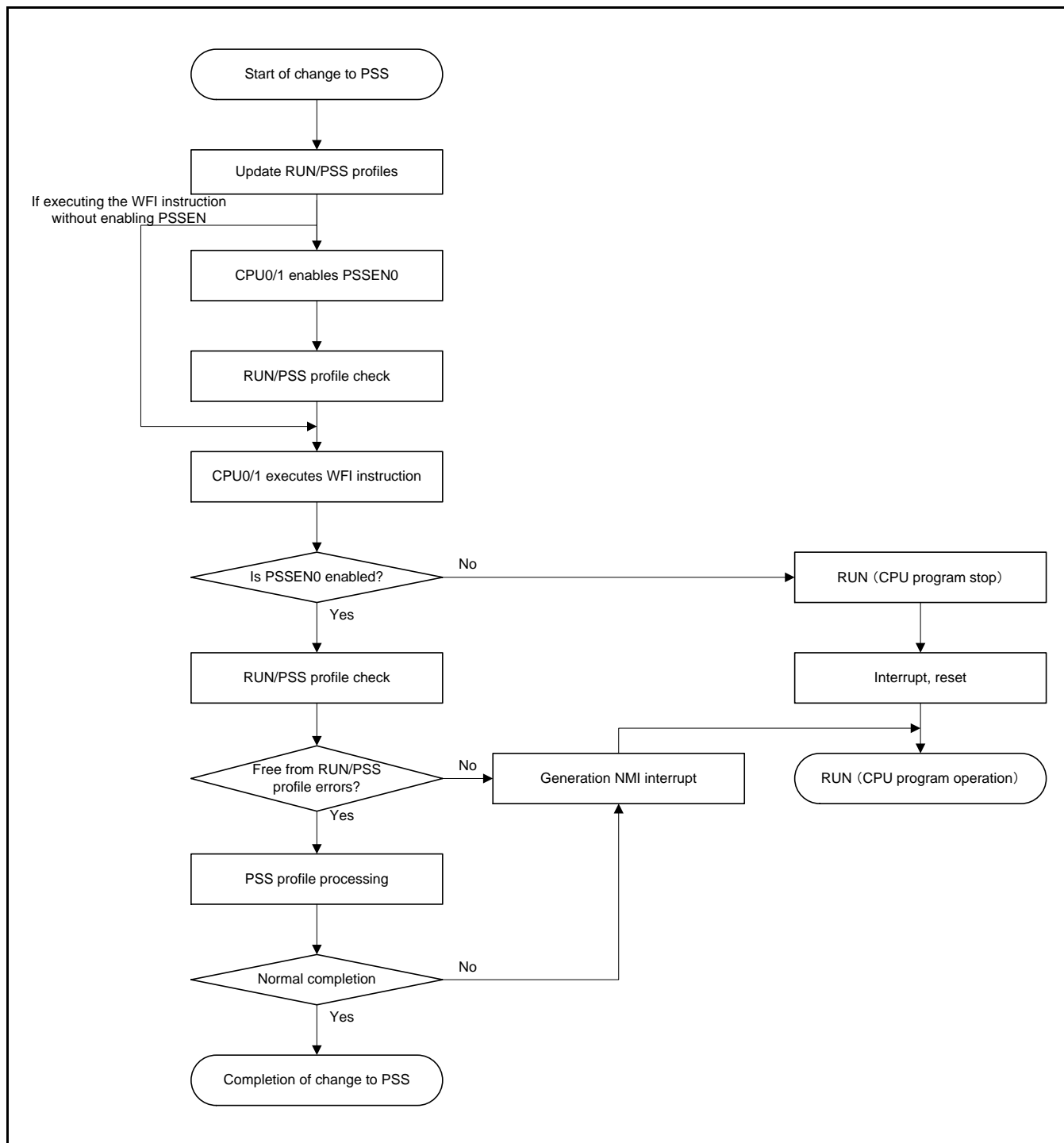


Figure 4-3 Flow Chart of RUN->PSS (Main State Control) Operation during 1CPU0/1 Mode Operation





**b) If Sub-state Control is to Change**

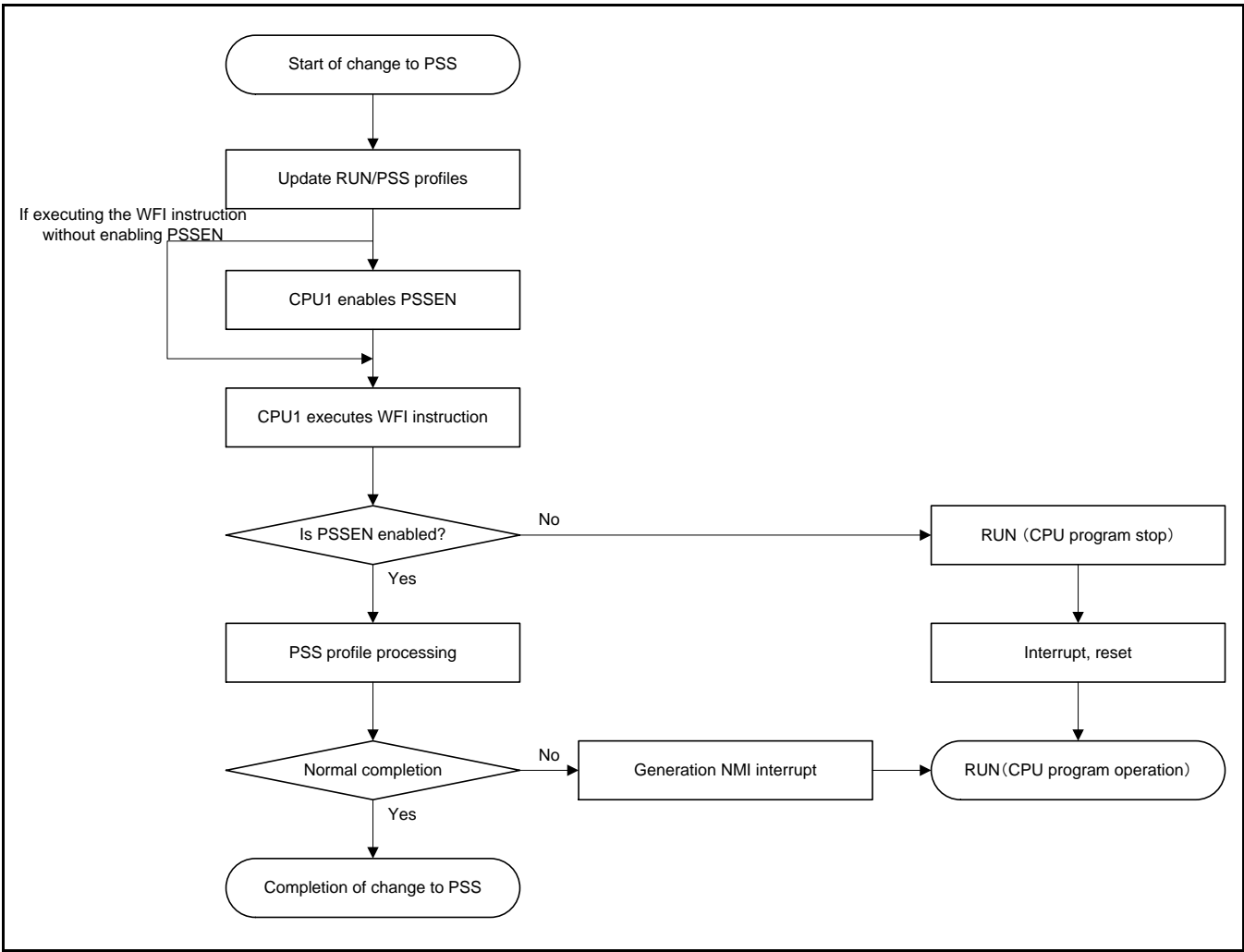
1. The user prepares a new PSS profile and a new RUN profile (sets registers). For the RUN profile, make a setting for returning from PSS.
2. CPU1 writes 0xBA to the PSS profile update enable register (SYSC\_PSSSEN:PSSEN1).
3. Have CPU1 execute the WFI instruction. With this instruction as a trigger, start a profile update.
4. The control circuit reflects the contents of the profiles as below.
  1. The PSS profile setting state of CPU1 (SYSC\_SYSSTSR: PSSSTS1) is set to "1".
  2. The contents of the PSS profile are copied to the APPLIED profile.
  3. Verify whether a WAKEUP1 request has been generated. If a WAKEUP1 request has been generated, the change to PSS will be canceled. In this case, an NMI interrupt will be generated, and system error interrupt factor register 1 (SYSC\_SYSEERRIR1:PSSTRGCIF1) will be set to "1".
  4. Clock operation settings (CPU clock source ON/OFF) are made.
  5. The new profiles are reflected.
  6. The PSS profile setting state of CPU1 (SYSC\_SYSSTSR: PSSSTS1) is set to "0", and "1" is set in the PSS profile completion flag of CPU1 (SYSC\_SYSSTSR:PSSDF1).
5. The change to PSS is completed.

**Note:**

- *If the WFI instruction is executed without enabling the PSS trigger register, the change to PSS will not be made, and the program will stop. If returning to program execution, use an interrupt, a reset, and so on to restore it.*

The following shows a RUN->PSS change (CPU1) operation flow chart.

Figure 4-4 RUN->PSS (CPU1) Operation Flow Chart

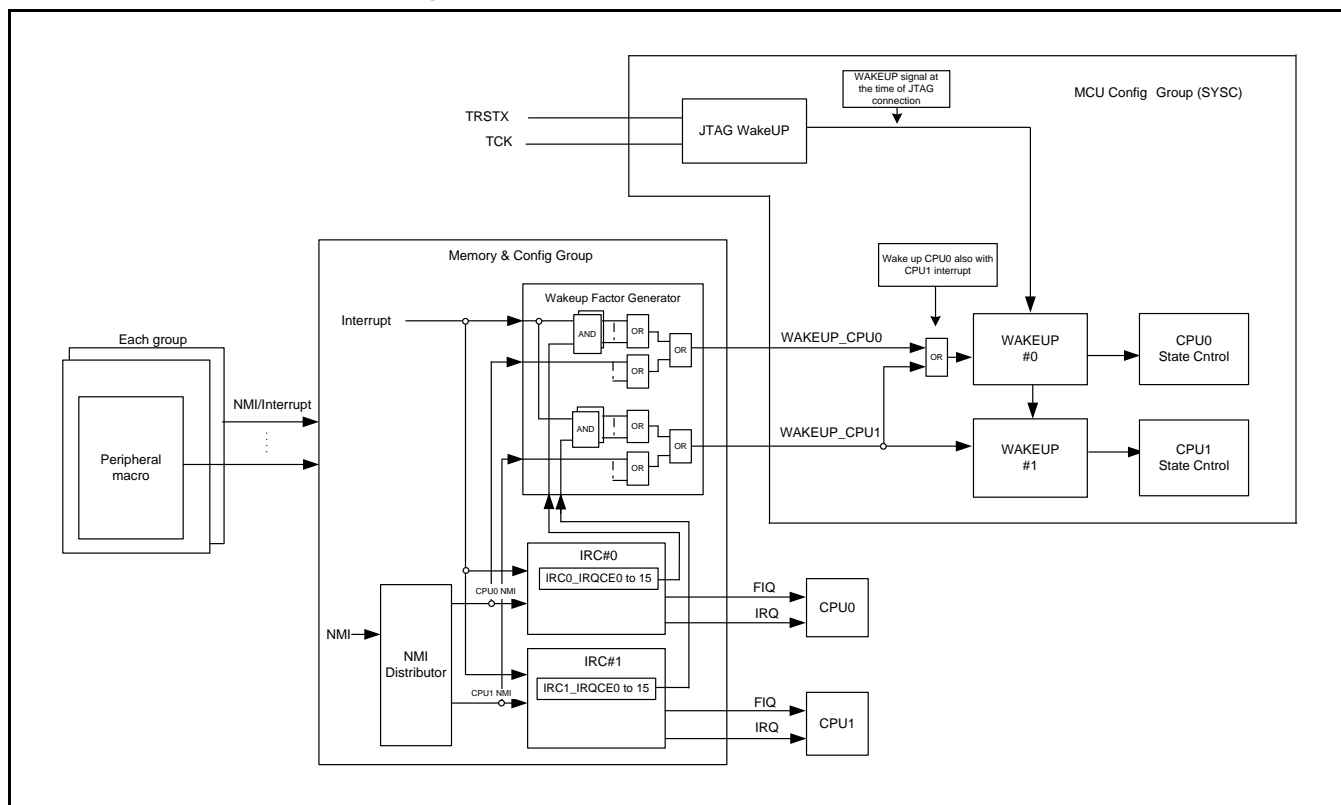


### (3) Change from PSS to RUN

To return to RUN, generate an interrupt and so on.

The following shows a schematic view of the Wakeup circuit.

Figure 4-5 Schematic View of the Wakeup Circuit



All interrupt factors can be set as Wakeup factors.

The following lists the return factors.

- Interrupt from each resource
- NMI
- Reset
- JTAG Wakeup

**Note:**

- The CPU to return can be selected with an appropriate register. The CPU controlling the main state control will always return from PSS.

**(4) If CPUs under Main State Control and Sub-state Control are to Return**

The following describes an example procedure for returning all CPUs from PSS to RUN.

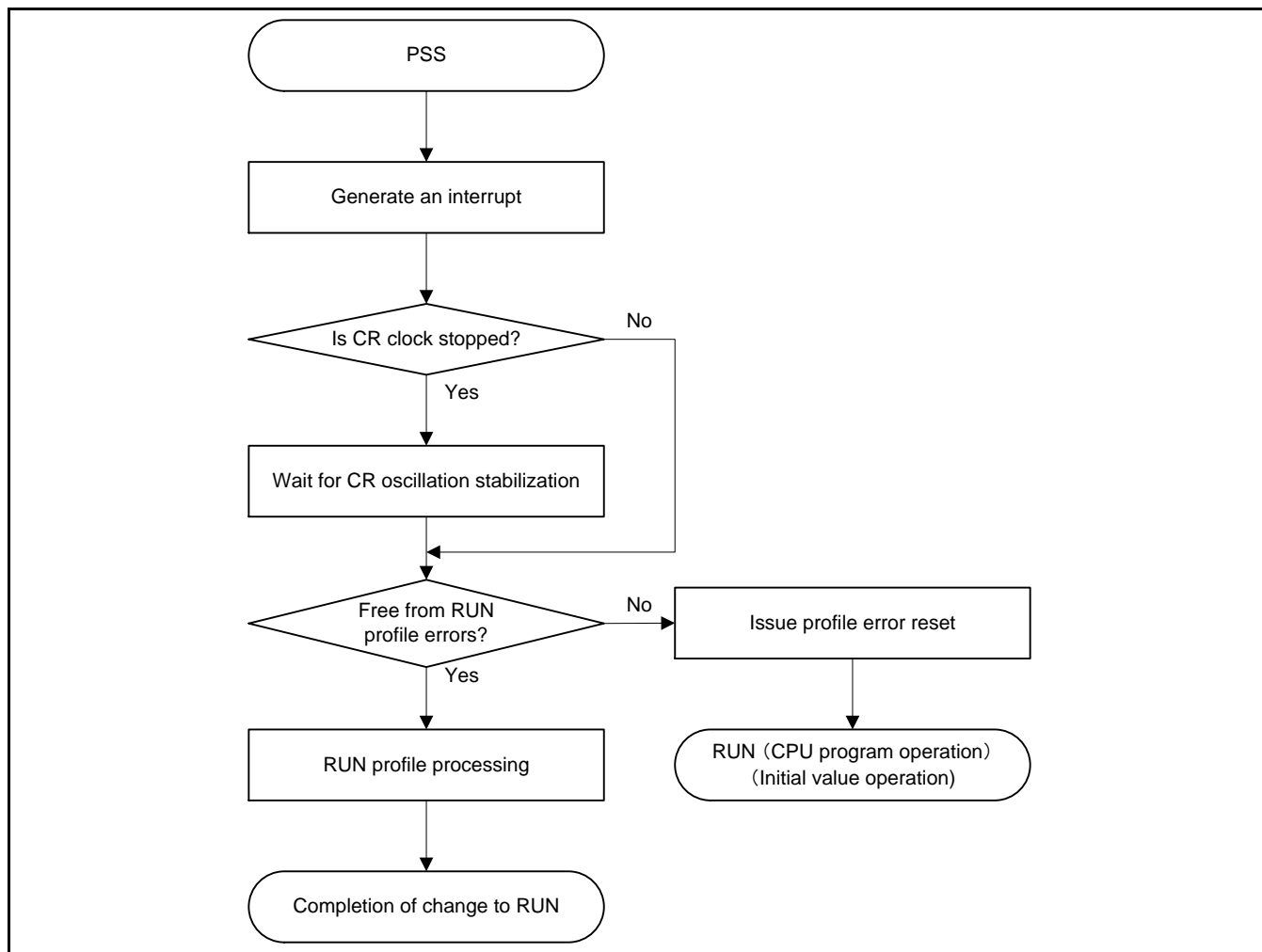
1. A return factor is generated, the control circuit recognizes the return factor first, and then it starts returning.
2. If the fast-CR clock has been stopped, wait for the oscillation of the fast-CR clock to stabilize.
3. If the contents of the RUN profile present problems (such as bit errors and PLL and CSV setting changes), a profile error reset will be generated.
4. If the RUN profile presents no problems, the control circuit will reflect the contents of the profile as below.
  1. The RUN profile setting state of each CPU (SYSC\_SYSSTSR: RUNSTS0-1) is set to "1".
  2. The contents of the RUN profile are copied to the APPLIED profile.
  3. Clock oscillation enable/stop (including waiting for oscillation stabilization), CSV setting changes, LVD setting changes, clock operation settings (source clock changes, division, and ON/OFF of each clock source), and clock stop settings (source clock stop) are performed. The RUN profile setting state of each CPU (SYSC\_SYSSTSR:RUNSTS0-1) is set to "0".
5. The change to RUN is made, and the CPUs start program operation.

**Note:**

- If a profile error occurs, a RUN profile of initial values will be set.

The following shows a PSS->RUN change operation flow chart.

Figure 4-6 PSS->RUN (CPU0-1) Operation Flow Chart



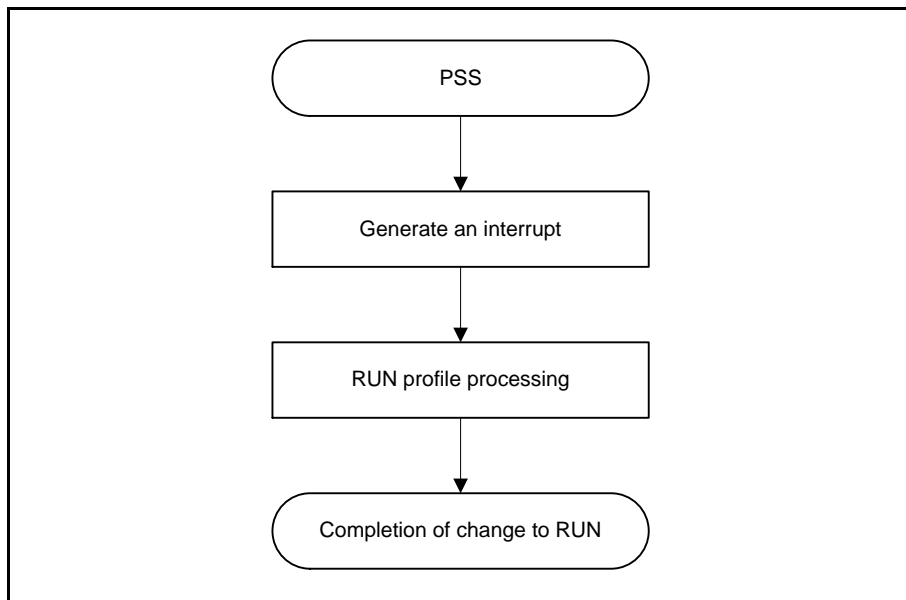
**(5) If only the CPU under Sub-state Control is to Return**

The following describes an example of the procedure for returning CPU1 from PSS to the RUN profile.

1. A return factor is generated, the control circuit recognizes the return factor first, and then it starts returning.
2. The control circuit reflects the contents of the profiles, as below.
  1. The system status register (SYSC\_SYSSTSR: RUNSTS0-1) is enabled.
  2. The contents of the RUN profile are copied to the APPLIED profile.
  3. Clock operation settings (CPU1 clock source ON) are made.
  4. The RUN profile setting state of each CPU (SYSC\_SYSSTSR: RUNSTS1) is set to "0".
3. The change to RUN is made, and the CPU starts program operation.

The following shows a PSS->RUN change operation flow chart.

**Figure 4-7 PSS->RUN (CPU1) Operation Flow Chart**







## 5. Registers

This section explains the low-power consumption registers.

The system controller registers consist of the following register groups.

- Protection register group
- RUN profile register group
- PSS profile register group
- APP profile register group
- Status profile register group
- System register group
- Clock supervisor register group
- Reset control register group
- Source clock timer register group for slow-CR
- Source clock timer register group for fast-CR
- Source clock timer register group for main oscillation
- Clock control register group
- Special setting register group
- Debug register group
  
- For details on the clock supervisor register group in the above items, see "CHAPTER: Clock Supervisor".
- For details on the reset control register group in the above items, see "CHAPTER: Reset".
- For details on the respective source clock timer register groups in the above items, see "CHAPTER: Source Clock Timer".
- For details on the clock control register group in the above items, see "CHAPTER: Clock System".

### Notes:

- *Protection is applied to the SYSC register. To write to the register, the protection key must first be released. Invalid written data is discarded.*
- *The following shows an example of the procedure for accessing the SYSC register.*
  1. *The accessing master releases the protection key.*
  2. *The master that released the protection key writes to the SYSC register.*
  3. *The hardware enables the protection key.*
  4. *Repeat steps 1 to 3 until completing the required settings.*
- *Protection is not applied to the debug register group.*

**Table 5-1 Protection Register Group**

Abbreviated Register Name	Register Name	See
SYSC_PROTKEYR	Protection Key Setting Register	5.1.1

**Table 5-2 RUN Profile Register Group**

Abbreviated Register Name	Register Name	See
SYSC_RUNCKSRER	RUN Clock Source Enable Register	5.2.1
SYSC_RUNCKSELR	RUN Clock Selection Register	5.2.2
SYSC_RUNCKER	RUN Clock Enable Register	5.2.3
SYSC_RUNCKDIVR0	RUN Clock Divider Register	5.2.4
SYSC_RUNCKDIVR1	RUN Clock Divider Register	5.2.5
SYSC_RUNCKDIVR2	RUN Clock Divider Register	5.2.6
SYSC_RUNPLLCNTR	RUN PLL Control Register	5.2.7
SYSC_RUNLVDCFGR	RUN Low-voltage Detection Configuration Register	5.2.8
SYSC_RUNCSVCFGR	RUN Clock Supervisor Configuration Register	5.2.9
SYSC_TRGRUNCNTR	RUN Profile Update Trigger Register	5.2.10

**Table 5-3 PSS Profile Register Group**

Abbreviated Register Name	Register Name	See
SYSC_PSSCKSRER	PSS Clock Source Enable Register	5.3.1
SYSC_PSSCKSELR	PSS Clock Selection Register	5.3.2
SYSC_PSSCKER	PSS Clock Enable Register	5.3.3
SYSC_PSSCKDIVR0	PSS Clock Divider Register 0	5.3.4
SYSC_PSSCKDIVR1	PSS Clock Divider Register 1	5.3.5
SYSC_PSSCKDIVR2	PSS Clock Divider Register 2	5.3.6
SYSC_PSSPLLCNTR	PSS PLL Control Register	5.3.7
SYSC_PSSLVDCFGR	PSS Low-voltage Detection Configuration Register	5.3.8
SYSC_PSSCSVCFGR	PSS clock Supervisor Configuration Register	5.3.9
SYSC_PSSSEN	PSS Profile Update Enable Register	5.3.10



**Table 5-4 APP Profile Register Group**

Abbreviated Register Name	Register Name	See
SYSC_APPCKSRER	APPLIED Clock Source Enable Register	5.4.1
SYSC_APPCKSELR	APPLIED Clock Selection Register	5.4.2
SYSC_APPCKER	APPLIED Clock Enable register	5.4.3
SYSC_APPCKDIVR0	APPLIED Clock Divider Register 0	5.4.4
SYSC_APPCKDIVR1	APPLIED Clock Divider Register 1	5.4.5
SYSC_APPCKDIVR2	APPLIED Clock Divider Register 2	5.4.6
SYSC_APPLLCNTR	APPLIED PLL Control Register	5.4.7
SYSC_APPLVDCFGR	APPLIED Low-voltage Detection Configuration Register	5.4.8
SYSC_APPCSVCFGR	APPLIED Clock Supervisor Configuration Register	5.4.9

**Table 5-5 STS Profile Register Group**

Abbreviated Register Name	Register Name	See
SYSC_STCKSRER	Status Clock Source Enable Register	5.5.1
SYSC_STCKSELR	Status Clock Selection Register	5.5.2
SYSC_STCKER	Status Clock Enable Register	5.5.3
SYSC_STCKDIVR0	Status Clock Divider Register 0	5.5.4
SYSC_STCKDIVR1	Status Clock Divider Register 1	5.5.5
SYSC_STCKDIVR2	Status Clock Divider Register 2	5.5.6
SYSC_STPLLCNTR	Status PLL Control Register	5.5.7
SYSC_STSLVDCFGR	Status Low-voltage Detection Configuration Register	5.5.8
SYSC_STSCSVCFGR	Status Clock Supervisor Configuration Register	5.5.9

**Table 5-6 System Register Group**

Abbreviated Register Name	Register Name	See
SYSC_SYSSTS	System Status Register	5.6.1
SYSC_SYSINTER	System Status Interrupt Enable Register	5.6.2
SYSC_SYSICLR	System Status Flag Interrupt Clear Register	5.6.3
SYSC_SYSERRIR0	System Error interrupt Factor Register 0	5.6.4
SYSC_SYSERRIR1	System Error interrupt Factor Register 1	5.6.5
SYSC_SYSERRICLR0	System Error interrupt Factor Clear Register 0	5.6.6
SYSC_SYSERRICLR1	System Error Interrupt Factor Clear Register 1	5.6.7
SYSC_SYSPROTSR	Profile Status Register	5.6.8
SYSC_SYSRUNPEFR	RUN Profile Error Flag Register	5.6.9
SYSC_SYSPSSPEFR	PSS Profile Error Flag Register	5.6.10

**Table 5-7 Special Setting Register Group**

<b>Abbreviated Register Name</b>	<b>Register Name</b>	<b>See</b>
SYSC_SPECFGR	System Special Configuration Register	5.7.1
SYSC_SPECCPUCFG	CPU Control Register	5.7.2

**Table 5-8 Debug Register Group**

<b>Abbreviated Register Name</b>	<b>Register Name</b>	<b>See</b>
SYSC_JTAGDETECT	JTAG Detection Register	5.8.1
SYSC_JTAGCNFG	JTAG Configuration Register	5.8.2
SYSC_JTAGWAKEUP	JTAG Wake-up Register	5.8.3



## 5.1. Protection Register Group

The register in this group is a control register for accessing SYSC registers.

### 5.1.1. Protection Key Setting Register (SYSC\_PROTKEYR)

This register has settings for releasing the protection of System Controller (SYSC) registers.

BIT_OFFSET	31-0
BIT_NAME	PROTKEY
ACCESS_TYPE	R,W
PROT_TYPE	WP
INITIAL_VALUE	00000000_00000000_00000000_00000000

#### [bit31:0] PROTKEY[31:0]: Protection release setting bits

These bits compose the register for releasing protection keys.

##### a) Write

Value	Description
0x5CAC_CE55	Lock release
Other than above	Invalid

##### b) Read

Value	Description
0x0000_0000	Locked state
0xFFFF_FFFF	Unlocked state

#### Notes:

- After the register in MCU config area is written irrespective of the master, the protection key is locked again.
- This register permits word access only. Any other access is invalid.
- These bits can also be subject to write access by CPU1.

## 5.2. RUN Profile Register Group

The registers in this group are RUN profile control setting registers.



### 5.2.1. RUN Clock Source Enable Register (SYSC\_RUNCKSRER)

RUN clock source enable register (SYSC\_RUNCKSRER) sets whether to enable or disable oscillation of the source clock in the RUN.

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	WPS
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved			PLLEN	Reserved	MOSCEN	SCROSCEN	CROSCEN
ACCESS_TYPE	R0,WX			R/W	R0,WX	R/W	R1,WX	R1,WX
PROT_TYPE	WPS							
INITIAL_VALUE	000			0	0	1	1	1

[bit31:5] Reserved: Reserved bits

#### [bit4] PLLEN: PLL clock oscillation enable bit

This bit controls oscillation of the PLL clock.

Value	Description
0	Disable clock oscillation
1	Enable clock oscillation

[bit3] Reserved: Reserved bit

#### [bit2] MOSCEN: Main clock oscillation enable bit

This bit controls oscillation of the main clock.

Value	Description
0	Disable clock oscillation
1	Enable clock oscillation

#### [bit1] SCROSCEN: Slow-CR clock oscillation enable bit

This bit controls oscillation of the slow-CR clock.

Value	Description
0	-
1	Enable clock oscillation

#### [bit0] CROSCEN: Fast-CR clock oscillation enable bit

This bit controls oscillation of the fast-CR clock.

Value	Description
0	-
1	Enable clock oscillation

### 5.2.2. RUN Clock Selection Register 0 (SYSC\_RUNCKSELR0)

RUN clock selection register 0 (SYSC\_RUNCKSELR0) sets the clock source for the clock domain in the RUN.

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	WPS
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved					CD0CSL		
ACCESS_TYPE	R0,WX					R/W		
PROT_TYPE	WPS							
INITIAL_VALUE	00000					000		

[bit31:3] Reserved: Reserved bits

#### [bit2:0] CD0CSL[2:0]: Clock domain 0 clock selection bits

These bits select the source clock for the clock domain 0.

Value	Description
000	Fast-CR clock
001	Slow-CR clock
010	Main clock / main clock divided by 2
011	Clock fixed at "L"
100	PLL clock
101	Clock fixed at "L"
110	Clock fixed at "L"
111	Clock fixed at "L"

**Note:**

- When switching to PLL clock, use clock gear. Switch with 50MHz or less and activate gear-up operation. For the details, see "CHAPTER: Clock System".





### 5.2.3. RUN Clock Enable Register (SYSC\_RUNCKER)

RUN clock enable register (SYSC\_RUNCKER) sets whether to disable or enable an internal clock in the RUN.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	ENCLKTRC	Reserved						
ACCESS_TYPE	R,WX	R0,WX						
PROT_TYPE	WPS							
INITIAL_VALUE	1	0000000						

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved		ENCLK PERI7	ENCLK PERI6	ENCLK PERI5	ENCLK PERI4	ENCLK PERI1	ENCLK PERI0
ACCESS_TYPE	R0,WX		R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	WPS							
INITIAL_VALUE	00		1	1	1	1	1	1

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved	ENCLK SYSCPD1	Reserved		ENCLK EXTBUS	Reserved	ENCLK MEMC	ENCLK DMA
ACCESS_TYPE	R0,WX	R1,WX	R0,WX		R/W	R0,WX	R/W	R/W
PROT_TYPE	WPS							
INITIAL_VALUE	0	1	00		1	0	1	1

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved	ENCLK HPMPD2	ENCLKATB	ENCLK DBG	Reserved		ENCLK CPU1	ENCLK CPU0
ACCESS_TYPE	R0,WX	R1,WX	R/W	R/W	R0,WX		R1,WX	R1,WX
PROT_TYPE	WPS							
INITIAL_VALUE	0	1	1	1	00		1	1

#### [bit31] ENCLKTRC: TRC internal clock oscillation enable bit

This bit sets whether to disable or enable TRC internal clock oscillation.

Value	Description
0	Disable clock oscillation
1	Enable clock oscillation

#### Notes:

- The read value is set the value in bit[5] ENCLKATB.
- Writing data in this bit does not affect operation.

#### [bit30:22] Reserved: Reserved bits

**[bit21] ENCLKPERI7: PERI7 internal operation clock oscillation enable bit**

This bit sets whether to disable or enable PERI7 internal clock oscillation.

Value	Description
0	Disable clock oscillation
1	Enable clock oscillation

**[bit20] ENCLKPERI6: PERI6 internal operation clock oscillation enable bit**

This bit sets whether to disable or enable PERI6 internal clock oscillation.

Value	Description
0	Disable clock oscillation
1	Enable clock oscillation

**[bit19] ENCLKPERI5: PERI5 internal operation clock oscillation enable bit**

This bit sets whether to disable or enable PERI5 internal clock oscillation.

Value	Description
0	Disable clock oscillation
1	Enable clock oscillation

**[bit18] ENCLKPERI4: PERI4 internal operation clock oscillation enable bit**

This bit sets whether to disable or enable PERI4 internal clock oscillation.

Value	Description
0	Disable clock oscillation
1	Enable clock oscillation

**[bit17] ENCLKPERI1: PERI1 internal operation clock oscillation enable bit**

This bit sets whether to disable or enable PERI1 internal clock oscillation.

Value	Description
0	Disable clock oscillation
1	Enable clock oscillation

**[bit16] ENCLKPERI0: PERI0 internal operation clock oscillation enable bit**

This bit sets whether to disable or enable PERI0 internal clock oscillation.

Value	Description
0	Disable clock oscillation
1	Enable clock oscillation

**[bit15] Reserved: Reserved bit**

**[bit14] ENCLKSYSCPD1: SYSCPD1 internal operation clock oscillation enable bit**

This bit sets whether to disable or enable SYSPD1 internal clock oscillation.

Value	Description
0	-
1	Enable clock oscillation

**[bit13:12] Reserved: Reserved bits**

**[bit11] ENCLKEXTBUS: EXTBUS internal operation clock oscillation enable bit**

This bit sets whether to disable or enable EXTBUS internal clock oscillation.

Value	Description
0	Disable clock oscillation
1	Enable clock oscillation

**Note:**

- Set this bit to "0" due to MB9D560 series has no external bus I/F.

**[bit10] Reserved: Reserved bit**

**[bit9] ENCLKMEMC: MEMC internal operation clock oscillation enable bit**

This bit sets whether to disable or enable MEMC internal clock oscillation.

Value	Description
0	Disable clock oscillation
1	Enable clock oscillation

**[bit8] ENCLKDMA: DMA internal operation clock oscillation enable bit**

This bit sets whether to disable or enable DMA internal clock oscillation.

Value	Description
0	Disable clock oscillation
1	Enable clock oscillation

**[bit7] Reserved: Reserved bit**

**[bit6] ENCLKHPMPD2: HPMPD2 internal operation clock oscillation enable bit**

This bit sets whether to disable or enable HPMPD2 internal clock oscillation.

Value	Description
0	-
1	Enable clock oscillation.

**[bit5] ENCLKATB: ATB internal operation clock oscillation enable bit**

This bit sets whether to disable or enable ATB internal clock oscillation.

Value	Description
0	Disable clock oscillation
1	Enable clock oscillation

**[bit4] ENCLKDBG: DBG internal operation clock oscillation enable bit**

This bit sets whether to disable or enable DBG internal clock oscillation.

Value	Description
0	Disable clock oscillation
1	Enable clock oscillation

**[bit3:2] Reserved: Reserved bits**

**[bit1] ENCLKCPU1: CPU1 internal operation clock oscillation enable bit**

This bit sets whether to disable or enable CPU1 internal clock oscillation.

Value	Description
0	-
1	Enable clock oscillation

**[bit0] ENCLKCPU0: CPU0 internal operation clock oscillation enable bit**

This bit sets whether to disable or enable CPU0 internal clock oscillation.

Value	Description
0	-
1	Enable clock oscillation



## 5.2.4. RUN Clock Divider Register 0 (SYSC\_RUNCKDIVR0)

RUN clock divider register 0 (SYSC\_RUNCKDIVR0) sets the division ratio of the clock in the RUN.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved		HPMDIV		Reserved		TRCDIV	
ACCESS_TYPE	R0,WX		R/W		R0,WX		R/W	
PROT_TYPE	WPS							
INITIAL_VALUE	00		00		00		00	

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved		ATBDIV		Reserved		DBGDIV	
ACCESS_TYPE	R0,WX		R,WX		R0,WX		R,WX	
PROT_TYPE	WPS							
INITIAL_VALUE	00		00		00		01	

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved			SYSDIV				
ACCESS_TYPE	R0,WX			R/W				
PROT_TYPE	WPS							
INITIAL_VALUE	000			00000				

**[bit31:30] Reserved: Reserved bits**

**[bit29:28] HPMDIV[1:0]: HPM clock divider setting bits**

These bits set the division ratio of the HPM clock, DMA clock and MEMC clock from the system clock.

Value	Description
00	No division
01	Divided by 2
10	Divided by 4
11	Divided by 8

**[bit27:26] Reserved: Reserved bits**

**[bit25:24] TRCDIV[1:0]: TRC clock divider setting bits**

These bits set the division ratio of the TRC clock from the system clock.

Value	Description
00	No division
01	Divided by 2
10	Divided by 4
11	Divided by 8

**[bit23:22] Reserved: Reserved bits**

**[bit21:20] ATBDIV[1:0]: ATB clock divider setting bits**

These bits set the division ratio of the ATB clock from the system clock.

Value	Description
00	No division
01	Divided by 2
10	Divided by 4
11	Divided by 8

**Notes:**

- The read values are set the values in bit[25:24] TRCDIV[1:0].
- Writing data in this bit does not affect operation.

**[bit19:18] Reserved: Reserved bits**

**[bit17:16] DBGDIV[1:0]: DBG clock divider setting bits**

These bits set the division ratio of the DBG clock from the ATB clock.

Value	Description
00	Reserved
01	Divided by 2
10	Reserved
11	Reserved

**Notes:**

- The read values are always "0b01".
- Writing this bit has no effect.

**[bit15:5] Reserved: Reserved bits**



**[bit4:0] SYSDIV[4:0]: SYS clock divider setting bits**

These bits set the division ratio of the system clock from the source clock.

Value	Description
0_0000	No division
0_0001	Divided by 2
0_0010	Divided by 3
...	...
11101	Divided by 30
11110	Divided by 31
11111	Divided by 32

### 5.2.5. RUN Clock Divider Register 1 (SYSC\_RUNCKDIVR1)

RUN clock divider register 1 (SYSC\_RUNCKDIVR1) sets the division ratio of the clock in the RUN.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved				SYSCPD1DIV			
ACCESS_TYPE	R0,WX				R/W			
PROT_TYPE	WPS							
INITIAL_VALUE	0000				0000			

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved	EXTBUSDIV			Reserved			
ACCESS_TYPE	R0,WX	R/W			R0,WX			
PROT_TYPE	WPS							
INITIAL_VALUE	0	000			0000			

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							

[bit31:28] Reserved: Reserved bits

#### [bit27:24] SYSCPD1DIV[3:0]: SYSC\_PD1 clock divider setting bits

These bits set the division ratio of the SYSC\_PD1 clock from the HPM clock.

Value	Description
0000	No division
0001	Divided by 2
0010	Divided by 3
...	
1101	Divided by 14
1110	Divided by 15
1111	Divided by 16

[bit23:15] Reserved: Reserved bits





**[bit14:12] EXTBUSDIV[2:0]: EXTBUS clock divider setting bits**

These bits set the division ratio of the EXTBUS clock from the HPM clock.

Value	Description
000	No division
001	Divided by 2
010	Divided by 4
011	Divided by 8
100	Divided by 16
101	Divided by 32
110	Divided by 64
111	Divided by 128

**[bit11:0] Reserved: Reserved bits**

### 5.2.6. RUN Clock Divider Register 2 (SYSC\_RUNCKDIVR2)

RUN clock divider register 2 (SYSC\_RUNCKDIVR2) sets the division ratio of the clock in the RUN.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	PERI7DIV				PERI6DIV			
ACCESS_TYPE	R/W				R/W			
PROT_TYPE	WPS							
INITIAL_VALUE	0000				0000			

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	PERI5DIV				PERI4DIV			
ACCESS_TYPE	R/W				R/W			
PROT_TYPE	WPS							
INITIAL_VALUE	0000				0000			

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	PERI1DIV				PERIODIV			
ACCESS_TYPE	R/W				R/W			
PROT_TYPE	WPS							
INITIAL_VALUE	0000				0000			

**[bit31:24] Reserved: Reserved bits**

#### **[bit23:20] PERI7DIV[3:0]: PERI7 clock divider setting bits**

These bits set the division ratio of the PERI7 clock from the PERI5 clock.

Value	Description
0000	No division
0001	Divided by 2
0010	Divided by 3
...	...
1101	Divided by 14
1110	Divided by 15
1111	Divided by 16



**[bit19:16] PERI6DIV[3:0]: PERI6 clock divider setting bits**

These bits set the division ratio of the PERI6 clock from the PERI4 clock.

Value	Description
0000	No division
0001	Divided by 2
0010	Divided by 3
...	...
1101	Divided by 14
1110	Divided by 15
1111	Divided by 16

**[bit15:12] PERI5DIV[3:0]: PERI5 clock divider setting bits**

These bits set the division ratio of the PERI5 clock from the HPM clock.

Value	Description
0000	No division
0001	Divided by 2
0010	Divided by 3
...	...
1101	Divided by 14
1110	Divided by 15
1111	Divided by 16

**[bit11:8] PERI4DIV[3:0]: PERI4 clock divider setting bits**

These bits set the division ratio of the PERI4 clock from the HPM clock.

Value	Description
0000	No division
0001	Divided by 2
0010	Divided by 3
...	...
1101	Divided by 14
1110	Divided by 15
1111	Divided by 16

**[bit7:4] PERI1DIV[3:0]: PERI1 clock divider setting bits**

These bits set the division ratio of the PERI1 clock from the PERI0 clock.

Value	Description
0000	No division
0001	Divided by 2
0010	Divided by 3
...	...
1101	Divided by 14
1110	Divided by 15
1111	Divided by 16

**[bit3:0] PERIODIV[3:0]: PERIO clock divider setting bits**

These bits set the division ratio of the PERIO clock from the HPM clock.

Value	Description
0000	No division
0001	Divided by 2
0010	Divided by 3
...	...
1101	Divided by 14
1110	Divided by 15
1111	Divided by 16



### 5.2.7. RUN PLL Control Register (SYSC\_RUNPLLCNTR)

RUN PLL control register (SYSC\_RUNPLLCNTR) controls PLL in the RUN.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved	PLLDIVN						
ACCESS_TYPE	R0,WX	R/W						
PROT_TYPE	WPS							
INITIAL_VALUE	0	0001100						

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved				PLLDIVM			
ACCESS_TYPE	R0,WX				R/W			
PROT_TYPE	WPS							
INITIAL_VALUE	0000				0001			

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved						PLLDIVL	
ACCESS_TYPE	R0,WX						R/W	
PROT_TYPE	WPS							
INITIAL_VALUE	000000						00	

[bit31:23] Reserved: Reserved bits

[bit22:16] PLLDIVN[6:0]: PLL clock N-multiplier setting bits

These bits set the multiplication rate of the PLL clock.

Value	Description
000_0000	Setting prohibited
...	Setting prohibited
000_1011	Setting prohibited
000_1100	Multiply by 13
000_1101	Multiply by 14
000_1110	Multiply by 15
...	
110_0001	Multiply by 98
110_0010	Multiply by 99
110_0011	Multiply by 100
110_0100	Setting prohibited
...	Setting prohibited
111_1111	Setting prohibited

**[bit15:12] Reserved: Reserved bits**

**[bit11:8] PLLDIVM[3:0]: PLL clock M-divider setting bits**

These bits set the division ratio for PLL clock output.

Value	Description
0000	Reserved (divided by 2)
0001	Divided by 2
0010	Divided by 4
0011	Divided by 6
0010	Divided by 8
0101	Divided by 10
0110	Divided by 12
0111	Divided by 14
1000	Divided by 16
1001	Divided by 18
1010	Divided by 20
1011	Divided by 22
1010	Divided by 24
1101	Divided by 26
1110	Divided by 28
1111	Divided by 30

**[bit7:2] Reserved: Reserved bits**

**[bit1:0] PLLDIVL[1:0]: PLL input clock divider setting bits**

These bits set the division ratio of the PLL input clock.

Value	Description
00	No division
01	Divided by 2
10	Divided by 4
11	Divided by 6



### 5.2.8. RUN Low-voltage Detection Configuration Register (SYSC\_RUNLVDCFGR)

RUN low-voltage detection configuration register (SYSC\_RUNLVDCFGR) sets operation of low-voltage detection in the RUN.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved				SV12			LVDE12
ACCESS_TYPE	R0,WX				R1,WX	R0,WX	R0,WX	R1,WX
PROT_TYPE	WPS							
INITIAL_VALUE	0000				100			1

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved	LVDS50	Reserved		SV50			LVDE50
ACCESS_TYPE	R0,WX	R/W	R0,WX		R/W			R/W
PROT_TYPE	WPS							
INITIAL_VALUE	0	0	00		110			1

[bit31:20] Reserved: Reserved bits

[bit19:17] SV12[2:0]: 1.2V reference voltage setting bits

These bits indicate detection voltage of low-voltage detection interrupt.

Value	Description
100	0.9V

[bit16] LVDE12: 1.2V low-voltage detection operation enable bit

This bit sets the low-voltage detection operation.

Value	Description
0	-
1	Enable operation.

**Note:**

- Low-voltage detection is unavailable until the low-voltage detection stability time elapses.

**[bit15:7] Reserved: Reserved bits**

**[bit6] LVDS50: low-voltage detection (5.0V) operation selection bit**

This bit sets the low-voltage detection operation.

Value	Description
0	Reset is generated
1	Interrupt is generated

**[bit5:4] Reserved: Reserved bits**

**[bit3:1] SV50[2:0]: 5.0V reference voltage setting**

These bits set the detection voltage.

Value	Description
000	Setting is prohibited
001	Setting is prohibited
010	Setting is prohibited
011	Setting is prohibited
100	Setting is prohibited
101	3.9 V
110	4.1 V
111	4.3 V

**Note:**

- When these bits are changed, low-voltage detection is unavailable until the low-voltage detection stability time elapses.

**[bit0] LVDE50: 5.0V low-voltage detection operation enable bit**

This bit sets the low-voltage detection operation.

Value	Description
0	Disable operation
1	Enable operation.

**Note:**

- After low-voltage detection is enabled, low-voltage detection is unavailable until the low-voltage detection stability time elapses.





### 5.2.9. RUN Clock Supervisor Configuration Register (SYSC\_RUNCSVCFGR)

RUN clock supervisor configuration register (SYSC\_RUNCSVCFGR) sets whether to enable or disable operation of each clock supervisor in the RUN.

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	WPS
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved					PLLCSVE	Reserved	MOCSVE
ACCESS_TYPE	R0,WX					R/W	R0,WX	R/W
PROT_TYPE	WPS							
INITIAL_VALUE	00000					0	0	0

**[bit31:3] Reserved: Reserved bits**

#### **[bit2] PLLCSVE: PLL clock supervisor enable bit**

This bit sets whether to enable or disable the PLL clock supervisor.

Value	Description
0	Disable the clock supervisor
1	Enable the clock supervisor

**[bit1] Reserved: Reserved bit**

#### **[bit0] MOCSVE: Main clock supervisor enable bit**

This bit sets whether to enable or disable the main clock supervisor.

Value	Description
0	Disable the clock supervisor
1	Enable the clock supervisor

### 5.2.10. RUN Profile Update Trigger Register (SYSC\_TRGRUNCNTR)

RUN profile update trigger register (SYSC\_TRGRUNCNTR) is a RUN profile update trigger.

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	WPS
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	APPLY_RUN							
ACCESS_TYPE	R0,W							
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							

[bit31:8] Reserved: Reserved bits

#### [bit7:0] APPLY\_RUN[7:0]: RUN profile update trigger setting bits

These bits compose the RUN profile update start register.

Value	Description
0xAB	Start RUN profile update
Other than above	invalid

#### Notes:

- If the value written in this register is invalid, an error flag (SYSC\_SYSEERRIR1:RUNTRGERRIF is "1") is set.
- If the RUN profile is updated, it cannot be re-updated until that the update process completes. If "0xAB" is written in this register during a RUN profile update, an error flag (SYSC\_SYSEERRIR1:TRGERRIF is "1") is set.
- The read values of this register are "0x0000\_0000".
- Whether main state control or sub state control switched automatically depending on CPU which writes this register and CPU operation mode. For details, see "(7) Relationship between CPU Operation Modes and State Controls" of "3.1. Low-power Consumption States".
- These bits can be accessed by CPU0 and CPU1.



### 5.3. PSS Profile Register Group

The registers in this group are PSS profile control setting registers.

### 5.3.1. PSS Clock Source Enable Register (SYSC\_PSSCKSRER)

PSS clock source enable register (SYSC\_PSSCKSRER) sets whether to enable or disable oscillation of the source clock in the PSS.

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	WPS
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved			PLLEN	Reserved	MOSCEN	SCROSCEN	CROSCEN
ACCESS_TYPE	R0,WX			R/W	R0,WX	R/W	R/W	R/W
PROT_TYPE	WPS							
INITIAL_VALUE	000			0	0	1	1	1

[bit31:5] Reserved: Reserved bits

#### [bit4] PLLEN: PLL clock oscillation enable bit

This bit controls oscillation of the PLL clock.

Value	Description
0	Disable clock oscillation
1	Enable clock oscillation

[bit3] Reserved: Reserved bit

#### [bit2] MOSCEN: Main clock oscillation enable bit

This bit controls oscillation of the main clock.

Value	Description
0	Disable clock oscillation
1	Enable clock oscillation

#### [bit1] SCROSCEN: Slow-CR clock oscillation enable bit

This bit controls oscillation of the slow-CR clock.

Value	Description
0	Disable clock oscillation
1	Enable clock oscillation

#### [bit0] CROSCEN: Fast-CR clock oscillation enable bit

This bit controls oscillation of the fast-CR clock.

Value	Description
0	Disable clock oscillation
1	Enable clock oscillation



### 5.3.2. PSS Clock Selection Register 0 (SYSC\_PSSCKSELR0)

PSS clock selection register 0 (SYSC\_PSSCKSELR0) sets the clock source for the clock domain in the PSS.

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	WPS
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved					CD0CSL		
ACCESS_TYPE	R0,WX					R/W		
PROT_TYPE	WPS							
INITIAL_VALUE	00000					000		

[bit31:3] Reserved: Reserved bits

#### [bit2:0] CD0CSL[2:0]: Clock domain 0 clock selection bits

These bits select the source clock for the clock domain 0.

Value	Description
000	Fast-CR clock
001	Slow-CR clock
010	Main clock / Main clock divided by 2
011	Clock fixed at "L"
100	PLL clock
101	Clock fixed at "L"
110	Clock fixed at "L"
111	Clock fixed at "L"

#### Notes:

- When switching to PLL clock, use clock gear. Switch with 50MHz or less and activate gear-up operation. For the details, see "CHAPTER: Clock System".
- When clock domain 0 clock is disabled, stop clock by PSS clock selection register 0 (SYSC\_PSSCKSELR0) and disabled all internal operation clock oscillation by PSS clock enable register (SYSC\_PSSCKER).

### 5.3.3. PSS Clock Enable Register (SYSC\_PSSCKER)

PSS clock enable register (SYSC\_PSSCKER) sets whether to disabled or enable an internal clock in the PSS.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	ENCLKTRC	Reserved						
ACCESS_TYPE	R,WX	R0,WX						
PROT_TYPE	WPS							
INITIAL_VALUE	1	0000000						

BITS_OFFSET	23	22	21	20	19	18	17	16
BITS_NAME	Reserved		ENCLK PERI7	ENCLK PERI6	ENCLK PERI5	ENCLK PERI4	ENCLK PERI1	ENCLK PERI0
ACCESS_TYPE	R0,WX		R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	WPS							
INITIAL_VALUE	00		1	1	1	1	1	1

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved	ENCLK SYSCPD1	Reserved		ENCLK EXTBUS	Reserved	ENCLK MEMC	ENCLK DMA
ACCESS_TYPE	R0,WX	R/W	R0,WX		R/W	R0,WX	R/W	R/W
PROT_TYPE	WPS							
INITIAL_VALUE	0	1	00		1	0	1	1

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved	ENCLK HPMPD2	ENCLK ATB	ENCLK DBG	Reserved		ENCLK CPU1	ENCLK CPU0
ACCESS_TYPE	R0,WX	R/W	R/W	R/W	R0,WX		R0,WX	R0,WX
PROT_TYPE	WPS							
INITIAL_VALUE	0	1	1	1	00		0	0

#### [bit31] ENCLKTRC: TRC internal clock oscillation enable bit

This bit sets whether to disable or enable PERI6 internal clock oscillation.

Value	Description
0	Disable clock oscillation
1	Enable clock oscillation

#### Notes:

- The read value is set the value in bit[5] ENCLKATB.
- Writing data in this bit does not affect operation.

#### [bit30:22] Reserved: Reserved bits



**[bit21] ENCLKPERI7: PERI7 internal operation clock oscillation enable bit**

This bit sets whether to disable or enable PERI7 internal clock oscillation.

Value	Description
0	Disable clock oscillation
1	Enable clock oscillation

**[bit20] ENCLKPERI6: PERI6 internal operation clock oscillation enable bit**

This bit sets whether to disable or enable PERI6 internal clock oscillation.

Value	Description
0	Disable clock oscillation
1	Enable clock oscillation

**[bit19] ENCLKPERI5: PERI5 internal operation clock oscillation enable bit**

This bit sets whether to disable or enable PERI5 internal clock oscillation.

Value	Description
0	Disable clock oscillation
1	Enable clock oscillation

**[bit18] ENCLKPERI4: PERI4 internal operation clock oscillation enable bit**

This bit sets whether to disable or enable PERI4 internal clock oscillation.

Value	Description
0	Disable clock oscillation
1	Enable clock oscillation

**[bit17] ENCLKPERI1: PERI1 internal operation clock oscillation enable bit**

This bit sets whether to disable or enable PERI1 internal clock oscillation.

Value	Description
0	Disable clock oscillation
1	Enable clock oscillation

**[bit16] ENCLKPERI0: PERI0 internal operation clock oscillation enable bit**

This bit sets whether to disable or enable PERI0 internal clock oscillation.

Value	Description
0	Disable clock oscillation
1	Enable clock oscillation

**[bit15] Reserved: Reserved bit**

**[bit14] ENCLKSYSCPD1: SYSCPD1 internal operation clock oscillation enable bit**

This bit sets whether to disable or enable SYSCPD1 internal clock oscillation.

Value	Description
0	Disable clock oscillation
1	Enable clock oscillation

**[bit13:12] Reserved: Reserved bits**

**[bit11] ENCLKEXTBUS: EXTBUS internal operation clock oscillation enable bit**

This bit sets whether to disable or enable EXTBUS internal clock oscillation.

Value	Description
0	Disable clock oscillation
1	Enable clock oscillation

**Note:**

- Set this bit to "0" due to MB9D560 series has no external bus I/F.

**[bit10] Reserved: Reserved bit**

**[bit9] ENCLKMEMC: MEMC internal operation clock oscillation enable bit**

This bit sets whether to disable or enable MEMC internal clock oscillation.

Value	Description
0	Disable clock oscillation
1	Enable clock oscillation

**[bit8] ENCLKDMA: DMA internal operation clock oscillation enable bit**

This bit sets whether to disable or enable DMA internal clock oscillation.

Value	Description
0	Disable clock oscillation
1	Enable clock oscillation

**[bit7] Reserved: Reserved bit**

**[bit6] ENCLKHPMPD2: HPMPD2 internal operation clock oscillation enable bit**

This bit sets whether to disable or enable HPMPD2 internal clock oscillation.

Value	Description
0	Disable clock oscillation
1	Enable clock oscillation

**[bit5] ENCLKATB: ATB internal operation clock oscillation enable bit**

This bit sets whether to disable or enable ATB internal clock oscillation.

Value	Description
0	Disable clock oscillation
1	Enable clock oscillation





**[bit4] ENCLKDBG: DBG internal operation clock oscillation enable bit**

This bit sets whether to disable or enable DBG internal clock oscillation.

Value	Description
0	Disable clock oscillation
1	Enable clock oscillation

**[bit3:2] Reserved: Reserved bits**

**[bit1] ENCLKCPU1: CPU1 internal operation clock oscillation enable bit**

This bit sets whether to disable or enable CPU1 internal clock oscillation.

Value	Description
0	Disable clock oscillation
1	-

**[bit0] ENCLKCPU0: CPU0 internal operation clock oscillation enable bit**

This bit sets whether to disable or enable CPU0 internal clock oscillation.

Value	Description
0	Disable clock oscillation
1	-

### 5.3.4. PSS Clock Divider Register 0 (SYSC\_PSSCKDIVR0)

PSS clock divider register 0 (SYSC\_PSSCKDIVR0) sets the clock division ratio in the PSS.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved		HPMDIV		Reserved		TRCDIV	
ACCESS_TYPE	R0,WX		R/W		R0,WX		R/W	
PROT_TYPE	WPS							
INITIAL_VALUE	00		00		00		00	

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved		ATBDIV		Reserved		DBGDIV	
ACCESS_TYPE	R0,WX		R,WX		R0,WX		R,WX	
PROT_TYPE	WPS							
INITIAL_VALUE	00		00		00		01	

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved			SYSDIV				
ACCESS_TYPE	R0,WX			R/W				
PROT_TYPE	WPS							
INITIAL_VALUE	000			00000				

[bit31:30] Reserved: Reserved bits

[bit29:28] HPMDIV[1:0]: HPM clock divider setting bits

Value	Description
00	No division
01	Divided by 2
10	Divided by 4
11	Divided by 8

[bit27:26] Reserved: Reserved bits



**[bit25:24] TRCDIV[1:0]: TRC clock divider setting bits**

Value	Description
00	No division
01	Divided by 2
10	Divided by 4
11	Divided by 8

**[bit23:22] Reserved: Reserved bits**

**[bit21:20] ATBDIV[1:0]: ATB clock divider setting bits**

Value	Description
00	No division
01	Divided by 2
10	Divided by 4
11	Divided by 8

**Notes:**

- The read values are set the values in bit[25:24] TRCDIV[1:0].
- Writing data in this bit does not affect operation.

**[bit19:18] Reserved: Reserved bits**

**[bit17:16] DBGDIV[1:0]: DBG clock divider setting bits**

Value	Description
00	-
01	Divided by 2
10	-
11	-

**Notes:**

- The read values are always "0b01".
- Writing this bit has no effect.

**[bit15:5] Reserved: Reserved bits**

**[bit4:0] SYSDIV[4:0]: SYS clock divider setting bits**

These bits set the division ratio from the source clock of the system clock.

Value	Description
0_0000	No division
0_0001	Divided by 2
0_0010	Divided by 3
...	
1_1101	Divided by 30
1_1110	Divided by 31
1_1111	Divided by 32

### 5.3.5. PSS Clock Divider Register 1 (SYSC\_PSSCKDIVR1)

PSS clock divider register 1 (SYSC\_PSSCKDIVR1) sets the clock division ratio in the PSS.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved				SYSCPD1DIV			
ACCESS_TYPE	R0,WX				R/W			
PROT_TYPE	WPS							
INITIAL_VALUE	0000				0000			

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved	EXTBUSDIV			Reserved			
ACCESS_TYPE	R0,WX	R/W			R0,WX			
PROT_TYPE	WPS							
INITIAL_VALUE	0	000			0000			

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							

**[bit31:28] Reserved: Reserved bits**

#### **[bit27:24] SYSCPD1DIV[3:0]: SYSC\_PD1 clock divider setting bits**

These bits set the division ratio of the SYSC\_PD1 clock from the HPM clock.

Value	Description
0000	No division
0001	Divided by 2
0010	Divided by 3
...	
1101	Divided by 14
1110	Divided by 15
1111	Divided by 16

**[bit23:15] Reserved: Reserved bits**



**[bit14:12] EXTBUSDIV [2:0]: EXTBUS clock divider setting bits**

These bits set the division ratio of the EXTBUS clock from the HPM clock.

Value	Description
000	No division
001	Divided by 2
010	Divided by 4
011	Divided by 8
100	Divided by 16
101	Divided by 32
110	Divided by 64
111	Divided by 128

**[bit11:0] Reserved: Reserved bits**

### 5.3.6. PSS Clock Divider Register 2 (SYSC\_PSSCKDIVR2)

PSS clock divider register 2 (SYSC\_PSSCKDIVR2) sets the clock division ratio in the PSS.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	PERI7DIV				PERI6DIV			
ACCESS_TYPE	R/W				R/W			
PROT_TYPE	WPS							
INITIAL_VALUE	0000				0000			

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	PERI5DIV				PERI4DIV			
ACCESS_TYPE	R/W				R/W			
PROT_TYPE	WPS							
INITIAL_VALUE	0000				0000			

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	PERI1DIV				PERIODIV			
ACCESS_TYPE	R/W				R/W			
PROT_TYPE	WPS							
INITIAL_VALUE	0000				0000			

**[bit31:24] Reserved: Reserved bits**

#### **[bit23:20] PERI7DIV[3:0]: PERI7 clock divider setting bits**

These bits set the division ratio of the PERI7 clock from the PERI5 clock.

Value	Description
0000	No division
0001	Divided by 2
0010	Divided by 3
...	
1101	Divided by 14
1110	Divided by 15
1111	Divided by 16



**[bit19:16] PERI6DIV[3:0]: PERI6 clock divider setting bits**

These bits set the division ratio of the PERI6 clock from the PERI4 clock.

Value	Description
0000	No division
0001	Divided by 2
0010	Divided by 3
...	
1101	Divided by 14
1110	Divided by 15
1111	Divided by 16

**[bit15:12] PERI5DIV[3:0]: PERI5 clock divider setting bits**

These bits set the division ratio of the PERI5 clock from the HPM clock.

Value	Description
0000	No division
0001	Divided by 2
0010	Divided by 3
...	
1101	Divided by 14
1110	Divided by 15
1111	Divided by 16

**[bit11:8] PERI4DIV[3:0]: PERI4 clock divider setting bits**

These bits set the division ratio of the PERI4 clock from the HPM clock.

Value	Description
0000	No division
0001	Divided by 2
0010	Divided by 3
...	
1101	Divided by 14
1110	Divided by 15
1111	Divided by 16

**[bit7:4] PERI1DIV[3:0]: PERI1 clock divider setting bits**

These bits set the division ratio of the PERI1 clock from the PERI0 clock.

Value	Description
0000	No division
0001	Divided by 2
0010	Divided by 3
...	
1101	Divided by 14
1110	Divided by 15
1111	Divided by 16

**[bit3:0] PERIODIV[3:0]: PERIO clock divider setting bits**

These bits set the division ratio of the PERIO clock from the HPM clock.

Value	Description
0000	No division
0001	Divided by 2
0010	Divided by 3
...	
1101	Divided by 14
1110	Divided by 15
1111	Divided by 16





### 5.3.7. PSS PLL Control Register (SYSC\_PSSPLLCNTR)

PSS PLL control register (SYSC\_PSSPLLCNTR) controls in the PSS.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved	PLLDIVN						
ACCESS_TYPE	R0,WX	R/W						
PROT_TYPE	WPS							
INITIAL_VALUE	0	0001100						

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved				PLLDIVM			
ACCESS_TYPE	R0,WX				R/W			
PROT_TYPE	WPS							
INITIAL_VALUE	0000				0001			

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved						PLLDIVL	
ACCESS_TYPE	R0,WX						R/W	
PROT_TYPE	WPS							
INITIAL_VALUE	000000						00	

[bit31:23] Reserved: Reserved bits

[bit22:16] PLLDIVN[6:0]: PLL clock N-multiplier setting bits

These bits set the multiplication rate of the PLL clock.

Value	Description
000_0000	Setting prohibited
...	Setting prohibited
000_1011	Setting prohibited
000_1100	Multiply by 13
000_1101	Multiply by 14
000_1110	Multiply by 15
...	
110_0001	Multiply by 98
110_0010	Multiply by 99
110_0011	Multiply by 100
110_0100	Setting prohibited
...	Setting prohibited
111_1111	Setting prohibited

**[bit15:12] Reserved: Reserved bits**

**[bit11:8] PLLDIVM[3:0]: PLL clock M-divider setting bits**

These bits set the division ratio for PLL clock output.

Value	Description
0000	Reserved (divided by 2)
0001	Divided by 2
0010	Divided by 4
0011	Divided by 6
0010	Divided by 8
0101	Divided by 10
0110	Divided by 12
0111	Divided by 14
1000	Divided by 16
1001	Divided by 18
1010	Divided by 20
1011	Divided by 22
1010	Divided by 24
1101	Divided by 26
1110	Divided by 28
1111	Divided by 30

**[bit7:2] Reserved: Reserved bits**

**[bit1:0] PLLDIVL[1:0]: PLL input clock divider setting bits**

These bits set the division ratio of the PLL input clock.

Value	Description
00	No division
01	Divided by 2
10	Divided by 4
11	Divided by 6



### 5.3.8. PSS Low-voltage Detection Configuration Register (SYSC\_PSSLVDCFGR)

PSS low-voltage detection configuration register (SYSC\_PSSLVDCFGR) sets operation of low-voltage detection in the PSS.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved				SV12			LVDE12
ACCESS_TYPE	R0,WX				R1,WX	R0,WX	R0,WX	R1,WX
PROT_TYPE	WPS							
INITIAL_VALUE	0000				100			1

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved	LVDS50	Reserved		SV50			LVDE50
ACCESS_TYPE	R0,WX	R/W	R0,WX		R/W			R/W
PROT_TYPE	WPS							
INITIAL_VALUE	0	0	00		110			1

**[bit31:20] Reserved: Reserved bits**

**[bit19:17] SV12[2:0]: 1.2V reference voltage setting bits**

These bits indicates detection voltage of low-voltage detection interrupt.

Value	Description
100	0.9V

**[bit16] LVDE12: 1.2V low-voltage detection operation enable bit**

This bit sets the low-voltage detection operation.

Value	Description
0	-
1	Enable operation.

**[bit15:7] Reserved: Reserved bits**

**[bit6] LVDS50: Low-voltage detection (5.0V) operation selection bit**

This bit sets the low-voltage detection operation.

Value	Description
0	Reset is generated
1	Interrupt is generated

**[bit5:4] Reserved: Reserved bits**

**[bit3:1] SV50[2:0]: 5.0V reference voltage setting**

These bits set the detection voltage.

Value	Description
000	Setting is prohibited
001	Setting is prohibited
010	Setting is prohibited
011	Setting is prohibited
100	Setting is prohibited
101	3.9 V
110	4.1 V
111	4.3 V

**Note:**

- When these bits are changed, low-voltage detection is unavailable until the low-voltage detection stability time elapses.

**[bit0] LVDE50: 5.0V low-voltage detection operation enable bit**

This bit sets the low-voltage detection operation.

Value	Description
0	Disable operation
1	Enable operation.

**Note:**

- After low-voltage detection is enabled, low-voltage detection is unavailable until the low-voltage detection stability time elapses.



### 5.3.9. PSS Clock Supervisor Configuration Register (SYSC\_PSSCSVCFGR)

PSS clocks supervisor configuration register (SYSC\_PSSCSVCFGR) sets whether to enable or disable operation of each clock supervisor in the PSS.

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	WPS
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved					PLLCSVE	Reserved	MOCSVE
ACCESS_TYPE	R0,WX					R/W	R0,WX	R/W
PROT_TYPE	WPS							
INITIAL_VALUE	00000					0	0	0

**[bit31:3] Reserved: Reserved bits**

#### **[bit2] PLLCSVE: PLL clock supervisor enable bit**

This bit sets whether to enable or disable the PLL clock supervisor.

Value	Description
0	Disable the clock supervisor
1	Enable the clock supervisor

**[bit1] Reserved: Reserved bit**

#### **[bit0] MOCSVE: Main clock supervisor enable bit**

This bit sets whether to enable or disable the main clock supervisor.

Value	Description
0	Disable the clock supervisor
1	Enable the clock supervisor

### 5.3.10. PSS Profile Update Enable Register (SYSC\_PSSSEN1)

This register enables PSS profile updates.

BIT_OFFSET	31-16
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	WPS
INITIAL_VALUE	00000000_00000000

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	PSSSEN1							
ACCESS_TYPE	R,W							
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	PSSSEN0							
ACCESS_TYPE	R,W							
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							

[bit31:16] Reserved: Reserved bits

#### [bit15:8] PSSSEN1: PSS profile (sub status control 1) update enable setting bits

These bits compose the PSS profile (sub status control 1) update start register.

Value	Description
0xBA	Enable transition to PSS
0x00	Clear transition to PSS
Other than above	Invalid

#### Notes:

- If the value written in this register invalid, an error flag (SYSC\_SYSEERRIR1:PSSSENERRIF1 is "1") is set.
- This bit can be subject to write access by CPU1 only.
- This register permits byte access only. Any other access is invalid.



**[bit7:0] PSSEN0: PSS profile (main status control) update enable setting bits**

These bits compose the PSS profile (main status control) update start register.

Value	Description
0xBA	Enable transition to PSS
0x00	Clear transition to PSS
Other than above	Invalid

**Notes:**

- If the value written in this register invalid, an error flag (SYSC\_SYSEERRIR1:PSSSENERRIF0 is "1") is set.
- This register permits byte access only. Any other access is invalid. If the above is not complied with, a bus error occurs.
- CPU which controls this register is switched depending on CPU operation mode.
- Write access to these bits depends on CPU operation mode. CPU0 can write in 2 CPU mode and 1 CPU 0 mode and CPU1 can write in 1 CPU 1 mode.

## **5.4. APPLIED Profile Register Group**

The registers in this group are APPLIED profile control setting registers.





### 5.4.1. APPLIED Clock Source Enable Register (SYSC\_APPCKSRER)

APPLIED clock source enable register (SYSC\_APPCKSRER) indicates the value for setting whether to enable/disable oscillation of the internal operating clock to be updated.

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	WPS
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved			PLLEN	Reserved	MOSCEN	SCROSCEN	CROSCEN
ACCESS_TYPE	R0,WX			R,WX	R0,WX	R,WX	R,WX	R,WX
PROT_TYPE	WPS							
INITIAL_VALUE	000			0	0	1	1	1

[bit31:5] Reserved: Reserved bits

#### [bit4] PLLEN: PLL clock oscillation enable bit

This bit indicates the set value of the PLL clock.

Value	Description
0	Disable clock oscillation
1	Enable clock oscillation

[bit3] Reserved: Reserved bit

#### [bit2] MOSCEN: Main clock oscillation enable bit

This bit indicates the set value of the main clock.

Value	Description
0	Disable clock oscillation
1	Enable clock oscillation

#### [bit1] SCROSCEN: Slow-CR clock oscillation enable bit

This bit indicates the set value of the slow-CR clock.

Value	Description
0	Disable clock oscillation
1	Enable clock oscillation

#### [bit0] CROSCEN: fast-CR clock oscillation enable bit

This bit indicates the set value of the fast-CR clock.

Value	Description
0	Disable clock oscillation
1	Enable clock oscillation

### 5.4.2. APPLIED Clock Selection Register 0 (SYSC\_APPCKSELR0)

APPLIED clock selection register 0 (SYSC\_APPCKSELR0) indicates the set values of the clock source for the clock domain to be updated.

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	WPS
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved					CD0CSL		
ACCESS_TYPE	R0,WX					R,WX		
PROT_TYPE	WPS							
INITIAL_VALUE	00000					000		

[bit31:3] Reserved: Reserved bits

#### [bit2:0] CD0CSL[2:0]: Clock domain 0 clock selection bits

These bits indicate the set value of clock domain 0 clock.

Value	Description
000	Fast-CR clock
001	Slow-CR clock
010	Main clock / Main clock divided by 2
011	Clock fixed at "L"
100	PLL clock
101	Clock fixed at "L"
110	Clock fixed at "L"
111	Clock fixed at "L"



### 5.4.3. APPLIED Clock Enable Register (SYSC\_APPCKER)

APPLIED clock enable register (SYSC\_APPCKER) indicates the value for setting whether to enable/disable oscillation of the internal operating clock to be updated.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	ENCLKTRC	Reserved						
ACCESS_TYPE	R,WX	R0,WX						
PROT_TYPE	WPS							
INITIAL_VALUE	1	0000000						

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved		ENCLK PERI7	ENCLK PERI6	ENCLK PERI5	ENCLK PERI4	ENCLK PERI1	ENCLK PERI0
ACCESS_TYPE	R0,WX		R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	WPS							
INITIAL_VALUE	00		1	1	1	1	1	1

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved	ENCLK SYSCPD1	Reserved		ENCLK EXTBUS	Reserved	ENCLK MEMC	ENCLK DMA
ACCESS_TYPE	R0,WX	R,WX	R0,WX		R,WX	R0,WX	R,WX	R,WX
PROT_TYPE	WPS							
INITIAL_VALUE	0	1	00		1	0	1	1

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved	ENCLK HPMPD2	ENCLK ATB	ENCLK DBG	Reserved		ENCLK CPU1	ENCLK CPU0
ACCESS_TYPE	R0,WX	R,WX	R,WX	R,WX	R0,WX		R,WX	R,WX
PROT_TYPE	WPS							
INITIAL_VALUE	0	1	1	1	00		1	1

#### [bit31] ENCLKTRC: TRC internal operation clock oscillation enable bit

This bit indicates whether to disable or enable TRC internal clock oscillation.

Value	Description
0	Disable clock oscillation.
1	Enable clock oscillation.

#### Notes:

- The read value is set the value in bit[5] ENCLKATB.
- Writing data in this bit does not affect operation.

#### [bit30:22] Reserved: Reserved bits

**[bit21] ENCLKPERI7: PERI7 internal operation clock oscillation enable bit**

This bit indicates whether to disable or enable PERI7 internal clock oscillation.

Value	Description
0	Disable clock oscillation
1	Enable clock oscillation

**[bit20] ENCLKPERI6: PERI6 internal operation clock oscillation enable bit**

This bit indicates whether to disable or enable PERI6 internal clock oscillation.

Value	Description
0	Disable clock oscillation
1	Enable clock oscillation

**[bit19] ENCLKPERI5: PERI5 internal operation clock oscillation enable bit**

This bit indicates whether to disable or enable PERI5 internal clock oscillation.

Value	Description
0	Disable clock oscillation
1	Enable clock oscillation

**[bit18] ENCLKPERI4: PERI4 internal operation clock oscillation enable bit**

This bit indicates whether to disable or enable PERI4 internal clock oscillation.

Value	Description
0	Disable clock oscillation
1	Enable clock oscillation

**[bit17] ENCLKPERI1: PERI1 internal operation clock oscillation enable bit**

This bit indicates whether to disable or enable PERI1 internal clock oscillation.

Value	Description
0	Disable clock oscillation
1	Enable clock oscillation

**[bit16] ENCLKPERI0: PERI0 internal operation clock oscillation enable bit**

This bit indicates whether to disable or enable PERI0 internal clock oscillation.

Value	Description
0	Disable clock oscillation
1	Enable clock oscillation

**[bit15] Reserved: Reserved bit**

**[bit14] ENCLKSYSCPD1: SYSCPD1 internal operation clock oscillation enable bit**

This bit indicates whether to disable or enable SYSCPD1 internal clock oscillation.

Value	Description
0	Disable clock oscillation
1	Enable clock oscillation

**[bit13:12] Reserved: Reserved bits**

**[bit11] ENCLKEXTBUS: EXTBUS internal operation clock oscillation enable bit**

This bit indicates whether to disable or enable EXTBUS internal clock oscillation.

Value	Description
0	Disable clock oscillation
1	Enable clock oscillation

**[bit10] Reserved: Reserved bit**

**[bit9] ENCLKMEMC: MEMC internal operation clock oscillation enable bit**

This bit indicates whether to disable or enable MEMC internal clock oscillation.

Value	Description
0	Disable clock oscillation
1	Enable clock oscillation

**[bit8] ENCLKDMA: DMA internal operation clock oscillation enable bit**

This bit indicates whether to disable or enable DMA internal clock oscillation.

Value	Description
0	Disable clock oscillation
1	Enable clock oscillation

**[bit7] Reserved: Reserved bit**

**[bit6] ENCLKHPMPD2: HPMPD2 internal operation clock oscillation enable bit**

This bit indicates whether to disable or enable HPMPD2 internal clock oscillation.

Value	Description
0	-
1	Enable clock oscillation.

**[bit5] ENCLKATB: ATB internal operation clock oscillation enable bit**

This bit indicates whether to disable or enable ATB internal clock oscillation.

Value	Description
0	Disable clock oscillation
1	Enable clock oscillation

**[bit4] ENCLKDBG: DBG internal operation clock oscillation enable bit**

This bit indicates whether to disable or enable DBG internal clock oscillation.

Value	Description
0	Disable clock oscillation
1	Enable clock oscillation

**[bit3:2] Reserved: Reserved bits**

**[bit1] ENCLKCPU1: CPU1 internal operation clock oscillation enable bit**

This bit indicates whether to disable or enable CPU1 internal clock oscillation.

Value	Description
0	Disable clock oscillation
1	Enable clock oscillation

**[bit0] ENCLKCPU0: CPU0 internal operation clock oscillation enable bit**

This bit indicates whether to disable or enable CPU0 internal clock oscillation.

Value	Description
0	Disable clock oscillation
1	Enable clock oscillation



#### 5.4.4. APPLIED Clock Divider Register 0 (SYSC\_APPCKDIVR0)

APPLIED clock divider register 0 (SYSC\_APPCKDIVR0) indicates the set values of the division ratio of each clock to be updated.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved		HPMDIV		Reserved		TRCDIV	
ACCESS_TYPE	R0,WX		R,WX		R0,WX		R,WX	
PROT_TYPE	WPS							
INITIAL_VALUE	00		00		00		00	

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved		ATBDIV		Reserved		DBGDIV	
ACCESS_TYPE	R0,WX		R,WX		R0,WX		R,WX	
PROT_TYPE	WPS							
INITIAL_VALUE	00		00		00		01	

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved			SYSDIV				
ACCESS_TYPE	R0,WX			R,WX				
PROT_TYPE	WPS							
INITIAL_VALUE	000			00000				

[bit31:30] Reserved: Reserved bits

[bit29:28] HPMDIV[1:0]: HPM clock divider setting bits

These bits indicate the division ratio of the HPM clock, DMA clock and MEMC clock from the system clock.

Value	Description
00	No division
01	Divided by 2
10	Divided by 4
11	Divided by 8

[bit27:26] Reserved: Reserved bits

**[bit25:24] TRCDIV[1:0]: TRC clock divider setting bits**

These bits indicate the division ratio of the TRC clock from the system clock.

Value	Description
00	No division
01	Divided by 2
10	Divided by 4
11	Divided by 8

**[bit23:22] Reserved: Reserved bits**

**[bit21:20] ATBDIV[1:0]: ATB clock divider setting bits**

These bits indicate the division ratio of the ATB clock from the system clock.

Value	Description
00	No division
01	Divided by 2
10	Divided by 4
11	Divided by 8

**Notes:**

- The read value of these bits are the value set to bit[25:24] TRCDIV[1:0] bits.
- Writing to this bit has no effect on operation,

**[bit19:18] Reserved: Reserved bits**

**[bit17:16] DBGDIV[1:0]: DBG clock divider setting bits**

These bits indicate the division ratio from the source clock of the system clock.

Value	Description
00	Reserved
01	Divided by 2
10	Reserved
11	Reserved

**Notes:**

- The read values are always "0b01".
- Writing this bit has no effect.

**[bit15:5] Reserved: Reserved bits**





**[bit4:0] SYSDIV[4:0]: SYS clock divider setting bits**

These bits indicate the division ratio from the source clock of the system clock.

Value	Description
0_0000	No division
0_0001	Divided by 2
0_0010	Divided by 3
...	
1_1101	Divided by 30
1_1110	Divided by 31
1_1111	Divided by 32

### 5.4.5. APPLIED Clock Divider Register 1 (SYSC\_APPCKDIVR1)

APPLIED clock divider register 1 (SYSC\_APPCKDIVR1) indicates the set values of the division ratio of each clock to be updated.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved				SYSCPD1DIV			
ACCESS_TYPE	R0,WX				R,WX			
PROT_TYPE	WPS							
INITIAL_VALUE	0000				0000			

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved	EXTBUSDIV			Reserved			
ACCESS_TYPE	R0,WX	R,WX			R0,WX			
PROT_TYPE	WPS							
INITIAL_VALUE	0	000			0000			

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							

[bit31:28] Reserved: Reserved bits

#### [bit27:24] SYSCPD1DIV[3:0]: SYSC\_PD1 clock divider setting bits

These bits indicate the division ratio of the SYSC\_PD1 clock from the HPM clock.

Value	Description
0000	No division
0001	Divided by 2
0010	Divided by 3
...	
1101	Divided by 14
1110	Divided by 15
1111	Divided by 16

[bit23:15] Reserved: Reserved bits



**[bit14:12] EXTBUSDIV[2:0]: EXTBUS clock divider setting bits**

These bits indicate the division ratio of the EXTBUS clock from the HPM clock.

Value	Description
000	No division
001	Divided by 2
010	Divided by 4
011	Divided by 8
100	Divided by 16
101	Divided by 32
110	Divided by 64
111	Divided by 128

**[bit11:0] Reserved: Reserved bits**

### 5.4.6. APPLIED Clock Divider Register 2 (SYSC\_APPCKDIVR2)

APPLIED clock divider register 2 (SYSC\_APPCKDIVR2) indicates the set values of the division ratio of each clock to be updated.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	PERI7DIV				PERI6DIV			
ACCESS_TYPE	R,WX				R,WX			
PROT_TYPE	WPS							
INITIAL_VALUE	0000				0000			

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	PERI5DIV				PERI4DIV			
ACCESS_TYPE	R,WX				R,WX			
PROT_TYPE	WPS							
INITIAL_VALUE	0000				0000			

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	PERI1DIV				PERIODIV			
ACCESS_TYPE	R,WX				R,WX			
PROT_TYPE	WPS							
INITIAL_VALUE	0000				0000			

**[bit31:24] Reserved: Reserved bits**

#### **[bit23:20] PERI7DIV[3:0]: PERI7 clock divider setting bits**

These bits indicate the division ratio of the PERI7 clock from the PERI5 clock.

Value	Description
0000	No division
0001	Divided by 2
0010	Divided by 3
...	
1101	Divided by 14
1110	Divided by 15
1111	Divided by 16



**[bit19:16] PERI6DIV[3:0]: PERI6 clock divider setting bits**

These bits indicate the division ratio of the PERI6 clock from the PERI4 clock.

Value	Description
0000	No division
0001	Divided by 2
0010	Divided by 3
...	
1101	Divided by 14
1110	Divided by 15
1111	Divided by 16

**[bit15:12] PERI5DIV[3:0]: PERI5 clock divider setting bits**

These bits indicate the division ratio of the PERI5 clock from the HPM clock.

Value	Description
0000	No division
0001	Divided by 2
0010	Divided by 3
...	
1101	Divided by 14
1110	Divided by 15
1111	Divided by 16

**[bit11:8] PERI4DIV[3:0]: PERI4 clock divider setting bits**

These bits indicate the division ratio of the PERI4 clock from the HPM clock.

Value	Description
0000	No division
0001	Divided by 2
0010	Divided by 3
...	
1101	Divided by 14
1110	Divided by 15
1111	Divided by 16

**[bit7:4] PERI1DIV[3:0]: PERI1 clock divider setting bits**

These bits indicate the division ratio of the PERI1 clock from the HPM clock.

Value	Description
0000	No division
0001	Divided by 2
0010	Divided by 3
...	
1101	Divided by 14
1110	Divided by 15
1111	Divided by 16

**[bit3:0] PERIODIV[3:0]: PERIO clock divider setting bits**

These bits indicate the division ratio of the PERIO clock from the HPM clock.

Value	Description
0000	No division
0001	Divided by 2
0010	Divided by 3
...	...
1101	Divided by 14
1110	Divided by 15
1111	Divided by 16



### 5.4.7. APPLIED PLL Control Register (SYSC\_APPLLCNTR)

APPLIED PLL control register (SYSC\_APPLLCNTR) indicates the division and the multiplier of the PLL to be updated.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	WPS							
INITIAL_VALUE	000000000							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved	PLLDIVN						
ACCESS_TYPE	R0,WX	R,WX						
PROT_TYPE	WPS							
INITIAL_VALUE	0	0001100						

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved				PLLDIVM			
ACCESS_TYPE	R0,WX				R,WX			
PROT_TYPE	WPS							
INITIAL_VALUE	0000				0001			

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved						PLLDIVL	
ACCESS_TYPE	R0,WX						R,WX	
PROT_TYPE	WPS							
INITIAL_VALUE	000000						00	

[bit31:23] Reserved: Reserved bits

#### [bit22:16] PLLDIVN[6:0]: PLL clock N-multiplier setting bits

These bits indicate the multiplication rate of the PLL clock.

Value	Description
000_0000	Setting is prohibited
...	Setting is prohibited
000_1011	Setting is prohibited
000_1100	Multiply by 13
000_1101	Multiply by 14
000_1110	Multiply by 15
...	
110_0001	Multiply by 98
110_0010	Multiply by 99
110_0011	Multiply by 100
110_0100	Setting is prohibited
...	Setting is prohibited
111_1111	Setting is prohibited

**[bit15:12] Reserved: Reserved bits**

**[bit11:8] PLLDIVM[3:0]: PLL clock M-divider setting bits**

These bits indicate the division ratio for PLL clock output.

Value	Description
0000	Reserved (divided by 2)
0001	Divided by 2
0010	Divided by 4
0011	Divided by 6
0010	Divided by 8
0101	Divided by 10
0110	Divided by 12
0111	Divided by 14
1000	Divided by 16
1001	Divided by 18
1010	Divided by 20
1011	Divided by 22
1010	Divided by 24
1101	Divided by 26
1110	Divided by 28
1111	Divided by 30

**[bit7:2] Reserved: Reserved bits**

**[bit1:0] PLLDIVL[1:0]: PLL input clock divider setting bits**

These bits indicate the division ratio of the PLL input clock.

Value	Description
00	No division
01	Divided by 2
10	Divided by 4
11	Divided by 6





### 5.4.8. APPLIED Low-voltage Detection Configuration Register (SYSC\_APPLVDCFGR)

APPLIED low-voltage detection configuration register (SYSC\_APPLVDCFGR) indicates each internal low-voltage detections.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved				SV12			LVDE12
ACCESS_TYPE	R0,WX				R1,WX	R0,WX	R0,WX	R1,WX
PROT_TYPE	WPS							
INITIAL_VALUE	0000				100			1

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved	LVDS50	Reserved		SV50			LVDE50
ACCESS_TYPE	R0,WX	R,WX	R0,WX		R,WX			R,WX
PROT_TYPE	WPS							
INITIAL_VALUE	0	0	00		110			1

[bit31:20] Reserved: Reserved bits

[bit19:17] SV12[1:0]: 1.2V reference voltage setting bits

These bits indicate detection voltage of low-voltage detection (1.2V).

Value	Description
100	0.9V

[bit16] LVDE12: 1.2V low-voltage detection operation enable bit

This bit indicates the low-voltage detection operation.

Value	Description
0	-
1	Enable operation.

[bit15:7] Reserved: Reserved bits

**[bit6] LVDS50: low-voltage detection (5.0V) operation selection bit**

This bit indicates the low-voltage detection operation.

Value	Description
0	Reset is generated
1	Interrupt is generated

**[bit5:4] Reserved: Reserved bits**

**[bit3:1] SV50[2:0]: 5.0V reference voltage setting**

These bits indicate detection voltage of low-voltage detection (5.0V).

Value	Description
000	Setting is prohibited
001	Setting is prohibited
010	Setting is prohibited
011	Setting is prohibited
100	Setting is prohibited
101	3.9 V
110	4.1 V
111	4.3 V

**Note:**

- When these bits are changed, low-voltage detection is unavailable until the low-voltage detection stability time elapses.

**[bit0] LVDE50: 5.0V low-voltage detection operation enable bit**

This bit indicates the low-voltage detection operation.

Value	Description
0	Disable operation
1	Enable operation.

**Note:**

- After low-voltage detection is enabled, low-voltage detection is unavailable until the low-voltage detection stability time elapses.



### 5.4.9. APPLIED Clock Supervisor Configuration Register (SYSC\_APPCSVCFGR)

APPLIED clock supervisor configuration register (SYSC\_APPCSVCFGR) indicates whether to enable or disable operation of each clock supervisor.

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	WPS
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved					PLLCSVE	Reserved	MOCSVE
ACCESS_TYPE	R0,WX					R,WX	R0,WX	R,WX
PROT_TYPE	WPS							
INITIAL_VALUE	00000					0	0	0

**[bit31:3] Reserved: Reserved bits**

#### **[bit2] PLLCSVE: PLL clock supervisor enable bit**

This bit indicates whether to enable or disable the PLL clock supervisor.

Value	Description
0	Disable the clock supervisor
1	Enable the clock supervisor

**[bit1] Reserved: Reserved bit**

#### **[bit0] MOCSVE: Main clock supervisor enable bit**

This bit indicates whether to enable or disable the main clock supervisor.

Value	Description
0	Disable the clock supervisor
1	Enable the clock supervisor

## 5.5. Status Profile Register Group

The registers in this group are Status profile control setting registers.



### 5.5.1. Status Clock Source Enable Register (SYSC\_STCKSRER)

Status clock source enable register (SYSC\_STCKSRER) indicates the value for setting whether to enable/disable oscillation of the source clock.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved			PLLRDY	Reserved	MOSCRDY	SCR OSCRDY	CR OSCRDY
ACCESS_TYPE	R0,WX			R,WX	R0,WX	R,WX	R,WX	R,WX
PROT_TYPE	WPS							
INITIAL_VALUE	000			0	0	0	0	0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved			PLLEN	Reserved	MOSCEN	SCR OSCEN	CR OSCEN
ACCESS_TYPE	R0,WX			R,WX	R0,WX	R,WX	R,WX	R,WX
PROT_TYPE	WPS							
INITIAL_VALUE	000			0	0	1	1	1

[bit31:21] Reserved: Reserved bits

[bit20] PLLRDY: PLL clock oscillation stabilization bit

This bit indicates the status of the PLL clock oscillation.

Value	Description
0	Stabilization wait state
1	Stable state (RDY state)

[bit19] Reserved: Reserved bit

[bit18] MOSCRDY: Main clock oscillation stabilization bit

This bit indicates the status of the main clock oscillation.

Value	Description
0	Stabilization wait state
1	Stable state (RDY state)

**[bit17] SCROSCRDY: Slow-CR clock oscillation stabilization bit**

This bit indicates the status of the slow-CR clock oscillation.

Value	Description
0	Stabilization wait state
1	Stable state (RDY state)

**Note:**

- The read value is always "1" since this bit can be read after oscillation stabilization.

**[bit16] CROSCRDY: Fast-CR clock oscillation stabilization bit**

This bit indicates the status of the fast-CR clock oscillation.

Value	Description
0	Stabilization wait state
1	Stable state (RDY state)

**Note:**

- The read value is always "1" since this bit can be read after oscillation stabilization.

**[bit15:5] Reserved: Reserved bits**

**[bit4] PLEN: PLL clock oscillation enable bit**

This bit indicates the set value for PLL clock oscillation control.

Value	Description
0	Disable clock oscillation
1	Enable clock oscillation

**[bit3] Reserved: Reserved bit**

**[bit2] MOSCEN: Main clock oscillation enable bit**

This bit indicates the set value for main clock oscillation control.

Value	Description
0	Disable clock oscillation
1	Enable clock oscillation

**[bit1] SCROSCEN: Slow-CR clock oscillation enable bit**

This bit indicates the set value for slow-CR clock oscillation control.

Value	Description
0	Disable clock oscillation
1	Enable clock oscillation



**[bit0] CROSCEN: Fast-CR clock oscillation enable bit**

This bit indicates the set value for fast-CR clock oscillation control.

Value	Description
0	Disable clock oscillation
1	Enable clock oscillation

### 5.5.2. Status Clock Selection Register 0 (SYSC\_STSCSELR0)

Status clock selection register 0 (SYSC\_STSCSELR0) indicates the set values of the clock source for the clock domain.

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	WPS
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved	CD0CM			Reserved	CD0CSL		
ACCESS_TYPE	R0,WX	R,WX			R0,WX	R,WX		
PROT_TYPE	WPS							
INITIAL_VALUE	0	000			0	000		

[bit31:7] Reserved: Reserved bits

#### [bit6:4] CD0CM[2:0]: Clock domain 0 clock selection monitoring bits

These bits indicate the state the source clock for the clock domain 0.

Value	Description
000	Fast-CR clock
001	Slow-CR clock
010	Main clock
011	Clock fixed at "L"
100	PLL clock
101	Clock fixed at "L"
110	Clock fixed at "L"
111	Clock fixed at "L"

[bit3] Reserved: Reserved bit

#### [bit2:0] CD0CSL[2:0]: Clock domain 0 clock selection bits

These bits indicate the setting value for selection of the source clock for the clock domain 0.

Value	Description
000	Fast-CR clock
001	Slow-CR clock
010	Main clock
011	Clock fixed at "L"
100	PLL clock
101	Clock fixed at "L"
110	Clock fixed at "L"
111	Clock fixed at "L"





### 5.5.3. Status Clock Enable Register (SYSC\_STSCER)

The status clock enable register (SYSC\_STSCER) indicates the set value for stopping or enabling the internal clock.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	ENCLKTRC	Reserved						
ACCESS_TYPE	R,WX	R0,WX						
PROT_TYPE	WPS							
INITIAL_VALUE	1	0000000						

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved		ENCLK PERI7	ENCLK PERI6	ENCLK PERI5	ENCLK PERI4	ENCLK PERI1	ENCLK PERI0
ACCESS_TYPE	R0,WX		R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	WPS							
INITIAL_VALUE	00		1	1	1	1	1	1

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved	ENCLK SYSCPD1	Reserved		ENCLK EXTBUS	Reserved	ENCLK MEMC	ENCLK DMA
ACCESS_TYPE	R0,WX	R,WX	R0,WX		R,WX	R0,WX	R,WX	R,WX
PROT_TYPE	WPS							
INITIAL_VALUE	0	1	00		1	0	1	1

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved	ENCLK HPMPD2	ENCLK ATB	ENCLK DBG	Reserved		ENCLK CPU1	ENCLK CPU0
ACCESS_TYPE	R0,WX	R,WX	R,WX	R,WX	R0,WX		R,WX	R,WX
PROT_TYPE	WPS							
INITIAL_VALUE	0	1	1	1	00		1	1

#### [bit31] ENCLKTRC: TRC internal operation clock oscillation enable bit

This bit indicates whether to disable or enable TRC internal clock oscillation.

Value	Description
0	Disable clock oscillation
1	Enable clock oscillation

#### [bit30:22] Reserved: Reserved bits

**[bit21] ENCLKPERI7: PERI7 internal operation clock oscillation enable bit**

This bit indicates whether to disable or enable PERI7 internal clock oscillation.

Value	Description
0	Disable clock oscillation
1	Enable clock oscillation

**[bit20] ENCLKPERI6: PERI6 internal operation clock oscillation enable bit**

This bit indicates whether to disable or enable PERI6 internal clock oscillation.

Value	Description
0	Disable clock oscillation
1	Enable clock oscillation

**[bit19] ENCLKPERI5: PERI5 internal operation clock oscillation enable bit**

This bit indicates whether to disable or enable PERI5 internal clock oscillation.

Value	Description
0	Disable clock oscillation
1	Enable clock oscillation

**[bit18] ENCLKPERI4: PERI4 internal operation clock oscillation enable bit**

This bit indicates whether to disable or enable PERI4 internal clock oscillation.

Value	Description
0	Disable clock oscillation
1	Enable clock oscillation

**[bit17] ENCLKPERI1: PERI1 internal operation clock oscillation enable bit**

This bit indicates whether to disable or enable PERI1 internal clock oscillation.

Value	Description
0	Disable clock oscillation
1	Enable clock oscillation

**[bit16] ENCLKPERI0: PERI0 internal operation clock oscillation enable bit**

This bit indicates whether to disable or enable PERI0 internal clock oscillation.

Value	Description
0	Disable clock oscillation
1	Enable clock oscillation

**[bit15] Reserved: Reserved bit**

**[bit14] ENCLKSYSCPD1: SYSCPD1 internal operation clock oscillation enable bit**

This bit indicates whether to disable or enable SYSPD1 internal clock oscillation.

Value	Description
0	Disable clock oscillation
1	Enable clock oscillation



**[bit13:12] Reserved: Reserved bits**

**[bit11] ENCLKEXTBUS: EXTBUS internal operation clock oscillation enable bit**

This bit indicates whether to disable or enable EXTBUS internal clock oscillation.

Value	Description
0	Disable clock oscillation
1	Enable clock oscillation

**[bit10] Reserved: Reserved bit**

**[bit9] ENCLKMEMC: MEMC internal operation clock oscillation enable bit**

This bit indicates whether to disable or enable MEMC internal clock oscillation.

Value	Description
0	Disable clock oscillation
1	Enable clock oscillation

**[bit8] ENCLKDMA: DMA internal operation clock oscillation enable bit**

This bit indicates whether to disable or enable DMA internal clock oscillation.

Value	Description
0	Disable clock oscillation
1	Enable clock oscillation

**[bit7] Reserved: Reserved bit**

**[bit6] ENCLKHPMPD2: HPMPD2 internal operation clock oscillation enable bit**

This bit indicates whether to disable or enable HPMPD2 internal clock oscillation.

Value	Description
0	Disable clock oscillation
1	Enable clock oscillation.

**[bit5] ENCLKATB: ATB internal operation clock oscillation enable bit**

This bit indicates whether to disable or enable ATB internal clock oscillation.

Value	Description
0	Disable clock oscillation
1	Enable clock oscillation

**[bit4] ENCLKPERIO: DBG internal operation clock oscillation enable bit**

This bit indicates whether to disable or enable DBG internal clock oscillation.

Value	Description
0	Disable clock oscillation
1	Enable clock oscillation

**[bit3:2] Reserved: Reserved bits**

**[bit1] ENCLKCPU1: CPU1 internal operation clock oscillation enable bit**

This bit indicates whether to disable or enable CPU1 internal clock oscillation.

Value	Description
0	Disable clock oscillation
1	Enable clock oscillation

**[bit0] ENCLKCPU0: CPU0 internal operation clock oscillation enable bit**

This bit indicates whether to disable or enable CPU0 internal clock oscillation.

Value	Description
0	Disable clock oscillation
1	Enable clock oscillation



### 5.5.4. Status Clock Divider Register 0 (SYSC\_STSCKDIVR0)

Status clock divider register 0 (SYSC\_STSDIVR0) indicates the set value of a clock division ratio.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved		HPMDIV		Reserved		TRCDIV	
ACCESS_TYPE	R0,WX		R,WX		R0,WX		R,WX	
PROT_TYPE	WPS							
INITIAL_VALUE	00		00		00		00	

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved		ATBDIV		Reserved		DBGDIV	
ACCESS_TYPE	R0,WX		R,WX		R0,WX		R,WX	
PROT_TYPE	WPS							
INITIAL_VALUE	00		00		00		01	

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved			SYSDIV				
ACCESS_TYPE	R0,WX			R,WX				
PROT_TYPE	WPS							
INITIAL_VALUE	000			00000				

[bit31:30] Reserved: Reserved bits

[bit29:28] HPMDIV[1:0]: HPM clock divider setting bits

These bits indicate the division ratio of the HPM clock, DMA clock and MEMC clock from the system clock.

Value	Description
00	No division
01	Divided by 2
10	Divided by 4
11	Divided by 8

[bit27:26] Reserved: Reserved bits

**[bit25:24] TRCDIV[1:0]: TRC clock divider setting bits**

These bits indicate the division ratio of the TRC clock from the system clock.

Value	Description
00	No division
01	Divided by 2
10	Divided by 4
11	Divided by 8

**[bit23:22] Reserved: Reserved bits**

**[bit21:20] ATBDIV[1:0]: ATB clock divider setting bits**

These bits indicate the division ratio of the ATB clock from the system clock.

Value	Description
00	No division
01	Divided by 2
10	Divided by 4
11	Divided by 8

**[bit19:18] Reserved: Reserved bits**

**[bit17:16] DBGDIV[1:0]: DBG clock divider setting bits**

These bits indicate the division ratio of the DBG clock from the ATB clock.

Value	Description
00	Reserved
01	Divided by 2
10	Reserved
11	Reserved

**Notes:**

- The read values are always "0b01".
- Writing this bit has no effect.

**[bit15:5] Reserved: Reserved bits**

**[bit4:0] SYSDIV[4:0]: SYS clock divider setting bits**

These bits indicate the division ratio of the system clock from the source clock.

Value	Description
0_0000	No division
0_0001	Divided by 2
0_0010	Divided by 3
...	
1_1101	Divided by 30
1_1110	Divided by 31
1_1111	Divided by 32



### 5.5.5. Status Clock Divider Register 1 (SYSC\_STSCKDIVR1)

Status clock divider register 1 (SYSC\_STSCKDIVR1) indicates the set value of a clock division ratio.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved				SYSCPD1DIV			
ACCESS_TYPE	R0,WX				R,WX			
PROT_TYPE	WPS							
INITIAL_VALUE	0000				0000			

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved	EXTBUSDIV			Reserved			
ACCESS_TYPE	R0,WX	R,WX			R0,WX			
PROT_TYPE	WPS							
INITIAL_VALUE	0	000			0000			

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							

[bit31:28] Reserved: Reserved bits

#### [bit27:24] SYSCPD1DIV[3:0]: SYSC\_PD1 clock divider setting bits

These bits indicate the division ratio of the SYSPD1 clock from the HPM clock.

Value	Description
0000	No division
0001	Divided by 2
0010	Divided by 3
...	
1101	Divided by 14
1110	Divided by 15
1111	Divided by 16

[bit23:15] Reserved: Reserved bits

**[bit14:12] EXTBUSDIV [2:0]: EXTBUS clock divider setting bits**

These bits indicate the division ratio of the EXTBUS clock from the HPM clock.

Value	Description
000	No division
001	Divided by 2
010	Divided by 4
011	Divided by 8
100	Divided by 16
101	Divided by 32
110	Divided by 64
111	Divided by 128

**[bit11:0] Reserved: Reserved bits**





### 5.5.6. Status Clock Divider Register 2 (SYSC\_STSCKDIVR2)

Status clock divider register 2 (SYSC\_STSDIVR2) indicates the set value of a clock division ratio.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	PERI7DIV				PERI6DIV			
ACCESS_TYPE	R,WX				R,WX			
PROT_TYPE	WPS							
INITIAL_VALUE	0000				0000			

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	PERI5DIV				PERI4DIV			
ACCESS_TYPE	R,WX				R,WX			
PROT_TYPE	WPS							
INITIAL_VALUE	0000				0000			

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	PERI1DIV				PERIODIV			
ACCESS_TYPE	R,WX				R,WX			
PROT_TYPE	WPS							
INITIAL_VALUE	0000				0000			

[bit31:24] Reserved: Reserved bits

#### [bit23:20] PERI7DIV[3:0]: PERI7 clock divider setting bits

These bits indicate the division ratio of the PERI7 clock from the PERI5 clock.

Value	Description
0000	No division
0001	Divided by 2
0010	Divided by 3
...	...
1101	Divided by 14
1110	Divided by 15
1111	Divided by 16

**[bit19:16] PERI6DIV[3:0]: PERI6 clock divider setting bits**

These bits indicate the division ratio of the PERI6 clock from the PERI4 clock.

Value	Description
0000	No division
0001	Divided by 2
0010	Divided by 3
...	
1101	Divided by 14
1110	Divided by 15
1111	Divided by 16

**[bit15:12] PERI5DIV[3:0]: PERI5 clock divider setting bits**

These bits indicate the division ratio of the PERI5 clock from the HPM clock.

Value	Description
0000	No division
0001	Divided by 2
0010	Divided by 3
...	
1101	Divided by 14
1110	Divided by 15
1111	Divided by 16

**[bit11:8] PERI4DIV[3:0]: PERI4 clock divider setting bits**

These bits indicate the division ratio of the PERI4 clock from the HPM clock.

Value	Description
0000	No division
0001	Divided by 2
0010	Divided by 3
...	
1101	Divided by 14
1110	Divided by 15
1111	Divided by 16

**[bit7:4] PERI1DIV[3:0]: PERI1 clock divider setting bits**

These bits indicate the division ratio of the PERI1 clock from the PERI0 clock.

Value	Description
0000	No division
0001	Divided by 2
0010	Divided by 3
...	
1101	Divided by 14
1110	Divided by 15
1111	Divided by 16



**[bit3:0] PERIODIV[3:0]: PERIO clock divider setting bits**

These bits indicate the division ratio of the PERIO clock from the HPM clock.

Value	Description
0000	No division
0001	Divided by 2
0010	Divided by 3
...	
1101	Divided by 14
1110	Divided by 15
1111	Divided by 16

### 5.5.7. Status PLL Control Register (SYSC\_STSPLLCNTR)

Status PLL control register (SYSC\_STSPLLCNTR) indicates the set values of the division ratio, multiplication rate, etc. for PLL.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	WPS							
INITIAL_VALUE	000000000							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved	PLLDIVN						
ACCESS_TYPE	R0,WX	R,WX						
PROT_TYPE	WPS							
INITIAL_VALUE	0	0001100						

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved				PLLDIVM			
ACCESS_TYPE	R0,WX				R,WX			
PROT_TYPE	WPS							
INITIAL_VALUE	0000				0001			

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved						PLLDIVL	
ACCESS_TYPE	R0,WX						R,WX	
PROT_TYPE	WPS							
INITIAL_VALUE	000000						00	

[bit31:23] Reserved: Reserved bits

#### [bit22:16] PLLDIVN[6:0]: PLL clock N-multiplier setting bits

These bits indicate the set value of the N multiplication rate of the PLL clock.

Value	Description
000_0000	Setting prohibited
...	Setting prohibited
000_1011	Setting prohibited
000_1100	Multiply by 13
000_1101	Multiply by 14
000_1110	Multiply by 15
...	
110_0001	Multiply by 98
110_0010	Multiply by 99
110_0011	Multiply by 100
110_0100	Setting prohibited
...	Setting prohibited
111_1111	Setting prohibited



**[bit15:12] Reserved: Reserved bits**

**[bit11:8] PLLDIVM[3:0]: PLL clock M-divider setting bits**

These bits indicate the set value of the M division ratio of the PLL output clock.

Value	Description
0000	Reserved (Divided by 2)
0001	Divided by 2
0010	Divided by 4
0011	Divided by 6
0010	Divided by 8
0101	Divided by 10
0110	Divided by 12
0111	Divided by 14
1000	Divided by 16
1001	Divided by 18
1010	Divided by 20
1011	Divided by 22
1010	Divided by 24
1101	Divided by 26
1110	Divided by 28
1111	Divided by 30

**[bit7:2] Reserved: Reserved bits**

**[bit1:0] PLLDIVL[1:0]: PLL input clock divider setting bits**

These bits indicate the set value of the L division ratio of the PLL input clock.

Value	Description
00	No division
01	Divided by 2
10	Divided by 4
11	Divided by 6

### 5.5.8. Status Low-voltage Detection Configuration Register (SYSC\_STSLVDCFGR)

Status low-voltage detection configuration register (SYSC\_STSLVDCFGR) indicates the status of low-voltage detections.

BITS	31	30	29	28	27	26	25	24
BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	VDRDY12	Reserved			SV12			LVDE12
ACCESS_TYPE	R,WX	R0,WX			R1,WX	R0,WX	R0,WX	R1,WX
PROT_TYPE	WPS							
INITIAL_VALUE	1	000			100			1

BITS	15	14	13	12	11	10	9	8
BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	VDRDY50	LVDS50	Reserved		SV50			LVDE50
ACCESS_TYPE	R,WX	R,WX	R0,WX		R,WX			R,WX
PROT_TYPE	WPS							
INITIAL_VALUE	1	0	00		110			1

**[bit31:24] Reserved: Reserved bits**

**[bit23] VDRDY12: 1.2V low-voltage detection ready status bit**

This bit indicates the low-voltage detection operation.

Value	Description
0	Wait for stabilization
1	Monitoring

**[bit22:20] Reserved: Reserved bits**

**[bit19:17] SV12[2:0]: 1.2V reference voltage setting bits**

These bits indicate the detection voltage of low-voltage detection (1.2V).

Value	Description
100	0.9V



**[bit16] LVDE12: 1.2V low-voltage detection operation enable bit**

This bit indicates the low-voltage detection operation.

Value	Description
0	-
1	Enable operation.

**[bit15:8] Reserved: Reserved bits**

**[bit7] VDRDY50: Low-voltage detection (5.0V) ready status bit**

This bit indicates the low-voltage detection operation.

Value	Description
0	Wait for stabilization or disenabled
1	Monitoring

**[bit6] LVDS50: Low-voltage detection (5.0V) operation selection bit**

This bit indicates the operation at the low-voltage detection.

Value	Description
0	Reset is generated
1	Interrupt is generated

**[bit5:4] Reserved: Reserved bits**

**[bit3:1] SV50[2:0]: 5.0V reference voltage setting bits**

These bits indicate detection voltage of low-voltage detection (5V).

Value	Description
000	Setting is prohibited
001	Setting is prohibited
010	Setting is prohibited
011	Setting is prohibited
100	Setting is prohibited
101	3.9 V
110	4.1 V
111	4.3 V

**Note:**

- When these bits are changed, low-voltage detection is unavailable until the low-voltage detection stability time elapses.

**[bit0] LVDE50: 5.0V low-voltage detection operation enable bit**

This bit indicates the low-voltage detection operation.

Value	Description
0	Disable operation
1	Enable operation.

**Note:**

- *After low-voltage detection is enabled, low-voltage detection is unavailable until the low-voltage detection stability time elapses.*





### 5.5.9. Status Clock Supervisor Configuration Register (SYSC\_STSCSVCFGR)

Status clock supervisor configuration register (SYSC\_STSCSVCFGR) indicates the value for setting whether to enable/disable operation of each clock supervisor.

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	WPS
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved					PLLCSVE	Reserved	MOCSVE
ACCESS_TYPE	R0,WX					R,WX	R0,WX	R,WX
PROT_TYPE	WPS							
INITIAL_VALUE	00000					0	0	0

**[bit31:3] Reserved: Reserved bits**

#### **[bit2] PLLCSVE: PLL clock supervisor enable bit**

This bit indicates whether to enable or disable the PLL clock supervisor.

Value	Description
0	Disable the clock supervisor
1	Enable the clock supervisor

**[bit1] Reserved: Reserved bit**

#### **[bit0] MOCSVE: Main clock supervisor enable bit**

This bit indicates whether to enable or disable the main clock supervisor.

Value	Description
0	Disable the clock supervisor
1	Enable the clock supervisor

## 5.6. System Register Group

The registers in this group are system control-related registers.



### 5.6.1. System Status Register (SYSC\_SYSTSR)

This register indicates each transition control status, the CPU status, the system operation status, and the transition control completion flag.

BIT_OFFSET	31-16
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	WPS
INITIAL_VALUE	00000000_00000000

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	PSSSTS1	RUNSTS1	PSSDF1	Reserved			CPUSTS1	DVSTS1
ACCESS_TYPE	R,WX	R,WX	R,WX	R0,WX			R,WX	R,WX
PROT_TYPE	WPS							
INITIAL_VALUE	0	0	0	000			0	1

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	PSSSTS0	RUNSTS0	PSSDF0	RUNDF0	Reserved		CPUSTS0	DVSTS0
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R0,WX		R,WX	R,WX
PROT_TYPE	WPS							
INITIAL_VALUE	0	0	0	0	00		0	1

[bit31:16] Reserved: Reserved bits

#### [bit15] PSSSTS1: PSS profile (sub status control 1) update status bit

This bit indicates the PSS profile (sub status control 1) update status.

Value	Description
0	The PSS profile update has completed, or the profile was not updated.
1	The PSS profile is being updated.

**Note:**

- This register does not become "1" immediately after the PSS profile update. The register becomes "1" after operation of the internal circuit.

#### [bit14] RUNSTS1: RUN profile (sub status control 1) update status bit

This bit indicates the RUN profile (sub status control 1) update status.

Value	Description
0	The RUN profile update has completed, or the profile was not updated.
1	The RUN profile is being updated.

**Note:**

- This register does not become "1" immediately after the PSS profile update. The register becomes "1" after operation of the internal circuit.

**[bit13] PSSDF1: PSS profile (sub status control 1) update completion flag bit**

This bit indicates the PSS profile (sub status control 1) update completion.

Value	Description
0	The PSS profile is being updated, or there is no update.
1	The PSS profile has been completed.

**Notes:**

- To clear this register, write "1" in SYSC\_SYSCCLR:PSSDFCLR1.
- The flags of this register and the RUNSTS0 register may appear to be "1" at circuit operation. This state shows that the RUN profile update has completed.

**[bit12:10] Reserved: Reserved bits**

**[bit9] CPUSTS1: CPU1 device status bit**

This bit indicates the device status of CPU1.

Value	Description
0	Operation status
1	WFI status

**[bit8] DVSTS1: Device status of sub status control 1 bit**

This bit indicates the device status of sub status control 1.

Value	Description
0	PSS
1	RUN

**[bit7] PSSSTS0: PSS profile (main status control) update status bit**

This bit indicates the PSS profile (main status control) update status.

Value	Description
0	The PSS profile update has completed, or the profile was not updated.
1	The PSS profile is being updated.

**Note:**

- This register does not become "1" immediately after the PSS profile update. The register becomes "1" after operation of the internal circuit.

**[bit6] RUNSTS0: RUN profile (main status control) update status bit**

This bit indicates the RUN profile (main status control) update status.

Value	Description
0	The RUN profile update has completed, or the profile was not updated.
1	The RUN profile is being updated.

**Note:**

- This register does not become "1" immediately after the PSS profile update. The register becomes "1" after operation of the internal circuit.



**[bit5] PSSDF0: PSS profile (main status control) update completion flag bit**

This bit indicates the PSS profile (main status control) update completion.

Value	Description
0	The PSS profile is being updated, or there is no update.
1	The PSS profile has been completed.

**Notes:**

- To clear this register, write "1" in SYSC\_SYSICLR:PSSDFCLR0.
- The flags of this register and the RUNSTS0 register may appear to be "1" at circuit operation. This state shows that the RUN profile update has completed.

**[bit4] RUNDF0: RUN profile (main status control) update completion flag bit**

This bit indicates the RUN profile (main status control) update completion.

Value	Description
0	The RUN profile is being updated, or there is no update.
1	The RUN profile has been completed.

**Notes:**

- To clear this register, write "1" in SYSC\_SYSICLR:RUNDFCLR0.
- To use this register as an interrupt, write "1" in SYSC\_SYSINTER:RUNDIE0.
- The flags of this register and the RUNSTS0 register may appear to be "1" at circuit operation. This state shows that the RUN profile update has completed.

**[bit3:2] Reserved: Reserved bits**

**[bit1] CPUSTS0: CPU0 device status bit**

This bit indicates the device status of CPU0.

Value	Description
0	Operation status
1	WFI status

**[bit0] DVSTS0: Device status of main status control bit**

This bit indicates the device status of main status control.

Value	Description
0	PSS
1	RUN

### 5.6.2. System Status Interrupt Enable Register (SYSC\_SYSINTER)

This register sets whether RUN profile execution completion interrupts are enabled.

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	WPS
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved			RUNDIE0	Reserved			
ACCESS_TYPE	R0,WX			R/W	R0,WX			
PROT_TYPE	WPS							
INITIAL_VALUE	000			0	0000			

**[bit31:5] Reserved: Reserved bits**

**[bit4] RUNDIE0: RUN profile (main status control) update completion interrupt enable bit**

This bit enables RUN profile (main status control) update completion interrupt.

Value	Description
0	Disable
1	Enable

**[bit3:0] Reserved: Reserved bits**



### 5.6.3. System Status Flag and Interrupt Clear Register (SYSC\_SYSICLR)

This register clears the RUN/PSS profile completion flag bit.

BIT_OFFSET	31-16
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	WPS
INITIAL_VALUE	00000000_00000000

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved		PSSDF CLR1	Reserved				
ACCESS_TYPE	R0,WX		R0,W	R0,WX				
PROT_TYPE	WPS							
INITIAL_VALUE	00		0	00000				

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved		PSSDF CLR0	RUNDF CLR0	Reserved			
ACCESS_TYPE	R0,WX		R0,W	R0,W	R0,WX			
PROT_TYPE	WPS							
INITIAL_VALUE	00		0	0	0000			

**[bit31:14] Reserved: Reserved bits**

#### **[bit13] PSSDFCLR1: PSS profile (sub status control 1) update completion flag clear bit**

This bit clears the PSS profile update (sub status control 1) completion flag bit.

Value	Description
0	No effect
1	Clear the flag

**Note:**

- This bit can also be subject to write access by CPU1.

**[bit12:6] Reserved: Reserved bits**

#### **[bit5] PSSDFCLR0: PSS profile (main status control) update completion flag clear bit**

This bit clears the PSS profile (main status control) update completion flag bit.

Value	Description
0	No effect
1	Clear the flag

**[bit4] RUNDFCLR0: RUN profile (main status control) update completion flag clear bit**

This bit clears the RUN profile (main status control) update completion flag bit.

Value	Description
0	No effect
1	Clear the flag

**[bit3:0] Reserved: Reserved bits**





#### 5.6.4. System Error Interrupt Factor Register 0 (SYSC\_SYSEIR0)

This register indicates the factors of low-voltage detection (interrupt) and clock supervisor interrupt requests.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							LVD50IF
ACCESS_TYPE	R0,WX							R,WX
PROT_TYPE	WPS							
INITIAL_VALUE	0000000							0

BIT_OFFSET	23-8							
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	WPS							
INITIAL_VALUE	00000000_00000000							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved			SSPMIF	Reserved	PMIF	Reserved	MOMIF
ACCESS_TYPE	R0,WX			R,WX	R0,WX	R,WX	R0,WX	R,WX
PROT_TYPE	WPS							
INITIAL_VALUE	000			0	0	0	0	0

[bit31:25] Reserved: Reserved bits

##### [bit24] LVD50IF: 5.0V low-voltage detection interrupt request bit

This bit retains the fact that an interrupt request was generated at the detection of internal low voltage.

Value	Description
0	No interrupt request is detected.
1	An interrupt request is detected.

[bit23:5] Reserved: Reserved bits

##### [bit4] SSPMIF: Sub-system PLL oscillation abnormal monitor detection error interrupt request bit

This bit retains the fact that an interrupt request was generated for a sub oscillation abnormal monitor detection error.

Value	Description
0	No interrupt request is detected.
1	An interrupt request is detected.

[bit3] Reserved: Reserved bit

**[bit2] PMIF: PLL oscillation abnormal monitor detection error interrupt request bit**

This bit retains the fact that an interrupt request was generated for a main oscillation abnormal monitor detection error.

Value	Description
0	No interrupt request is detected.
1	An interrupt request is detected.

**[bit1] Reserved: Reserved bit**

**[bit0] MOMIF: Main oscillation abnormality detection error interrupt request bit**

This bit retains the fact that an interrupt request was generated for a main oscillation abnormality detection error.

Value	Description
0	No interrupt request is detected.
1	An interrupt request is detected.



### 5.6.5. System Error Interrupt Factor Register 1 (SYSC\_SYSEIRIR1)

This register retains the factors of profile error interrupt requests.

BIT_OFFSET	31-16
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	WPS
INITIAL_VALUE	00000000_00000000

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved				PSSENE RRIF1	PSSTRG CIF1	Reserved	
ACCESS_TYPE	R0,WX				R,WX	R,WX	R0,WX	
PROT_TYPE	WPS							
INITIAL_VALUE	0000				0	0	00	

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved	PSSERR IF0	RUNWK ERRIF0	RUNERR IF0	PSSENER RIF0	PSSTRGC IF0	RUNTR GERRIF	TRGERRIF
ACCESS_TYPE	R0,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	WPS							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:12] Reserved: Reserved bits

#### [bit11] PSSENERRIF1: PSS profile update enable write error interrupt request bit

This bit retains the fact that an interrupt request was generated for a write error in the PSS profile update enable register (SYSC\_PSSENR:PSSEN1).

Value	Description
0	No interrupt request is detected.
1	An interrupt request is detected.

#### [bit10] PSSTRGCIF1: PSS trigger cancel interrupt request bit

This bit retains the fact that a cancel request was generated during a PSS profile (sub status control 1) update.

Value	Description
0	No interrupt request is detected.
1	An interrupt request is detected.

[bit9:7] Reserved: Reserved bits

**[bit6] PSSERRIF0: PSS profile error interrupt request bit**

This bit retains the fact that an interrupt request was generated for a PSS profile error.

Value	Description
0	No interrupt request is detected.
1	An interrupt request is detected.

**[bit5] RUNWKERRIF0: RUN profile (PSS recovery time) error interrupt request bit**

This bit retains the fact that an interrupt request was generated for a RUN profile error.

Value	Description
0	No interrupt request is detected.
1	An interrupt request is detected.

**[bit4] RUNERRIF0: RUN profile error interrupt request bit**

This bit retains the fact that an interrupt request was generated for an error during a RUN profile update.

Value	Description
0	No interrupt request is detected.
1	An interrupt request is detected.

**[bit3] PSSENERIF0: PSS profile update enable write error interrupt request bit**

This bit retains the fact that an interrupt request was generated for a write error in the PSS profile update enable register (SYSC\_PSSENR:PSSEN0).

Value	Description
0	No interrupt request is detected.
1	An interrupt request is detected.

**[bit2] PSSTRGCIF0: PSS trigger (main status control) cancel interrupt request bit**

This bit retains the fact that a cancel request was generated during a PSS profile (main status control) update.

Value	Description
0	No interrupt request is detected.
1	An interrupt request is detected.

**[bit1] RUNTRGERRIF: RUN profile update enable write error interrupt request bit**

This bit retains the fact that an interrupt request was generated for a write error in the RUN profile update enable register.

Value	Description
0	No interrupt request is detected.
1	An interrupt request is detected.



**[bit0] TRGERRIF: Trigger error interrupt request bit**

This bit retains that fact that an interrupt request was generated for a trigger error that caused an attempt to update the RUN profile again during a RUN profile update.

Value	Description
0	No interrupt request is detected.
1	An interrupt request is detected.

### 5.6.6. System Error Interrupt Factor Clear Register 0 (SYSC\_SYSEERRICLR0)

This register clears low-voltage detection interrupt requests and clock supervisor interrupt requests.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							LVD50ICLR
ACCESS_TYPE	R0,WX							R0,W
PROT_TYPE	WPS							
INITIAL_VALUE	0000000							0

BIT_OFFSET	23-8						
BIT_NAME	Reserved						
ACCESS_TYPE	R0,WX						
PROT_TYPE	WPS						
INITIAL_VALUE	00000000_00000000						

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved			SSPMICLR	Reserved	PMICLR	Reserved	MOMICLR
ACCESS_TYPE	R0,WX			R0,W	R0,WX	R0,W	R0,WX	R0,W
PROT_TYPE	WPS							
INITIAL_VALUE	000			0	0	0	0	0

**[bit31:25] Reserved: Reserved bits**

#### **[bit24] LVD50ICLR: Internal low-voltage detection interrupt request clear bit**

This bit clears the interrupt request of the detection of internal low voltage.

Value	Description
0	No effect
1	Clear interrupt request

**[bit23:5] Reserved: Reserved bits**

#### **[bit4] SSPMICLR: Sub-system PLL oscillation abnormal monitor detection error interrupt request clear bit**

This bit clears the interrupt request of a sub-system PLL oscillation abnormal monitor detection error.

Value	Description
0	No effect
1	Clear interrupt request

**[bit3] Reserved: Reserved bit**



**[bit2] PMICLR: PLL oscillation abnormal monitor detection error interrupt request clear bit**

This bit clears the interrupt request of a PLL oscillation abnormal monitor detection error.

Value	Description
0	No effect
1	Clear interrupt request

**[bit1] Reserved: Reserved bit**

**[bit0] MOMICLR: Main oscillation abnormality detection error interrupt request clear bit**

This bit clears the interrupt request of a main oscillation abnormality detection error.

Value	Description
0	No effect
1	Clear interrupt request

### 5.6.7. System Error Interrupt Factor Clear Register 1 (SYSC\_SYSEERRICLR1)

This register clears profile error interrupt requests.

BIT_OFFSET	31-16
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	WPS
INITIAL_VALUE	00000000_00000000

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved				PSSENE RRICLR1	PSSTRG CICLR1	Reserved	
ACCESS_TYPE	R0,WX				R0,W	R0,W	R0,WX	
PROT_TYPE	WPS							
INITIAL_VALUE	0000				0	0	00	

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved	PSSERRICLR0	RUNWKERRICLR0	RUNERRICLR0	PSSERRICLR0	PSSTRGCICLR0	RUNTRGERRICLR	TRGERRICLR
ACCESS_TYPE	R0,WX	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W
PROT_TYPE	WPS							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:12] Reserved: Reserved bits

[bit11] PSSERRICLR1: PSS profile update write error interrupt request clear bit

Value	Description
0	No effect
1	Clear interrupt request

**Note:**

- This bit can also be subject to write access by CPU1.

[bit10] PSSTRGCICLR1: PSS trigger (sub status control 1) cancel interrupt request clear bit

Value	Description
0	No effect
1	Clear interrupt request

**Note:**

- This bit can also be subject to write access by CPU1.

[bit9:7] Reserved: Reserved bits



**[bit6] PSSERRICLR0: PSS profile error interrupt request clear bit**

Value	Description
0	No effect
1	Clear interrupt request

**[bit5] RUNWKERRICLR0: RUN profile error interrupt request clear bit**

Value	Description
0	No effect
1	Clear interrupt request

**[bit4] RUNERRICLR0: RUN profile error interrupt request clear bit**

Value	Description
0	No effect
1	Clear interrupt request

**[bit3] PSSENERRICLR0: PSS profile update enable write error interrupt request clear bit**

Value	Description
0	No effect
1	Clear interrupt request

**[bit2] PSSTRGCICLR0: PSS trigger (main status control) cancel interrupt request clear bit**

Value	Description
0	No effect
1	Clear interrupt request

**[bit1] RUNTRGERRICLR: RUN profile update enable write error interrupt request clear bit**

Value	Description
0	No effect
1	Clear interrupt request

**[bit0] TRGERRICLR: Trigger error interrupt request clear bit**

Value	Description
0	No effect
1	Clear interrupt request

### 5.6.8. Profile Status Register (SYSC\_SYSPROSTSR)

This register indicates the error status of each profile.

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	WPS
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved					PSSPSTS	RUNWK PSTS	RUNPSTS
ACCESS_TYPE	R0,WX					R,WX	R,WX	R,WX
PROT_TYPE	WPS							
INITIAL_VALUE	00000					0	0	0

**[bit31:3] Reserved: Reserved bits**

#### **[bit2] PSSPSTS: PSS profile setting status bit**

This bit indicates the PSS profile setting status during a transition from RUN to PSS.

Value	Description
0	There is no PSS profile error.
1	There is a PSS profile error.

**Note:**

- We recommend confirming that there is no problem in the PSS profile by checking this register value before the transition to PSS.

#### **[bit1] RUNWKPSTS: RUN profile (in PSS recovery) setting status bit**

This bit indicates the RUN profile setting status during a transition from PSS to RUN.

Value	Description
0	There is no RUN profile error.
1	There is a RUN profile error.

**Note:**

- We recommend confirming that there is no problem in the RUN profile by checking this register value before the transition to PSS.



**[bit0] RUNPSTS: RUN profile setting status bit**

This bit indicates the RUN profile setting status.

Value	Description
0	There is no RUN profile error.
1	There is a RUN profile error.

**Note:**

- *We recommend confirming that there is no problem in the RUN profile by checking this register value when updating the RUN profile.*

### 5.6.9. RUN Profile Error Flag Register (SYSC\_SYSRUNPEFR)

This register indicates each RUN profile error flag.

BIT_OFFSET	31-16
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	WPS
INITIAL_VALUE	00000000_00000000

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved						PEF9	PEF8
ACCESS_TYPE	R0,WX						R,WX	R,WX
PROT_TYPE	WPS							
INITIAL_VALUE	000000						0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	PEF7	PEF6	PEF5	Reserved	PEF3	PEF2	PEF1	PEF0
ACCESS_TYPE	R,WX	R,WX	R,WX	R0,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	WPS							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:10, 4] Reserved: Reserved bits

#### [bit9:5, bit3:0] PEF<n>: Profile error flag bits

These bits indicate the error status of each RUN profile. For detailed contents of profile errors, see "(3) Profile Errors".

Value	Status
0	There is no error.
1	There is an error.

<n> is a number from the following: 9 to 5 and 3 to 0



### 5.6.10. PSS Profile Error Flag Register (SYSC\_SYSPSSPEFR)

This register indicates each PSS profile error flag.

BIT_OFFSET	31-16
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	WPS
INITIAL_VALUE	00000000_00000000

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved					PEF10	PEF9	PEF8
ACCESS_TYPE	R0,WX					R,WX	R,WX	R,WX
PROT_TYPE	WPS							
INITIAL_VALUE	00000					0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	PEF7	PEF6	PEF5	Reserved	PEF3	Reserved	PEF1	PEF0
ACCESS_TYPE	R,WX	R,WX	R,WX	R0,WX	R,WX	R0,WX	R,WX	R,WX
PROT_TYPE	WPS							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:11, 4, 2] Reserved: Reserved bits

#### [bit10:3,bit1:0] PEF<n>: Profile error flag bits

These bits indicate each PSS profile error status. For detailed contents of profile errors, see "(3) Profile Errors".

Value	Status
0	There is no error.
1	There is an error.

<n> is a number from the following: 10 to 3 and 1 to 0.

## 5.7. Special Setting Register Group

The registers in this group are special setting registers.

### 5.7.1. System Special Configuration Register (SYSC\_SPECFGR)

This register sets I/O control.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	PSSPADCT RL	Reserved						
ACCESS_TYPE	R/W	R0,WX						
PROT_TYPE	WPS							
INITIAL_VALUE	0	0000000						

BIT_OFFSET	15-0							
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	WPS							
INITIAL_VALUE	00000000_00000000							

**[bit31:24] Reserved: Reserved bits**

#### **[bit23] PSSPADCTRL: Port configuring bit in the PSS**

This bit sets whether to perform high impedance control of I/O in the PSS.

Value	Description
0	Do not perform high impedance control.
1	Perform high impedance control.

**[bit22:0] Reserved: Reserved bits**



### 5.7.2. CPU Control Register (SYSC\_SPECPUCFGR)

This register is used for making CPU-related control settings.

BIT_OFFSET	31-16
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	WPS
INITIAL_VALUE	00000000_00000000

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved					CPUMD		
ACCESS_TYPE	R0,WX					R/W		
PROT_TYPE	WPS							
INITIAL_VALUE	00000					000		

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved						HEN1	HEN0
ACCESS_TYPE	R0,WX						R,W	R,W
PROT_TYPE	WPS							
INITIAL_VALUE	000000						*	*

[bit31:11] Reserved: Reserved bits

[bit10:8] CPUMD[2:0]: CPU operation mode setting bits

These bits set the operation mode of a CPU.

Value	Description
000	Multi CPU (2CPU mode)
001	Reserved
01X	Reserved
100	Single CPU (1CPU0 mode)
101	Single CPU (1CPU1 mode)
11X	Reserved

X: Don't care

**Note:**

- For details, see "CHAPTER: Operation Mode".

[bit7:2] Reserved: Reserved bits

**[bit1] HEN1: CPU1HALT control bit**

This bit controls the HALT signal for CPU1.

Value	Description
0	Disabled
1	Enabled

**Notes:**

- It is possible to write to this register only once after a hard reset if the HALT signal is enabled.
- The initial value varies depending on the CPU operation mode.
  - Initial value of 2CPU mode: 1
  - Initial value of 1CPU0 mode: 1
  - Initial value of 1CPU1 mode: 0

**[bit0] HEN0: CPU0HALT control bit**

This bit controls the HALT signal for CPU0.

Value	Description
0	Disabled
1	Enabled

**Notes:**

- It is possible to write to this register only once after a hard reset if the HALT signal is enabled.
- The initial value varies depending on the CPU operation mode.
  - Initial value of 2CPU mode: 0
  - Initial value of 1CPU0 mode: 0
  - Initial value of 1CPU1 mode: 1





## 5.8. Debug Register Group

The registers in this group are debug-related registers.

### 5.8.1. JTAG Detection Register (SYSC\_JTAGDETECT)

This register indicates the status of the connection with the debugger.

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved							DBGCON
ACCESS_TYPE	R0,WX							R,WX
PROT_TYPE	-							
INITIAL_VALUE	00000000							0

[bit31:1] Reserved: Reserved bits

#### [bit0] DBGCON: Debugger connection status bit

This bit indicates the status of the connection with the debugger.

Value	Description
0	Not connected
1	Connected

### 5.8.2. JTAG Configuration Register (SYSC\_JTAGCNFG)

This register indicates the completion of debugger settings (for BootROM).

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved							DBGDONE
ACCESS_TYPE	R0,WX							R/W
PROT_TYPE	-							
INITIAL_VALUE	0000000							1

**[bit31:1] Reserved: Reserved bits**

**[bit0] DBGDONE: Debugger status bit**

This bit indicates the debugger setting status.

Value	Description
0	The debugger is connected.
1	The debugger is not connected, or debugger setting has completed.



### 5.8.3. JTAG Wakeup Register (SYSC\_JTAGWAKEUP)

This register sets whether to enable wakeup of a debugger connection.

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved							DBGWKEN
ACCESS_TYPE	R0,WX							R/W
PROT_TYPE	-							
INITIAL_VALUE	00000000							0

[bit31:1] Reserved: Reserved bits

#### [bit0] DBGWKEN: Debugger wakeup enable bit

This bit sets wakeup of a debugger connection.

Value	Description
0	Mask recovery.
1	Enable recovery.

#### Notes:

- To enable JTAG-Wakeup when connecting the debugger, write "1" to this bit.
- If recovery is masked by this bit, JTAG-Wakeup recovery is disabled.

## **6. Other**

This section describes the precautions on use of low-power consumption.

### **6.1. Restrictions on Transitioning to the PSS**

This section describes restrictions on transitioning to the PSS.

When changing to PSS, make sure that the relationship between the CPU clock, the Memory Config clock, and the SCU clock is 1:1:1. If the mutual clock relationship is not maintained, a malfunction may result.





## CHAPTER 7: Low-voltage Detection

This chapter explains the low-voltage detection.

---

1. Overview
2. Configuration
3. Operation
4. Setting Procedure Examples
5. Operation Examples
6. Precautions for Using



## 1. Overview

This section provides an overview of the low-voltage detection.

### Features

#### a) Internal Low-voltage Detection (1.2V)

This section explains the features of internal low-voltage detection.

- Function: An internal low-voltage detection reset is generated upon the detection of a voltage within  $\pm 0.1V$  from the preset detection voltage.
- Preset detection voltage: 0.9V
- Operation: Always active

The RAM contents after a power-on reset cannot be guaranteed.

#### b) External Low-voltage Detection (5.0V)

This section explains the features of external low-voltage detection.

- Function: An external low-voltage detection reset or interrupt is generated upon the detection of a voltage within  $\pm 0.2V$  from the preset detection voltage.
- Detection voltage: The selectable values are 3.9V, 4.1V, and 4.3V.
- Operation: Operation can be switched between the active and inactive states by user settings.

The selection to generate an interrupt or reset is also available.

Furthermore, the settings for RUN/PSS can be switched using a profile.

In the case where a reset is generated, the reset is issued after all clocks are stopped in order to guarantee the RAM contents after an external low-voltage detection reset.

For details, see "CHAPTER: Reset".

#### Note:

- In the MB9D560 series, a low-voltage detection interrupt corresponds to an NMI.

2. Configuration

This section explains block diagrams of the low-voltage detection.  
Figure 2-1 and Figure 2-2 are configuration diagrams of the low-voltage detection.

Configuration diagrams of the low-voltage detection

Figure 2-1 Internal Low-voltage Detection (1.2V)

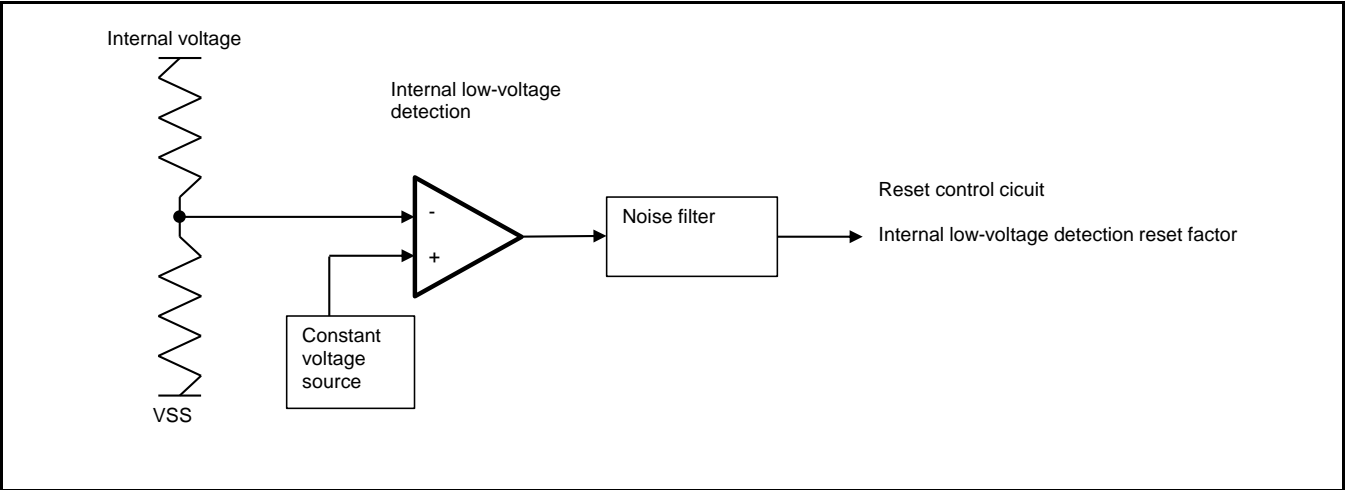
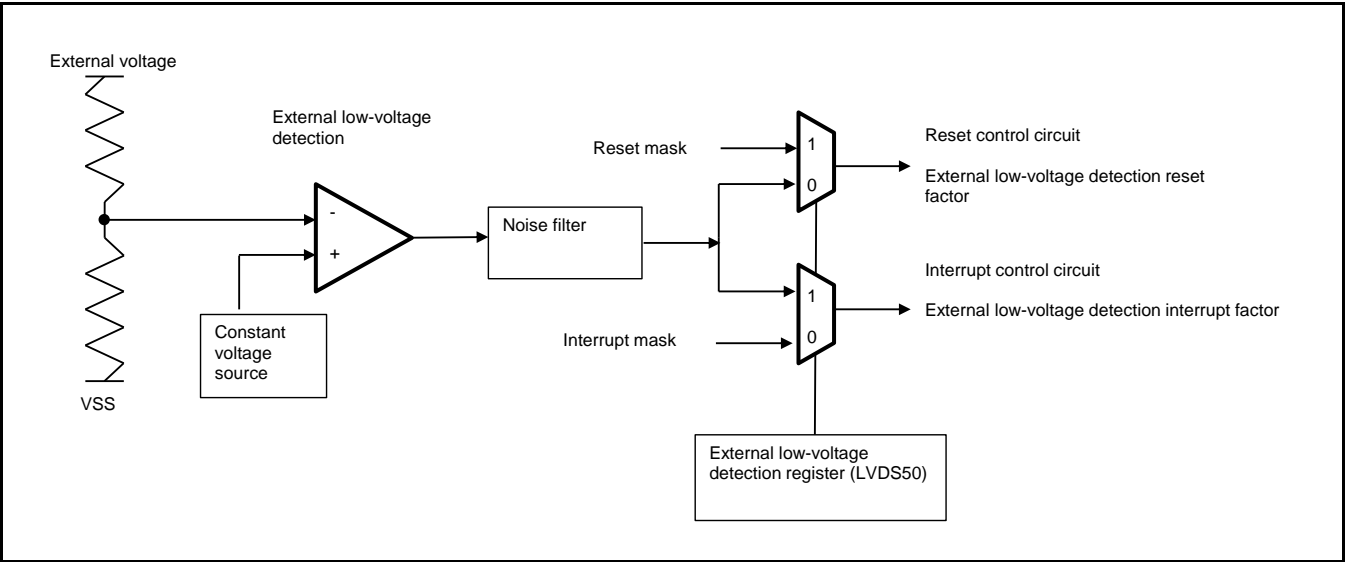


Figure 2-2 External Low-voltage Detection (5.0V)







### 3. Operation

This section explains the operation of the low-voltage detection.

#### (1) Internal Low-voltage Detection Circuit

The internal low-voltage detection circuit enters the monitoring state after power-on and remains in this state. It detects a low voltage when the internal supply voltage drops below the detection voltage. Upon detecting a low voltage, it generates a reset. The low-voltage detection is released when the internal supply voltage becomes higher than the release voltage.

##### a) Explanation of the Internal Low-voltage Detection Operation

The low-voltage detection voltage is fixed. The circuit enters the monitoring state after power-on and remains in this state.

##### b) Internal Low-voltage Detection Reset

An internal low-voltage reset is generated when the internal supply voltage drops below the set voltage. When internal low-voltage detection reset is generated, PONR bit and LVD12R bit (power-on reset detection bit and internal low-voltage detection bit) of user reset factor register (SYSC\_RSTCAUSEUR) are set. For details, see "CHAPTER: Reset".

#### (2) External Low-voltage Detection Circuit

The 5.0V power supply low-voltage detection circuit operates in the same way as the 1.2V power supply low-voltage detection circuit.

The low-voltage detection circuit enters the monitoring state after power-on. It detects a low voltage when the monitored supply voltage drops below the detection voltage. Either an interrupt or reset can be selected to be generated upon the detection of a low voltage. The low-voltage detection is released when the monitored supply voltage becomes higher than the release voltage.

##### a) Explanation of the External Low-voltage Detection Operation

The available selections using the RUN profile low-voltage detection configuration register (SYSC\_RUNLVDCFGR) / PSS profile low-voltage detection configuration register (SYSC\_PSSLVDCFGR) are whether to enable or disable the low-voltage detection (LVDE50 bit), the low-voltage detection voltage (SV50[2:0] bits), and whether to generate a reset or interrupt (LVDS50 bit) when a low voltage is detected. Executing the RUN/PSS profile changes the settings to LVDE50 bit, SV50[2:0] bit and LVDS bit of Status low-voltage detection configuration register (SYSC\_STSLVDCFGR). After the execution of the profile, the low-voltage detection status register reflects the settings. After power-on or a change in its settings, the low-voltage detection circuit is given a stabilization wait time (about 100μs). During the stabilization wait time, VDRDY50 bit of status low-voltage detection configuration becomes "0" and the detection result is masked. After the stabilization wait time has elapsed, VDRDY50 bit becomes "1" and supply voltage monitoring begins.

##### b) External Low-voltage Detection Interrupt

If an interrupt has been selected to be generated upon the detection of a low voltage, LVD50IF bit of System error interrupt factor register 0 (SYSC\_SYSERRIR0) becomes "1" when the external supply voltage drops below the set voltage. At this point, an interrupt is generated.

##### c) Clearing an Interrupt Factor of Low-voltage Detection

To clear the low-voltage detection interrupt factor, write "1" to LVD50ICLR bit of system error interrupt factor clear register 0 (SYSC\_SYSERRICLR0). This clears the low-voltage detection interrupt factor. Note that if "1" is written to LVD50ICLR bit while the supply voltage is still below the preset detection voltage, the low-voltage detection interrupt factor is not cleared, so the interrupt request is retained.

**d) External Low-voltage Detection Reset**

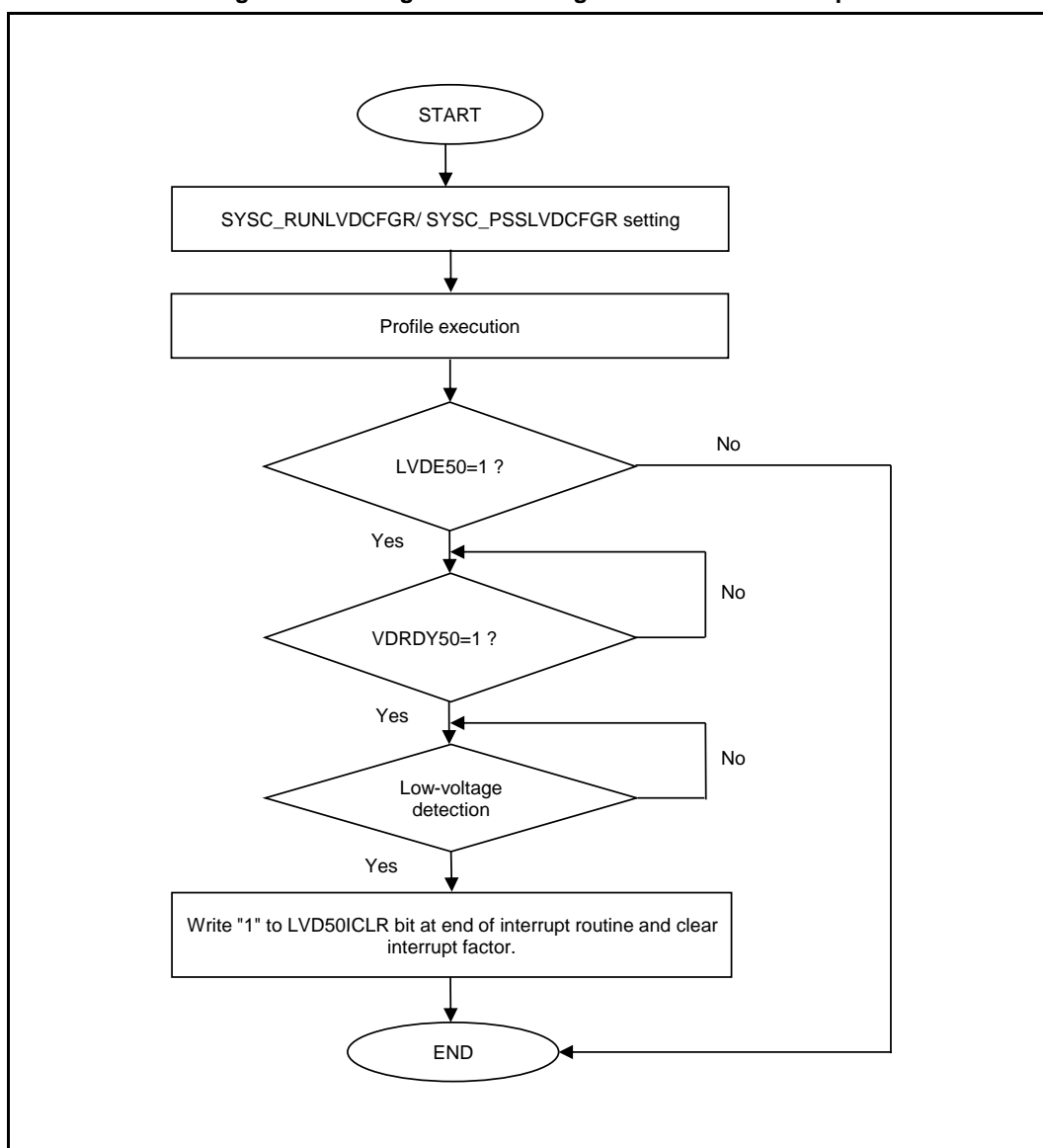
If a reset has been selected to be generated upon the detection of a low voltage, LVD50R bit of user reset factor register (SYSC\_RSTCAUSEUR) becomes "1" and a reset is generated when the external supply voltage drops below the set voltage. For details, see "CHAPTER: Reset".

## 4. Setting Procedure Examples

This section explains setting procedure examples of the low-voltage detection.

### (1) Setting the Low-voltage Detection for Interrupts

Figure 4-1 Setting the Low-voltage Detection for Interrupts

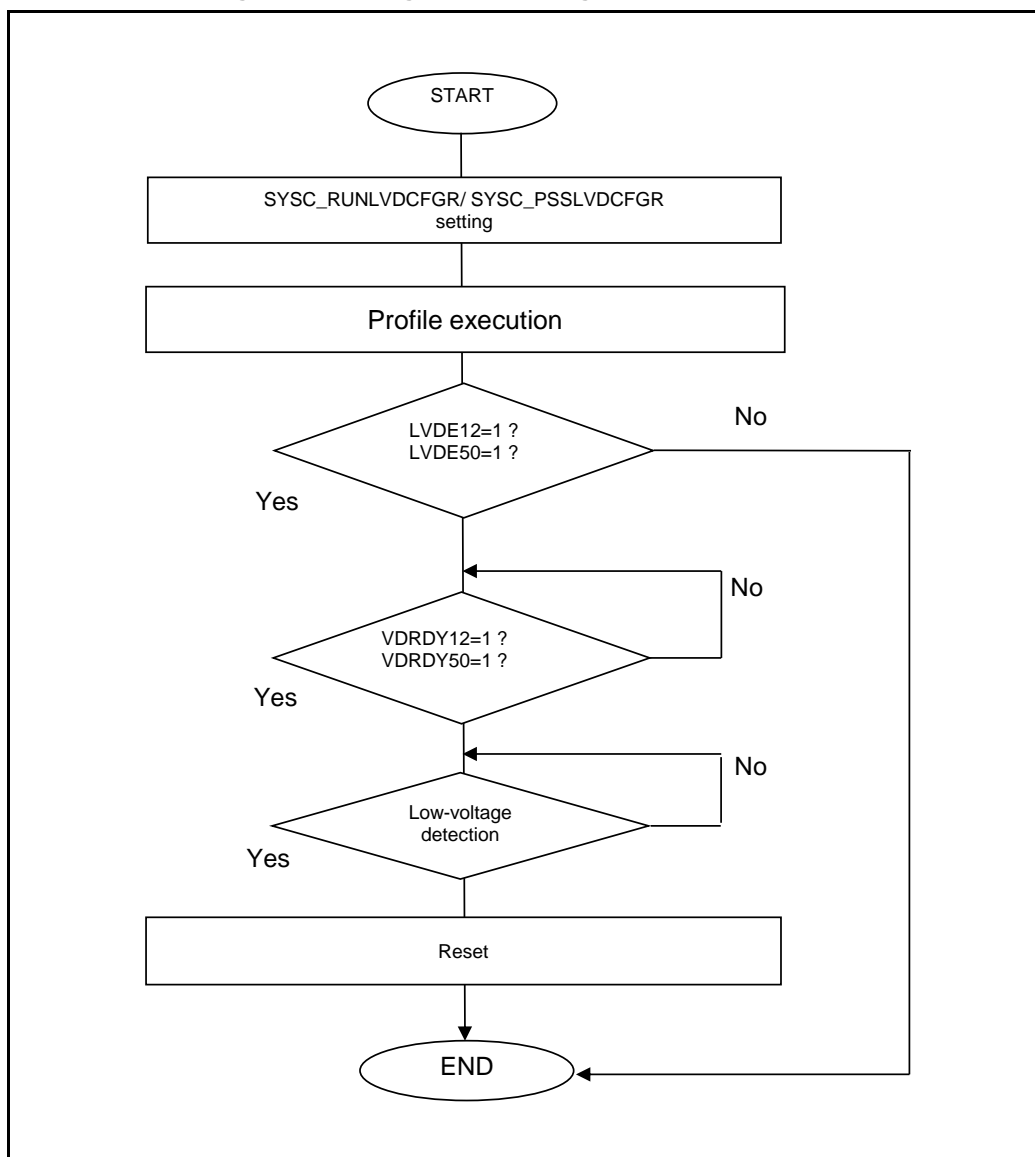


The execution of a profile applies the low-voltage detection settings made for the profile register. Low-voltage monitoring begins when the ready bit (VDRDY50) becomes "1". Then, an interrupt is generated upon the detection of a low voltage.

To return from the interrupt, clear the interrupt factor by writing "1" to LVD50ICLR bit at the end of the interrupt routine.

## (2) Setting the Low-voltage Detection for Resets

Figure 4-2 Setting the Low-voltage Detection for Resets



The execution of a profile applies the low-voltage detection settings made for the profile register.

Low-voltage monitoring begins when the ready bit (VDRDY50) becomes "1". Then, a reset is generated upon the detection of a low voltage.

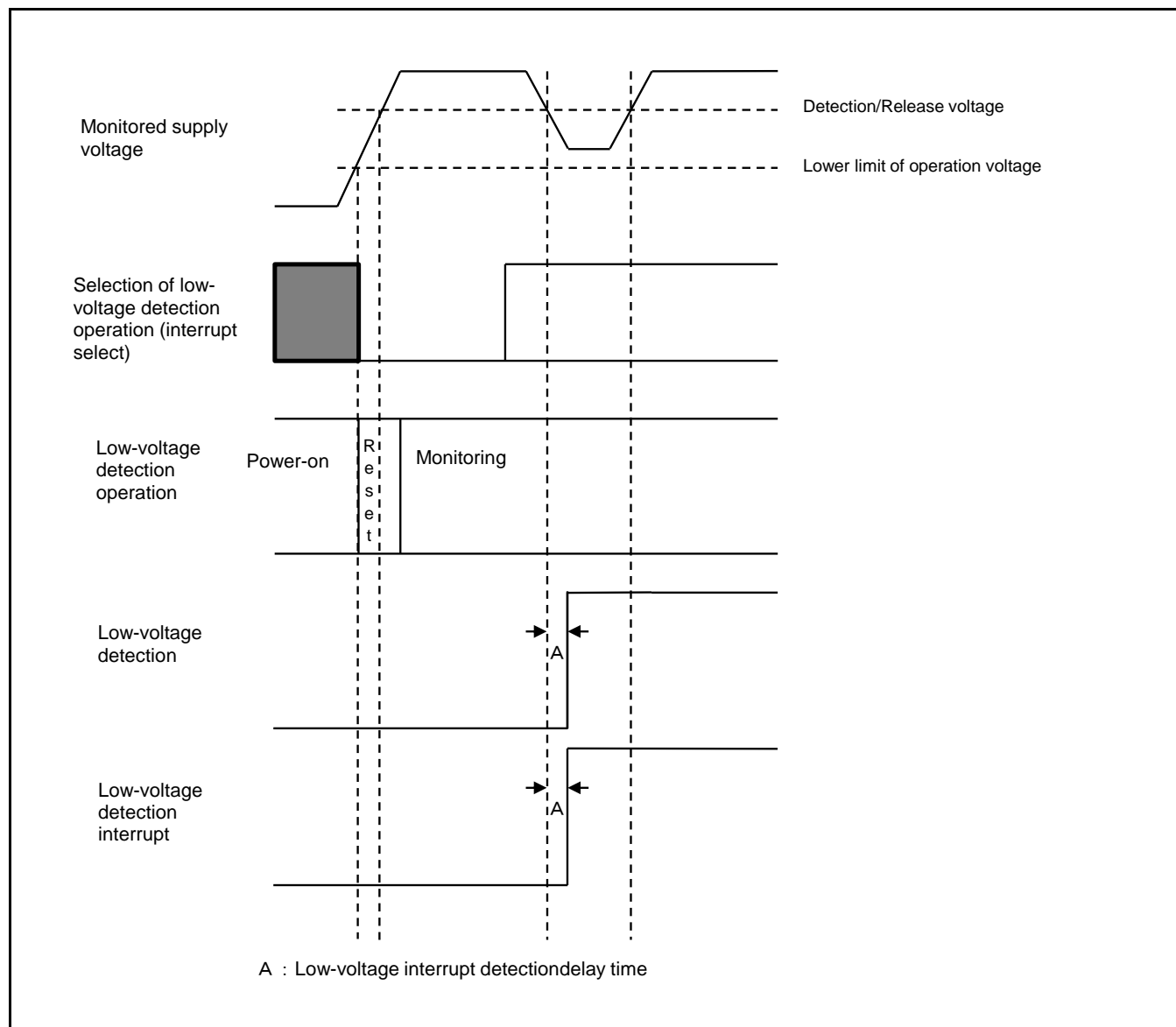
## 5. Operation Examples

This section explains operation examples of the low-voltage detection.

### (1) When the Detection Circuit for Interrupt Is Enabled (5.0V).

If low-voltage detection interrupts are enabled, the detection flag is set and an interrupt is generated upon the detection of a low voltage, as shown in the following timing chart.

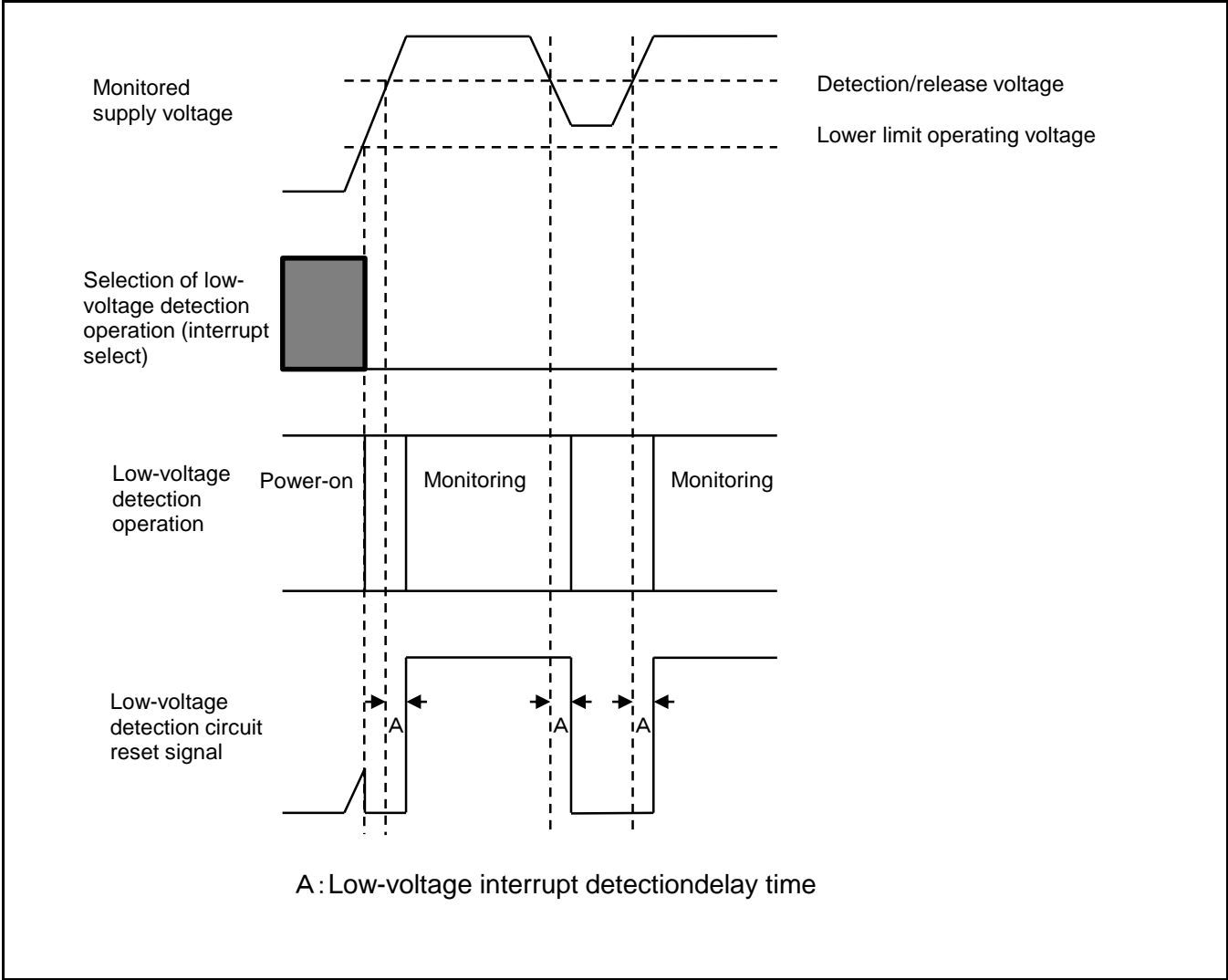
Figure 5-1 Timing Chart when Low-voltage Detection Interrupts are Enabled



**(2) When the Detection Circuit for Resets is Enabled (1.2V/5.0V)**

If the detection circuit for reset is enabled, a reset is generated upon the detection of a low voltage, as shown in the following timing chart. At the 1.2V, reset is always generated without operation select.

**Figure 5-2 Timing Chart when the Detection Circuit for Reset is Enabled**

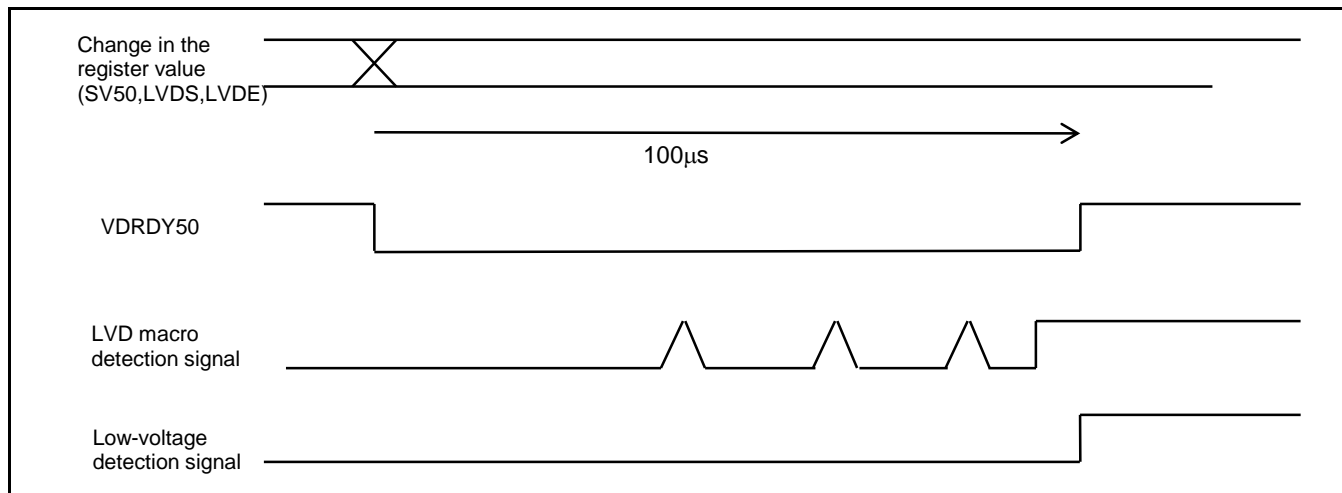




### (3) When the Register Setting Value is Changed (5.0V)

If the register setting value (SV50, LVDS, LVDE) has changed, the detection signal of the LVD macro is masked until VDRDY50 becomes "1", as shown below.

Figure 5-3 Timing Chart when the Register Setting Value is Changed



## 6. Precautions for Using

This section explains precautions on use of the low-voltage detection.

### (1) Low-voltage Detection (Internal Low-voltage Detection)

This section describes notes of low-voltage detection (internal low-voltage detection).

#### Hysteresis of Detection/Release

The release voltage becomes set value +0.05V so that detection/release may have the hysteresis of 0.05V. For example, when LVD:1.0V  $\pm$ 0.1V is set, the release voltage becomes 1.05V  $\pm$ 0.1V.

### (2) Low-voltage Detection (External Low-voltage Detection)

This section describes notes of low-voltage detection (external low-voltage detection).

#### a) Operation During the PSS

The low-voltage detection reset can continue to operate even in the PSS by settings. If a low-voltage is then detected in the PSS, the settings initialization reset is generated and the PSS is released. If an interrupt is selected, the system returns from the interrupt.

#### b) Hysteresis of Detection/Release

The release voltage becomes set value +0.125V so that detection/release may have the hysteresis of 0.125V. For example, when LVD:4.1V  $\pm$ 0.2V is set, the release voltage becomes 4.225V  $\pm$ 0.2V.

#### c) Masking of a Detection Result

Following any low-voltage setting change, the detection result is masked for about 100 $\mu$ s.







## CHAPTER 8: Clock Supervisor

This chapter shows the functions and operations of the clock supervisor.

---

1. Overview
2. Configuration
3. Operation
4. Setting Procedure Example
5. Registers
6. Precautions for Using



## 1. Overview

This section provides an overview of the clock supervisor.

The clock supervisor can detect the stopped clocks and occurrences of frequency-range abnormalities that are due to some kind of problem with a clock. Clocks of 2 systems are used: one clock becomes the monitoring target, and the other clock is a reference clock for measuring a certain time period.

The clock supervisor is equipped with the following supervisors. Each of them is independent and can be separately set to enabled or disabled.

- Main clock supervisor
- PLL clock supervisor
- Sub-system PLL clock supervisor

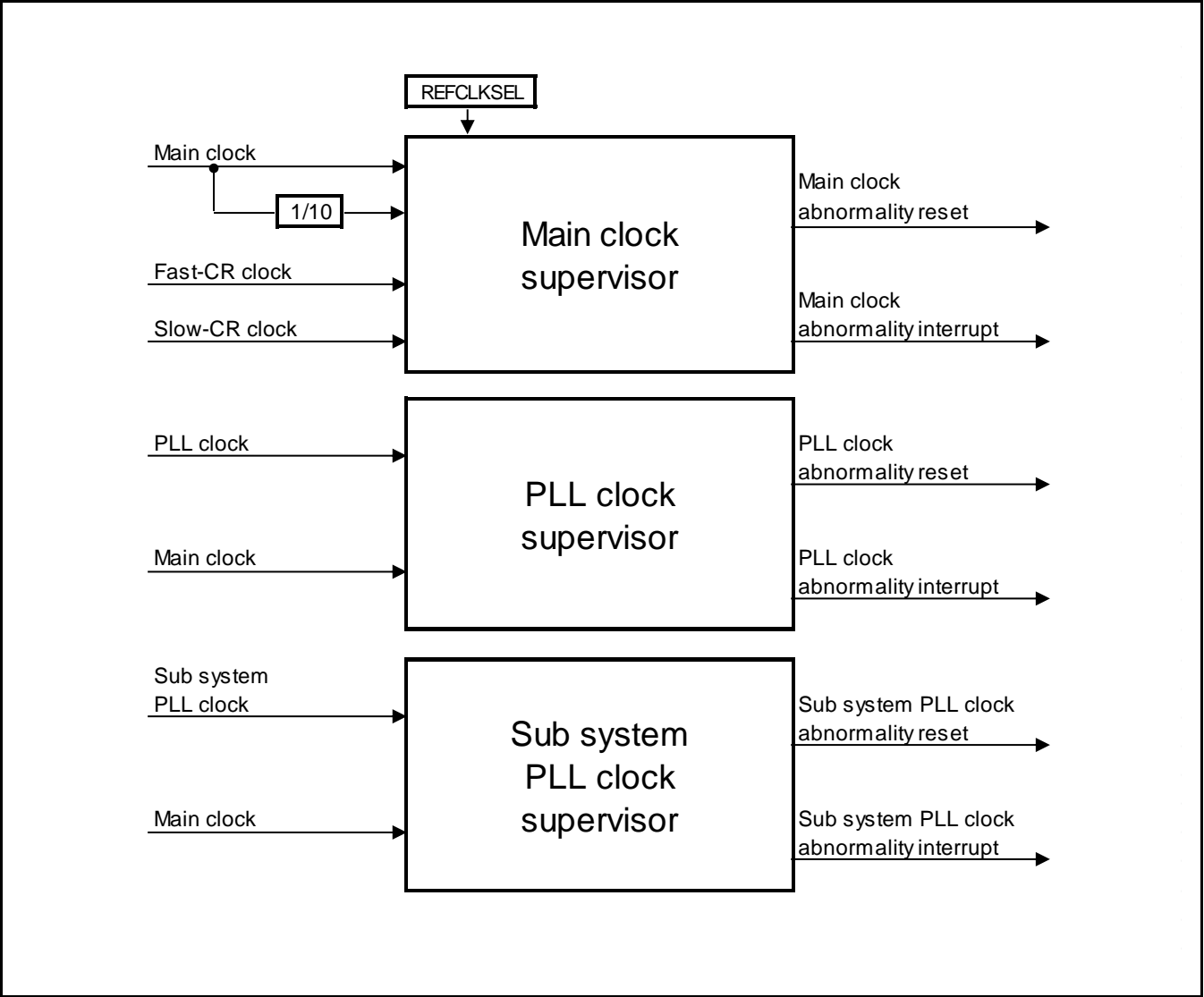
**Note:**

- *The MB9D560 series is equipped with a FlexRay/RDC PLL as the sub-system PLL clock.*

2. Configuration

The following figure is a block diagram of the clock supervisor.

Figure 2-1 Clock Supervisor Configuration





**a) Main Clock Supervisor**

- The main clock supervisor monitors the main clock. The reference clock used varies depending on the state. The fast-CR clock serves as the reference clock in the RUN. The fast-CR clock or slow-CR clock can be selected with the reference clock selection bit (REFCLKSEL) in the PSS. If the slow-CR clock is selected as the reference clock, the monitoring clock is the clock resulting from division of the main clock by 10.
- Upon detecting an abnormality of the main clock, the supervisor generates a reset or interrupt.

**b) PLL Clock Supervisor**

- The PLL clock supervisor monitors the PLL clock. The reference clock is the main clock.
- Upon detecting an abnormality of the PLL clock, the supervisor generates a reset or interrupt.

**c) Sub-system PLL Clock Supervisor**

- The sub-system PLL clock supervisor monitors the sub-system PLL clock. The reference clock is the main clock.
- Upon detecting an abnormality of the sub-system PLL clock, the supervisor generates a reset or interrupt.

**Note:**

- *The MB9D560 series is equipped with a FlexRay/RDC PLL as the sub-system PLL clock.*

Figure 2-2 Block Diagram of Main Clock Supervisor

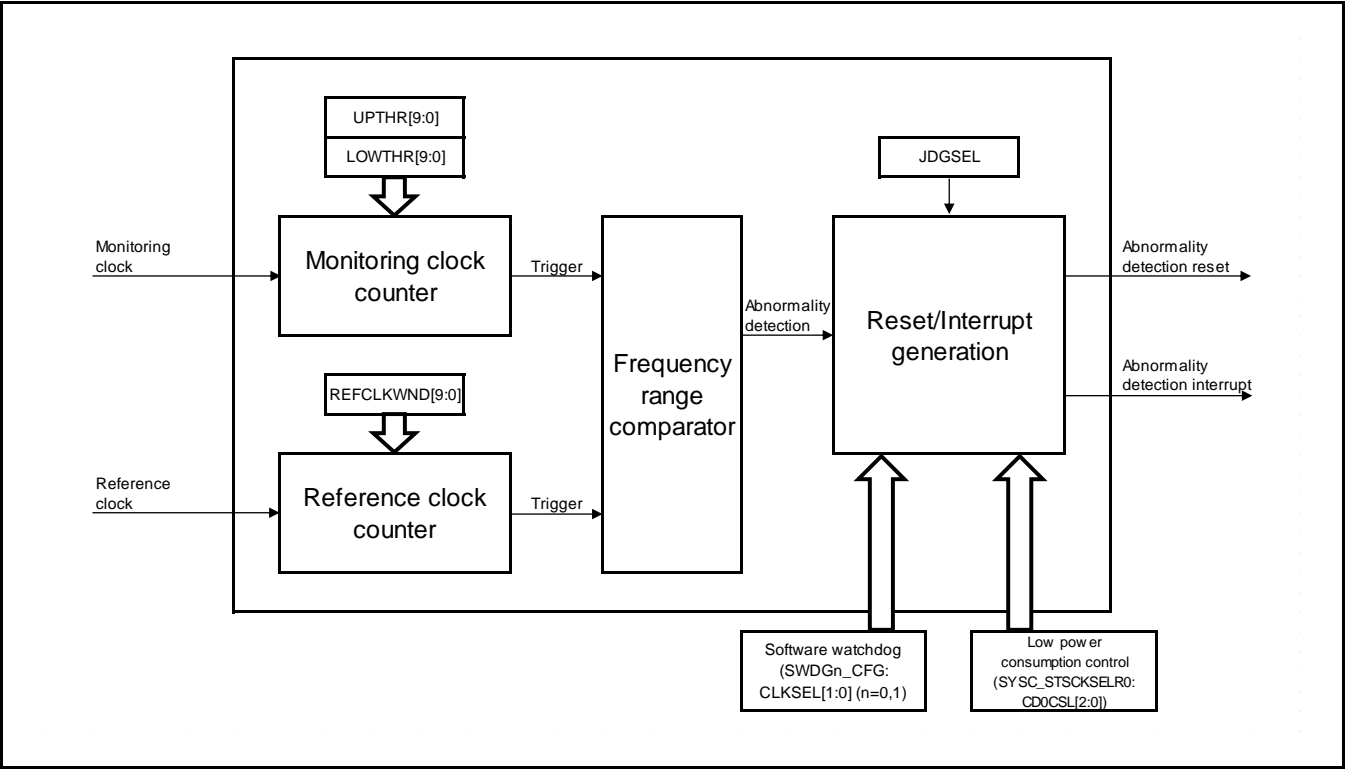


Figure 2-3 Block Diagram of PLL Clock Supervisor

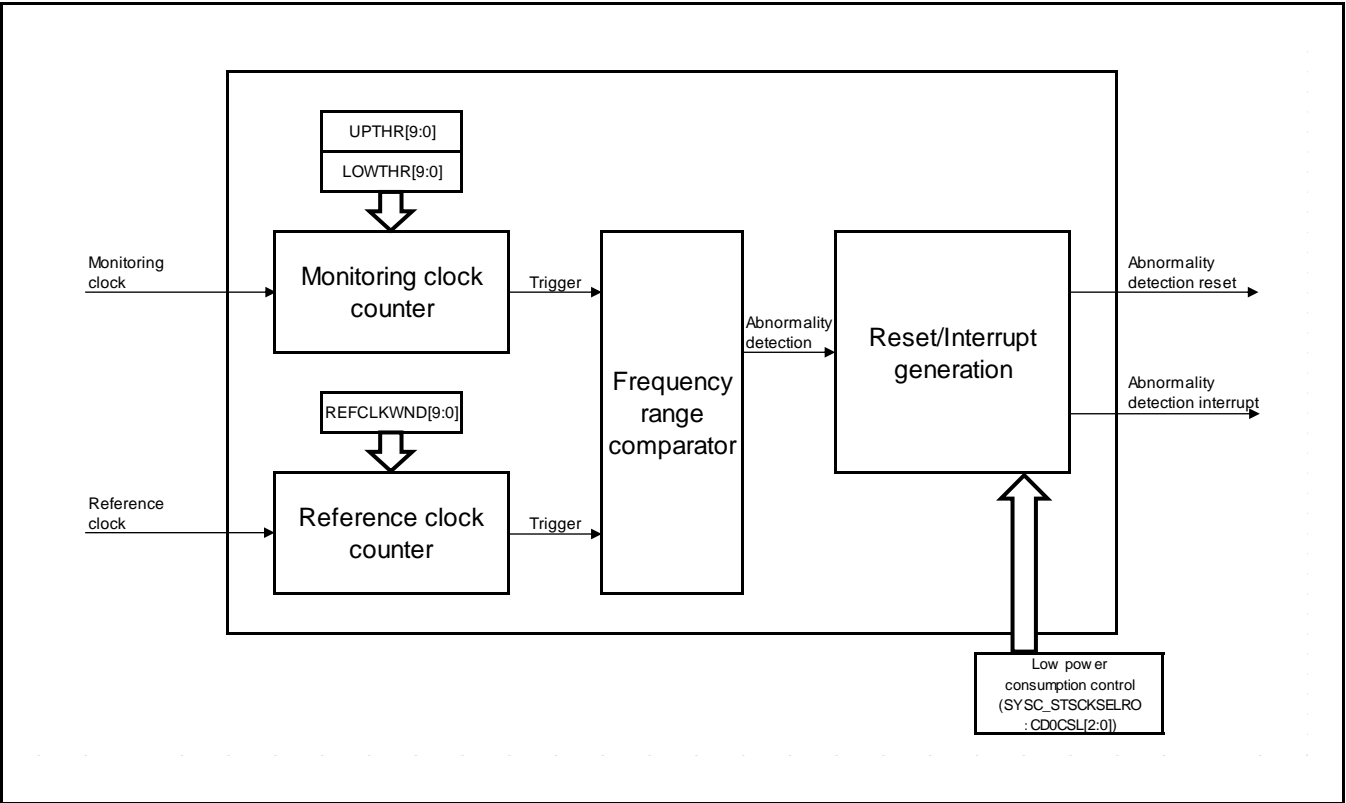
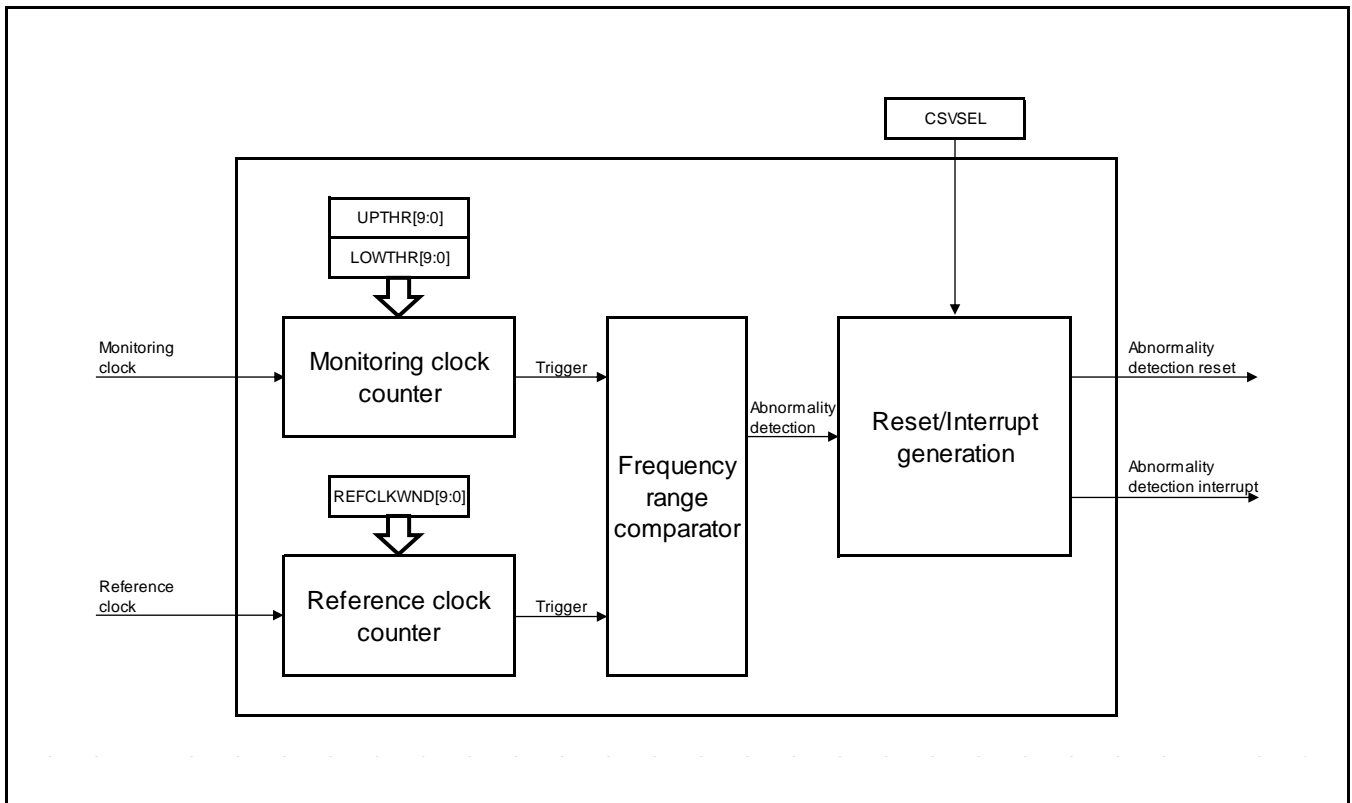


Figure 2-4 Block Diagram of Sub-system PLL Clock Supervisor



#### d) Monitoring Clock Counter

There are 2 10-bit down counters for counting with the monitoring clock. The counters load the values that are set in the upper-limit threshold value bits (UPTHR[9:0]) and lower-limit threshold value bits (LOWTHR[9:0]), and count down. When a counter value reaches "0", the counter sends a trigger to the frequency range comparator to make a comparison. The counter reloads the register values and restarts counting down when the frequency comparator makes the comparison.

#### e) Reference Clock Counter

There is a 10-bit down counter for counting with the reference clock. The counter loads the value that is set in the reference clock count period bits (REFCLKWND[9:0]), and counts down. When the counter value reaches "0", the counter sends a trigger to the frequency range comparator to make a comparison. The counter reloads the register value and restarts counting down when the frequency range comparator makes the comparison.

#### f) Frequency Range Comparator

The frequency range comparator compares frequency ranges according to triggers from the monitoring clock counters and reference clock counter. The lower-limit threshold value bits (LOWTHR[9:0]) and upper-limit threshold value bits (UPTHR[9:0]) of the monitoring clock counters set a frequency range. The comparator judges a range outside this frequency range to be an abnormality.

#### g) Reset/Interrupt generator

The frequency range comparator compares frequency ranges according to triggers from the monitoring clock counters and reference clock counter. The lower-limit threshold value bits (LOWTHR[9:0]) and upper-limit threshold value bits (UPTHR[9:0]) of the monitoring clock counters set a frequency range. The comparator judges a range outside this frequency range to be an abnormality.

For details, see "CHAPTER: FlexRay/RDC Dedicated Clock".

### 3. Operation

This section explains the operation of the clock supervisor.

The clock supervisor has monitoring clock counters for counting with the monitoring clock and a reference clock counter for counting with the reference clock. The monitoring clock counters consist of 2 10-bit down counters. They load the values that are set in the upper-limit threshold value bits (UPTHR[9:0]) and lower-limit threshold value bits (LOWTHR[9:0]), and count down. The reference clock counter is a 10-bit down counter. It loads the value that is set in the reference clock count period bits (REFCLKWND[9:0]), and counts down. When the count value of any of the counters reaches "0", the counter sends a trigger signal to the frequency range comparator. The upper-limit threshold value bits (UPTHR[9:0]) and lower-limit threshold value bits (LOWTHR[9:0]) set a frequency range. If the frequency range of the monitoring clock is outside that frequency range, the frequency range comparator detects an abnormality through a trigger signal.

The counter reloads a register value and restarts counting down when the frequency range comparator compares the frequency ranges.

The reset/interrupt generator generates a reset or interrupt when the frequency range comparator detects an abnormality.

#### a) Main Clock Supervisor

- If the fast-CR clock is selected as the reference clock, settings are made with main clock supervisor configuration register 00 (SYSC\_CSMOCFGR00) and main clock supervisor configuration register 01 (SYSC\_CSMOCFGR01). If the slow-CR clock is selected as the reference clock, settings are made with main clock supervisor configuration register 10 (SYSC\_CSMOCFGR10) and main clock supervisor configuration register 11 (SYSC\_CSMOCFGR11).
- The main clock supervisor determines whether to generate a reset or interrupt according to the settings of the clock domain 0 clock selection bits (CD0CSL[2:0]) in status clock selection register 0 (SYSC\_STCKSELR0) and the clock selection bits (CLKSEL[1:0]) in the software watchdog configuration register (SWDGN\_CFG(n=0 to 1)), when an abnormality is detected.
- The judgment selection bit (JDGSEL) enables selection of criteria for judging whether to generate a reset or interrupt.
- Upon detecting an abnormality of the main clock and generating a reset, the main clock supervisor can confirm it with the main clock supervisor reset detection bit (CSVMOR) in the user reset factor register (SYSC\_RSTCAUSEUR).
- Upon detecting an abnormality of the main clock and generating an interrupt, the main clock supervisor sets the MOMIF bit in system error interrupt factor register 0 (SYSC\_SYSEIRR0).

#### b) PLL Clock Supervisor

- PLL clock supervisor configuration register 0 (SYSC\_CSVPLLCFGR0) and PLL clock supervisor configuration register 1 (SYSC\_CSVPLLCFGR1) are used for settings.
- The setting of the clock domain 0 clock selection bits (CD0CSL[2:0]) in status clock selection register 0 (SYSC\_STCKSELR0) when an abnormality is detected determines whether to generate a reset or interrupt.
- If the PLL clock is selected as the clock of clock domain 0, a reset is generated. If not selected, an interrupt is generated.
- Upon detecting an abnormality of the PLL clock and generating a reset, the PLL clock supervisor can confirm it with the PLL clock supervisor reset detection bit (CSVPR) in the user reset factor register (SYSC\_RSTCAUSEUR).
- Upon detecting an abnormality of the PLL clock and generating an interrupt, the PLL clock supervisor sets the PMIF bit in system error interrupt factor register 0 (SYSC\_SYSEIRR0).

#### c) Sub-system PLL Clock Supervisor

- Sub-system PLL clock supervisor configuration register 0 (SYSC\_CSVSSCFGR0) and sub-system PLL clock supervisor configuration register 1 (SYSC\_CSVSSCFGR1) are used for settings.
- The supervisor can select whether to generate a reset or interrupt at the abnormal state detection.





For the detail, see "CHAPTER: FlexRay/RDC Dedicated Clock".

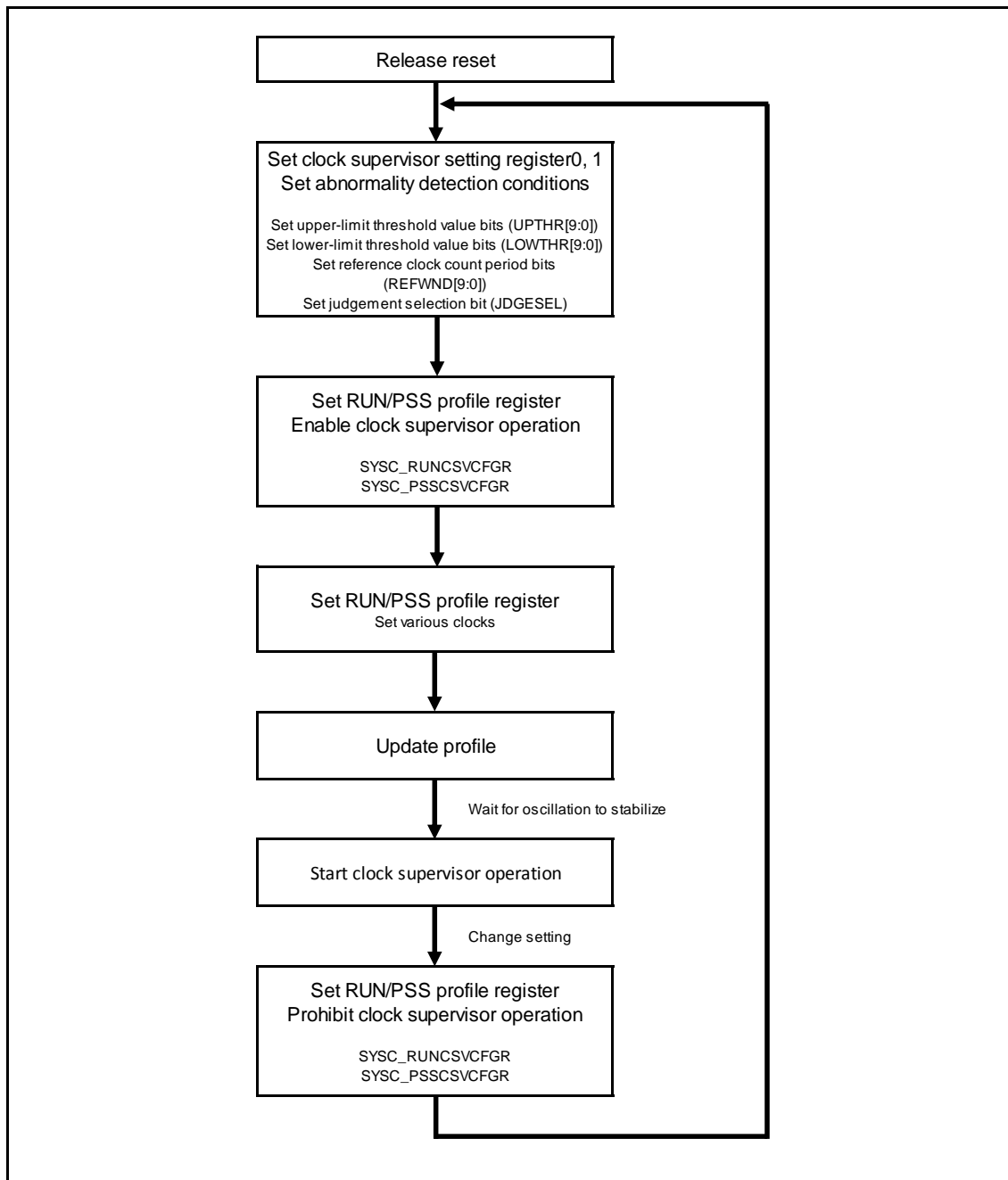
- Upon detecting an abnormality of the sub-system PLL clock and generating a reset, the sub-system PLL clock supervisor can confirm it with the sub-system PLL clock supervisor reset detection bit (CSVSSR) in the extended CSV reset factor register (SYSC\_EXCSVRSTCAUSEUR).
- Upon detecting an abnormality of the sub-system PLL clock and generating an interrupt, the sub-system PLL clock supervisor sets the SSPMIF bit in system error interrupt factor register 0 (SYSC\_SYSERRIR0).

## 4. Setting Procedure Example

This section shows an example of the clock supervisor setting procedure.

The following figure shows an example of the clock supervisor setting procedure.

Figure 4-1 Example of Clock Supervisor Setting Procedure



**Note:**

- For details on clock settings, see "CHAPTER: Clock System" and "CHAPTER: Low-power Consumption".



- After the hardware reset is released, all clock supervisors are in the operation disabled state.
- Profile settings are used to enable/disable the main clock supervisor and PLL clock supervisor. Any violation in the profile settings is judged to be a profile error.
- For details on profile errors, see "CHAPTER: Low-power Consumption".
- The clock supervisor does not operate until monitoring clock and reference clock oscillation has stabilized.
- The clock supervisor can enter the clock abnormality detection state by gating the monitoring clock for self-testing.

#### a) Example of Main Clock Supervisor Settings

##### Conditions

Main clock frequency: 8 [MHz]

Fast-CR clock frequency: 6 to 10 [MHz] (typ: 8 MHz)

Acceptable main clock frequency range: 4 to 20 [MHz]

##### Calculation formulas

Lower-limit threshold value =  $4/10 \times$  reference clock count value

Upper-limit threshold value =  $20/6 \times$  reference clock count value

##### Register setting values

Reference clock count period = 60

Lower-limit threshold value = 24

Upper-limit threshold value = 200

#### b) Example of PLL Clock Supervisor Settings

##### Conditions

PLL clock frequency: 200 MHz

Main clock frequency: 8 [MHz]

Acceptable PLL clock frequency range: 80 MHz to 500 MHz

##### Calculation formulas

Lower-limit threshold value =  $80/8 \times$  reference clock count value

Upper-limit threshold value =  $500/8 \times$  reference clock count value

##### Register setting values

Reference clock count period = 12

Lower-limit threshold value = 120

Upper-limit threshold value = 750

## 5. Registers

This section lists the clock supervisor registers.

The clock supervisor registers are as follows:

- RUN profile registers
- PSS profile registers
- APPLIED profile registers
- Status profile registers
- CSV configuration register group

For details on the RUN profile registers, PSS profile registers, APPLIED profile registers and status profile registers, see "CHAPTER: Low-power Consumption".

**Table 5-1 List of Clock Supervisor Registers (CSV Configuration Registers)**

Abbreviated Register Name	Register Name	See
SYSC_CSMOCFGR00	Main Clock Supervisor Configuration Register 00	5.1
SYSC_CSMOCFGR01	Main Clock Supervisor Configuration Register 01	5.2
SYSC_CSMOCFGR10	Main Clock Supervisor Configuration Register 10	5.3
SYSC_CSMOCFGR11	Main Clock Supervisor Configuration Register 11	5.4
SYSC_CSVPLLCFGR0	PLL Clock Supervisor Configuration Register 0	5.5
SYSC_CSVPLLCFGR1	PLL Clock Supervisor Configuration Register 1	5.6
SYSC_CSVSSCFGR0	Sub-system PLL Clock Supervisor Configuration Register 0	5.7
SYSC_CSVSSCFGR1	Sub-system PLL Clock Supervisor Configuration Register 1	5.8
SYSC_CSVOUTER	Clock Supervisor Output Enable Register	5.9
SYSC_CSVTESTR	Clock Supervisor Test Register	5.10



## 5.1. Main Clock Supervisor Configuration Register 00 (SYSC\_CSMOCFGR00)

Main clock supervisor configuration register 00 (SYSC\_CSMOCFGR00) sets the upper-limit threshold value and lower-limit threshold value of a frequency range when the fast-CR clock operates as a reference clock.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved						UPTHR[9:8]	
ACCESS_TYPE	R0,WX						R/W	
PROT_TYPE	WPS							
INITIAL_VALUE	000000						00	

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	UPTHR[7:0]							
ACCESS_TYPE	R/W							
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved						LOWTHR[9:8]	
ACCESS_TYPE	R0,WX						R/W	
PROT_TYPE	WPS							
INITIAL_VALUE	000000						00	

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	LOWTHR[7:0]							
ACCESS_TYPE	R/W							
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							

**[bit31:26] Reserved: Reserved bits**

**[bit25:16] UPTHR[9:0]: Upper-limit threshold value bits**

These bits set the clock upper-limit threshold value for comparison with the monitoring clock count value. If the monitoring clock counter value exceeds the upper-limit threshold value, the state is judged as abnormal.

**[bit15:10] Reserved: Reserved bits**

**[bit9:0] LOWTHR[9:0]: Lower-limit threshold value bits**

These bits set the clock lower-limit threshold value for comparison with the monitoring clock count value. If the monitoring clock counter value falls below the lower-limit threshold value, the state is judged as abnormal.

**Notes:**

- *This register must not be modified during operation of the main clock supervisor. The bus error response will occur if write operation is done.*
- *For the lower-limit threshold value bits (LOWTHR), set a value larger than that obtained from dividing the reference clock cycle by the monitoring clock cycle.*
- *Set the upper-limit threshold value bit (UPTHR) value to be greater than the lower-limit threshold value bit (LOWTHR) value.*



## 5.2. Main Clock Supervisor Configuration Register 01 (SYSC\_CSMOCFGR01)

Main clock supervisor configuration register 01 (SYSC\_CSMOCFGR01) can set a reference clock count period when the fast-CR clock operates as a reference clock. It can also select whether to generate a reset or interrupt at the abnormal state detection.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved							JDGSEL
ACCESS_TYPE	R0,WX							R/W
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved						REFCLKWND[9:8]	
ACCESS_TYPE	R0,WX						R/W	
PROT_TYPE	WPS							
INITIAL_VALUE	000000						00	

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	REFCLKWND[7:0]							
ACCESS_TYPE	R/W							
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							

**[bit31:17] Reserved: Reserved bits**

### **[bit16] JDGSEL: Judgment selection bit**

This bit selects criteria for judging whether to generate a reset or interrupt at the abnormal state detection.

Value	Description
0	Generates a reset if the main clock is selected as clock domain 0 or the software watchdog timer. If not selected for either, an interrupt is generated.
1	Generates a reset if the main clock is selected as clock domain 0. If not selected, an interrupt is generated.

**[bit15:10] Reserved: Reserved bits**

**[bit9:0] REFCLKWND[9:0]: Reference clock count period bits**

These bits set the reference clock count number used to trigger a comparison by the frequency range comparator.

**Note:**

- *This register must not be modified during operation of the main clock supervisor. The bus error response will occur if write operation is done.*





### 5.3. Main Clock Supervisor Configuration Register 10 (SYSC\_CSMOCFGR10)

Main clock supervisor configuration register 10 (SYSC\_CSMOCFGR10) sets the upper-limit threshold value and lower-limit threshold value of a frequency range when the slow-CR clock operates as a reference clock.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved						UPTHR[9:8]	
ACCESS_TYPE	R0,WX						R/W	
PROT_TYPE	WPS							
INITIAL_VALUE	000000						00	

BITS_OFFSET	23	22	21	20	19	18	17	16
BITS_NAME	UPTHR[7:0]							
ACCESS_TYPE	R/W							
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved						LOWTHR[9:8]	
ACCESS_TYPE	R0,WX						R/W	
PROT_TYPE	WPS							
INITIAL_VALUE	000000						00	

BITS_OFFSET	7	6	5	4	3	2	1	0
BITS_NAME	LOWTHR[7:0]							
ACCESS_TYPE	R/W							
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							

**[bit31:26] Reserved: Reserved bits**

**[bit25:16] UPTHR[11:0]: Upper-limit threshold value bits**

These bits set the clock upper-limit threshold value for comparison with the monitoring clock count value. If the monitoring clock counter value exceeds the upper-limit threshold value, the state is judged as abnormal.

**[bit15:10] Reserved: Reserved bits**

**[bit9:0] LOWTHR[9:0]: Lower-limit threshold value bits**

These bits set the clock lower-limit threshold value for comparison with the monitoring clock count value. If the monitoring clock counter value falls below the lower-limit threshold value, the state is judged as abnormal.

**Notes:**

- *This register must not be modified during operation of the main clock supervisor. The bus error response will occur if write operation is done.*
- *For the lower-limit threshold value bits (LOWTHR), set a value larger than that obtained from dividing the reference clock cycle by the monitoring clock cycle.*
- *Set the upper-limit threshold value bit (UPTHR) value to be greater than the lower-limit threshold value bit (LOWTHR) value.*
- *If the slow-CR clock is selected as a reference clock, the monitoring clock is the clock resulting from division of the main clock by 10.*



## 5.4. Main Clock Supervisor Configuration Register 11 (SYSC\_CSVMOCFGR11)

Main clock supervisor configuration register 11 (SYSC\_CSVMOCFGR11) can select a reference clock count in the PSS. It can also set a reference clock count period when the slow-CR clock operates as a reference clock.

BITS	31	30	29	28	27	26	25	24
BIT_OFFSET								
BIT_NAME	Reserved							REFCLKSEL
ACCESS_TYPE	R0,WX							R/W
PROT_TYPE	WPS							
INITIAL_VALUE	0000000							0

BITS	23	22	21	20	19	18	17	16
BIT_OFFSET								
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							

BITS	15	14	13	12	11	10	9	8
BIT_OFFSET								
BIT_NAME	Reserved						REFCLKWND[9:8]	
ACCESS_TYPE	R0,WX						R/W	
PROT_TYPE	WPS							
INITIAL_VALUE	000000						00	

BITS	7	6	5	4	3	2	1	0
BIT_OFFSET								
BIT_NAME	REFCLKWND[7:0]							
ACCESS_TYPE	R/W							
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							

[bit31:25] Reserved: Reserved bits

### [bit24] REFCLKSEL: Reference clock selection bit

This bit selects a reference clock in the PSS.

Value	Description
0	Fast-CR clock
1	Slow-CR clock

[bit23:10] Reserved: Reserved bits

**[bit9:0] REFCLKWND[9:0]: Reference clock count period bits**

These bits set the reference clock count number used to trigger a comparison by the frequency range comparator.

**Note:**

- *This register must not be modified during operation of the main clock supervisor. The bus error response will occur if write operation is done.*



## 5.5. PLL Clock Supervisor Configuration Register 0 (SYSC\_CSVPLLCFGR0)

PLL clock supervisor configuration register 0 (SYSC\_CSVPLLCFGR0) sets the upper-limit threshold value and lower-limit threshold value of a frequency range.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved						UPTHR[9:8]	
ACCESS_TYPE	R0,WX						R/W	
PROT_TYPE	WPS							
INITIAL_VALUE	000000						00	

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	UPTHR[7:0]							
ACCESS_TYPE	R/W							
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved						LOWTHR[9:8]	
ACCESS_TYPE	R0,WX						R/W	
PROT_TYPE	WPS							
INITIAL_VALUE	000000						00	

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	LOWTHR[7:0]							
ACCESS_TYPE	R/W							
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							

**[bit31:26] Reserved: Reserved bits**

**[bit25:16] UPTHR[9:0]: Upper-limit threshold value bits**

These bits set the clock upper-limit threshold value for comparison with the monitoring clock count value. If the monitoring clock counter value exceeds the upper-limit threshold value, the state is judged as abnormal.

**[bit15:10] Reserved: Reserved bits**

**[bit9:0] LOWTHR[9:0]: Lower-limit threshold value bits**

These bits set the clock lower-limit threshold value for comparison with the monitoring clock count value. If the monitoring clock counter value falls below the lower-limit threshold value, the state is judged as abnormal.

**Notes:**

- *This register must not be modified during operation of the PLL clock supervisor. The bus error response will occur if write operation is done.*
- *For the lower-limit threshold value bits (LOWTHR), set a value larger than that obtained from dividing the reference clock cycle by the monitoring clock cycle.*
- *Set the upper-limit threshold value bit (UPTHR) value to be greater than the lower-limit threshold value bit (LOWTHR) value.*



## 5.6. PLL Clock Supervisor Configuration Register 1 (SYSC\_CSVPLLCFGR1)

PLL clock supervisor configuration register 1 (SYSC\_CSVPLLCFGR1) sets a reference clock count period.

BIT_OFFSET	31-16
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	WPS
INITIAL_VALUE	00000000_00000000

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved						REFCLKWND[9:8]	
ACCESS_TYPE	R0,WX						R/W	
PROT_TYPE	WPS							
INITIAL_VALUE	000000						00	

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	REFCLKWND[7:0]							
ACCESS_TYPE	R/W							
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							

**[bit31:10] Reserved: Reserved bits**

**[bit9:0] REFCLKWND[9:0]: Reference clock count period bits**

These bits set the reference clock count number used to trigger a comparison by the frequency range comparator.

**Note:**

- This register must not be modified during operation of the PLL clock supervisor. The bus error response will occur if write operation is done.

## 5.7. Sub-system PLL Clock Supervisor Configuration Register 0 (SYSC\_CSVSSCFGR0)

Sub-system PLL clock supervisor configuration register 0 (SYSC\_CSVSSCFGR0) sets the upper-limit threshold value and lower-limit threshold value of a frequency range.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved						UPTHR[9:8]	
ACCESS_TYPE	R0,WX						R/W	
PROT_TYPE	WPS							
INITIAL_VALUE	000000						00	

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	UPTHR[7:0]							
ACCESS_TYPE	R/W							
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved						LOWTHR[9:8]	
ACCESS_TYPE	R0,WX						R/W	
PROT_TYPE	WPS							
INITIAL_VALUE	000000						00	

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	LOWTHR[7:0]							
ACCESS_TYPE	R/W							
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							

**[bit31:26] Reserved: Reserved bits**

**[bit25:16] UPTHR[9:0]: Upper-limit threshold value bits**

These bits set the clock upper-limit threshold value for comparison with the monitoring clock count value. If the monitoring clock counter value exceeds the upper-limit threshold value, the state is judged as abnormal.

**[bit15:10] Reserved: Reserved bits**





**[bit9:0] LOWTHR[9:0]: Lower-limit threshold value bits**

These bits set the clock lower-limit threshold value for comparison with the monitoring clock count value. If the monitoring clock counter value falls below the lower-limit threshold value, the state is judged as abnormal.

**Notes:**

- *The MB9D560 series is equipped with a FlexRay/RDC PLL as the sub-system PLL clock.*
- *This register must not be modified during operation of the sub-system PLL clock supervisor. The bus error response will occur if write operation is done.*
- *For the lower-limit threshold value bits (LOWTHR), set a value larger than that obtained from dividing the reference clock cycle by the monitoring clock cycle.*
- *Set the upper-limit threshold value bit (UPTHR) value to be greater than the lower-limit threshold value bit (LOWTHR) value.*

## 5.8. Sub-system PLL Clock Supervisor Configuration Register 1 (SYSC\_CSVSSCFGR1)

Sub-system PLL clock supervisor configuration register 1 (SYSC\_CSVSSCFGR1) sets a reference clock count period.

BIT_OFFSET	31-16
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	WPS
INITIAL_VALUE	00000000_00000000

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved						REFCLKWND[9:8]	
ACCESS_TYPE	R0,WX						R/W	
PROT_TYPE	WPS							
INITIAL_VALUE	000000						00	

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	REFCLKWND[7:0]							
ACCESS_TYPE	R/W							
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							

**[bit31:10] Reserved: Reserved bits**

**[bit9:0] REFCLKWND[9:0]: Reference clock count period bits**

These bits set the reference clock cycle used to trigger a comparison by the frequency range comparator.

**Notes:**

- The MB9D560 series is equipped with a FlexRay/RDC PLL as the sub-system PLL clock.
- This register must not be modified during operation of the sub-system PLL clock supervisor. The bus error response will occur if write operation is done.



## 5.9. Clock Supervisor Output Enable Register (SYSC\_CSVOUTER)

The clock supervisor output enable register (SYSC\_CSVOUTER) enables output of the abnormality detection bit to a port.

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	WPS
INITIAL_VALUE	00000000 00000000 00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved							OUTEN
ACCESS_TYPE	R0,WX							R/W
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							0

**[bit31:1] Reserved: Reserved bits**

### **[bit0] OUTEN: Clock supervisor output enable bit**

This bit is the enable bit for outputting the abnormality detection bit to a port.

Value	Description
0	Disables output of the abnormality detection bit from a port
1	Enables output of the abnormality detection bit from a port

For output ports, see the list of pin functions in "CHAPTER: Overview" and "CHAPTER: I/O Port".

**Note:**

- This bit is cleared to "0" by power-on reset, internal power supply low-voltage detection reset, and simultaneous input of "L" to the NMIX pin + RSTX pin. It is not affected by other types of reset.

## 5.10. Clock Supervisor Test Register (SYSC\_CSVTESTR)

The clock supervisor test register (SYSC\_CSVTESTR) can perform function tests using input clock gating.

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	WPS
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved			SSCLK GATE	Reserved	PLLCLK GATE	Reserved	MOCLK GATE
ACCESS_TYPE	R0,WX			R/W	R0,WX	R/W	R0,WX	R/W
PROT_TYPE	WPS							
INITIAL_VALUE	000			0	0	0	0	0

**[bit31:5] Reserved: Reserved bits**

### **[bit4] SSCLKGATE: Sub-system PLL clock supervisor test bit**

This bit can be used for function tests using input clock gating.

Value	Description
0	Normal operation
1	Input clock gating

**Note:**

- The MB9D560 series is equipped with a FlexRay/RDC PLL as the sub-system PLL clock.

**[bit3] Reserved: Reserved bit**

### **[bit2] PLLCLKGATE: PLL clock supervisor test bit**

This bit can be used for function tests using input clock gating.

Value	Description
0	Normal operation
1	Input clock gating

**[bit1] Reserved: Reserved bit**

### **[bit0] MOCLKGATE: Main clock supervisor test bit**

This bit can be used for function tests using input clock gating.

Value	Description
0	Normal operation
1	Input clock gating



## 6. Precautions for Using

This section explains precautions for using the clock supervisor.

- Profile settings are used to enable/disable operations of the clock supervisor. For details on the profile settings, see "CHAPTER: Low-power Consumption".
- To operate the clock supervisor, enable oscillation of the monitoring clock and reference clock.
- To stop the clock supervisor, simultaneously stop oscillation of the monitoring clock.
- If the slow-CR clock is selected as the main clock supervisor reference clock, the monitoring clock is the clock resulting from division of the main clock by 10.
- After a reset is generated by the detection of a clock abnormality, the main clock supervisor reference clock returns to the fast-CR clock. The clock of the detected abnormality must not be selected again.
- To protect hardware, clock supervisor configuration registers 0 and 1 must not be modified during operation of the clock supervisor. To change either of them, do so only after disabling operation of the clock supervisor. The bus error response will occur if write operation is done.



## CHAPTER 9: Source Clock Timer

This chapter explains the functions and operations of the source clock timer.

---

1. Overview
2. Configuration
3. Operation
4. Setting Procedure Example
5. Registers
6. Precautions for Using



## 1. Overview

This section provides an overview of the source clock timer.

The source clock timer is a timer for gating the clock output until the end of the clock oscillation stabilization wait time. After the clock oscillation stabilizes, this timer can be used as an ordinary timer. The 3 source clocks are the fast-CR clock, slow-CR clock, and main clock, and they are prepared separately.

### a) Fast-CR Clock Timer

The fast-CR clock timer counts the internal fast-CR clock. This timer is used to count time to wait for the fast-CR clock oscillation to stabilize.

The fast-CR clock timer has an oscillation stabilization wait time of 0.24 ms.

### b) Slow-CR Clock Timer

The slow-CR clock timer counts the internal slow-CR clock. This timer is used to count time to wait for the slow-CR clock oscillation to stabilize.

The slow-CR clock timer has an oscillation stabilization wait time of 0.64 ms.

### c) Main Clock Timer

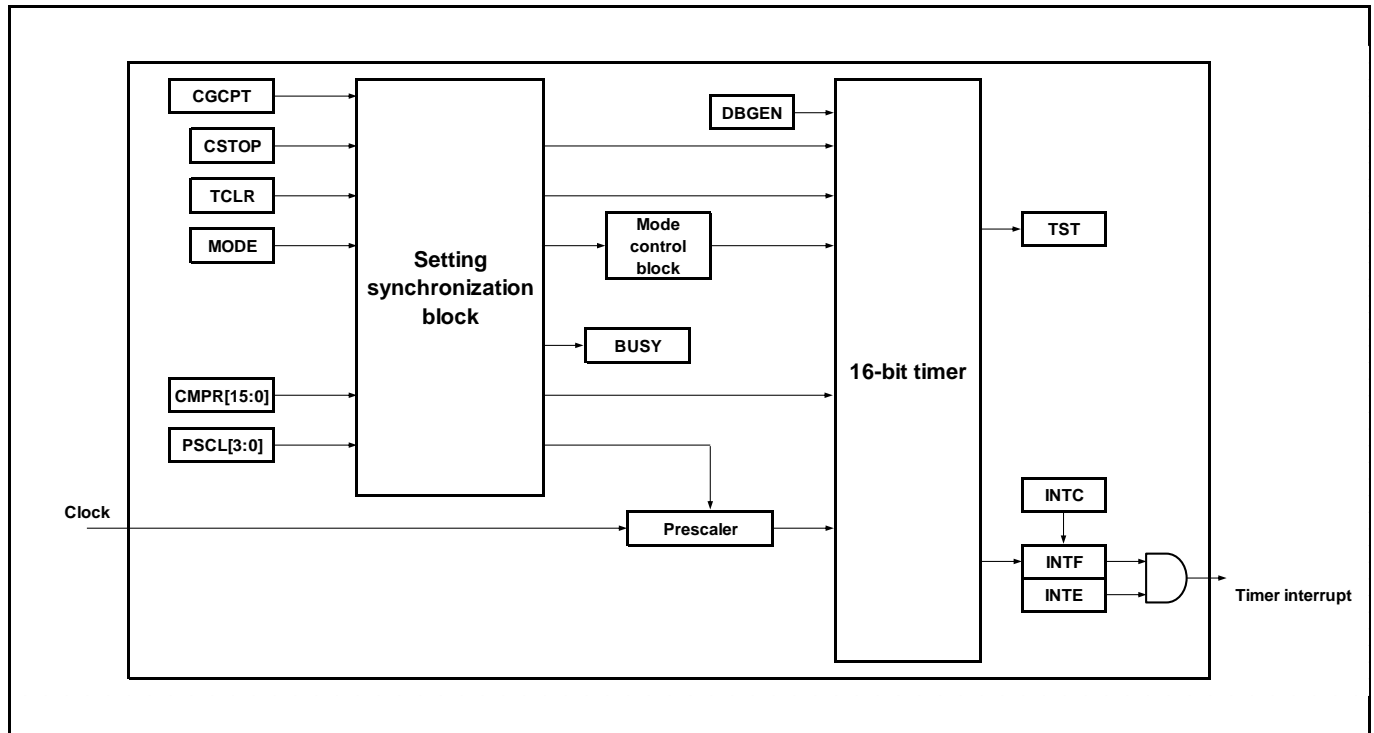
The main clock timer counts the main clock. This timer is used to count time to wait for the main clock oscillation to stabilize.

The main clock timer has an initial value of 8.19 ms (at 4 MHz) for the oscillation stabilization wait time.

## 2. Configuration

This section shows a block diagram of the source clock timer.

Figure 2-1 Block Diagram of Source Clock Timer



### a) Setting Synchronization Block

- The setting synchronization block synchronizes the start of the source clock timer or the resetting of the timer. The timer setting capture bit (CGCPT) is set to "1" to acquire the contents of the timer settings.

### b) Mode Control Block

- The mode control block controls the operation mode of the 16-bit timer.
- The 2 operation modes for the source clock timer are single-shot mode and continuous mode. One of the modes is selected with the mode control bit (MODE).
- Single-shot mode: This mode clears the value to "0x0000" and stops counting up when the count value of the 16-bit timer is equal to or greater than the compare value.
- Continuous mode: This mode clears the value to "0x0000" and restarts counting up when the count value of the 16-bit timer is equal to or greater than the compare value.
- The mode control bit (MODE) is used to select an operation mode.
- If the debug enable bit (DBGEN) is set to "1", the timer stops when operation stops at a breakpoint during debugging.

### c) Prescaler

- The prescaler divides the input clock. For division, a power of 2 can be selected with the prescale bits (PSCL[3:0]). The range for the selection is from no division (divided by 1) to divided by 32768. The divided clock is supplied to the 16-bit timer.





**d) 16-bit timer**

- The 16-bit timer is configured as a 16-bit counter and 16-bit comparator. This timer counts the clock supplied from the prescaler. The interrupt flag bit (INTF) is set to "1" and the count value is cleared to "0x0000" when the count value is equal to or greater than the set value for the compare value bits (CMPR[15:0]).
- In single-shot mode, the interrupt flag bit (INTF) is set to "1" only once. In continuous mode, "1" is set at a regular interval.

### 3. Operation

This section explains the operations of the source clock timer.

#### (1) Oscillation Stabilization Wait Operation

- The source clock timer starts the oscillation stabilization wait operation when a hardware reset is released or oscillation is enabled. The source clock timer gates the clock output until the oscillation stabilization wait time ends.
- Enabling oscillation from the disabled state initializes the setting synchronization block. The oscillation stabilization wait operation counts the stabilization wait time with the initial value setting regardless of the value in the timer compare prescaler register.
- If data is written to a trigger register, timer control register, timer compare prescaler register, interrupt enable register, or interrupt clear register during the oscillation stabilization wait operation, release the sequence protection in privileged mode. If the above is not complied with, a bus error is returned.

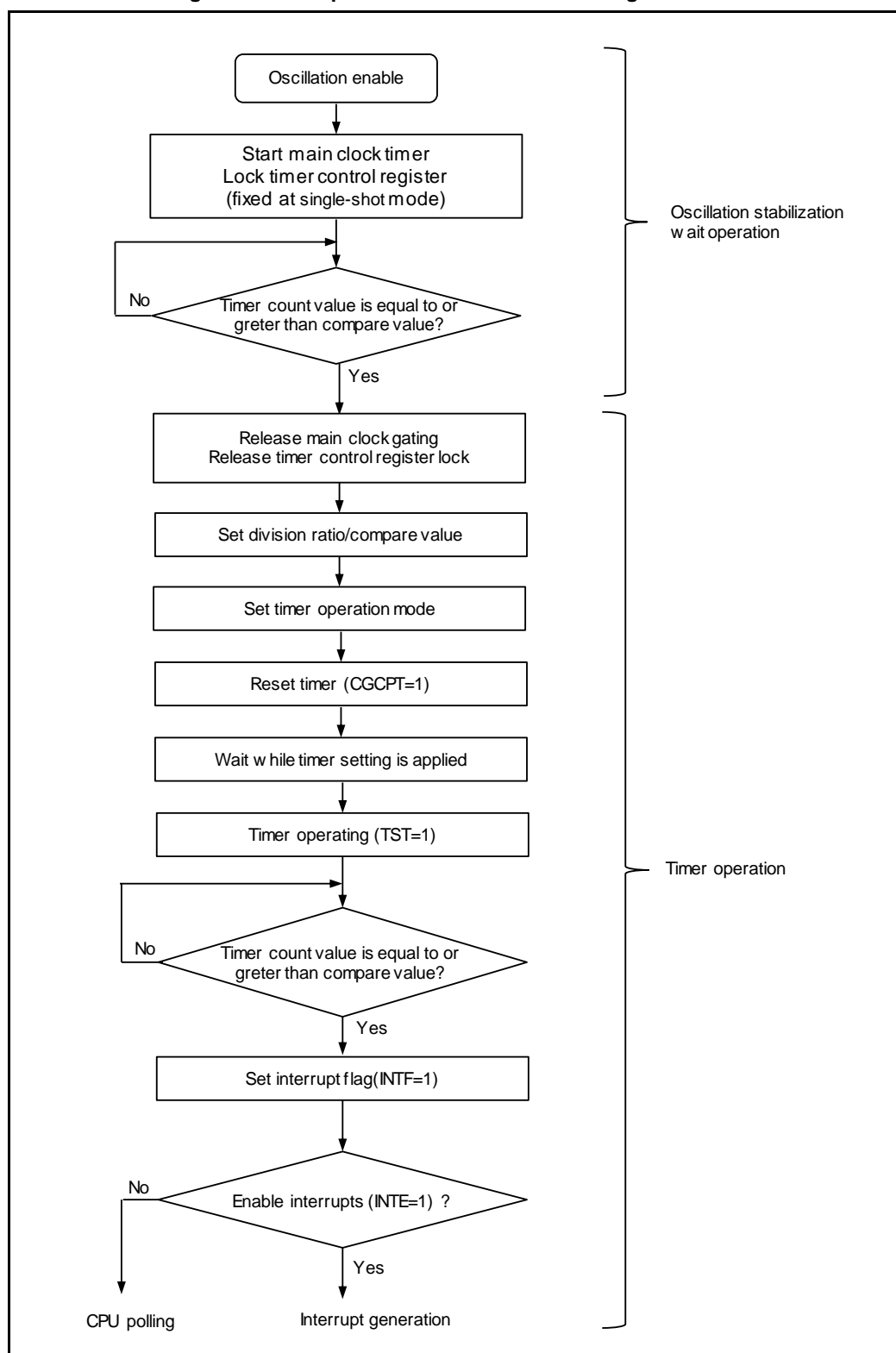
#### (2) Timer Operation

- After the clock oscillation stabilizes, the source clock timer can be used as an ordinary timer.
- This timer can generate an interrupt within a certain period.

## 4. Setting Procedure Example

This section shows an example of the source clock timer setting procedure.

Figure 4-1 Example of Main Clock Timer Setting Procedure



## 5. Registers

This section lists the source clock timer registers.

**Table 5-1 List of Fast-CR Clock Timer Registers**

Abbreviated Register Name	Register Name	See
SYSC_FCRCTTRGR	Fast-CR Clock Timer Trigger Register	5.1
SYSC_FCRCTCNTR	Fast-CR Clock Timer Control Register	5.2
SYSC_FCRCTCPR	Fast-CR Clock Timer Compare Prescaler Register	5.3
SYSC_FCRCTSTR	Fast-CR Clock Timer Status Register	5.4
SYSC_FCRCTINTER	Fast-CR Clock Timer Interrupt Enable Register	5.5
SYSC_FCRCTICLR	Fast-CR Clock Timer Interrupt Clear Register	5.6

**Table 5-2 List of Slow-CR Clock Timer Registers**

Abbreviated Register Name	Register Name	See
SYSC_SCRCTTRGR	Slow-CR Clock Timer Trigger Register	5.7
SYSC_SCRCTCNTR	Slow-CR Clock Timer Control Register	5.8
SYSC_SCRCTCPR	Slow-CR Clock Timer Compare Prescaler Register	5.9
SYSC_SCRCTSTR	Slow-CR Clock Timer Status Register	5.10
SYSC_SCRCTINTER	Slow-CR Clock Timer Interrupt Enable Register	5.11
SYSC_SCRCTICLR	Slow-CR Clock Timer Interrupt Clear Register	5.12

**Table 5-3 List of Main Clock Timer Registers**

Abbreviated Register Name	Register Name	See
SYSC_MOCTTRGR	Main Clock Timer Trigger Register	5.13
SYSC_MOCTCNTR	Main Clock Timer Control Register	5.14
SYSC_MOCTCPR	Main Clock Timer Compare Prescaler Register	5.15
SYSC_MOCTSTR	Main Clock Timer Status Register	5.16
SYSC_MOCTINTER	Main Clock Timer Interrupt Enable Register	5.17
SYSC_MOCTICLR	Main Clock Timer Interrupt Clear Register	5.18



## 5.1. Fast-CR Clock Timer Trigger Register (SYSC\_FCRCTTRGR)

The fast-CR clock timer trigger register (SYSC\_FCRCTTRGR) is used to clear the count value of the timer, stop counting, and change the timer settings.

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved					TCLR	CSTOP	CGCPT
ACCESS_TYPE	R0,WX					R0,W	R0,W	R0,W
PROT_TYPE	-							
INITIAL_VALUE	00000					0	0	0

[bit31:3] Reserved: Reserved bits

### [bit2] TCLR: Timer clear bit

This bit clears the count value of the fast-CR clock timer.

Value	Description
0	No effect
1	Clear the timer count

**Note:**

- Setting the timer setting capture bit (CGCPT) to "1" at the same time as this bit is set to "1" clears the count value of the fast-CR clock timer.

### [bit1] CSTOP: Count stop bit

This bit stops the counting of the fast-CR clock timer.

Value	Description
0	No effect
1	Stop the timer counting

**Note:**

- Setting the timer setting capture bit (CGCPT) to "1" at the same time as this bit is set to "1" stops the fast-CR clock timer.

### [bit0] CGCPT: Timer setting capture bit

This bit is used to change the fast-CR clock timer settings and to start the timer counting. Setting this bit to "1" causes the fast-CR clock timer to acquire the changed setting contents.

Value	Description
0	No effect
1	Change the timer settings / Start the timer counting

## 5.2. Fast-CR Clock Timer Control Register (SYSC\_FCRCTCNTR)

The fast-CR clock timer control register (SYSC\_FCRCTCNTR) selects the operation mode of the timer and sets whether the counter operates or stops during debugging.

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved						DBGEN	MODE
ACCESS_TYPE	R0,WX						R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	000000						0	0

**[bit31:2] Reserved: Reserved bits**

**[bit1] DBGEN: Debug enable bit**

This bit sets the operation of the fast-CR clock timer counter during debugging.

Value	Description
0	Continue the count operation at the breakpoint
1	Stop the count operation at the breakpoint

**Note:**

- This bit cannot be written during the oscillation stabilization wait operation. If writing is attempted, a bus error is returned.

**[bit0] MODE: Mode control bit**

This bit selects the operation mode of the fast-CR clock timer.

Single-shot mode clears the count and stops the count operation when the timer counter value is equal to or greater than the compare value (CMPR[15:0]).

Continuous mode clears the count and restarts the count operation when the timer counter value is equal to or greater than the compare value (CMPR[15:0]).

Value	Description
0	Single-shot mode
1	Continuous mode

**Notes:**

- Setting the timer setting capture bit (CGCPT) to "1" after this bit is set changes the operation mode of the fast-CR clock timer.
- This bit cannot be written during the oscillation stabilization wait operation. If writing is attempted, a bus error is returned.



### 5.3. Fast-CR Clock Timer Compare Prescaler Register (SYSC\_FCRCTCPR)

The fast-CR clock timer compare prescaler register (SYSC\_FCRCTCPR) selects the division ratio of the input clock and sets the compare value of the timer.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved				PSCL			
ACCESS_TYPE	R0,WX				R/W			
PROT_TYPE	-							
INITIAL_VALUE	0000				0110			

BIT_OFFSET	15-0							
BIT_NAME	CMPR							
ACCESS_TYPE	R/W							
PROT_TYPE	-							
INITIAL_VALUE	00000000_00011110							

**[bit31:20] Reserved: Reserved bits**

**[bit19:16] PSCL[3:0]: Prescale bits**

These bits select the division ratio of the input clock of the fast-CR clock timer.

Value	Description
0000	No division (divided by 1)
0001	Divided by 2
0010	Divided by 4
0011	Divided by 8
0100	Divided by 16
0101	Divided by 32
0110	Divided by 64
0111	Divided by 128
1000	Divided by 256
1001	Divided by 512
1010	Divided by 1024
1011	Divided by 2048
1100	Divided by 4096
1101	Divided by 8192
1110	Divided by 16384
1111	Divided by 32768

**Note:**

- Setting the timer setting capture bit (CGCPT) to "1" after these bits are set changes the division ratio of the fast-CR clock timer.

**[bit15:0] CMPR[15:0]: Compare value bits**

These bits set the compare value of the fast-CR clock timer.

The interrupt flag bit (INTF) of the fast-CR clock timer is set to "1" when the count value of the fast-CR clock timer is equal to or greater than the compare value.

**Notes:**

- Setting the timer setting capture bit (CGCPT) to "1" after these bits are set changes the compare value of the fast-CR clock timer.
- During timer operation, CMPR[15:0] bits are not changeable.

Oscillation stabilization wait time of the fast-CR clock

Prescale PSCL[3:0]	Compare Value CMPR[15:0]	Oscillation Stabilization Wait Time (Fast-CR Clock Cycle x Prescale Value x Compare Value)
0110 (divided by 64)	0x001E(30)	0.24 [ms]





## 5.4. Fast-CR Clock Timer Status Register (SYSC\_FCRCTSTR)

The fast-CR clock timer status register (SYSC\_FCRCTSTR) indicates the update status of a timer reset, the operation status of the timer, and an interrupt flag.

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved					BUSY	TST	INTF
ACCESS_TYPE	R0,WX					R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	00000					0	0	0

**[bit31:3] Reserved: Reserved bits**

### **[bit2] BUSY: Setting update status bit**

This bit indicates the timer setting update status of the fast-CR clock timer.

If the timer setting capture bit (CGCPT) is set to "1", this setting value remains "1" until the timer setting update is completed.

Value	Description
0	Setting completed
1	Setting update in progress

**Note:**

- Registers other than the interrupt enable register and interrupt clear register of the fast-CR clock timer must not be changed when the read value of this bit is "1".

### **[bit1] TST: Timer status bit**

This bit indicates the operation status of the fast-CR clock timer.

Value	Description
0	Timer stopped
1	Timer operating

**[bit0] INTF: Interrupt flag bit**

This bit is the interrupt flag bit of the fast-CR clock timer.

"1" is set when the fast-CR clock timer value is equal to or greater than the set value in the compare bits (CMPR[15:0]). At this time, if the interrupt enable bit (INTE) is "1", the fast-CR clock timer generates an interrupt request.

To clear the interrupt flag bit to "0", set the interrupt clear bit (INTC) to "1".

Value	Description
0	No interrupt factor is detected
1	An interrupt factor is detected

**Note:**

- *This bit is set only while the timer is operating. It is not set during the oscillation stabilization wait operation.*



## 5.5. Fast-CR Clock Timer Interrupt Enable Register (SYSC\_FCRCTINTER)

The fast-CR clock timer interrupt enable register (SYSC\_FCRCTINTER) sets whether to enable or disable interrupts of the timer.

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved							INTE
ACCESS_TYPE	R0,WX							R/W
PROT_TYPE	-							
INITIAL_VALUE	00000000							0

**[bit31:1] Reserved: Reserved bits**

**[bit0] INTE: Interrupt enable bit**

This bit sets whether to enable or disable interrupts of the fast-CR clock timer.

Value	Description
0	Disable interrupts
1	Enable interrupts

## 5.6. Fast-CR Clock Timer Interrupt Clear Register (SYSC\_FCRCTICLR)

The fast-CR clock timer interrupt clear register (SYSC\_FCRCTICLR) clears an interrupt flag to "0".

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved							INTC
ACCESS_TYPE	R0,WX							R0,W
PROT_TYPE	-							
INITIAL_VALUE	00000000							0

**[bit31:1] Reserved: Reserved bits**

**[bit0] INTC: Interrupt clear bit**

This bit clears the interrupt flag bit (INTF) of the fast-CR clock timer to "0".

Value	Description
0	No effect
1	Clear the interrupt flag bit (INTF) to "0"



## 5.7. Slow-CR Clock Timer Trigger Register (SYSC\_SCRCTTRGR)

The slow-CR clock timer trigger register (SYSC\_SCRCTTRGR) is used to clear the count value of the timer, stop counting, and change the timer settings.

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved					TCLR	CSTOP	CGCPT
ACCESS_TYPE	R0,WX					R0,W	R0,W	R0,W
PROT_TYPE	-							
INITIAL_VALUE	00000					0	0	0

[bit31:3] Reserved: Reserved bits

### [bit2] TCLR: Timer clear bit

This bit clears the count value of the slow-CR clock timer.

Value	Description
0	No effect
1	Clear the timer count

**Note:**

- Setting the timer setting capture bit (CGCPT) to "1" at the same time as this bit is set to "1" clears the count value of the slow-CR clock timer.

### [bit1] CSTOP: Count stop bit

This bit stops the counting of the slow-CR clock timer.

Value	Description
0	No effect
1	Stop the timer counting

**Note:**

- Setting the timer setting capture bit (CGCPT) to "1" at the same time as this bit is set to "1" stops the slow-CR clock timer.

### [bit0] CGCPT: Timer setting capture bit

This bit is used to change the slow-CR clock timer settings and to start the timer counting. Setting this bit to "1" causes the slow-CR clock timer to acquire the changed setting contents.

Value	Description
0	No effect
1	Change the timer settings / Start the timer counting

## 5.8. Slow-CR Clock Timer Control Register (SYSC\_SCRTCNTR)

The slow-CR clock timer control register (SYSC\_SCRTCNTR) selects the operation mode of the timer and sets whether the counter operates or stops during debugging.

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved						DBGEN	MODE
ACCESS_TYPE	R0,WX						R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	000000						0	0

**[bit31:2] Reserved: Reserved bits**

**[bit1] DBGEN: Debug enable bit**

This bit sets the operation of the slow-CR clock timer counter during debugging.

Value	Description
0	Continue the count operation at the breakpoint
1	Stop the count operation at the breakpoint

**Note:**

- This bit cannot be written during the oscillation stabilization wait operation. If writing is attempted, a bus error is returned.

**[bit0] MODE: Mode control bit**

This bit selects the operation mode of the slow-CR clock timer.

Single-shot mode clears the count and stops the count operation when the timer counter value is equal to or greater than the compare value (CMPR[15:0]).

Continuous mode clears the count and restarts the count operation when the timer counter value is equal to or greater than the compare value (CMPR[15:0]).

Value	Description
0	Single-shot mode
1	Continuous mode

**Notes:**

- Setting the timer setting capture bit (CGCPT) to "1" after this bit is set changes the operation mode of the slow-CR clock timer.
- This bit cannot be written during the oscillation stabilization wait operation. If writing is attempted, a bus error is returned.



## 5.9. Slow-CR Clock Timer Compare Prescaler Register (SYSC\_SCRCTCPR)

The slow-CR clock timer compare prescaler register (SYSC\_SCRCTCPR) selects the division ratio of the input clock and sets the compare value of the timer.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved				PSCL			
ACCESS_TYPE	R0,WX				R/W			
PROT_TYPE	-							
INITIAL_VALUE	0000				0110			

BIT_OFFSET	15-0							
BIT_NAME	CMPR							
ACCESS_TYPE	R/W							
PROT_TYPE	-							
INITIAL_VALUE	00000000_00000001							

**[bit31:20] Reserved: Reserved bits**

**[bit19:16] PSCL[3:0]: Prescale bits**

These bits select the division ratio of the input clock of the slow-CR clock timer.

Value	Description
0000	No division (divided by 1)
0001	Divided by 2
0010	Divided by 4
0011	Divided by 8
0100	Divided by 16
0101	Divided by 32
0110	Divided by 64
0111	Divided by 128
1000	Divided by 256
1001	Divided by 512
1010	Divided by 1024
1011	Divided by 2048
1100	Divided by 4096
1101	Divided by 8192
1110	Divided by 16384
1111	Divided by 32768

**Note:**

- Setting the timer setting capture bit (CGCPT) to "1" after these bits are set changes the division ratio of the slow-CR clock timer.

**[bit15:0] CMPR[15:0]: Compare value bits**

These bits set the compare value of the slow-CR clock timer.

The interrupt flag bit (INTF) is set to "1" when the count value of the slow-CR clock timer is equal to or greater than the compare value.

**Notes:**

- Setting the timer setting capture bit (CGCPT) to "1" after these bits are set changes the compare value of the slow-CR clock timer.
- During timer operation, CMPR[15:0] bits are not changeable.

Oscillation stabilization wait time of the slow-CR clock

Prescale PSCL[3:0]	Compare Value CMPR[15:0]	Oscillation Stabilization Wait Time (Slow-CR Clock Cycle x Prescale Value x Compare Value)
0110 (divided by 64)	0x0001(1)	0.64 [ms]





## 5.10. Slow-CR Clock Timer Status Register (SYSC\_SCRCTSTR)

The slow-CR clock timer status register (SYSC\_SCRCTSTR) indicates the update status of a timer reset, the operation status of the timer, and an interrupt flag.

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved					BUSY	TST	INTF
ACCESS_TYPE	R0,WX					R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	00000					0	0	0

**[bit31:3] Reserved: Reserved bits**

### **[bit2] BUSY: Setting update status bit**

This bit indicates the setting update status of the slow-CR clock timer.

If the timer setting capture bit (CGCPT) is set to "1", this setting value remains "1" until the timer setting update is completed.

Value	Description
0	Setting completed
1	Setting update in progress

**Note:**

- Registers other than the interrupt enable register and interrupt clear register of the slow-CR clock timer must not be changed when the read value of this bit is "1".

### **[bit1] TST: Timer status bit**

This bit indicates the operation status of the slow-CR clock timer.

Value	Description
0	Timer stopped
1	Timer operating

**[bit0] INTF: Interrupt flag bit**

This bit is the interrupt flag bit of the slow-CR clock timer.

"1" is set when the slow-CR clock timer value is equal to or greater than the set value in the compare bits (CMPR[15:0]). At this time, if the interrupt enable bit (INTE) is "1", the slow-CR clock timer generates an interrupt request.

To clear the interrupt flag bit to "0", set the interrupt clear bit (INTC) to "1".

Value	Description
0	No interrupt factor is detected
1	An interrupt factor is detected

**Note:**

- This bit can be set only while the timer is operating. It cannot be set during the oscillation stabilization wait operation.



## 5.11. Slow-CR Clock Timer Interrupt Enable Register (SYSC\_SCRCTINTER)

The slow-CR clock timer interrupt enable register (SYSC\_SCRCTINTER) sets whether to enable or disable interrupts of the timer.

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved							INTE
ACCESS_TYPE	R0,WX							R/W
PROT_TYPE	-							
INITIAL_VALUE	0000000							0

**[bit31:1] Reserved: Reserved bits**

**[bit0] INTE: Interrupt enable bit**

This bit sets whether to enable or disable interrupts of the slow-CR clock timer.

Value	Description
0	Disable interrupts
1	Enable interrupts

## 5.12. Slow-CR Clock Timer Interrupt Clear Register (SYSC\_SCRCTICLR)

The slow-CR clock timer interrupt clear register (SYSC\_SCRCTICLR) clears an interrupt flag to "0".

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved							INTC
ACCESS_TYPE	R0,WX							R0,W
PROT_TYPE	-							
INITIAL_VALUE	00000000							0

**[bit31:1] Reserved: Reserved bits**

**[bit0] INTC: Interrupt clear bit**

This bit clears the interrupt flag bit (INTF) of the slow-CR clock timer to "0".

Value	Description
0	No effect
1	Clear the interrupt flag bit (INTF) to "0"



### 5.13. Main Clock Timer Trigger Register (SYSC\_MOCTTRGR)

The main clock timer trigger register (SYSC\_MOCTTRGR) is used to clear the count value of the timer, stop counting, and change the timer settings.

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved					TCLR	CSTOP	CGCPT
ACCESS_TYPE	R0,WX					R0,W	R0,W	R0,W
PROT_TYPE	-							
INITIAL_VALUE	00000					0	0	0

[bit31:3] Reserved: Reserved bits

#### [bit2] TCLR: Timer clear bit

This bit clears the count value of the main clock timer.

Value	Description
0	No effect
1	Clear the timer count

**Note:**

- Setting the timer setting capture bit (CGCPT) to "1" at the same time as this bit is set to "1" clears the count value of the main clock timer.

#### [bit1] CSTOP: Count stop bit

This bit stops the counting of the main clock timer.

Value	Description
0	No effect
1	Stop the timer counting

**Note:**

- Setting the timer setting capture bit (CGCPT) to "1" at the same time as this bit is set to "1" stops the main clock timer.

#### [bit0] CGCPT: Timer setting capture bit

This bit is used to change the main clock timer settings and to start the timer counting. Setting this bit to "1" causes the main clock timer to acquire the changed setting contents.

Value	Description
0	No effect
1	Change the timer settings / Start the timer counting

## 5.14. Main Clock Timer Control Register (SYSC\_MOCTCNTR)

The main clock timer control register (SYSC\_MOCTCNTR) selects the operation mode of the timer and sets whether the counter operates or stops during debugging.

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved						DBGEN	MODE
ACCESS_TYPE	R0,WX						R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	000000						0	0

**[bit31:2] Reserved: Reserved bits**

**[bit1] DBGEN: Debug enable bit**

This bit sets the operation of the main clock timer counter during debugging.

Value	Description
0	Continue the count operation at the breakpoint
1	Stop the count operation at the breakpoint

**Note:**

- This bit cannot be written during the oscillation stabilization wait operation. If writing is attempted, a bus error is returned.

**[bit0] MODE: Mode control bit**

This bit selects the operation mode of the main clock timer.

Single-shot mode clears the count and stops the count operation when the timer counter value is equal to or greater than the compare value (CMPR[15:0]).

Continuous mode clears the count and restarts the count operation when the timer counter value is equal to or greater than the compare value (CMPR[15:0]).

Value	Description
0	Single-shot mode
1	Continuous mode

**Notes:**

- Setting the timer setting capture bit (CGCPT) to "1" after this bit is set changes the operation mode of the main clock timer.
- This bit cannot be written during the oscillation stabilization wait operation. If writing is attempted, a bus error is returned.



## 5.15. Main Clock Timer Compare Prescaler Register (SYSC\_MOCTCPR)

The main clock timer compare prescaler register (SYSC\_MOCTCPR) selects the division ratio of the input clock and sets the compare value of the timer.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved				PSCL			
ACCESS_TYPE	R0,WX				R/W			
PROT_TYPE	-							
INITIAL_VALUE	0000				0110			

BIT_OFFSET	15-0							
BIT_NAME	CMPR							
ACCESS_TYPE	R/W							
PROT_TYPE	-							
INITIAL_VALUE	00000010_00000000							

**[bit31:20] Reserved: Reserved bits**

**[bit19:16] PSCL[3:0]: Prescale bits**

These bits select the division ratio of the input clock of the main clock timer.

Value	Description
0000	No division (divided by 1)
0001	Divided by 2
0010	Divided by 4
0011	Divided by 8
0100	Divided by 16
0101	Divided by 32
0110	Divided by 64
0111	Divided by 128
1000	Divided by 256
1001	Divided by 512
1010	Divided by 1024
1011	Divided by 2048
1100	Divided by 4096
1101	Divided by 8192
1110	Divided by 16384
1111	Divided by 32768

**Note:**

- Setting the timer setting capture bit (CGCPT) to "1" after these bits are set changes the division ratio of the main clock timer.

**[bit15:0] CMPR[15:0]: Compare value bits**

These bits set the compare value of the main clock timer.

The interrupt flag bit (INTF) is set to "1" when the count value of the main clock timer is equal to or greater than the compare value.

**Notes:**

- Setting the timer setting capture bit (CGCPT) to "1" after these bits are set changes the compare value of the main clock timer.
- During timer operation, CMPR[15:0] bits are not changeable.

Oscillation stabilization wait time of the main clock

Prescale PSCL[3:0]	Compare Value CMPR[15:0]	Oscillation Stabilization Wait Time (Main Clock Cycle x Prescale Value x Compare Value)
0110 (divided by 64)	0x0200(512)	8.19 [ms]





## 5.16. Main Clock Timer Status Register (SYSC\_MOCTSTR)

The main clock timer status register (SYSC\_MOCTSTR) indicates the update status of a timer reset, the operation status of the timer, and an interrupt flag.

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved					BUSY	TST	INTF
ACCESS_TYPE	R0,WX					R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	00000					0	0	0

**[bit31:3] Reserved: Reserved bits**

### **[bit2] BUSY: Setting update status bit**

This bit indicates the setting update status of the main clock timer.

If the timer setting capture bit (CGCPT) is set to "1", this setting value remains "1" until the timer setting update is completed.

Value	Description
0	Setting completed
1	Setting update in progress

**Note:**

- Registers other than the interrupt enable register and interrupt clear register of the main clock timer must not be changed when the read value of this bit is "1".

### **[bit1] TST: Timer status bit**

This bit indicates the operation status of the main clock timer.

Value	Description
0	Timer stopped
1	Timer operating

**[bit0] INTF: Interrupt flag bit**

This bit is the interrupt flag bit of the main clock timer.

"1" is set when the main clock timer value is equal to or greater than the set value in the compare bits (CMPR[15:0]). At this time, if the interrupt enable bit (INTE) is "1", the main clock timer generates an interrupt request.

To clear the interrupt flag bit to "0", set the interrupt clear bit (INTC) to "1".

Value	Description
0	No interrupt factor is detected
1	An interrupt factor is detected

**Note:**

- This bit can be set only while the timer is operating. It cannot be set during the oscillation stabilization wait operation.



## 5.17. Main Clock Timer Interrupt Enable Register (SYSC\_MOCTINTER)

The main clock timer interrupt enable register (SYSC\_MOCTINTER) sets whether to enable or disable interrupts of the timer.

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved							INTE
ACCESS_TYPE	R0,WX							R/W
PROT_TYPE	-							
INITIAL_VALUE	00000000							0

**[bit31:1] Reserved: Reserved bits**

**[bit0] INTE: Interrupt enable bit**

This bit sets whether to enable or disable interrupts of the main clock timer.

Value	Description
0	Disable interrupts
1	Enable interrupts

## 5.18. Main Clock Timer Interrupt Clear Register (SYSC\_MOCTICLR)

The main clock timer interrupt clear register (SYSC\_MOCTICLR) clears an interrupt flag.

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved							INTC
ACCESS_TYPE	R0,WX							R0,W
PROT_TYPE	-							
INITIAL_VALUE	00000000							0

**[bit31:1] Reserved: Reserved bits**

**[bit0] INTC: Interrupt clear bit**

This bit clears the interrupt flag bit (INTF) of the main clock timer to "0".

Value	Description
0	No effect
1	Clear the interrupt flag bit (INTF) to "0"



## 6. Precautions for Using

This section explains precautions for using the source clock timer.

- To operate a source clock timer with a new setting after stopping the timer operation halfway, perform the timer initialization.  
Timer initialization indicates initialization of 16-bit timer and prescaler.  
When initialing the 16-bit timer, set 0x0000 in compare value bits (CMPR[15:0]) and "1" in timer setting capture bit (CGCPT).  
No interrupt signal is generated at 16-bit timer initialization.  
Confirm the initialization completion by the timer is stopped (TST=0).  
After 16-bit timer initialization, perform prescaler initialization continuously.
- When initialing the prescaler, set 0x0000 in compare value bits (CMPR[15:0]), "0001" in prescale bit (PSCL[3:0]) and "1" in timer setting capture bit (CGCPT).  
No interrupt signal is generated at prescaler initialization.  
Confirm the initialization completion by setting update status bit (BUSY) is "0".  
The timer can work with a new setting by the above-mentioned timer initialization.

[Timer initialization procedure]

16-bit timer initialization procedure

1. Set "0x0000" in the compare value bits (CMPR).
2. Set "1" in the timer setting capture bit (CGCPT).
3. Read the timer status bit (TST), and then confirm that the timer is stopped (TST = 0).

Prescaler initialization procedure

1. Set "0x0000" in compare value bits (CMPR[15:0]) and "0001" in prescale bits (PSCL[3:0]).
2. Set "1" in the timer setting capture bit (CGCPT).
3. Read the timer status bit (TST) and then confirm that the timer is stopped (TST=0).

- When the timer is stopped by using the count stop bit (CSTOP), the timer counting is suspended halfway in operation. If "1" is set in the timer setting capture bit (CGCPT) with the same setting before the timer stops, the timer operation is resumed from the suspended state. To operate the timer with a new setting, initialize the timer.
- The timer clear bit (TCLR) clears the count value of the 16-bit timer. The count operation is not stopped. To operate the timer with a new setting, initialize the timer.
- When changing prescale bits (PSCL[3:0]) from "0000" to other than "0000", initialize the prescaler according to prescaler initialization procedure.



# CHAPTER 10: Interrupt Controller

**This Chapter Explains the Interrupt Controller.**

---

1. Overview
2. Configuration
3. Operation
4. Setting Procedure Examples
5. Registers
6. Others



## 1. Overview

This section provides an overview of the interrupt controller.

The interrupt controller supports 512 channels for normal interrupts (IRQs) and 32 channels for non-maskable interrupts (NMIs). All IRQs are processed so that they issue a single interrupt request (nIRQ) to the CPU. All NMIs are processed so that they issue a single fast interrupt request (nFIQ) to the CPU.

The interrupt controller has the following functions.

- Controls the issue of interrupts (nIRQ and nFIQ) to the CPU.
- Supports 512 IRQ channels for nIRQ generation.
- Supports 32 NMI channels for nFIQ generation.
- Default interrupt priority levels. (Those channels with the lower channel numbers take precedence.)
- Internally synchronizes asynchronous interrupts.
- IRQ priority level control which allows 32-level settings.
- Controls IRQ priority level masking.
- NMI priority level control which allows 16-level settings. (Note that only NMI0 is fixed to level 0.)
- Supports vectored interrupt for both IRQ and NMI.
- A 32-bit vector address per interrupt channel.
- Generates a software interrupt for both IRQ and NMI.
- Supports privileged mode for access restriction.

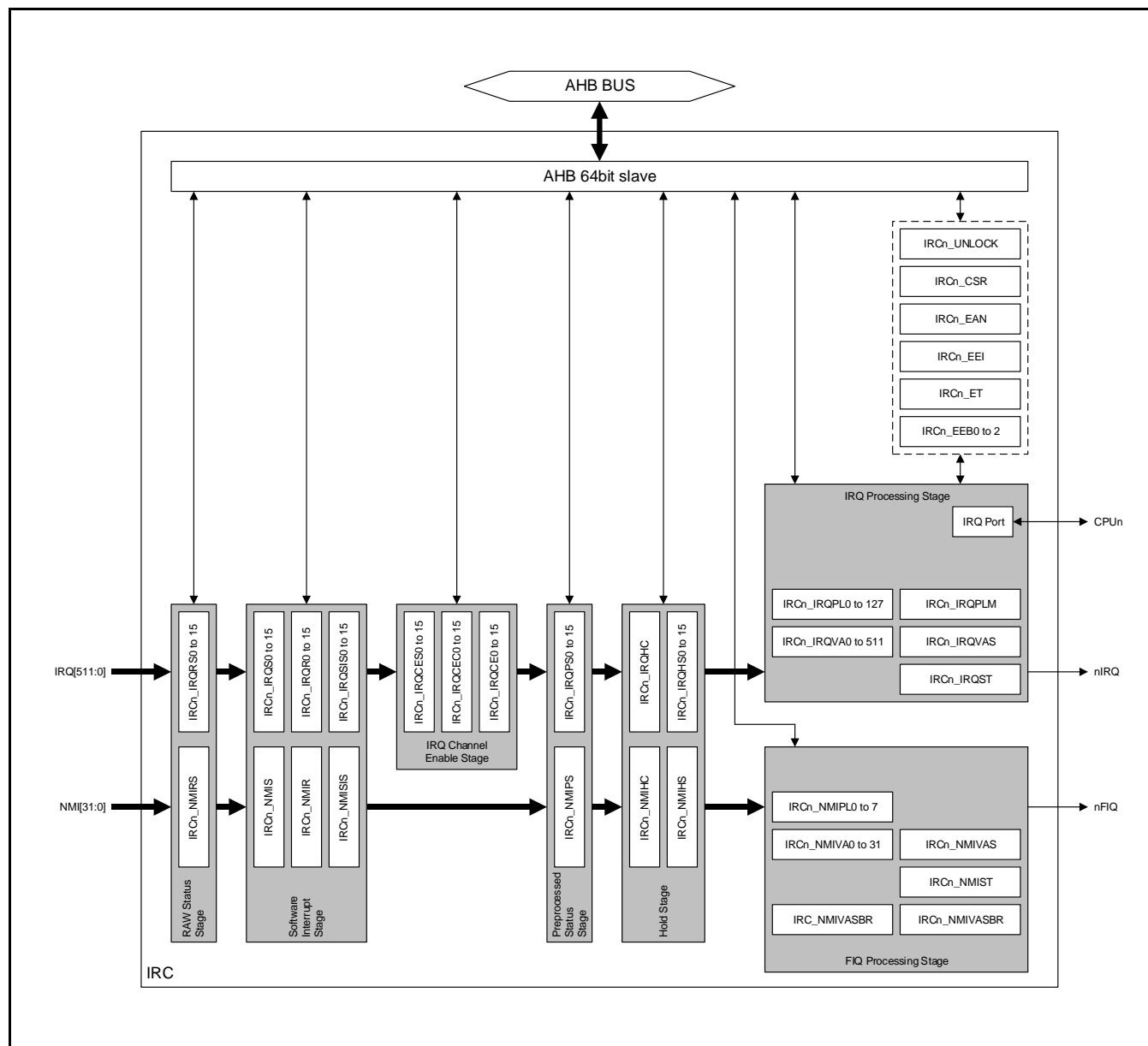
### Notes:

- In this chapter, you may see the abbreviated notation, *IRC*, which stands for *InterRupt Controller*. You may also see the notations *ISR* and *VIC*, which stand for *Interrupt Service Routine* and *Vectored Interrupt Controller*, respectively.
- Regarding the abbreviations used for the registers of the interrupt controller, the "n" in "IRCn\_\*\*\*" corresponds to the CPU number, "m" in "IRCn\_\*\*m" to the NMI channel number (0 to 31), and "r" in "IRCn\_\*\*\*r" to the IRQ channel number (0 to 511).
- For details on the interrupt factor assignment to each NMI/IRQ channel, see "List of Interrupt Factor and DMA Activation Factor" in "CHAPTER: Appendix".

## 2. Configuration

This section explains the block diagram of the interrupt controller.

Figure 2-1 Block Diagram of Interrupt Controller







### 3. Operation

This section explains the operation of the interrupt controller.

#### (1) Interrupt Controller Stages

##### a) RAW Status Stage

For each IRQ/NMI channel, the interrupt controller posts notification of the RAW status. The RAW status is the interrupt status existing immediately after an input to the interrupt controller. In this status, the interrupt controller does not execute any processing. For both IRQ and NMI, notification of the status is obtained by reading the RAW status register (IRCn\_IRQRS0 to IRCn\_IRQRS15, IRCn\_NMIRS).

##### b) Software Interrupt Stage

The interrupt controller can generate a software interrupt for each IRQ/NMI channel. By writing "1" to a bit in the IRQ software interrupt set register (IRCn\_IRQS0 to IRCn\_IRQS15), a software interrupt is set for the corresponding IRQ channel. By writing "1" to a bit of the IRQ software interrupt reset register (IRCn\_IRQR0 to IRCn\_IRQR15), the software interrupt for the corresponding IRQ channel is reset. The IRQ software interrupt status is obtained by reading the IRQ software interrupt status register (IRCn\_IRQSIS0 to IRCn\_IRQSIS15).

Similarly, for NMI, the interrupt controller has the NMI software interrupt set register (IRCn\_NMIS), NMI software interrupt reset register (IRCn\_NMIR), and NMI software interrupt status register (IRCn\_NMISIS). The operation is same as for IRQ.

Although the interrupt controller supports 512 channels for IRQ and 32 channels for NMI, not every channel is assigned an interrupt factor. Therefore some channels are unused. These unused channels can be used for software interrupts. You can use software interrupts to test the ISRs for hardware interrupts.

##### c) IRQ Channel Enable Stage

For each IRQ channel, the interrupt controller can set either enable or disable. It cannot disable NMI channels. By writing "1" to a bit in the IRQ channel enable set register (IRCn\_IRQCES0 to IRCn\_IRQCES15), the corresponding IRQ channel is enabled. By writing "1" to a bit of the IRQ channel enable clear register (IRCn\_IRQCEC0 to IRCn\_IRQCEC15), the enabled status of the corresponding IRQ channel is cleared. The IRQ channel enable status is obtained by reading the IRQ channel enable setting register (IRCn\_IRQCE0 to IRCn\_IRQCE15). The interrupt controller can also be set either enable or disable by writing directly to the registers from IRCn\_IRQCE0 to IRCn\_IRQCE15.

##### d) Preprocessed Status Stage

The interrupt controller posts notification of the preprocessed interrupt status. In this case, preprocessing means the processing which is executed in the software interrupt stage and the IRQ channel enable stage. For both IRQ and NMI, the status is obtained by reading the preprocessed interrupt status register (IRCn\_IRQPS0 to IRCn\_IRQPS15, IRCn\_NMIPS).

##### e) Hold Stage

For each IRQ/NMI channel, the interrupt controller holds the interrupt hold status. When the IRQ is applied to the CPU, the corresponding bit in the IRQ hold status register (IRCn\_IRQHS0 to IRCn\_IRQHS15) becomes "1". When the NMI is applied to the CPU and the CPU which has accepted the nFIQ reads IRC\_NMIVASBR, the corresponding bit in the NMI hold status register (IRCn\_NMIHS) becomes "1". After ISR ends, the corresponding hold status bit must be cleared. By writing, to the hold clear register (IRCn\_IRQHC, IRCn\_NMIHC), the number of the channel for which the hold status is to be cleared, the corresponding hold status bit is cleared.

#### **f) IRQ Processing Stage**

IRQs are processed according to their software priority which is set in the IRQ priority level register (IRCN\_IRQPL0 to IRCN\_IRQPL127) and their hardware priority, and the channel having the highest priority level is applied. There are 32 software priority levels that can be set. If 2 requests have the same software priority level, that with the higher hardware priority level takes precedence. Hardware priority is fixed and that channel having the lower channel number has the higher priority level.

For IRQ, the interrupt controller has an interrupt mask function provided through the IRQ priority level mask register (IRCN\_IRQPLM). If an IRQ channel has a software priority level (which is set in registers IRCN\_IRQPL0 to IRCN\_IRQPL127) that is higher than the value set in IRCN\_IRQPLM, the channel is masked.

#### **g) FIQ Processing Stage**

NMIs are processed according to their software priority which is set in the NMI priority level register (IRCN\_NMIPL0 to IRCN\_NMIPL7) and their hardware priority, and the channel having the highest priority level is applied. There are 16 software priority levels that can be set. If 2 requests have the same software priority level, that with the higher hardware priority level takes precedence. Hardware priority is fixed and that channel having the lower channel number has the higher priority level. The software priority level of NMI0 is fixed to "0". Since NMI0 has the highest hardware priority level, it always has the highest priority.

#### **(2) Nesting IRQ and NMI**

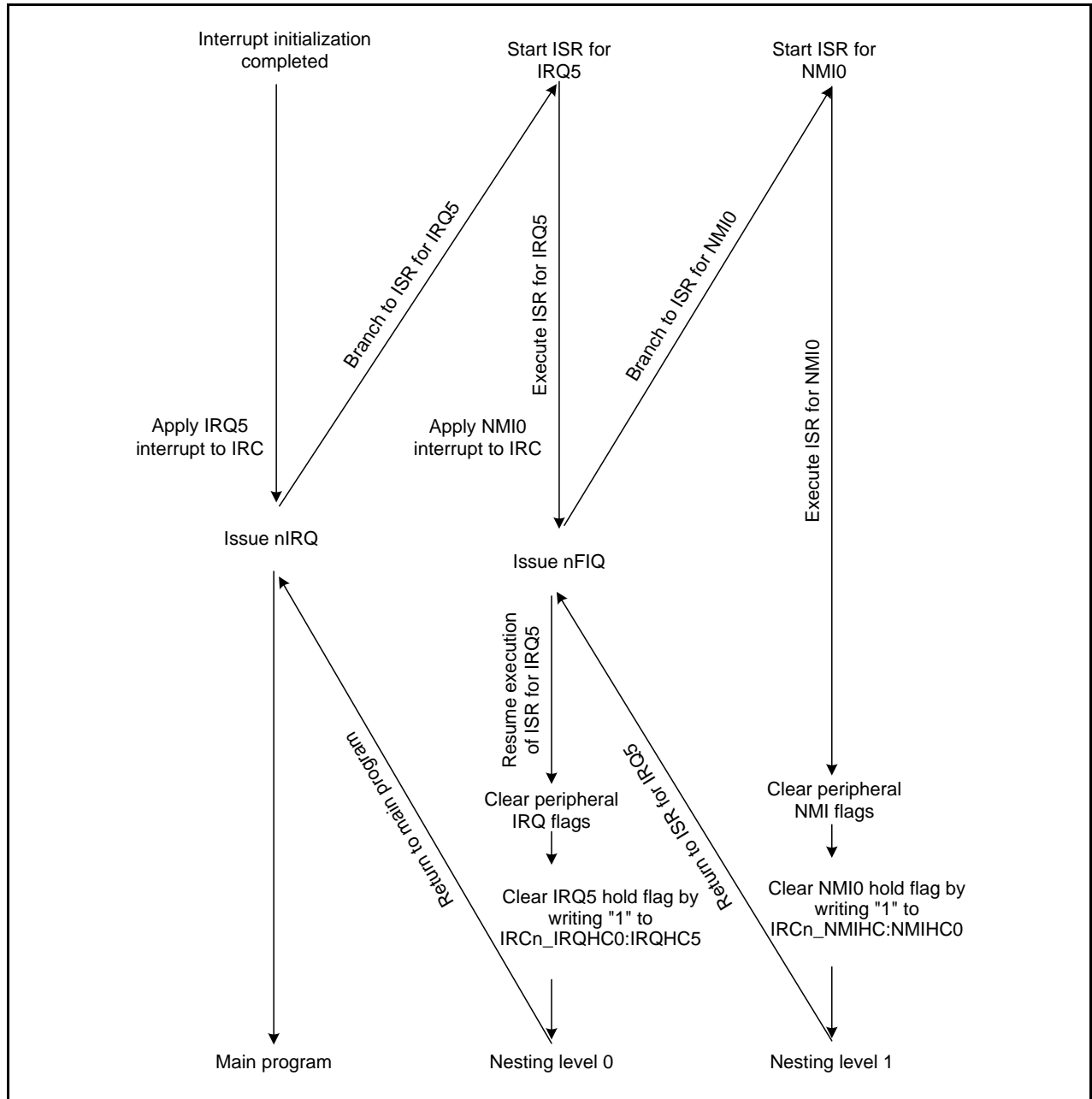
The ARM® Cortex™-R5 register set does not automatically process nested interrupts. This processing should be provided within the ISR by applications. If this is not provided in the ISR, the first active IRQ is executed, but next IRQ will not be accepted and ISR will not be executed until the active ISR ends. This is also true for NMIs. However, since the interrupt controller uses nIRQ for IRQs and nFIQ for NMIs, and since NMIs have a higher hardware priority than that of IRQs, NMIs can interrupt IRQs.

For details on the save/restore of interrupts, see also the Cortex™-R5 Revision:r1p2 Technical Reference Manual (ARM DDI 0460D).



a) Interrupt Operation when Particular Software Is Not Used for Nested Processing

Figure 3-1 ISR Processing Flow when NMI Generation Causes Nested IRQ

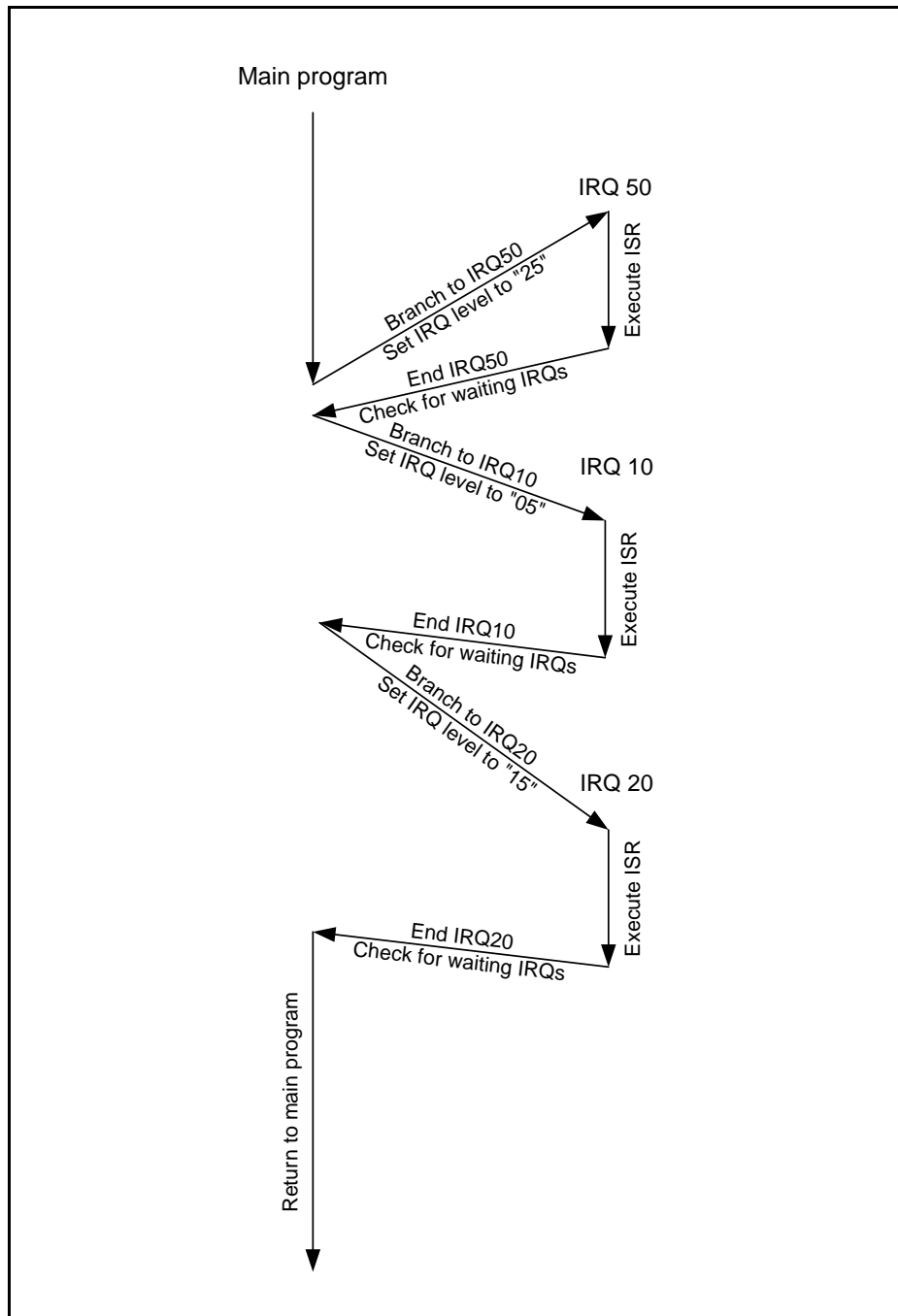


The following paragraphs explain the example shown in Figure 3-1.

When IRQ5 occurred, the interrupt controller issues nIRQ to the CPU. The interrupt controller sets the hold bit corresponding to IRQ5. The CPU holds the current program execution and saves CPU's current execution status. It then reads the vector address of IRQ5 and starts executing IRQ5's ISR. If NMI0 occurs during the previous process, the controller issues nFIQ to the CPU. Since NMIs have a higher priority than IRQs, the CPU stops executing the ISR for IRQ5 and saves CPU's current execution status. It then reads the vector address of NMI0. At this point, interrupt controller sets the hold flag corresponding to NMI0. Then starts executing NMI0's ISR. Before the ISR for NMI0 ends, the ISR clears the peripheral interrupt cause of NMI0. Then also clears NMI0's hold bit. After the ISR for NMI0 ends, the CPU resumes the ISR for IRQ5. Before the ISR for IRQ5 ends, the ISR clears the peripheral interrupt cause of IRQ5. Then also clears IRQ5's hold bit. After the ISR for IRQ5 ends, the CPU returns to the main program.

If the interrupt that is applied here is a software interrupt caused by a write operation to a register from IRCn\_IRQS0 to IRCn\_IRQS15 or IRCn\_NMIS, the ISR needs to reset the software interrupt. This must be done by writing to a register between IRCn\_IRQR0 and IRCn\_IRQR15 or IRCn\_NMIR, rather than by clearing the peripheral interrupt.

Figure 3-2 ISR Processing Flow When There is No Nested IRQ

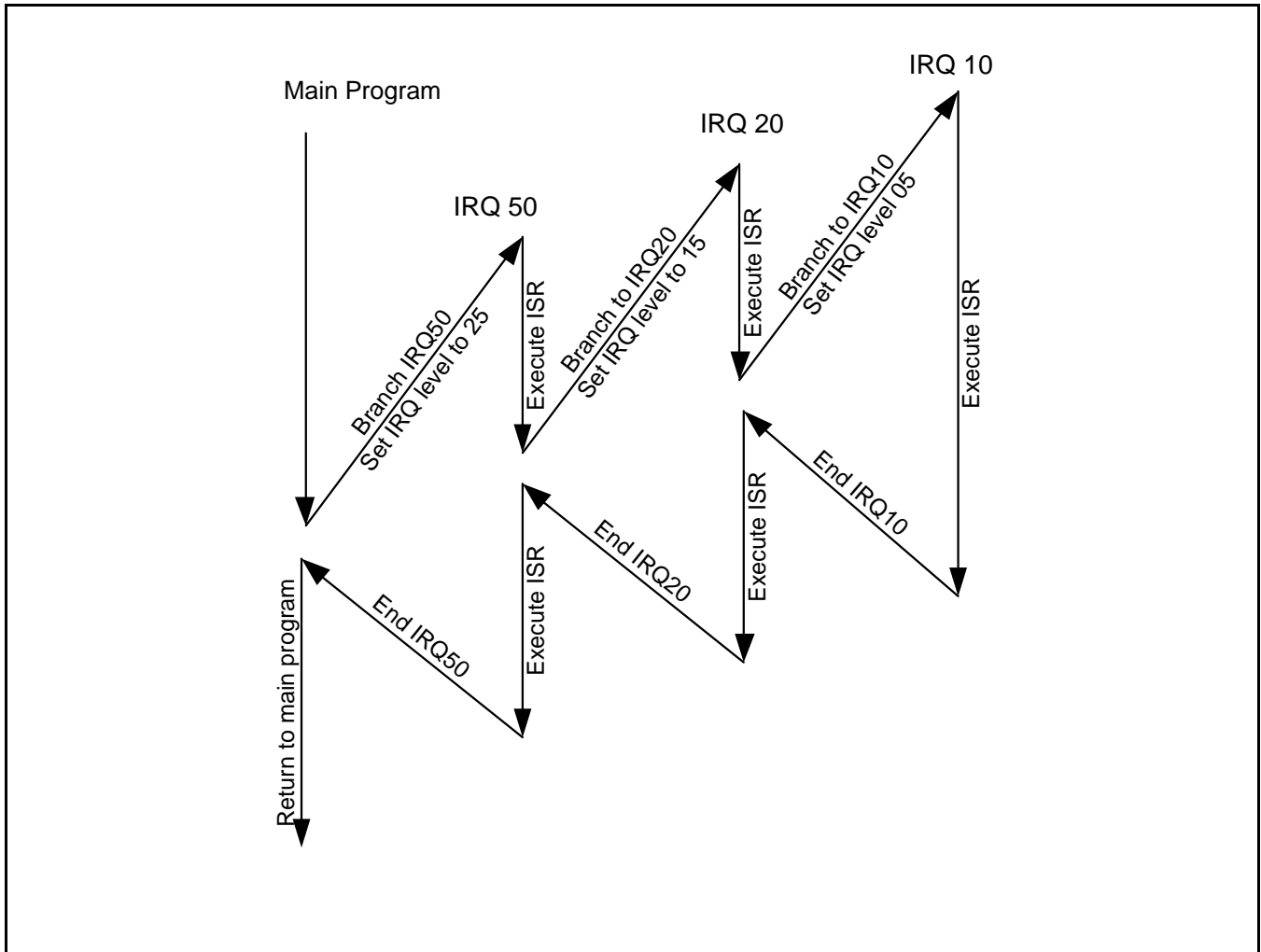


The following paragraph explains the example shown in Figure 3-2. The priority levels of IRQ10, 20, and 50 used in this example are assumed to be "5", "15", and "25", respectively.

First, IRQ50 occurs. Since there are no waiting IRQs, IRQ50 is executed. At this point, "25" can be read from IRCn\_IRQST:IRQPS. During the execution of the ISR for IRQ50, IRQ20 occurs. IRQ20 will wait because IRS for IRQ50 has not ended. Then similarly, IRQ10 occurs. As IRQ20, IRQ10 waits until the ISR for IRQ50 ends. When the ISR for IRQ50 ends, IRQ10 and 20 enter the wait state. Since the priority level of IRQ10 is higher than that of IRQ20, which occurred earlier than IRQ10, the ISR for IRQ10 is executed next. At this point, "5" can be read from IRCn\_IRQST:IRQPS. After the ISR for IRQ10 ends, the ISR for IRQ20 is executed. At this point, "15" can be read from IRCn\_IRQST:IRQPS. After the ISR for IRQ20 ends, there are no other waiting IRQs. Therefore, the CPU returns to the main program.

**b) Interrupt Operation when Particular Software is Used for Nested Processing**

**Figure 3-3 ISR Processing Flow when There are Nested IRQs**



The following paragraphs explain the example shown in Figure 3-3.

As in the case of Figure 3-2, the priority levels of IRQ10, 20, and 50, used in this example, are assumed to be "5", "15", and "25", respectively.

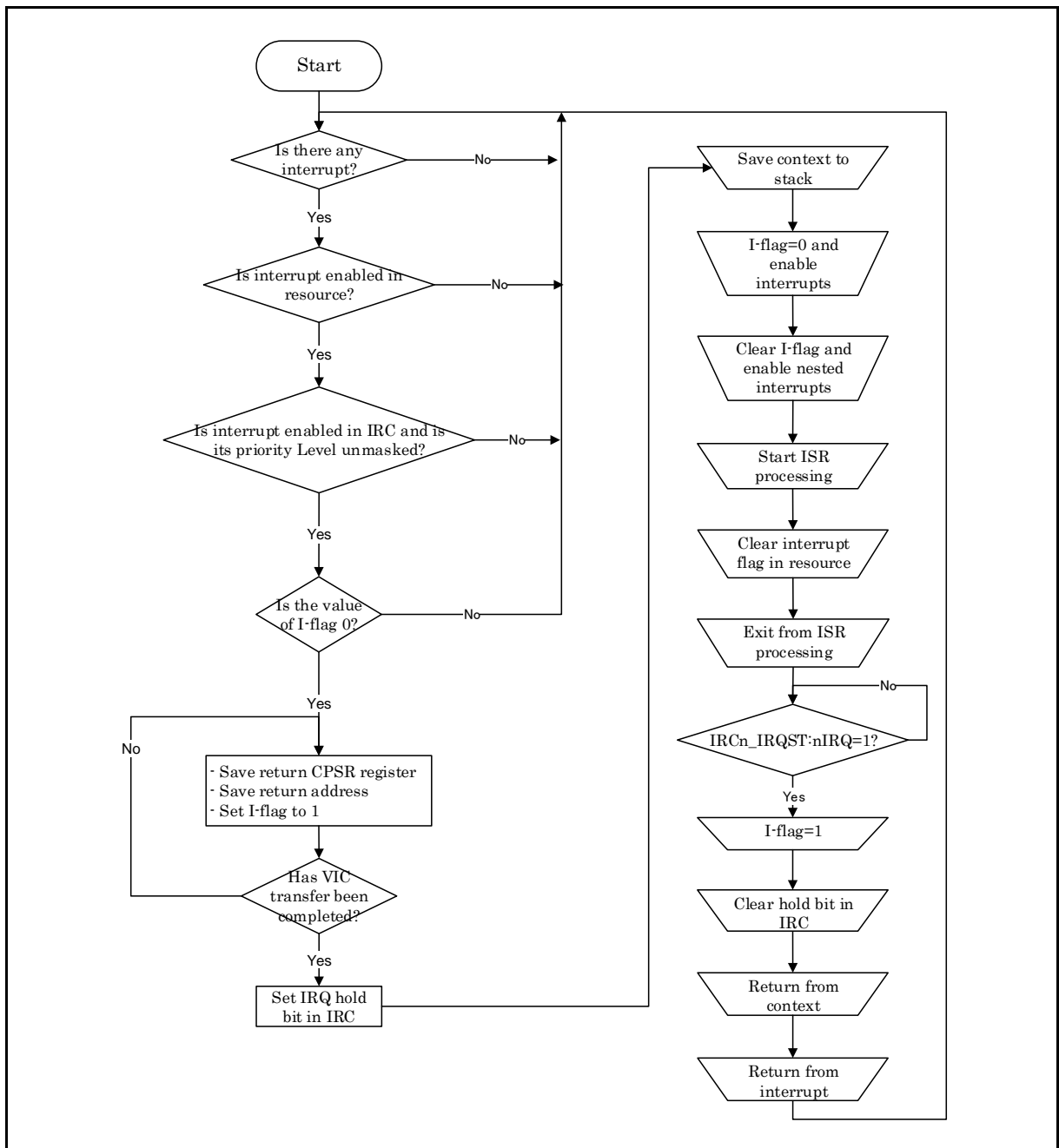
First, IRQ50 occurs. Since there are no waiting IRQs, IRQ50 is executed. At this point, "25" can be read from IRCn\_IRQST:IRQPS. During the execution of the ISR for IRQ50, IRQ20 occurs. The ISR for IRQ50 is stopped and the ISR for IRQ20 is executed. At this point, "15" can be read from IRCn\_IRQST:IRQPS. Then, similarly, IRQ10 occurs and the ISR for IRQ20 is stopped. At this point, "5" can be read from IRCn\_IRQST:IRQPS. The ISR for IRQ10 is executed. After the ISR for IRQ10 ends, the ISR for IRQ20 will resume. After the ISR for IRQ20 ends, the ISR for IRQ50 will resume. After the ISR for IRQ50 ends, there are no other waiting IRQs. Therefore the CPU returns to the main program.

### (3) Example of Software Used to Process Nested IRQ/NMI

#### a) IRQ Processing

Since the ARM architecture does not support nested nIRQ interrupts, applications should be used to handle any nested processing. The ISR should internally save the values of 2 registers (LR\_irq and SPSR\_irq) and registers which values will be changed by ISR process in the system stack and then allow interrupts again. Then, if a new IRQ occurs and it has a higher priority level than that of the ISR that is being executed, it can interrupt the ISR. When the ISR ends, the saved values of the 2 registers (LR\_irq and SPSR\_irq) and registers which values will be changed by ISR process should be restored from the system stack.

Figure 3-4 Nest Supported IRQ Processing Flow



Below is an Example of Nest Supported ISR in Assembler Code.

```
IRQxx_Handler:
SUB          lr, lr, #4
SRSFD        SP!,#0x1f          ; Save LR_irq and SPS_irq to the system stack.
CPS          #0x1f              ; Switch to system mode.
PUSH         {r0-r3, r12}       ; Save the remaining registers to the system stack.
AND          r1, sp, #4         ; Confirm that the stack is aligned to 8 bytes.
SUB          sp, sp, r1
PUSH         {r1, lr}
CPSIE        i                  ; Enable the interrupt.

...
; ISR code that may be interrupted by other interrupts having a higher priority.
; End of ISR
CPSID        i                  ; Disable the interrupt.
POP          {r1, lr}           ; Restore LR_sys and return the system stack to its previous state.
ADD          sp, sp, r1
POP          {r0-r3, r12}       ; Restore the saved registers.
RFEFD        SP!
```

**Notes:**

- *If you use nested IRQs or NMIs (or both), you should calculate the additional system stack size needed according to the depth of the nesting level being used.*
- *All registers which values will be changed by ISR process must be saved to system stack.*





### b) NMI Processing

Fast interrupts are set as NMIs. The NMI flag is placed in the interrupt source module. This differs from IRQs in that there is no enable bit in the resource or interrupt controller.

The NMI entry is processed as follows:

1. The NMI interrupt flag is set by hardware or software and issued to the interrupt controller.
2. The interrupt controller evaluates the priority level. If multiple NMI interrupts are generated, the NMI with the highest NMI priority level is selected. And the NMI is issued to the CPU.
3. The CPSR is copied to SPSR\_fiq, and the return address is set in r14\_fiq. Then, in the CPSR, the mode is set in fiq, and the I-flag and the F-flag are set. (This disables the acceptance of new NMIs.) This processing is all executed by the hardware.
4. The processor reads the register in the interrupt controller that retains the vector address corresponding to this NMI. This read access sets the hold bit for the corresponding NMI. This read operation is executed by the BootROM code (instruction in the FIQ entry).
5. The processing moves to the vector address that was read from the register in the interrupt controller, and then the corresponding ISR is executed.
6. If nested NMIs need to be supported, the F-flag must be cleared. This operation must be performed by software. Before the F-flag is cleared, the processing described in step 7. must be performed.
7. Before the F-flag is cleared, the context must be saved to the stack. This operation must be performed by software.
8. After the F-flag is cleared, another NMI having a higher priority level can interrupt the current NMI by nesting.

If nested NMIs are to be supported, the next processing must be executed using the steps described in 6. to 8., and the context must be saved to the stack. This is an example, and the processing can also be performed by using other assembler instructions.

- SUB (return address subtraction). Subtract 4 from the value of r14\_fiq.
- Execute SRS (save the return state). This instruction saves r14\_fiq and SPSR\_fiq to the stack (the address is specified through r13\_fiq).
- Execute PUSH (save multiple registers). This instruction saves the defined number of registers to the stack that is specified in r13\_fiq.
- Clear the F-flag (enable nested NMIs).

Each time a new NMI is accepted by the CPU, the F-flag is set. As long as the F-flag is not cleared, therefore, the above-mentioned series of saving processes will never be interrupted by other NMIs.

The steps for ending an NMI are as follows:

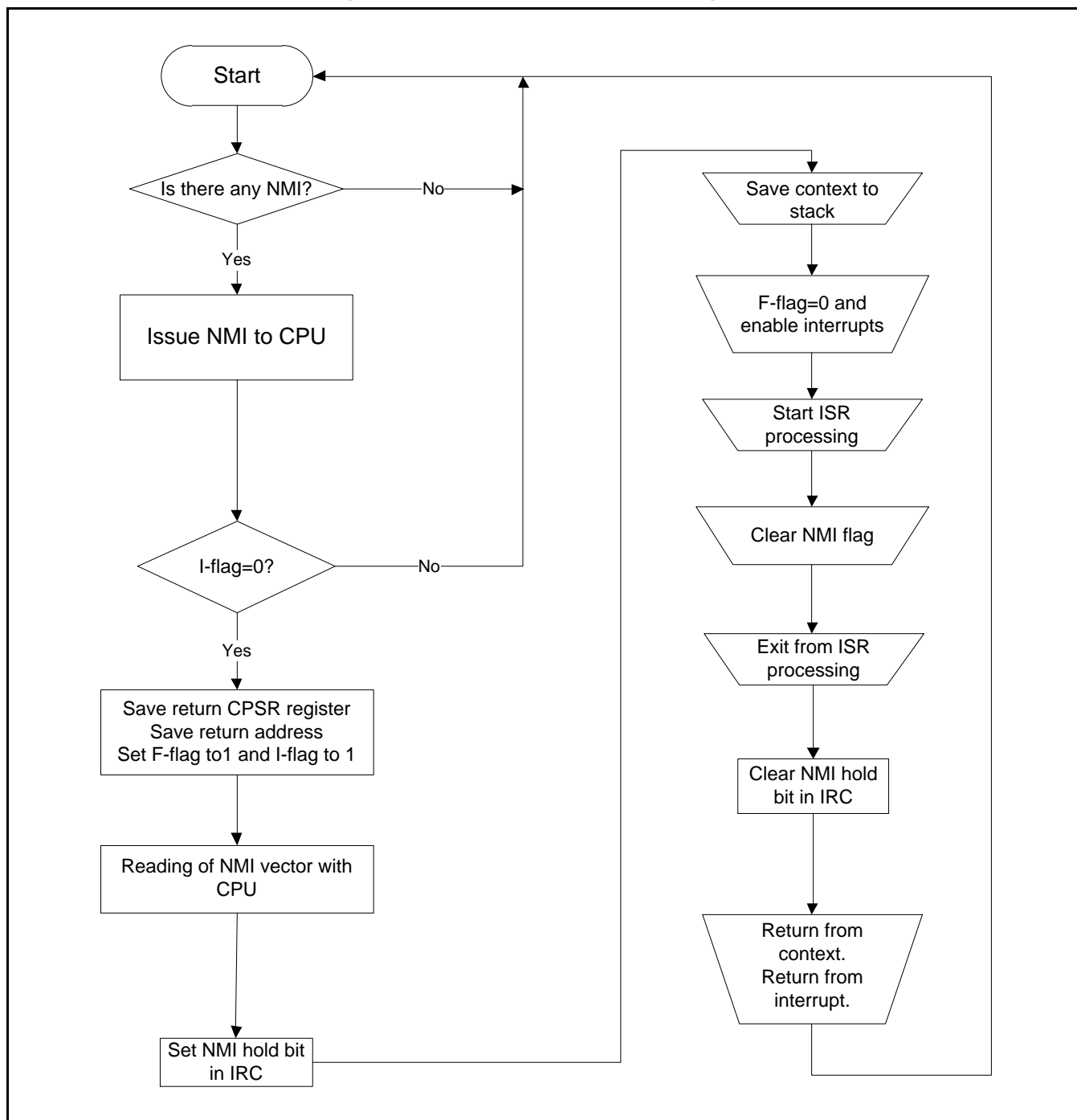
1. In the ISR, the NMI flag that became a factor must be cleared by software. This must be performed in the ISR as soon as possible so that the NMI flag is not received by the interrupt controller.
2. In the ISR, the hold bit in the interrupt controller should be cleared by writing the NMI number to an appropriate register (IRCh\_NMIHC) in the interrupt controller. This must also be performed by the software.
3. After the above processing ends, the context saved to the stack that contains the PC and CPSR is restored to return from the interrupt. This operation must be performed by software. When the CPSR is restored, the system is restored to the state of the register immediately before the generated NMI. This is also true for the interrupt enabled state.

To support nested NMIs, the following processing must be executed from software to restore the context and restore from the interrupt.

- POP (restore multiple registers). This instruction restores the saved user register from the stack.
- RFE (end the exception handling). This instruction restores the saved CPSR and PC from the stack.

The NMI processing flow is shown in Figure 3-5.

**Figure 3-5 Nest Supported NMI Processing Flow**





#### (4) ECC Support and Test

The interrupt controller is equipped with SRAM which has an ECC protection function. Assume that a 1-bit error (correctable) occurs in the ECC. The `IRCN_EEI:EEIS` bit is set and the IRQ is generated for both `IRQVAr` register reading and reading from the IRQ processing stage. If an error of 2 or more bits (uncorrectable) occurs, the `IRCN_EEI:EENS` bit is set and the NMI is generated at `IRQVAr` register reading. Upon reading from the IRQ processing stage, transition is made to the ISR indicated by `IRCN_IRQEEVA`. Therefore, make sure that a valid ISR address is set for `IRCN_IRQEEVA`.

##### a) ECC Test Function

The interrupt controller has a function for generating pseudo ECC errors by corrupting the data read from SRAM. Using this function, you can check the operation of the ECC error handler while developing applications. By setting the ECC error bit registers (`IRCN_EEB0` to `IRCN_EEB2`), you can invert the data read from SRAM by using the bitwise operation. This test function is enabled/disabled using the ECC test register (`IRCN_ET`).

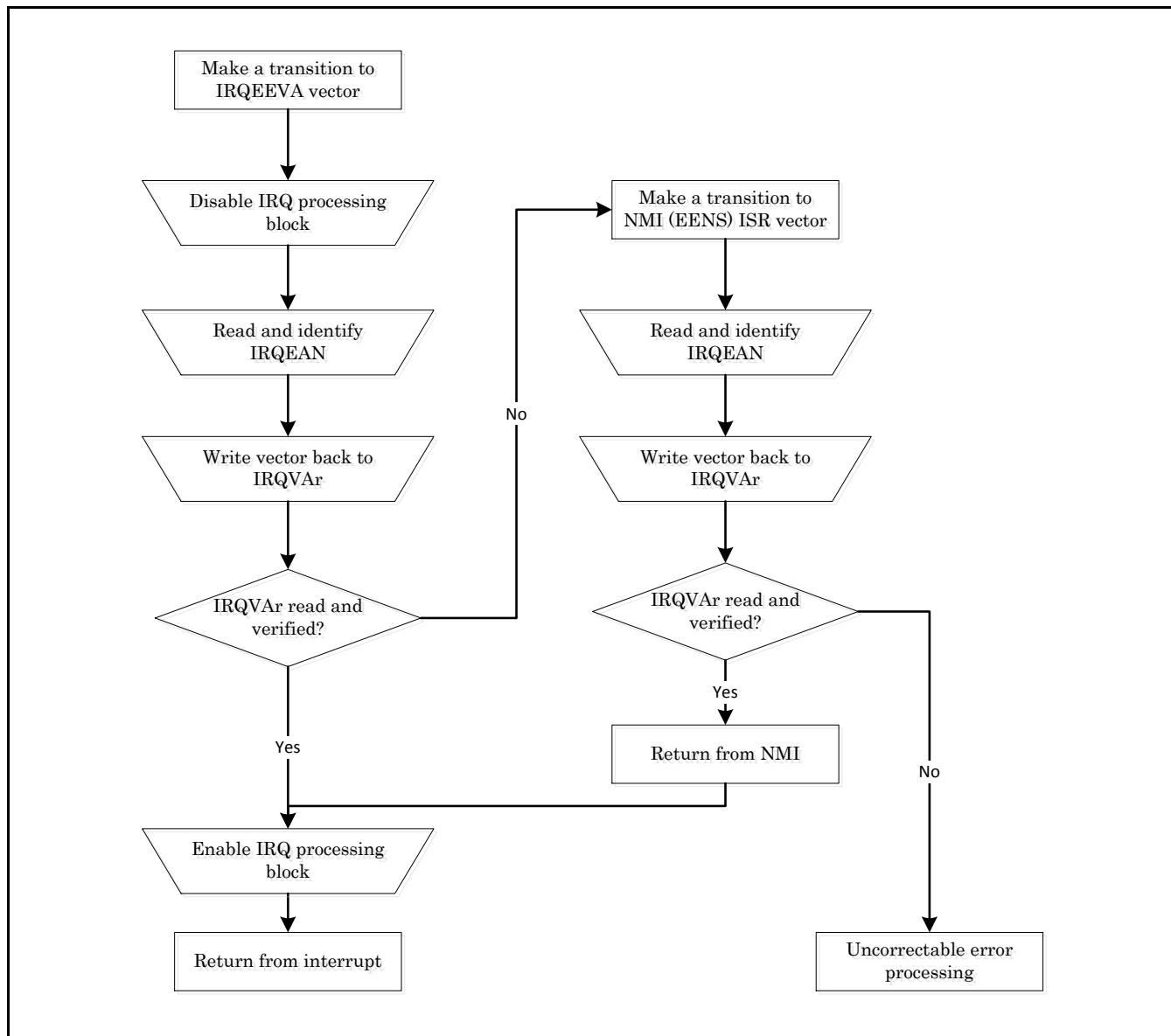
##### b) About `IRCN_IRQEEVA`

In the IRQ processing stage, if an SRAM 2-bit ECC error is detected, a transition is made to the ISR configured in `IRCN_IRQEEVA`. This behavior always occurs upon the occurrence of an error. So, `IRCN_IRQEEVA` must be properly set.

When a transition is made to this ISR, the IRQ hold bit for the erroneous `IRCN_IRQVAr` channel number is not generated. Moreover, this `IRCN_IRQEEVA` interrupt has no hold bit. Therefore, the interrupt factor need not be cleared within this ISR.

The ISR of IRCn\_IRQEEVA needs to perform processing for correcting SRAM errors, using the following procedure.

**Figure 3-6 SRAM 2-bit ECC Error Correction Flow by IRQEEVA**



In this ISR, a software reset can be issued without any error correction, by rewriting to IRCn\_IRQVAr.



### (5) How to Control Multiple Interrupts

The IRC controls the interrupt priority at a timing different from the CPU timing. So, if interrupt enable or disable is performed or if an interrupt level is changed, the software and hardware need to be synchronized according to a given procedure.

#### a) Procedure for Initializing and Setting the Interrupt Function

When initializing all interrupts for the first time, initialize and set the interrupts in the following order.

1. Setting peripheral function interrupts
2. Clear CPU I-FLAG
3. Setting the IRCn\_CSR:IRQEN bit

#### b) Temporarily Enabling/Disabling Interrupts Using IRC\_CSR\_IRQEN with Interrupts Enabled

Before rewriting IRCn\_CSR:IRQEN, verify that the IRCn\_IRQST:nIRQ bit is 1. The following shows an example.

```
while(IRCn_IRQST:nIRQ==0) ; // if 0, repeat reading
IRCn_CSR:IRQEN=0; // safe for changing enable bit
```

#### c) Changing an Interrupt Level Using IRC\_IRQPLM and IRC\_IRQPLx with Interrupts Enabled

Before rewriting IRC\_IRQPLM and IRC\_IRQPLx, verify that the IRC\_IRQST:nIRQ bit is 1. The following shows an example.

```
while(IRCn_IRQST:nIRQ==0) ; // if 0, repeat reading
IRCn_IRQPLM=0x2; // safe for changing interrupt level
```

#### d) Hold Clear Procedure Using IRC\_IRQHC with Interrupts Enabled

Assume that writing to IRCn\_IRQHC is to be performed after the I-FLAG clear instruction is executed within the interrupt handler. Before rewriting IRCn\_IRQHC, verify that the IRCn\_IRQST:nIRQ bit is 1. The following shows an example.

```
while(IRCn_IRQST:nIRQ==0) ; // if 0, repeat reading
IRCn_IRQHC=0x80; // safe for changing interrupt level
```

#### e) Processing at the Head of the IRQ Interrupt Handler

Until interrupt acceptance is enabled with the I-FLAG clear instruction in the interrupt processing handler, directly write the desired values to each of the IRCn\_CSR:IRQEN, IRCn\_IRQPLM, and IRCn\_IRQHC registers, rather than using the procedure described above. Moreover, even if these register values do not need to be changed, write dummy values in one of the IRC registers listed below.

- IRCn\_IRQVAR
- IRCn\_IRQPL0 to 127
- IRCn\_IRQS0 to 15
- IRCn\_IRQR0 to 15
- IRCn\_IRQCES0 to 15
- IRCn\_IRQCEC0 to 15
- IRCn\_IRQCE0 to 15
- IRCn\_IRQHC
- IRCn\_IRQPLM
- IRCn\_CSR

**Notes:**

- *If dummy values are not written to any of the registers within the IRC, the generation of a new IRQ interrupt is suppressed until writing is complete. After writing to any of the registers within the IRC, the acceptance of a new IRQ interrupt is resumed.*
- *If the condition for returning a bus error is written during writing, interrupt acceptance is not released.*

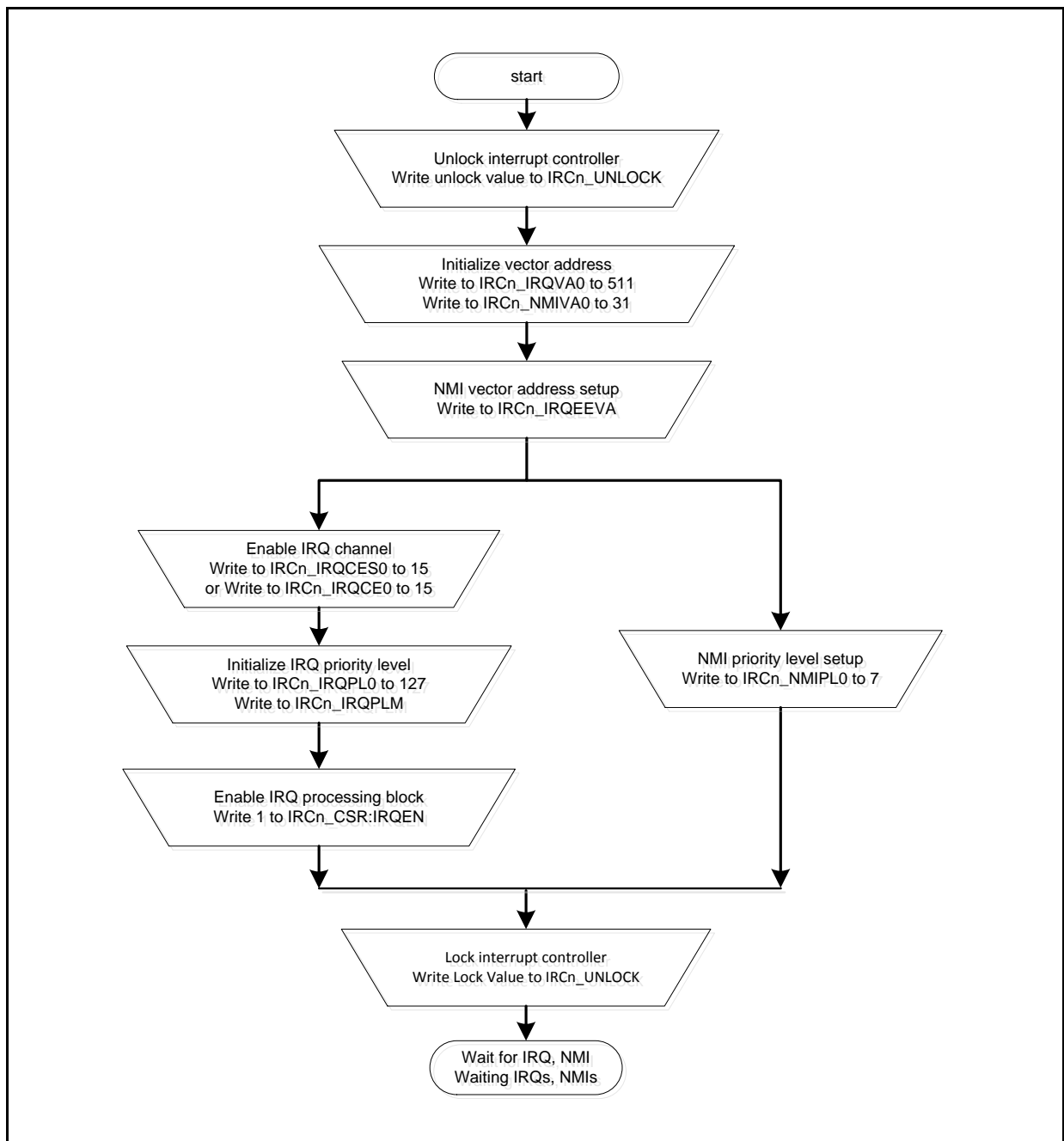


## 4. Setting Procedure Examples

This section provides examples of the setting procedure for the interrupt controller.

### Initial Setting

Figure 4-1 Interrupt Controller Initial Setting Flow



## 5. Registers

This section explains the registers that are used by the interrupt controller.

**Table 5-1 List of Interrupt Controller Registers**

Abbreviated Register Name	Register Name	See
IRCN_NMIVAS	IRC NMI Vector Address Status Register	5.1
IRCN_NMIST	IRC NMI Status Register	5.2
IRCN_IRQVAS	IRC IRQ Vector Address Status Register	5.3
IRCN_IRQST	IRC IRQ Status Register	5.4
IRCN_NMIVAm	IRC NMI Vector Address Register	5.5
IRCN_IRQVAr	IRC IRQ Vector Address Register	5.6
IRCN_NMIPL0	IRC NMI Priority Level Register	5.7
IRCN_NMIPL1 to IRCN_NMIPL7	IRC NMI Priority Level Register	5.8
IRCN_IRQPL0 to IRCN_IRQPL127	IRC IRQ Priority Level Register	5.9
IRCN_NMIS	IRC NMI Software Interrupt Set Register	5.10
IRCN_NMIR	IRC NMI Software Interrupt Reset Register	5.11
IRCN_NMISIS	IRC NMI Software Interrupt Status Register	5.12
IRCN_IRQS0 to IRCN_IRQS15	IRC IRQ Software Interrupt Set Register	5.13
IRCN_IRQR0 to IRCN_IRQR15	IRC IRQ Software Interrupt Reset Register	5.14
IRCN_IRQSIS0 to IRCN_IRQSIS15	IRC IRQ Software Interrupt Status Register	5.15
IRCN_IRQCES0 to IRCN_IRQCES15	IRC IRQ Channel Enable set Register	5.16
IRCN_IRQCEC0 to IRCN_IRQCEC15	IRC IRQ Channel Enable Clear Register	5.17
IRCN_IRQCE0 to IRCN_IRQCE15	IRC IRQ Channel Enable Setting Register	5.18
IRCN_NMIHC	IRC NMI Hold Clear Register	5.19
IRCN_NMIHS	IRC NMI Hold Status Register	5.20
IRCN_IRQHC	IRC IRQ Hold Clear Register	5.21
IRCN_IRQHS0 to IRCN_IRQHS15	IRC IRQ Hold Status Register	5.22
IRCN_IRQPLM	IRC IRQ Priority Level Mask Register	5.23
IRCN_CSR	IRC Control/Status Register	5.24
IRCN_NMIRS	IRC NMI RAW Status Register	5.25
IRCN_NMIPS	IRC NMI Preprocessed Status Register	5.26
IRCN_IRQRS0 to IRCN_IRQRS15	IRC IRQ RAW Status Register	5.27
IRCN_IRQPS0 to IRCN_IRQPS15	IRC IRQ Preprocessed Status Register	5.28
IRCN_UNLOCK	IRC Unlock Register	5.29
IRCN_EEI	IRC ECC Error Interrupt Register	5.30
IRCN_EAN	IRC ECC Address Number Register	5.31
IRCN_ET	IRC ECC Test Register	5.32
IRCN_EEB0 to IRCN_EEB1	IRC ECC Error Bit Register	5.33
IRCN_EEB2	IRC ECC Error Bit Register	5.34
IRCN_NMIVASBR	IRC NMI Vector Address Status Register	5.35
IRCN_NMIVASBR	IRC NMI Vector Address Status Mirror Register	5.36
IRCN_IRQEEVA	IRC ECC Error Vector Address Register	5.37





## 5.1. IRC NMI Vector Address Status Register (IRCn\_NMIVAS)

This register indicates the vector address status of the NMI that the CPU accepted most recently.

BIT_OFFSET	31-0
BIT_NAME	NMIVAS
ACCESS_TYPE	R,WX
PROT_TYPE	RP
INITIAL_VALUE	00000000_00000000_00000000_00000000

### [bit31:0] NMIVAS[31:0]: NMI vector address status bits

These bits indicate the vector address status of the NMI that the CPU accepted most recently.

**Note:**

- By reading the IRC\_NMIVASBR by the CPU which accepted the nFIQ, the interrupt controller will determine that the interrupt has been accepted and then update the register.

## 5.2. IRC NMI Status Register (IRCn\_NMIST)

This register indicates the priority level and channel number status of the NMI that the CPU accepted most recently.

BIT_OFFSET	31-16
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved				NMIPS			
ACCESS_TYPE	R0,WX				R,WX			
PROT_TYPE	-							
INITIAL_VALUE	0000				1111			

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved		NMISN					
ACCESS_TYPE	R0,WX		R,WX					
PROT_TYPE	-							
INITIAL_VALUE	00		100000					

**[bit31:12] Reserved: Reserved bits**

**[bit11:8] NMIPS: NMI priority status bits**

These bits indicate the priority status (0 to 15) of the NMI that the CPU accepted most recently.

**[bit7:6] Reserved: Reserved bits**

**[bit5:0] NMISN: NMI channel number bits**

These bits indicate the channel number status (0 to 31) of the NMI that the CPU accepted most recently. The initial value indicates that no interrupt has been accepted; bit5 will never be "1" unless it is in the initial state.

**Note:**

- By reading the IRC\_NMIVASBR by the CPU to which the nFIQ is applied, the interrupt controller will determine that the interrupt has been accepted and then update the register.



### 5.3. IRC IRQ Vector Address Status Register (IRCn\_IRQVAS)

This register indicates the vector address status of the IRQ that the CPU accepted most recently.

BIT_OFFSET	31-0
BIT_NAME	IRQVAS
ACCESS_TYPE	R,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000_00000000

#### [bit31:0] IRQVAS[31:0]: IRQ vector address status bits

These bits indicate the vector address status of the IRQ that the CPU accepted most recently.

**Notes:**

- By receiving acknowledgment of the VIC port from the CPU which accepted the nIRQ, the interrupt controller will determine that the interrupt has been accepted and then update the register.
- If a 2-bit ECC error is detected in the IRQ processing block and a branch is made to the IRQEEVA handler, the value of the erroneous IRQ is read as an IRQVAS value.

## 5.4. IRC IRQ Status Register (IRCN\_IRQST)

This register indicates the priority level and channel number status of the IRQ that the CPU accepted most recently. This register also indicates the suppress status of newly accepted interrupts.

BITS_OFFSET	31	30	29	28	27	26	25	24
BITS_NAME	Reserved							nIRQ
ACCESS_TYPE	R0,WX							R,WX
PROT_TYPE	-							
INITIAL_VALUE	0000000							1

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved			IRQPS				
ACCESS_TYPE	R0,WX			R,WX				
PROT_TYPE	-							
INITIAL_VALUE	000			11111				

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved						IRQSN[9:8]	
ACCESS_TYPE	R0,WX						R,WX	
PROT_TYPE	-							
INITIAL_VALUE	000000						10	

BITS_OFFSET	7	6	5	4	3	2	1	0
BITS_NAME	IRQSN[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

**[bit31:25] Reserved: Reserved bits**

**[bit24] nIRQ: IRQ interrupt status bit**

If 0 is read from this bit, the IRC is either generating the IRQ for the CPU or IRQ generation to CPU is stopped by IRC. Once this bit is read, the IRC will not accept any subsequent interrupts. Moreover, if data is written to any one of the IRC registers listed below, the stopped interrupt acceptance is released.

- IRCn\_IRQVar
- IRCn\_IRQPL0 to 127
- IRCn\_IRQS0 to 15
- IRCn\_IRQR0 to 15
- IRCn\_IRQCES0 to 15
- IRCn\_IRQCEC0 to 15
- IRCn\_IRQCE0 to 15
- IRCn\_IRQHC
- IRCn\_IRQPLM
- IRCn\_CSR



**Note:**

- *If the condition for returning a bus error is written during writing, no interrupt acceptance is released.*

**[bit23:21] Reserved: Reserved bits**

**[bit20:16] IRQPS: IRQ priority status bits**

These bits indicate the priority status (0 to 31) of the IRQ that the CPU accepted most recently.

**[bit15:10] Reserved: Reserved bits**

**[bit9:0] IRQSN: IRQ channel number bits**

These bits indicate the channel number status (0 to 511) of the IRQ that the CPU accepted most recently. The initial value means that no interrupt has been accepted; bit9 will never be "1" unless it is in the initial state.

**Notes:**

- *By receiving the acknowledgment of VIC port from the CPU to which the nIRQ is applied, the interrupt controller will determine that the interrupt has been accepted and will then update the register.*
- *If a 2-bit ECC error is detected in the IRQ processing block and a branch is made to the IRQEEVA handler, the value of the erroneous IRQ is read as the IRQSN and IRQPS register values.*

## 5.5. IRC NMI Vector Address Register (IRCn\_NMIVAm)

This register indicates the 32-bit vector address of individual NMI channels. Before the corresponding NMI peripheral is enabled, the vector address should be initialized by software. The same type of register is installed for each NMI channel. The "m" in the abbreviated register name corresponds to the NMI channel number, m (0 to 31).

BIT_OFFSET	31-8
BIT_NAME	NMIVA[31:8]
ACCESS_TYPE	R/W
PROT_TYPE	WPS
INITIAL_VALUE	XXXXXXXX_XXXXXXXX_XXXXXXXX

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	NMIVA[7:2]						NMIVA[1:0]	
ACCESS_TYPE	R/W						R0,WX	
PROT_TYPE	WPS							
INITIAL_VALUE	XXXXXX						00	

### [bit31:0] NMIVA[31:0]: NMI vector address bits

These bits set the vector address of NMIm.

#### Notes:

- Since all vector addresses are 32 bits long, the lowest 2 bits are fixed to "00".
- The bit configurations of registers IRCn\_NMIVA0 to IRCn\_NMIVA31 are all the same.



## 5.6. IRC IRQ Vector Address Register (IRCn\_IRQVAr)

This register indicates the 32-bit vector address of individual IRQ channels. Before the corresponding IRQ channel is enabled, the vector address should be initialized by software. The same type of register is installed for each IRQ channel. The "r" in the abbreviated register name corresponds to the IRQ channel number, r (0 to 511). This register is on SRAM of the interrupt controller.

BIT_OFFSET	31-8
BIT_NAME	IRQVA[31:8]
ACCESS_TYPE	R/W
PROT_TYPE	WPS
INITIAL_VALUE	XXXXXXXX_XXXXXXXX_XXXXXXXX

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	IRQVA[7:2]						IRQVA[1:0]	
ACCESS_TYPE	R/W						R0,WX	
PROT_TYPE	WPS						-	
INITIAL_VALUE	XXXXXX						00	

### [bit31:0] IRQVA[31:0]: IRQ vector address bits

These bits set the vector address of IRQr.

#### Notes:

- Before this register is read, the vector address should be initialized. For initialization, use either 32-bit or 64-bit access.
- Since all vector addresses are 32 bits long, the lowest 2 bits are fixed to "00".
- The bit configurations of registers IRCn\_IRQVA0 to IRCn\_IRQVA511 are all the same.
- This register consists of SRAM having the ECC protection function. If an ECC Error occurs, overwrite the register with the correct values.

## 5.7. IRC NMI Priority Level Register (IRCn\_NMIPL0)

This register indicates the priority level of NMI0. It also sets the priority level (0 to 15) of each of channels NMI1 to NMI3. For each channel, a different value can be set. The smaller the setting value, the higher the priority. The initial value for channels other than channel 0 is "15", which indicates the lowest level. For channel 0 only, the value is fixed to "0", which is the highest level.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved				NMIPL3			
ACCESS_TYPE	R0,WX				R/W			
PROT_TYPE	WPS							
INITIAL_VALUE	0000				1111			

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved				NMIPL2			
ACCESS_TYPE	R0,WX				R/W			
PROT_TYPE	WPS							
INITIAL_VALUE	0000				1111			

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved				NMIPL1			
ACCESS_TYPE	R0,WX				R/W			
PROT_TYPE	WPS							
INITIAL_VALUE	0000				1111			

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved				NMIPL0			
ACCESS_TYPE	R0,WX				R0,WX			
PROT_TYPE	WPS							
INITIAL_VALUE	0000				0000			

**[bit31:28] Reserved: Reserved bits**

**[bit27:24] NMIPL3: NMI3 priority level bits**

These bits set the priority level of NMI3.

**[bit23:20] Reserved: Reserved bits**

**[bit19:16] NMIPL2: NMI2 priority level bits**

These bits set the priority level of NMI2.

**[bit15:12] Reserved: Reserved bits**

**[bit11:8] NMIPL1: NMI1 priority level bits**

These bits set the priority level of NMI1.





**[bit7:4] Reserved: Reserved bits**

**[bit3:0] NMIPLO: NMI0 priority level bits**

These bits indicate the priority level of NMI0. The priority level of NMI0 is fixed to "0".

## 5.8. IRC NMI Priority Level Register (IRCn\_NMIPL1 to IRCn\_NMIPL7)

These registers set the priority level (0 to 15) of each NMI channel. For each channel, a different value can be set. The smaller the setting value, the higher the priority. The initial value is "15", which is the lowest level. The 7 registers are of the same type, and each is identified by the number at the end of the abbreviated register name. For each register, 4 channels are assigned, so that 32 channels are supported by the 8 registers including IRCn\_NMIPL0.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved				NMIPL7			
ACCESS_TYPE	R0,WX				R/W			
PROT_TYPE	WPS							
INITIAL_VALUE	0000				1111			

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved				NMIPL6			
ACCESS_TYPE	R0,WX				R/W			
PROT_TYPE	WPS							
INITIAL_VALUE	0000				1111			

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved				NMIPL5			
ACCESS_TYPE	R0,WX				R/W			
PROT_TYPE	WPS							
INITIAL_VALUE	0000				1111			

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved				NMIPL4			
ACCESS_TYPE	R0,WX				R/W			
PROT_TYPE	WPS							
INITIAL_VALUE	0000				1111			

**[bit31:28] Reserved: Reserved bits**

**[bit27:24] NMIPL7[3:0]: NMI7 priority level bits**

These bits set the priority level of NMI7.

**[bit23:20] Reserved: Reserved bits**

**[bit19:16] NMIPL6[3:0]: NMI6 priority level bits**

These bits set the priority level of NMI6.

**[bit15:12] Reserved: Reserved bits**



**[bit11:8] NMIP5[3:0]: NMI5 priority level bits**

These bits set the priority level of NMI5.

**[bit7:4] Reserved: Reserved bits**

**[bit3:0] NMIP4[3:0]: NMI4 priority level bits**

These bits set the priority level of NMI4.

**Notes:**

- The register bit configuration described in this section is for `IRCN_NMIP1`, which is representative of those registers that are all the same. The bit configurations of registers `IRCN_NMIP1` to `IRCN_NMIP7` are all the same. The description of `IRCN_NMIP0` is separated from others because `NMI0` is handled differently from other channels.
- `NMI4` to `NMI7` are assigned to `IRCN_NMIP1`, and the subsequent channels are assigned in the same way until `NMI28` to `NMI31` are assigned to `IRCN_NMIP7`.
- The "m" in bit field name `NMIPm` corresponds to the NMI channel number, m (0 to 31).

## 5.9. IRC IRQ Priority Level Register (IRCn\_IRQPL0 to IRCn\_IRQPL127)

These registers set the priority level (0 to 31) of each IRQ channel. For each channel, a different value can be set. The smaller the setting value, the higher the priority. The initial value is "31", which is the lowest level. The 128 registers are all of the same type, and each is identified by the number at the end of the abbreviated register name. For each register, 4 channels are assigned, so that 512 channels are supported by the 128 registers.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved			IRQPL3				
ACCESS_TYPE	R0,WX			R/W				
PROT_TYPE	WPS							
INITIAL_VALUE	000			11111				

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved			IRQPL2				
ACCESS_TYPE	R0,WX			R/W				
PROT_TYPE	WPS							
INITIAL_VALUE	000			11111				

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved			IRQPL1				
ACCESS_TYPE	R0,WX			R/W				
PROT_TYPE	WPS							
INITIAL_VALUE	000			11111				

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved			IRQPL0				
ACCESS_TYPE	R0,WX			R/W				
PROT_TYPE	WPS							
INITIAL_VALUE	000			11111				

**[bit31:29] Reserved: Reserved bits**

**[bit28:24] IRQPL3[4:0]: IRQ3 priority level bits**

These bits set the priority level of IRQ3.

**[bit23:21] Reserved: Reserved bits**

**[bit20:16] IRQPL2[4:0]: IRQ2 priority level bits**

These bits set the priority level of IRQ2.

**[bit15:13] Reserved: Reserved bits**



**[bit12:8] IRQPL1[4:0]: IRQ1 priority level bits**

These bits set the priority level of IRQ1.

**[bit7:5] Reserved: Reserved bits**

**[bit4:0] IRQPL0[4:0]: IRQ0 priority level bits**

These bits set the priority level of IRQ0.

**Notes:**

- In case of writing to this register, perform it in any of the following states (= IRC is stopping).
  - Until writing the target register from reading that *nIRQ* bit of *IRCN\_IRQST* register is "1". Refer to the description of *nIRQ* bit for the target register.
  - The state that *IRQEN* bit of *IRCN\_CSR* register is "0" (immediately after reset, etc).
- The register bit configuration described in this section is for *IRCN\_IRQPL0*, which is representative of those registers that are all the same. The bit configurations of registers *IRCN\_IRQPL0* to *IRCN\_IRQPL127* are all the same.
- *IRQ0* to *IRQ3* are assigned to *IRCN\_IRQPL0*, and the subsequent channels are assigned in the same way until *IRQ508* to *IRQ511* are assigned to *IRCN\_IRQPL127*.
- The "r" in bit field name *IRQPLr* corresponds to the *IRQ* channel number, *r* (0 to 511).

## 5.10. IRC NMI Software Interrupt Set Register (IRCn\_NMIS)

This register sets a software interrupt for individual NMI channels. For each channel, 1 bit is assigned, and the bit location corresponds to the NMI channel number, m (0 to 31).

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	NMIS31	NMIS30	NMIS29	NMIS28	NMIS27	NMIS26	NMIS25	NMIS24
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W
PROT_TYPE	WPS							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	NMIS23	NMIS22	NMIS21	NMIS20	NMIS19	NMIS18	NMIS17	NMIS16
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W
PROT_TYPE	WPS							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	NMIS15	NMIS14	NMIS13	NMIS12	NMIS11	NMIS10	NMIS9	NMIS8
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W
PROT_TYPE	WPS							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	NMIS7	NMIS6	NMIS5	NMIS4	NMIS3	NMIS2	NMIS1	NMIS0
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W
PROT_TYPE	WPS							
INITIAL_VALUE	0	0	0	0	0	0	0	0

### [bit31] to [bit0] NMISm: NMIm software interrupt set bits

These bits set a software interrupt for the corresponding NMIm.

Value	Description
0	Invalid (no effect on operation)
1	Set a software interrupt for the corresponding NMIm.



## 5.11. IRC NMI Software Interrupt Reset Register (IRCn\_NMIR)

This register resets the software interrupts for individual NMI channels. For each channel, 1 bit is assigned, and the bit location corresponds to the NMI channel number, m (0 to 31).

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	NMIR31	NMIR30	NMIR29	NMIR28	NMIR27	NMIR26	NMIR25	NMIR24
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	NMIR23	NMIR22	NMIR21	NMIR20	NMIR19	NMIR18	NMIR17	NMIR16
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	NMIR15	NMIR14	NMIR13	NMIR12	NMIR11	NMIR10	NMIR9	NMIR8
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	NMIR7	NMIR6	NMIR5	NMIR4	NMIR3	NMIR2	NMIR1	NMIR0
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

### [bit31] to [bit0] NMIRm: NMIRm software interrupt reset bits

These bits reset a software interrupt for the corresponding NMIRm.

Value	Description
0	Invalid (no effect on operation)
1	Reset a software interrupt for the corresponding NMIRm.

## 5.12. IRC NMI Software Interrupt Status Register (IRCn\_NMISIS)

This register indicates the software interrupt status of individual NMI channels. For each channel, 1 bit is assigned, and the bit location corresponds to the NMI channel number, m (0 to 31).

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	NMISIS31	NMISIS30	NMISIS29	NMISIS28	NMISIS27	NMISIS26	NMISIS25	NMISIS24
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	NMISIS23	NMISIS22	NMISIS21	NMISIS20	NMISIS19	NMISIS18	NMISIS17	NMISIS16
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	NMISIS15	NMISIS14	NMISIS13	NMISIS12	NMISIS11	NMISIS10	NMISIS9	NMISIS8
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	NMISIS7	NMISIS6	NMISIS5	NMISIS4	NMISIS3	NMISIS2	NMISIS1	NMISIS0
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

### [bit31] to [bit0] NMISISm: NMIm software interrupt status bits

These bits indicate the software interrupt status of the corresponding NMIm.

Value	Description
0	Software interrupt is not set.
1	Software interrupt is set.





### 5.13. IRC IRQ Software Interrupt Set Register (IRCN\_IRQS0 to IRCN\_IRQS15)

These registers set a software interrupt for individual IRQ channels. For each channel, 1 bit is assigned. The 16 registers are all of the same type, and each is identified by the number at the end of the abbreviated register name. For each register, 32 channels are assigned, so that 512 channels are supported by the 16 registers.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	IRQS31	IRQS30	IRQS29	IRQS28	IRQS27	IRQS26	IRQS25	IRQS24
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W
PROT_TYPE	WPS							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	IRQS23	IRQS22	IRQS21	IRQS20	IRQS19	IRQS18	IRQS17	IRQS16
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W
PROT_TYPE	WPS							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	IRQS15	IRQS14	IRQS13	IRQS12	IRQS11	IRQS10	IRQS9	IRQS8
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W
PROT_TYPE	WPS							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	IRQS7	IRQS6	IRQS5	IRQS4	IRQS3	IRQS2	IRQS1	IRQS0
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W
PROT_TYPE	WPS							
INITIAL_VALUE	0	0	0	0	0	0	0	0

#### [bit31] to [bit0] IRQSr: IRQr software interrupt set bits

These bits set a software interrupt for the corresponding IRQr.

Value	Description
0	Invalid (no effect on operation)
1	Set a software interrupt for the corresponding IRQr.

#### Notes:

- The register bit configuration described in this section is for IRCN\_IRQS0, which is representative of those registers that are all the same. The bit configurations of registers IRCN\_IRQS0 to IRCN\_IRQS15 are all the same.
- IRQ0 to IRQ31 are assigned to IRCN\_IRQS0, and the subsequent channels are assigned in the same way until IRQ480 to IRQ511 are assigned to IRCN\_IRQS15.
- The "r" in bit field name IRQSr corresponds to the IRQ channel number, r (0 to 511).

## 5.14. IRC IRQ Software Interrupt Reset Register (IRCN\_IRQR0 to IRCN\_IRQR15)

These registers reset a software interrupt for individual IRQ channels. For each channel, 1 bit is assigned. The 16 registers are all of the same type, and each is identified by the number at the end of the abbreviated register name. For each register, 32 channels are assigned, so that 512 channels are supported by the 16 registers.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	IRQR31	IRQR30	IRQR29	IRQR28	IRQR27	IRQR26	IRQR25	IRQR24
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	IRQR23	IRQR22	IRQR21	IRQR20	IRQR19	IRQR18	IRQR17	IRQR16
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	IRQR15	IRQR14	IRQR13	IRQR12	IRQR11	IRQR10	IRQR9	IRQR8
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	IRQR7	IRQR6	IRQR5	IRQR4	IRQR3	IRQR2	IRQR1	IRQR0
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

### [bit31] to [bit0] IRQRr: IRQR software interrupt reset bits

These bits reset a software interrupt for the corresponding IRQR.

Value	Description
0	Invalid (no effect on operation)
1	Reset a software interrupt for the corresponding IRQR.

#### Notes:

- The register bit configuration described in this section is for IRCN\_IRQR0, which is representative of those registers that are all the same. The bit configurations of registers IRCN\_IRQR0 to IRCN\_IRQR15 are all the same.
- IRQ0 to IRQ31 are assigned to IRCN\_IRQR0, and the subsequent channels are assigned in the same way until IRQ480 to IRQ511 are assigned to IRCN\_IRQR15.
- The "r" in bit field name IRQRr corresponds to the IRQ channel number, r (0 to 511).



## 5.15. IRC IRQ Software Interrupt Status Register (IRCN\_IRQSI0 to IRCN\_IRQSI15)

These registers indicate the software interrupt status of the individual IRQ channels. For each channel, 1 bit is assigned. The 16 registers are all of the same type, and each is identified by the number at the end of the abbreviated register name. For each register, 32 channels are assigned, so that 512 channels are supported by the 16 registers.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	IRQSI31	IRQSI30	IRQSI29	IRQSI28	IRQSI27	IRQSI26	IRQSI25	IRQSI24
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	IRQSI23	IRQSI22	IRQSI21	IRQSI20	IRQSI19	IRQSI18	IRQSI17	IRQSI16
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	IRQSI15	IRQSI14	IRQSI13	IRQSI12	IRQSI11	IRQSI10	IRQSI9	IRQSI8
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	IRQSI7	IRQSI6	IRQSI5	IRQSI4	IRQSI3	IRQSI2	IRQSI1	IRQSI0
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

### [bit31] to [bit0] IRQSIr: IRQr software interrupt status bits

These bits indicate the software interrupt status of the corresponding IRQr.

Value	Description
0	Software interrupt is not set.
1	Software interrupt is set.

#### Notes:

- The register bit configuration described in this section is for IRCN\_IRQSI0, which is representative of those registers that are all the same. The bit configurations of registers IRCN\_IRQSI0 to IRCN\_IRQSI15 are all the same.
- IRQ0 to IRQ31 are assigned to IRCN\_IRQSI0, and the subsequent channels are assigned in the same way until IRQ480 to IRQ511 are assigned to IRCN\_IRQSI15.
- The "r" in bit field name IRQSIr corresponds to the IRQ channel number, r (0 to 511).

## 5.16. IRC IRQ Channel Enable Set Register (IRCN\_IRQCES0 to IRCN\_IRQCES15)

These registers set enable for individual IRQ channels. For each channel, 1 bit is assigned. The 16 registers are all of the same type, and each is identified by the number at the end of the abbreviated register name. For each register, 32 channels are assigned, so that 512 channels are supported by the 16 registers.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	IRQCES31	IRQCES30	IRQCES29	IRQCES28	IRQCES27	IRQCES26	IRQCES25	IRQCES24
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W
PROT_TYPE	WPS							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	IRQCES23	IRQCES22	IRQCES21	IRQCES20	IRQCES19	IRQCES18	IRQCES17	IRQCES16
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W
PROT_TYPE	WPS							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	IRQCES15	IRQCES14	IRQCES13	IRQCES12	IRQCES11	IRQCES10	IRQCES9	IRQCES8
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W
PROT_TYPE	WPS							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	IRQCES7	IRQCES6	IRQCES5	IRQCES4	IRQCES3	IRQCES2	IRQCES1	IRQCES0
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W
PROT_TYPE	WPS							
INITIAL_VALUE	0	0	0	0	0	0	0	0

### [bit31] to [bit0] IRQCESr: IRQr channel enable set bits

These bits set channel enable for the corresponding IRQr.

Value	Description
0	Invalid (no effect on operation)
1	Set channel enable for IRQr.

#### Notes:

- The register bit configuration described in this section is for IRCN\_IRQCES0, which is representative of those registers that are all the same. The bit configurations of registers IRCN\_IRQCES0 to IRCN\_IRQCES15 are all the same.
- IRQ0 to IRQ31 are assigned to IRCN\_IRQCES0, and the subsequent channels are assigned in the same way until IRQ480 to IRQ511 are assigned to IRCN\_IRQCES15.
- The "r" in bit field name IRQCESr corresponds to the IRQ channel number, r (0 to 511).



### 5.17. IRC IRQ Channel Enable Clear Register (IRCN\_IRQCEC0 to IRCN\_IRQCEC15)

These registers clear enable for individual IRQ channels. For each channel, 1 bit is assigned. The 16 registers are all of the same type, and each is identified by the number at the end of the abbreviated register name. For each register, 32 channels are assigned, so that 512 channels are supported by the 16 registers.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	IRQCEC31	IRQCEC30	IRQCEC29	IRQCEC28	IRQCEC27	IRQCEC26	IRQCEC25	IRQCEC24
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W
PROT_TYPE	WPS							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	IRQCEC23	IRQCEC22	IRQCEC21	IRQCEC20	IRQCEC19	IRQCEC18	IRQCEC17	IRQCEC16
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W
PROT_TYPE	WPS							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	IRQCEC15	IRQCEC14	IRQCEC13	IRQCEC12	IRQCEC11	IRQCEC10	IRQCEC9	IRQCEC8
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W
PROT_TYPE	WPS							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	IRQCEC7	IRQCEC6	IRQCEC5	IRQCEC4	IRQCEC3	IRQCEC2	IRQCEC1	IRQCEC0
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W
PROT_TYPE	WPS							
INITIAL_VALUE	0	0	0	0	0	0	0	0

#### [bit31] to [bit0] IRQCECr: IRQr channel enable clear bits

These bits clear channel enable for the corresponding IRQr.

Value	Description
0	Invalid (no effect on operation)
1	Clear channel enable for IRQr.

#### Notes:

- The register bit configuration described in this section is for IRCN\_IRQCEC0, which is representative of those registers that are all the same. The bit configurations of registers IRCN\_IRQCEC0 to IRCN\_IRQCEC15 are all the same.
- IRQ0 to IRQ31 are assigned to IRCN\_IRQCEC0, and the subsequent channels are assigned in the same way until IRQ480 to IRQ511 are assigned to IRCN\_IRQCEC15.
- The "r" in bit field name IRQCECr corresponds to the IRQ channel number, r (0 to 511).

## 5.18. IRC IRQ Channel Enable Setting Register (IRCN\_IRQCE0 to IRCN\_IRQCE15)

These registers set enable of individual IRQ channels. For each channel, 1 bit is assigned. The 16 registers are all of the same type, and each is identified by the number at the end of the abbreviated register name. For each register, 32 channels are assigned, so that 512 channels are supported by the 16 registers.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	IRQCE31	IRQCE30	IRQCE29	IRQCE28	IRQCE27	IRQCE26	IRQCE25	IRQCE24
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	WPS							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	IRQCE23	IRQCE22	IRQCE21	IRQCE20	IRQCE19	IRQCE18	IRQCE17	IRQCE16
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	WPS							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	IRQCE15	IRQCE14	IRQCE13	IRQCE12	IRQCE11	IRQCE10	IRQCE9	IRQCE8
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	WPS							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	IRQCE7	IRQCE6	IRQCE5	IRQCE4	IRQCE3	IRQCE2	IRQCE1	IRQCE0
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	WPS							
INITIAL_VALUE	0	0	0	0	0	0	0	0

### [bit31] to [bit0] IRQCEr: IRQr channel enable setting bits

These bits set channel enable for the corresponding IRQr.

Value	Description
0	Clear channel enable for IRQr.
1	Set channel enable for IRQr.

#### Notes:

- The register bit configuration described in this section is for IRCN\_IRQCE0, which is representative of those registers that are all the same. The bit configurations of registers IRCN\_IRQCE0 to IRCN\_IRQCE15 are all the same.
- IRQ0 to IRQ31 are assigned to IRCN\_IRQCE0, and the subsequent channels are assigned in the same way until IRQ480 to IRQ511 are assigned to IRCN\_IRQCE15.
- The "r" in bit field name IRQCEr corresponds to the IRQ channel number, r (0 to 511).



## 5.19. IRC NMI Hold Clear Register (IRCn\_NMIHC)

This register clears the hold bit for the ISR for the current NMI. The NMI channel number for which the hold bit is to be cleared is written.

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	WP
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved			NMIHCN				
ACCESS_TYPE	R0,WX			R0,W				
PROT_TYPE	WP							
INITIAL_VALUE	000			00000				

**[bit31:5] Reserved: Reserved bits**

**[bit4:0] NMIHCN[4:0]: Hold clear NMI channel number bits**

These bits set the channel number corresponding to the ISR for the NMI.

## 5.20. IRC NMI Hold Status Register (IRCn\_NMIHS)

This register indicates the hold status of the individual NMI channels. For each channel, 1 bit is assigned, and the bit location corresponds to the NMI channel number, m (0 to 31).

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	NMIHS31	NMIHS30	NMIHS29	NMIHS28	NMIHS27	NMIHS26	NMIHS25	NMIHS24
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	NMIHS23	NMIHS22	NMIHS21	NMIHS20	NMIHS19	NMIHS18	NMIHS17	NMIHS16
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	NMIHS15	NMIHS14	NMIHS13	NMIHS12	NMIHS11	NMIHS10	NMIHS9	NMIHS8
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	NMIHS7	NMIHS6	NMIHS5	NMIHS4	NMIHS3	NMIHS2	NMIHS1	NMIHS0
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

### [bit31] to [bit0] NMIHSm: NMIm hold status bits

These bits indicate the hold status for NMIm.

Value	Description
0	The interrupt for NMIm is not applied to the CPU.
1	The interrupt for NMIm is being applied to the CPU.





## 5.21. IRC IRQ Hold Clear Register (IRCn\_IRQHC)

This register clears the hold bit for the ISR for the current IRQ. The IRQ channel number for which the hold bit is to be cleared is written.

BIT_OFFSET	31-16
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	WP
INITIAL_VALUE	00000000_00000000

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved							IRQHCN[8]
ACCESS_TYPE	R0,WX							R0,W
PROT_TYPE	WP							
INITIAL_VALUE	00000000							0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	IRQHCN[7:0]							
ACCESS_TYPE	R0,W							
PROT_TYPE	WP							
INITIAL_VALUE	00000000							

**[bit31:9] Reserved: Reserved bits**

**[bit8:0] IRQHCN[8:0]: Bits for IRQ channel number for which hold is to be cleared.**

These bits set the channel number corresponding to the ISR for the IRQ.

**Notes:**

- This register is forbidden to be cleared by byte access write. Access size must be larger than half size. Bus error will occur if write operation in byte size is done.
- In case of writing to this register, perform it in any of the following states (= IRC is stopping).
  - Until writing the target register from reading that nIRQ bit of IRCn\_IRQST register is "1". Refer to the description of nIRQ bit for the target register.
  - The state that IRQEN bit of IRCn\_CSR register is "0" (immediately after reset, etc).

## 5.22. IRC IRQ Hold Status Register (IRCN\_IRQHS0 to IRCN\_IRQHS15)

These registers indicate the hold status of individual IRQ channels. For each channel, 1 bit is assigned. The 16 registers are all of the same type, and each is identified by the number at the end of the abbreviated register name. For each register, 32 channels are assigned, so that 512 channels can be supported by the 16 registers.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	IRQHS31	IRQHS30	IRQHS29	IRQHS28	IRQHS27	IRQHS26	IRQHS25	IRQHS24
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	IRQHS23	IRQHS22	IRQHS21	IRQHS20	IRQHS19	IRQHS18	IRQHS17	IRQHS16
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	IRQHS15	IRQHS14	IRQHS13	IRQHS12	IRQHS11	IRQHS10	IRQHS9	IRQHS8
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	IRQHS7	IRQHS6	IRQHS5	IRQHS4	IRQHS3	IRQHS2	IRQHS1	IRQHS0
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

### [bit31] to [bit0] IRQHSr: IRQr hold status bits

These bits indicate the hold status of IRQr.

Value	Description
0	The interrupt for IRQr is not applied to the CPU.
1	The interrupt for IRQr is being applied to the CPU.

#### Notes:

- The register bit configuration described in this section is for IRCN\_IRQHS0, which is representative of those registers that are all the same. The bit configurations of registers IRCN\_IRQHS0 to IRCN\_IRQHS15 are all the same.
- IRQ0 to IRQ31 are assigned to IRCN\_IRQHS0, and the subsequent channels are assigned in the same way until IRQ480 to IRQ511 are assigned to IRCN\_IRQHS15.
- The "r" in bit field name IRQHSr corresponds to the IRQ channel number, r (0 to 511).



### 5.23. IRC IRQ Priority Level Mask Register (IRCn\_IRQPLM)

This register sets the interrupt mask based on the priority level. The IRQ channel whose priority level is equal to or higher than the value set in this register is masked. The initial value for this register is 32. That is, no priority levels are masked in the initial state.

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	WPS
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved		IRQPLM					
ACCESS_TYPE	R0,WX		R/W					
PROT_TYPE	WPS							
INITIAL_VALUE	00		100000					

**[bit31:6] Reserved: Reserved bits**

**[bit5:0] IRQPLM[5:0]: IRQ priority level mask bits**

These bits set the priority level mask value.

**Note:**

- In case of writing to this register, perform it in any of the following states (= IRC is stopping).
  - Until writing the target register from reading that nIRQ bit of IRCn\_IRQST register is "1". Refer to the description of nIRQ bit for the target register.
  - The state that IRQEN bit of IRCn\_CSR register is "0" (immediately after reset, etc).

## 5.24. IRC Control/Status Register (IRCn\_CSR)

This register sets enable/disable of the IRQ processing block of the interrupt controller. It also indicates the lock status of the interrupt controller.

BITS_OFFSET	31	30	29	28	27	26	25	24
BITS_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							

BITS_OFFSET	23	22	21	20	19	18	17	16
BITS_NAME	Reserved							LST
ACCESS_TYPE	R0,WX							R,WX
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							1

BITS_OFFSET	15	14	13	12	11	10	9	8
BITS_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							

BITS_OFFSET	7	6	5	4	3	2	1	0
BITS_NAME	Reserved							IRQEN
ACCESS_TYPE	R0,WX							R/W
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							0

[bit31:17] Reserved: Reserved bits

### [bit16] LST: Interrupt controller lock status

This bit indicates the lock status of the interrupt controller.

Value	Description
0	Unlocked state
1	Locked state

[bit15:1] Reserved: Reserved bits



**[bit0] IRQEN: IRQ processing block enable/disable setting bit**

This bit sets enable/disable of the IRQ processing block of the interrupt controller. If the IRQ processing block is disabled, the interrupt controller does not accept a new IRQ. The accepted interrupt is provided to the CPU.

Value	Description
0	IRQ processing block is disabled.
1	IRQ processing block is enabled.

**Note:**

- In case of writing to this register, perform it in any of the following states (= IRC is stopping).
- Until writing the target register from reading that *nIRQ* bit of *IRCN\_IRQST* register is "1". Refer to the description of *nIRQ* bit for the target register.
- The state that *IRQEN* bit of *IRCN\_CSR* register is "0" (immediately after reset, etc).

## 5.25. IRC NMI RAW Status Register (IRCn\_NMIRS)

This register indicates the RAW status of individual NMI channels. For each channel, 1 bit is assigned, and the bit location corresponds to the NMI channel number, m (0 to 31).

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	NMIRS31	NMIRS30	NMIRS29	NMIRS28	NMIRS27	NMIRS26	NMIRS25	NMIRS24
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	NMIRS23	NMIRS22	NMIRS21	NMIRS20	NMIRS19	NMIRS18	NMIRS17	NMIRS16
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	NMIRS15	NMIRS14	NMIRS13	NMIRS12	NMIRS11	NMIRS10	NMIRS9	NMIRS8
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	NMIRS7	NMIRS6	NMIRS5	NMIRS4	NMIRS3	NMIRS2	NMIRS1	NMIRS0
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

### [bit31] to [bit0] NMIRSm: RAW status bits for NMIm

These bits indicate the RAW status of NMIm.

Value	Description
0	Interrupt is not set.
1	Interrupt is set.



## 5.26. IRC NMI Preprocessed Status Register (IRCn\_NMIPS)

This register indicates the preprocessed status of individual NMI channels. For each channel, 1 bit is assigned, and the bit location corresponds to the NMI channel number, m (0 to 31).

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	NMIPS31	NMIPS30	NMIPS29	NMIPS28	NMIPS27	NMIPS26	NMIPS25	NMIPS24
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	NMIPS23	NMIPS22	NMIPS21	NMIPS20	NMIPS19	NMIPS18	NMIPS17	NMIPS16
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	NMIPS15	NMIPS14	NMIPS13	NMIPS12	NMIPS11	NMIPS10	NMIPS9	NMIPS8
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	NMIPS7	NMIPS6	NMIPS5	NMIPS4	NMIPS3	NMIPS2	NMIPS1	NMIPS0
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

### [bit31] to [bit0] NMIPSm: Preprocessed status bits for NMIm

These bits indicate the preprocessed status of NMIm.

Value	Description
0	Interrupt is not set.
1	Interrupt is set.

## 5.27. IRC IRQ RAW Status Register (IRCN\_IRQRS0 to IRCN\_IRQRS15)

These registers indicate the RAW status of individual IRQ channels. For each channel, 1 bit is assigned. The 16 registers are all of the same type, and each is identified by the number at the end of the abbreviated register name. For each register, 32 channels are assigned, so that 512 channels are supported by the 16 registers.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	IRQRS31	IRQRS30	IRQRS29	IRQRS28	IRQRS27	IRQRS26	IRQRS25	IRQRS24
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	IRQRS23	IRQRS22	IRQRS21	IRQRS20	IRQRS19	IRQRS18	IRQRS17	IRQRS16
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	IRQRS15	IRQRS14	IRQRS13	IRQRS12	IRQRS11	IRQRS10	IRQRS9	IRQRS8
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	IRQRS7	IRQRS6	IRQRS5	IRQRS4	IRQRS3	IRQRS2	IRQRS1	IRQRS0
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

### [bit31] to [bit0] IRQRSr: IRQr RAW status bits

These bits indicate the RAW status of IRQr.

Value	Description
0	Interrupt is not set.
1	Interrupt is set.

#### Notes:

- The register bit configuration described in this section is for IRCN\_IRQRS0, which is representative of those registers that are all the same. The bit configurations of registers IRCN\_IRQRS0 to IRCN\_IRQRS15 are all the same.
- IRQ0 to IRQ31 are assigned to IRCN\_IRQRS0, and the subsequent channels are assigned in the same way until IRQ480 to IRQ511 are assigned to IRCN\_IRQRS15.
- The "r" in bit field name IRQRSr corresponds to the IRQ channel number, r (0 to 511).





## 5.28. IRC IRQ Preprocessed Status Register (IRCN\_IRQPS0 to IRCN\_IRQPS15)

These registers indicate the preprocessed status of individual IRQ channels. For each channel, 1 bit is assigned. The 16 registers are all of the same type, and each is identified by the number at the end of the abbreviated register name. For each register, 32 channels are assigned, so that 512 channels are supported by the 16 registers.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	IRQPS31	IRQPS30	IRQPS29	IRQPS28	IRQPS27	IRQPS26	IRQPS25	IRQPS24
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	IRQPS23	IRQPS22	IRQPS21	IRQPS20	IRQPS19	IRQPS18	IRQPS17	IRQPS16
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	IRQPS15	IRQPS14	IRQPS13	IRQPS12	IRQPS11	IRQPS10	IRQPS9	IRQPS8
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	IRQPS7	IRQPS6	IRQPS5	IRQPS4	IRQPS3	IRQPS2	IRQPS1	IRQPS0
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

### [bit31] to [bit0] IRQPSr: IRQr preprocessed status bits

These bits indicate the preprocessed status of IRQr.

Value	Description
0	Interrupt is not set.
1	Interrupt is set.

#### Notes:

- The register bit configuration described in this section is for IRCN\_IRQPS0, which is representative of those registers that are all the same. The bit configurations of registers IRCN\_IRQPS0 to IRCN\_IRQPS15 are all the same.
- IRQ0 to IRQ31 are assigned to IRCN\_IRQPS0, and the subsequent channels are assigned in the same way until IRQ480 to IRQ511 are assigned to IRCN\_IRQPS15.
- The "r" in bit field name IRQPSr corresponds to the IRQ channel number, r (0 to 511).

## 5.29. IRC Unlock Register (IRCn\_UNLOCK)

This register controls write lock for the interrupt controller to each register.

BIT_OFFSET	31-0
BIT_NAME	UNLOCK
ACCESS_TYPE	R0,W
PROT_TYPE	WP
INITIAL_VALUE	00000000_00000000_00000000_00000000

### [bit31:0] UNLOCK[31:0]: Interrupt controller unlock bits

These bits control write lock for the interrupt controller to the setting registers.

The target registers for sequence protection by this register are those other than IRCn\_NMIHC, IRCn\_IRQHC, IRCn\_NMIR, and IRCn\_IRQR0 to IRCn\_IRQR15.

Value	Description
0x17ACC911	Unlock value (which enables writing)
0x17B10C11	Lock value (which disables writing)
Other than above	Setting prohibited (bus error returned)

#### Note:

- The values of all 32 bits of this register are needed to determine the lock status. Therefore, writing to this register must be done with 32-bit access. Neither 8-bit nor 16-bit access is allowed.



### 5.30. IRC ECC Error Interrupt Register (IRCn\_EEI)

This register indicates the ECC error interrupt status. It also clears the interrupt. There are 2 types of ECC error interrupt: NMI and IRQ.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							EEIS
ACCESS_TYPE	R0,WX							R,WX
PROT_TYPE	WPS							
INITIAL_VALUE	0000000							0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved							EEIC
ACCESS_TYPE	R0,WX							R0,W
PROT_TYPE	WPS							
INITIAL_VALUE	0000000							0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved							EENS
ACCESS_TYPE	R0,WX							R,WX
PROT_TYPE	WPS							
INITIAL_VALUE	0000000							0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved							EENC
ACCESS_TYPE	R0,WX							R0,W
PROT_TYPE	WPS							
INITIAL_VALUE	0000000							0

**[bit31:25] Reserved: Reserved bits**

**[bit24] EEIS: ECC error IRQ status bit**

This bit indicates the status of the ECC error IRQ. When data is read from SRAM installed in the interrupt controller, this bit becomes "1" if a single-bit ECC error occurs.

Value	Description
0	Single bit ECC error has not occurred.
1	Single bit ECC error has occurred.

**[bit23:17] Reserved: Reserved bits**

**[bit16] EEIC: ECC error IRQ clear bit**

This bit clears the ECC error IRQ.

Value	Description
0	Invalid (no effect on operation)
1	Clear the ECC error IRQ (EEIS).

**[bit15:9] Reserved: Reserved bits**

**[bit8] EENS: ECC error NMI status bit**

This bit indicates the status of the ECC error NMI. When data is read from SRAM installed in the interrupt controller through the IRQVAR register, this bit becomes "1" if a double-bit ECC error occurs.

Value	Description
0	Double bit ECC error has not occurred.
1	Double bit ECC error has occurred.

**Note:**

- This flag is not set in accessing the IRQ register in the IRQ processing stage.

**[bit7:1] Reserved: Reserved bits**

**[bit0] EENC: ECC error NMI clear bit**

This bit clears the ECC error NMI.

Value	Description
0	Invalid (no effect on operation)
1	Clear ECC error NMI (EENS).

**Note:**

- EENS must be cleared only after the IRQ vector address is reinitialized.



### 5.31. IRC ECC Address Number Register (IRCn\_EAN)

When a single-bit or double-bit ECC error occurs, this register indicates the SRAM address of the error.

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	EAN							
ACCESS_TYPE	R,WX							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

**[bit31:8] Reserved: Reserved bits**

**[bit7:0] EAN[7:0]: ECC error occurrence address bits**

These bits indicate the SRAM address where the ECC error is generated.

**Notes:**

- If a single-bit error occurs after a double-bit error, SRAM address will not be updated while  $IRCn\_EEI:EENS=1$ .
- If a single-bit or double-bit error occurs continuously, this register indicates the address of the last error.

### 5.32. IRC ECC Test Register (IRCn\_ET)

This register sets enable/disable of the test mode for the ECC protection function of SRAM installed in the interrupt controller.

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	WPS
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved							ET
ACCESS_TYPE	R0,WX							R/W
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							0

**[bit31:1] Reserved: Reserved bits**

**[bit0] ET: ECC test enable/disable setting bit**

This bit sets enable/disable of the ECC test mode.

Value	Description
0	ECC test mode is disabled.
1	ECC test mode is enabled.



### 5.33. IRC ECC Error Bit Register (IRCn\_EEB0 to IRCn\_EEB1)

These registers are used in ECC test mode. In the data read from SRAM, you can spuriously corrupt any of the bits in the data area.

BIT_OFFSET	31-8
BIT_NAME	EEB[29:6]
ACCESS_TYPE	R/W
PROT_TYPE	WPS
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	EEB[5:0]						Reserved	
ACCESS_TYPE	R/W						R0,WX	
PROT_TYPE	WPS							
INITIAL_VALUE	000000						00	

#### [bit31:2] EEB[29:0]: ECC error occurrence bits

These bits are used to invert the data read from SRAM using the bitwise operation. If IRCn\_ET:ET is "1", [29:0] of the data read from SRAM is XORed with EEB, and the result is handled as the read data. This means that the bits corresponding to those to which "1" is written within EEB are inverted. As a result, the SRAM data is spuriously corrupted and used for the test for the ECC protection function.

#### [bit1:0] Reserved: Reserved bits

**Note:**

- The register bit configuration described in this section is for IRCn\_EEB0, which is representative of those registers that are all the same. The bit configuration of register IRCn\_EEB1 is the same. IRCn\_EEB1:EEB corresponds to the bits [66:37] of the data read from SRAM.

### 5.34. IRC ECC Error Bit Register (IRCn\_EEB2)

This register is used in ECC test mode. In the data read from SRAM, you can spuriously corrupt any of the bits in the ECC parity area.

BIT_OFFSET	31-16
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	WPS
INITIAL_VALUE	00000000_00000000

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved	EEBO						
ACCESS_TYPE	R0,WX	R/W						
PROT_TYPE	WPS							
INITIAL_VALUE	0	0000000						

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved	EEBE						
ACCESS_TYPE	R0,WX	R/W						
PROT_TYPE	WPS							
INITIAL_VALUE	0	0000000						

**[bit31:15] Reserved: Reserved bits**

#### **[bit14:8] EEBO[6:0]: ECC error occurrence bits**

These bits are used to invert the data read from SRAM using the bitwise operation. If IRCn\_ET:ET is "1", [73:67] of the data read from SRAM is XORed with EEBO, and the result is handled as the read data. This means the bits corresponding to those to which "1" is written within EEBO are inverted. As a result, the SRAM data is spuriously corrupted and used for the test for the ECC protection function.

**[bit7] Reserved: Reserved bit**

#### **[bit6:0] EEBE[6:0]: ECC error occurrence bits**

These bits are used to invert the data read from SRAM using the bitwise operation. If IRCn\_ET:ET is "1", [36:30] of the data read from SRAM is XORed with EEBE, and the result is handled as the read data. This means that the bits corresponding to those to which "1" is written within EEBE are inverted. As a result, the SRAM data is spuriously corrupted and used for the test for the ECC protection function.





### 5.35. IRC NMI Vector Address Status Register (IRCn\_NMIVASBR)

This register indicates the vector address status of the NMI that was applied to the CPU most recently.

BIT_OFFSET	31-0
BIT_NAME	NMIVAS
ACCESS_TYPE	R,WX
PROT_TYPE	RP
INITIAL_VALUE	00000000_00000000_00000000_00000000

#### [bit31:0] NMIVAS[31:0]: NMI vector address status bits

These bits indicate the vector address status of the NMI that was applied to the CPU most recently.

### 5.36. IRC NMI Vector Address Status Mirror Register (IRC\_NMIVASBR)

This register is a mirror register of IRCn\_NMIVASBR, and is dedicated to BootROM. The CPU can read this register and branch to the ISR for the NMI.

BIT_OFFSET	31-0
BIT_NAME	NMIVAS
ACCESS_TYPE	R,WX
PROT_TYPE	RP
INITIAL_VALUE	00000000_00000000_00000000_00000000

#### [bit31:0] NMIVAS[31:0]: NMI vector address status bits

These bits indicate the vector address status of the NMI that was applied to the CPU most recently.

**Notes:**

- *This register is dedicated to BootROM and is used as follows:*
  1. An NMI occurs.
  2. The execution code jumps to the corresponding entry in the fixed exception vector table of BootROM.
  3. The load instruction at the exception entry point loads the value stored in IRC\_NMIVASBR.
  4. The code jumps to the vector address of the accepted NMI and executes the ISR for the NMI.
- *Each CPU reads the same address from the register where the stored address is mapped to that which is common to all CPUs.*
- *When a read operation is executed by CPU<sub>n</sub>, the register for IRC<sub>n</sub> is read. AHB masters other than CPUs cannot read this register.*



### 5.37. IRC ECC Error Vector Address Register (IRCn\_IRQEEVA)

This register is used to perform correct error handling when SRAM holding the IRQ vector is read by the IRQ processing block and a 2-bit ECC error is detected.

BIT_OFFSET	31-8
BIT_NAME	IRQVA[31:8]
ACCESS_TYPE	R/W
PROT_TYPE	WPS
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	IRQVA[7:2]						IRQVA[1:0]	
ACCESS_TYPE	R/W						R0,WX	
PROT_TYPE	WPS							
INITIAL_VALUE	000000						00	

#### [bit31:0] IRQVA[31:0]: IRQ vector address bits

These bits set the address of the error handler when SRAM holding the IRQ vector is read by the IRQ processing block and a 2-bit ECC error is detected.

#### Notes:

- Since all vector addresses are 32 bits long, the lowest 2 bits are fixed to "00".
- If a 2-bit ECC error occurs in vector SRAM during IRQ processing, transition is made to this vector address, rather than to the interrupt vector of the IRQ that has occurred. A transition to this vector cannot be changed. Therefore, the vector address must be set to this register to enable correct error handling.
- Set a value in this register before enabling the IRQ processing block.

## 6. Others

This section lists precautions to be observed when using the interrupt controller.

- Software interrupts must be cleared by ISR.
- If 2 interrupts of the same priority level are asserted at the same time, that with the smaller channel number takes precedence. If an interrupt occurs, followed by another of the same priority level, the latter interrupt will be masked until the service routine for the former is completed.
- All vector addresses should be set during the initialization process.
- For the interrupt controller to work properly, the peripheral interrupt must be cleared before the corresponding hold bit in the interrupt controller is cleared.
- In the case of a software interrupt, the software interrupt setting bit in the interrupt controller must be cleared before the corresponding hold bit in the interrupt controller is cleared.
- If a double-bit ECC error in SRAM is detected by reading IRCn\_IRQVAr by CPU, a correct vector address must be written again in the relevant IRCn\_IRQVAr register with the corresponding NMI handler. Then, the error bit must be cleared by writing to IRCn\_EEI:EENC.
- If a double-bit ECC error in SRAM is detected by reading IRCn\_IRQVAr by IRQ processing unit, correct vector address must be rewritten to IRCn\_IRQVAr register corresponded to IRQEEVA handler.

The conditions under which a bus error is returned for register access in the interrupt controller are shown below:

- Sequence protection violation by IRCn\_UNLOCK
- Writing values other than the lock value and unlock value to IRCn\_UNLOCK
- Privilege protection violation
- Read or write attempt in which access is made only to an undefined register area
- Read or write attempt in which access is made only to reserved bits
- Write attempt in which access is made only to bits having the WX write attribute (including the undefined register area)
- When write by byte access is performed to IRCn\_IRQHC register





# CHAPTER 11: BootROM Hardware Interface

This chapter explains the hardware interface.

---

1. Overview
2. Configuration
3. Operation
4. Setting Procedure Example
5. Registers
6. Others



## 1. Overview

This section provides an overview of the BootROM hardware interface.

The BootROM hardware interface is the interface between CPU and BootROM. The BootROM hardware interface sets the setting registers for the user-defined exception handler. The interface has 2 exception vector register sets. When one of the sets is in use, the setting of the other one can be changed.

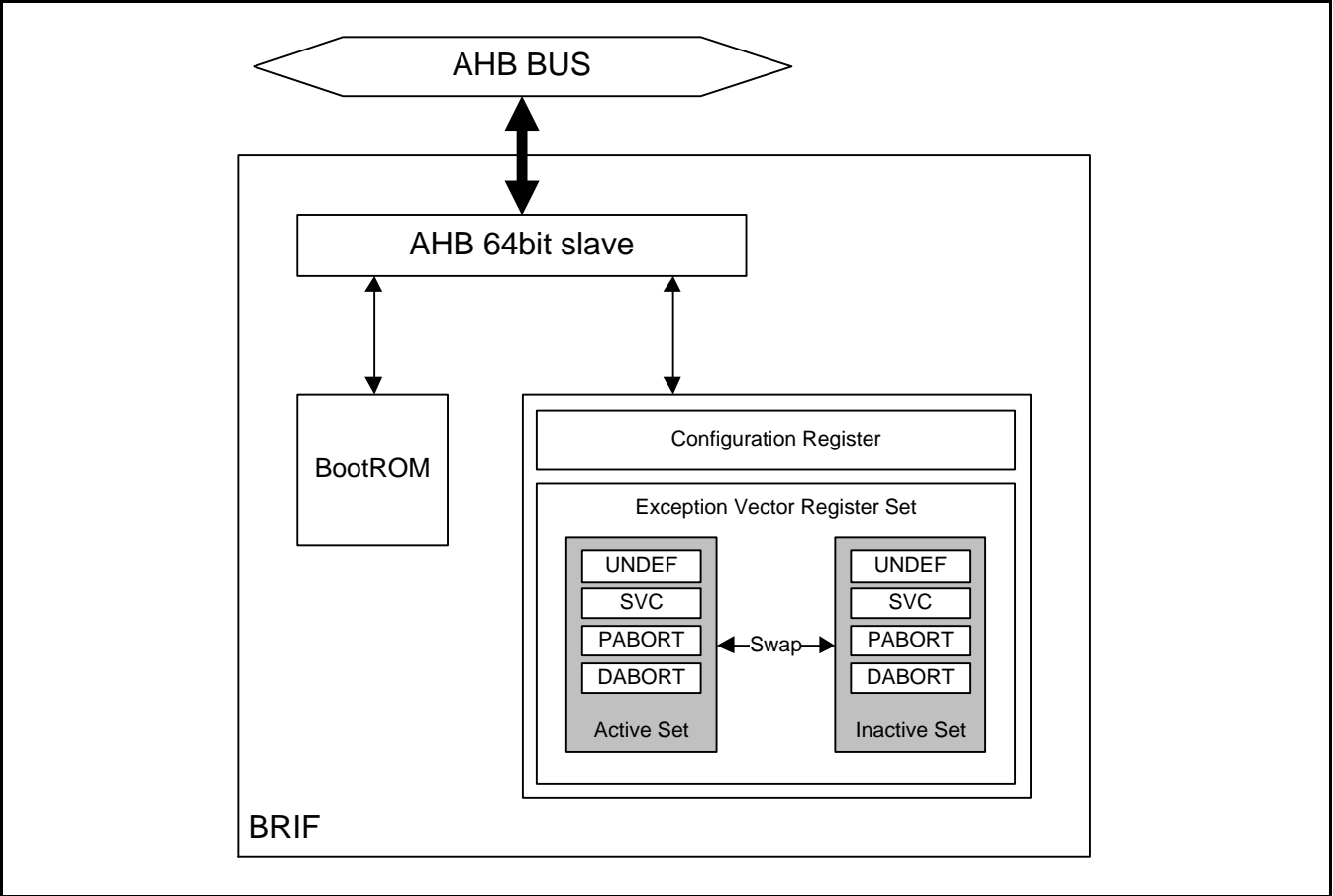
**Notes:**

- *In this chapter, the BootROM hardware interface is sometimes referred to by its abbreviation BRIF (BootRom hardware InterFace).*
- *Among the abbreviated register names of the BootROM hardware interface, "EXCFG" in "EXCFG\_\*\*\*\*" indicates exception vector configuration (EXception vector ConFiGuration).*

2. Configuration

This section explains a block diagram of the BootROM hardware interface.

Figure 2-1 BootROM Hardware Interface Block Diagram







### 3. Operation

This section explains the operation of the BootROM hardware interface.

#### (1) Sequence Protection by the Lock Release Register

The BootROM hardware interface provides the sequence protection function made available by the lock release register (EXCFG\_UNLOCK). To update the exception vector configuration, the lock release value must be written in EXCFG\_UNLOCK in advance to release locking. Write the lock value after update of the exception vector configuration to lock it again.

#### (2) ARM® Cortex™-R5 Exception Vector Processing

ARM Cortex-R5 processes exceptions in the following steps:

1. Exception occurs.
2. The control jumps to the execution code of the corresponding entry of the fixed exception vector table of BootROM.
3. The load instruction at the exception entry loads the value of each exception vector register of PC.
4. The control jumps to the starting address of the exception handler, and the exception handler is executed.

The exception entry for the BootROM vector table is fixed. However, in order to make the jump destination changeable, the starting address of the ARM Cortex-R5 exception handler (inline literal) is stored in registers. These registers are mapped to addresses lower than BootROM, and are referred to by the BootROM exception entry when an exception occurs. This arrangement allows the exception vectors to be dynamically defined.

In order to allow all exception vector registers to be redefined by a single access, there are two register sets: active and inactive sets. These two sets are mapped to addresses lower than BootROM. The address to which the active set is mapped is at an active position that the BootROM exception entry refers to at the time of exception occurrence. The address to which the inactive set is mapped is at an inactive position, in which the values can be changed. Writing "1" to the exception vector register set swap bit of the setting register (EXCFG\_CNFG:SWAP) swaps the contents of the active and inactive sets. This action changes the settings of the active set.

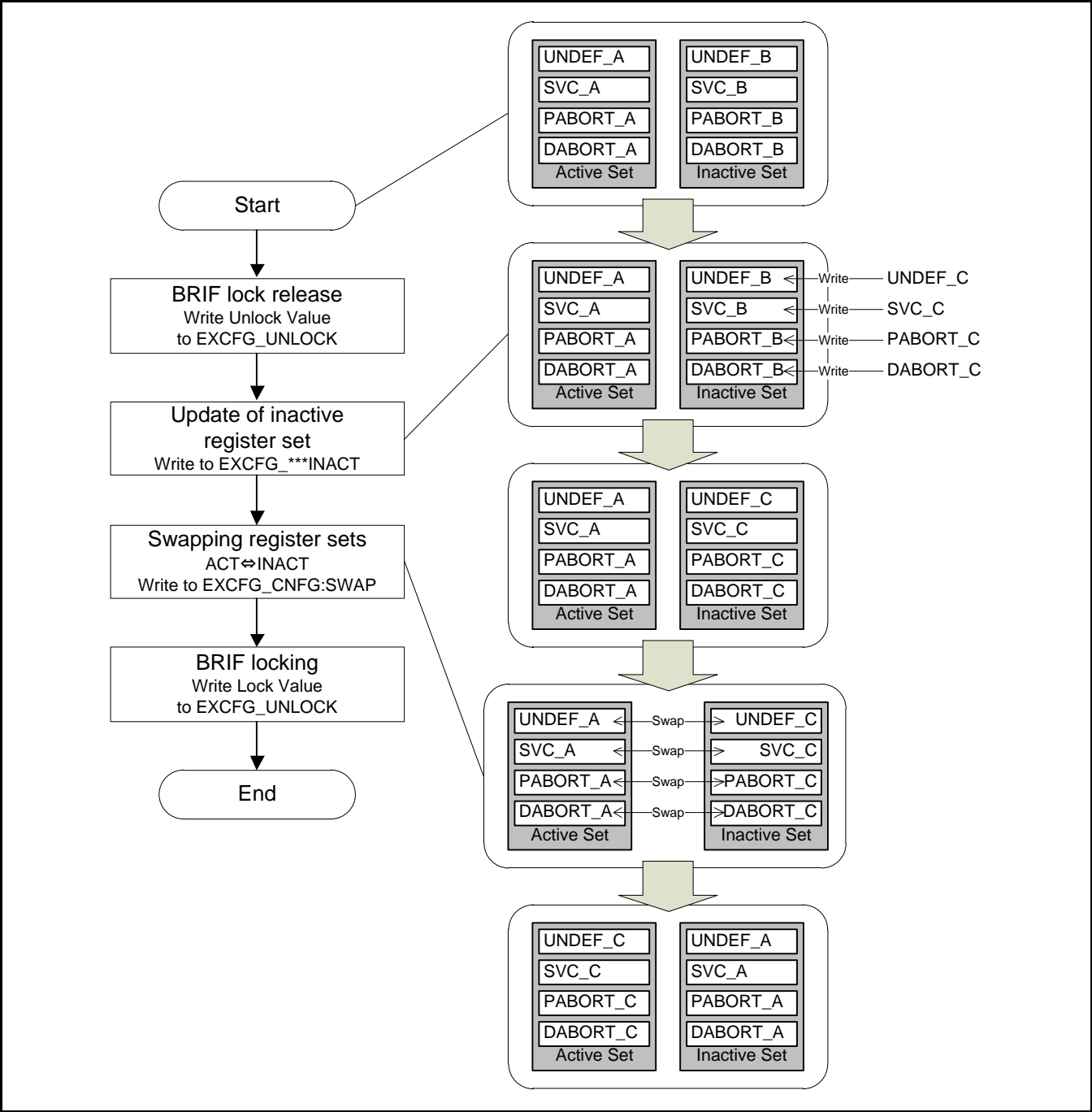
**Note:**

- It takes 30 to 40 clocks (CLK\_MEMC) supplied to this block after writing "1" to EXCFG\_CNFG:SWAP until the exception handler is enabled with the completion of swap. It is unpredictable which handler will be executed during the swapping period.

4. Setting Procedure Example

This section explains a setting procedure example of BootROM hardware interface.

Figure 4-1 Setting Flow of BootROM Hardware Interface





## 5. Registers

This section explains the registers used by the BootROM hardware interface.

**Table 5-1 List of BootROM Hardware Interface Registers**

Abbreviated Register Name	Register Name	See
EXCFG_UNLOCK	EXCFG Lock Release Register	5.1
EXCFG_CNFG	EXCFG Setting Register	5.2
EXCFG_UNDEFINACT	EXCFG Inactive Set - Undefined Instruction Vector Register	5.3
EXCFG_SVCINACT	EXCFG Inactive Set - Supervisor Call Vector Register	5.4
EXCFG_PABORTINACT	EXCFG Inactive Set - Prefetch Abort Vector Register	5.5
EXCFG_DABORTINACT	EXCFG Inactive Set - Data Abort Vector Register	5.6
EXCFG_UNDEFACT	EXCFG Active Set - Undefined Instruction Vector Register	5.7
EXCFG_SVCACT	EXCFG Active Set - Supervisor Call Vector Register	5.8
EXCFG_PABORTACT	EXCFG Active Set - Prefetch Abort Vector Register	5.9
EXCFG_DABORTACT	EXCFG Active Set - Data Abort Vector Register	5.10

## 5.1. EXCFG Lock Release Register (EXCFG\_UNLOCK)

This register controls the write lock of the registers of the BootROM hardware interface.

BIT_OFFSET	31-0
BIT_NAME	UNLOCK
ACCESS_TYPE	R0,W
PROT_TYPE	WP
INITIAL_VALUE	00000000_00000000_00000000_00000000

### [bit31:0] UNLOCK[31:0]: Lock release bits of BootROM hardware interface

These bits control the write lock of the setting registers of the BootROM hardware interface. All registers except EXCFG\_UNLOCK are subject to sequence protection by this register.

Value	Description
0xACC5B007	Unlock value (which enables writing)
0xB007ECF6	Lock value (which disables writing)
Other than above	Setting prohibited (bus error returned)

**Note:**

- The value of all 32 bits of this register is needed to determine the lock status. Therefore, writing to this register must be made with 32-bit access. The writing of 8-bit, 16-bit, or 64-bit is not allowed.



## 5.2. EXCFG Setting Register (EXCFG\_CNFG)

This register indicates the lock status of the BootROM hardware interface. This register also swaps the contents of the active and inactive sets of the exception vector registers.

BIT_OFFSET	31-16
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	WPS
INITIAL_VALUE	00000000_00000000

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved							SWAP
ACCESS_TYPE	R0,WX							R0,W
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved							LST
ACCESS_TYPE	R0,WX							R,WX
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							1

**[bit31:9] Reserved: Reserved bits**

### **[bit8] SWAP: Exception vector register swap bit**

Writing "1" to this bit swaps the contents of the active and inactive sets of the exception vector registers.

Value	Description
0	Invalid (no effect on operation)
1	Swap execution

**[bit7:1] Reserved: Reserved bits**

### **[bit0] LST: BootROM hardware interface lock status bit**

This bit indicates the lock status of the BootROM hardware interface.

Value	Description
0	Unlocked state
1	Locked state

**Note:**

- It takes 30 to 40 clocks (CLK\_MEMC) supplied to this block after writing "1" to EXCFG\_CNFG:SWAP until the exception handler is enabled with the completion of swap. It is unpredictable which handler will be executed during the swapping period.

### 5.3. EXCFG Inactive Set - Undefined Instruction Vector Register (EXCFG\_UNDEFINACT)

This register is included in the inactive set of the exception vector registers, and stores the starting address of the exception handler for the undefined instruction exception.

BIT_OFFSET	31-0
BIT_NAME	UNDEFVEC
ACCESS_TYPE	R,W
PROT_TYPE	WPS
INITIAL_VALUE	11111111_11111111_00000000_00100100

#### [bit31:0] UNDEFVEC[31:0]: Undefined instruction vector bits

These bits set the starting address of the exception handler for the undefined instruction exception.

**Note:**

- Writing "1" to EXCFG\_CNFG:SWAP swaps the value of this register and the value of EXCFG\_UNDEFINACT.



## 5.4. EXCFG Inactive Set - Supervisor Call Vector Register (EXCFG\_SVCINACT)

This register is included in the inactive set of the exception vector registers, and stores the starting address of the exception handler for the supervisor call exception.

BIT_OFFSET	31-0
BIT_NAME	SVCVEC
ACCESS_TYPE	R,W
PROT_TYPE	WPS
INITIAL_VALUE	11111111_11111111_00000000_00101000

### [bit31:0] SVCVEC[31:0]: Supervisor call vector bits

These bits set the starting address of the exception handler for the supervisor call exception.

**Note:**

- Writing "1" to EXCFG\_CNFG:SWAP swaps the value of this register and the value of EXCFG\_SVCACT.

## 5.5. EXCFG Inactive Set - Prefetch Abort Vector Register (EXCFG\_PABORTINACT)

This register is included in the inactive set of the exception vector registers, and stores the starting address of the exception handler for the prefetch abort exception.

BIT_OFFSET	31-0
BIT_NAME	PABORTVEC
ACCESS_TYPE	R,W
PROT_TYPE	WPS
INITIAL_VALUE	11111111_11111111_00000000_00101100

### [bit31:0] PABORTVEC[31:0]: Prefetch abort vector bits

These bits set the starting address of the exception handler for the prefetch abort exception.

**Note:**

- Writing "1" to EXCFG\_CNFG:SWAP swaps the value of this register and the value of EXCFG\_PABORTACT.





## 5.6. EXCFG Inactive Set - Data Abort Vector Register (EXCFG\_DABORTINACT)

This register is included in the inactive set of the exception vector registers, and stores the starting address of the exception handler for the data abort exception.

BIT_OFFSET	31-0
BIT_NAME	DABORTVEC
ACCESS_TYPE	R,W
PROT_TYPE	WPS
INITIAL_VALUE	11111111_11111111_00000000_00110000

### [bit31:0] DABORTVEC[31:0]: Data abort vector bits

These bits set the starting address of the exception handler for the data abort exception.

**Note:**

- Writing "1" to EXCFG\_CNFG:SWAP swaps the value of this register and the value of EXCFG\_DABORTACT.

## 5.7. EXCFG Active Set - Undefined Instruction Vector Register (EXCFG\_UNDEFACT)

This register is included in the active set of the exception vector registers, and stores the starting address of the exception handler for the undefined instruction exception.

BIT_OFFSET	31-0
BIT_NAME	UNDEFVEC
ACCESS_TYPE	R,W
PROT_TYPE	WPS
INITIAL_VALUE	11111111_11111111_00000000_00100100

### [bit31:0] UNDEFVEC[31:0]: Undefined instruction vector bits

These bits store the starting address of the exception handler for the undefined instruction exception.

#### Notes:

- Writing to this register is prohibited. In this case, operation is not guaranteed. To set this register, set the value in the inactive set, and then execute swapping. For details on the setting procedure, see Section "3. Operation" and Section "4. Setting Procedure Example".
- Writing "1" to EXCFG\_CNFG:SWAP swaps the value of this register and the value of EXCFG\_UNDEFINACT.



## 5.8. EXCFG Active Set - Supervisor Call Vector Register (EXCFG\_SVCACT)

This register is included in the active set of the exception vector registers, and stores the starting address of the exception handler for the supervisor call exception.

BIT_OFFSET	31-0
BIT_NAME	SVCVEC
ACCESS_TYPE	R,W
PROT_TYPE	WPS
INITIAL_VALUE	11111111_11111111_00000000_00101000

### [bit31:0] SVCVEC[31:0]: Supervisor call vector bits

These bits store the starting address of the exception handler for the supervisor call exception.

#### Notes:

- *Writing to this register is prohibited. In this case, operation is not guaranteed. To set this register, set the value in the inactive set, and then execute swapping. For details on the setting procedure, see Section "3. Operation" and Section "4. Setting Procedure Example".*
- *Writing "1" to EXCFG\_CNFG:SWAP swaps the value of this register and the value of EXCFG\_SVCINACT.*

## 5.9. EXCFG Active Set - Prefetch Abort Vector Register (EXCFG\_PABORTACT)

This register is included in the active set of the exception vector registers, and stores the starting address of the exception handler for the prefetch abort exception.

BIT_OFFSET	31-0
BIT_NAME	PABORTVEC
ACCESS_TYPE	R,W
PROT_TYPE	WPS
INITIAL_VALUE	11111111_11111111_00000000_00101100

### [bit31:0] PABORTVEC[31:0]: Prefetch abort vector bits

These bits store the starting address of the exception handler for the prefetch abort exception.

#### Notes:

- *Writing to this register is prohibited. In this case, operation is not guaranteed. To set this register, set the value in the inactive set, and then execute swapping. For details on the setting procedure, see Section "3. Operation" and Section "4. Setting Procedure Example".*
- *Writing "1" to EXCFG\_CNFG:SWAP swaps the value of this register and the value of EXCFG\_PABORTINACT.*



## 5.10. EXCFG Active Set - Data Abort Vector Register (EXCFG\_DABORTACT)

This register is included in the active set of the exception vector registers, and stores the starting address of the exception handler for the data abort exception.

BIT_OFFSET	31-0
BIT_NAME	DABORTVEC
ACCESS_TYPE	R,W
PROT_TYPE	WPS
INITIAL_VALUE	11111111_11111111_00000000_00110000

### [bit31:0] DABORTVEC[31:0]: Data abort vector bits

These bits store the starting address of the exception handler for the data abort exception.

#### Notes:

- Writing to this register is prohibited. In this case, operation is not guaranteed. To set this register, set the value in the inactive set, and then execute swapping. For details on the setting procedure, see Section "3. Operation" and Section "4. Setting Procedure Example".
- Writing "1" to EXCFG\_CNFG:SWAP swaps the value of this register and the value of EXCFG\_DABORTINACT.

## 6. Others

This section explains precautions on the use of the BootROM hardware interface.

Register access or ROM access of the BootROM hardware interface returns a bus error in the following cases:

- Sequence protection violation by EXCFG\_UNLOCK
- Writing in EXCFG\_UNLOCK a value other than lock release or locking
- Privilege protection violation
- Read or write attempt in which access is made only to a register undefined area
- Write attempt in which access is made only to bits of WX attribute (including register undefined area)
- Writing to ROM





## CHAPTER 12: BootROM Software Interface

This section explains the BootROM Software Interface.

---

1. Overview
2. BootROM Markers
3. BootROM Operation
4. Notes





## 1. Overview

The BootROM software is built-in firmware that is executed after reset and before execution of user applications.

The BootROM software chiefly performs mode judgment, security setting, and hardware watchdog timer setting.

This section explains the functions of the BootROM software and a configuration based on the marker in the flash memory.

### BootROM Functions

The following lists the functions of the BootROM software.

1. Performs a CRC check to verify the normality of BootROM.
2. Makes a setting for enabling the VIC port and floating-point operations.
3. Judges the operating mode based on the MODE register and MD pin, and branches to a processing handler according to the mode.
4. Makes a security setting according to the value of the Security Description Record (SDR).
5. Waits for debugger connection based on the value of BDR\_DWEM of the Boot Description Record (BDR).
6. Makes a setting for the hardware watchdog timer according to the Watchdog Description Record (WDR) value.
7. For operation in 2CPU mode, CPU0 releases the HALT state of CPU1.
8. Judges the start address of a user application according to the values of BDR\_ABVM and BDR\_ABVEM of the Boot Description Record (BDR).
9. The exception vector table of the core is fixed at 0xFFFF0000. Therefore, the BootROM software references the interrupt controller, which is user-configurable, and the BootROM hardware interface to branch to an exception handler.
10. Issues a software trigger hard reset if an exception occurs before the user exception handler is set.

## 2. BootROM Markers

This section explains the BootROM markers.

- 2.1. Overview of BootROM Markers
- 2.2. Marker List
- 2.3. Flash Security Marker (SDR\_FSECM)
- 2.4. Debugger Connection Enable Marker (SDR\_DSM)
- 2.5. Debugger Security Key Marker 0 (SDR\_DSKM0)
- 2.6. Debugger Security Key Marker 1 (SDR\_DSKM1)
- 2.7. Debugger Security Key Marker 2 (SDR\_DSKM2)
- 2.8. Debugger Security Key Marker 3 (SDR\_DSKM3)
- 2.9. Debugger Connection Wait Enable Marker (BDR\_DWEM)
- 2.10. Alternative Boot Vector Marker (BDR\_ABVM)
- 2.11. Alternative Boot Vector Enable Marker (BDR\_ABVEM)
- 2.12. Hardware Watchdog Interrupt Configuration Marker (WDR\_INTM)
- 2.13. Hardware Watchdog Trigger 0 Configuration Marker (WDR\_TRG0CFGM)
- 2.14. Hardware Watchdog Trigger 1 Configuration Marker (WDR\_TRG1CFGM)
- 2.15. Hardware Watchdog Lower Limit RUN Setting Marker (WDR\_RUNLLM)
- 2.16. Hardware Watchdog Upper Limit RUN Setting Marker (WDR\_RUNULM)
- 2.17. Hardware Watchdog Lower Limit PSS Setting Marker (WDR\_PSSLLM)
- 2.18. Hardware Watchdog Upper Limit PSS Setting Marker (WDR\_PSSULM)
- 2.19. Hardware Watchdog Reset Delay Counter Marker (WDR\_RSTDLYM)
- 2.20. Hardware Watchdog Configuration Marker (WDR\_CFGM)
- 2.21. Hardware Watchdog Configuration Enable Marker (WDR\_CEM)



## 2.1. Overview of BootROM Markers

This section provides an overview of the BootROM markers.

### (1) Overview

The BootROM markers are setting data located in the TCFLASH. The BootROM software reads BootROM markers and controls hardware functions.

### (2) Types of BootROM Markers

There are the following 3 types of BootROM markers.

1. Security Description Record (SDR)  
Markers used for making security settings.
2. Boot Description Record (BDR)  
Markers used for making startup setting
3. Watchdog Description Record (WDR)  
Markers used for making hardware watchdog timer setting

**Note:**

- *In a multi-CPU configuration, there are multiple TCFLASHs because there is a TCFLASH for each CPU. For a multi-CPU configuration, set the BootROM markers in all TCFLASHs.*

## 2.2. Maker List

This section explains the lists of the BootROM markers.

The BootROM markers consist of the following three marker groups.

- Security Description Record (SDR) marker group
- Boot Description Record (BDR) marker group
- Watchdog Description Record (WDR) marker group

**Table 2-1 SDR Marker List**

Abbreviated Register Name	Register Name	See
SDR_FSECM	Flash Security Marker	2.3
SDR_DSM	Debugger Connection Enable Marker	2.4
SDR_DSKM0	Debugger Security Key Marker 0	2.5
SDR_DSKM1	Debugger Security Key Marker 1	2.6
SDR_DSKM2	Debugger Security Key Marker 2	2.7
SDR_DSKM3	Debugger Security Key Marker 3	2.8

**Table 2-2 BDR Marker List**

Abbreviated Register Name	Register Name	See
BDR_DWEM	Debugger Connection Wait Enable Marker	2.9
BDR_ABVM	Alternative Boot Vector Marker	2.10
BDR_ABVEM	Alternative Boot Vector Enable Marker	2.11

**Table 2-3 WDR Marker List**

Abbreviated Register Name	Register Name	See
WDR_INTM	Hardware Watchdog Interrupt Configuration Marker	2.12
WDR_TRG0CFGM	Hardware Watchdog Trigger 0 Configuration Marker	2.13
WDR_TRG1CFGM	Hardware Watchdog Trigger 1 Configuration Marker	2.14
WDR_RUNLLM	Hardware Watchdog Lower Limit RUN Setting Marker	2.15
WDR_RUNULM	Hardware Watchdog Upper Limit RUN Setting Marker	2.16
WDR_PSSLLM	Hardware Watchdog Lower Limit PSS Setting Marker	2.17
WDR_PSSULM	Hardware Watchdog Upper Limit PSS Setting Marker	2.18
WDR_RSTDLYM	Hardware Watchdog Reset Delay Counter Marker	2.19
WDR_CFGM	Hardware Watchdog Configuration Marker	2.20
WDR_CEM	Hardware Watchdog Configuration Enable Marker	2.21

### Memory Layout of the BootROM Markers

The BootROM markers are located in the beginning area of sector 0 (SA0) of the TCFLASH flash memory A.

**Table 2-4 BootROM Marker Area**

BootROM Markers	Offset in TCM Region	Offset in AXI Region
SDR	+0x007F_0000	+0x00FF_0000
BDR	+0x007F_0040	+0x00FF_0040
WDR	+0x007F_0060	+0x00FF_0060

## 2.3. Flash Security Marker (SDR\_FSECM)

This marker is used to set flash security.

BIT_OFFSET	31-16
BIT_NAME	Reserved

BIT_OFFSET	15-0
BIT_NAME	FSECM

**[bit31:16] Reserved: Reserved bits**

**[bit15:0] FSECM[15:0]: Flash security marker**

These bits are used to specify whether to enable flash security.

Value	Description
0x0001	Flash security enabled
Other than above	Flash security disabled



## 2.4. Debugger Connection Enable Marker (SDR\_DSM)

This marker is used to enable debugger connection.

BIT_OFFSET	31-16
BIT_NAME	Reserved

BIT_OFFSET	15-0
BIT_NAME	DSEM

**[bit31:16] Reserved: Reserved bits**

### **[bit15:0] DSEM[15:0]: Debugger connection enable marker**

These bits are used to enable debugger connection. The settings that depend on the CPU operating mode are listed below.

- For 2CPU mode

Value	Description
0x0XXX 0xX0XX 0xFF0X 0xFFFF0	Debugger connection disabled
Other than above	Debugger connection enabled

"X" means "don't care" (the value is ignored).

- For 1CPU0 mode/1CPU1 mode

Value	Description
0x59F7	Debugger connection enabled
Other than above	Debugger connection disabled

## 2.5. Debugger Security Key Marker 0 (SDR\_DSKM0)

This marker is used to set an authentication key for debugger connection.

BIT_OFFSET	31-0
BIT_NAME	DSKM[127:96]

### [bit31:0] DSKM (bit127:96): Debugger security key marker (bit127:96)

These bits are a 128-bit authentication key for debugger connection when debugger connection is enabled by Debugger Connection Enable Marker (DSEM [15:0] bits of the SDR\_DSM register). These bits are bit127:96 of the key.

The effective authentication keys that depend on the CPU operating mode are listed below.

**Table 2-5 Effective Authentication Keys that Depend on the CPU Operating Mode**

CPU Operation Mode	Effective Authentication Key
2CPU mode	<ul style="list-style-type: none"><li>- If the value of DSKM in TCFLASH0 is the same as that in TCFLASH1 The value set in DSKM is the authentication key.</li><li>- If the value of DSKM in TCFLASH0 differs from that in TCFLASH1 The exclusive OR of the DSKM in TCFLASH0 and the DSKM in TCFLASH1 is the authentication key.</li></ul>
1CPU0 mode	<ul style="list-style-type: none"><li>- The DSKM in TCFLASH0 is the authentication key.</li></ul>
1CPU1 mode	<ul style="list-style-type: none"><li>- The DSKM in TCFLASH1 is the authentication key.</li></ul>





## 2.6. Debugger Security Key Marker 1 (SDR\_DSKM1)

This marker is used to set an authentication key for debugger connection.

BIT_OFFSET	31-0
BIT_NAME	DSKM[95:64]

### [bit31:0] DSKM (bit95:64): Debugger security key marker (bit95:64)

These bits are a 128-bit authentication key for debugger connection when debugger connection is enabled by Debugger Connection Enable Marker (DSEM [15:0] bits of the SDR\_DSM register). These bits are bit95:64 of the key.

For details on the effective authentication keys that depend on the CPU operating mode, see Table 2-5.

## 2.7. Debugger Security Key Marker 2 (SDR\_DSKM2)

This marker is used to set an authentication key for debugger connection.

BIT_OFFSET	31-0
BIT_NAME	DSKM[63:32]

### [bit31:0] DSKM (bit63:32): Debugger security key marker (bit63:32)

These bits are a 128-bit authentication key for debugger connection when debugger connection is enabled by Debugger Connection Enable Marker (DSEM [15:0] bits of the SDR\_DSM register). These bits are bit63:32 of the key.

For details on the effective authentication keys that depend on the CPU operating mode, see Table 2-5.



## 2.8. Debugger Security Key Marker 3 (SDR\_DSKM3)

This marker is used to set an authentication key for debugger connection.

BIT_OFFSET	31-0
BIT_NAME	DSKM[31:0]

### [bit31:0] DSKM (bit31:0): Debugger security key marker (bit31:0)

These bits are a 128-bit authentication key for debugger connection when debugger connection is enabled by Debugger Connection Enable Marker (DSEM [15:0] bits of the SDR\_DSM register). These bits are bit31:0 of the key.

For details on the effective authentication keys that depend on the CPU operating mode, see Table 2-5.

## 2.9. Debugger Connection Wait Enable Marker (BDR\_DWEM)

Wait for debugger connection after clearing a hard reset is enabled.

BIT_OFFSET	31-0
BIT_NAME	DWEM[31:0]

### [bit31:0] DWEM[31:0]: Debugger connection wait enable marker

These bits are used to specify whether to enable wait for debugger connection after clearing a hard reset.

Value	Description
0x292D3A7B	Wait for debugger connection after clearing a hard reset is not enabled. – Debugger connection is not awaited and control is immediately handed over to a user program.
Other than above	Wait for debugger connection after clearing a hard reset is enabled.

If wait for debugger connection is enabled, control is not handed over from the BootROM software to a user program if either of the following conditions is satisfied.

- Debugger connection verification and debugger setting completion verification
- Elapse of the maximum wait time



## 2.10. Alternative Boot Vector Marker (BDR\_ABVM)

This marker can be used to set the start address of a user program.

BIT_OFFSET	31-0
BIT_NAME	ABVM

### [bit31:0] ABVM[31:0]: Alternative boot vector marker

These bits are used to set the start address of a user program.

If the Alternative Boot Vector Enable Marker (BDR\_ABVEM) is used to enable the setting of the start address of a user program, the start address set in these bits is valid. If the Alternative Boot Vector Enable Marker (BDR\_ABVEM) is not used and the setting of the start address of a user program is not enabled, the start address of the user program is 0x00800000 (fixed address).

## 2.11. Alternative Boot Vector Enable Marker (BDR\_ABVEM)

This marker is used to enable the setting of the start address of a user program.

BIT_OFFSET	31-0
BIT_NAME	ABVEM

### [bit31:0] ABVEM[31:0]: Alternative boot vector enable marker

These bits are used to enable the setting of the start address of a user program.

Value	Description
0x292D3A7B	The setting of the start address of a user program is enabled. <ul style="list-style-type: none"><li>- The start address of a user program is the value set in the Alternative Boot Vector Marker (BDR_ABVM).</li></ul>
Other than above	The setting of the start address of a user program is not enabled. <ul style="list-style-type: none"><li>- The start address of a user program is 0x00800000 (fixed address).</li></ul>



## 2.12. Hardware Watchdog Interrupt Configuration Marker (WDR\_INTM)

This marker is used to set the hardware watchdog interrupt configuration register (HWDG\_INT).

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved						RSTENM	IRQENM

BIT_OFFSET	15-0							
BIT_NAME	Reserved							

**[bit31:18] Reserved: Reserved bits**

### **[bit17] RSTENM: Reset enable marker**

This bit is a marker that controls output when a watchdog error occurs.

Value	Description
0	Generate an NMI when a watchdog error occurs.
1	Generate a reset when a watchdog error occurs.

### **[bit16] IRQENM: Prior warning interrupt enable marker**

This bit is a marker that enables a prior warning interrupt.

Value	Description
0	Do not enable a prior warning interrupt.
1	Enable a prior warning interrupt.

**[bit15:0] Reserved: Reserved bits**



**2.13.        Hardware Watchdog Trigger 0 Configuration Marker  
(WDR\_TRG0CFGM)**

This marker is used to set the hardware watchdog trigger 0 configuration register (HWDG\_TRG0CFG).

BIT_OFFSET	31-8						
BIT_NAME	Reserved						

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	WDGTRG0CFGM							

**[bit31:8] Reserved: Reserved bits**

**[bit7:0] WDGTRG0CFGM[7:0]: Watchdog trigger 0 configuration marker**

These bits are a marker that is used to set a value to be written to the hardware watchdog trigger 0 register (HWDG\_TRG0) for execution of the watchdog counter clear protection trigger sequence.





## 2.14. Hardware Watchdog Trigger 1 Configuration Marker (WDR\_TRG1CFGM)

This marker is used to set the hardware watchdog trigger 1 configuration register (HWDG\_TRG1CFG).

BIT_OFFSET	31-8
BIT_NAME	Reserved

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	WDGTRG1CFGM							

**[bit31:8] Reserved: Reserved bits**

**[bit7:0] WDGTRG1CFGM[7:0]: Watchdog trigger 1 configuration marker**

These bits are a marker that is used to set a value to be written to the hardware watchdog trigger 1 register (HWDG\_TRG1) for execution of the watchdog counter clear protection trigger sequence.

## 2.15.      **Hardware Watchdog Lower Limit RUN Setting Marker (WDR\_RUNLLM)**

This marker is used to set the hardware watchdog lower limit RUN setting register (HWDG\_RUNLLS).

BIT_OFFSET	31-0
BIT_NAME	WDGRUNLLM

### **[bit31:0] WDGRUNLLM[31:0]: Window lower limit for RUN setting marker**

These bits are a marker that is used to set the window lower limit value for RUN.



## 2.16. Hardware Watchdog Upper Limit RUN Setting Marker (WDR\_RUNULM)

This marker is used to set the hardware watchdog upper limit RUN setting register (HWDG\_RUNULS).

BIT_OFFSET	31-0
BIT_NAME	WDGRUNULM

### [bit31:0] WDGRUNULM[31:0]: Window upper limit for RUN setting marker

These bits are a marker that is used to set the window upper limit value for RUN.



**2.17.        Hardware Watchdog Lower Limit PSS Setting Marker  
              (WDR\_PSSLLM)**

This marker is used to set the hardware watchdog lower limit PSS setting register (HWDG\_PSSLLS).

BIT_OFFSET	31-0
BIT_NAME	WDGPSSLLM

**[bit31:0] WDGPSSLLM[31:0]: Window lower limit for PSS setting marker**  
These bits are a marker that is used to set the window lower limit value for PSS.



## 2.18. Hardware Watchdog Upper Limit PSS Setting Marker (WDR\_PSSULM)

This marker is used to set the hardware watchdog upper PSS setting register (HWDG\_PSSULS).

BIT_OFFSET	31-0
BIT_NAME	WDGPSSULM

### [bit31:0] WDGPSSULM[31:0]: Window upper limit for PSS setting marker

These bits are a marker that is used to set the window upper limit value for PSS.

## 2.19. Hardware Watchdog Reset Delay Counter Marker (WDR\_RSTDLYM)

This marker is used to set the hardware watchdog reset delay counter register (HWDG\_RSTDLY).

BIT_OFFSET	31-16
BIT_NAME	Reserved

BIT_OFFSET	15-0
BIT_NAME	WDGRSTDLYM

**[bit31:16] Reserved: Reserved bits**

**[bit15:0] WDGRSTDLYM[15:0]: Reset/NMI delay counter marker**

These bits are used to set the number of cycles for the delay time that is to be inserted before generation of the watchdog reset request or watchdog interrupt request (NMI).



## 2.20. Hardware Watchdog Configuration Marker (WDR\_CFGM)

This marker is used to set the hardware watchdog configuration register (HWDG\_CFG).

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved				OBSSELM			

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved						CLKSELM	

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved							

**[bit31:21] Reserved: Reserved bits**

### **[bit20:16] OBSSELM[4:0]: Watchdog counter monitor bit output selection marker**

These bits are used to set the monitor target to select any 1 bit in the watchdog counter (32 bits) as the output of the watchdog counter monitor bit.

Value	Description
00000	Bit0 is selected as the output of the watchdog counter monitor bit.
00001	Bit1 is selected as the output of the watchdog counter monitor bit.
00010	Bit2 is selected as the output of the watchdog counter monitor bit.
...	...
11111	Bit31 is selected as the output of the watchdog counter monitor bit.

**[bit15:10] Reserved: Reserved bits**

### **[bit9:8] CLKSELM[1:0]: Clock selection marker**

These bits are used to set the selection of a source clock for the watchdog counter.

Value	Description
X0	Fast-CR clock selected
X1	Slow-CR clock selected

"X" means "don't care" (the value is ignored).

**[bit7:0] Reserved: Reserved bits**

## 2.21. Hardware Watchdog Configuration Enable Marker (WDR\_CEM)

This marker is used to enable various marker settings of the hardware watchdog (WDR).

BIT_OFFSET	31-0
BIT_NAME	CEM

### [bit31:0] CEM[31:0]: Configuration Enable Marker

These bits are used to enable various markers of the hardware watchdog (WDR).

Value	Description
0x292D3A7B	The hardware watchdog is started based on the settings defined with the WDR.
Other than above	The hardware watchdog is not started given the settings defined with the WDR. The hardware watchdog operates based on the default settings.





### 3. BootROM Operation

This section explains the operation of the BootROM.

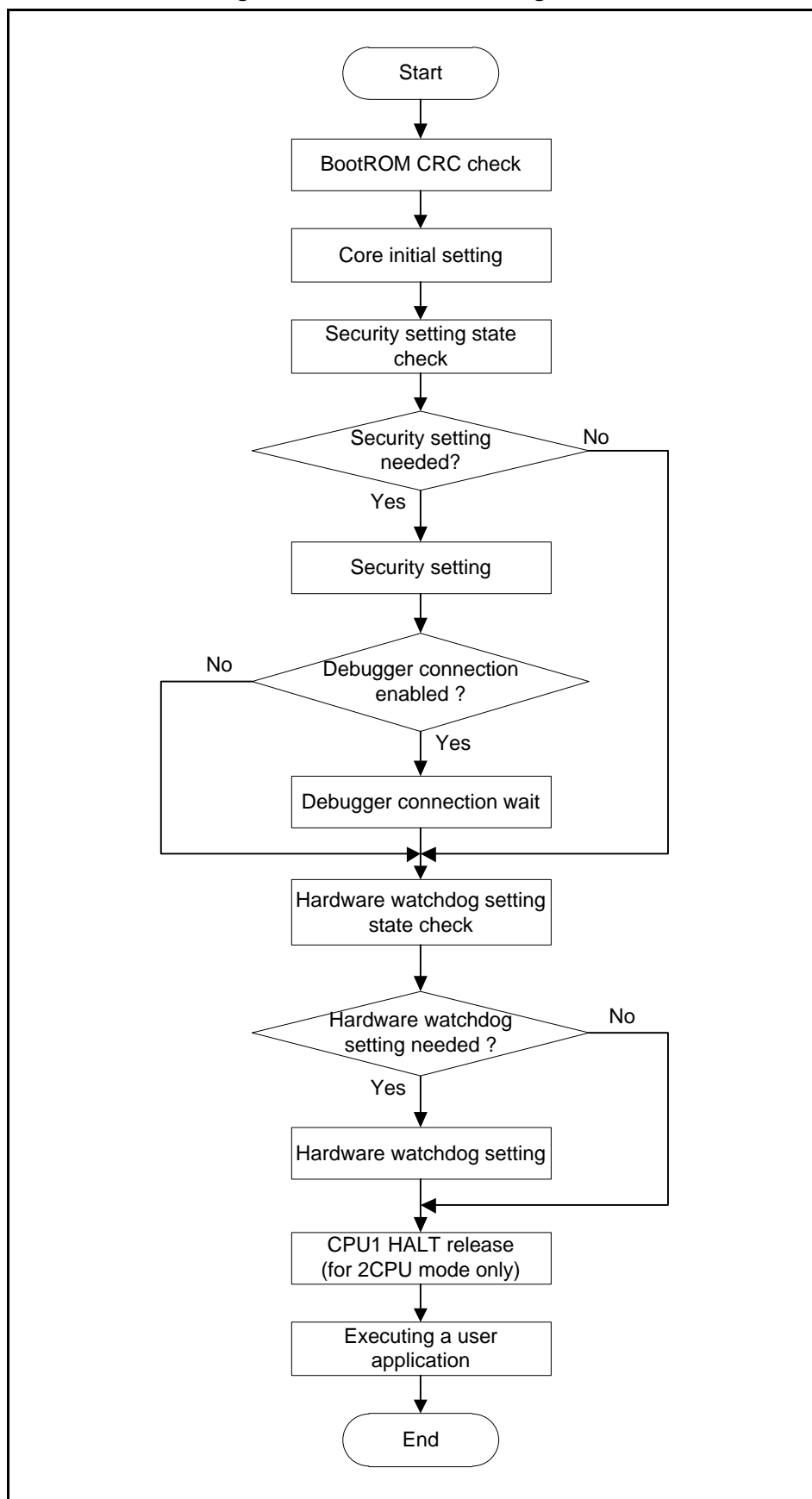
#### Starting the Device

After any type of reset, the BootROM software is performed first.

If the BootROM software selects user mode with an external pin, a user application is started.

The following shows a process flow of the BootROM software in user mode.

Figure 3-1 User Mode Processing Flow





#### a) BootROM CRC Check

This performs a CRC check on the BootROM area as the first processing by the BootROM software.

It writes BootROM data to the CRC ch.0 CRCIN register and compares the value of the CRCR register with the CRC data stored in BootROM beforehand to verify the normality of BootROM.

If a mismatch is found in the CRC check, a software trigger hard reset is issued.

#### b) Core Initial Setting

Regarding core initial setting, the general-purpose registers are initialized, the stack pointer for the mode used in the BootROM software is set, and the VIC port enable setting and FPU enable setting are made.

The modes used in the BootROM software are as below.

1. System mode
2. ABORT
3. Undefined instruction exception

#### c) Security Setting

As to the security setting, access limit setting of the flash memory, the enabling of the use of the debugger, and the security key setting are performed.

The security setting is performed using the value of the Security Description Record (SDR).

The conditions for enabling the use of the debugger differ between 2CPU mode and 1CPU mode.

The following are the conditions for enabling the use of the debugger in each mode.

- For operation in 2CPU mode only
  - The value of the Debugger Connection Enable Marker (SDR\_DSM) is other than 0x0XXX, 0xX0XX, 0XX0X, and 0XXX0.
- For operation in 2CPU mode or 1CPU mode
  - The value of the Debugger Connection Enable Marker (SDR\_DSM) is 0x59F7.
  - The value of the Debugger Security Key Marker x (SDR\_DSKMx) is other than 0x00000000 and 0xFFFFFFFF.
  - The value of the Alternative Boot Vector Enable Marker (BDR\_ABVEM) is 0x292D3A7B.
  - The value of the Alternative Boot Vector Marker (BDR\_ABVM) is in the TCFLASH or WorkFLASH area.
  - The read data of the address indicated by the Alternative Boot Vector Marker (BDR\_ABVM) is other than 0x00000000 and 0xFFFFFFFF.
  - In reading the Debugger Connection Enable Marker (SDR\_DSM), the Debugger Security Key Marker x (SDR\_DSKMx), the Alternative Boot Vector Enable Marker (BDR\_ABVEM), and the Alternative Boot Vector Marker (BDR\_ABVM), 1-bit ECC error correction has not been performed.

When the reset factor is a software reset, since the security setting is not initialized, the security setting is not made.

For access restriction by security setting, see "CHAPTER: Security".

#### d) Debugger Connection Wait

When debugger connection is enabled by the security setting and debugger connection wait is enabled by the Debugger Connection Wait Enable Marker (BDR\_DWEM) setting, the debugger connection wait processing is performed.

This processing enables debugging from the first instruction of an application.

The following shows details of the processing.

1. Whether debugger connection is enabled by the security setting is checked.
2. If the debugger connection is not enabled, the next BootROM software processing is performed without debugger connection wait being performed.
3. Debugger Connection Wait Enable Marker (BDR\_DWEM) is checked.
4. If the debugger connection wait is not enabled, the next BootROM software processing is performed without debugger connection wait being performed.
5. Whether the debugger is connected is checked.
6. If the debugger is not connected for a period of 19,200 fast-CR clock cycles, the connection wait is canceled and the next BootROM software processing is performed.
7. If the debugger is connected, whether the debugger setting is completed is checked.
8. If there is no notification of setting completion from the debugger, the wait state is canceled after 8,388,608 (2 to 23rd power) fast-CR clock cycles.

#### e) Hardware Watchdog Setting

The setting of the hardware watchdog is made according to the value of the Watchdog Description Record (WDR).

If the reset factor is a software reset, the setting of the hardware watchdog is not made.

The lock bit of the hardware watchdog is set with its initial value retained and the hardware watchdog is placed in the operating state, provided the following conditions are satisfied: The reset factor is other than a software reset and the value of the Hardware Watchdog Configuration Enable Marker (WDR\_CEM) is other than 0x292D3A7B.

The value of the Watchdog Description Record (WDR) and the lock bit are set in the hardware watchdog and the hardware watchdog is placed in the operating state provided the following conditions are satisfied: The reset factor is other than a software reset and the value of the Hardware Watchdog Configuration Enable Marker (WDR\_CEM) is 0x292D3A7B.

#### f) Executing a User Application

The BootROM software jumps to the user application in its last processing.

If the Alternative Boot Vector Enable Marker (BDR\_ABVEM) is enabled, it jumps to the value of the Alternative Boot Vector Marker (BDR\_ABVM).

If the Alternative Boot Vector Enable Marker is not enabled, the software jumps to the fixed address of 0x00800000.

Before jumping to the user application, the BootROM software clears the stack area and general-purpose registers R1 to R13 it has used.



#### g) Exception Vector Table

In the BootROM, there is a core exception vector table.

In the exception vector table, processing for branching to exception handlers is stored.

An exception handler address is specified by a register of the interrupt controller or BootROM hardware interface, and the BootROM software jumps to the value of the register.

The following table includes the registers that are referenced by the BootROM software when exceptions occur.

**Table 3-1 Reference Registers when Exceptions Occur**

Exception	Reference Register
Undefined instruction exception	EXCFG_UNDEFACT
SVC	EXCFG_SVCACT
Instruction abort	EXCFG_PABORTACT
Data abort	EXCFG_DABORTACT
NMI(FIQ)	IRC_NMIVASBR

## 4. Notes

This section provides notes regarding the BootROM.

### (1) RAM Area Used by the BootROM Software

For the execution of the BootROM software, the TCRAM area (address range of 0x00000000 to 0x000000CF) is used.

Since the area used by the BootROM software is initialized to 0 at the end of the BootROM software processing, its contents are changed from those before the reset.

### (2) Interruption Setting

The BootROM software does not set the I bit and F bit of the core internal register CPSR.

(The default value is 1, which means "Disable.")

Before using IRQ or FIQ, a user application shall set the I and F bits to 0.





## CHAPTER 13: NMI Distribution

This chapter describes NMI distribution.

---

1. Overview
2. Configuration
3. Operation
4. Setting Procedure Examples
5. Registers
6. Others





## 1. Overview

This section provides the overview of NMI distribution.

A multiple CPU configuration requires appropriate distribution of NMI from the input NMI sources to different CPUs. This requirement is satisfied by hardware through incorporation of the NMI distribution unit.

The NMI distribution unit receives NMI from the NMI sources, and distributes it to different CPUs. NMI of 32 channels is supported.

It has registers for distribution setting and an AHB 64-bit slave interface for register access.

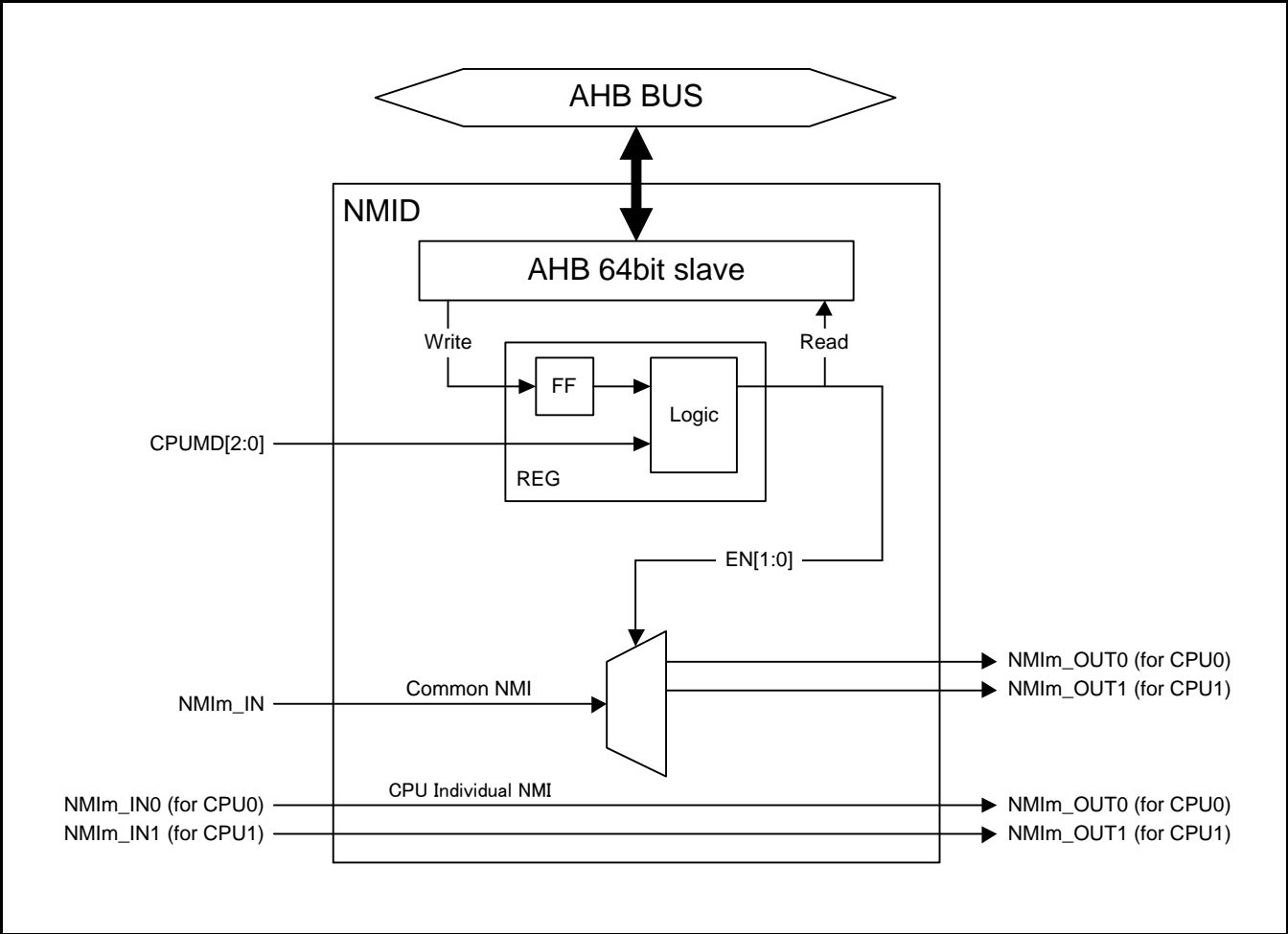
### **Notes:**

- *This chapter sometimes refers the NMI distribution unit by its abbreviation NMID (NMI Distributor).*
- *This chapter denotes each NMI channel by NMIm. "m" indicates the channel number.*
- *Abbreviated register names of the NMI distribution unit have the format of "NMID\_DISTm:ENn", where "n" is a CPU number and "m" is an NMI channel number.*

2. Configuration

This section provides a block diagram of the NMI distribution unit.

Figure 2-1 Block Diagram of NMI Distribution Unit





### 3. Operation

This section describes the operation of the NMI distribution unit.

#### (1) Sequence Protection by the Lock Release Register

The NMI distribution unit has the sequence protection feature using a lock release register (NMID\_UNLOCK). The lock must be released by writing the lock release value in NMID\_UNLOCK before distribution setting can be updated. Writing the locking value after making a distribution setting sets the locked state again.

#### (2) NMI Distribution

An NMI distribution enable register (NMID\_DISTm) is included for each NMI channel. Writing "1" in NMID\_DISTm:ENn permits NMIm to be distributed to CPU<sub>n</sub>. Writing "0" disables the distribution. An NMI must be distributed to at least 1 CPU. Setting is not permitted that disables distribution to all CPUs. Such setting is invalid and NMID\_DISTm is not updated. All channels share the same initial state, in which distribution is allowed only to CPU0.

Some NMI sources are input to particular individual CPUs. These NMI sources are called individual NMIs. A channel assigned with an individual NMI does not follow the distribution specified by the NMID\_DISTm setting, and its input is always delivered to the corresponding CPU without affected by the value written into NMID\_DISTm.

Channels assigned with individual NMIs are the following:

NMI7	SW-WDT watchdog
NMI8	Interrupt controller ECC error
NMI9	CPU live lock
NMI18	Time protection

In contrast to individual NMIs, NMIs that are distributed based on the NMID\_DISTm setting is called common NMI.

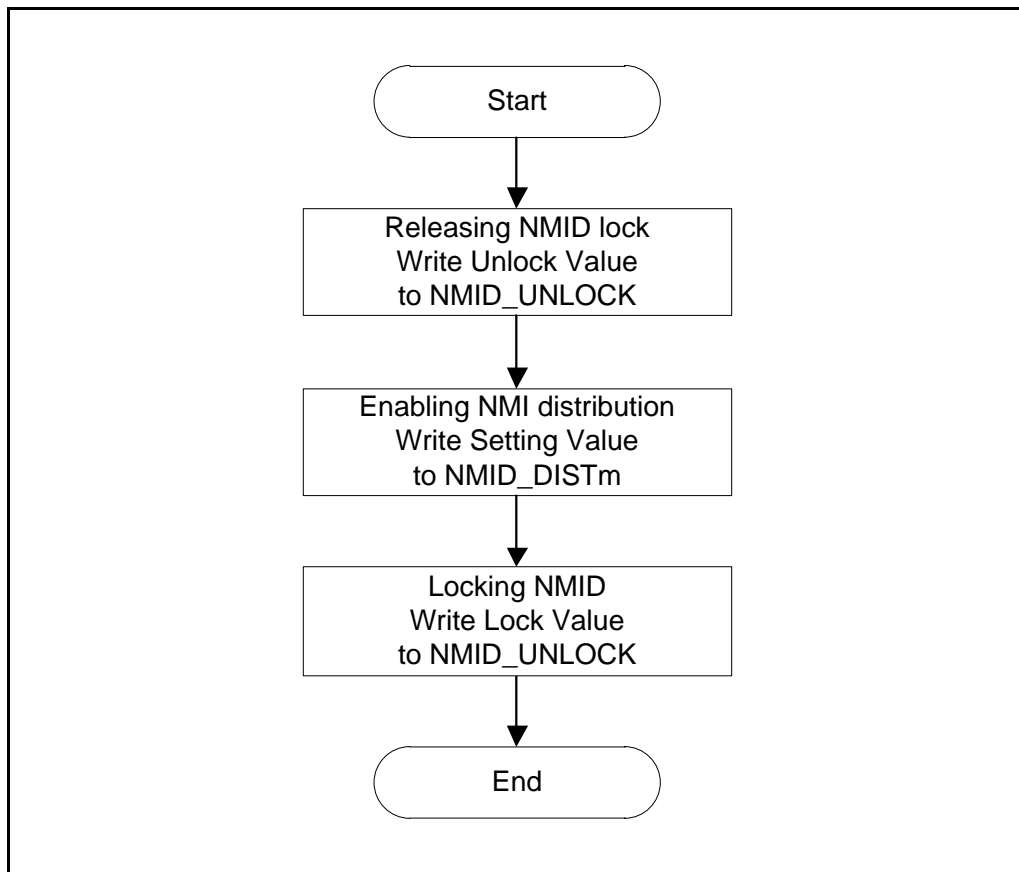
For details on NMI source assignment to different channels, see Section "List of Interrupt Factors and DMA Activation Factors" in "CHAPTER: Appedix".

The NMI distribution unit has the function to forcibly change the distribution in accordance with the CPU operation mode (SYSC\_SPECPUFCGR:CPUMD). The single CPU mode ignores the NMID\_DISTm setting and distributes signals to only 1 operational CPU. In this case, the NMID\_DISTm setting is retained, but when it is read, the returned value reflects the forcibly configured mode (the actual distribution condition), not the value being set. For details on the CPU operation mode, see "CHAPTER: Low-power Consumption".

## 4. Setting Procedure Examples

This section provides an example of the NMI distribution setting procedure of the NMI distribution unit.

Figure 4-1 NMI Distribution Setting Flow of NMI Distribution Unit





## 5. Registers

This section describes the registers used by the NMI distribution unit.

**Table 5-1 List of NMI Distribution Registers**

Abbreviated Register Name	Register Name	See
NMID_UNLOCK	NMID lock release register	5.1
NMID_LST	NMID lock status register	5.2
NMID_DISTm	NMID NMIm distribution enable register	5.3

## 5.1. NMID Lock Release Register (NMID\_UNLOCK)

This register controls the write lock of the registers of the NMI distribution unit.

BIT_OFFSET	31-0
BIT_NAME	UNLOCK
ACCESS_TYPE	R0,W
PROT_TYPE	WP
INITIAL_VALUE	00000000_00000000_00000000_00000000

### [bit31:0] UNLOCK[31:0]: Lock release bits of NMI distribution unit

These bits control the write lock of the configuration registers of the NMI distribution unit.

NMID\_DISTm is subject to sequence protection by this register.

Value	Description
0x17ACC911	Value for lock release (which enables writing)
0x17B10C11	Value for locking (which disables writing)
Other than above	Setting prohibited (bus error returned)

**Note:**

- The value of all 32 bits of this register is needed to determine the lock status. Therefore, writing to this register must be made with 32-bit access. 8-bit or 16-bit access is not allowed.



## 5.2. NMID Lock Status Register (NMID\_LST)

This register indicates the lock status of the NMI distribution unit.

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved							LST
ACCESS_TYPE	R0,WX							R,WX
PROT_TYPE	-							
INITIAL_VALUE	00000000							0

**[bit31:1] Reserved: Reserved bits**

**[bit0] LST: Lock status of the NMI distribution unit**

This bit indicates the lock status of the NMI distribution unit.

Value	Description
0	Lock release state
1	Locked state

### 5.3. NMID NMIm Distribution Enable Register (NMID\_DISTm)

This register specifies enable/disable of NMI distribution to each CPU. Registers of the same type are provided for individual NMI channels, and "m" in the abbreviated register name indicates the NMI channel number (0 to 31). Each CPU is assigned with 1 bit, and the bit position corresponds to the CPU number n (0 to 1).

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved						EN1	EN0
ACCESS_TYPE	R0,WX						R/W	R/W
PROT_TYPE	WPS							
INITIAL_VALUE	000000						0	0

**[bit7:2] Reserved: Reserved bits**

**[bit1] EN1: NMIm distribution enable/disable bit for CPU1**

**[bit0] EN0: NMIm distribution enable/disable bit for CPU0**

These bits specify enable/disable of distribution of NMI channel m to each CPU.

A write value specifies NMI distribution in the multi-CPU mode. A read value indicates the distribution state that is forcibly changed in accordance with the CPU operation mode (SYSC\_SPECPUFCFR:CPUMD). In the multi-CPU mode, the register value is indicated without change. In the single-CPU mode, the indicated value is such that the bit corresponding to the operating CPU is "1" and other bits are "0", regardless of the register setting state.

Value	Description	
	Write	Read
0	Disable distribution of NMIm to CPU <sub>n</sub> .	Actual distribution disable state of NMIm to CPU <sub>n</sub>
1	Enable distribution of NMIm to CPU <sub>n</sub> .	Actual distribution enable state of NMIm to CPU <sub>n</sub>

**Notes:**

- Writing that simultaneously sets all bits other than Reserved bits to "0" is prohibited. An attempt of prohibition setting does not update the register, and will have a bus error returned.
- The register bit configuration is the same among NMID\_DIST0 to NMID\_DIST31.





## 6. Others

This section describes the notes when using the NMI distribution unit.

Prohibition setting of NMID\_DISTm (the setting that prohibits distribution to any CPU) is determined for each channel individually. Writing in half word or larger size configures multiple channels simultaneously. In such a case, any channel with prohibition setting causes a bus error to be returned, but the configuration of channels other than the prohibition-setting channel is effective and the register is updated.

Prohibition setting can be confirmed by reading the register that has been set. The comparison of the read and write values reveals that the channels without prohibition setting have accepted the setting to update the register, showing a match in the comparison. The channel having prohibition setting has rejected the setting to keep the register unchanged, showing a difference in comparison.

An access to the register of the NMI distribution unit returns a bus error in the following conditions:

- Sequence protection violation by NMID\_UNLOCK
- Writing in NMID\_UNLOCK a value other than that of lock release or locking.
- Privilege protection error
- Read or write attempt in which access is made only to a register undefined area
- Attempt of write access to bits of WX attribute only (including register undefined area)
- Writing a prohibition setting on NMID\_DISTm



## CHAPTER 14: External Interrupt

This chapter explains the functions and operations of external interrupts.

---

1. Overview
2. Configuration
3. Operation
4. Setting Procedure Example
5. Registers



## 1. Overview

This section provides an overview of external interrupts.

### External Interrupt Functions

The external interrupt functions detect signal input to the external interrupt pin and generate interrupt requests.

- External interrupt requests available at 5 levels (H, L, rising edge, falling edge, and both (rising and falling) edges)
- 8 pins for External interrupt function
- Noise filter bypass possible
- Non-maskable interrupt supported
- DMA supported with 4 channels (ch.4 to ch.7)

2. Configuration

This section explains the configuration for external interrupts.

2.1. Block Diagrams

This section shows block diagrams of external interrupt circuits.

Figure 2-1 External Interrupt Block Diagram

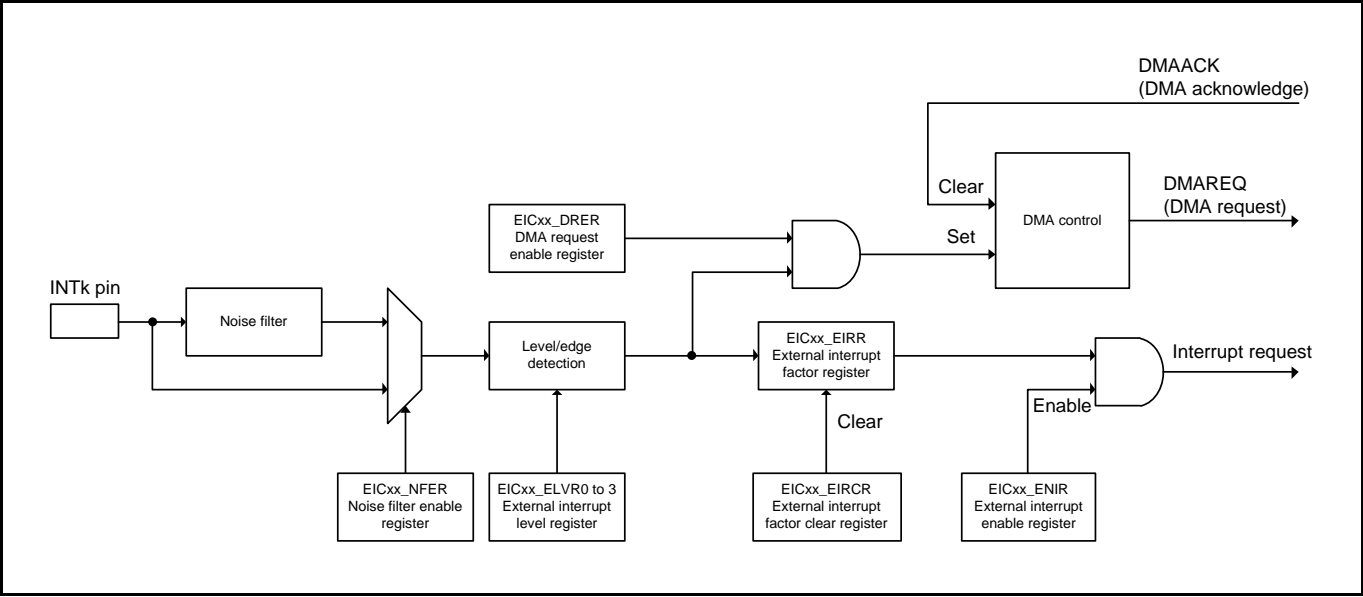
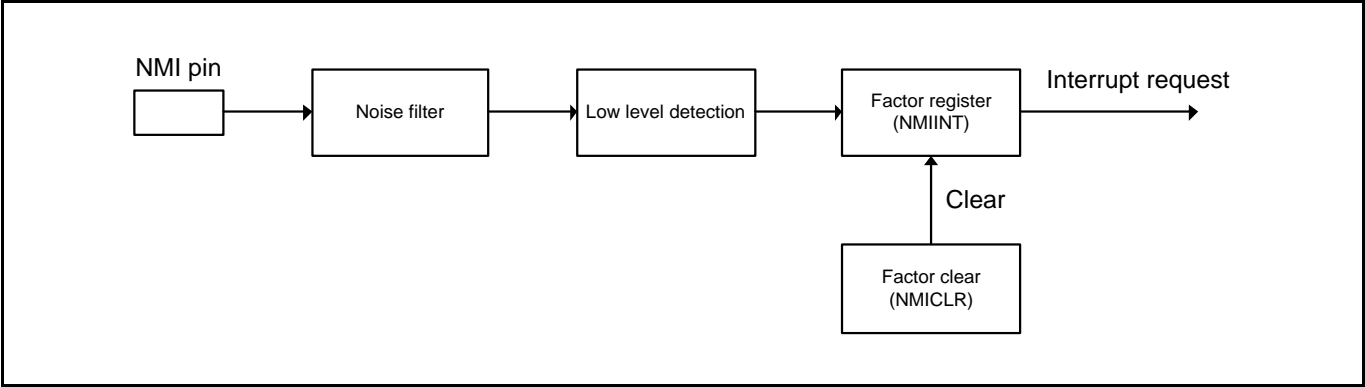


Figure 2-2 NMI (Non-maskable Interrupt) Block Diagram





### 3. Operation

This section explains the operations of external interrupts.

#### (1) Clearing of DMA Request

The DMA flag and DMAREQ are cleared by DMAACK. DMAREQ clear by DMAACK has priority over the DMAREQ settings.

#### (2) Clearing of Interrupt Flag

Clear the interrupt flag EICxx\_EIRR:ER in the interrupt handler to use it as an external interrupt. If it is not cleared after the first interrupt handler is completed, the same handler is executed again.

Even after the interrupt flag is cleared, if level detection is specified as an event input, the flag is set again as long as the input pin keeps it at the active level. In such cases, clear the requested external factor or interrupt enable bit. Clearing by software has priority over the hardware settings.

#### (3) Noise Filter

The noise filter removes noise from the signals input through the INT pin.

#### (4) External Interrupt Request Level

If edge detection is specified as an event input and the noise filter is enabled, a rule applies to the pulse width of an input signal to be recognized as an input edge. For the minimum value of the pulse width, see the MB9D560 data sheet.

If level detection is specified as an event input and the specified level is input, the interrupt flag retains even after a change of the input signal to the inactive level. See Figure 3-2. Clear the interrupt flag to clear the request.

Figure 3-1 Clearing of Interrupt Factor Register at Level Setting Time

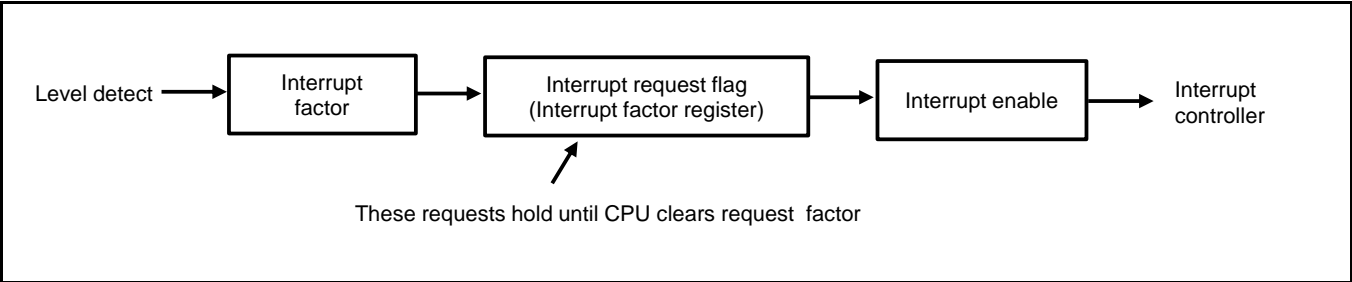


Figure 3-2 Interrupt Factor and Interrupt Request to Interrupt Controller while Interrupts are Enabled

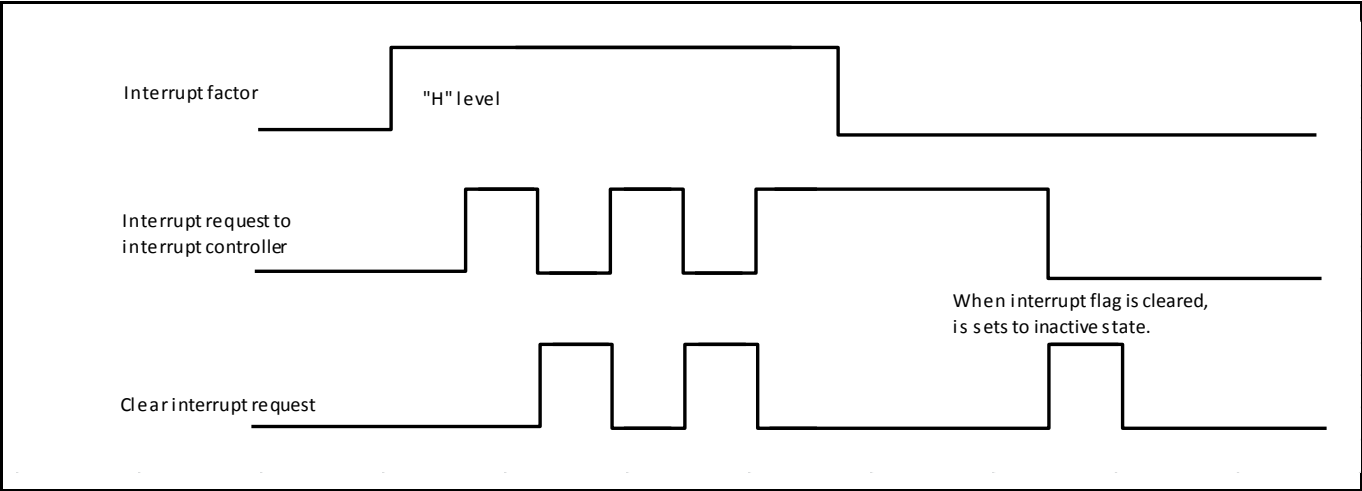
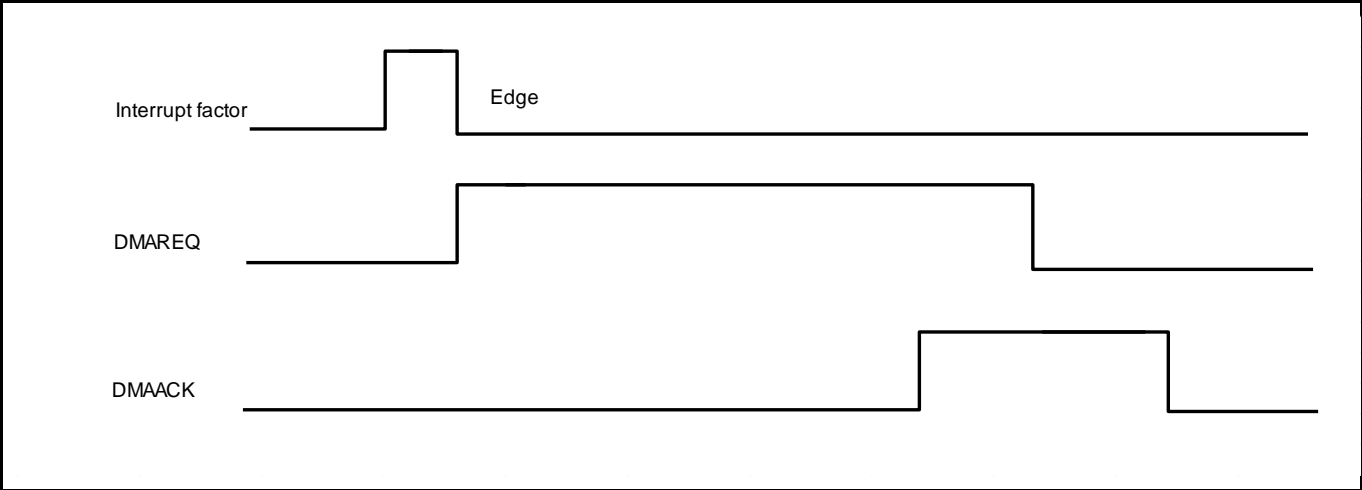


Figure 3-3 Interrupt Factor and DMAREQ to DMA while DMA is Enabled





## 4. Setting Procedure Example

This section shows an example of the external interrupt setting procedure.

### External interrupt programming procedure

Use the following procedure to configure the existing registers in the external interrupt block.

1. Set disable in the bits that are the target of the external interrupt enable register (EICxx\_ENIR).
2. Set the bits that are the target of an external interrupt level setting register (EICxx\_ELVR0 to 3) or the noise filter enable register (EICxx\_NFER).
3. Confirm the writing completion by reading external interrupt level setting register (EICxx\_ELVR0 to 3) or the noise filter enable register (EICxx\_NFER).
4. Clear the bits that are the target of the external interrupt factor register (EICxx\_EIRR).
5. Set enable in the bits that are the target of the external interrupt enable register (EICxx\_ENIR).

#### Note:

- *The enable register must be disabled before the registers in this module are configured. Also, be sure to clear the factor register before enabling the enable register. This prevents an interrupt factor from being incorrectly generated when the registers are configured or when interrupts are enabled.*

## 5. Registers

This section lists the registers.

The following table lists the external interrupt and NMI controller registers.

The additional character xx in the register name indicates the unit.

EIC00 supports channels 0 to 31.

This product has 8 channels, ch.0 to ch.7.

**Table 5-1 List of External Interrupt/NMI Controller Registers**

Abbreviated Register Name	Register Name	See
EICxx_ENIR	External Interrupt Enable Register	5.1
EICxx_ENISR	External Interrupt Enable Set Register	5.2
EICxx_ENICR	External Interrupt Enable Clear Register	5.3
EICxx_EIRR	External Interrupt Factor Register	5.4
EICxx_EIRCR	External Interrupt Factor Clear Register	5.5
EICxx_NFER	Noise Filter Enable Register	5.6
EICxx_NFESR	Noise Filter Enable Set Register	5.7
EICxx_NFECR	Noise Filter Enable Clear Register	5.8
EICxx_ELVR0 to 3	External Interrupt Level Register	5.9
EICxx_NMIR	Non-maskable Interrupt Register	5.10
EICxx_DRER	DMA Request Enable Register	5.11
EICxx_DRESR	DMA Request Enable Set Register	5.12
EICxx_DRECR	DMA Request Enable Clear Register	5.13
EICxx_DRFR	DMA Request Flag Register	5.14





## 5.1. External Interrupt Enable Register (EICxx\_ENIR)

This register is used for mask control of external interrupt factor output.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	EN15	EN14	EN13	EN12	EN11	EN10	EN9	EN8
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	EN7	EN6	EN5	EN4	EN3	EN2	EN1	EN0
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

### [bit31:0] EN31 to EN0: External interrupt enable bits

These bits set enable for external interrupts of each channel.

Value	Description
0	Disable generation of external interrupt factors.
1	Enable generation of external interrupt factors.

#### Note:

- This register can be set/cleared not only by writing "1" or "0" directly to the EICxx\_ENIR register but also by writing "1" to the EICxx\_ENISR/EICxx\_ENICR register.

## 5.2. External Interrupt Enable Set Register (EICxx\_ENISR)

This register is used for set control of the external interrupt enable register.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	ENS31	ENS30	ENS29	ENS28	ENS27	ENS26	ENS25	ENS24
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	ENS23	ENS22	ENS21	ENS20	ENS19	ENS18	ENS17	ENS16
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	ENS15	ENS14	ENS13	ENS12	ENS11	ENS10	ENS9	ENS8
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	ENS7	ENS6	ENS5	ENS4	ENS3	ENS2	ENS1	ENS0
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

### [bit31:0] ENS31 to ENS0: External interrupt enable set bits

These bits configure the set control of the EICxx\_ENIR register.

Value	Description
0	Invalid
1	Set the EICxx_ENIR:ENn bit to "1".



### 5.3. External Interrupt Enable Clear Register (EICxx\_ENICR)

This register is used for the clear control of the external interrupt enable register.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	ENC31	ENC30	ENC29	ENC28	ENC27	ENC26	ENC25	ENC24
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	ENC23	ENC22	ENC21	ENC20	ENC19	ENC18	ENC17	ENC16
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	ENC15	ENC14	ENC13	ENC12	ENC11	ENC10	ENC9	ENC8
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	ENC7	ENC6	ENC5	ENC4	ENC3	ENC2	ENC1	ENC0
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

#### [bit31:0] ENC31 to ENC0: External interrupt enable clear bits

These bits configure the clear control of the EICxx\_ENIR register.

Value	Description
0	Invalid
1	Clear the EICxx_ENIR:ENn bit to "0".

## 5.4. External Interrupt Factor Register (EICxx\_EIRR)

This register indicates the status when a pin detects an external interrupt factor. This register is read-only. Writing to this register returns a bus error.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	ER31	ER30	ER29	ER28	ER27	ER26	ER25	ER24
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	WP							
INITIAL_VALUE	X	X	X	X	X	X	X	X

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	ER23	ER22	ER21	ER20	ER19	ER18	ER17	ER16
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	WP							
INITIAL_VALUE	X	X	X	X	X	X	X	X

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	ER15	ER14	ER13	ER12	ER11	ER10	ER9	ER8
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	WP							
INITIAL_VALUE	X	X	X	X	X	X	X	X

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	ER7	ER6	ER5	ER4	ER3	ER2	ER1	ER0
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	WP							
INITIAL_VALUE	X	X	X	X	X	X	X	X

### [bit31:0] ER31 to ER0: External interrupt factor detection bits

These bits retain the detection of an external interrupt request.

Value	Description
0	No external interrupt factor is detected.
1	An external interrupt factor is detected.



## 5.5. External Interrupt Factor Clear Register (EICxx\_EIRCR)

This register is used for the clear control of the external interrupt factor register.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	ERC31	ERC30	ERC29	ERC28	ERC27	ERC26	ERC25	ERC24
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	ERC23	ERC22	ERC21	ERC20	ERC19	ERC18	ERC17	ERC16
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	ERC15	ERC14	ERC13	ERC12	ERC11	ERC10	ERC9	ERC8
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	ERC7	ERC6	ERC5	ERC4	ERC3	ERC2	ERC1	ERC0
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

### [bit31:0] ERC31 to ERC0: External interrupt factor clear bits

These bits configure the clear control of the EICxx\_EIRR register.

Value	Description
0	Invalid
1	Clear the EICxx_EIRR:ERn bit to "0".

## 5.6. Noise Filter Enable Register (EICxx\_NFER)

This register can set whether to use the noise filter for a corresponding external interrupt factor.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	NFE31	NFE30	NFE29	NFE28	NFE27	NFE26	NFE25	NFE24
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	NFE23	NFE22	NFE21	NFE20	NFE19	NFE18	NFE17	NFE16
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	NFE15	NFE14	NFE13	NFE12	NFE11	NFE10	NFE9	NFE8
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	NFE7	NFE6	NFE5	NFE4	NFE3	NFE2	NFE1	NFE0
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

### [bit31:0] NFE31 to NFE0: Noise filter enable set bits

These bits configure the noise filter control of each external interrupt.

Value	Description
0	Disable the noise filter.
1	Enable the noise filter.

#### Note:

- This register can be set/cleared not only by writing "1" or "0" directly to the EICxx\_NFER register but also by writing "1" to the EICxx\_NFESR/EICxx\_NFECSR register.



## 5.7. Noise Filter Enable Set Register (EICxx\_NFESR)

This register is used for the set control of the noise filter enable register.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	NFES31	NFES30	NFES29	NFES28	NFES27	NFES26	NFES25	NFES24
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	NFES23	NFES22	NFES21	NFES20	NFES19	NFES18	NFES17	NFES16
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	NFES15	NFES14	NFES13	NFES12	NFES11	NFES10	NFES9	NFES8
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	NFES7	NFES6	NFES5	NFES4	NFES3	NFES2	NFES1	NFES0
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

### [bit31:0] NFES31 to NFES0: Noise filter enable set bits

These bits configure the set control of the EICxx\_NFER register.

Value	Description
0	Invalid
1	Set the EICxx_NFER:NFE <sub>n</sub> bit to "1".

## 5.8. Noise Filter Enable Clear Register (EICxx\_NFECR)

This register is used for the clear control of the noise filter enable register.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	NFEC31	NFEC30	NFEC29	NFEC28	NFEC27	NFEC26	NFEC25	NFEC24
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	NFEC23	NFEC22	NFEC21	NFEC20	NFEC19	NFEC18	NFEC17	NFEC16
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	NFEC15	NFEC14	NFEC13	NFEC12	NFEC11	NFEC10	NFEC9	NFEC8
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	NFEC7	NFEC6	NFEC5	NFEC4	NFEC3	NFEC2	NFEC1	NFEC0
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

### [bit31:0] NFEC31 to NFEC0: Noise filter enable clear bits

These bits configure the clear control of the EICxx\_NFER register.

Value	Description
0	Invalid
1	Clear the EICxx_NFER:NFE <sub>n</sub> bit to "0".





## 5.9. External Interrupt Level Registers (EICxx\_ELVR0 to 3)

These registers select the level and edge of the signals to be detected as external interrupt requests.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved	LC7	LB7	LA7	Reserved	LC6	LB6	LA6
ACCESS_TYPE	R0,WX	R/W	R/W	R/W	R0,WX	R/W	R/W	R/W
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved	LC5	LB5	LA5	Reserved	LC4	LB4	LA4
ACCESS_TYPE	R0,WX	R/W	R/W	R/W	R0,WX	R/W	R/W	R/W
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved	LC3	LB3	LA3	Reserved	LC2	LB2	LA2
ACCESS_TYPE	R0,WX	R/W	R/W	R/W	R0,WX	R/W	R/W	R/W
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved	LC1	LB1	LA1	Reserved	LC0	LB0	LA0
ACCESS_TYPE	R0,WX	R/W	R/W	R/W	R0,WX	R/W	R/W	R/W
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

### [bit31:0] LC7 to LC0/LB7 to LB0/LA7 to LA0: Level selection bits for external interrupt request detection

These bits select the level and edge of the signals to be detected as external interrupt requests.

EICxx_ELVRm:LCn	EICxx_ELVRm:LBn	EICxx_ELVRm:LA n	Description
0	0	0	Input of the L level to the pin
0	0	1	Input of the H level to the pin
0	1	0	Input of a rising edge to the pin
0	1	1	Input of a falling edge to the pin
1	X	X	Both edges (rising edge and falling edge)

m=0 to 3, n=7 to 0, X: don't care

Setting	Description
EICxx_ELVR3	LC/LB/LA7
EICxx_ELVR3	LC/LB/LA6
EICxx_ELVR3	LC/LB/LA5
EICxx_ELVR3	LC/LB/LA4
EICxx_ELVR3	LC/LB/LA3
EICxx_ELVR3	LC/LB/LA2
EICxx_ELVR3	LC/LB/LA1
EICxx_ELVR3	LC/LB/LA0

Setting		Description
EICxx_ELVR2	LC/LB/LA7	INT23 interrupt level setting
EICxx_ELVR2	LC/LB/LA6	INT22 interrupt level setting
EICxx_ELVR2	LC/LB/LA5	INT21 interrupt level setting
EICxx_ELVR2	LC/LB/LA4	INT20 interrupt level setting
EICxx_ELVR2	LC/LB/LA3	INT19 interrupt level setting
EICxx_ELVR2	LC/LB/LA2	INT18 interrupt level setting
EICxx_ELVR2	LC/LB/LA1	INT17 interrupt level setting
EICxx_ELVR2	LC/LB/LA0	INT16 interrupt level setting
EICxx_ELVR1	LC/LB/LA7	INT15 interrupt level setting
EICxx_ELVR1	LC/LB/LA6	INT14 interrupt level setting
EICxx_ELVR1	LC/LB/LA5	INT13 interrupt level setting
EICxx_ELVR1	LC/LB/LA4	INT12 interrupt level setting
EICxx_ELVR1	LC/LB/LA3	INT11 interrupt level setting
EICxx_ELVR1	LC/LB/LA2	INT10 interrupt level setting
EICxx_ELVR1	LC/LB/LA1	INT9 interrupt level setting
EICxx_ELVR1	LC/LB/LA0	INT8 interrupt level setting
EICxx_ELVR0	LC/LB/LA7	INT7 interrupt level setting
EICxx_ELVR0	LC/LB/LA6	INT6 interrupt level setting
EICxx_ELVR0	LC/LB/LA5	INT5 interrupt level setting
EICxx_ELVR0	LC/LB/LA4	INT4 interrupt level setting
EICxx_ELVR0	LC/LB/LA3	INT3 interrupt level setting
EICxx_ELVR0	LC/LB/LA2	INT2 interrupt level setting
EICxx_ELVR0	LC/LB/LA1	INT1 interrupt level setting
EICxx_ELVR0	LC/LB/LA0	INT0 interrupt level setting



## 5.10. Non-maskable Interrupt Register (EICxx\_NMIR)

This register configures the non-maskable interrupt register.

BIT_OFFSET	31-16
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	WP
INITIAL_VALUE	00000000_00000000

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved							NMICLR
ACCESS_TYPE	R0,WX							R0,W
PROT_TYPE	WP							
INITIAL_VALUE	00000000							0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved							NMIINT
ACCESS_TYPE	R0,WX							R,WX
PROT_TYPE	WP							
INITIAL_VALUE	00000000							0

[bit31:9] Reserved: Reserved bits

[bit8] NMICLR: Non-maskable interrupt clear bit

This bit configures the clear control of non-maskable interrupts.

Value	Description
0	Invalid
1	Clear the EICxx_NMIR:NMIINT bit to "0".

[bit7:1] Reserved: Reserved bits

[bit0] NMIINT: Non-maskable interrupt request detection bit

This bit retains the detection of non-maskable interrupt request.

Value	Description
0	No non-maskable interrupt request is detected.
1	A non-maskable interrupt request is detected.

## 5.11. DMA Request Enable Register (EICxx\_DRER)

This register enables DMA for external interrupt requests.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	DRE31	DRE30	DRE29	DRE28	DRE27	DRE26	DRE25	DRE24
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	DRE23	DRE22	DRE21	DRE20	DRE19	DRE18	DRE17	DRE16
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	DRE15	DRE14	DRE13	DRE12	DRE11	DRE10	DRE9	DRE8
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	DRE7	DRE6	DRE5	DRE4	DRE3	DRE2	DRE1	DRE0
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

### [bit31:0] DRE31 to DRE0: DMA request enable bits

These bits set enable for DMA requests for external interrupts.

Value	Description
0	Disable DMA.
1	Enable DMA.

#### Note:

- This register can be set/cleared not only by writing "1" or "0" directly to the EICxx\_DRER register but also by writing "1" to the EICxx\_DRESR/EICxx\_DRECR register.



## 5.12. DMA Request Enable Set Register (EICxx\_DRESR)

This register is used for the set control of the DMA request enable register.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	DRES31	DRES30	DRES29	DRES28	DRES27	DRES26	DRES25	DRES24
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	DRES23	DRES22	DRES21	DRES20	DRES19	DRES18	DRES17	DRES16
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	DRES15	DRES14	DRES13	DRES12	DRES11	DRES10	DRES9	DRES8
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	DRES7	DRES6	DRES5	DRES4	DRES3	DRES2	DRES1	DRES0
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

### [bit31:0] DRES31 to DRES0: DMA request enable set bits

These bits configure the set control of the EICxx\_DRER register.

Value	Description
0	Invalid
1	Set the EICxx_DRER:DREn bit to "1".

### 5.13. DMA Request Enable Clear Register (EICxx\_DRECR)

This register is used for the clear control of the DMA request enable register.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	DREC31	DREC30	DREC29	DREC28	DREC27	DREC26	DREC25	DREC24
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	DREC23	DREC22	DREC21	DREC20	DREC19	DREC18	DREC17	DREC16
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	DREC15	DREC14	DREC13	DREC12	DREC11	DREC10	DREC9	DREC8
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	DREC7	DREC6	DREC5	DREC4	DREC3	DREC2	DREC1	DREC0
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

#### [bit31:0] DREC31 to DREC0: DMA request enable clear bits

These bits configure the clear control of the EICxx\_DRER register.

Value	Description
0	Invalid
1	Clear the EICxx_DRER:DREn to "0".



## 5.14. DMA Request Flag Register (EICxx\_DRFR)

This register indicates the status about when a DMA request is detected. This register is read-only. Writing to this register returns a bus error.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	DRF31	DRF30	DRF29	DRF28	DRF27	DRF26	DRF25	DRF24
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	WP							
INITIAL_VALUE	X	X	X	X	X	X	X	X

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	DRF23	DRF22	DRF21	DRF20	DRF19	DRF18	DRF17	DRF16
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	WP							
INITIAL_VALUE	X	X	X	X	X	X	X	X

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	DRF15	DRF14	DRF13	DRF12	DRF11	DRF10	DRF9	DRF8
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	WP							
INITIAL_VALUE	X	X	X	X	X	X	X	X

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	DRF7	DRF6	DRF5	DRF4	DRF3	DRF2	DRF1	DRF0
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	WP							
INITIAL_VALUE	X	X	X	X	X	X	X	X

### [bit31:0] DRF31 to DRF0: DMA request detection bits

These bits retain the detection of a DMA request.

Value	Description
0	No DMA request is detected.
1	A DMA request is detected.

#### Notes:

- Depending on the response from DMA, the EICxx\_DRFR register bits are cleared.
- An EICxx\_DRFR register bit is set when EICxx\_DREr is enabled and a pin detects an interrupt event.



## CHAPTER 15: Security

This chapter provides an overview of the security of this product, together with some general notes.

---

1. Overview
2. Access Restriction with Each Security
3. Notes on Security





## 1. Overview

This section explains the security function of this product.

This product offers a function for preventing the reading of the contents of flash memory by third parties and thus protects the intellectual property of the customer.

This section provides an overview of the above-mentioned security function of this product, together with some general notes.

## 2. Access Restriction with Each Security

This section describes flash memory and debugger access restriction.

This product can prevent the reading of data in flash memory by third parties by means of the access restrictions described below.

- Restriction on access to flash memory in other than user mode (flash security)
- Password authentication for enabling the use of the debugger in user mode (debugger security)

Table 2-1 describes the restriction on access to flash memory and the restriction on enabling debugger use.

**Table 2-1 Access Restriction in Each Mode**

Operation Mode/ Security Setting		User Mode/ Security ON	User Mode/ Security OFF	Other than User Mode <sup>*1</sup> /Security ON	Other than User Mode <sup>*1</sup> /Security OFF
Access to flash memory	Macro erase (TCFLASH)	Not restricted	Not restricted	Restriction on erasure order <sup>*2</sup>	Not restricted
	Macro erase (WORKFLASH)	Impossible	Impossible	Possible <sup>*4</sup>	Impossible
	Sector erase	Not restricted	Not restricted	Ignored	Not restricted
	Write	Not restricted	Not restricted	Ignored	Not restricted
	Read	Not restricted	Not restricted	0xFFFFFFFF is always read.	Not restricted
Enabling of debugger use		Following 2 ways Enabled after password authentication Not enabled <sup>*3</sup>	Enabled	-	-

\*1: Writer mode, manufacturer test mode, etc.

\*2: There is a restriction on the erase order. Erase the flash memory to which security settings were written most recently. (Flash memory A of TCFASH#0 and TCFASH#1 can be subject to macro erasure after the macro erasure command for another flash memory is completed.) For details, see "CHAPTER: TCFASH".

\*3: Can be enabled after password authentication with software processing.

\*4: The exact method of the macro erase operation in non-user mode will not be made public.



### 3. Notes on Security

This section provides general notes on security.

The security function of this product can prevent Flash memory from being read by third parties, bearing the following general notes on security in mind.

#### **a) Enabling of Debugger Use and Password Setting**

When turning security ON, disable the debugger or set the password for enabling debugger use.

#### **b) Restriction on Total Flash Memory Erasure (Macro Erasure)**

This product does not place any restrictions on total flash memory erasure (macro erasure). The reasons for this are as follows.

- To make it possible to return the product to the state existing immediately after its purchase by using a writer to perform total erasure.
- Restricting total flash memory erasure does not contribute to the "inhibition of reading by third parties."

#### **c) Security Marker Setting**

To reinforce security, set the markers for both TCFLASH#0 and TCFLASH#1.



## CHAPTER 16: TCFLASH

This chapter explains the functions and operations of TCFLASH.

---

1. Overview
2. Configuration
3. Operation
4. Setting Procedure Example
5. Registers
6. Others



## 1. Overview

This section explains the features of TCFLASH.

### Features of TCFLASH

- TCFLASH is a flash memory to mainly store programs. If this product is in user mode, it is mapped to 2 regions, the TCM region and the AXI region. If accessed via the TCM region, TCFLASH is treated as an L1 memory on the ARM(R) architecture. Therefore, non-cacheable and low-latency access is possible. If TCFLASH is accessed via the AXI region, it is treated as an L2 memory in the context of ARM architecture.
- TCFLASH can be extended to up to 16 MB, logically. However, only 8 MB of it can be accessed via the TCM region. If it is accessed via the AXI region, 16 MB of the entire region can be accessed.
- Programming or erasing in TCFLASH is performed if a program sequence is sent when it is accessed via the AXI region. Programming or erasing in TCFLASH cannot be performed when it is accessed via the TCM region.
- The AXI region in TCFLASH is dedicated to operations during flash memory programming/erasing.
- Data can be read from the CPU in units of 8 bits/16 bits/32 bits/64 bits.
- If ECC is enabled, programming from the CPU is possible only in 32-bit. If ECC is disabled, programming from the CPU is possible in 8-/16-/32-bit.
- Non-aligned read for the access size is supported. However, non-aligned programming for the access size is not supported. If non-aligned programming for the access size is detected, TCFLASH responds by issuing a bus error.
- Wait cycle count that is inserted during data access can be specified.
- Interleaved access to large sector areas by two flash memories is possible. However, interleaved access to small sector areas is not performed.
- ECC is supported with the same calculation formula as that for the ARM Cortex(TM)-R5F core, realizing 1-bit error correction and 2-bit error detection.
- The ECC check for reading via the TCM region is performed with a logic in the ARM Cortex-R5F core.
- ECC logic check can be performed by inserting an error. Error injection is valid for both reading via the TCM region and reading via the AXI region.
- Register setting values are protected using a protection key.
- Programming and erasing are possible only when access is in privileged mode.
- The flash security function is implemented.
- Bus error response is sent when access is to the reserved area.

## 2. Configuration

This section explains the configuration within TCFLASH.

### 2.1. Block Diagrams

- There is 1 TCFLASH for each Cortex-R5F core (TCFLASH has 2 flash memories).
- When Cortex-R5F core accesses via the TCM region, TCFLASH is handled as a read-only L1 memory. On the other hand, when Cortex-R5F core accesses via the AXI region, TCFLASH is handled as an L2 memory that can both write and read.

**Figure 2-1 Position of TCFLASH**

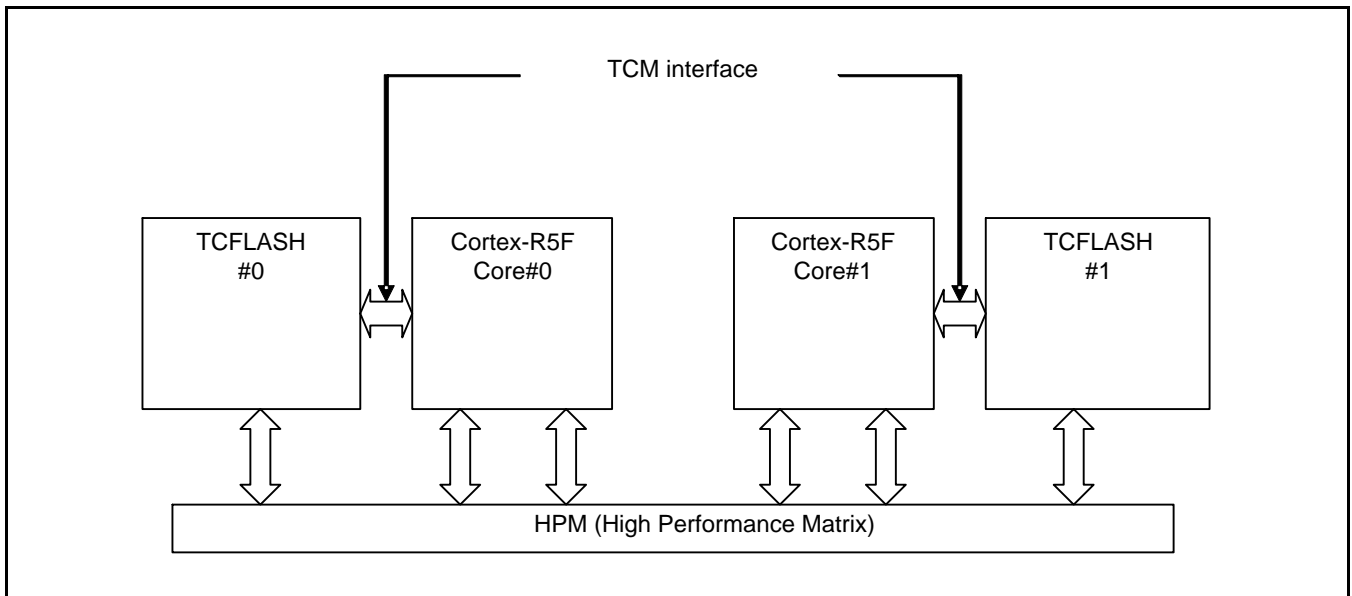
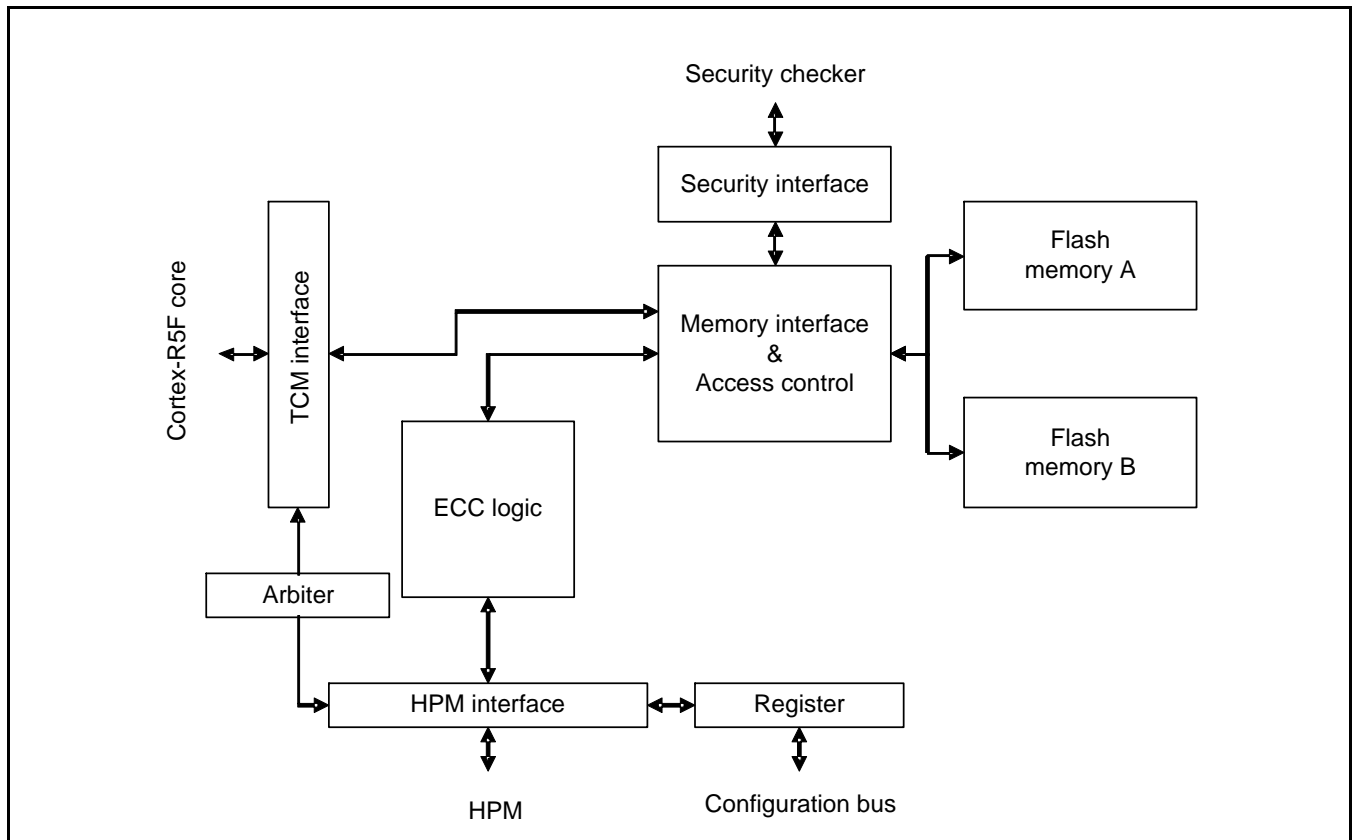


Figure 2-2 Configuration of TCFLASH



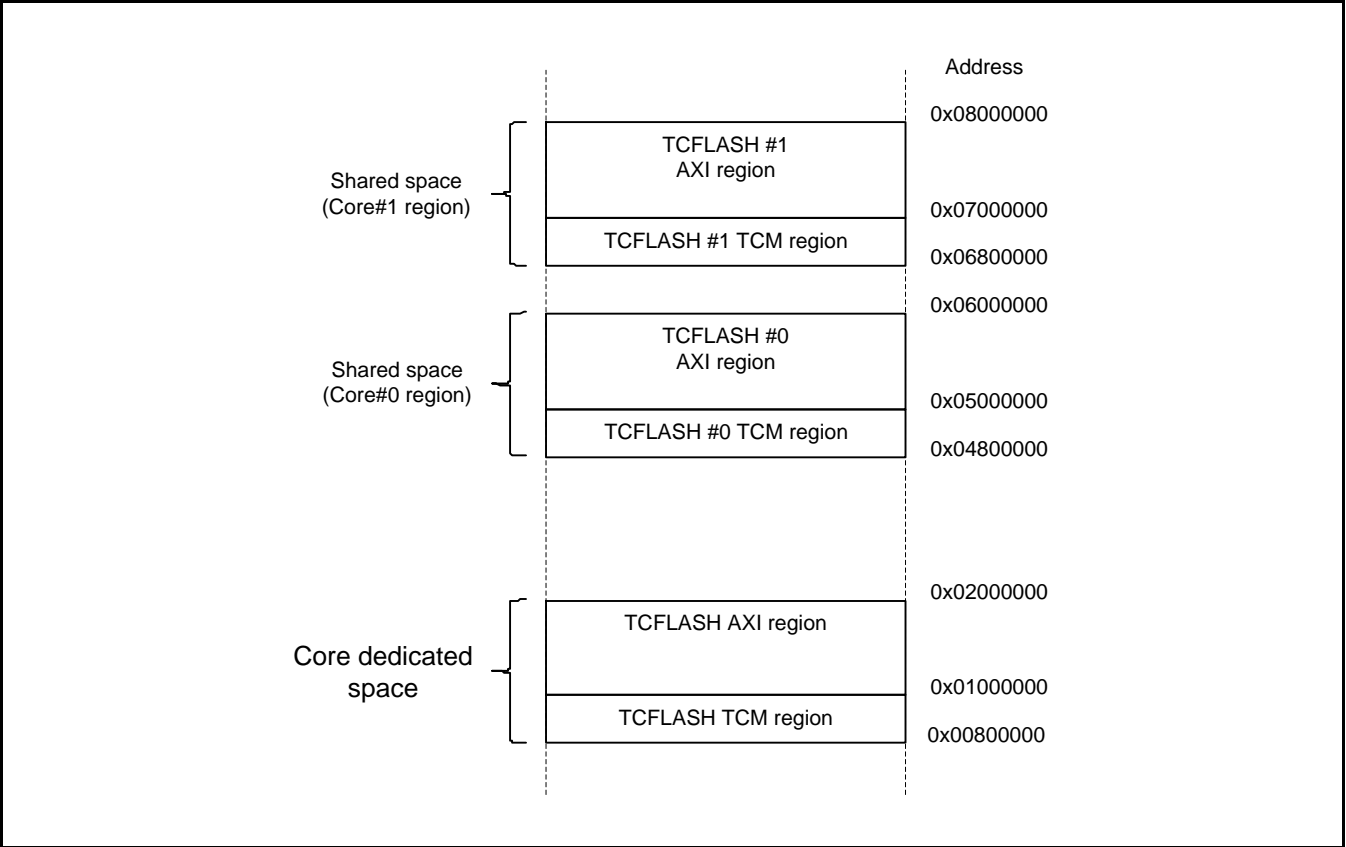
2.2. Address/Sector Map of TCFLASH

This product has 2 CPU cores.

A TCFLASH exists for each CPU core and is placed on the memory map by the program as shown in Figure 2-3.

The AXI region in TCFLASH is dedicated to operations during flash memory programming/erasing. The TCM region is used during normal operation.

Figure 2-3 Position of TCFLASH on the Memory Map

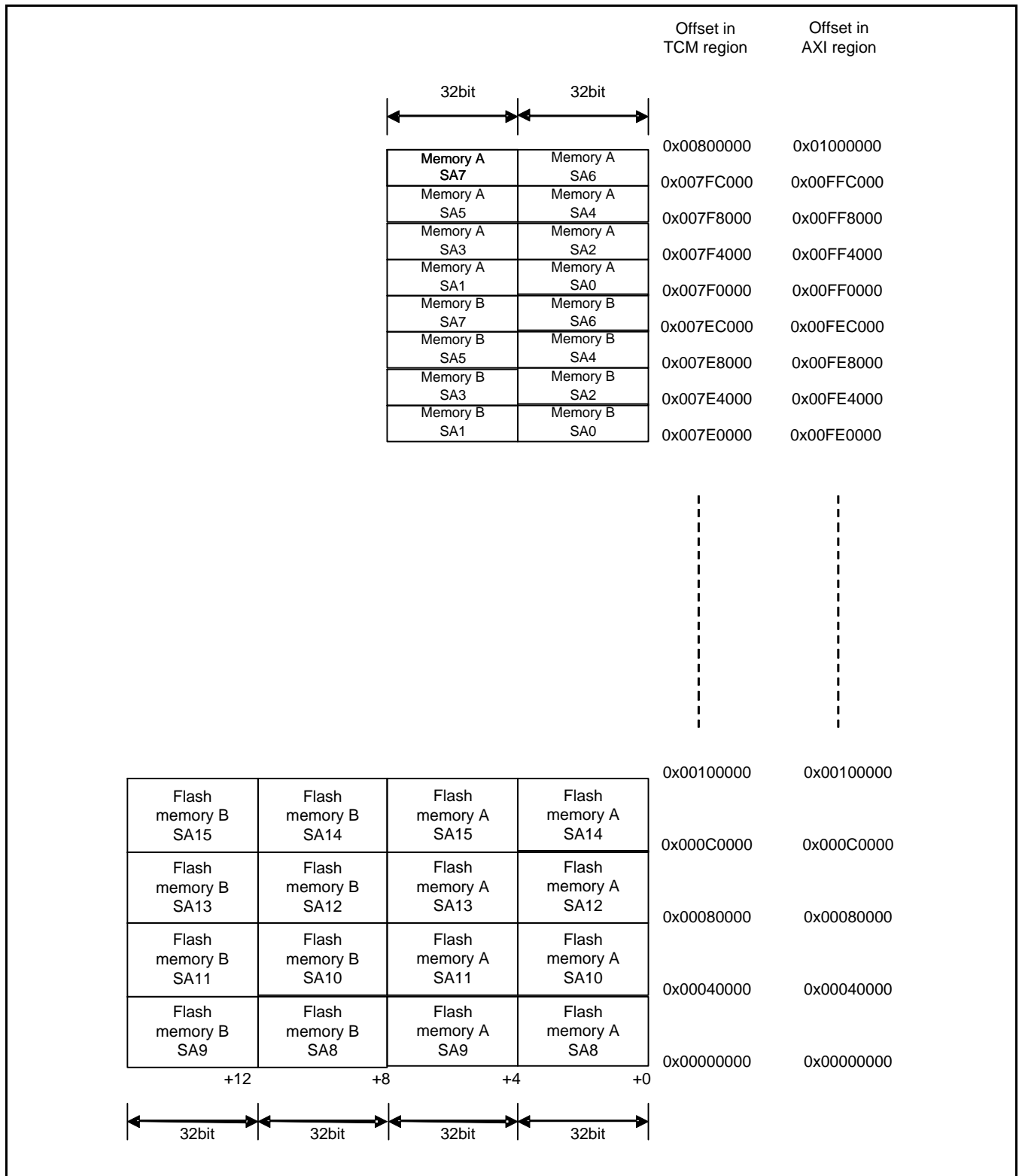


The size of the TCM region for each TCFLASH is 8 MB. The size of the AXI region is 16 MB. The sector placement for the respective region is shown in Figure 2-4.





Figure 2-4 Address/Sector Map of TCFLASH



### 3. Operation

This section explains the operation of TCFLASH.

#### 3.1. Operation Mode of TCFLASH

If this product is in user mode, the CPU or other bus master can access TCFLASH. A Cortex-R5F core can access TCFLASH that is connected to itself, via the TCM interface or via AXI. It can also access TCFLASH that is connected to another Cortex-R5F core, via AXI or via TCM from the AXI slave port of the Cortex-R5F core to which TCFLASH is connected. A bus master (non-Cortex-R5F core) such as DMAC can also access TCFLASH, via AXI or via TCM from the AXI slave port of the Cortex-R5F core to which TCFLASH is connected.

In user mode, programming or erasing in TCFLASH via AXI interface can be performed. However, programming or erasing via the TCM interface cannot be performed. Programming and erasing are performed by executing a program access sequence to start an automatic algorithm. ECC is generated in programming by using the TCFLASH interface.<sup>\*1</sup>

\*1: Do not perform the read operation via the TCM interface if you have run the program access sequence via AXI. Otherwise, a bus error occurs and an undefined value is output.

#### 3.2. Programming and Erasing

If this LSI is in user mode, programming and erasing in TCFLASH are performed by writing the programming access sequence in the flash memory via the AXI region.

For the programming access sequence, see Section "3.5.1. Command Sequence". For the programming procedure, see Section "4.3. Programming Procedure". For macro erase in non-user mode, restriction of the erasing order exists. For the order, see Section "4.7. Flash Security Unlocking Method".

- If ECC is enabled, only programming in 32-bit is possible. If ECC is disabled, programming in 8-/16-/32-bit is possible.
- To generate and write ECC, 32-bit programming must be repeated twice for the same address and data.
- When ECC is enabled and 32-bit programming is performed, ECC data writing is controlled by the WR32F bit in the TCFCFGn\_FSTAT0/1 register. If the WR32F bit value is "0", only the lower 16 bits of the 32-bit data are programmed. In this case, ECC is neither generated nor written. On the other hand, if the WR32F bit value is "1", ECC is generated from the 32-bit data and then written together with the upper 16 bits of the data. To prevent an error from being detected during the read operation when only the lower 16 bits are written, an ECC check is not performed if the WR32F bit value is "1".
- If the WE bit value in the TCFCFGn\_FCFGR register is "0" (the programming of TCFLASH is disabled), the value of the WR32F bit in the TCFCFGn\_FSTAT0/1 register of the same unit is reset to "0".

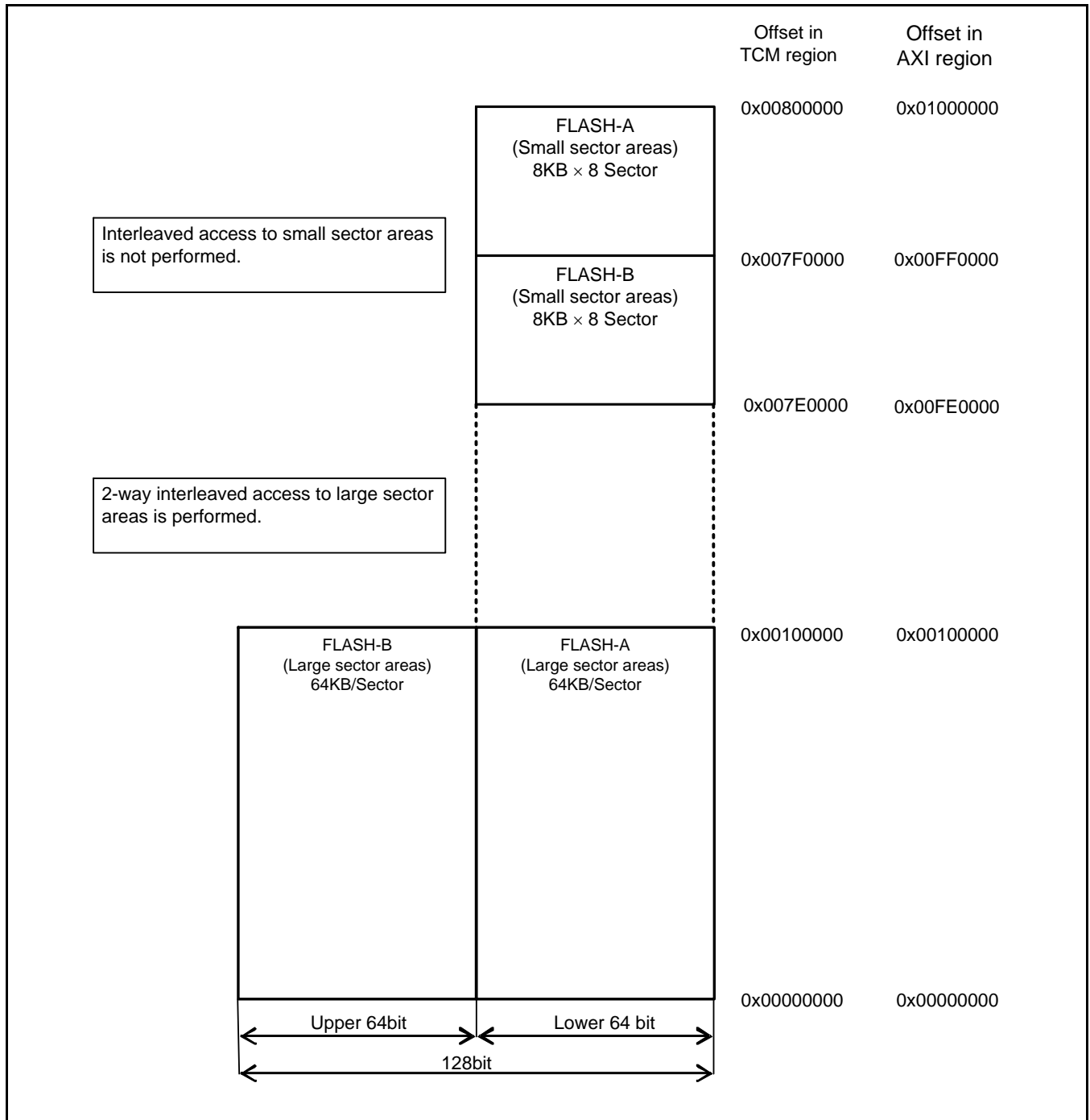


### 3.3. Interleaved Access

2-way interleaved access using flash memory A and flash memory B is performed for large sector areas in TCFLASH. Interleaved access is performed in units of 8 bytes because flash memory has a bus width of 64 bits. A 64-bit word in flash memory A is at an even-number-th address ( $16n + 0$ ,  $n = 0, 1, 2, \dots$ ). A 64-bit word in flash memory B is at an odd-number-th address ( $16n + 8$ ,  $n = 0, 1, 2, \dots$ ).

Interleaved access to small sector areas is not performed.

Figure 3-1 Interleave



### 3.4. TCM Region and AXI Region

TCFLASH has a TCM-connected region and an AXI-connected region.

The AXI region is dedicated to operations during flash memory programming/erasing. The operations during programming/erasing are performed with TCFCFGn\_FCFGR.WE set to "1".

In this case, the following operations are possible for the AXI region in TCFLASH.

- Executing an automatic algorithm
  - Reset
  - Read
  - Write
  - Macro erase
  - Sector erase
- Reading the hardware sequence flag

If the WE bit in the TCFCFGn\_FCFGR register is "1", programming TCFLASH is enabled. In this case, access via the TCM region causes a bus error.

The TCM region is used for reading during normal operation. For reading from a bus master (non-the Cortex-R5F core) that is connected to the TCM region itself, access is made via TCM from the AXI slave port of the Cortex-R5F core to which TCFLASH is connected.



### 3.5. Automatic Algorithm

Programming and erasing can be performed by sending the program access sequence to flash memory to start an automatic algorithm. The available commands in the automatic algorithm include reset, read, program, macro erase, and sector erase. For the sector erase command, it is possible to control the suspension and resumption of its execution.

#### 3.5.1. Command Sequence

To start an automatic algorithm, perform operation to program given data to a given address once to six times consecutively, depending on the command type.

Table 3-1 Command Sequence List

Command	Number of Write Operations	1st Time		2nd Time		3rd Time		4th Time		5th Time		6th Time	
		Addr ess	Data	Addr ess	Data	Addr ess	Data	Addr ess	Data	Addr ess	Data	Addr ess	Data
Reset	1	CA0	0xF0	-	-	-	-	-	-	-	-	-	-
Read	1	RA	RD	-	-	-	-	-	-	-	-	-	-
Program	4	CA0	0xAA	CA1	0x55	CA0	0xA0	PA	PD	-	-	-	-
Macro Erase	6	CA0	0xAA	CA1	0x55	CA0	0x80	CA0	0xAA	CA1	0x55	CA0	0x10
Sector Erase	6	CA0	0xAA	CA1	0x55	CA0	0x80	CA0	0xAA	CA1	0x55	SA	0x30
Sector Erase Suspend	1	SA	0xB0	-	-	-	-	-	-	-	-	-	-
Sector Erase Resume	1	SA	0x30	-	-	-	-	-	-	-	-	-	-

**Note:**

- Set the size of the command data according to the size of program data "PD".

Table 3-2 Address in the Command

Operation Mode	Code (Table 3-1)	Small Sector	Large Sector (Memory A)	Large Sector (Memory B)
User Mode	CA0	0x****1550	0x****2AA0	0x****2AA8
	CA1	0x****0AA8	0x****1550	0x****1558

- Specify the values shown in Table 3-2 for "CA0" and "CA1" in Table 3-1.
- "\*\*\*\*\*" of Table 3-2 is an arbitrary value that specifies the address range used by the flash memory on which to execute the command. For the memory map, see Section "2.2. Address/Sector Map of TCFLASH".
- Except for the program data which is represented as "PD" in the table, the upper 24 bits (bit31 to bit8) of data to be programmed to start an automatic algorithm are ignored.
- The address PA given upon the 4th write cycle is the address in which program data PD is written.
- Program address PA for 32-bit programming must be a 4-byte aligned value.
- The address SA given upon the 6th program cycle of the sector erase command indicates the address of the sector to be erased. SA is specified in the same format as PA.
- Flash memory is reset and transitions to read mode if an invalid address or data set is programmed as a command or if commands are programmed in the wrong order.
- The read operation from flash memory is possible even when the command sequence is being programmed. The automatic algorithm starts upon completion of the last cycle of the program sequence.
- If the program size is 8 or 16 bits, an ECC bit is not written.
- Flash memory must not be read-accessed until the status register RDY changes to "1" after the sector erase suspend command is issued.



## 3.6. Automatic Algorithm Execution State

This section explains the hardware sequence flag. The detailed state of flash memory during the execution of an automatic algorithm can be confirmed from the hardware sequence flag.

### 3.6.1. Hardware Sequence Flag

If flash memory is read during the execution of an automatic algorithm, the hardware sequence flag is read instead of the data. The state of programming and erasing operations that are being executed can be confirmed by the hardware sequence flag.

The hardware sequence flag consists of 16-bit data. This is repeatedly extended 4 times to 64-bit data by the flash memory interface in TCFLASH, and is then output. Therefore, software reads the hardware sequence flag from the same address as that where the data was written and checks the sequence execution state.

The sequence hardware flag consists of DQ[7, 15], DQ[6, 14], DQ[5, 13], DQ[3, 11], and DQ[2, 10]. The function of each of them is as follows.

- DQ[7, 15] is for data polling. The inverted value of DATA[7, 15], written into the same bit position, is read.
- DQ[6, 14] is toggle bit 1. During the execution of an automatic algorithm, the value is inverted whenever it is read.
- DQ[5, 13] indicates whether the automatic algorithm exceeds the timing limit.
- DQ[3, 11] indicates whether an erase operation has started in flash memory after the sector erase command is written. Once the sector erase operation has started in flash memory, "1" is read from DQ[3, 11].
- DQ[2, 10] is toggle bit 2. The value is inverted whenever the sector of the suspended erase is read.

**Table 3-3 Bit Assignment of Hardware Sequence Flag**

Bit Position	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Flag Name	DQ15	DQ14	DQ13	-	DQ11	DQ10	-	-	DQ7	DQ6	DQ5	-	DQ3	DQ2	-	-

The values in DQ[4, 12], DQ[1, 9], and DQ[0, 8] are undefined. For software using hardware sequence flag values, set the program to ignore the values in these flags.

When performing 32-bit programming, the hardware sequence flag indicates the programming state of the lower 16 bits during the 1st execution of the sequence. It shows the programming state of the upper 16 bits during the 2nd execution of the sequence.

The data polling flag is provided separately for each byte position and indicates whether the writing to the byte position has ended. First, the inverted value of the written data is output. When it is finished, the same value as that of the written data is output. Other hardware sequence flags show the status of all the flash memory and the same value is read regardless of the byte position.

The hardware sequence flag values can be read regardless of the value of the WE bit in the TCFCFGn\_FCFGR register.

When reading the hardware sequence flag, an ECC check is not performed.

You can confirm that the automatic algorithm is being executed by reading the RDY bit in the TCFCFGn\_FSTAT0/1 register. However, more detailed information can be obtained by reading the hardware sequence flag.

After execution of the automatic algorithm has ended, the flash memory transitions to the state in which the next command can be received. Before performing the next operation, confirm that the previous operation has ended by reading the RDY bit in the TCFCFGn\_FSTAT0/1 register or the hardware sequence flag values for the target flash memory. In addition, when the sector erase command sequence is written multiple

consecutive times, the hardware sequence flag can be used to confirm whether the command sequence has been received for the 2nd and subsequent times.

Table 3-4 shows the relationship between the hardware sequence flag values and the flash memory state.

**Table 3-4 Values of Hardware Sequence Flag**

State		DQ [7, 15]	DQ [6, 14]	DQ [5, 13]	DQ [3, 11]	DQ [2, 10]	HANG
Reset		DATA [7, 15]	DATA [6, 14]	DATA [5, 13]	DATA [3, 11]	DATA [2, 10]	0
Command		DATA [7, 15]	DATA [6, 14]	DATA [5, 13]	DATA [3, 11]	DATA [2, 10]	0
Program		/DATA	T	0	0	0	0
Macro erase		0	T	0	1	T	0
Sector Erase Wait Period		0	T	0	0	T	0
Sector erase		0	T	0	1	T	0
Sector Erase Suspend	Sector and Adjacent Sector Read During Erasure	0	0	0	1	T	0
	Non-target sector read during erasure	DATA [7, 15]	DATA [6, 14]	DATA [5, 13]	DATA [3, 11]	DATA [2, 10]	0
Hang-up 1	Program	/DATA	T	1	0	0	1
	Macro erase	0	T	1	1	T	1
	Sector erase	0	T	1	1	T	1

**Notes:**

- "DATA" in Table 3-4 represents a value read from flash memory.
- "/DATA" in Table 3-4 represents the inverted value of bit 7 of each byte of the program data PD.
- "T" in Table 3-4 indicates that a value is inverted whenever it is read.
- The HANG bit in the TCFCFGn\_FSTAT0/1 register indicates whether flash memory is in the hang-up 1 state.
- The hardware sequence flag values depend on the command that caused the hang-up.





### 3.6.2. Data Polling Flag

The data polling flag represents whether the automatic algorithm is being executed.

#### (1) Data Polling Flag (DQ[7, 15])

The value shown in the DQ[7, 15] column in Table 3-4 is read in the data polling flag depending on the flash memory state.

#### (2) Program

If flash memory is read during the execution of programming when using an automatic algorithm, inverted values of bit 7 and 15 of data being written in the flash memory are read regardless of the address. Once the programming has been completed, a normal read operation is performed to read the written data. Therefore, to correctly confirm that the programming has been completed, read the data polling flag using the same address as that specified for the programming.

#### (3) 16-bit and 8-bit Programming

During the execution of 16-bit programming, DQ7 contains the inverted value of bit 7 of the program data PD. DQ15 contains the inverted value of bit 15 of PD. When 8-bit programming is executed with an even byte address specified, the inverted value of bit 7 of PD is read into DQ7 during the execution of the programming. Similarly, when 8-bit programming is executed with an odd byte address specified, the inverted value of bit 15 of PD is read into DQ15 during the execution of the programming.

#### (4) Macro Erase and Sector Erase

During the execution of an erase operation using an automatic algorithm, "0" is read in the data polling flag regardless of the address. The cell value of the erased flash memory becomes "1". Therefore, when erase is complete, the erased cell value is read as "1".

#### (5) Sector Erase Suspend

When sector erase is suspended, if you read an address which is in the sector to be erased, "0" is read in the data polling flag. However, if you read an address which belongs to other than the sector to be erased, the contents written to the address are output.

Whether sector erase is suspended can be determined by combining the data polling flag value and the toggle flag value. While sector erase is suspended, it is possible to determine which sector is to be erased.

### 3.6.3. Toggle Bit Flag

The toggle bit flag is used in combination with the data polling flag to determine whether the automatic algorithm is being executed. To confirm that the value of the toggle bit flag is being toggled, take this reading at least twice and then compare the values.

#### (1) Toggle Bit Flag (DQ[6, 14])

The value shown in the DQ[6, 14] column in Table 3-4 is read in the toggle bit flag depending on the flash memory state.

#### (2) Program, Macro Erase, and Sector Erase

The value of the toggle bit flag during the execution of an automatic algorithm is inverted whenever it is read. After the execution of the automatic algorithm has been completed, the value written to the specified address is read by a read operation. Therefore, after the erase has been completed, "1" is read.

#### (3) Sector erase suspend

When sector erase is suspended, if you read an address which is in the sector to be erased, "0" is read in the toggle bit flag. On the other hand, if the read address is not in the sector to be erased, the value written to the address is read.

### 3.6.4. Timing Limit Exceeded Flag

The timing limit exceeded flag indicates that the timing limit set in flash memory is exceeded during the execution of an automatic algorithm.

#### (1) Timing Limit Exceeded Flag (DQ[5, 13])

The value shown in the DQ[5, 13] column in Table 3-4 is read in the timing limit exceeded flag depending on the flash memory state.

#### (2) Program, Macro Erase, and Sector Erase

For a program or erase operation using an automatic algorithm, the execution time limit of the automatic algorithm is set in flash memory. If the automatic algorithm is being executed and does not exceed the execution time limit, "0" is read in the timing limit exceeded flag. If it exceeds the execution time limit, "1" is read.

A failure to execute a program or erase operation is detected by determining that the automatic algorithm is being executed and then finding that the timing limit exceeded flag has changed to "1" during execution using the data polling flag and toggle bit flag.

This status can also be found by using the HANG bit in the TCFCFGn\_FSTAT0/1 register.

If the timing limit exceeded is detected, execute the read/reset command.

#### (3) Sector Erase Suspend

While sector erase is suspended, if you read an address which belongs to the sector to be erased, the timing limit exceeded flag reads "0".



### 3.6.5. Sector Erase Timer Flag

The sector erase timer flag indicates whether a sector erase operation has started in flash memory.

#### (1) Sector Erase Timer Flag (DQ[3, 11])

The value in the DQ[3, 11] column shown in Table 3-4 is read in the sector erase timer flag depending on the flash memory state.

#### (2) Sector Erase

After writing the sector erase command sequence, the flash memory enters the sector erase wait period. During the sector erase wait period, you can additionally specify a sector to be erased by writing another sector erase command.

If a read operation is executed during the execution of the command, the sector erase timer flag is read regardless of the specified address. During the sector erase wait period, "0" is read in the sector erase timer flag. When the actual erase operation has started in flash memory after the sector erase wait period, the sector erase timer flag reads "1". While "1" is read in the sector erase timer flag, the writing of another sector erase command is ignored.

Before and after writing the sector erase command to flash memory, check the value of the sector erase timer flag. If "1" is read in the sector erase timer flag immediately after writing the sector erase command, the results of command execution cannot be assured.

#### (3) Sector Erase Suspend

When sector erase is suspended, if you read an address which is in the sector to be erased, "1" is read in the sector erase timer flag. If you read an address which is in the sector adjacent to the sector to be erased in the same memory, "1" is also read in the sector erase timer flag.

If it is assumed that the sector to be erased is  $SAn$ , the adjacent sector would be  $SAn+1$  when  $n$  is an even number and  $SAn-1$  when  $n$  is an odd number. This is for large sector regions and not for small sector regions.

If the read address is not in the sector to be erased, the content written to the address is read.

### 3.6.6. Toggle Bit 2 Flag

The toggle bit 2 flag indicates whether the sector to which the read address belongs is in the sector erase suspend state. In addition, the toggle bit 2 flag can be used to determine which sector is being erased.

#### (1) Toggle Bit 2 Flag (DQ[2, 10])

By combining the toggle bit 2 flag and toggle bit flag, you can determine whether flash memory is in the sector erase suspend state or currently executing sector erase. During the execution of an automatic algorithm, the value shown in the DQ[2, 10] column in Table 3-4 is read in the toggle bit 2 flag.

#### (2) Sector Erase

If you read an address which is in the sector to be erased, the value of the toggle bit 2 flag is inverted whenever it is read. If you read an address which is in another sector, "1" is read in the toggle bit 2 flag. For all the sectors to be read during a macro erase operation, the value of the toggle bit 2 flag is inverted whenever it is read.

#### (3) Sector Erase Suspend

If you read an address which is in the sector to be erased, the value of the toggle bit 2 flag is inverted whenever it is read. If you read an address which is in another sector, the value written to the address is read. When sector erase is suspended, only sector erase resume and read operations can be performed.

If you read an address in the sector more than once, the value of the toggle bit flag is not inverted but the value of the toggle bit 2 flag is inverted whenever it is read. Therefore, you can determine that sector erase is suspended for the sector.

#### (4) Timing Limit Exceeded Flag

If the timing limit exceeded is detected, you can confirm which sector is causing the limit to be exceeded by using the toggle bit 2 flag. If you read the sector that causes the timing limit to be exceeded, the toggle bit 2 flag is inverted whenever it is read. For other sectors, the toggle bit 2 bit reads "1".



### 3.7. Insert Wait Cycle

You can insert wait cycles when flash memory is accessed by setting the FAWC bit in the CFCFGn\_FCFGR register.

If the system clock frequency is low, you can even set the FAWC bit to "0". The wait cycle count set for the FAWC bit is applied when flash memory is accessed for both reading and programming. The value set for the FAWC bit is valid until the setting is changed. The value of the FAWC bit is "3" after reset.

### 3.8. ECC Generation and Check

TCFLASH is capable of 1-bit error detection and correction and 2-bit error detection by adding the 7-bit error check code (ECC: Error Check Code) per 32 bits.

The ECC logic used in TCFLASH performs ECC generation during write-access and syndrome check during read-access in the same way as with 32-bit ECC in the ARM Cortex-R5F core.

During write-access, 7-bit ECC is generated for the 32-bit data, which is programmed along with the data. For the procedure for adding ECC to program data, see Section "4.3. Programming Procedure".

During read-access via the AXI space, a syndrome is calculated to determine any of "No error," "1-bit error," or "2-bit-or-more error." If the result is "2-bit-or-more error," error correction cannot be performed. If the content of flash memory is erased, an error is never detected. During read-access via the TCM space, TCFLASH does not detect or correct any errors. TCFLASH directly sends data and the ECC bit that are read from flash memory to the Cortex-R5F core. Error detection/correction for the data and ECC bit read via the TCM space is performed in the Cortex-R5F core.

The contents described from Section 3.8.1 to 3.8.3 are the operations performed in TCFLASH when it is read-accessed via the AXI space.

To test the check function, the ECC logic has a function for injecting an error into the data and ECC read from flash memory. Error injection is performed during read-access both via the TCM space and via the AXI space.

ECC is enabled/disabled by setting the ECCOFF bit in the TCFCFGn\_FECCCTRL register.

#### 3.8.1. ECC Generation

During writing, the 7-bit check bits CB[6:0] are generated from the 32-bit write data D[31:0] according to the calculation formula shown in Table 3-5. The generated check bits are exclusively ORed with 0x73 and then written to Flash memory.

Table 3-5 ECC Calculation Formula

Bit	Calculation Formula
CB[6]	$D[31] \wedge D[30] \wedge D[29] \wedge D[28] \wedge D[27] \wedge D[26] \wedge D[25] \wedge D[24] \wedge D[23] \wedge D[22] \wedge D[21] \wedge D[20] \wedge D[19] \wedge D[18] \wedge D[17] \wedge D[16]$
CB[5]	$D[31] \wedge D[30] \wedge D[29] \wedge D[28] \wedge D[27] \wedge D[26] \wedge D[25] \wedge D[24] \wedge D[7] \wedge D[6] \wedge D[5] \wedge D[4] \wedge D[3] \wedge D[2] \wedge D[1] \wedge D[0]$
CB[4]	$D[31] \wedge D[30] \wedge D[23] \wedge D[22] \wedge D[21] \wedge D[20] \wedge D[19] \wedge D[18] \wedge D[15] \wedge D[14] \wedge D[13] \wedge D[12] \wedge D[11] \wedge D[10] \wedge D[7] \wedge D[6]$
CB[3]	$\sim (D[29] \wedge D[28] \wedge D[27] \wedge D[23] \wedge D[22] \wedge D[21] \wedge D[17] \wedge D[16] \wedge D[15] \wedge D[14] \wedge D[13] \wedge D[9] \wedge D[8] \wedge D[5] \wedge D[4] \wedge D[3])$
CB[2]	$\sim (D[31] \wedge D[29] \wedge D[26] \wedge D[25] \wedge D[23] \wedge D[20] \wedge D[19] \wedge D[16] \wedge D[15] \wedge D[12] \wedge D[11] \wedge D[8] \wedge D[7] \wedge D[5] \wedge D[2] \wedge D[1])$
CB[1]	$D[28] \wedge D[26] \wedge D[24] \wedge D[22] \wedge D[20] \wedge D[18] \wedge D[17] \wedge D[16] \wedge D[14] \wedge D[12] \wedge D[10] \wedge D[9] \wedge D[8] \wedge D[4] \wedge D[2] \wedge D[0]$
CB[0]	$D[31] \wedge D[29] \wedge D[28] \wedge D[26] \wedge D[23] \wedge D[22] \wedge D[20] \wedge D[16] \wedge D[13] \wedge D[11] \wedge D[10] \wedge D[9] \wedge D[6] \wedge D[3] \wedge D[1] \wedge D[0]$



### 3.8.2. Syndrome Calculation

When read-accessing, the 7-bit check bits CB[6:0] are calculated from the data D[31:0] which is read from flash memory in accordance with the calculation formulas given in Table 3-5. The calculated check bits are used together with the check bits EDOR[6:0] that are read from the flash memory to generate a syndrome S[6:0] in accordance with the calculation formulas given in Table 3-6.

**Table 3-6 Syndrome Calculation Formulas**

Bit	Calculation Formula
S[6]	$\sim (CB[6] \wedge EDOR[6])$
S[5]	$\sim (CB[5] \wedge EDOR[5])$
S[4]	$\sim (CB[4] \wedge EDOR[4])$
S[3]	$(CB[3] \wedge EDOR[3])$
S[2]	$(CB[2] \wedge EDOR[2])$
S[1]	$\sim (CB[1] \wedge EDOR[1])$
S[0]	$\sim (CB[0] \wedge EDOR[0])$

### 3.8.3. Error Detection

Based on the calculated value of syndrome S[6:0], the following decision is made: "no error detected," "1-bit error detected," "2-bit error detected," or "error of 3 or more bits detected." Table 3-7 shows the relationship between syndrome values and decision results. The meanings of the symbols used in this table are as follows.

- "+" : No error is detected.
- "C[n] (0≤n≤6) " : A 1-bit error is detected. The value of the check bit CB[n] is erroneous.
- "D[m]" (0≤m≤31) : A 1-bit error is detected. The value of the data bit D[m] is erroneous.
- "T" : A 2-bit error is detected. It cannot be corrected.
- "M" : An error of 3 or more bits is detected. It cannot be corrected.

Any 1-bit errors can be detected in all cases, and can be corrected. The 2-bit errors can be detected in all cases, but they cannot be corrected. The errors of 3 or more bits may be detected in some cases, as shown in Table 3-7. However, they cannot be detected in all cases. Note that errors of 3 or more bits also cannot be corrected.

**Table 3-7 Meanings of Syndrome Values**

		Value in S[6:4]							
		0x0	0x1	0x2	0x3	0x4	0x5	0x6	0x7
Value in S[3:0]	0x0	+	C[4]	C[5]	T	C[6]	T	T	D[30]
	0x1	C[0]	T	T	D[6]	T	M	M	T
	0x2	C[1]	T	T	M	T	D[18]	D[24]	T
	0x3	T	D[10]	D[0]	T	M	T	T	M
	0x4	C[2]	T	T	D[7]	T	D[19]	D[25]	T
	0x5	T	D[11]	D[1]	T	M	T	T	D[31]
	0x6	T	D[12]	D[2]	T	M	T	T	M
	0x7	M	T	T	M	T	D[20]	D[26]	T
	0x8	C[3]	T	T	M	T	D[21]	D[27]	T
	0x9	T	D[13]	D[3]	T	M	T	T	M
	0xA	T	D[14]	D[4]	T	D[17]	T	T	M
	0xB	D[9]	T	T	M	T	D[22]	D[28]	T
	0xC	T	D[15]	D[5]	T	M	T	T	M
	0xD	M	T	T	M	T	D[23]	D[29]	T
	0xE	D[8]	T	T	M	T	M	M	T
	0xF	T	M	M	T	D[16]	T	T	M





### 3.9. Interrupt

TCFLASH can issue an interrupt request in the following cases.

- When a program or erase operation is completed and flash memory is ready to start the next operation  
The RDYINT bit in the TCFCFGn\_FSTAT0/1 register reads "1". In this case, if the RDYIE bit in the TCFCFGn\_FICTRL0/1 register for the same memory of the same unit is "1", TCFLASH generates an interrupt request.
- When the hang-up 1 state is detected  
When a timeout is detected during the execution of an automatic algorithm, etc., the hang-up 1 state is detected, causing the HANGINT bit in the TCFCFGn\_FSTAT0/1 register to show "1". In this case, if the HANGIE bit in the TCFCFGn\_FICTRL0/1 register for the same memory of the same unit is "1", TCFLASH generates an error interrupt request.
- When the ECC logic detects a 1-bit error (correctable)  
The SECINT bit in the TCFCFGn\_FSECIR register is "1". In this case, if the SECIE bit of the same register is "1", TCFLASH generates an interrupt request.

### 3.10. Bus Error Response

TCFLASH generates a bus error response in the following cases.

- When an uncorrectable error is detected. (A 2-bit-or-more error is detected with the ECC check.)
- When a program or erase operation in the non-privileged state is attempted
- When a program or erase operation is attempted while the WE bit in the TCFCFGn\_FCFGR register is "0"
- When 8 or 16-bit programming is attempted while ECC is enabled
- When 8 or 16-bit programming is attempted while the 2nd cycle program sequence of 32-bit programming is being executed
- When read-access of a size exceeding 64 bits is attempted
- When write-access of a size exceeding 32 bits is attempted
- When register access of a size exceeding 64 bits is attempted
- When programming an address that is not boundary aligned with the program size is attempted
- When read-access via the TCM interface is attempted while the WE bit in the TCFCFGn\_FCFGR register is "1"
- When read-access is attempted via the TCM interface during execution of the automatic algorithm
- When access is attempted to the reserved area in the TCM space, reserved area in the AXI space, or reserved area in the register space
- When a change of setting is attempted by programming registers in the non-privileged state
- When write-access of the 2nd and subsequent cycle in the TCFCFGn\_FECCCTRL register is attempted
- When a change of settings is attempted in a manner deviating from the unlock sequence. Specifically, the following cases are applicable.
  - When write-access to the protected register is attempted in the locked state
  - When 2 consecutive write-accesses are attempted for the TCFCFGn\_FCPROTKEY
  - When write-access of the wrong protection key is attempted for the TCFCFGn\_FCPROTKEY
  - When write-access of the protected register is attempted by a bus master other than the one that unlocked programming of that protected register For example, if write-access to the protected register in the same unit is attempted by DMAC while CPU #0 is programming the correct protection key for the TCFCFGn\_FCPROTKEY and unlocking programming of the protected register, TCFLASH issues a bus error response to DMAC.
- When write-access to the read-only register is attempted



### 3.11. Flash Security

#### (1) Security Information Area

This product is equipped with a flash security function to prevent unauthorized reading or falsification of the contents of TCFLASH and WorkFLASH.

The flash security function is controlled by the content written to the security information area. In this product, the first 96 bytes of the small sector regions of TCFLASH are the security information area.

The layout of the security information area is shown in Table 3-8.

**Table 3-8 Layout of Security Information Area**

Offset	SA1				SA0			
	+7	+6	+5	+4	+3	+2	+1	+0
0x0_00000	reserved				SDR_FSECM			
0x0_00008	reserved				SDR_DSM			
0x0_00010	reserved				SDR_DSKM0			
0x0_00018	reserved				SDR_DSKM1			
0x0_00020	reserved				SDR_DSKM2			
0x0_00028	reserved				SDR_DSKM3			
0x0_00030	reserved				reserved			
0x0_00038	reserved				reserved			

#### a) SDR\_FSECM

For details, see "CHAPTER: BootROM Software Interface".

#### b) SDR\_DSM

For details, see "CHAPTER: BootROM Software Interface".

#### c) SDR\_DSKM 0, 1, 2, 3

For details, see "CHAPTER: BootROM Software Interface".

#### (2) Flash Security ON/OFF Determination

If the value of SDR\_FSECM[15:0] (hereinafter referred to as flash security marker) is 0x0001, the flash security is in the ON state. The flash security marker byte line is the little endian.

After the reset is released, the value of the flash security marker is read from the security information area in the TCFLASH interface. If the read value of the flash security marker is 0x0001, the flash security is determined to be in the ON state.

A product with multiple Cortex-R5F cores has as many TCFLASHs as cores. In this case, if the value of the flash security marker is 0x0001 in either one of the TCFLASHs, the flash security is determined to be in the ON state. For greater security of all the chips, set the value of the security marker in each TCFLASH to 0x0001.

After the reset is released, ECC is enabled for TCFLASH. If an uncorrectable error is detected as a result of reading the flash security marker, the state can be determined by using the value before ECC arithmetic operations. If a correctable error is detected, it can be determined using the value after correction.

#### (3) TCFLASH Access Restrictions when Security Is ON

To prevent the data and programs in TCFLASH from being read by third parties, this product's operations on TCFLASH are restricted as described in "CHAPTER: Security".

## 4. Setting Procedure Example

This section explains the TCFLASH setting procedures.

### 4.1. Setting Wait Cycle Count

If the operating frequency of the system is higher than the maximum operating frequency of the flash memory, it is necessary to insert wait cycles when accessing the flash memory, by setting an appropriate value in FAWC[1:0] in the TCFCFGn\_FCFGR register.

The wait settings must not be changed while TCFLASH is being accessed (during read-access or program execution). The changed wait settings are enabled after the specific cycle. For this reason, perform a dummy reading of the register in TCFLASH twice, and then access TCFLASH after the wait settings are applied.

You can calculate the value to set for FAWC[1:0] according to the following formula.

$$\text{FAWC}[1:0] = \text{CEILING} (\text{System operating frequency} / \text{Maximum operating frequency of flash memory}) - 1$$

CEILING() in the above formula represents a function that rounds up the decimal part of the argument to make it an integer.

The maximum operating frequency of the flash memory built into this product is 80MHz. Therefore, if the system operating frequency is 200MHz, for example, the setting value for FAWC[1:0] should be "2" as calculated using the above formula.

### 4.2. Reading Flash Memory State/Transition to Reset State

Program the read/reset command sequence in flash memory. If execution of the automatic algorithm terminates abnormally (due to a timeout, for example), the state of flash memory can be initialized. Because the state of flash memory is read/reset immediately after this product is reset, you do not need to program the read/reset command sequence in flash memory.

Successful program or erase operation by flash memory is not affected by programming of the read/reset command sequence. In addition, a suspended sector erase cannot be canceled by programming the read/reset command sequence.



### 4.3. Programming Procedure

To program data in flash memory, write the program command sequence to flash memory. The destination address must be aligned with the data size to be programmed.

When 8- or 16-bit programming is performed, the data is directly written to the address specified in flash memory. When ECC is enabled and 8- or 16-bit programming is attempted, TCFLASH issues a bus error response.

32-bit programming must be repeated twice for the same data and address. When ECC is enabled, ECC is generated from the 32-bit data and written at the same time. When 32-bit programming is performed, the lower 16 bits of the 32-bit data are written to flash memory by the 1st write. Then, the upper 16 bits are written to flash memory by the 2nd write. Therefore, to wait for the completion of the 1st write, the hardware sequence flag must be read and polled using the same written address. To wait for the completion of the 2nd write, the hardware sequence flag must be read and polled using the written address + 2.

The cell value of flash memory cannot be changed from "0" to "1" by programming. Therefore, if "1" is written in the cell in which "0" has already been written, flash memory is indicated as being in the hang-up 1 state.

During execution of program operation, all the command sequences additionally written in flash memory are ignored.

If a bus error response is received while the command sequence is being written, issue software reset by writing "1" bit to the SWFRST bit to initialize the macro (for flash memory to be ready to accept a new command rather than erasing the macro).

If this product is reset during execution of program operation or flash memory is reset because "1" is written to the SWFRST bit in the TCFCFGn\_FCFGR register, contents of the cell programmed after reset cannot be assured.

Figure 4-1 Programming Procedure (Using Hardware Sequence Flag)

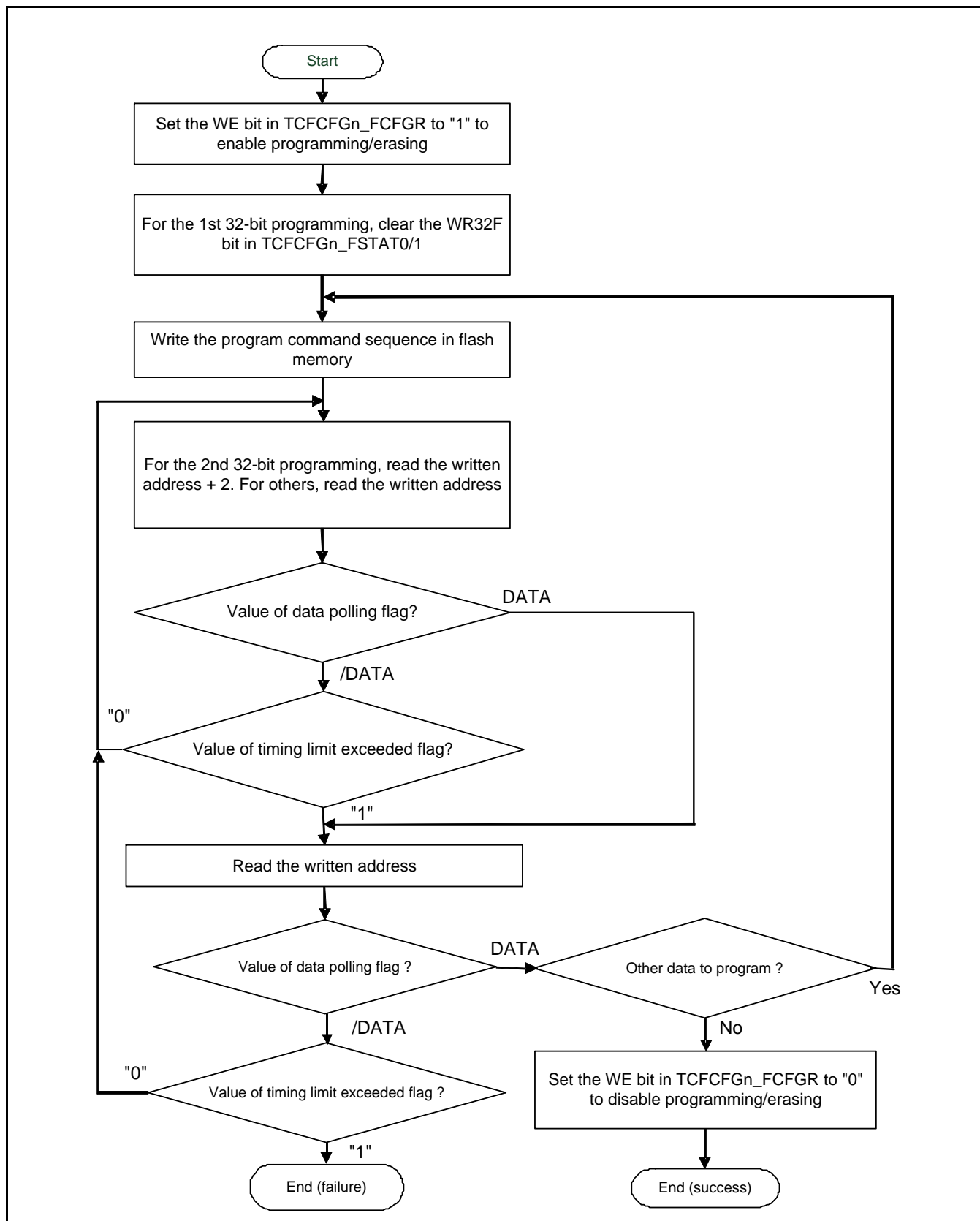
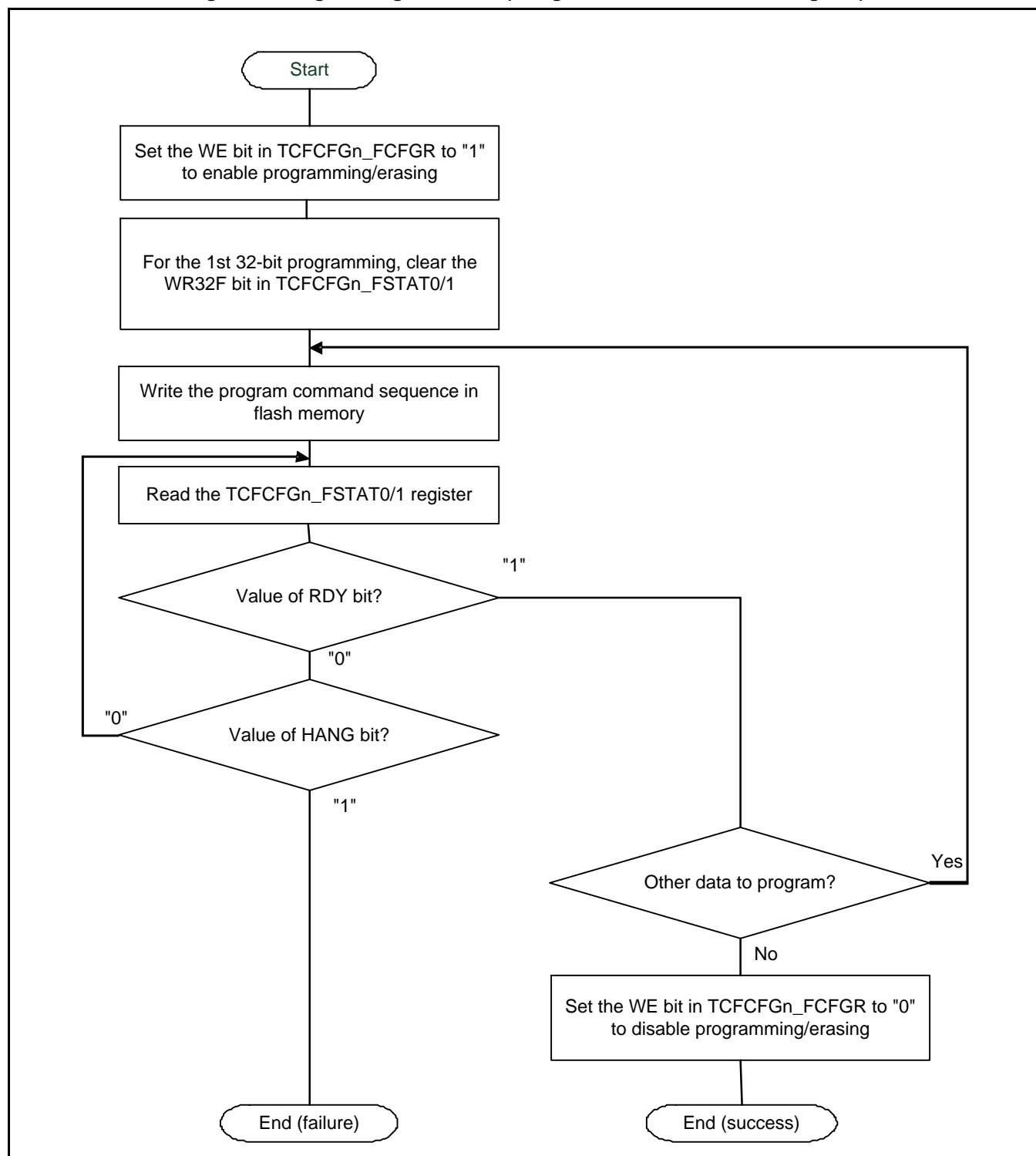


Figure 4-2 Programming Procedure (Using the TCFCFGn\_FSTAT0/1 Register)



## 4.4. Macro Erasure Procedure

Write the macro erase command sequence in flash memory. If the last cycle of programming completes in a total of 6 cycles of programming that comprise the sequence, flash memory starts the macro erase operation. After the macro erase operation completes, the values of all the memory cells in the targeted flash memory are "1".

During execution of the macro erase operation, the write operation of any command sequence is ignored in flash memory.

If this product is reset during execution of the macro erasure operation or flash memory is reset because "1" is written to the SWFRST bit in the TCFCFGn\_FCFGR register, the contents of the flash memory after reset cannot be assured.

The execution status of macro erasure can be confirmed from the values of the hardware sequence flag.

## 4.5. Sector Erasure Procedure

You can erase data in units of sectors by writing the sector erase command sequence in the flash memory. The sector to which the address specified in the command sequence belongs is to be erased.

When the sector erase command sequence is written, the flash memory enters the sector erase wait period with a minimum of 40  $\mu$ s. When writing 0x30 (the last code of the sector erase command sequence) to an address in another sector within the sector erase wait period, that sector will also be erased. Therefore, you can use this function to erase multiple sectors. While the sector erase command sequence is being written, the sequence waits for a minimum of 40  $\mu$ s after the last time 0x30 is written to flash memory, and then starts erasing the target sector. For multiple sector erase, read the sector erase timer flag (DQ[3, 11]) of the hardware sequence flag and confirm that the written code (0x30) has been received.

After erase operation completes, all the values of the flash memory cells in the target sectors are "1".

Figure 4-3 and Figure 4-4 show a sector erase flow example. "Timeout marker" shown in Figure 4-4 represents a Boolean variable used to record the detection of a program timeout (it is not a hardware flag).



Figure 4-3 Flow Example when a Single Sector is Erased

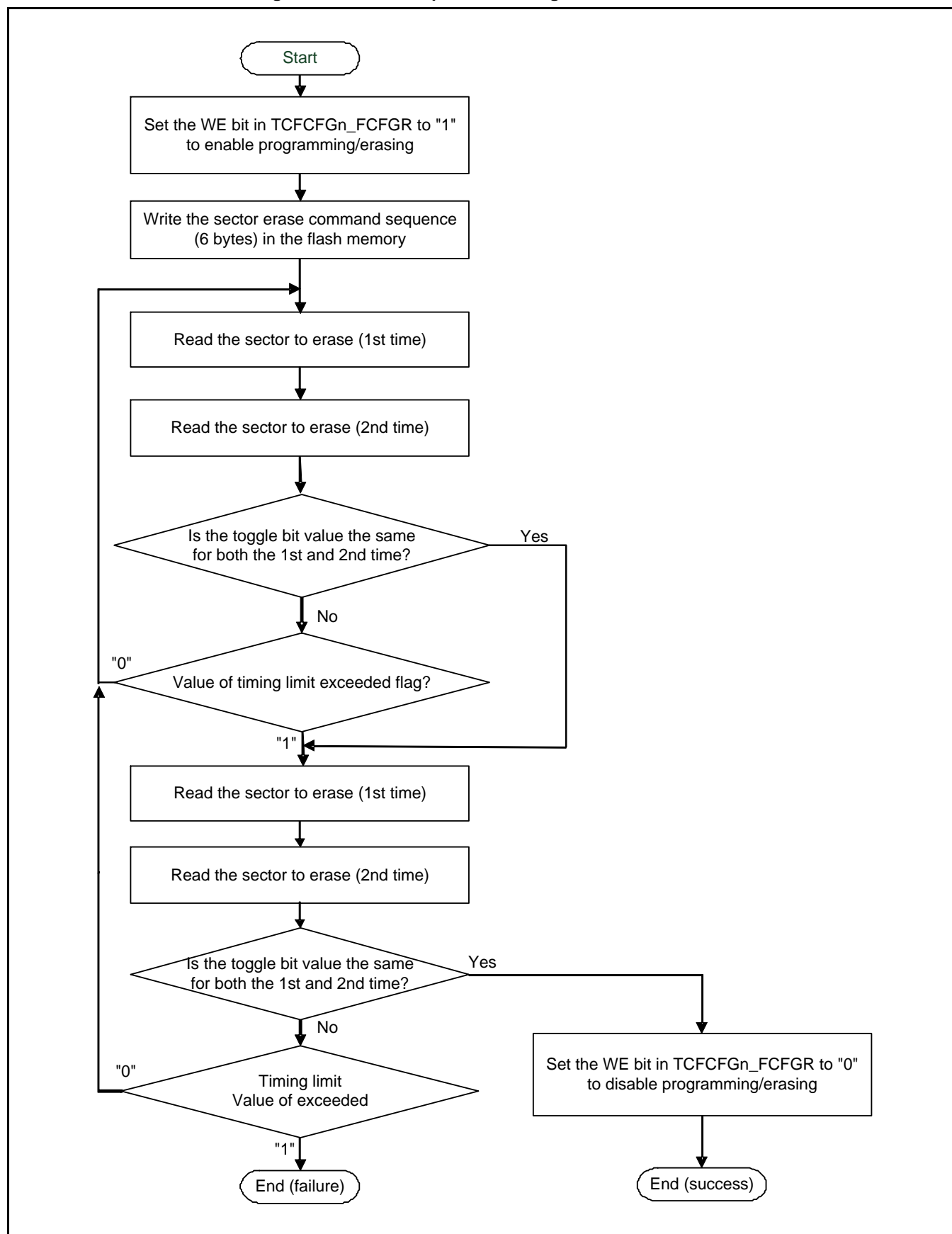


Figure 4-4 Flow Example when Multiple Sectors are Erased

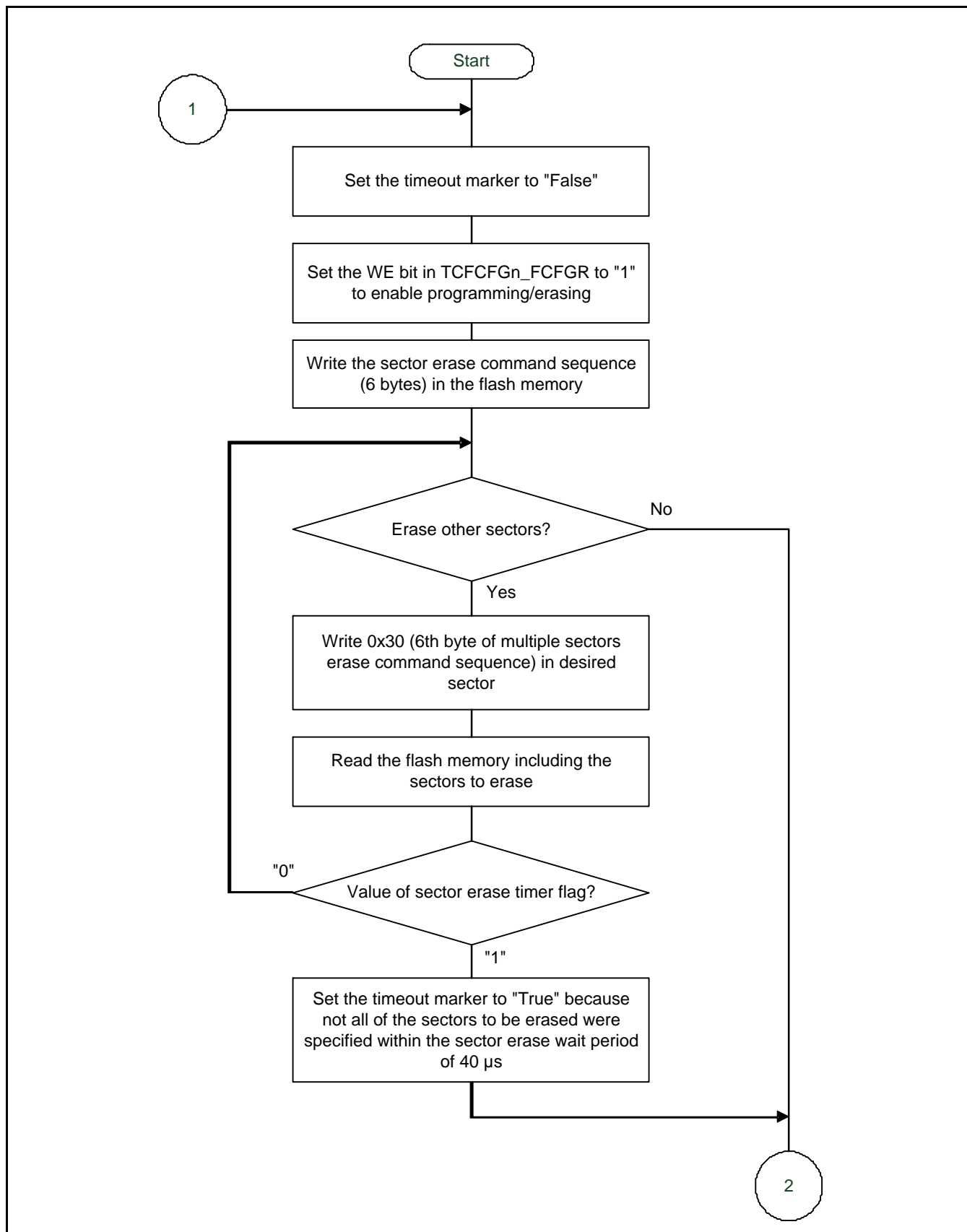
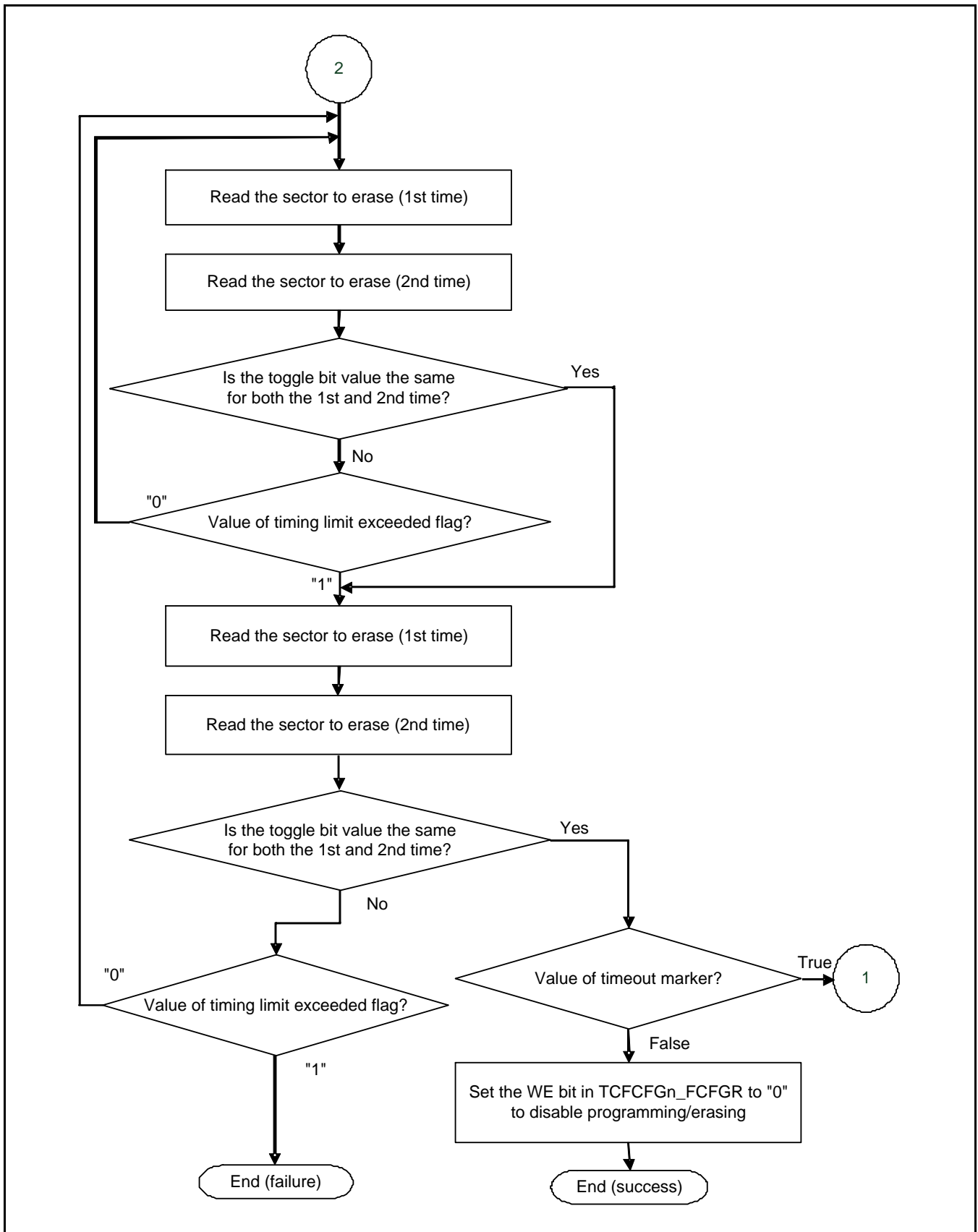


Figure 4-5 Flow Example when Multiple Sectors are Erased (Continued)



#### 4.5.1.    **Sector Erase Suspend**

Write the sector erase suspend command sequence in the flash memory that performs the sector erase operation that you want to suspend. At this time, specify the address that belongs to the target sector for which sector erase is to be suspended.

If the sector erase suspend command sequence is written while the flash memory is in the sector erase wait period, the sector erase operation is immediately canceled in the flash memory. If the sector erase suspend command sequence is written while the flash memory is executing sector erase, it takes the flash memory a maximum of 20  $\mu$ s to enter the sector erase suspend state.

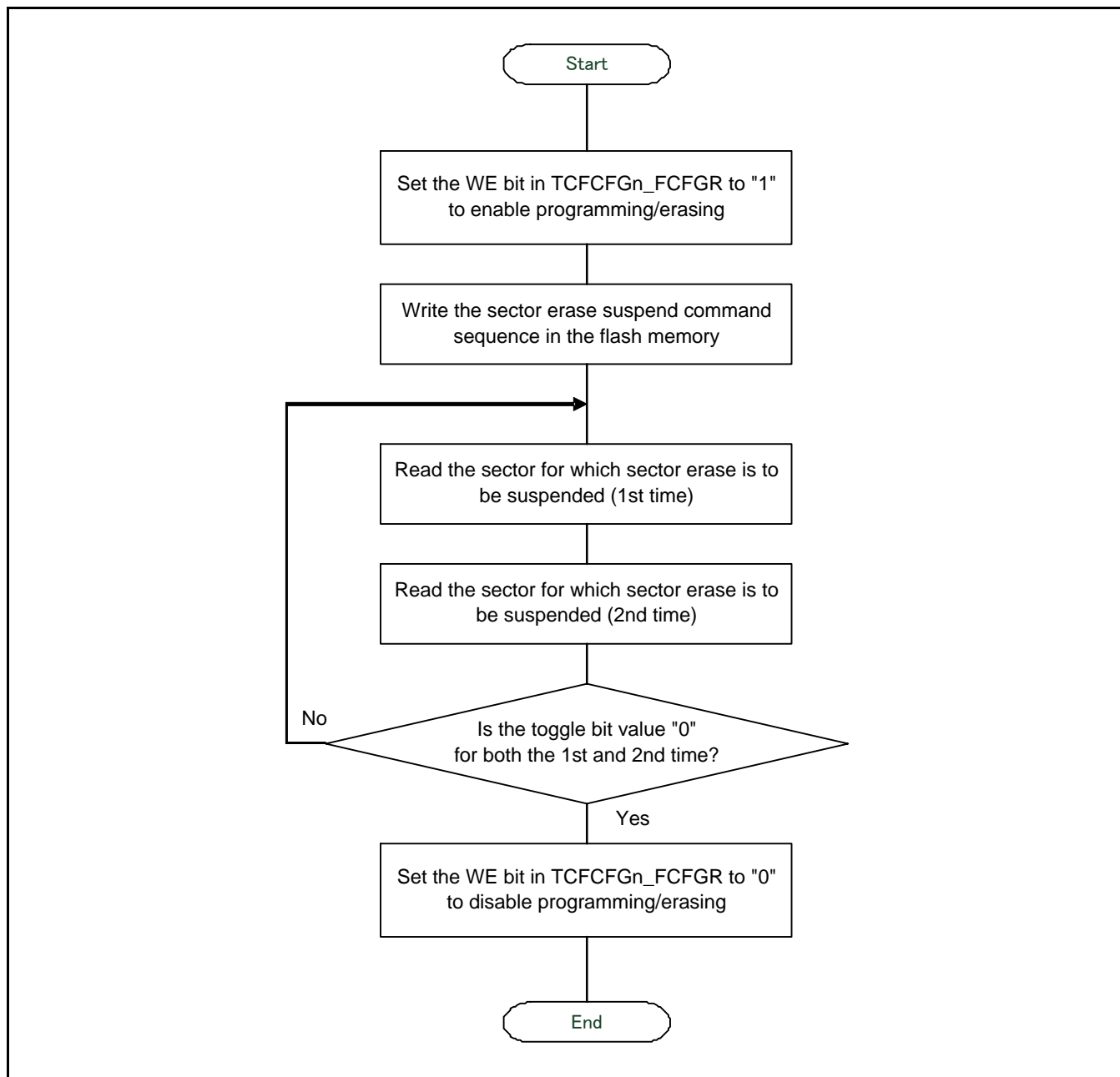
The sector erase suspend command sequence is valid only when the flash memory is in the sector erase state. The sector erase suspend command sequence is ignored by the flash memory if the programming or macro erase operation is being performed or the flash memory is already in the sector erase suspend state.

Do not write a command sequence other than the read/reset or sector erase resume command in the flash memory in the sector erase suspend state.

After the sector erase suspend command is issued, do not read flash memory until the RDY bit in the status register becomes "1".

Figure 4-6 shows a flow example of sector erase suspend.

Figure 4-6 Flow of Sector Erase Suspend Process (Example)



#### **4.5.2. Sector Erase Resume**

Write the sector erase resume command sequence in the flash memory that performs sector erase operation that you want to resume. At this time, specify the address that belongs to the target sector for which sector erase is suspended.

The sector erase resume command sequence is ignored by the flash memory if the sector erase is not suspended.



## 4.6. Flash Security Setting Method

This product has as many TCFLASHs as Cortex-R5F cores, and each TCFLASH has a security information area at the beginning of the small sector region in flash memory A. To turn flash security ON, 0x0001 must be written to the security marker, which exists in either of the security information areas. By writing 0x0001 to the security marker in either of the TCFLASH security information areas and then resetting this product, the flash security state turns ON.

After the reset, ECC is enabled for TCFLASH. For this reason, write to the security marker with ECC enabled.

## 4.7. Flash Security Unlocking Method

This product has as many TCFLASHs as Cortex-R5F cores, and each TCFLASH has a security information area at the beginning of the small sector region in flash memory A. To turn flash security OFF, all security marker values in the security information areas must be changed to other than 0x0001.

If this product is in user mode, no restrictions are imposed on programming and erasing flash memory. Therefore, all the security marker values in the security information areas can be changed to other than 0x0001 by programming, sector erase, or macro erase.

By changing all the security marker values to other than 0x0001 and then resetting this product, the flash security state turns OFF.

If this product is in other than user mode (production test mode), flash security cannot be released either by programming or sector erase. Therefore, turn the flash security OFF by performing macro erase, as follows.

1. To erase the contents, perform macro erase for the flash memory in all the WorkFLASHs and flash memory B in all the TCFLASHs.  
This can be done for the flash memory in all the WorkFLASHs and flash memory B in the TCFLASHs in any order.
2. To erase the contents, perform macro erase for flash memory A in all the TCFLASHs.  
This can be done for flash memory A in the TCFLASHs in any order. For example, a product having 2 Cortex-R5F cores has a total of 2 TCFLASHs, each of which is for each core. In this case, you can execute macro erase for flash memory A in TCFLASH#0 and flash memory A in TCFLASH#1 in any order.
3. Reset the product.

If this product is reset before step 1 or 2, above, is finished, repeat the procedure from step 1.

## 5. Registers

This section explains the registers in TCFLASH.

The register areas in TCFLASH are located in the peripheral functions area, each of which has a size of 1 KB.

Table 5-1 shows the placement of registers in the register areas. The register name shown in the table with "n" preceded by "TCFCFG" has a one-digit TCFLASH unit number (number of Cortex-R5F core to which TCFLASH is connected). For example, this would be "TCFCFG1\_" for the TCFLASH connected to Cortex-R5F core #1.

**Table 5-1 List of TCFLASH Registers**

Abbreviated Register Name	Register Name	See
TCFCFGn_FCPROTKEY	TCFLASH Configuration Protection Key Register	5.1
TCFCFGn_FCFGR	TCFLASH Configuration Register	5.2
TCFCFGn_FECCCTRL	TCFLASH ECC Control Register	5.3
TCFCFGn_FDATEIR	TCFLASH Data Bit Error Injection Register	5.4
TCFCFGn_FECCEIR	TCFLASH ECC Bit Error Injection Register	5.5
TCFCFGn_FICTRL0	TCFLASH Interrupt Control Register (Flash memory A)	5.6
TCFCFGn_FICTRL1	TCFLASH Interrupt Control Register (Flash memory B)	5.6
TCFCFGn_FSTAT0	TCFLASH Status Register (Flash memory A)	5.7
TCFCFGn_FSTAT1	TCFLASH Status Register (Flash memory B)	5.7
TCFCFGn_FSECIR	TCFLASH SEC Interrupt Register	5.8
TCFCFGn_FECCEAR	TCFLASH SEC Error Address Register	5.9
TCFCFGn_FMIDR	TCFLASH Module Identification Register	5.10
TCFCFGn_FUCEDIR	TCFLASH Uncorrectable Error Detection Interrupt Register	5.11
TCFCFGn_FUCEAR	TCFLASH Uncorrectable Error Address Register	5.12





## 5.1. TCFLASH0/1/2/3 Configuration Protection Key Register (TCFCFGn\_FCPROTKEY)

The TCFLASH configuration protection key register (TCFCFGn\_FCPROTKEY) is used to protect the registers belonging to the same units, listed below, from unintentional programming.

- TCFLASH0/1/2/3 configuration register (TCFCFGn\_FCFGR)
- TCFLASH0/1/2/3 ECC control register (TCFCFGn\_FECCCTRL)
- TCFLASH0/1/2/3 data bit error injection register (TCFCFGn\_FDATEIR)
- TCFLASH0/1/2/3 ECC bit error injection register (TCFCFGn\_FECCEIR)

Before programming these registers, you must write the correct configuration protection key value (0xCF61F1A5) to the TCFLASH0/1/2/3 configuration protection key register in the same unit to unlock programming of these registers.

Programming of the above registers is again locked by programming addresses in the same group (Memory & Config Group) area from the accessible bus master. Note, however, that this locking is not possible by programming to the same group area (except the TCFLASH register area) from another accessible master.

If the value of the configuration protection key is wrong or an invalid procedure is used to program the above registers, such that programming is attempted in these registers without unlocking programming, TCFLASH generates a bus error response. For access with a CPU, a data abort exception is generated due to the bus error response.

Be sure to program in 32-bit to the TCFLASH0/1/2/3 configuration protection key register.

BIT_OFFSET	31-0
BIT_NAME	FCPROTKEY
ACCESS_TYPE	R,W
PROT_TYPE	WP
INITIAL_VALUE	00000000_00000000_00000000_00000000

### [bit31:0] FCPROTKEY[31:0]: Configuration protection key

Writing the correct configuration protection key value (0xCF61F1A5) to this field unlocks programming to the TCFLASH0/1/2/3 configuration register (TCFCFGn\_FCFGR), TCFLASH0/1/2/3 ECC control register (TCFCFGn\_FECCCTRL), TCFLASH0/1/2/3 data bit error injection register (TCFCFGn\_FDATEIR), and TCFLASH0/1/2/3 ECC bit error injection register (TCFCFGn\_FECCEIR) of the same unit, allowing new values to be set.

When programming to these registers is unlocked, 0xFFFFFFFF is output from this register. When programming to these registers is locked, 0x00000000 is output from this register.

## 5.2. TCFLASH0/1/2/3 Configuration Register (TCFCFGn\_FCFGR)

TCFLASH0/1/2/3 configuration register has the following functions:

- Resets flash memory
- Determines whether access via the TCM port or that via the AXI has higher priority.
- Enables programming of flash memory
- Specifies the wait cycle count when flash memory is accessed

The setting of this register can be changed only when the correct configuration protection key value is written to the TCFLASH0/1/2/3 configuration protection key register in the same unit and programming to this register is unlocked.

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	WPS
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved	SWFRST	TCMPR	WE	Reserved		FAWC	
ACCESS_TYPE	R0,WX	R0,W	R/W	R/W	R0,WX		R/W	
PROT_TYPE	WPS							
INITIAL_VALUE	0	0	0	0	00		11	

[bit31:7] Reserved: Reserved bits

[bit6] SWFRST: Software reset

Value	Description
0	There is no effect on operation.
1	Flash memory is reset.

[bit5] TCMPR: TCM priority enable

Determine whether read access via TCM or read access via AXI has higher priority.

This function is not supported.

The written value does not affect operation.

(AXI write access has the highest priority)

Value	Description
0	Toggle the priority between the AXI access and TCM access every 16 accesses.
1	Give priority to the TCM access.

**[bit4] WE: Program enable**

It sets enable/disable programming to flash memory. If programming of the command sequence to flash memory is detected while this bit is "0", TCFLASH generates a bus error response.

When read-access via the TCM interface is attempted while this bit is "1", TCFLASH issues a bus error response.

Perform the program/erase operation in FLASH by starting the automatic algorithm after setting the program enable bit at WE. After the program/erase operation in FLASH completes, set the program disable bit.

Value	Description
0	Programming command sequence is prohibited.
1	Programming command sequence is permitted.

**[bit3:2] Reserved: Reserved bits****[bit1:0] FAWC[1:0]: Flash wait control**

These bits set the wait cycle count when flash memory is accessed. Set the appropriate value according to the operating frequency of the flash memory and flash memory access time.

Value	Description
00	Do not insert wait cycle.
01	Insert 1 wait cycle.
10	Insert 2 wait cycles.
11	Insert 3 wait cycles.

### 5.3. TCFLASH0/1/2/3 ECC Control Register (TCFCFGn\_FECCCTRL)

This register is used to control the operation of the ECC logic. The setting of this register can be changed only when the correct configuration protection key value is written to the TCFLASH0/1/2/3 configuration protection key register in the same unit and programming to this register is unlocked. You can change the setting value of this register only once. If a 2nd and subsequent write access is attempted to this register, TCFLASH generates a bus error response.

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	WPS
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved							ECCOFF
ACCESS_TYPE	R0,WX							R/W
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							0

**[bit31:1] Reserved: Reserved bits**

**[bit0] ECCOFF: ECC off**

This bit enables or disables operation of the ECC logic for AXI accesses. ECC generation and check for TCM accesses are controlled by the settings in the Cortex-R5F core.

Value	Description
0	Perform ECC generation and check for AXI accesses.
1	Do not perform ECC generation and check for AXI accesses.



#### 5.4. TCFLASH0/1/2/3 Data Bit Error Injection Register (TCFCFGn\_FDATEIR)

The TCFLASH0/1/2/3 data bit error injection register is used to test operation of the ECC logic by injecting an error to the data bit read from the flash memory.

The setting of this register can be changed only when the correct configuration protection key value is written to the TCFLASH0/1/2/3 configuration protection key register in the same unit and programming to this register is unlocked.

BIT_OFFSET	31-0
BIT_NAME	FDATEIR
ACCESS_TYPE	R/W
PROT_TYPE	WPS
INITIAL_VALUE	00000000_00000000_00000000_00000000

##### [bit31:0] FDATEIR[31:0]: Data bit error injection point

The exclusive OR of the value in this field and the value of the data bits that are read from the flash memory is sent to the ECC inspection logic. The output from the flash memory has 32-bit x 2 lanes, with 64-bit width. To prohibit error injection into the specific lane, specify it in the LMASK field in the TCFCFGn\_FECCEIR register.

bit [i] (31≥i≥0)	Description
0	Send those bits in the same location as the data read from Flash memory, to the ECC check logic as is.
1	Invert the bits in the same location as the data read from Flash memory, before sending them to the ECC check logic.

## 5.5. TCFLASH0/1/2/3 ECC Bit Error Injection Register (TCFCFGn\_FECCEIR)

The TCFLASH0/1/2/3 ECC bit error injection register is used to test operation of the ECC logic by injecting an error into the ECC bit from the flash memory.

The setting of this register can be changed only when the correct configuration protection key value is written to the TCFLASH0/1/2/3 configuration protection key register in the same unit and programming to this register is unlocked.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved						LMASK	
ACCESS_TYPE	R0,WX						R/W	
PROT_TYPE	WPS							
INITIAL_VALUE	000000						00	

BIT_OFFSET	23-8							
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	WPS							
INITIAL_VALUE	00000000_00000000							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved	FECCEIR						
ACCESS_TYPE	R0,WX	R/W						
PROT_TYPE	WPS							
INITIAL_VALUE	0	0000000						

**[bit31:26] Reserved: Reserved bits**

**[bit25:24] LMASK[1:0]: Error injection lane mask**

These bits specify whether to insert an error in the data read from the flash memory.

bit [i] (1≥i≥0)	Description
0	Insert an error between bit (32*(i+1)-1) and bit (32*i) of the data read from the flash memory, as specified by the TCFCFGn_FDATEIR register. Similarly, insert an error between bit (7*(i+1)-1) and bit (7*i) of the ECC data read from the flash memory, as specified by the TCFCFGn_FECCEIR register.
1	Do not insert an error between bit (32*(i+1)-1) and bit (32*i) of the data read from the flash memory. Also, do not insert an error between bit (7*(i+1)-1) and bit (7*i) of the ECC data read from the flash memory.

**[bit23:7] Reserved: Reserved bits**



**[bit6:0] FECCEIR[6:0]: ECC bit error injection point**

The exclusive OR of the value in this field and the value of the ECC bits that are read from the flash memory is sent to the ECC inspection logic. To prohibit error injection into the specific lane, specify it in the LMASK field in the TCFCFGn\_FECCEIR register.

bit [i] (6≥i≥0)	Description
0	Directly send the bits with the same position as the ECC data being read from the flash memory, to the ECC inspection logic.
1	Send the bits with the same position as the ECC data being read from the flash memory, to the ECC inspection logic after inverting their values.

## 5.6. TCFLASH0/1/2/3 Interrupt Control Register (TCFCFGn\_FICTRL0/1)

This register can enable interrupt request generation for each factor and clear an interrupt factor. There are as many of these registers as flash memories.

BIT_OFFSET	31-16
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	WP
INITIAL_VALUE	00000000_00000000

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved					WR32FC	HANGIC	RDYIC
ACCESS_TYPE	R0,WX					R0,W	R0,W	R0,W
PROT_TYPE	WP							
INITIAL_VALUE	00000					0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved						HANG	RDY
ACCESS_TYPE	R0,WX						R,WX	R,WX
PROT_TYPE	WP							
INITIAL_VALUE	000000						0	0

**[bit31:11] Reserved: Reserved bits**

**[bit10] WR32FC: 32-bit programming flag clear**

Value	Description
0	Writing "0" to this bit has no meaning.
1	Writing "1" to this bit clears the WR32F bit in the TCFCFGn_FSTAT0/1 register for the same memory in the same unit.

**[bit9] HANGIC: Hang interrupt clear**

Value	Description
0	Writing "0" to this bit has no meaning.
1	Writing "1" to this bit clears the HANGINT bit in the TCFCFGn_FSTAT0/1 register for the same memory in the same unit.

**[bit8] RDYIC: Ready interrupt clear**

Value	Description
0	Writing "0" to this bit has no meaning.
1	Writing "1" to this bit clears the RDYINT bit in the TCFCFGn_FSTAT0/1 register for the same memory in the same unit.

**[bit7:2] Reserved: Reserved bits**





**[bit1] HANGIE: Hang interrupt enable**

Value	Description
0	Disable the generation of hang interrupt requests.
1	Enable the generation of hang interrupt requests.

**[bit0] RDYIE: Programming/erasing ready interrupt enable**

Value	Description
0	Disable the generation of programming/erasing ready interrupt requests.
1	Enable the generation of programming/erasing ready interrupt requests.

## 5.7. TCFLASH0/1/2/3 Status Register (TCFCFGn\_FSTAT0/1)

There are as many of these registers as flash memories, and are read-only registers that retain the state of the flash memory and individual interrupt factors. Writing to this register returns a bus error.

BIT_OFFSET	31-16
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved						HANGINT	RDYINT
ACCESS_TYPE	R0,WX						R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	000000						0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved			WR32F	Reserved		HANG	RDY
ACCESS_TYPE	R0,WX			R,WX	R0,WX		R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	000			0	00		0	0

**[bit31:10] Reserved: Reserved bits**

**[bit9] HANGINT: Hang-up interrupt**

This bit indicates that a hung up interrupt request is generated because transition of the flash memory to the hung up 1 state is detected.

This bit is cleared by writing "1" to the HANGIC bit in the TCFCFGn\_FICTRL0/1 register for the same memory in the same unit.

Value	Description
0	No hang-up interrupt request has been generated.
1	A hang-up interrupt request has been generated.

**[bit8] RDYINT: Programming/erasing ready interrupt**

This bit indicates that a ready interrupt request was generated because a transition of a Flash memory to the ready state was detected. This bit is cleared by writing "1" to the RDYIC bit in the TCFCFGn\_FICTRL0/1 register for the same memory in the same unit.

Value	Description
0	No programming/erasing ready interrupt request is generated.
1	A programming/erasing ready interrupt request is generated.

**[bit7:5] Reserved: Reserved bits**

**[bit4] WR32F: 32-bit programming control flag**

The value of this bit has meaning only for 32-bit programming in flash memory. The value of this bit is inverted whenever the write command sequence in the flash memory is completed. The value of this bit goes to "0" as a result of writing "1" to the WR32FC bit in the TCFCFGn\_FICTRL0/1 register of the same memory in the same unit. The value of this bit also goes to "0" while the WE bit in the TCFCFGn\_FCFGR in the same unit is "0".

Value	Description
0	When 32-bit programming is performed for TCFLASH, only the lower 16 bits are written to flash memory.
1	When 32-bit programming is performed for TCFLASH, the upper 16 bits and ECC bits calculated from all of the 32-bit data are written to flash memory.

**[bit3:2] Reserved: Reserved bits****[bit1] HANG: Hang-up**

This bit indicates whether the flash memory is in hang-up 1 state. The flash memory changes to hang-up 1 state in the following cases.

- When write access with "1" is attempted to a cell with the value of "0"
- When execution of the automatic algorithm has not completed within the given time

Value	Description
0	Flash memory is not in the hang-up 1 state.
1	Flash memory is in the hang-up 1 state.

**[bit0] RDY: Programming/erasing ready**

This bit indicates whether the flash memory is in the programming/erasing ready state. The flash memory can start execution of a new programming/erasing command in the programming/erasing ready state.

Value	Description
0	Indicate that the flash memory is performing program or erase operation using the automatic algorithm. In this state, only the read/reset command or sector erase suspend command can be received.
1	Indicate that the flash memory completes program or erase operation using the automatic algorithm and is ready to start the next command.

## 5.8. TCFLASH0/1/2/3 SEC Interrupt Register (TCFCFGn\_FSECIR)

This register holds status flags, enable bits, and clear bits concerning 1-bit error correction interrupt.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved	SYN						
ACCESS_TYPE	R0,WX	R,WX						
PROT_TYPE	WP							
INITIAL_VALUE	0	0000000						

BITS_OFFSET	23	22	21	20	19	18	17	16
BITS_NAME	Reserved							SECINT
ACCESS_TYPE	R0,WX							R,WX
PROT_TYPE	WP							
INITIAL_VALUE	0000000							0

BITS_OFFSET	15	14	13	12	11	10	9	8
BITS_NAME	Reserved							SECIC
ACCESS_TYPE	R0,WX							R0,W
PROT_TYPE	WP							
INITIAL_VALUE	0000000							0

BITS_OFFSET	7	6	5	4	3	2	1	0
BITS_NAME	Reserved							SECIE
ACCESS_TYPE	R0,WX							R/W
PROT_TYPE	WP							
INITIAL_VALUE	0000000							0

**[bit31] Reserved: Reserved bit**

**[bit30:24] SYN[6:0]: Syndrome**

These bits store the syndrome from the detection of a 1-bit error. If a 1-bit error is detected in both the upper 32 bits and lower 32 bits when 64-bit read access is performed, syndrome for the error detected in the lower 32 bits is stored. Use the value of bit2 in the TCFCFGn\_FECCEAR register to determine whether the syndrome stored in this field is detected in the upper 32-bits or lower 32 bits.

**[bit23:17] Reserved: Reserved bits**

**[bit16] SECINT: 1-bit error correction interrupt**

This bit indicates whether there is a 1-bit error correction interrupt request. If the ECC check performed during reading has detected and corrected a 1-bit error, a 1-bit error correction interrupt request is generated. This bit is read-only. This bit is cleared by writing "1" to the SECIC bit in the same register.

Value	Description
0	A 1-bit error correction interrupt request has not been generated.
1	A 1-bit error correction interrupt request has been generated.



[bit15:9] Reserved: Reserved bits

[bit8] SECIC: 1-bit error correction interrupt clear

Value	Description
0	Writing "0" to this bit has no meaning.
1	Writing "1" to this bit clears the SECINT bit in the same register.

[bit7:1] Reserved: Reserved bits

[bit0] SECIE: 1-bit error correction interrupt enable

Value	Description
0	Disable generation of 1-bit error correction interrupt request.
1	Enable generation of 1-bit error correction interrupt request.

## 5.9. TCFLASH0/1/2/3 ECC Error Address Register (TCFCFGn\_FECCEAR)

This register retains the address at which a 1-bit error was detected during reading. If the 1-bit error was detected multiple times, the register retains the address at which the error was last detected.

This register is read-only. Writing to this register returns a bus error.

BIT_OFFSET	31-0
BIT_NAME	FECCEAR
ACCESS_TYPE	R,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000_00000000

### [bit31:0] FECCEAR[31:0]: Error address

These bits hold an address (word address) at which a 1-bit error is detected in the ECC check when read accessed. If an error is detected multiple times, they retain the address of the error detected last.

If a 1-bit error is detected in both the upper 32 bits and lower 32 bits when 64-bit read access is performed, the address of the lower 32 bits is stored.



## 5.10. TCFLASH0/1/2/3 Module Identification Register (TCFCFGn\_FMIDR)

This register reads a numerical value that contains the ID, version, and patch level of the TCFLASH unit.

This register is read-only. Writing to this register returns a bus error.

BIT_OFFSET	31-0
BIT_NAME	MID
ACCESS_TYPE	R,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000_00000000

### [bit31:0] MID[31:0]: Module ID

These bits indicate the value that contains the ID, version, or patch level of the TCFLASH unit.

## 5.11. TCFLASH0/1/2/3 Uncorrectable Error Detection Interrupt Register (TCFCFGn\_FUCEDIR)

This register holds status flags and clear bits concerning the uncorrectable error detection interrupt.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved	SYN						
ACCESS_TYPE	R0,WX	R,WX						
PROT_TYPE	WP							
INITIAL_VALUE	0	0000000						

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved							UCEDINT
ACCESS_TYPE	R0,WX							R,WX
PROT_TYPE	WP							
INITIAL_VALUE	0000000							0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved							UCEDIC
ACCESS_TYPE	R0,WX							R0,W
PROT_TYPE	WP							
INITIAL_VALUE	0000000							0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	WP							
INITIAL_VALUE	00000000							

**[bit31] Reserved: Reserved bit**

**[bit30:24] SYN[6:0]: Syndrome**

These bits store the syndrome from the detection of a 2-bit error. If an uncorrectable error is detected in both the upper 32 bits and lower 32 bits when 64-bit read access is performed, syndrome for the error detected in the lower 32 bits is stored. Use the value of bit2 in the TCFCFGn\_FUCEAR register to determine whether the stored syndrome is for an that error occurred in the upper 32 bits or an error that occurred in the lower 32 bits.

**[bit23:17] Reserved: Reserved bits**



**[bit16] UCEDINT: Uncorrectable error detection interrupt**

This bit indicates whether an uncorrectable error detection interrupt request exists. An uncorrectable error detection interrupt request is generated if an uncorrectable error is detected in the ECC check at read access. This bit is read-only. The value written in this bit has no meaning. This bit is cleared by writing "1" to the UCEDIC bit in this register.

Value	Description
0	Indicate that an uncorrectable error detection interrupt request is not generated.
1	Indicate that an uncorrectable error detection interrupt request is generated.

**[bit15:9] Reserved: Reserved bits****[bit8] UCEDIC: Uncorrectable error detection interrupt clear**

Value	Description
0	Writing "0" to this bit has no meaning.
1	Writing "1" to this bit clears the UCEDINT bit in this register.

"0" is always read from this bit.

**[bit7:0] Reserved: Reserved bits**

## 5.12. TCFLASH0/1/2/3 Uncorrectable Error Address Register (TCFCFGn\_FUCEAR)

This register holds an address (word address) at which an uncorrectable error is detected when read accessed. If multiple uncorrectable errors are detected, it holds the address at which the error is detected last. If an uncorrectable error is detected in both the upper 32 bits and lower 32 bits when 64-bit read access is performed, the address of the lower 32 bits is stored.

This register is read-only. Writing to this register returns a bus error.

BIT_OFFSET	31-0
BIT_NAME	UCEA
ACCESS_TYPE	R,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000_00000000

### [bit31:0] UCEA[31:0]: Uncorrectable error address

These bits indicate the address at which an uncorrectable error is detected in the ECC check at read access. If multiple errors are detected, they hold the address at which the error is detected last.



## 6. Others

This section explains precautions on the use of TCFLASH.

### (1) Handling of Values Read from Reserved Bits

"0" is read from the reserved bits that exist in registers in TCFLASH. However, do not allow the values read from the reserved bits to have meanings when programming, from the standpoint of software compatibility with the future products.

### (2) Reset during Execution of Program or Erase Operation

If this product is reset during execution of the program operation, the contents of the target address are shown as undefined. If it is reset during execution of the sector erase operation, the contents of the target sector are shown as undefined. In these cases, retry the suspended program or erase operation after reset completes.

### (3) Register Settings after Reset

If you perform a reset on TCFLASH from software, TCFLASH is reset with a minimum of 600 ns. After the reset, the software must wait for the RDY bit in the TCFCFGn\_FSTAT register of the unit to be configured to become "1" before reading from flash memory and writing a command.

### (4) Instruction Fetch from Flash Memory

During execution of the program or erase operation, you cannot read data from TCFLASH. For this reason, copy the required data and programs from TCFLASH to RAM before starting the write or erase operation, so that they do not need to be read partway through writing or erasing.

### (5) Waiting for the Completion of a Reset from Software

Flash memory is reset by writing "1" to the SWFRST bit in the TCFCFGn\_FCFGR register. After performing a reset on a unit, monitor the RDY bit in the TCFCFGn\_FSTAT register of the unit, and then wait for the completion of the reset before accessing it.



## CHAPTER 17: TCRAM Interface

This chapter provides an overview of the interface between the Cortex(TM)-R5F BTCM port and SRAM, and describes their configuration, operation, and registers.

---

1. Overview
2. Configuration
3. Operation
4. Setting Procedure Examples
5. Registers
6. Notes



## 1. Overview

This section describes the features of the TCRAM interface.

### Features

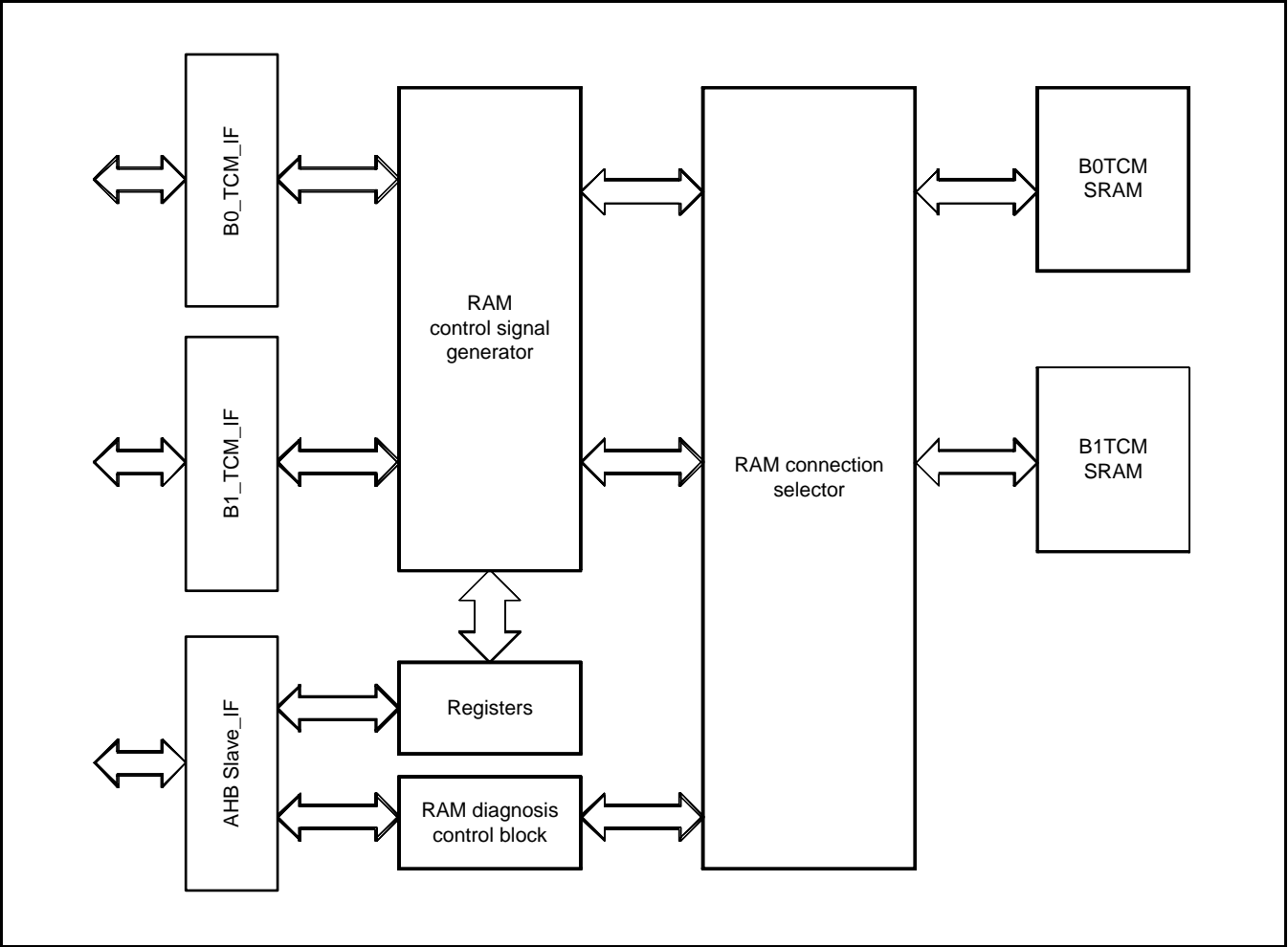
The TCRAM interface has the following features.

- Connection between the Cortex-R5F BTCM port and SRAM (64 KB x 2 modules)  
For the BTCM port, you can use 2 ports as options (B0TCM and B1TCM), which can be interleaved. When 2 ports are used, the capacity of each port is half of all the BTCM ports.
- Wait control function  
The Cortex-R5F BTCM port has a signal for waiting for data read (BxTCWAIT).  
The TCRAM interface sets and controls the number of cycles of the wait signal (for data write, there is no wait control).
- Generation of a bus error response  
The TCRAM interface generates an error when non-privilege access or unsupported access to the configuration register is attempted.  
For details on unsupported access, see "(2) Generation of a Bus Error Response" in Section "3. Operation".
- ECC error insertion function  
The Cortex-R5F performs an ECC check.  
The TCRAM interface can insert an error in data read from the TCRAM for the ECC function test performed by the Cortex-R5F.
- RAM diagnosis and initialization function  
Diagnosis and initialization are performed for the TCRAM. One or more methods from the following can be selected to perform diagnosis.
  - Unique (unique data is {address [6:0], 4 {address [7:0]}}. Upper 7 bits are ECC data.)
  - Checker
  - March (performed in the order of all "0's" to all "1's")

2. Configuration

Figure 2-1 is a block diagram of the TCRAM interface that uses B0TCM and B1TCM.

Figure 2-1 Block Diagram of TCRAM Interface





**(1) B0\_TCM\_IF**

Interface for the Cortex-R5F B0TCM port

**(2) B1\_TCM\_IF**

Interface for the Cortex-R5F B1TCM port

**(3) AHB Slave I/F**

AHB Slave interface, which is used as a bus for configuration

**(4) Register block**

Includes the registers of the TCRAM interface.

The registers are common to the B0TCM and B1TCM, and set values are reflected in both ports.

**(5) RAM control signal generation block**

Generates control signals of the TCRAM from signals input at the Cortex-R5F BTCM port.

**(6) RAM diagnosis control block**

Controls diagnosis and initialization of the SRAM.

**(7) RAM connection signal selection block**

Selects the BTCM port or RAM diagnosis control block as a path for connecting to the SRAM.

### 3. Operation

This section describes the operation of the TCRAM interface.

#### (1) Wait Control

The wait status can be configured by setting TRCFGn\_TCMCFG0:DWAIT via the configuration bus when privilege access is allowed and the TRCFGn\_TCMCFG0:LOCKSTATUS bit is "0".

The number of waits that can be set is from 0 to 3, and the default set value is the least number of waits (0 waits).

The asserted BxTCWAIT signal indicates the wait status.

The Cortex-R5F is kept waiting for acquisition of data read from the SRAM while the BxTCWAIT signal is asserted. There is no wait regarding write operation.

#### (2) Generation of a Bus Error Response

The TCRAM interface returns an error response to the configuration bus under the following conditions.

- Non-privilege write access to a TCRM interface register
- Write access to a TCRAM interface register that is not the TRCFGn\_TCMUNLOCK register in the locked state
- Write access to the TRCFGn\_TCMUNLOCK register with the unlock value or any value except the lock value
- Write access that is not 32-bit access to the TRCFGn\_TCMUNLOCK register
- Write access to a reserved space
- Read or write access to a space that is not used as a register space
- Write access when all the write target bits are reserved or read-only bits

**Note:**

- When writing to TRCFGn\_TTCR[15:8] in byte size, a bus error response is not generated.

#### (3) ECC Error Insertion

This interface has a function to insert an error into data that is read from the SRAM so that the 32-bit ECC function of the Cortex-R5F can be tested.

When the TRCFGn\_TCMCFG0:LOCKSTATUS bit is "0", this function sets "1" in a bit in the configuration register that causes an ECC error in the RAM.

To generate an ECC error by data area destruction, set the TRCFGn\_TCMCFG1:ERRBIT[31:0] bits, and to generate one by ECC area destruction, set the TRCFGn\_TCMCFG0:ERRECC[6:0] bits.

For read from the TCRAM the read data is XOR-ed with TRCFGn\_TCMCFG1:ERRBIT[31:0], and the ECC data is XOR-ed with TRCFGn\_TCMCFG0:ERRECC[6:0].

**Note:**

- During normal operation, keep TRCFGn\_TCMCFG1:ERRBIT[31:0] and TRCFGn\_TCMCFG0:ERRECC[6:0] set to "0".





#### (4) RAM Diagnosis and Initialization Function

##### a) RAM Diagnosis

Select RAM diagnosis items to perform for the TCRAM from the following. (Multiple items can be selected.)

- Unique (unique data is {address [6:0], 4 {address [7:0]}}. Upper 7 bits are ECC data.)
- Checker
- March (performed in the order of all "0's" to all "1's")

The RAM diagnosis is performed only in the following sequence.

- Unique
- Checker
- March

Diagnosis items to be performed are selected according to the setting of the TTYP[2:0] bits in the test diagnosis function register (TRCFGn\_TTCR). (unique and checker are selected by default.)

The range for Diagnosis can be defined with test start address register (TRCFGn\_TASAR) and test end address register (TRCFGn\_TAEAR).

Diagnosis is initiated with the key code control by software.

The following shows the RAM diagnosis start procedure.

1. Wait until TRUN bit of test diagnosis function register (TRCFGn\_TTCR) and IRUN bit of test initialization function register both become "0" by reading these bits.
2. Clear TRCFGn\_TTCR:TCI and TRCFGn\_TICR:ICI.  
To clear TRCFGn\_TTCR:TCI and TRCFGn\_TICR:ICI, write "1" to the corresponding bit of TRCFGn\_TSCR.
3. To the test key code control register (TRCFGn\_TKCCR), write "0x02", "0x42", "0x82", "0xC2" in a row in this sequence to test key code control register to start Diagnosis.
4. Wait until Diagnosis finishes (TRCFGn\_TTCR:TCI=1).

##### Notes:

- When you use this function, all the data bits of the RAM contain "1" after the march diagnosis, or some values after diagnosis other than it.
- If data on the RAM is read before initialization or 32- or 64-bit write access, an ECC error occurs. Therefore, be sure to perform initialization or write access.
- Also, it is prohibited to access the RAM when it is under diagnosis.

#### b) RAM Initialization

For the TCRAM initialization operation, you can select either of the following by the specification of the ITYP bit in the test initialization function register (TRCFGn\_TICR).

- Writing all "0's" (default)
- Writing all "1's"

To the ECC area, the value corresponding to the written value is written.

You can specify a range for performing RAM initialization with the test start address register (TRCFGn\_TASAR) and test end address register (TRCFGn\_TAEAR).

The following shows the RAM initialization start procedure.

1. Wait until TRUN bit of test diagnosis function register (TRCFGn\_TTCR) and IRUN bit of test initialization function register both become "0" by reading these bits.
2. Clear TRCFGn\_TTCR:TCI and TRCFGn\_TICR:ICI.  
To clear TRCFGn\_TTCR:TCI and TRCFGn\_TICR:ICI, write "1" to the corresponding bit of TRCFGn\_TSCR.
3. To the test key code control register (TRCFGn\_TKCCR), write "0x01", "0x41", "0x81", and "0xC1" in this sequence in a row to start initialization.
4. Wait until diagnosis finishes (TRCFGn\_TICR:ICI=1).

**Note:**

- *It is prohibited to access the RAM when it is being initialized.*

#### c) Forcibly Stopping RAM Diagnosis and Initialization

You can forcibly stop the execution of diagnosis and initialization of the TCRAM.

The following shows the procedure for the forcible stop.

1. During the diagnosis or initialization, to the test key code control register (TRCFGn\_TKCCR), write "0x00", "0x40", "0x80", and "0xC0" in this sequence in a row.
2. Wait until TRCFGn\_TTCR:TRUN becomes "0" for diagnosis, and wait until TRCFGn\_TICR:IRUN is "0" for initialization.



#### d) Operation When a RAM Diagnosis Error Occurs

If an error occurs during RAM diagnosis, the contents of the diagnosis in which the error occurs and the address where it occurs are stored in the TRCFGn\_TEAR0 to 2 registers.

When 4 or more errors occur, TRCFGn\_TTCR:OVFLW is set to "1".

#### e) RAM Diagnosis Interrupt

You can generate an interrupt for RAM diagnosis and initialization.

Table 3-1 shows interrupt factors, generation flag bits, and enable bits.

**Table 3-1 Correspondence between Interrupt Factors, Generation Flag Bits, and Enable Bits**

Interrupt Factor	Generation Flag Bit	Enable Bit
Diagnosis end interrupt	TRCFGn_TTCR:TCI	TRCFGn_TTCR:TCIE
Diagnosis error interrupt	TRCFGn_TTCR:TEI	TRCFGn_TTCR:TEIE
Initialization end interrupt	TRCFGn_TICR:ICI	TRCFGn_TICR:ICIE

The TCRAM interface output 1 interrupt signal collectively indicates the above 3 types of interrupts.

You can check what type of interrupt has occurred by reading the TCI and TEI bits in TRCFGn\_TTCR and the ICI bit in TRCFGn\_TICR.

(By reading data at the offset address 0x0000\_0040 in units of words, you can check it by 1-time read access.)

#### f) Generation of a RAM Diagnosis Pseudo Error

With this function, you can generate a pseudo error on purpose for debugging software.

The operation for generating a pseudo error is set as the following procedure.

1. Selecting an error type by setting the Test pseudo error generation control register (TRCFGn\_TFECR)
  - Set a diagnosis pattern that generates a pseudo error in TRCFGn\_TFECR:ETYP[2:0].
  - Identify a diagnosis pattern that generates a pseudo error by writing "1" in TRCFGn\_TFECR:FERR.
2. Starting RAM diagnosis (For the procedure see Figure 4-2.)

4. Setting Procedure Examples

Figure 4-1 shows a procedure for register setting of the TCRAM interface, and Figure 4-2 shows a procedure for starting RAM diagnosis and initialization.

Figure 4-1 Register Setting Flow

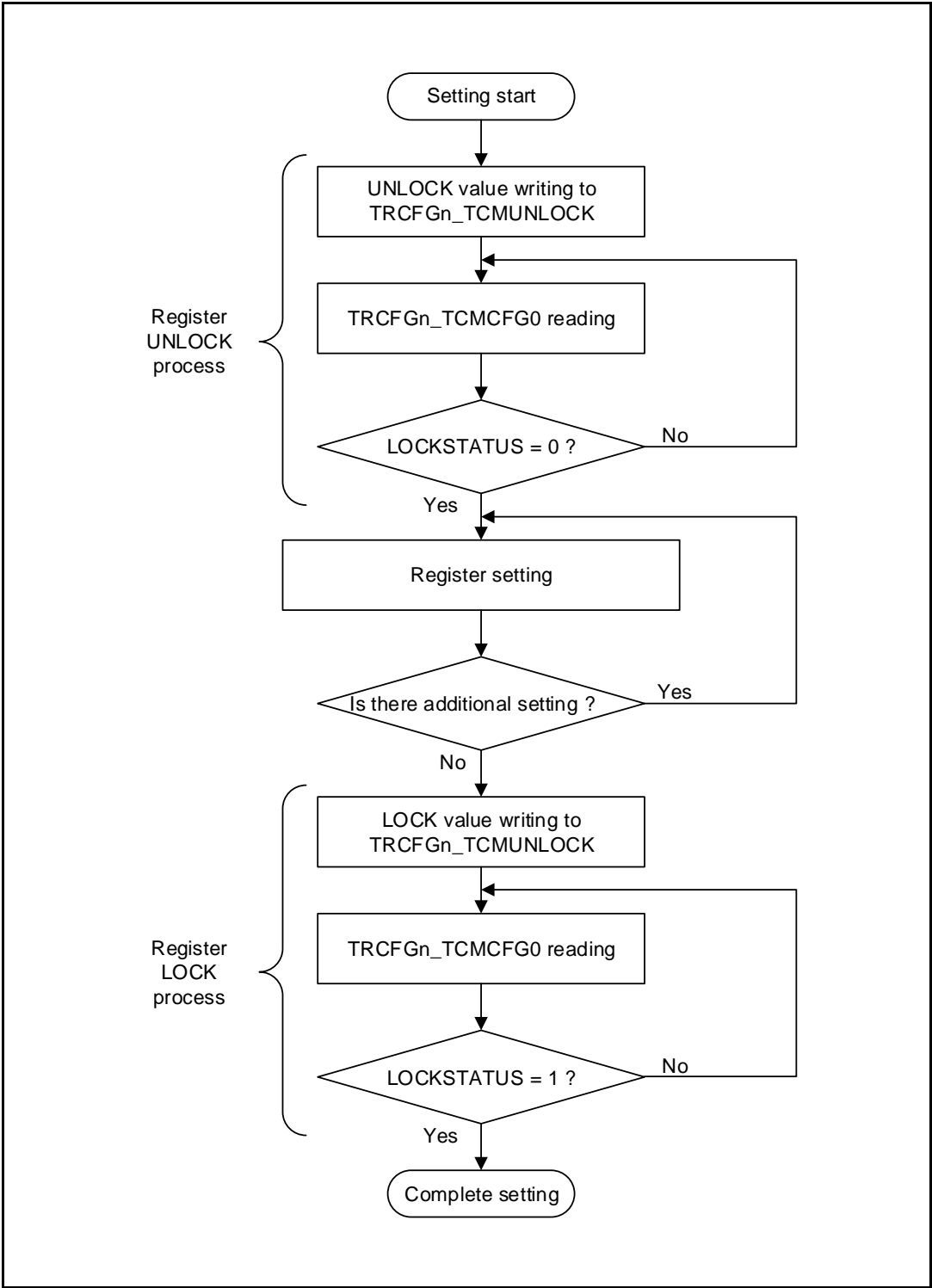
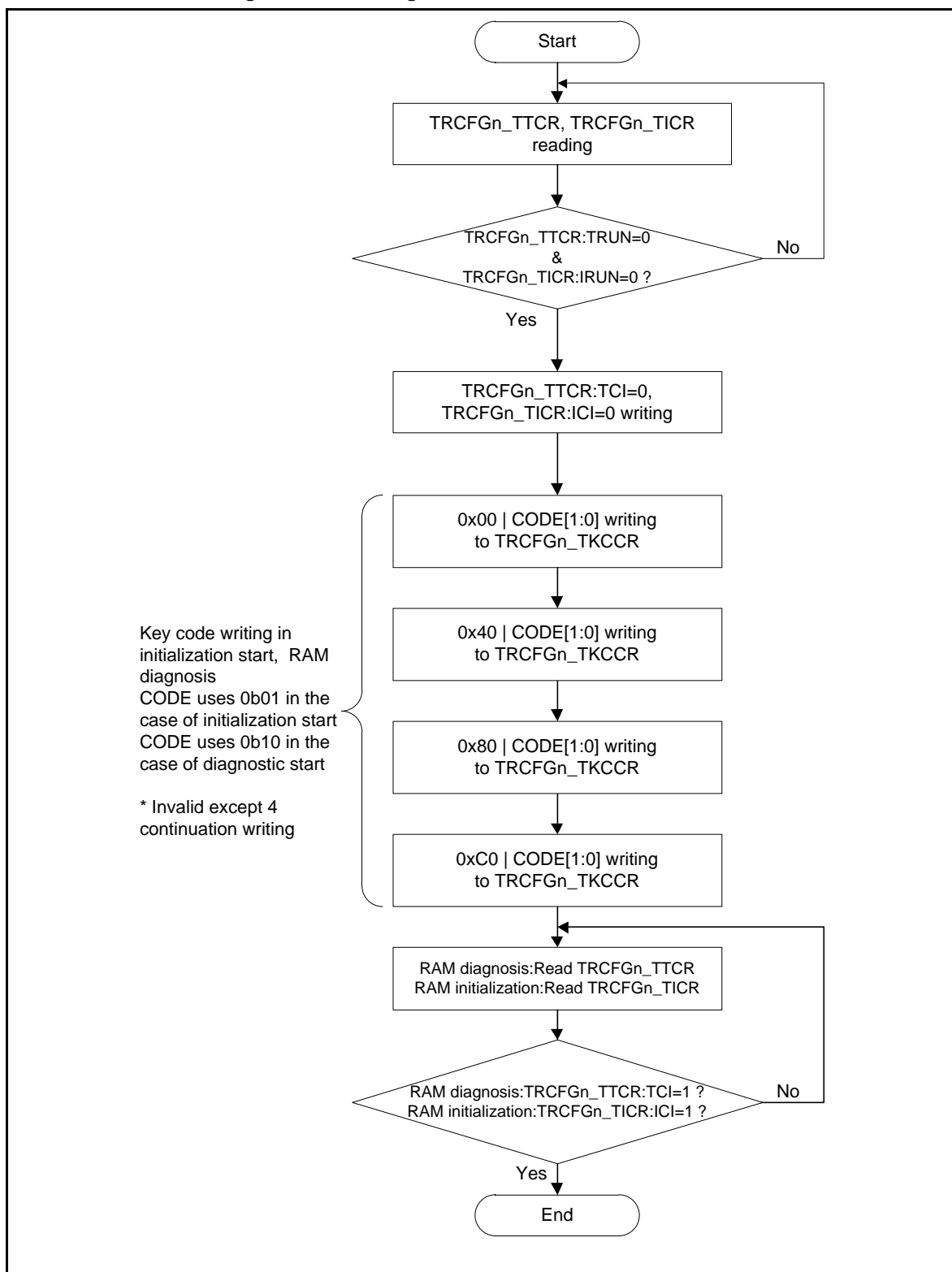


Figure 4-2 RAM Diagnosis and Initialization Start Flow



## 5. Registers

This section provides a list of the TCRAM interface registers.

When you use the B1TCM port, register setting affects both B0TCM and B1TCM.

**Table 5-1 List of TCRAM Interface Registers**

Abbreviated Register Name	Register Name	See
TRCFGn_TCMCFG0	Configuration Register 0	5.1
TRCFGn_TCMCFG1	Configuration Register 1	5.2
TRCFGn_TCMUNLOCK	Unlock Register	5.3
TRCFGn_TEAR0	Test Error Address Register 0	5.4
TRCFGn_TEAR1	Test Error Address Register 1	5.5
TRCFGn_TEAR2	Test Error Address Register 2	5.6
TRCFGn_TAEAR	Test End Address Register	5.7
TRCFGn_TASAR	Test Start Address Register	5.8
TRCFGn_TFEER	Test Pseudo Error Generation Control Register	5.9
TRCFGn_TICR	Test Initialization Function Register	5.10
TRCFGn_TTCR	Test Diagnosis Function Register	5.11
TRCFGn_TSRCR	Test Soft Reset Generation Control Register	5.12
TRCFGn_TKCCR	Test Key Code Control Register	5.13



## 5.1. TCRAM IF Configuration Register 0 (TRCFGn\_TCMCFG0)

This register has 7 bits for ERRECC data for the BTCM port, 2 bits for setting the number of waits, and the LOCKSTATUS bit that indicates the locked state or unlocked state of the TCRAM IF configuration register.

You can perform the write operation for this register only when privilege access is possible and TRCFGn\_TCMCFG0:LOCKSTATUS is "0".

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved						DWAIT	
ACCESS_TYPE	R0,WX						R/W	
PROT_TYPE	WPS							
INITIAL_VALUE	000000						00	

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved							LOCKSTATUS
ACCESS_TYPE	R0,WX							R,WX
PROT_TYPE	WPS							
INITIAL_VALUE	0000000							1

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved	ERRECC						
ACCESS_TYPE	R0,WX	R/W						
PROT_TYPE	WPS							
INITIAL_VALUE	0	0000000						

[bit31:26] Reserved: Reserved bits

[bit25:24] DWAIT[1:0]: Number of data wait bits

bit25:24	Description
Write	The number of read waits for the BTCM port (0 to 3) is set.
Read	The number of read waits for the BTCM port that has been set can be read.

[bit23:9] Reserved: Reserved bits

[bit8] LOCKSTATUS: Lock status bit

bit	Description
Write	No effect
Read	0: The TCRAM interface registers are in the unlocked state. 1: The TCRAM interface registers are in the locked state. (Initial value)

**[bit7] Reserved: Reserved bit**

**[bit6:0] ERRECC[6:0]: ECC data error insertion bits**

bit6:0	Description
Write	The data resulting from the XOR operation of these bits and ECC data read from the SRAM is input to the Cortex-R5F BTCM port. In the case of 64-bit read, data whose higher 7 bits and lower 7 bits are XOR-ed with these bits is input to the BTCM port.
Read	Set value read





## 5.2. TCRAM IF Configuration Register 1 (TRCFGn\_TCMCFG1)

This register is used to insert an error in data read from the TCRAM for the purpose of testing the ECC function of the Cortex-R5F BTCM port.

You can perform the write operation for this register only when privilege access is possible and TRCFGn\_TCMCFG0:LOCKSTATUS is "0".

BIT_OFFSET	31-0
BIT_NAME	ERRBIT
ACCESS_TYPE	R/W
PROT_TYPE	WPS
INITIAL_VALUE	00000000_00000000_00000000_00000000

### [bit31:0] ERRBIT[31:0]: ECC error insertion bits

bit31:0	Description
Write	The data resulting from the XOR operation of these bits and data read from the SRAM is input to the Cortex-R5F BTCM port. In the case of 64-bit read, data whose higher word and lower word are XOR-ed with these bits is input to the BTCM port.
Read	Set value read

### 5.3. TCRAM IF Unlock Register (TRCFGn\_TCMUNLOCK)

This register locks or unlocks write access to the TCRAM interface registers.

Writing the correct unlock value (0xACC55ECC) to this register enables write access to the registers.

After setting registers, writing the correct lock value (0x5ECCB10C) to this register disables write access to the registers.

You must access this register in units of words, and can access it only with privilege access.

BIT_OFFSET	31-0
BIT_NAME	UNLOCK
ACCESS_TYPE	R0,W
PROT_TYPE	WPS
INITIAL_VALUE	00000000_00000000_00000000_00000000

#### [bit31:0] UNLOCK[31:0]: Lock/Unlock bits

bit31:0	Description
Write	<p>Locks and unlocks write accesses to the registers.</p> <p>0xACC55ECC: TRCFGn_TCMCFG0:LOCKSTATUS becomes "0" and write access to the TCRAM interface registers is enabled.</p> <p>0x5ECCB10C: TRCFGn_TCMCFG0:LOCKSTATUS becomes "1" and write access to the TCRAM interface registers is disabled.</p> <p>Other values: An error is generated.</p> <p>For details on the register setting procedure, see Figure 4-1.</p>
Read	"0" is always read.



## 5.4. TCRAM IF Test Error Address Register 0 (TRCFGn\_TEAR0)

This register holds the address for when an error occurs during RAM diagnosis.

Also, this register holds the source flag indicating the diagnosis pattern that has caused this error.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	TER			Reserved				
ACCESS_TYPE	R,WX			R0,WX				
PROT_TYPE	-							
INITIAL_VALUE	000			00000				

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved	ERR_ADDR[14:8]						
ACCESS_TYPE	R0,WX	R,WX						
PROT_TYPE	-							
INITIAL_VALUE	0	0000000						

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	ERR_ADDR[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

### [bit31:29]TER[2:0]: Diagnosis error source identification

bit31:29	Description
Write	Writing data to these bits causes a bus error.
Read	<p>Holds the diagnosis pattern when an error occurs during RAM diagnosis. ERR_ADDR[14:0] is effective only when any of these bits is set to "1".</p> <p>0b001: Error occurred in the march diagnosis</p> <p>0b010: Error occurred in the checker diagnosis</p> <p>0b100: Error occurred in the unique diagnosis</p> <p>0b000: No error occurred</p> <p>A RAM diagnosis start instruction triggers the hardware to initialize (clear to "000") these bits.</p>

### [bit28:15] Reserved: Reserved bits

**[bit14:0] ERR\_ADDR[14:0]: Error occurrence address**

bit14:0	Description
Write	Writing data to these bits causes a bus error.
Read	Holds the address when an error occurs during RAM diagnosis. These bits indicate a valid value only when TER is not "000". Triggered by the specification of RAM diagnosis start, the hardware initializes these bits (clears them to "0b0000000000000000").

**Notes:**

- When any of the TER bits becomes "1", even if an error occurs during another diagnosis item, the corresponding error source bit does not become "1".
- ERR\_ADDR is an offset address in units of words (RAM address).
- To calculate the absolute address, add {ERR\_ADDR, 0b00} to the base address.



## 5.5. TCRAM IF Test Error Address Register 1 (TRCFGn\_TEAR1)

This register holds the address for when an error occurs during RAM diagnosis at an address that is different from the one held in TRCFGn\_TEAR0.

Also, this register holds the source flag indicating the diagnosis pattern that has caused this error.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	TER[2:0]			Reserved				
ACCESS_TYPE	R,WX			R0,WX				
PROT_TYPE	-							
INITIAL_VALUE	000			00000				

BITS	23	22	21	20	19	18	17	16
BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved	ERR_ADDR[14:8]						
ACCESS_TYPE	R0,WX	R,WX						
PROT_TYPE	-							
INITIAL_VALUE	0	0000000						

BITS	7	6	5	4	3	2	1	0
BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	ERR_ADDR[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

### [bit31:29]TER[2:0]: Diagnosis error source identification

bit31:29	Description
Write	Writing data to these bits causes a bus error.
Read	<p>Holds the diagnosis pattern when an error occurs during RAM diagnosis. ERR_ADDR[14:0] is effective only when any of these bits is set to "1".</p> <p>0b001: Error occurred in the march diagnosis  0b010: Error occurred in the checker diagnosis  0b100: Error occurred in the unique diagnosis  0b000: No error occurred</p> <p>A RAM diagnosis start instruction triggers the hardware to initialize (clear to "000") these bits.</p> <p>Note:  When any of these bits becomes "1", even if an error occurs during another diagnosis item, the corresponding error source bit does not become "1".</p>

### [bit28:15] Reserved: Reserved bits

**[bit14:0] ERR\_ADDR[14:0]: Error occurrence address**

bit14:0	Description
Write	Writing data to these bits causes a bus error.
Read	Holds the address when an error occurs during RAM diagnosis. These bits are effective only when TER is not "000". A RAM diagnosis start instruction triggers the hardware to initialize (clear to "0b0000000000000000") these bits.

**Notes:**

- When any of the TER bits becomes "1", even if an error occurs during another diagnosis item, the corresponding error source bit does not become "1".
- ERR\_ADDR is an offset address in units of words (RAM address).
- To calculate the absolute address, add {ERR\_ADDR, 0b00} to the base address.



## 5.6. TCRAM IF Test Error Address Register 2 (TRCFGn\_TEAR2)

This register holds the address for when an error occurs during RAM diagnosis at an address that is different from the one held in TRCFGn\_TEAR0 and TRCFGn\_TEAR1.

Also, this register holds the source flag indicating the diagnosis pattern that has caused this error.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	TER			Reserved				
ACCESS_TYPE	R,WX			R0,WX				
PROT_TYPE	-							
INITIAL_VALUE	000			00000				

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved	ERR_ADDR[14:8]						
ACCESS_TYPE	R0,WX	R,WX						
PROT_TYPE	-							
INITIAL_VALUE	0	0000000						

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	ERR_ADDR[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

### [bit31:29]TER[2:0]: Diagnosis error source identification

bit31:29	Description
Write	Writing data to these bits causes a bus error.
Read	<p>Holds the diagnosis pattern when an error occurs during RAM diagnosis. ERR_ADDR[14:0] is effective only when any of these bits is set to "1".</p> <p>0b001: Error occurred in the march diagnosis  0b010: Error occurred in the checker diagnosis  0b100: Error occurred in the unique diagnosis  0b000: No error occurred</p> <p>A RAM diagnosis start instruction triggers the hardware to initialize (clear to "000") these bits.</p> <p>Note:  When any of these bits becomes "1", even if an error occurs during another diagnosis item, the corresponding error source bit does not become "1".</p>

### [bit28:15] Reserved: Reserved bits

**[bit14:0] ERR\_ADDR[14:0]: Error occurrence address**

bit14:0	Description
Write	Writing data to these bits causes a bus error.
Read	Holds the address when an error occurs during RAM diagnosis. These bits are effective only when TER is not "000". A RAM diagnosis start instruction triggers the hardware to initialize (clear to "0b0000000000000000") these bits.

**Notes:**

- When any of the TER bits becomes "1", even if an error occurs during another diagnosis item, the corresponding error source bit does not become "1".
- ERR\_ADDR is an offset address in units of words (RAM address).
- To calculate the absolute address, add {ERR\_ADDR, 0b00} to the base address.





## 5.7. TCRAM IF Test End Address Register (TRCFGn\_TAEAR)

This register specifies the end address for performing RAM diagnosis or initialization.

You can perform the write operation for this register only when privilege access is possible and TRCFGn\_TCMCFG0:LOCKSTATUS is "0".

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved	EADDR[14:8]						
ACCESS_TYPE	R0,WX	R/W						
PROT_TYPE	WPS							
INITIAL_VALUE	0	1111111						

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	EADDR[7:0]							
ACCESS_TYPE	R/W							
PROT_TYPE	WPS							
INITIAL_VALUE	11111111							

[bit15] Reserved: Reserved bit

[bit14:0] EADDR[14:0]: RAM diagnosis end address

bit14:0	Description
Write	You can set an address at which RAM diagnosis or initialization operation ends. A word address of the RAM is set.
Read	You can read the address at which specified RAM diagnosis or initialization operation ends.

### Notes:

- EADDR is an offset in word units (RAM address).
- Obtain the absolute address by adding {EADDR, 0b00} to the base address.
- Setting the address where RAM area is exceeded to EADDR[14:0] is prohibited.

## 5.8. TCRAM IF Test Start Address Register (TRCFGn\_TASAR)

This register specifies the start address for performing RAM diagnosis or initialization.

You can perform the write operation for this register only when privilege access is possible and TRCFGn\_TCMCFG0:LOCKSTATUS is "0".

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved	SADDR[14:8]						
ACCESS_TYPE	R0,WX	R/W						
PROT_TYPE	WPS							
INITIAL_VALUE	0	0000000						

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	SADDR[7:0]							
ACCESS_TYPE	R/W							
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							

[bit15] Reserved: Reserved bit

[bit14:0] SADDR[14:0]: RAM diagnosis start address

bit14:0	Description
Write	You can set an address at which RAM diagnosis or initialization operation starts. A word address of the RAM is set.
Read	You can read the address at which specified RAM diagnosis or initialization operation starts.

**Notes:**

- SADDR is an offset address in units of words (RAM address).
- To calculate the absolute address, add {SADDR, 0b00} to the base address.



## 5.9. TCRAM IF Test Pseudo Error Generation Control Register (TRCFGn\_TFECR)

This register specifies the RAM diagnosis operation in which a pseudo error is to be generated.

You can perform the write operation for this register only when privilege access is possible and TRCFGn\_TCMCFG0:LOCKSTATUS is "0".

BIT_OFFSET	7	6	5	4	3	2	1	0
Reserved	Reserved				FERR	ETYP		
R0,WX	R0,WX				R/W	R/W		
PROT_TYPE	-							
INITIAL_VALUE	0000				0	000		

**[bit7:4] Reserved: Reserved bits**

**[bit3]FERR: RAM diagnosis pseudo error generation enable**

bit	Description
Write	0: Disable generation of pseudo errors. (Normal operation) 1: Enable generation of pseudo errors. Erroneous data is deliberately written in accordance with the ETYP setting.
Read	The enable/disable setting for generation of pseudo-errors can be read.

**[bit2:0] ETYP[2:0]: Pseudo error generation processing specification**

bit2:0	Description
Write	0bxx1: Generate pseudo-errors during march diagnosis. 0bx1x: Generate pseudo-errors during checker diagnosis. 0b1xx: Generate pseudo-errors during unique diagnosis. 0b000: Generate no pseudo error. (x: don't care)
Read	This register can read the setting value for the diagnosis in which the pseudo-error is generated.

## 5.10. TCRAM IF Test Initialization Function Register (TRCFGn\_TICR)

This register specifies contents of RAM initialization, and holds the result and state of the initialization.

You can perform the write operation for this register only when privilege access is possible and TRCFGn\_TCMCFG0:LOCKSTATUS is "0".

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved				ICIE	ICI	ITYP	IRUN
ACCESS_TYPE	R0,WX				R/W	R,W	R/W	R,WX
PROT_TYPE	WPS							
INITIAL_VALUE	0000				0	0	0	0

**[bit7:4] Reserved: Reserved bits**

**[bit3] ICIE: RAM initialization end source interrupt enable**

bit	Description
Write	0: Disable interrupts triggered by RAM initialization end. 1: Enable interrupts triggered by RAM initialization end.
Read	You can read the setting for enabling or disabling the initialization end source interrupt.

**[bit2] ICI: RAM initialization completion**

bit	Description
Write	No effect
Read	0: Initialization has not been completed or has not been started. 1: RAM initialization has been completed. This bit is not set for forcible stop by the key code.

**[bit1] ITYP: RAM initialization contents specification**

bit	Description
Write	0: Initialize with all "0's". 1: Initialize with all "1's".
Read	You can read the setting of the initialization contents specification.

**[bit0] IRUN: RAM initialization operation status**

bit	Description
Write	No effect
Read	0: Initialization is not in progress. 1: Initialization is in progress.



## 5.11. TCRAM IF Test Diagnosis Function Register (TRCFGn\_TTCR)

This register specifies the contents of RAM diagnosis, and holds the result and state of the diagnosis.

You can perform the write operation for this register only when privilege access is possible and TRCFGn\_TCMCFG0:LOCKSTATUS is "0".

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved						TSTAT	OVFLW
ACCESS_TYPE	R0,WX						R,WX	R,WX
PROT_TYPE	WPS							
INITIAL_VALUE	000000						0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	TEIE	TEI	TCIE	TCI	TTYP			TRUN
ACCESS_TYPE	R/W	R,W	R/W	R,W	R/W			R,WX
PROT_TYPE	WPS							
INITIAL_VALUE	0	0	0	0	110			0

[bit15:10] Reserved: Reserved bits

[bit9] TSTAT: RAM diagnosis error detection

bit	Description
Write	No effect
Read	0: No error has been detected in RAM diagnosis. 1: An error has been detected in RAM diagnosis. A RAM diagnosis start instruction triggers the hardware to initialize (clear to "0") this bit.

[bit8] OVFLW: RAM diagnosis error overflow

bit	Description
Write	No effect
Read	0: RAM diagnosis errors have occurred at 3 addresses or less. 1: RAM diagnosis errors have occurred at 4 addresses or more. A RAM diagnosis start instruction triggers the hardware to initialize (clear to "0") this bit.

[bit7] TEIE: Enable error occurrence interrupt during diagnosis

bit	Description
Write	0: Disable interrupts triggered by RAM diagnosis errors. 1: Enable interrupts triggered by RAM diagnosis errors. When TTCR:TEI is "1", "H" is output to the interrupt signal.
Read	You can read the setting for enabling or disabling the error interrupts during diagnosis.

[bit6] TEI: Diagnosis-time error generation

bit	Description
Write	No effect
Read	0: TRCFGn_TTCR:TSTAT = "0" when RAM diagnosis ends. 1: TRCFGn_TTCR:TSTAT = "1" when RAM diagnosis ends.

**[bit5] TCIE: Diagnosis end source interrupt enable**

bit	Description
Write	0: Disable interrupts triggered by RAM diagnosis end. 1: Enable interrupts triggered by RAM diagnosis end. When TTCR:TCI is "1", "H" is output to the interrupt signal.
Read	You can read the setting for enabling or disabling the diagnosis end source interrupt.

**[bit4] TCI: Diagnosis end**

bit	Description
Write	No effect
Read	0: The diagnosis has not completed or has been stopped. 1: RAM diagnosis has been completed. This bit is not set for forcible stop by the key code.

**[bit3:1] TYP[2:0]: RAM diagnosis contents specification**

bit3:1	Description
Write	0bxx1: Perform march diagnosis. 0bxx0: Do not perform march diagnosis. 0bx1x: Perform checker diagnosis. 0bx0x: Do not perform checker diagnosis. 0b1xx: Perform unique diagnosis. 0b0xx: Do not perform unique diagnosis. RAM diagnoses are performed in the following order. Unique diagnosis (The address itself is used as the data.) Checker diagnosis March diagnosis (all "0" and then all "1")
Read	You can read the set value for the types to be executed in RAM diagnosis.

**[bit0] TRUN: RAM diagnosis operation status**

bit	Description
Write	No effect
Read	0: RAM diagnosis is not in progress. 1: RAM diagnosis is in progress.

**Note:**

- Be sure to make the setting for this register before starting RAM diagnosis.



## 5.12. TCRAM IF Test Soft Reset Generation Control Register (TRCFGn\_TSRCR)

This register resets the entire circuit related to RAM diagnosis except this register itself.

You can perform the write operation for this register only when privilege access is possible and TRCFGn\_TCMCFG0:LOCKSTATUS is "0".

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	SRST	Reserved						
ACCESS_TYPE	R0,W	R0,WX						
PROT_TYPE	WPS							
INITIAL_VALUE	0	0000000						

### [bit7] SRST: Software reset

bit	Description
Write	0: No effect 1: Reset all circuits that are related to RAM diagnosis except for this register.
Read	"0" is always read.

### [bit6:0] Reserved: Reserved bits

### 5.13. TCRAM IF Test Key Code Control Register (TRCFGn\_TKCCR)

This register is used to start or to forcibly stop RAM diagnosis or initialization.

You can perform the write operation for this register only when privilege access is possible and TRCFGn\_TCMCFG0:LOCKSTATUS is "0".

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	KEY		Reserved				CODE	
ACCESS_TYPE	R0,W		R0,WX				R/W	
PROT_TYPE	WPS							
INITIAL_VALUE	00		0000				00	

#### [bit7:6] KEY[1:0]: Key code control

bit7:6	Description
Write	<p>These bits control the key code.</p> <p>The operation specified in the CODE bits is performed by writing in the order of 0b00, 0b01, 0b10, and 0b11.</p> <p>In addition, if you have accessed another RAM test register or have performed an operation other than that described above (read operation or a series of write operations other than the above) during key code operation, start the operation over again.</p> <p>For the detailed setting procedure of these bits, see Figure 4-2.</p>
Read	"0" is always read.

#### [bit5:2] Reserved: Reserved bits

#### [bit1:0] CODE[1:0]: Operation specification

bit1:0	Description
Write	<p>0b00: Forcibly terminate</p> <p>0b01: Activate initialization</p> <p>0b10: Activate diagnosis</p> <p>0b11: Setting prohibited</p>
Read	The set value of operation specification can be read.

#### Notes:

- During KEY bits operation, set the same value in the CODE bits.
- Setting the value 0b11 to the CODE bits is prohibited.





## 6. Notes

This section explains notes when using the TCRAM.

### **ECC Check**

Since the following access to an uninitialized area of the TCRAM causes an ECC error, 32-bit initialization is necessary before using it.

- 8- or 16-bit write access
- Read access

The Cortex-R5F performs an ECC check for the TCRAM in units of 32 bits. For 32- or 64-bit write operation, add ECC to data and then write it to the TCRAM. For 8- or 16-bit write operation, perform read-modify-write access so that the correct ECC can be generated. An ECC check is also performed during read operation in read-modify-write access.



## CHAPTER 18: WorkFLASH

This chapter explains WorkFLASH.

---

1. Overview
2. Configuration
3. Operation
4. Setting Procedure
5. Registers
6. Others



## 1. Overview

This section provides an overview of WorkFLASH.

WorkFLASH is rewritable, non-volatile data memory with a built-in MCU.

Some features of WorkFLASH are listed below.

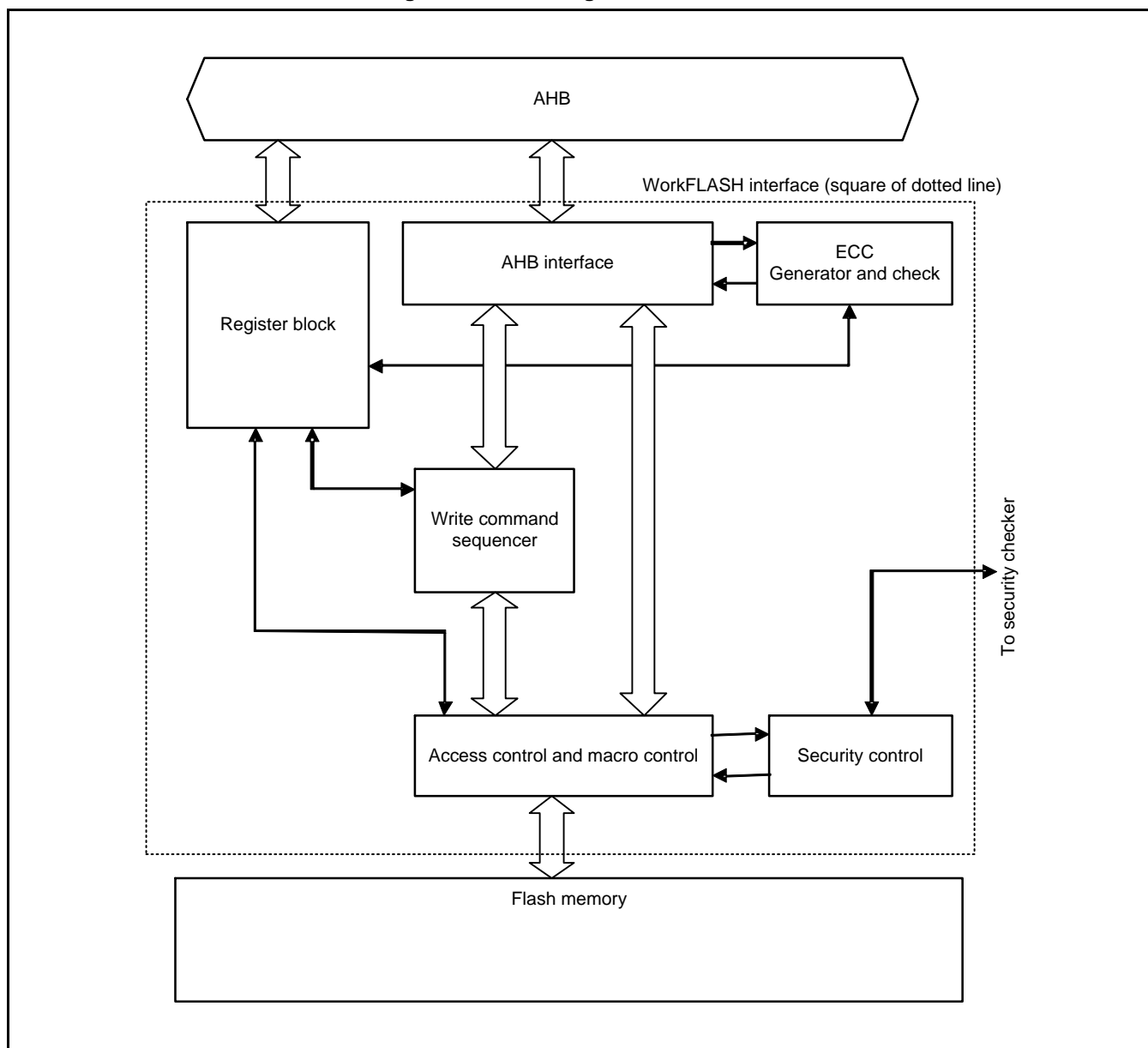
- Enables reading in units of 8/16/32/64 bits.
- Provides an ECC (SEC/DED) function.
- Enables switching between the enabled and disabled states of the ECC function through a register setting.
- Enables switching between ECC-enabled access and ECC-disabled access by using two mirror areas.
- Enables writing in units of 32 bits when ECC is enabled.
- Enables writing in units of 8/16/32 bits when ECC is disabled.
- Equipped with an ECC check circuit test function with error injection.
- Enables writing from not only a CPU but also the DMAC.
- Enables writing from an unprivileged mode.
- Equipped with a control register setting protection function using a key code.
- Equipped with a Flash security function for data protection.
- Equipped with a bus error response function for access to reserved areas.
- Enables interrupt generation with writing/erasing completed.

## 2. Configuration

This section explains the configuration within WorkFLASH.

## 2.1. Block Diagrams

### Figure 2-1 Block Diagram of WorkFLASH





## 2.2. Address/Sector Maps in User Mode

This section explains WorkFLASH address/sector maps in user mode.

In this series, an individual product is provided with 1 to 4 WorkFLASHs depending on the product. (This product has 2 WorkFLASHs as it has 2 CPUs.)

WorkFLASH has the addresses placed on the memory map in a 3-MB area between 0x0E000000 and 0x0E2FFFFFF. As shown in Figure 2-2, this area is further divided into three mirror areas.

Mirror area 1 is an area provided for access to WorkFLASH via ECC logic.

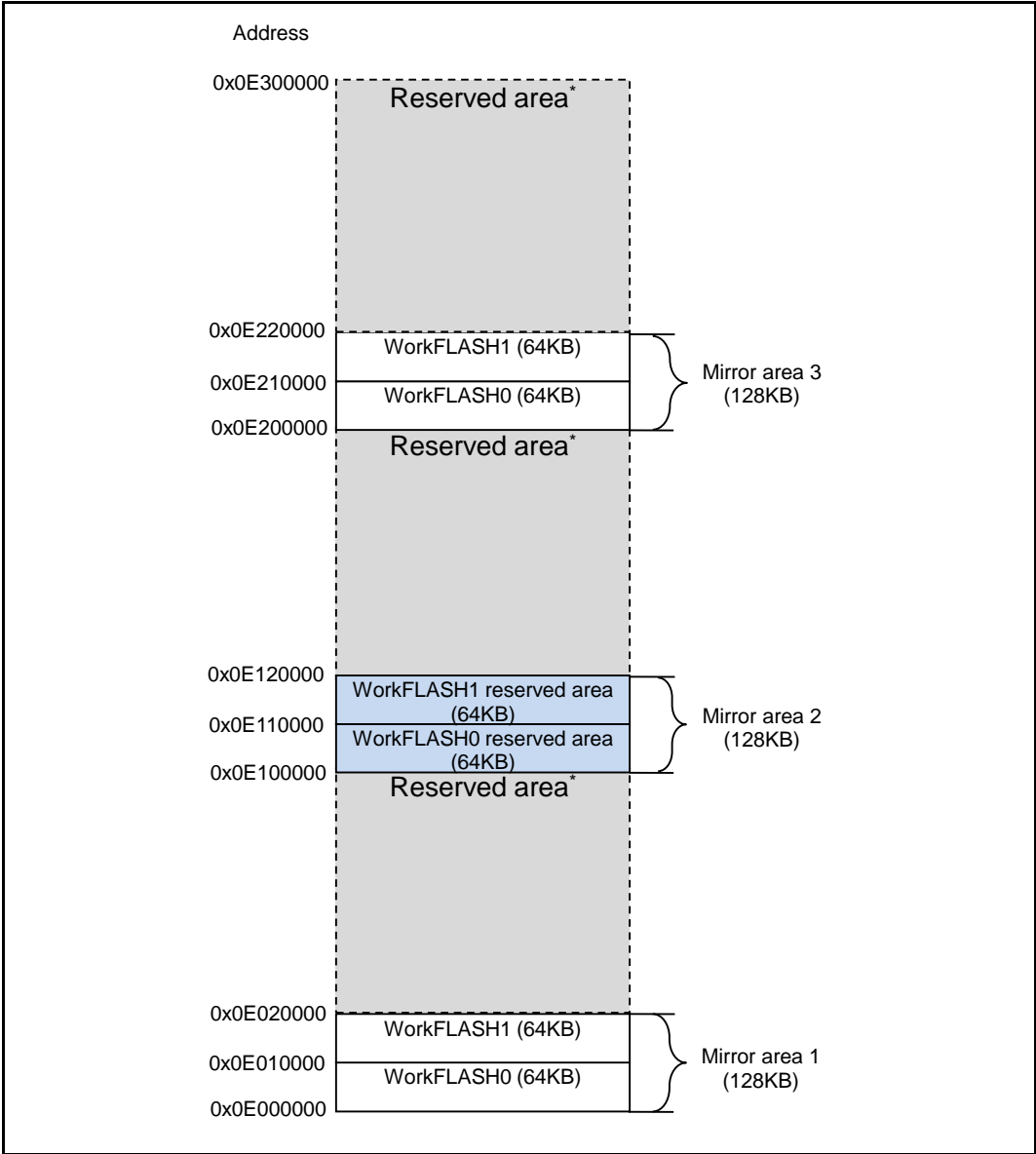
When WorkFLASH is read via mirror area 1 with ECC enabled for the unit to be read, an ECC check is performed. If the check detects a 1-bit error, the error is corrected. If it detects an error of 2 or more bits, a bus error response is made. However, in cases where ECC is disabled, the ECC check, error detection, and correction are not performed. In these cases, even if there is an error, no action such as error correction and bus error response is taken for it. When writing to WorkFLASH via mirror area 1 with ECC enabled for the unit to be written, an ECC is generated and written. However, WorkFLASH can generate an ECC for only 32-bit data, so for any attempt to write 8 bits or 16 bits via mirror area 1 with ECC enabled, WorkFLASH makes a bus error response. In cases where ECC is disabled, no ECC is generated, so 8 bits and 16 bits can also be written via mirror area 1. "xx" for a register name denotes a WorkFLASH unit number. For example, "xx" for WorkFLASH0 is "00".

Mirror area 2 is entirely a reserved area. For this reason, WorkFLASH makes a bus error response for any attempt to read and write or erase data in mirror area 2.

Mirror area 3 is an area provided for access to WorkFLASH by bypassing ECC logic. If WorkFLASH is read via mirror area 3, the ECC check and error detection are not performed. Therefore, even if an error occurs, there is neither the 1-bit error correction nor the bus error response associated with the detection of an error of 2 or more bits. If writing is performed via mirror area 3, no ECC is generated and written regardless of whether ECC is enabled or disabled.

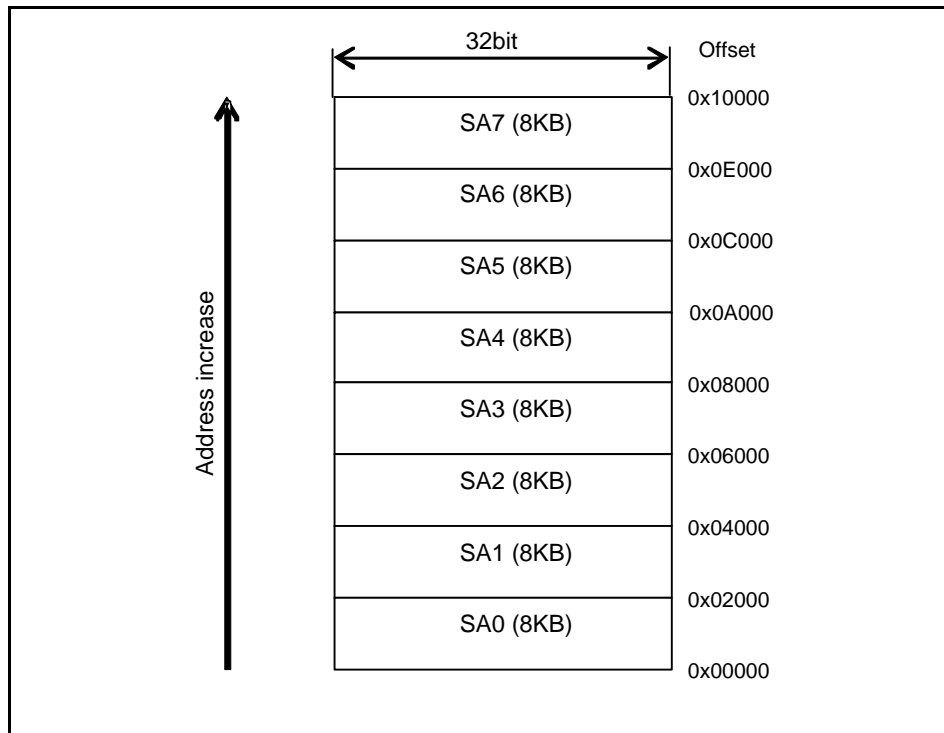
A single WorkFLASH has an area of 64 KB. The 64-KB area consists of 8 small sectors, and the size of each small sector is 8 KB. These sectors are placed as shown in Figure 2-3. Their places in an overall address map is shown in Figure 2-2.

Figure 2-2 Address Map of WorkFLASH in User Mode



\*: Access to a reserved area gives rise to a bus error response but the reserved area access (WFCFG\_BERR:RESA) bit is not detected.

Figure 2-3 Sector Configuration of WorkFLASH in User Mode



### 3. Operation

This section explains the operation of WorkFLASH.

#### 3.1. Read

Reading from WorkFLASH can be done with a size of 8 bits/16 bits/32 bits/64 bits.

However, reading from WorkFLASH cannot be performed while the command sequencer is operating. Before reading, verify that the command sequencer is idle.

If reading is attempted from an existing reserved area inside mirror areas 1 to 3, WorkFLASH makes a bus error response. Thus, any attempt by software to read a reserved area causes a data abort to occur.

If the ECCOFF bit in the WFCFGxx\_ECR register is "0", an ECC check is performed during reading via mirror area 1. If the check finds a 1-bit error, the error is corrected, and the reading itself ends normally. The detection of the 1-bit error can be posted with an interrupt. If the check detects a 2-bit error, WorkFLASH makes a bus error response.

No ECC check is performed for reading via mirror area 3. The value of the ECCOFF bit in this WFCFGxx\_ECR register in such cases has no significance.

#### 3.2. Write and Erase

When data is written to an address that is not in a reserved area of mirror area 1 or mirror area 3, the WorkFLASH hardware automatically sends the programming access sequence of the write command to Flash memory.

If the ECCOFF bit in the WFCFGxx\_ECR register is "0" in the writing of 32 bits via mirror area 1, an ECC is generated and written. If the ECCOFF bit in the WFCFGxx\_ECR register is "0" at the time of an attempt to write 8 bits or 16 bits to mirror area 1, WorkFLASH makes a bus error response.

If the ECCOFF bit in the WFCFGxx\_ECR register is "1" at the time of an attempt to write 8 bits or 16 bits to mirror area 1, WorkFLASH performs the writing but does not make a bus error response. However, an ECC is not generated and written.

For mirror area 3, writing with a size of 8 bits/16 bits/32 bits is possible. However, in writing via mirror area 3, no ECC is generated and written.

An erase operation is executed with the ERS7/6/5/4/3/2/1/0E bits in the WFCFGxx\_SEQCM register specifying the sector to erase, and the OPC[1:0] bits specifying the erase operation.

You cannot suspend and resume operations for macro and sector erase because the command sequence cannot be directly written into flash memory.

The command sequencer can handle one write operation at a time. If another write request is received during execution of the write command sequence, a bus error response is made. While the write command sequence is not being executed, the value of the ST[1:0] bits in the WFCFGxx\_WSR register is "00".





### 3.3. Transfer of Write Data with DMA

WorkFLASH enables writing by DMA transfer.

When the DMAEN bit in the WFCFGxx\_WCR register is "1", a DMA transfer request is generated each time the write command sequencer becomes idle (the ST[1:0] bits in the WFCFGxx\_WSR register are "00").

If an error is detected during DMA transfer, the DMA transfer is stopped and an interrupt is generated.

The command sequencer can use DMA transfer in either block or burst mode. However, for the DMA transfer of data to the command sequencer, use block mode DMA transfer with a block size of 32 bits.

**Note:**

- If DMA transfer stops because of an error, follow the processing flow below.
  1. The detection of an error in WorkFLASH cancels a DMA transfer, but a WorkFLASH DMA transfer request is asserted again.  
(If an error is detected while a write operation is in progress, any posterior attempt to write further data will not be guaranteed.)
  2. A WorkFLASH interrupt request is asserted.
  3. From the CPU, write DMAEN="0" (DMA transfer release) in the WFCFGxx\_WCR register.
  4. After step 3, WorkFLASH DMA transfer requests are disabled (initialized state).  
(To perform a DMA transfer after step 4, set and perform it from the beginning.)

### 3.4. Insert Wait Cycle

- You can insert wait cycles when flash memory is accessed by setting the FAWC bit in the WFCFGxx\_CR register. (For details on the FAWC bit, see Section "5.2. WorkFLASH00/01/02/03 Configuration Register (WFCFGxx\_CR)".
- If the system clock frequency is low, you can even set the FAWC bit to "0".
- The wait cycle count set for the FAWC bit is applied when flash memory is accessed for both reading and programming.
- The value set for the FAWC bit is valid until the setting is changed. The value of the FAWC bit is "3" after reset.
- The setting value of the WFCFGxx\_CR register can be updated (applied) by executing one of the following commands. (It cannot be updated when flash memory is accessed.)
  - Write command
  - Sector erase command
  - Reset command for flash memory
  - Software Reset (WFCFGxx\_CR:SWFRST)

**Note:**

- Technically, the setting value can be applied by executing one of the above 4 commands in WorkFLASH. Operationally, however, the value should be updated by issuing the reset command for flash memory or performing a software reset.

### 3.5. ECC Generation and Check

WorkFLASH adds a 7-bit error check code (ECC: Error Check Code) for every 32 bits, so that it can detect and correct 1-bit errors and detect 2-bit errors.

The ECC logic of WorkFLASH generates an ECC during writing and performs a syndrome check during reading, by the same way 32-bit ECC operations are executed inside the ARM<sup>®</sup> Cortex<sup>™</sup>-R5F core.

During writing, a 7-bit ECC is generated for 32-bit data and written together with the data. For details on how to write data with an ECC added, see Section "4.2. Write Procedure (Example)".

During reading, a syndrome is calculated for the following decision: "no error detected," "1-bit error detected," or "error of 2 or more bits detected." If the result is "error of 2 or more bits detected," the error cannot be corrected. If the Flash memory contents are erased, errors are never detected.

To test the check function, the ECC logic has a function to insert an error into the data and ECC read from Flash memory.

ECC can be turned on/off with the setting of the ECCOFF bit in the WFCFGxx\_ECR register.

#### 3.5.1. ECC Generation

During writing, the 7-bit check bits CB[6:0] are generated from the 32-bit write data D[31:0] according to the calculation formula shown in Table 3-5. The generated check bits are XOR'ed with 0x73 and then written to Flash memory.

Table 3-1 ECC Calculation Formula

bit	Calculation Formula
CB[6]	$D[31] \oplus D[30] \oplus D[29] \oplus D[28] \oplus D[27] \oplus D[26] \oplus D[25] \oplus D[24] \oplus D[23] \oplus D[22] \oplus D[21] \oplus D[20] \oplus D[19] \oplus D[18] \oplus D[17] \oplus D[16]$
CB[5]	$D[31] \oplus D[30] \oplus D[29] \oplus D[28] \oplus D[27] \oplus D[26] \oplus D[25] \oplus D[24] \oplus D[7] \oplus D[6] \oplus D[5] \oplus D[4] \oplus D[3] \oplus D[2] \oplus D[1] \oplus D[0]$
CB[4]	$D[31] \oplus D[30] \oplus D[23] \oplus D[22] \oplus D[21] \oplus D[20] \oplus D[19] \oplus D[18] \oplus D[15] \oplus D[14] \oplus D[13] \oplus D[12] \oplus D[11] \oplus D[10] \oplus D[7] \oplus D[6]$
CB[3]	$\sim (D[29] \oplus D[28] \oplus D[27] \oplus D[23] \oplus D[22] \oplus D[21] \oplus D[17] \oplus D[16] \oplus D[15] \oplus D[14] \oplus D[13] \oplus D[9] \oplus D[8] \oplus D[5] \oplus D[4] \oplus D[3])$
CB[2]	$\sim (D[31] \oplus D[29] \oplus D[26] \oplus D[25] \oplus D[23] \oplus D[20] \oplus D[19] \oplus D[16] \oplus D[15] \oplus D[12] \oplus D[11] \oplus D[8] \oplus D[7] \oplus D[5] \oplus D[2] \oplus D[1])$
CB[1]	$D[28] \oplus D[26] \oplus D[24] \oplus D[22] \oplus D[20] \oplus D[18] \oplus D[17] \oplus D[16] \oplus D[14] \oplus D[12] \oplus D[10] \oplus D[9] \oplus D[8] \oplus D[4] \oplus D[2] \oplus D[0]$
CB[0]	$D[31] \oplus D[29] \oplus D[28] \oplus D[26] \oplus D[23] \oplus D[22] \oplus D[20] \oplus D[16] \oplus D[13] \oplus D[11] \oplus D[10] \oplus D[9] \oplus D[6] \oplus D[3] \oplus D[1] \oplus D[0]$



### 3.5.2. Syndrome Calculation

When read-accessing, the 7-bit check bits CB[6:0] are calculated from the data D[31:0] which is read from flash memory in accordance with the calculation formulas given in Table 3-5. The calculated check bits are used together with the check bits EDOR[6:0] that are read from the flash memory to generate a syndrome S[6:0] in accordance with the calculation formulas given in Table 3-6.

**Table 3-2 Syndrome Calculation Formulas**

bit	Calculation Formula
S[6]	$\sim (CB[6] \wedge EDOR[6])$
S[5]	$\sim (CB[5] \wedge EDOR[5])$
S[4]	$\sim (CB[4] \wedge EDOR[4])$
S[3]	$CB[3] \wedge EDOR[3]$
S[2]	$CB[2] \wedge EDOR[2]$
S[1]	$\sim (CB[1] \wedge EDOR[1])$
S[0]	$\sim (CB[0] \wedge EDOR[0])$

### 3.5.3. Error Detection

One of the following is determined based on the calculated value of the syndrome S[6:0].

- No error is detected.
- 1-bit error detected.
- 2-bit error detected.
- Error of 3 bits or greater detected.

Table 3-3 shows the relationship between syndrome values and decision results. The meanings of the symbols used in the table are as follows.

- "+" : No error is detected.
- "C[n]" (0≤n≤6) : A 1-bit error is detected. The value of the check bit CB[n] is erroneous.
- "D[m]" (0≤m≤31) : A 1-bit error is detected. The value of the data bit D[m] is erroneous.
- "T" : A 2-bit error is detected. It cannot be corrected.
- "M" : An error of 3 or more bits is detected. It cannot be corrected.

The 1-bit errors can be detected in all cases, and can be corrected. The 2-bit errors can be detected in all cases, but they cannot be corrected. The errors of 3 or more bits may be detected in some cases, as shown in Table 3-3. However, they cannot be detected in all cases. Note that errors of 3 or more bits also cannot be corrected.

**Table 3-3 Meanings of Syndrome Values**

		Value in S[6:4]							
		0x0	0x1	0x2	0x3	0x4	0x5	0x6	0x7
Value in S[3:0]	0x0	+	C[4]	C[5]	T	C[6]	T	T	D[30]
	0x1	C[0]	T	T	D[6]	T	M	M	T
	0x2	C[1]	T	T	M	T	D[18]	D[24]	T
	0x3	T	D[10]	D[0]	T	M	T	T	M
	0x4	C[2]	T	T	D[7]	T	D[19]	D[25]	T
	0x5	T	D[11]	D[1]	T	M	T	T	D[31]
	0x6	T	D[12]	D[2]	T	M	T	T	M
	0x7	M	T	T	M	T	D[20]	D[26]	T
	0x8	C[3]	T	T	M	T	D[21]	D[27]	T
	0x9	T	D[13]	D[3]	T	M	T	T	M
	0xA	T	D[14]	D[4]	T	D[17]	T	T	M
	0xB	D[9]	T	T	M	T	D[22]	D[28]	T
	0xC	T	D[15]	D[5]	T	M	T	T	M
	0xD	M	T	T	M	T	D[23]	D[29]	T
	0xE	D[8]	T	T	M	T	M	M	T
	0xF	T	M	M	T	D[16]	T	T	M



### 3.6. Interrupt

WorkFLASH can generate interrupt requests in cases such as the following.

- When one write or erase operation is completed and the next one can begin  
An interrupt request is generated in WorkFLASH when the RDYINT bit in the WFCFGxx\_SR register is "1" and the RDYIE bit in the WFCFGxx\_ICR register is "1".
- When the hang-up state is detected  
The hang-up state resulting from a timeout detected while the command sequencer is operating causes the HANGINT bit in the WFCFGxx\_SR register to change to "1". If the HANGIE bit in the WFCFGxx\_ICR register is "1" at this time, an error interrupt request is generated in WorkFLASH.
- When the ECC logic detects a 1-bit error (correctable)  
An interrupt request is generated in WorkFLASH when the SECINT bit in the WFCFGxx\_SECIR register is "1" and the SECIE bit in the WFCFGxx\_SECIR register is "1".
- When an error is detected during a write  
An error interrupt request is generated in WorkFLASH when the ERRINT bit in the WFCFGxx\_SR register is "1" and the ERRIE bit in the WFCFGxx\_ICR register is "1".

### 3.7. Bus Error Response

WorkFLASH makes a bus error response in cases such as those described below.

- An uncorrectable error (error of 2 or more bits) has been detected with an ECC.
- An attempt was made to write to mirror areas 1 or 3 when the WE bit in the WFCFGxx\_CR register is "0".
- An attempt was made to write in units of 8 or 16 bits via mirror area 1 when ECC is enabled.
- An attempt was made to access a reserved area in WorkFLASH.  
(The same applies to not only the reserved areas in mirrors area 1 to 3 but also the free areas in the register areas.)
- An attempt was made to write in an unprivileged state to a register, except for the following registers:
  - WFCFGxx\_WCR register
  - WFCFGxx\_SEQCM register
  - WFCFGxx\_BERRCLR register
- An attempt to overwrite the WFCFGxx\_ECR register was made after its value is already determined
- An illegal write-access to protected registers was attempted in a manner violating the defined sequence for unlocking write-accesses to registers. Specifically, the following cases apply.
  - An attempt was made to write to a protected register in the locked state.
  - An attempt was made to write to a protected register from the other master in the unlocked state.
  - An attempt was made to write to the WFCFGxx\_CPR register twice in a row.
  - An attempt was made to write an incorrect protection key value to the WFCFGxx\_CPR register.
- An attempt to write to WorkFLASH or execute a command from the WFCFGxx\_SEQCM register was made when the command sequencer was in a state other than the idle state. (Any subsequent writing or command at this time is ignored.)
- A read-only register was write-accessed.

### 3.8. WorkFLASH Access Restrictions

To prevent the data and programs in WorkFLASH from being read by third parties, operations on WorkFLASH are restricted. (See Table 3-4.)

**Table 3-4 WorkFLASH Access Restrictions**

Product Mode		User Mode		Non-user Mode	
Security setting		ON	OFF	ON	OFF
Operation on flash memory	Macro erase	No	No	Yes <sup>*2</sup>	No
	Sector erase	Yes	Yes	No effect on the operation	Yes
	Write	Yes	Yes	No effect on the operation	Yes
	Read	Yes	Yes	No <sup>*1</sup>	Yes

\*1: "0xFFFFFFFF" is always read.

\*2: The macro erase execution procedure is not disclosed because it is a non-user mode function.



## 4. Setting Procedure

This section explains the WorkFLASH setting procedures.

### 4.1. Setting the Number of Wait Cycles

If the operating frequency of the system (CLK\_HPM\_PD2) is higher than the maximum operating frequency of the flash memory, it is necessary to insert wait cycles when accessing the flash memory, by setting an appropriate value in FAWC[1:0] in the WFCFGxx\_CR register.

The wait settings must not be changed while WorkFLASH is being accessed (during read-access or program execution).

(For details on when the register settings are reflected, see Section "3.4. Insert Wait Cycle".)

You can calculate the value to set for FAWC[1:0] according to the following formula.

$$\text{FAWC}[1:0] = \text{CEILING}(\text{System operating frequency} / \text{Maximum operating frequency of flash memory}) - 1$$

CEILING() in the above formula represents a function that rounds up the decimal part of the argument to make it an integer.

The maximum operating frequency of flash memory built in this product is 80 MHz. For example, if the system operating frequency is 200 MHz, for example, the setting value for FAWC[1:0] should be "2" as calculated using the above formula.

### 4.2. Write Procedure (Example)

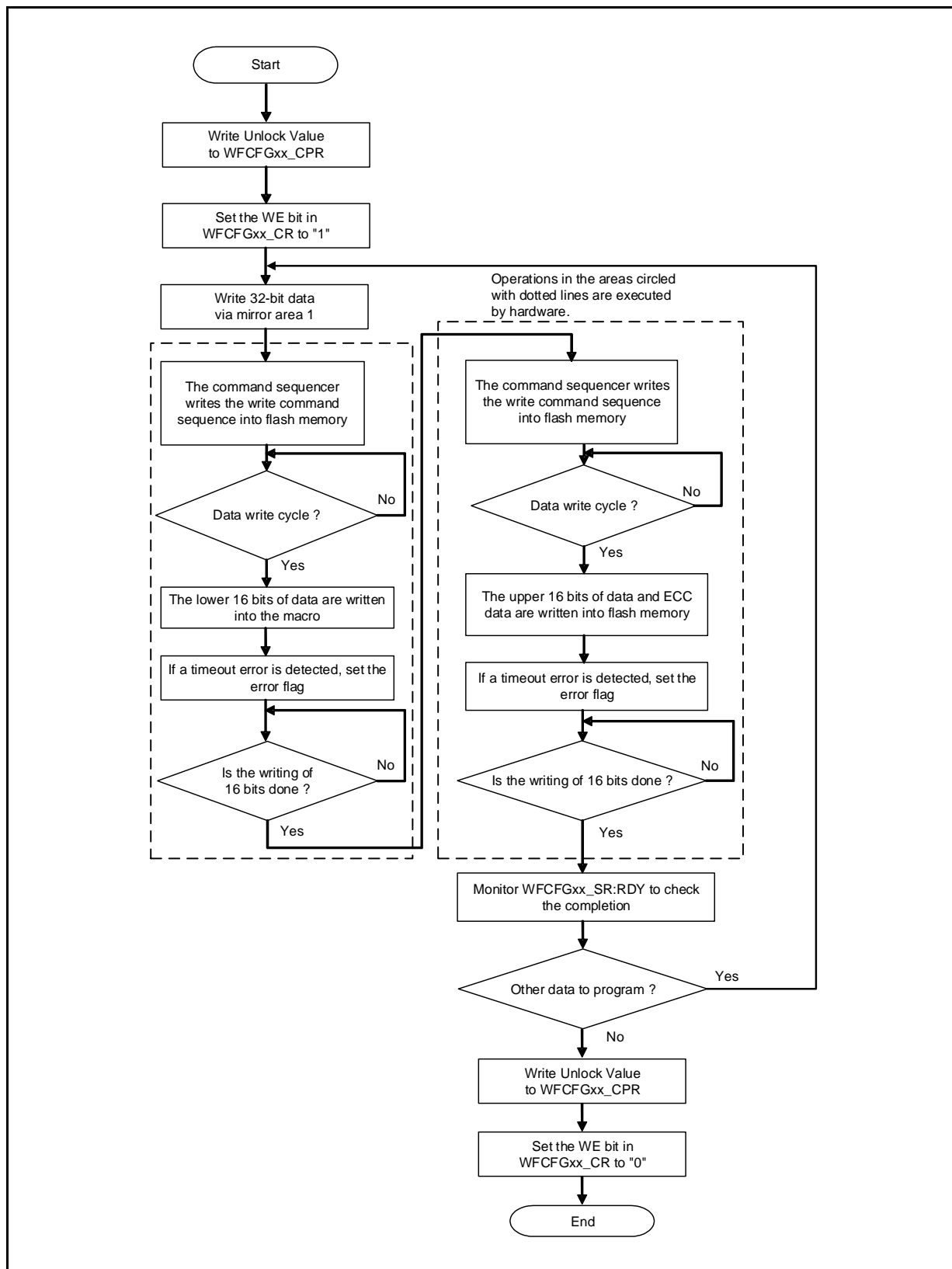
Figure 4-1 shows the procedure for 32-bit data writing with an ECC attached. To execute writing with an ECC attached in the procedure shown in that figure, the ECCOFF bit in the WFCFGxx\_ECR register must be "0" (default value).

To write only data without attaching an ECC, write the data via mirror area 3 instead of mirror area 1. If you write via mirror area 3, the upper 16-bit data is written but the ECC data is not written as described in the right half of the figure.

To write either 8 or 16 bits, similarly write the data via mirror area 3. If the address of the 8- or 16-bit data to write is within the range of the lower 16 bits of the 32 bits (if bit2 of the address is "0"), the hardware executes only the left sequence of the figure to write the data. If the address is within the range of the upper 16 bits (if bit2 of the address is "1"), the hardware executes only the left sequence of the figure except the part for writing the ECC data.

The RDY flag in the WFCFGxx\_SR register is polled to wait for the completion of writing, as shown in the figure. Another method of detecting the completion is to use a RDY interrupt.

Figure 4-1 Procedure for 32-bit Data Writing with ECC Attached (Example)







### 4.3. Sector Erase Procedure

Writing the necessary information in the WFCFGxx\_SEQCM register makes it possible to erase in units of sectors. The command sequencer starts the sector erase operation on the specified sector as a result of the following. "1" has been set between the ERS7E and ERS0E bits, at the bit location corresponding to the sector to erase. Also, data with the OPC[1:0] bits set to 0b10 (code representing erase) has been written to the WFCFGxx\_SEQCM register.

To wait for the completion of a sector erase operation, the same method as that used for the write can be used.

## 5. Registers

This section explains the registers in WorkFLASH.

The register areas in WorkFLASH are located in the peripheral functions area, each of which has a size of 1KB.

Table 5-1 shows the placement of registers in the register areas. In register names in the table, "xx" preceded by "WFCFG" represents a 2-digit WorkFLASH unit number. For example, the register name for WorkFLASH0 would include "WFCFG00\_" having "00".

**Table 5-1 List of WorkFLASH Registers**

Abbreviated Register Name	Register Name	See
WFCFGxx_CPR	WorkFLASH Configuration Protection Key Register	5.1
WFCFGxx_CR	WorkFLASH Configuration Register	5.2
WFCFGxx_ECR	WorkFLASH ECC Control Register	5.3
WFCFGxx_WCR	WorkFLASH Write Command Sequencer Configuration Register	5.4
WFCFGxx_WSR	WorkFLASH Write Command Sequencer Status Register	5.5
WFCFGxx_DBEIR	WorkFLASH Data Bit Error Injection Register	5.6
WFCFGxx_EEIR	WorkFLASH ECC Bit Error Injection Register	5.7
WFCFGxx_ICR	WorkFLASH Interrupt Control Register	5.8
WFCFGxx_SR	WorkFLASH Status Register	5.9
WFCFGxx_SECIR	WorkFLASH SEC Interrupt Register	5.10
WFCFGxx_EEAR	WorkFLASH ECC Error Address Register	5.11
WFCFGxx_MIR	WorkFLASH Module Identification Register	5.12
WFCFGxx_SEQCM	WorkFLASH Sequencer Command Register	5.13
WFCFGxx_BERR	WorkFLASH Bus Error Response Factor Register	5.14
WFCFGxx_BERRCL	WorkFLASH Bus Error Response Factor Clear Register	5.15
WFCFGxx_UCESR	WorkFLASH Uncorrectable Error Status Register	5.16
WFCFGxx_UCEAR	WorkFLASH Uncorrectable Error Address Register	5.17



## 5.1. WorkFLASH00/01/02/03 Configuration Protection Key Register (WFCFGxx\_CPR)

The WorkFLASH00/01/02/03 configuration protection key register (WFCFGxx\_CPR) is used to protect the following registers from unintended writing:

- WorkFLASH00/01/02/03 configuration register (WFCFGxx\_CR)
- WorkFLASH00/01/02/03 ECC control register (WFCFGxx\_ECR)
- WorkFLASH00/01/02/03 data bit error injection register (WFCFGxx\_DBEIR)
- WorkFLASH00/01/02/03 ECC bit error injection register (WFCFGxx\_EEIR)

Before writing to these registers, write the correct configuration protection key value (0xCF6DF1A5) to the WorkFLASH configuration protection key register of the same unit to set the unlock state. When unlocked, writing (setting change) is enabled.

After being unlocked, writing to the above registers is locked again by writing to the addresses in the same group (Memory & Config Group) area regardless of the master. However, it is not locked, by writing, to the same group area (except the WorkFLASH register area) from another master.

For any attempted writing to the above registers using an invalid procedure, WorkFLASH makes a bus error response. For example, suppose the configuration protection key value is incorrect, or an attempt is made to write to the above registers without unlocking writing. For access with a CPU, a data abort exception is generated because of the bus error response.

Be sure to write 32 bits to the WorkFLASH configuration protection key register.

BIT_OFFSET	31-0
BIT_NAME	CPR
ACCESS_TYPE	R,W
PROT_TYPE	WP
INITIAL_VALUE	00000000_00000000_00000000_00000000

### [bit31:0] CPR[31:0]: Configuration protection key

Writing to the following protected registers is unlocked when the correct configuration protection key value (0xCF6DF1A5) is written to this register so that new values can be set:

- WorkFLASH00/01/02/0 configuration register (WFCFGxx\_CR)
- WorkFLASH00/01/02/03 ECC Control Register (WFCFGxx\_ECR)
- WorkFLASH00/01/02/03 data bit error injection register (WFCFGxx\_DBEIR)
- WorkFLASH00/01/02/03 ECC bit error injection register (WFCFGxx\_EEIR)

When writing to these registers is unlocked, 0xFFFFFFFF is read from this register. When writing to these registers is locked, 0x00000000 is read from this register.

## 5.2. WorkFLASH00/01/02/03 Configuration Register (WFCFGxx\_CR)

The WorkFLASH configuration register has the following functions for controlling the Flash memory in WorkFLASH:

- Specifies the wait cycle count when flash memory is accessed
- Enabling writing to the Flash memory
- Resetting the Flash memory

The setting of this register can be changed only when the correct configuration protection key value is written to the WorkFLASH configuration protection key register in the same unit and writing to this register is unlocked.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved							SWFRST
ACCESS_TYPE	R0,WX							R0,W
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved							WE
ACCESS_TYPE	R0,WX							R/W
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved						FAWC	
ACCESS_TYPE	R0,WX						R/W	
PROT_TYPE	WPS							
INITIAL_VALUE	000000						11	

**[bit31:17] Reserved: Reserved bits**

**[bit16] SWFRST: Software reset**

Value	Description
0	There is no effect on operation.
1	Reset the Flash memory and the command sequencer.

**[bit15:9] Reserved: Reserved bits**

**[bit8] WE: Program enable**

This bit enables/disables Flash memory writing/erasing. The detection of any writing or erasing of the Flash memory when this bit is "0" causes a bus error response by WorkFLASH.

Value	Description
0	Disable writing and erasing.
1	Enable writing and erasing.

**[bit7:2] Reserved: Reserved bits****[bit1:0] FAWC[1:0]: Flash wait control**

These bits set the wait cycle count when flash memory is accessed. Set the appropriate value according to the operating frequency of the flash memory and flash memory access time.

Value	Description
00	Do not insert wait cycle.
01	Insert 1 wait cycle.
10	Insert 2 wait cycles.
11	Insert 3 wait cycles.

### 5.3. WorkFLASH00/01/02/03 ECC Control Register (WFCFGxx\_ECR)

This register is used to control the operation of the ECC logic. The setting of this register can be changed only when the correct configuration protection key value is written to the WorkFLASH configuration protection key register in the same unit and writing to this register is unlocked. You can change the setting value of this register only once. For any second or subsequent writing to this register, WorkFLASH makes a bus error response (ECRWL).

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	WPS
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved							ECCOFF
ACCESS_TYPE	R0,WX							R/W
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							0

**[bit31:1] Reserved: Reserved bits**

**[bit0] ECCOFF: ECC off**

This bit enables or disables ECC generation and check by access via mirror area 1. If ECC generation and check are disabled, no ECC is generated and no ECC check is performed even during access via mirror area 1.

Value	Description
0	Generate an ECC and perform an ECC check by access via mirror area 1.
1	Neither generate an ECC nor perform an ECC check by access via mirror area 1.



## 5.4. WorkFLASH00/01/02/03 Write Command Sequencer Configuration Register (WFCFGxx\_WCR)

This register can set whether to operate the write using DMA transfer.

BIT_OFFSET	31-16
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved							Reserved
ACCESS_TYPE	R0,WX							R/W1
PROT_TYPE	-							
INITIAL_VALUE	0000000							1

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved							DMAEN
ACCESS_TYPE	R0,WX							R/W
PROT_TYPE	-							
INITIAL_VALUE	0000000							0

**[bit31:1] Reserved: Reserved bits**

### **[bit0] DMAEN: DMA enable**

When this bit is "1" and the ST[1:0] bits in the WorkFLASH write command sequencer status register is "00", WorkFLASH generates a DMA transfer request and makes a request to the DMAC for the data to be written next.

Value	Description
0	Do not generate a DMA transfer request.
1	Generate a DMA transfer request.

## 5.5. WorkFLASH00/01/02/03 Write Command Sequencer Status Register (WFCFGxx\_WSR)

Reading this register can provide information on the status of the write command sequencer.

This register is read-only. Writing to this register returns a bus error.

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved						ST	
ACCESS_TYPE	R0,WX						R,WX	
PROT_TYPE	-							
INITIAL_VALUE	000000						00	

**[bit31:2] Reserved: Reserved bits**

**[bit1:0] ST[1:0]: Write command sequencer status**

These bits represent the status of the write command sequencer. Do not access Flash memory when the write command sequencer is in a state other than the idle state.

Value	Description
00	Indicates that the write command sequencer is in the idle state.
01	Indicates that the write command sequencer is sending a command sequence to Flash memory.
10	Indicates that the write command sequencer is waiting for the completion of a Flash memory operation.
11	Reserved. This state never occurs.





## 5.6. WorkFLASH00/01/02/03 Data Bit Error Injection Register (WFCFGxx\_DBEIR)

This register is used to perform an ECC logic operation test by injecting errors into the data bits read from Flash memory.

The setting of this register can be changed only when the correct configuration protection key value is written to the WorkFLASH configuration protection key register in the same unit and writing to this register is unlocked.

BIT_OFFSET	31-0
BIT_NAME	DBEIR
ACCESS_TYPE	R/W
PROT_TYPE	WPS
INITIAL_VALUE	00000000_00000000_00000000_00000000

### [bit31:0] DBEIR[31:0]: Data bit error injection location

These bits send a calculated value to the ECC check logic. The value is the exclusive OR of the value in this register and the value of the data bits read from Flash memory.

bit [i] (31≥i≥0)	Description
0	Send the bits at the same location as the data read from flash memory, to the ECC check logic as is.
1	Invert the bits at the same location as the data read from flash memory, before sending them to the ECC check logic.

## 5.7. WorkFLASH00/01/02/03 ECC Bit Error Injection Register (WFCFGxx\_EEIR)

This register is used to perform an ECC logic operation test (dummy normal test) by injecting errors into the ECC bits read from Flash memory.

The setting of this register can be changed only when the correct configuration protection key value is written to the WorkFLASH configuration protection key register in the same unit and writing to this register is unlocked.

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	WPS
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved	EEIR						
ACCESS_TYPE	R0,WX	R/W						
PROT_TYPE	WPS							
INITIAL_VALUE	0	0000000						

**[bit31:7] Reserved: Reserved bits**

### **[bit6:0] EEIR[6:0]: ECC bit error injection location**

These bits send a calculated value to the ECC check logic. The value is the exclusive OR of the value in this register and the value of the ECC data read from Flash memory.

bit [i] (6≥i≥0)	Description
0	Send the EEIR bits at the same location as the ECC data read from Flash memory, to the ECC check logic as they are.
1	Invert the EEIR bits at the same location as the ECC data read from Flash memory, before sending them to the ECC check logic.



## 5.8. WorkFLASH00/01/02/03 Interrupt Control Register (WFCFGxx\_ICR)

This register can enable interrupt request generation for each factor and clear an interrupt factor.

BIT_OFFSET	31-16
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	WP
INITIAL_VALUE	00000000_00000000

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved						HANGIC	RDYIC
ACCESS_TYPE	R0,WX						R0,W	R0,W
PROT_TYPE	WP							
INITIAL_VALUE	000000						0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved					Reserved	HANGIE	RDYIE
ACCESS_TYPE	R0,WX					R/W0	R/W	R/W
PROT_TYPE	WP							
INITIAL_VALUE	00000					0	0	0

[bit31:10] Reserved: Reserved bits

[bit9] HANGIC: Hang interrupt clear

Value	Description
0	Writing "0" to this bit has no meaning.
1	Writing "1" to this bit clears the HANGINT bit in the WFCFGxx_SR register of the same unit.

[bit8] RDYIC: Ready interrupt clear

Value	Description
0	Writing "0" to this bit has no meaning.
1	Writing "1" to this bit clears the RDYINT bit in the WFCFGxx_SR register of the same unit.

[bit7:2] Reserved: Reserved bits

[bit1] HANGIE: Hang interrupt enable

Value	Description
0	Disable the generation of hang interrupt requests.
1	Enable the generation of hang interrupt requests.

[bit0] RDYIE: Ready interrupt enable

Value	Description
0	Disable generation of ready interrupt requests.
1	Enable generation of ready interrupt requests.

## 5.9. WorkFLASH00/01/02/03 Status Register (WFCFGxx\_SR)

This register is a read-only register that retains the state of Flash memory and individual interrupt factors. Writing to this register returns a bus error.

BIT_OFFSET	31-16
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved						HANGINT	RDYINT
ACCESS_TYPE	R0,WX						R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0000000						0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved							RDY
ACCESS_TYPE	R0,WX							R,WX
PROT_TYPE	-							
INITIAL_VALUE	0000000							0

### [bit31:10] Reserved: Reserved bits

### [bit9] HANGINT: Hang-up interrupt

This bit indicates that a hang-up interrupt request was generated because a transition of Flash memory to the hang-up 1 state was detected.

Flash memory changes to the hang-up 1 state in the following cases.

- An attempt is made to write "1" to a cell whose value is "0".
- Writing or erasing is not completed within the prescribed period.

This bit is cleared when "1" is written to the HANGIC bit in the WFCFGxx\_ICR register of the same unit.

Value	Description
0	No hang-up interrupt request has been generated.
1	A hang-up interrupt request has been generated.

### [bit8] RDYINT: Ready interrupt

This bit indicates that a ready interrupt request was generated because a transition of Flash memory to the ready state was detected. This bit is cleared when "1" is written to the RDYIC bit in the WFCFGxx\_ICR register of the same unit.

Value	Description
0	No ready interrupt request has been generated.
1	A ready interrupt request has been generated.



**[bit7:1] Reserved: Reserved bits**

**[bit0] RDY: Ready**

This bit indicates whether Flash memory is in the ready state. In the ready state, Flash memory can start the execution of a new operation.

Value	Description
0	Flash memory is operating. In this state, it can receive only the read/reset command.
1	Flash memory can start the execution of the next command.

## 5.10. WorkFLASH00/01/02/03 SEC Interrupt Register (WFCFGxx\_SECIR)

This register contains the status flags, enable bits, and clear bits related to 1-bit error correction interrupts.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved	SYN						
ACCESS_TYPE	R0,WX	R,WX						
PROT_TYPE	WP							
INITIAL_VALUE	0	0000000						

BITS	23	22	21	20	19	18	17	16
BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved							SECINT
ACCESS_TYPE	R0,WX							R,WX
PROT_TYPE	WP							
INITIAL_VALUE	0000000							0

BITS	15	14	13	12	11	10	9	8
BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved							SECIC
ACCESS_TYPE	R0,WX							R0,W
PROT_TYPE	WP							
INITIAL_VALUE	0000000							0

BITS	7	6	5	4	3	2	1	0
BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved							SECIE
ACCESS_TYPE	R0,WX							R/W
PROT_TYPE	WP							
INITIAL_VALUE	0000000							0

**[bit31] Reserved: Reserved bit**

**[bit30:24] SYN[6:0]: Syndrome**

These bits store the syndrome from the detection of a 1-bit error.

**[bit23:17] Reserved: Reserved bits**

**[bit16] SECINT: 1-bit error correction interrupt**

This bit indicates whether there is a 1-bit error correction interrupt request. If the ECC check during reading has detected and corrected a 1-bit error, a 1-bit error correction interrupt request is generated.

This bit is read-only. Writing to this bit has no meaning. This bit is cleared when "1" is written to the SECIC bit in the same register.

Value	Description
0	A 1-bit error correction interrupt request has not been generated.
1	A 1-bit error correction interrupt request has been generated.



[bit15:9] Reserved: Reserved bits

[bit8] SECIC: 1-bit error correction interrupt clear

Value	Description
0	Writing "0" to this bit has no meaning.
1	Writing "1" to this bit clears the SECINT bit in the same register.

[bit7:1] Reserved: Reserved bits

[bit0] SECIE: 1-bit error correction interrupt enable

Value	Description
0	Disable generation of 1-bit error correction interrupt request.
1	Enable generation of 1-bit error correction interrupt request.

### 5.11.      **WorkFLASH00/01/02/03 ECC Error Address Register (WFCFGxx\_EEAR)**

This register retains the address at which a 1-bit error was detected during reading. If the 1-bit error was detected multiple times, the register retains the address at which the error was last detected.

This register is read-only. Writing to this register returns a bus error.

BIT_OFFSET	31-0
BIT_NAME	EEAR
ACCESS_TYPE	R,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000_00000000

#### **[bit31:0] EEAR[31:0]: Error address**

These bits have the address at which a 1-bit error was detected by the ECC check during reading. If the error was detected multiple times, they retain the address at which the error was last detected.





## 5.12. WorkFLASH00/01/02/03 Module Identification Register (WFCFGxx\_MIR)

This register reads numerical values, including the WorkFLASH unit ID, version, and patch level.

This register is read-only. Writing to this register returns a bus error.

BIT_OFFSET	31-0
BIT_NAME	MID
ACCESS_TYPE	R,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000_00000000

### [bit31:0] MID[31:0]: Module ID

These bits are numerical values, including the WorkFLASH unit ID, version, and patch level.

### 5.13. WorkFLASH00/01/02/03 Sequencer Command Register (WFCFGxx\_SEQCM)

This register is used to specify the execution of the read/reset command or the sector erase command for the command sequencer. If the execution of the command is completed with "1" set in the RDY bit in the WFCFGxx\_SR register of the same unit, this register is cleared.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	ERS7E	ERS6E	ERS5E	ERS4E	ERS3E	ERS2E	ERS1E	ERS0E
ACCESS_TYPE	R,W	R,W	R,W	R,W	R,W	R,W	R,W	R,W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved						OPC	
ACCESS_TYPE	R0,WX						R,W	
PROT_TYPE	-							
INITIAL_VALUE	000000						00	

[bit31:24] Reserved: Reserved bits

[bit23] ERS7E: Sector 7 erase enable

Value	Description
0	Sector 7 erase enable is invalid.
1	Sector 7 erase enable is valid. (This sector is to be erased.)

[bit22] ERS6E: Sector 6 erase enable

Value	Description
0	Sector 6 erase enable is invalid.
1	Sector 6 erase enable is valid. (This sector is to be erased.)

**[bit21] ERS5E: Sector 5 erase enable**

Value	Description
0	Sector 5 erase enable is invalid.
1	Sector 5 erase enable is valid. (This sector is to be erased.)

**[bit20] ERS4E: Sector 4 erase enable**

Value	Description
0	Sector 4 erase enable is invalid.
1	Sector 4 erase enable is valid. (This sector is to be erased.)

**[bit19] ERS3E: Sector 3 erase enable**

Value	Description
0	Sector 3 erase enable is invalid.
1	Sector 3 erase enable is valid. (This sector is to be erased.)

**[bit18] ERS2E: Sector 2 erase enable**

Value	Description
0	Sector 2 erase enable is invalid.
1	Sector 2 erase enable is valid. (This sector is to be erased.)

**[bit17] ERS1E: Sector 1 erase enable**

Value	Description
0	Sector 1 erase enable is invalid.
1	Sector 1 erase enable is valid. (This sector is to be erased.)

**[bit16] ERS0E: Sector 0 erase enable**

Value	Description
0	Sector 0 erase enable is invalid.
1	Sector 0 erase enable is valid. (This sector is to be erased.)

**[bit15:2] Reserved: Reserved bits**

**[bit1:0] OPC[1:0]: Command**

Value	Description
00	No operation
01	Execute the read/reset command.
10	Execute the sector erase/macro erase command. ERS7E to ERS0E specify the sector erase target.
11	Reserved. Must not be set.

## 5.14. WorkFLASH00/01/02/03 Bus Error Response Factor Register (WFCFGxx\_BERR)

This register is a read-only register that retains the flags representing the factors of the bus error responses by WorkFLASH. Each flag can be cleared by the writing of "1" to the corresponding bit in the WFCFGxx\_BERRCLR register of the same unit.

This register is read-only. Writing to this register returns a bus error.

BIT_OFFSET	31-16
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved						WTTM	ACCIGN
ACCESS_TYPE	R0,WX						R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	000000						0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	ECRWL	UNACC	RESA	Reserved		SIZE	CRWE	DED
ACCESS_TYPE	R,WX	R,WX	R,WX	R0,WX		R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	00		0	0	0

**[bit31:10] Reserved: Reserved bits**

**[bit9] WTTM: Writing to the corresponding reserved area in mirror area 2**

Value	Description
0	Writing to the corresponding reserved area in mirror area 2 has not been detected.
1	This indicates that a bus error response was made because writing to the corresponding area in mirror area 2 was detected.

**[bit8] ACCIGN: Command overrun**

Value	Description
0	No command overrun has been detected.
1	This indicates that a new command was written to Flash memory during execution of the preceding command, and the subsequent command was ignored. At this time, WorkFLASH made a bus error response to the bus master that wrote the subsequent command.

**[bit7] ECRWL: Protection sequence violation**

Value	Description
0	Writing to the protected register in violation of the protection sequence has not been detected.
1	WorkFLASH has detected a violation of the protection sequence and made a bus error response, in the following cases: <ul style="list-style-type: none"> <li>- Writing twice in a row to the protection key register WFCFGxx_CPR</li> <li>- Incorrect key value written to the protection key register</li> <li>- Attempt to write to a protected register in the locked state</li> <li>- Attempt to write to the ECCOFF bit in the WFCFGxx_ECR register a second or subsequent time</li> </ul>

**[bit6] UNACC: Unprivileged writing**

Value	Description
0	No unprivileged writing has been detected.
1	This indicates that a bus error response was made because writing to a register area in the unprivileged state was detected.

**[bit5] RESA: Reserved area access**

Value	Description
0	Access to a reserved area has not been detected.
1	This indicates that a bus error response was made because access to a reserved area in Flash memory (mirror area 2) or the register area was detected. However, this bit does not indicate whether writing to mirror area 2 was detected.

**[bit4:3] Reserved: Reserved bits**
**[bit2] SIZE: Access size violation**

Value	Description
0	Access with an unsupported size has not been detected.
1	This indicates that access with an invalid size to Flash memory has been detected, followed by a bus error response.

**[bit1] CRWE: Writing prohibition violation**

Value	Description
0	No writing prohibition violation has been detected.
1	This indicates that a bus error response was made because of an attempt to write to Flash memory when the WE bit in the WFCFGxx_CR register was "0".

**[bit0] DED: Uncorrectable error detection**

Value	Description
0	No uncorrectable error has been detected.
1	This indicates that an uncorrectable error was detected by the ECC check during reading, followed by a bus error response.

**Notes:**

- *Reading this register specifies the factor of a bus error response.*
- *If the factor flags in this register are all "0" when a bus error response is generated, the following access is the factor of the bus error response.*
- *Write access to a read-only register*



### 5.15. WorkFLASH00/01/02/03 Bus Error Response Factor Clear Register (WFCFGxx\_BERRCLR)

This register can clear each flag representing the factor of the bus error response by WorkFLASH. To clear each flag, write "1" to the bit corresponding to the flag bit in the WFCFGxx\_BERR register of the same unit.

BIT_OFFSET	31-16
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved						WTTMCLR	ACCIGNCLR
ACCESS_TYPE	R0,WX						R0,W	R0,W
PROT_TYPE	-							
INITIAL_VALUE	000000						0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	ECRWLCLR	UNACCLR	RESACLR	Reserved		SIZECLR	CRWECLR	DEDCLR
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,WX		R0,W	R0,W	R0,W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	00		0	0	0

[bit31:10] Reserved: Reserved bits

[bit9] WTTMCLR: WTTM clear

Value	Description
0	Writing "0" to this bit has no meaning.
1	Writing "1" to this bit clears the WTTM bit in the WFCFGxx_BERR register of the same unit.

[bit8] ACCIGNCLR: ACCIGN clear

Value	Description
0	Writing "0" to this bit has no meaning.
1	Writing "1" to this bit clears the ACCIGN bit in the WFCFGxx_BERR register of the same unit.

[bit7] ECRWLCLR: ECRWL clear

Value	Description
0	Writing "0" to this bit has no meaning.
1	Writing "1" to this bit clears the ECRWL bit in the WFCFGxx_BERR register of the same unit.

[bit6] UNACCLR: UNACC clear

Value	Description
0	Writing "0" to this bit has no meaning.
1	Writing "1" to this bit clears the UNACC bit in the WFCFGxx_BERR register of the same unit.

**[bit5] RESACLR: RESA clear**

Value	Description
0	Writing "0" to this bit has no meaning.
1	Writing "1" to this bit clears the RESA bit in the WFCFGxx_BERR register of the same unit.

**[bit4:3] Reserved: Reserved bits**

**[bit2] SIZECLR: SIZE clear**

Value	Description
0	Writing "0" to this bit has no meaning.
1	Writing "1" to this bit clears the SIZE bit in the WFCFGxx_BERR register of the same unit.

**[bit1] CRWECLR: CRWE clear**

Value	Description
0	Writing "0" to this bit has no meaning.
1	Writing "1" to this bit clears the CRWE bit in the WFCFGxx_BERR register of the same unit.

**[bit0] DEDCLR: DED clear**

Value	Description
0	Writing "0" to this bit has no meaning.
1	Writing "1" to this bit clears the DED bit in the WFCFGxx_BERR register of the same unit.





## 5.16. WorkFLASH00/01/02/03 Uncorrectable Error Status Register (WFCFGxx\_UCESR)

This register contains fields representing information about uncorrectable error detection interrupts.

This register is read-only. Writing to this register returns a bus error.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved	SYN						
ACCESS_TYPE	R0,WX	R,WX						
PROT_TYPE	-							
INITIAL_VALUE	0	0000000						

BIT_OFFSET	23-0
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000

**[bit31] Reserved: Reserved bit**

**[bit30:24] SYN[6:0]: Syndrome**

These bits store the syndrome from the detection of a 2-bit error.

**[bit23:0] Reserved: Reserved bits**

### 5.17.      **WorkFLASH00/01/02/03 Uncorrectable Error Address Register (WFCFGxx\_UCEAR)**

This register retains the address at which an uncorrectable error was detected during reading. If the uncorrectable error was detected multiple times, the register retains the address at which the error was last detected.

This register is read-only. Writing to this register returns a bus error.

BIT_OFFSET	31-0
BIT_NAME	UCEA
ACCESS_TYPE	R,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000_00000000

#### **[bit31:0] UCEA[31:0]: Uncorrectable error address**

These bits have the address at which an uncorrectable error was detected by the ECC check during reading. If the error was detected multiple times, they retain the address at which the error was last detected.



## 6. Others

This section provides notes on using WorkFLASH.

### (1) Handling the Values Read from Reserved Bits

"0" is read from the reserved bits in the registers in WorkFLASH. From the viewpoint of ensuring compatibility with future products and software, make sure that in programming, no significance is attached to any value that is read from a reserved bit.

### (2) Reset during Execution of Writing or Erasing

If this product is reset during execution of writing, the contents of the address targeted for writing are shown as undefined. Alternatively, if it is reset partway through a sector erase, the contents of the sector targeted for erasing are shown as undefined. In such cases, after the completion of the reset, retry suspended writing or erasing.

### (3) Register Settings after Reset

If you perform a reset on WorkFLASH from software, WorkFLASH asserts a reset for at least 1000ns. After the reset, the software must wait for the RDY bit in the WFCFGxx\_SR register of the unit to be configured to become "1" before reading from the flash memory and writing a command.

### (4) Reading from Flash Memory

Data cannot be read from WorkFLASH while writing or erasing of WorkFLASH is being executed. For this reason, copy the required data and programs from WorkFLASH to RAM before starting the write or erase operation, so that they do not need to be read partway through writing or erasing.

### (5) Waiting for the Completion of a Reset from Software

Flash memory is reset by the writing of "1" to the SWFRST bit in the WFCFGxx\_CR register. After performing a reset on a unit, monitor the RDY bit in the WFCFGxx\_SR register of the unit, and wait for the completion of the reset before accessing it.



## CHAPTER 19: Hardware Watchdog Timer

This chapter describes the hardware watchdog timer.

---

1. Overview
2. Configuration
3. Operation
4. Setting Procedure Example
5. Operation Examples
6. Registers
7. Precautions for Using



## 1. Overview

This section provides an overview of the hardware watchdog timer.

The hardware watchdog timer is positioned in the MCU configuration group, used for detecting a runaway state caused by a user program, also the hardware watchdog timer assigned to main system control CPU0, monitors control of the entire system performed by CPU0.

### Hardware Watchdog Timer Features

- Detecting a runaway state caused by a user program generates a reset request. (\*a)
- Clearing the hard reset starts the hardware watchdog timer. (\*a)
- The watchdog counter source clock is selected from 2 types. (\*a)
- (Fast-CR clock (default) or slow-CR clock)
- The BootROM program is used to set registers. (\*a)
- It operates as a 32-bit watchdog counter.
- Different initializations between soft reset and hard reset can be applied.
- Soft reset clears only the watchdog counter.
- Hard reset initializes the watchdog counter, register setting, and internal circuits.
- It monitors the watchdog counter clear protection trigger sequence.
- The window watchdog function is implemented.
- Different operation settings are possible for each device state (RUN and PSS).
- It monitors the watchdog register write protection sequence.
- The majority circuits for bits that affect important functions are implemented.
- It can generate the watchdog reset request or watchdog interrupt request (NMI) in response to a watchdog error.
- It can generate a prior warning interrupt request before the reset request or interrupt request (NMI).
- The watchdog counter can be stopped in the debugging state of the processor.

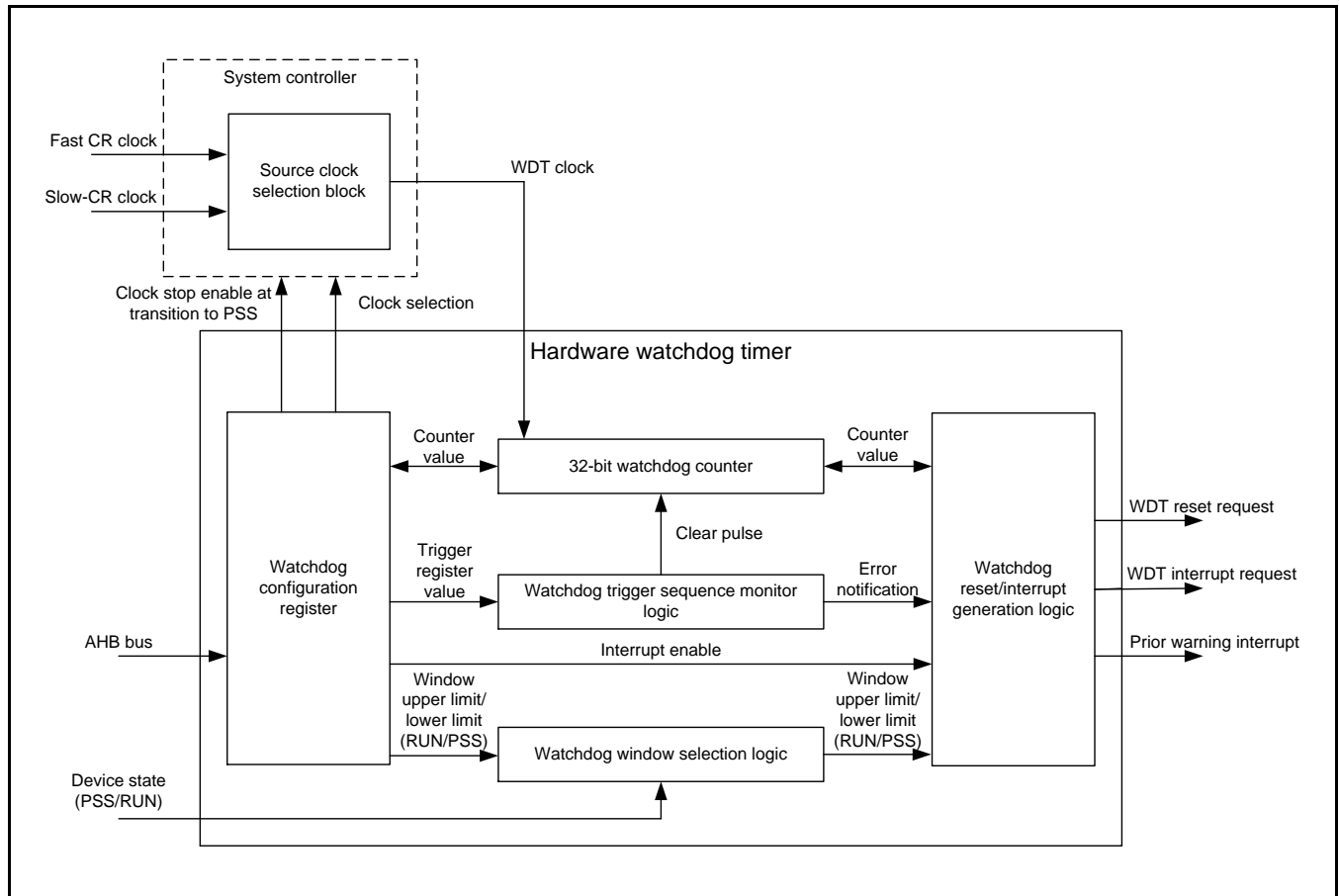
### Notes:

- The notation (\*a) indicates a feature that is specific to the hardware watchdog timer compared with the software watchdog.
- A watchdog error is a factor that causes the watchdog reset request or watchdog interrupt request (NMI).

## 2. Configuration

This section describes the block diagrams of the hardware watchdog timer.

Figure 2-1 Hardware Watchdog Timer Block Diagram



- Watchdog configuration register:  
In this block, the set value of each register is stored. I/O operations for information including the counter value for read operation, trigger register value necessary for clearing the counter, and window upper limit/lower limit values are performed.
- 32-bit watchdog counter:  
This is the block of the 32-bit up counter.
- Watchdog window selection logic:  
This block fetches the window upper limit value and lower limit value for each device state (RUN and PSS) and selects a window setting according to the device state transition.
- Watchdog reset/interrupt generation logic:  
This block generates the watchdog reset request, watchdog interrupt request (NMI), and prior warning interrupt request.



### 3. Operation

This section describes operation of the hardware watchdog timer.

#### (1) Hardware Watchdog Timer Functions

The hardware watchdog timer implements the following functions.

##### a) Reset Request Generation When Hardware Failure Causes Runaway State

The hardware watchdog timer performs monitoring to assure that there is no runaway state caused by a user program. The timer starts monitoring when it is activated by clearing a hard reset. If the watchdog counter is not cleared in the range between the upper limit value and the lower limit value of the window specified beforehand, this state is judged to be a runaway state caused by a user program. In this case, to the entire MCU, a reset request or an interrupt request (NMI) is generated.

**Note:**

- Detectable runaway states caused by hardware failures include, for example, one in which the BootROM program goes out of control as a result of a memory bit error after a hard reset is cleared.

##### b) Hardware Watchdog Timer Start by Clearing Hard Reset

Clearing a hard reset starts the hardware watchdog timer. Specifically, operation starts by using the clock generated by the fast-CR oscillation circuit immediately after the hardware watchdog is canceled.

##### c) Hardware Watchdog Timer Control by Main System Control CPU0

The hardware watchdog timer is assigned to main system control CPU0, and monitors control by CPU0 over the entire system. The number of hardware watchdog timers to be mounted in the MCU is 1.

##### d) Watchdog Counter Source Clock Selection (2 types)

The hardware watchdog timer sets values in the CLKSEL bits in the hardware watchdog configuration register (HWDG\_CFG) to select a source clock for the watchdog counter from among 2 types of source clocks. Specifically, it selects a source clock from the fast-CR clock or slow-CR clock. (The initial setting is for the fast-CR clock.)

Selection of a source clock is a clock request to the system controller. Source clock switching is performed on the clock system side of the system controller. For details on the procedure, see "(2) Switching of Watchdog Counter Source Clock" in Section "7. Precautions for Using".

##### e) Register Setting by BootROM Program

Hardware watchdog timer registers can be set by using the BootROM program.

#### f) 32-bit Watchdog Counter

The hardware watchdog timer operates by using the 32-bit watchdog counter (up counter) (the initial value is "0x0000\_0000").

The watchdog counter operates when all the following conditions are met.

1. The device is operating in user mode.
2. The watchdog counter has not reached the window upper limit value.
3. The device state is RUN and the WDENRUN bit is "1".

(See "6.15. Hardware Watchdog Configuration Register (HWDG\_CFG)".)

4. The device state is PSS and the WDENPSS bit is "1".

(See "6.15. Hardware Watchdog Configuration Register (HWDG\_CFG)".)

**Note:**

- When the watchdog counter reaches the window upper limit value, the count operation stops.

The following Table 3-1 shows the relationship between the window upper limit value and count time.

**Table 3-1 Relationship between Hardware Window Upper Limit Value of Watchdog Timer and Count Time**

Input Clock Frequency	Window Upper Limit Value	Count Time	Remarks
8MHz	"0x0100_0000"	About 2.1 s	It operates at the initial value of the hardware watchdog upper RUN setting register (HWDG_RUNULS).
12MHz	"0x0100_0000"	About 1.4 s	
8MHz	"0x8000_0000"	About 268 s	It operates at the initial value of the hardware watchdog upper PSS setting register (HWDG_PSSULS).
12MHz	"0x8000_0000"	About 178 s	

**Note:**

- The above table is an example involving fast-CR clock frequencies of 8 MHz and 12MHz.

#### g) Different Initializations between Soft Reset and Hard Reset

For the hardware watchdog timer, different initializations between soft reset and hard reset can be applied. The following table 3-2 shows the ranges of initialization for soft reset and hard reset.

**Table 3-2 Ranges of Initialization for Soft Reset and Hard Reset**

Conditions	Reset Name	Range of Initialization
1	Soft reset	Watchdog counter
2	Hard reset	Watchdog counter and settings for all registers





#### h) Watchdog Counter Clear Protection Trigger Sequence

The hardware watchdog timer monitors the watchdog counter clear protection trigger sequence to clear the watchdog counter.

The watchdog counter clear protection trigger sequence must meet the following conditions.

- Write data to the hardware watchdog trigger 1 register (HWDG\_TRG1) after writing data to the hardware watchdog trigger 0 register (HWDG\_TRG0).
- Do not execute the watchdog counter clear protection trigger sequence when the LOCK bit in the hardware watchdog configuration register (HWDG\_CFG) is "0".
- The value to be written to the hardware watchdog trigger 0 register (HWDG\_TRG0) must match the hardware watchdog trigger 0 configuration register (HWDG\_TRG0CFG).
- The value to be written to the hardware watchdog trigger 1 register (HWDG\_TRG1) must match the hardware watchdog trigger 1 configuration register (HWDG\_TRG1CFG).
- The watchdog counter must already have reached the window lower limit value when the watchdog counter clear protection trigger sequence is completed.

**Note:**

- For details on the watchdog counter clear protection trigger sequence, see Figure 3-2.

#### i) Window Watchdog Function

The window watchdog function is a function for setting a range in which the watchdog counter value can be cleared with upper limit and lower limit values. The following shows the range in which the watchdog counter can be cleared for 2 set values.

$$(\text{Window lower limit value}) \leq (\text{watchdog counter}) < (\text{window upper limit value})$$

For example, suppose that a runaway state occurs as a result of a user program and the watchdog counter clear protection trigger sequence is executed continuously. In this case, you can set a value other than "0x00000000" as a window lower limit value to detect the continuous watchdog counter clear as an abnormal operation.

The window setting for RUN is defined with the following 2 registers.

Hardware watchdog lower limit RUN current register (HWDG\_RUNLLC)

Hardware watchdog upper limit RUN current register (HWDG\_RUNULC)

The window setting for PSS is defined with the following 2 registers.

Hardware watchdog lower limit PSS current register (HWDG\_PSSLLC)

Hardware watchdog upper limit PSS current register (HWDG\_PSSULC)

**j) Different Operation Settings for each Device State (RUN and PSS)**

The hardware watchdog timer can make different operation settings for each device state (RUN and PSS). The operation setting is switched to the RUN or PSS operation setting according to the device state transition.

**RUN operation setting**

- When the LOCK bit and WDENRUN bit in the hardware watchdog configuration register (HWDG\_CFG) are set to "1", operation is performed by using the window setting for RUN.
- When the watchdog counter is not cleared in the range between the window upper limit value and lower limit value, or there is a violation in the watchdog counter clear protection trigger sequence, a watchdog error is detected. The watchdog reset request or watchdog interrupt request (NMI) is generated for this watchdog error. (For details on watchdog errors, see "(4) Watchdog Error" in Section "7. Precautions for Using".)

**PSS operation setting**

- The window setting is switched to the window setting for PSS when the device state transitions from RUN to PSS. When the WDENPSS bit in the hardware watchdog configuration register (HWDG\_CFG) is set to "1", operation is performed by using the window setting for PSS.
- When the device state transitions from PSS to RUN, the enable bit is switched from the WDENPSS bit in the hardware watchdog configuration register (HWDG\_CFG) to the WDENRUN bit in the same register. However, the window setting is not switched immediately. Use of the window setting for PSS continues until the watchdog counter clear protection trigger sequence is executed.

See Figure 3-4 and Figure 3-5 for the details on the watchdog operation in RUN and PSS, respectively.



### k) Watchdog Register Write Protection Sequence

The hardware watchdog timer monitors the watchdog register write protection sequence to write a setting value in the register.

The watchdog register write protection sequence must meet the following conditions.

- It writes a setting value in the register after writing data in the hardware watchdog protection register (HWDG\_PROT).
- It writes data in the hardware watchdog protection register (HWDG\_PROT) in privileged mode.
- The value written in the hardware watchdog protection register (HWDG\_PROT) must match the lock release key for register write protection "0xEDAC\_CE55".
- It does not write a setting value in any register in another module after writing data in the hardware watchdog protection register (HWDG\_PROT).
- It does not write data in the hardware watchdog protection register (HWDG\_PROT) twice in a row.
- It writes a value in the register when the LOCK bit in the hardware watchdog configuration register (HWDG\_CFG) is "0" and the mode is privileged mode.

**Note:**

- For details on the watchdog register write protection sequence, see Figure 3-1.

**Table 3-3 Key Value for Register Programming**

Key Value	Target Register	Purpose
"0xEDAC_CE55"	HWDG_PROT:KEY	Releasing protection lock for register write

**Note:**

- Setting the LOCK bit in the hardware watchdog configuration register (HWDG\_CFG) to "1" prevents the following registers from being rewritten.
  - Hardware watchdog interrupt configuration register (HWDG\_INT)
  - Hardware watchdog trigger 0 configuration register (HWDG\_TRG0CFG)
  - Hardware watchdog trigger 1 configuration register (HWDG\_TRG1CFG)
  - Hardware watchdog lower limit RUN setting register (HWDG\_RUNLLS)
  - Hardware watchdog upper limit RUN setting register (HWDG\_RUNULS)
  - Hardware watchdog lower limit PSS setting register (HWDG\_PSSLLS)
  - Hardware watchdog upper limit PSS setting register (HWDG\_PSSULS)
  - Hardware watchdog reset delay counter register (HWDG\_RSTDLY)
  - Hardware watchdog configuration register (HWDG\_CFG)

### I) Majority Circuits for Bits that Affect Important Functions

The hardware watchdog timer has majority circuits for bits that affect important functions. The majority circuit consists of 3 FFs. When the bit in an FF bit is inverted due to noise or because of another factor, the correct value can be selected by majority decision among the 3 FFs.

The bits that are equipped with majority circuits are as follows.

- RSTEN bit                      \*1
- LOCK bit                        \*2

#### Notes:

- The notation \*1 indicates a bit in the hardware watchdog interrupt configuration register (HWDG\_INT).
- The notation \*2 indicates a bit in the hardware watchdog configuration register (HWDG\_CFG).

### m) Generation of Watchdog Reset Request or Watchdog Interrupt Request (NMI)

The hardware watchdog timer generates the watchdog reset request or watchdog interrupt request (NMI) when detecting a watchdog error. Use the watchdog interrupt request (NMI) as a test function. (For example, you can use the watchdog interrupt request (NMI) to set a break point at the beginning of an interrupt handler.)

Also, the hardware watchdog timer can generate the prior warning interrupt request before the watchdog reset request (or watchdog interrupt request (NMI)). (For example, you can use the prior warning interrupt request so that the processor can save important data to the memory before the watchdog reset request is generated.)

The hardware watchdog timer generates these requests by using the following procedure.

- When a watchdog error occurs, the IRQFLAG bit in the hardware watchdog interrupt configuration register (HWDG\_INT) is set. At this time, if the IRQEN bit in the hardware watchdog interrupt configuration register (HWDG\_INT) is "1", the prior warning interrupt request is generated.
- As many cycles as are set in the hardware watchdog reset delay counter register (HWDG\_RSTDLY) are inserted between the prior warning interrupt request and the watchdog reset interrupt request or watchdog interrupt request (NMI) as a delay time.
- Once the prior warning interrupt request is generated, the watchdog counter cannot be cleared with the watchdog counter clear protection trigger sequence. Also, if a new watchdog error occurs, it is ignored.
- After the elapse of the time corresponding to the number of cycles for delay time set in the hardware watchdog reset delay counter register (HWDG\_RSTDLY), the watchdog reset request or watchdog interrupt request (NMI) is generated. If the hardware watchdog reset delay counter register (HWDG\_RSTDLY) is "0x0000", no delay time occurs.

#### Notes:

- When you generate the prior warning interrupt request, set the hardware watchdog reset delay counter register (HWDG\_RSTDLY) to a value other than "0x0000".
- The use of the watchdog interrupt request (NMI) is prohibited except when used as a test function. (This is prohibited because if such use is allowed, security will be compromised during the execution of an application.)
- Insertion of a number of cycles equal to that for the delay time set in the hardware watchdog reset delay counter register (HWDG\_RSTDLY) is valid only for 1 watchdog reset request or watchdog interrupt request (NMI). (It remains invalid until the next hard reset occurs.)



**n) Watchdog Counter Stop in Debugging State of Processor**

The watchdog counter stops when the processor enters the debugging state. When the processor returns from the debugging state, the watchdog counter resumes operation from the point where it had stopped.

**Note:**

- *For the definition of the debugging state of the processor, see the material from ARM® (Cortex™-R5 Revision:r1p2 Technical Reference Manual (ARM DDI 0460D)).*

## (2) Differences between Software Watchdog and Hardware Watchdog

The following Table 3-4 shows the main differences between the software watchdog and hardware watchdog.

**Table 3-4 Differences between Software Watchdog and Hardware Watchdog**

Item	Software Watchdog	Hardware Watchdog
Watchdog Counter	32-bit up counter	Same as on left
Different Initializations between Soft Reset and Hard Reset	Supported	Same as on left
Trigger Sequence for Watchdog Counter Clear	Supported	Same as on left
Window Watchdog Function	Supported	Same as on left
Different Operation Settings for Each Device State (RUN and PSS)	Supported	Same as on left
Watchdog Counter Source Clock Selection	3 types - Fast-CR clock - Slow-CR clock - Main clock	2 types - Fast-CR clock - Slow-CR clock
Watchdog Register Write Protection Sequence	Supported (However, prevention of rewriting of the window setting by the LOCK bit is not supported.)	Same as on left (However, prevention of rewriting of the window setting by the LOCK bit is supported.)
Majority Circuits for Bits That Affect Important Functions	Supported (4 target bits)	Same as on left (2 target bits)
Reset Request or Interrupt Request (NMI) Generation	Supported	Same as on left
Prior Warning Interrupt Request Generation	Supported	Same as on left
Watchdog Counter Stop in the Debugging State of the Processor	Supported	Same as on left
Watchdog Error Conditions	5 types	Same as on left
Bus Error Response Conditions	8 types	Same as on left
Watchdog Timer Start	Register write by a user program	Hard reset clear
Register Setting Value Write	User program	BootROM program
Watchdog Counter Enable Bit Control	Register write by a user program	Stop only in PSS (Fixed hardware)

### Notes:

- Unlike the case of the software watchdog timer, for the hardware watchdog timer, the BootROM program sets registers during the initial operation setting. (Write and read operation from the CPU can also be performed.)
- The following register setting values are determined by a marker in the flash memory (BootROM marker).
  - HWDT\_INT
  - HWDT\_TRG0CFG
  - HWDT\_TRG1CFG
  - HWDT\_RUNLLS
  - HWDT\_RUNULS
  - HWDT\_PSSLLS



- `HWDT_PSSULS`
- `HWDT_RSTDLY`
- `HWDT_CFG`
- *For details on the registers, see Section "6. Registers".*

Figure 3-1 Setting Procedure for Watchdog Register Write Protection Sequence

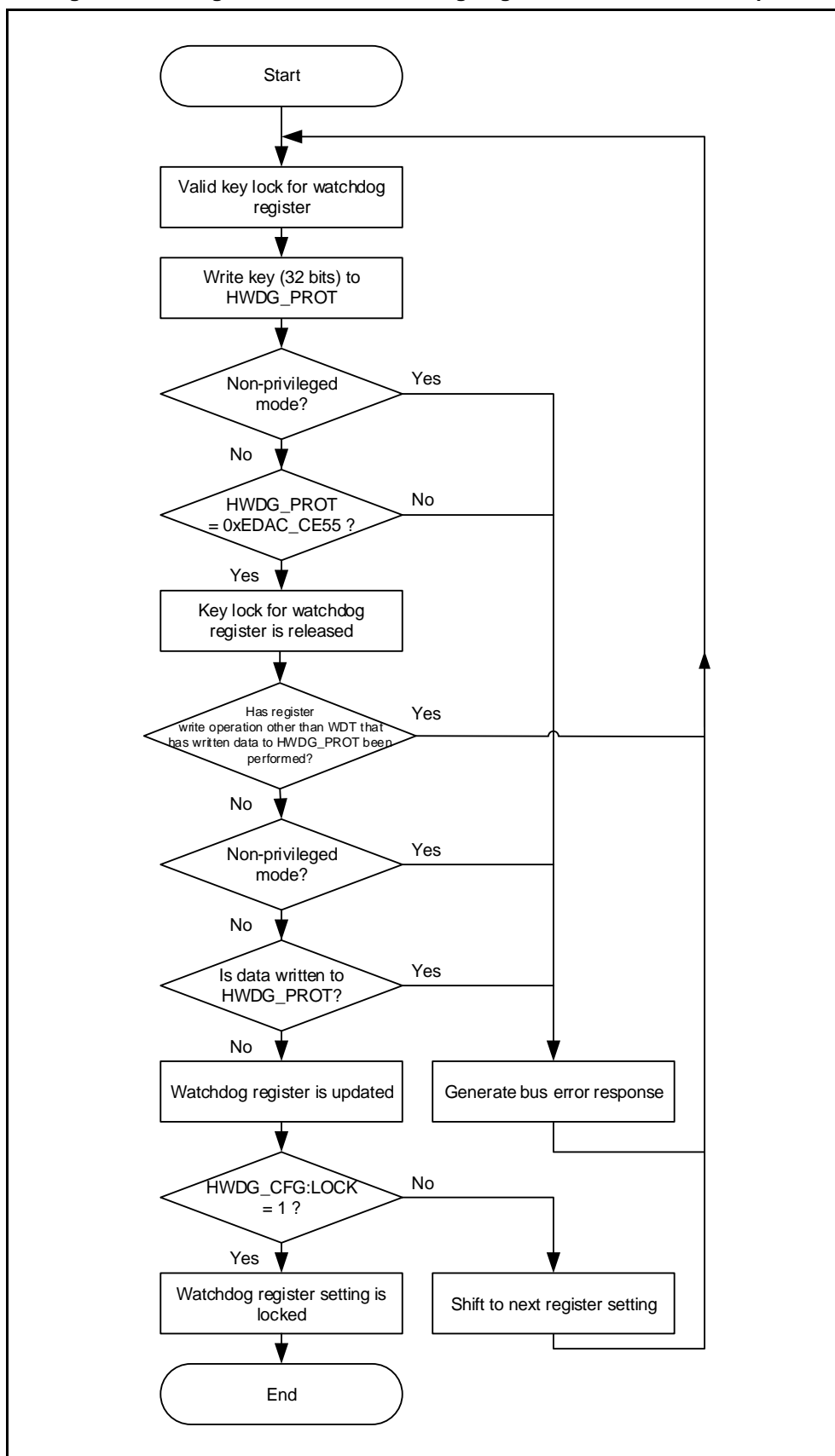
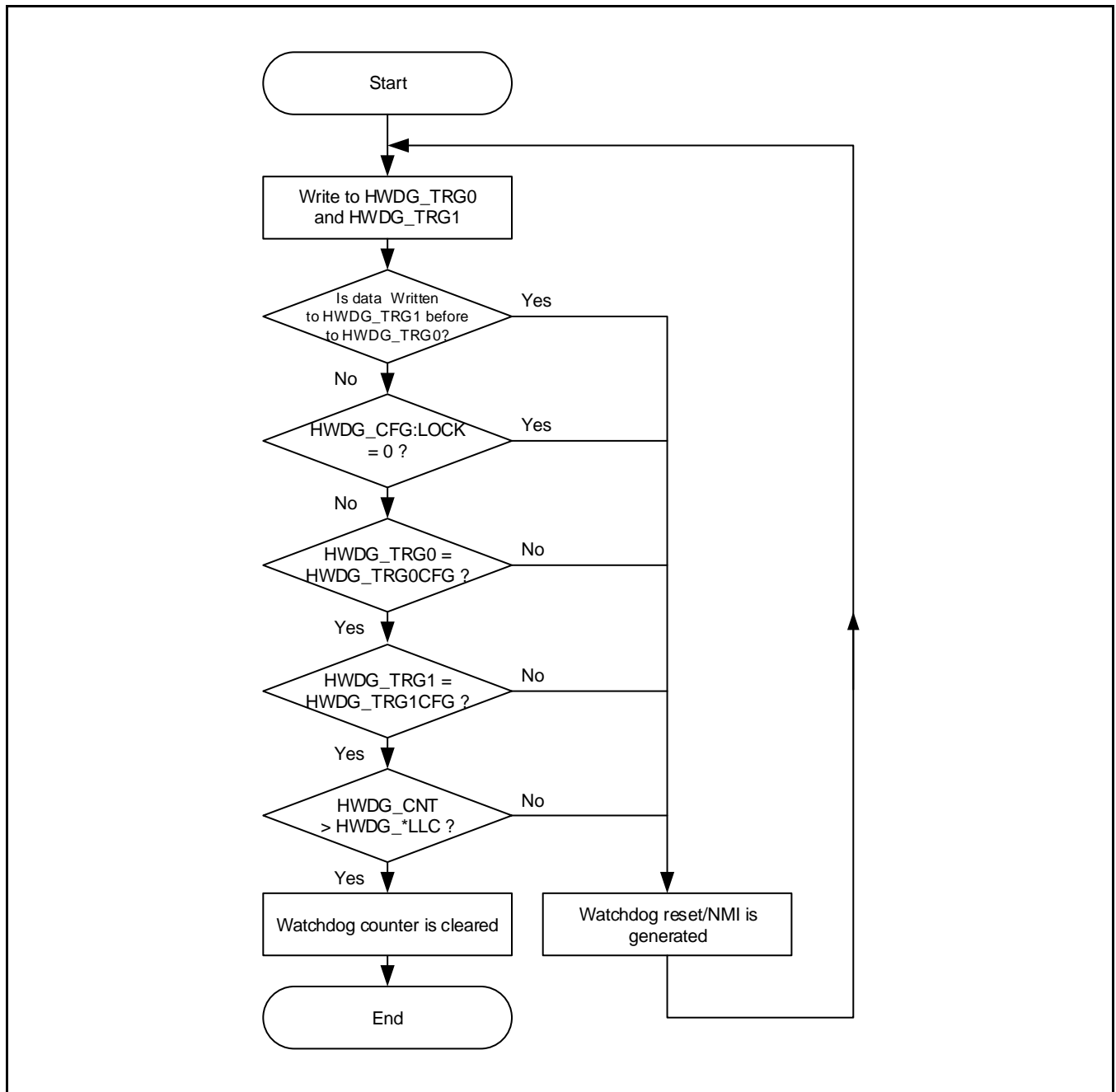




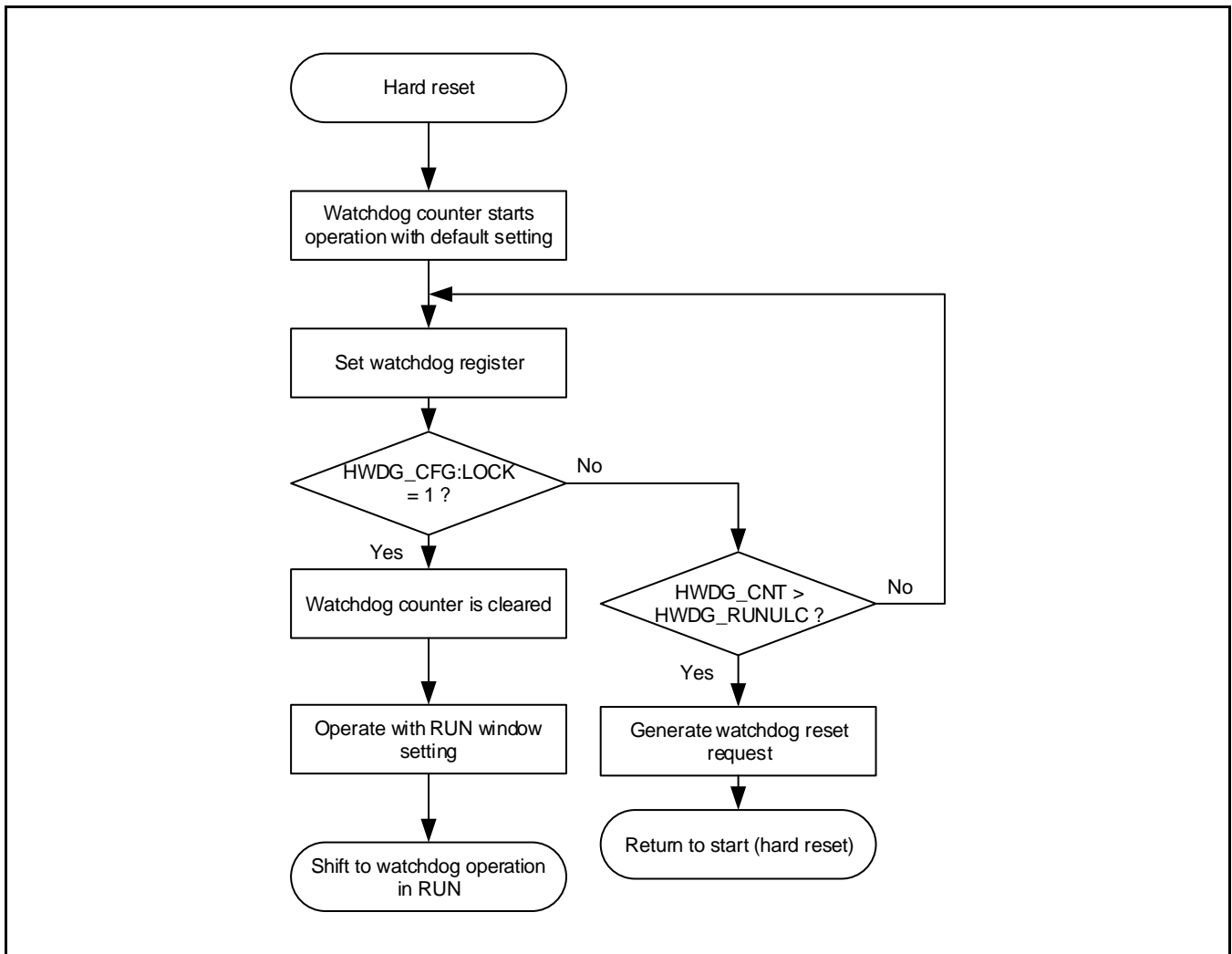
Figure 3-2 Setting Procedure for Watchdog Counter Clear Protection Trigger Sequence



**Note:**

- The notation \* in Figure 3-2 is replaced with device state RUN or PSS.

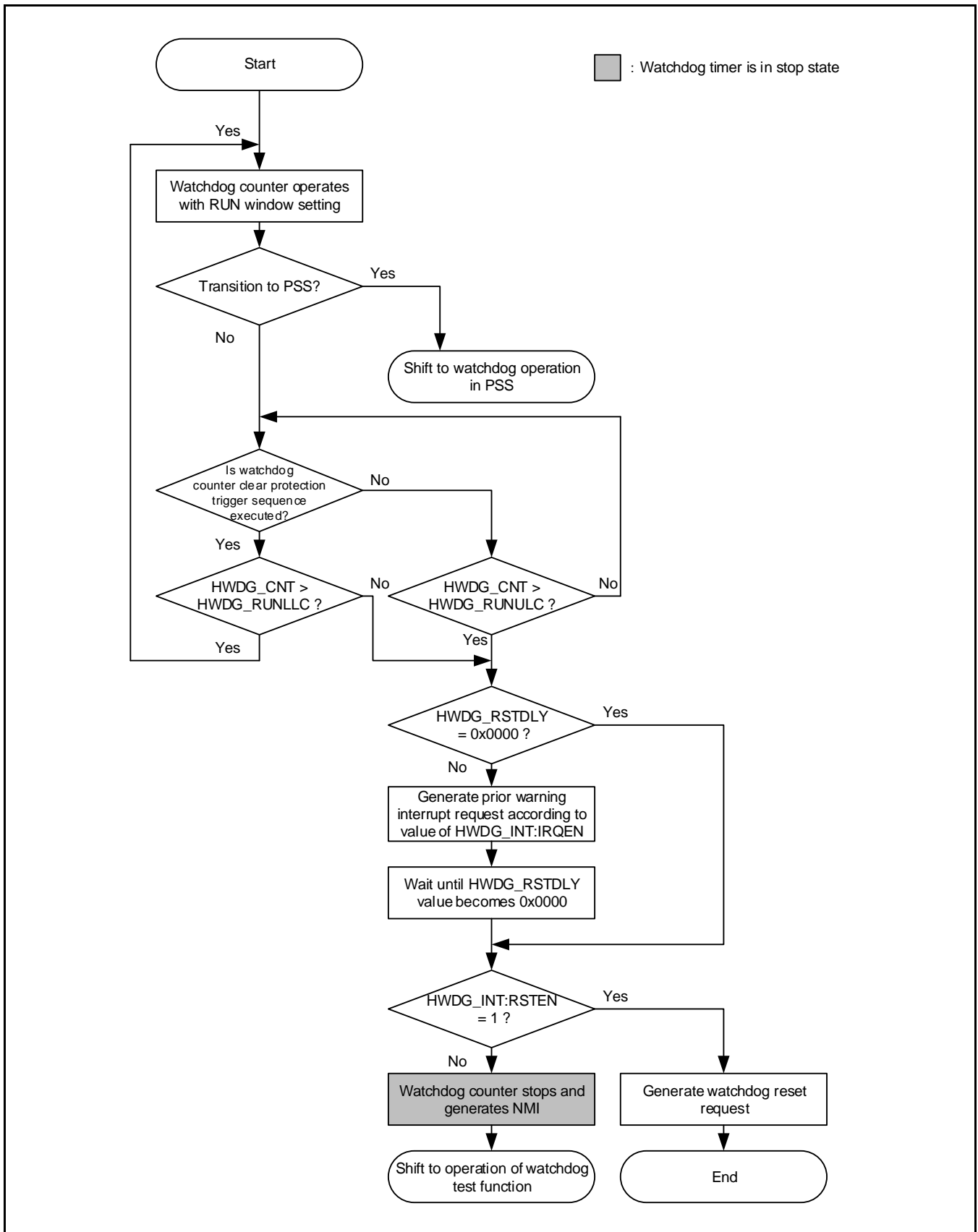
Figure 3-3 Startup of Hardware Watchdog Timer



1. The hard reset initializes the hardware watchdog timer. Then, immediately after the hard reset is cleared, the operating state is indicated.
2. The default window setting indicates the window lower limit value (= "0x0000\_0000") and window upper limit value (= "0x0100\_0000").
3. You can change the register setting as long as the LOCK bit in the hardware watchdog configuration register (HWDG\_CFG) is "0". However, complete the setting change within the time it takes to reach the window upper limit value.
4. Setting the LOCK bit in the hardware watchdog configuration register (HWDG\_CFG) to "1" clears the watchdog counter.

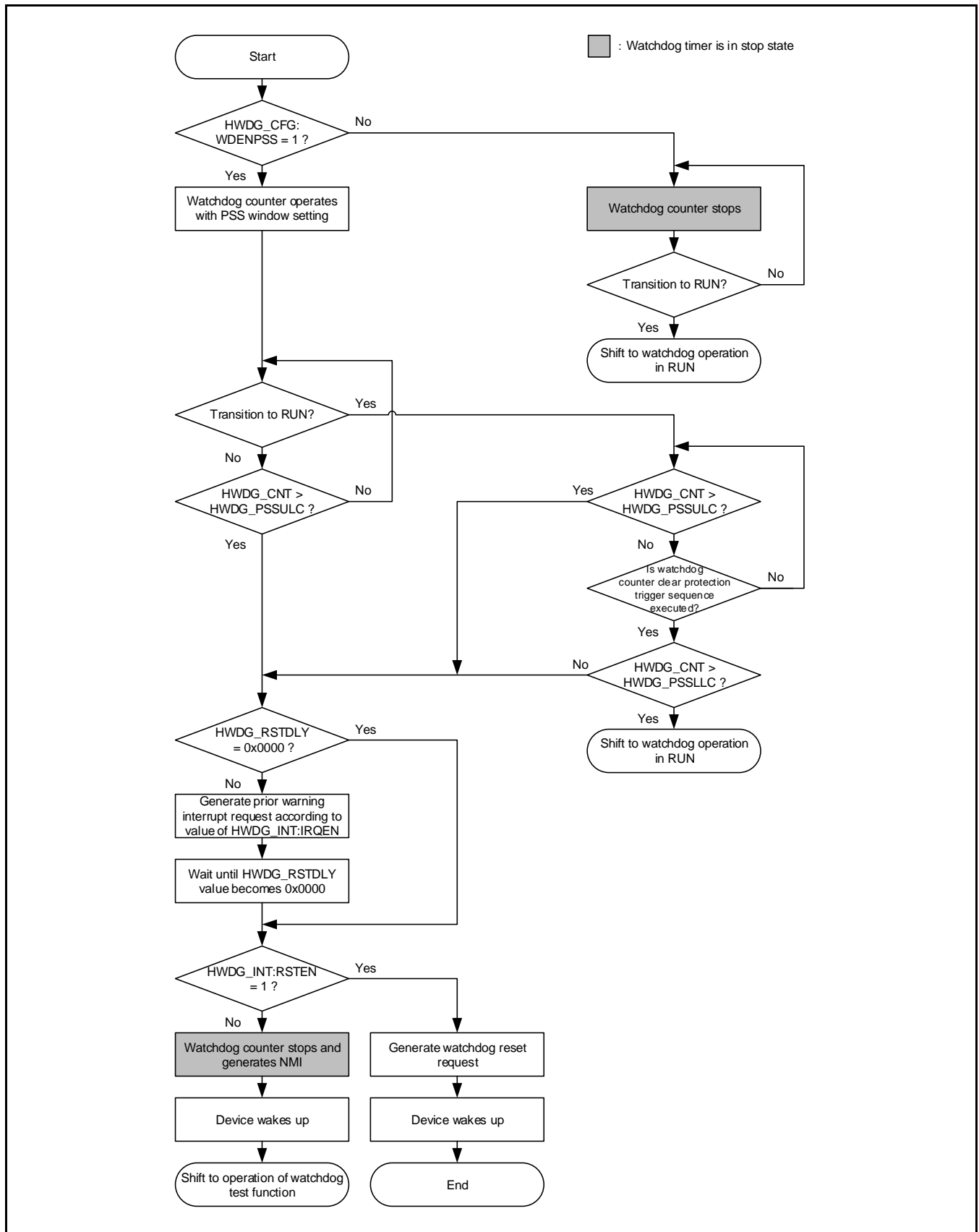
A chain of settings are executed by BootROM software.

Figure 3-4 Hardware Watchdog Operation in RUN



1. The watchdog counter in RUN always operates.
2. When the device state transitions from RUN to PSS, the processing transitions to the operation of the watchdog in PSS. For details, see Figure 3-5.
3. During the operation of the watchdog counter, execute the watchdog counter clear protection trigger sequence within the range between the window upper limit value and lower limit value that have been specified beforehand. Do this to clear the watchdog counter periodically.
4. If a runaway state caused by a user program occurs and prevents periodic clearing, the watchdog counter reaches the window upper limit value. Then, processing transitions to the flow that generates the watchdog reset request or watchdog interrupt request (NMI).
5. According to the value of the IRQEN bit in the hardware watchdog interrupt configuration register (HWDG\_INT), the prior warning interrupt request is generated. At the same time, as many cycles as are set for the delay time in the hardware watchdog reset delay counter register (HWDG\_RSTDLY) are inserted.
6. After the elapse of the time corresponding to the number of cycles for the delay time, the watchdog reset request or watchdog interrupt request (NMI) is generated according to the value of the RSTEN bit in the hardware watchdog interrupt configuration register (HWDG\_INT). When the watchdog interrupt request (NMI) is generated, the processing transitions to the operation of the test function of the watchdog. For details, see Figure 3-6.

Figure 3-5 Hardware Watchdog Operation in PSS

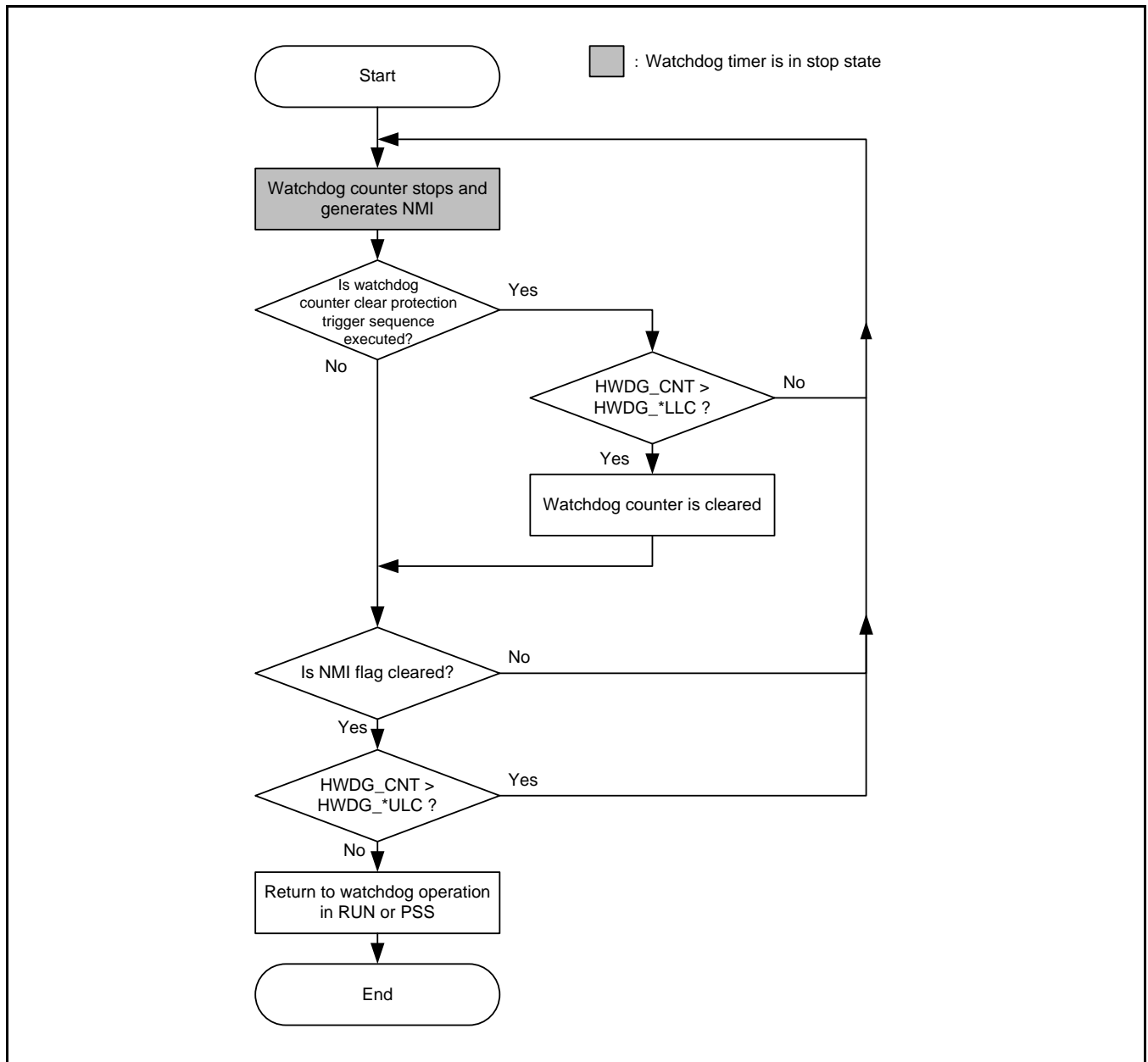


1. The watchdog counter in PSS operates or stops according to the value of the WDENPSS bit in the hardware watchdog configuration register (HWDG\_CFG).
2. When the device state transitions from PSS to RUN, the watchdog counter operation is started immediately. However, the PSS window setting continues to be used until the watchdog counter clear protection trigger sequence is executed.
3. During the operation of the watchdog counter, execute the watchdog counter clear protection trigger sequence within the range between the window upper limit value and lower limit value that have been specified beforehand. Continuously clearing the watchdog counter needs to be done periodically.
4. If a runaway state caused by a user program and prevents periodic clearing, the watchdog counter reaches the window upper limit value. Then, processing transitions to the flow that generates the watchdog reset request or watchdog interrupt request (NMI).
5. According to the value of the IRQEN bit in the hardware watchdog interrupt configuration register (HWDG\_INT), the prior warning interrupt request is generated. At the same time, as many cycles as are set for the delay time in the hardware watchdog reset delay counter register (HWDG\_RSTDLY) are inserted.
6. After the elapse of the time corresponding to the number of cycles for the delay time, the watchdog reset request or watchdog interrupt request (NMI) is generated according to the value of the RSTEN bit in the hardware watchdog interrupt configuration register (HWDG\_INT). When the watchdog interrupt request (NMI) is generated, the processing transitions to the operation of the test function of the watchdog. For details, see Figure 3-6.

**Note:**

- *If the watchdog counter clear protection trigger sequence is executed during operation with the PSS window setting, this means that the device state has already returned from PSS to RUN.*

Figure 3-6 Operation of Hardware Watchdog Test Function



1. The watchdog counter is in the stop state.
2. If the watchdog counter clear protection trigger sequence is executed in the stop state of the watchdog counter and the window lower limit value has already been reached at this time, the watchdog counter is cleared. (If this occurs outside the counter range, no special change occurs.)
3. When the NMIFLAG bit in the hardware watchdog interrupt configuration register (HWDG\_INT) is cleared, the watchdog interrupt request (NMI) is cleared.
4. When the watchdog counter has not reached the window upper limit value, processing returns to operation in RUN or PSS. When it has already reached the window upper limit value, no special change occurs.

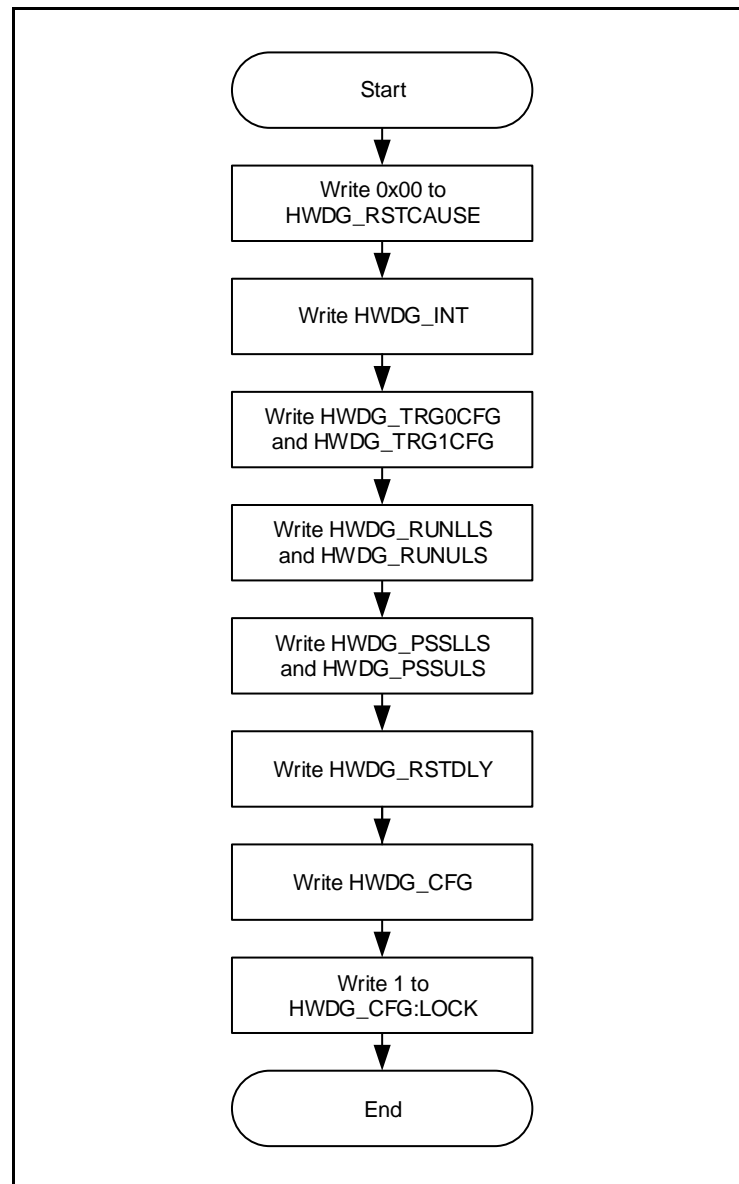
**Note:**

- The notation \* in Figure 3-6 is replaced with device state RUN or PSS.

## 4. Setting Procedure Example

This section explains an example of a procedure for setting the hardware watchdog timer.

Figure 4-1 Example of Hardware Watchdog Timer Register Setting Procedure



**Note:**

- The above registers are set by BootROM software.

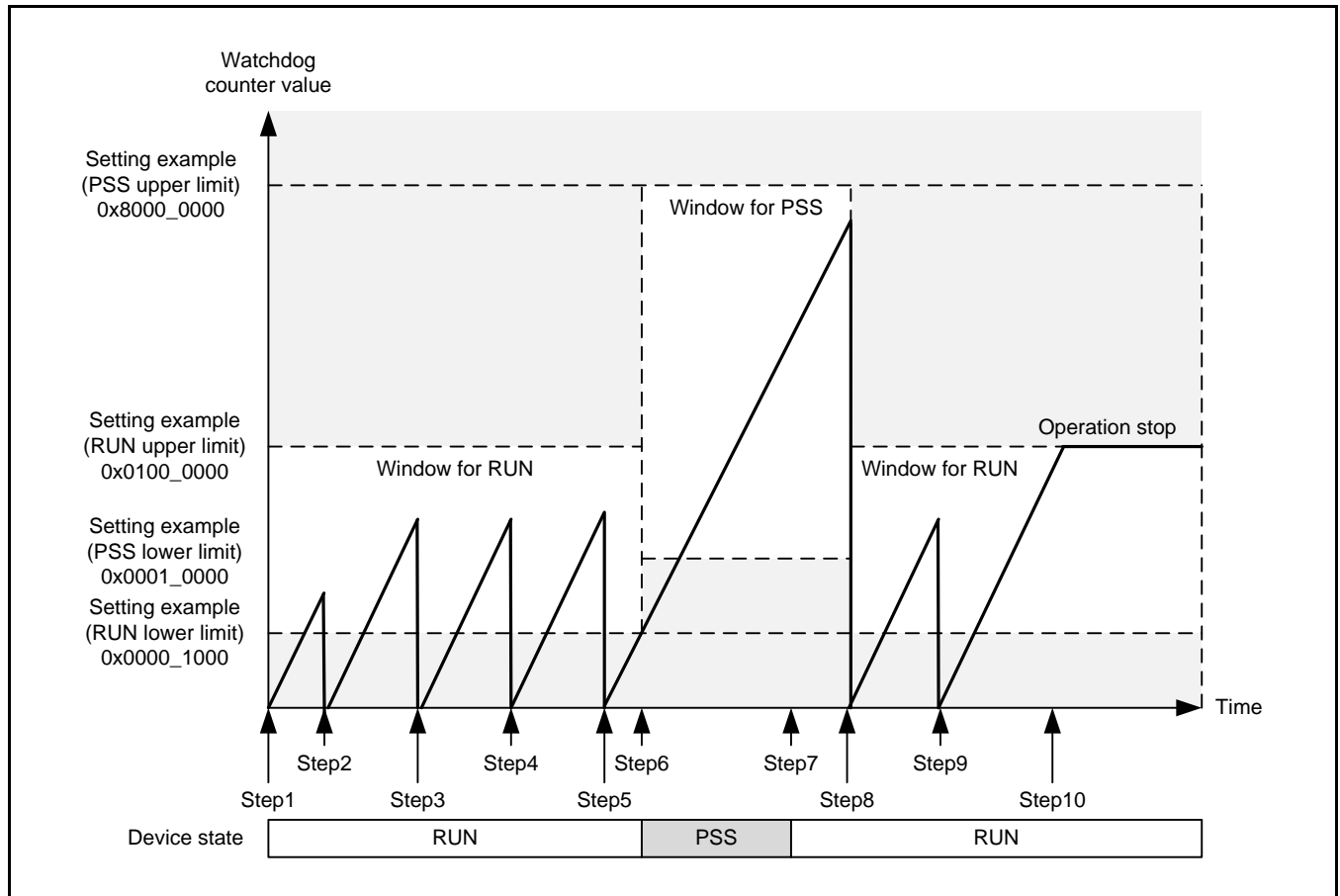




## 5. Operation Examples

This section describes examples of operating the hardware watchdog timer.

**Figure 5-1 Example of Operating Hardware Watchdog Timer (When Operation in PSS is Enabled)**



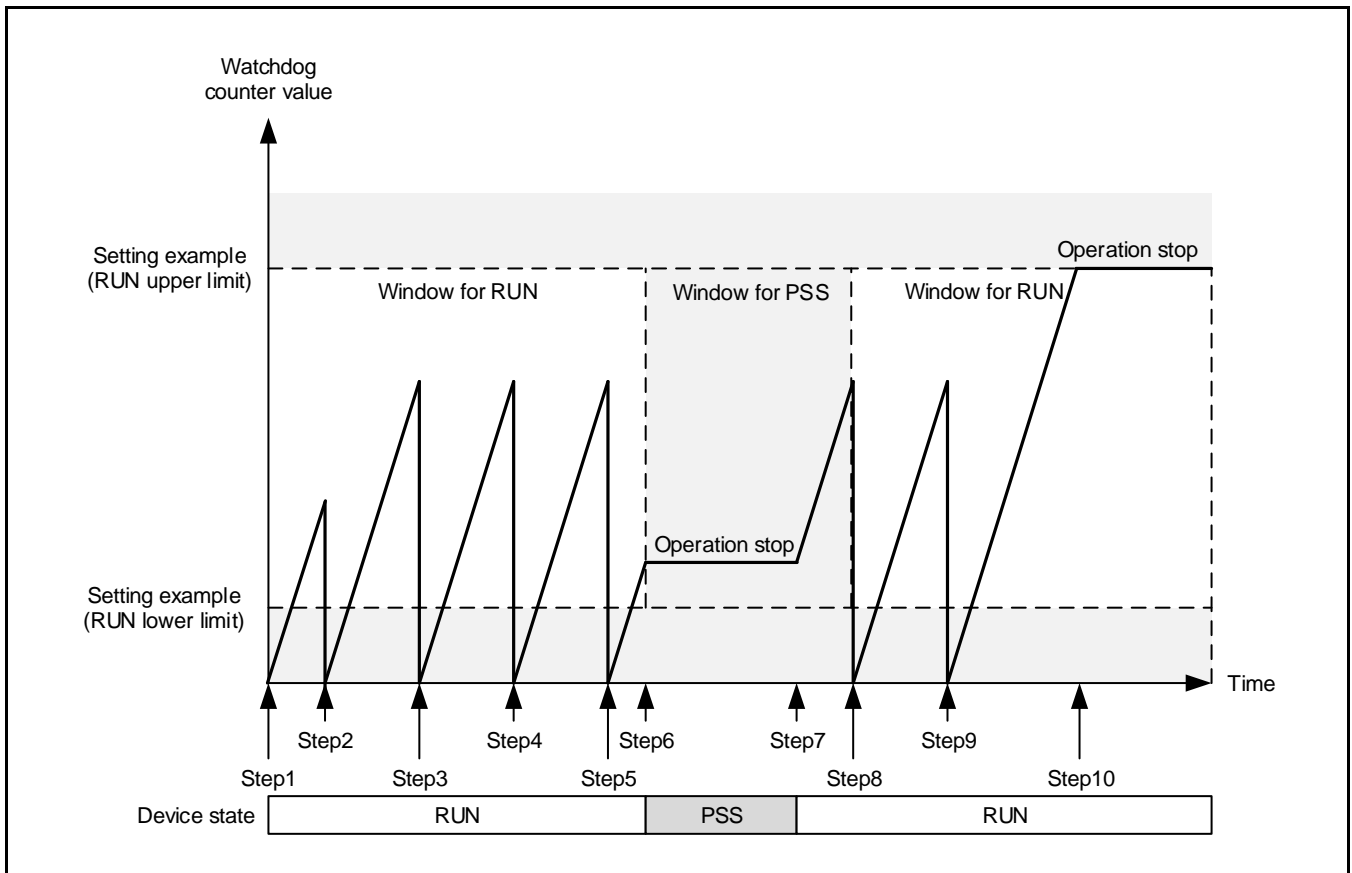
1. After power-on, the hardware watchdog timer starts operation. (The initial value of the window upper limit is "0x01000000").
2. After writing setting values in the registers, the LOCK bit in the hardware watchdog configuration register (HWDG\_CFG) is set to "1". At this time, the watchdog counter is automatically cleared, and it starts up-counting from "0x00000000".
3. The watchdog counter clear protection trigger sequence is executed, and the watchdog counter is cleared.
4. The watchdog counter clear protection trigger sequence is executed, and the watchdog counter is cleared.
5. The watchdog counter clear protection trigger sequence is executed, and the watchdog counter is cleared.
6. The device state transitions from RUN to PSS. At this time, switching to the PSS window setting occurs.
7. The device state returns from PSS to RUN. At this time, the switching to the RUN window setting does not occur immediately.
8. The watchdog counter clear protection trigger sequence is executed, and the watchdog counter is cleared. At this time, the switching to the RUN window setting occurs.
9. The watchdog counter clear protection trigger sequence is executed, and the watchdog counter is cleared.

10. When the watchdog counter reaches the window upper limit value, the watchdog reset request or watchdog interrupt request (NMI) is generated.

**Notes:**

- *The window setting is not immediately switched to the window setting for RUN when the device state is returned from PSS to RUN.*
- *Before the transition from RUN to PSS, be sure to clear the watchdog counter.*

Figure 5-2 Example of Operating Hardware Watchdog Timer (When Operation in PSS is Disabled)



1. After power-on, the hardware watchdog timer starts operation. (The initial value of the window upper limit is "0x0100\_0000".)
2. After writing setting values in the registers, the LOCK bit in the hardware watchdog configuration register (HWDG\_CFG) is set to "1". At this time, the watchdog counter is automatically cleared, and it starts up-counting from "0x00000000".
3. The watchdog counter clear protection trigger sequence is executed, and the watchdog counter is cleared.
4. The watchdog counter clear protection trigger sequence is executed, and the watchdog counter is cleared.
5. The watchdog counter clear protection trigger sequence is executed, and the watchdog counter is cleared.
6. The device state transitions from RUN to PSS. At this time, the hardware watchdog timer stops operation.
7. The device state returns from PSS to RUN. At this time, the hardware watchdog timer immediately starts operation.
8. The watchdog counter clear protection trigger sequence is executed, and the watchdog counter is cleared.
9. The watchdog counter clear protection trigger sequence is executed, and the watchdog counter is cleared.
10. When the watchdog counter reaches the window upper limit value, the watchdog reset request or watchdog interrupt request (NMI) is generated.

## 6. Registers

This section explains the registers of the hardware watchdog timer.

**Table 6-1 List of Hardware Watchdog Timer Registers**

Abbreviated Register Name	Register Name	See
HWDG_PROT	Hardware Watchdog Protection Register	6.1
HWDG_CNT	Hardware Watchdog Counter Register	6.2
HWDG_RSTCAUSE	Hardware Watchdog Reset Factor register	6.3
HWDG_TRG0	Hardware Watchdog Trigger 0 Register	6.4
HWDG_TRG1	Hardware Watchdog Trigger 1 Register	6.5
HWDG_INT	Hardware Watchdog Interrupt Configuration Register	6.6
HWDG_INTCLR	Hardware Watchdog Interrupt Clear Register	6.7
HWDG_TRG0CFG	Hardware Watchdog Trigger 0 Configuration Register	6.8
HWDG_TRG1CFG	Hardware Watchdog Trigger 1 Configuration Register	6.9
HWDG_RUNLLS	Hardware Watchdog Lower Limit RUN Setting Register	6.10
HWDG_RUNULS	Hardware Watchdog Upper Limit RUN Setting Register	6.11
HWDG_PSSLLS	Hardware Watchdog Lower Limit PSS Setting Register	6.12
HWDG_PSSULS	Hardware Watchdog Upper Limit PSS Setting Register	6.13
HWDG_RSTDLY	Hardware Watchdog Reset Delay Counter Register	6.14
HWDG_CFG	Hardware Watchdog Configuration Register	6.15
HWDG_RUNLLC	Hardware Watchdog Lower Limit RUN Current Register	6.16
HWDG_RUNULC	Hardware Watchdog Upper Limit RUN Current Register	6.17
HWDG_PSSLLC	Hardware Watchdog Lower Limit PSS Current Register	6.18
HWDG_PSSULC	Hardware Watchdog Upper Limit PSS Current Register	6.19



## 6.1. Hardware Watchdog Protection Register (HWDG\_PROT)

This register is used to execute the watchdog register write protection sequence. Write the correct key ("0xEDAC\_CE55") to this register before performing write access to each register. (However, the hardware watchdog trigger 0/1 register (HWDG\_TRG0 and HWDG\_TRG1) are excluded. ) Writing the correct key releases the write protection lock for subsequent register access. Write access to each register enables write protection lock for the registers. (Read access to each register does not affect the lock. ) For write access to this register, you must write 32-bit data.

BIT_OFFSET	31-0
BIT_NAME	KEY
ACCESS_TYPE	R,W
PROT_TYPE	WP
INITIAL_VALUE	00000000_00000000_00000000_00000000

### [bit31:0] KEY[31:0]: Protection bits

bit31:0	Description
During write operation	<p>When "0xEDAC_CE55" is written: Protection lock for register write is released.</p> <p>When a value other than "0xEDAC_CE55" is written: Protection lock for register write is not released.</p>
During read operation	<p>When "0xFFFF_FFFF" is read: Protection lock for register write has been released.</p> <p>When "0x0000_0000" is read: Protection lock for register write has been enabled.</p>

#### Notes:

- An AHB transfer error response is generated when write access to this register is performed under any of the conditions below. (This error response invokes the CPU exception handler.)
  - A wrong key is written to this register.
  - Non-32-bit data is written to this register.
  - Data is written to this register twice in a row.

For details on the watchdog register write protection sequence, see Figure 3-1.

A protect key will be locked again by writing to the address where is in the same group area (MCU Config Group), however protect key will not be locked by writing to no protect target register.

## 6.2. Hardware Watchdog Counter Register (HWDG\_CNT)

This register indicates the current up count value of the watchdog counter.

BIT_OFFSET	31-0
BIT_NAME	WDGCNT
ACCESS_TYPE	R,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000_00000000

### [bit31:0] WDGCNT[31:0]: Watchdog counter bits

bit31:0	Description
During write operation	Invalid
During read operation	Returns the current watchdog counter value. The watchdog counter value is sampled by using the bus clock.

**Note:**

- An AHB transfer error response is generated where there is write access to this register.



### 6.3. Hardware Watchdog Reset Factor Register (HWDG\_RSTCAUSE)

When performing write access to this register, follow the watchdog register write protection sequence. This register is the status register that indicates the factor of the watchdog reset request or watchdog interrupt request (NMI). This register is not initialized by reset. To check the factor of the watchdog reset request, read this register.

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	WPS
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved			RST CAUSE4	RST CAUSE3	RST CAUSE2	RST CAUSE1	RST CAUSE0
ACCESS_TYPE	R0,WX			R,W	R,W	R,W	R,W	R,W
PROT_TYPE	WPS							
INITIAL_VALUE	000			X	X	X	X	X

**[bit31:5] Reserved: Reserved bits**

#### [bit4] RSTCAUSE4: Reset factor bit 4

If the watchdog counter clear protection trigger sequence is executed when the LOCK bit in the hardware watchdog configuration register (HWDG\_CFG) is "0", this bit is set to "1".

bit	Description
During write operation	When "0" is written: This bit cleared When "1" is written: Invalid
During read operation	When "0" is read: Indicating no detection of a reset/NMI factor When "1" is read: Indicating detection of a reset/NMI factor

#### [bit3] RSTCAUSE3: Reset factor bit 3

If the watchdog counter clear protection trigger sequence is executed before the watchdog counter reaches the window lower limit value, this bit is set to "1".

bit	Description
During write operation	When "0" is written: This bit cleared When "1" is written: Invalid
During read operation	When "0" is read: Indicating no detection of a reset/NMI factor When "1" is read: Indicating detection of a reset/NMI factor

**[bit2] RSTCAUSE2: Reset factor bit 2**

When the watchdog counter reaches the window upper limit value, this bit is set to "1".

bit	Description
During write operation	When "0" is written: This bit cleared When "1" is written: Invalid
During read operation	When "0" is read: Indicating no detection of a reset/NMI factor When "1" is read: Indicating detection of a reset/NMI factor

When the prior warning interrupt request is used, RSTCAUSE2 cannot be cleared during the period from the occurrence of a prior warning interrupt until a watchdog reset or watchdog interrupt (NMI).

**[bit1] RSTCAUSE1: Reset factor bit 1**

When there is a violation of the watchdog counter clear protection trigger sequence, this bit is set to "1".  
For details, see Figure 3-2.

bit	Description
During write operation	When "0" is written: This bit cleared When "1" is written: Invalid
During read operation	When "0" is read: Indicating no detection of a reset/NMI factor When "1" is read: Indicating detection of a reset/NMI factor

**[bit0] RSTCAUSE0: Reset factor bit 0**

If the value written in the hardware watchdog trigger 0/1 register (HWDG\_TRG0 or HWDG\_TRG1) does not match a proper value, this bit is set to "1".

bit	Description
During write operation	When "0" is written: This bit cleared When "1" is written: Invalid
During read operation	When "0" is read: Indicating no detection of a reset/NMI factor When "1" is read: Indicating detection of a reset/NMI factor

**Notes:**

- When using the hardware watchdog timer, be sure to clear this register.
- The value in this register becomes valid when the watchdog reset request is generated.





## 6.4. Hardware Watchdog Trigger 0 Register (HWDG\_TRG0)

When performing write access to this register, you do not need to follow the watchdog register write protection sequence. This register is used to execute the watchdog counter clear protection trigger sequence. Write the value defined in the hardware watchdog trigger 0 configuration register (HWDG\_TRG0CFG) to this register. When a value other than the one in the hardware watchdog trigger 0 configuration register (HWDG\_TRG0CFG) is written, the watchdog reset request or watchdog interrupt request (NMI) is generated.

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	WDGTRG0							
ACCESS_TYPE	R0,W							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

**[bit31:8] Reserved: Reserved bits**

### **[bit7:0] WDGTRG0[7:0]: Watchdog trigger 0 bits**

These bits are used to execute the watchdog counter clear protection trigger sequence to clear the watchdog counter.

bit7:0	Description
During write operation	When the HWDG_TRG0CFG value is written: 1 condition for executing the watchdog counter clear protection trigger sequence is met. When a value other than the HWDG_TRG0CFG value is written: A watchdog error is generated.
During read operation	Reads "0b00000000".

**Note:**

- Figure 3-2 shows the flow of watchdog counter clear protection trigger sequence.

## 6.5. Hardware Watchdog Trigger 1 Register (HWDG\_TRG1)

When performing write access to this register, you do not need to follow the watchdog register write protection sequence. This register is used to execute the watchdog counter clear protection trigger sequence. Write the value defined in the hardware watchdog trigger 1 configuration register (HWDG\_TRG1CFG) to this register. When a value other than the one in the hardware watchdog trigger 1 configuration register (HWDG\_TRG1CFG) is written, the watchdog reset request or watchdog interrupt request (NMI) is generated.

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	WDGTRG1							
ACCESS_TYPE	R0,W							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

**[bit31:8] Reserved: Reserved bits**

### **[bit7:0] WDGTRG1[7:0]: Watchdog trigger 1 bits**

These bits are used to execute the watchdog counter clear protection trigger sequence to clear the watchdog counter.

bit7:0	Description
During write operation	When the HWDG_TRG1CFG value is written: 1 condition for executing the watchdog counter clear protection trigger sequence is met. When a value other than the HWDG_TRG1CFG value is written: A watchdog error is generated.
During read operation	Reads "0b00000000".

**Note:**

- Figure 3-2 shows the flow of watchdog counter clear protection trigger sequence.



## 6.6. Hardware Watchdog Interrupt Configuration Register (HWDG\_INT)

When performing write access to this register, follow the watchdog register write protection sequence. This register is used to make settings related to the watchdog interrupt request (NMI) and prior warning interrupt request. This register also includes interrupt status flags.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved						RSTEN	IRQEN
ACCESS_TYPE	R0,WX						R/W	R/W
PROT_TYPE	WPS							
INITIAL_VALUE	000000						1	0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved						NMIFLAG	IRQFLAG
ACCESS_TYPE	R0,WX						R,WX	R,WX
PROT_TYPE	WPS							
INITIAL_VALUE	00000000_000000						0	0

**[bit31:18] Reserved: Reserved bits**

### **[bit17] RSTEN: Reset/NMI enable bit**

This bit is used to generate either the watchdog reset request or watchdog interrupt request (NMI) in response to a watchdog error. Prohibiting to set this bit to "0" except for the purpose of using it as a test function. (If this bit is set to "0" during application execution, security is compromised.) This bit is equipped with a majority circuit with 3 FFs as a measure against a bit invert caused by the effects of noise or another factor.

bit	Description
During write operation	When "0" is written: The watchdog interrupt request (NMI) is generated. When "1" is written: The watchdog interrupt request is generated.
During read operation	Set value read

**[bit16] IRQEN: Prior warning interrupt enable bit**

This bit is used to enable generation of the prior warning interrupt request.

bit	Description
During write operation	When "0" is written: The prior warning interrupt request is not generated. When "1" is written: The prior warning interrupt request is generated.
During read operation	Set value read

**[bit15:2] Reserved: Reserved bits**

**[bit1] NMIFLAG: NMI flag**

This bit is set by a watchdog error when the RSTEN bit in the hardware watchdog interrupt configuration register (HWDG\_INT) is "0". When the RSTEN bit in the hardware watchdog interrupt clear register (HWDG\_INT) is "1", the watchdog reset request is generated. You can clear this bit by writing "0" to the NMICLR bit in the hardware watchdog interrupt configuration register (HWDG\_INTCLR).

bit	Description
During write operation	Invalid
During read operation	When "0" is read: No detection of a watchdog error (NMI) is indicated. When "1" is read: Detection of a watchdog error (NMI) is indicated.

**[bit0] IRQFLAG: IRQ flag**

This bit is set by a watchdog error. The prior warning interrupt is generated when the IRQEN bit in the hardware watchdog interrupt clear register (HWDG\_INT) is "1". You can clear this bit by writing "1" to the IRQCLR bit in the hardware watchdog interrupt configuration register (HWDG\_INTCLR).

bit	Description
During write operation	Invalid
During read operation	When "0" is read: No detection of a watchdog error (IRQ) is indicated. When "1" is read: Detection of a watchdog error (IRQ) is indicated.

**Notes:**

- For details on the watchdog interrupt request (NMI), see "(m) Generation of Watchdog Reset Request or Watchdog Interrupt Request (NMI)" in Section "3. Operation".
- For details on watchdog errors, see "(4) Watchdog Error" in Section "7. Precautions for Using".



## 6.7. Hardware Watchdog Interrupt Clear Register (HWDG\_INTCLR)

When performing write access to this register, follow the watchdog register write protection sequence. This register is used to clear the NMI flag and IRQ flag in the hardware watchdog interrupt configuration register (HWDG\_INT). You can write data to this register even after the LOCK bit in the hardware watchdog configuration register (HWDG\_CFG) is set.

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	WPS
INITIAL_VALUE	00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved						NMICLR	IRQCLR
ACCESS_TYPE	R0,WX						R0,W	R0,W
PROT_TYPE	WPS							
INITIAL_VALUE	000000						0	0

**[bit31:2] Reserved: Reserved bits**

### **[bit1] NMICLR: NMI clear bit**

This bit is used to clear the NMIFLAG bit in the hardware watchdog interrupt configuration register (HWDG\_INT).

bit	Description
During write operation	When "0" is written: Invalid When "1" is written: The NMI flag is cleared.
During read operation	"0" is read.

You must clear the NMI after waiting for watchdog counter clear (0x0000\_0000). Check the watchdog counter value with HWDG\_CNT.

### **[bit0] IRQCLR: Prior warning interrupt clear bit**

This bit is used to clear the IRQFLAG bit in the hardware watchdog interrupt configuration register (HWDG\_INT).

bit	Description
During write operation	When "0" is written: Invalid When "1" is written: The IRQ flag is cleared.
During read operation	"0" is read.

## 6.8. Hardware Watchdog Trigger 0 Configuration Register (HWDG\_TRG0CFG)

When performing write access to this register, follow the watchdog register write protection sequence. This register is used to define a valid value to be written to the hardware watchdog trigger 0 register (HWDG\_TRG0) when the watchdog counter clear protection trigger sequence is performed.

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	WPS
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	WDGTRG0CFG							
ACCESS_TYPE	R/W							
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							

**[bit31:8] Reserved: Reserved bits**

### **[bit7:0] WDGTRG0CFG[7:0]: Watchdog trigger 0 configuration bits**

These bits are used to define a value to be written to the hardware watchdog trigger 0 register (HWDG\_TRG0) for execution of the watchdog counter clear protection trigger sequence.

bit7:0	Description
During write operation	Set value written
During read operation	Set value read



## 6.9. Hardware Watchdog Trigger 1 Configuration Register (HWDG\_TRG1CFG)

When performing write access to this register, follow the watchdog register write protection sequence. This register is used to define a valid value to be written to the hardware watchdog trigger 1 register (HWDG\_TRG1) when the watchdog counter clear protection trigger sequence is performed.

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	WPS
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	WDGTRG1CFG							
ACCESS_TYPE	R/W							
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							

**[bit31:8] Reserved: Reserved bits**

### **[bit7:0] WDGTRG1CFG[7:0]: Watchdog trigger 1 configuration bits**

These bits are used to define a value to be written to the hardware watchdog trigger 1 register (HWDG\_TRG1) for execution of the watchdog counter clear protection trigger sequence.

bit7:0	Description
During write operation	Set value written
During read operation	Set value read

## 6.10. Hardware Watchdog Lower Limit RUN Setting Register (HWDG\_RUNLLS)

When performing write access to this register, follow the watchdog register write protection sequence. A setting value of the window lower limit value for RUN is written to this register. However, this register value is different from the window lower limit value for RUN that is actually used. For details on how to reflect this register value in the window lower limit value for RUN actually used, see Section "6.16. Hardware Watchdog Lower Limit RUN Current Register (HWDG\_RUNLLC)". You cannot change the set value of this register after the LOCK bit in the hardware watchdog configuration register (HWDG\_CFG) is set.

BIT_OFFSET	31-0
BIT_NAME	WDGRUNLLS
ACCESS_TYPE	R/W
PROT_TYPE	WPS
INITIAL_VALUE	00000000_00000000_00000000_00000000

### [bit31:0] WDGRUNLLS[31:0]: Window lower limit value for RUN set bits

These bits are used to define the window lower limit value for RUN. The reading of these bits returns the set value regardless of the hardware watchdog configuration register (HWDG\_CFG) LOCK bit value.

bit31:0	Description
During write operation	When "0" is written to all 32 bits: The window function is disabled. When any value except "0" is written: The window function is enabled.
During read operation	Set value read

**Note:**

- This register value is different from the window lower limit value that is actually used. The window lower limit value actually used is the value in the hardware watchdog lower limit RUN current register (HWDG\_RUNLLC).





## 6.11. Hardware Watchdog Upper Limit RUN Setting Register (HWDG\_RUNULS)

When performing write access to this register, follow the watchdog register write protection sequence. A setting value of the window upper limit value for RUN is written to this register. However, this register value is different from the window upper limit value for RUN that is actually used. For details on how to reflect this register value in the window upper limit value for RUN actually used, see Section "6.17. Hardware Watchdog Upper Limit RUN Current Register (HWDG\_RUNULC)". You cannot change the set value of this register after the LOCK bit in the hardware watchdog configuration register (HWDG\_CFG) is set.

BIT_OFFSET	31-0
BIT_NAME	WDGRUNULS
ACCESS_TYPE	R/W
PROT_TYPE	WPS
INITIAL_VALUE	00000001_00000000_00000000_00000000

### [bit31:0] WDGRUNULS[31:0]: Window upper limit value for RUN set bits

These bits are used to define the window upper limit value for RUN. The reading of these bits returns the set value regardless of the hardware watchdog configuration register (HWDG\_CFG) LOCK bit value.

bit31:0	Description
During write operation	Set value written
During read operation	Set value read

**Note:**

- This register value is different from the window upper limit value that is actually used. The window lower limit value actually used is the value in the hardware watchdog lower limit RUN current register (HWDG\_RUNLLC).

## 6.12. Hardware Watchdog Lower Limit PSS Setting Register (HWDG\_PSSLLS)

When performing write access to this register, follow the watchdog register write protection sequence. A setting value of the window lower limit value for PSS is written to this register. However, this register value is different from the window lower limit value for PSS that is actually used. For details on how to reflect this register value in the window lower limit value for PSS actually used, see Section "6.18. Hardware Watchdog Lower Limit PSS Current Register (HWDG\_PSSLLC)". You cannot change the set value of this register after the LOCK bit in the hardware watchdog configuration register (HWDG\_CFG) is set.

BIT_OFFSET	31-0
BIT_NAME	WDGPSSLLS
ACCESS_TYPE	R/W
PROT_TYPE	WPS
INITIAL_VALUE	00000000_00000000_00000000_00000000

### [bit31:0] WDGPSSLLS[31:0]: Window lower limit value for PSS set bits

These bits are used to define the window lower limit value for PSS. The reading of these bits returns the set value regardless of the hardware watchdog configuration register (HWDG\_CFG) LOCK bit value.

bit31:0	Description
During write operation	When "0" is written to all 32 bits: The window function is disabled. When any value except "0" is written: The window function is enabled.
During read operation	Set value read

**Note:**

- This register value is different from the window lower limit value that is actually used. The window lower limit value actually used is the value in the hardware watchdog lower limit PSS current register (HWDG\_PSSLLC).



### 6.13. Hardware Watchdog Upper Limit PSS Setting Register (HWDG\_PSSULS)

When performing write access to this register, follow the watchdog register write protection sequence. A setting value of the window upper limit value for PSS is written to this register. However, this register value is different from the window upper limit value for PSS that is actually used. For details on how to reflect this register value in the window upper limit value for PSS actually used, see Section "6.19. Hardware Watchdog Upper Limit PSS Current Register (HWDG\_PSSULC)". You cannot change the set value of this register after the LOCK bit in the hardware watchdog configuration register (HWDG\_CFG) is set.

BIT_OFFSET	31-0
BIT_NAME	WDGPSSULS
ACCESS_TYPE	R/W
PROT_TYPE	WPS
INITIAL_VALUE	10000000_00000000_00000000_00000000

#### [bit31:0] WDGPSSULS[31:0]: Window upper limit value for PSS set bits

These bits are used to define the window upper limit value for PSS. The reading of these bits returns the set value regardless of the hardware watchdog configuration register (HWDG\_CFG) LOCK bit value.

bit31:0	Description
During write operation	Set value written
During read operation	Set value read

**Note:**

- This register value is different from the window upper limit value that is actually used. The window upper limit value actually used is the value in the hardware watchdog upper limit PSS current register(HWDG\_PSSULC).

## 6.14. Hardware Watchdog Reset Delay Counter Register (HWDG\_RSTDLY)

When performing write access to this register, follow the watchdog register write protection sequence. This register is used to set the number of cycles for the delay time from the occurrence of a watchdog error to the watchdog reset request or watchdog interrupt request (NMI). The reference clock for the delay time is the source clock of the watchdog counter. When the IRQEN bit in the hardware watchdog interrupt configuration register (HWDG\_INT) is "1", this register indicates the number of cycles for the delay time from the prior warning interrupt request to the watchdog reset request or watchdog interrupt request (NMI).

BIT_OFFSET	31-16
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	WPS
INITIAL_VALUE	00000000_00000000

BIT_OFFSET	15-0
BIT_NAME	WDGRSTDLY
ACCESS_TYPE	R0,W
PROT_TYPE	WPS
INITIAL_VALUE	00000000_00000000

[bit31:16] Reserved: Reserved bits

### [bit15:0] WDGRSTDLY[15:0]: Reset/NMI delay counter bits

These bits define the number of cycles for the delay time that is to be inserted before generation of the watchdog reset request or watchdog interrupt request (NMI). The reference clock for this delay time is the source clock of the watchdog counter.

bit15:0	Description
During write operation	Set value written
During read operation	"0x0000" is read.



## 6.15. Hardware Watchdog Configuration Register (HWDG\_CFG)

When performing write access to this register, follow the watchdog register write protection sequence. This register is used to make the operation settings of the hardware watchdog timer.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							LOCK
ACCESS_TYPE	R0,WX							R,W
PROT_TYPE	WPS							
INITIAL_VALUE	0000000							0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved			OBSSEL				
ACCESS_TYPE	R0,WX			R/W				
PROT_TYPE	WPS							
INITIAL_VALUE	000			00000				

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved						CLKSEL	
ACCESS_TYPE	R0,WX						R/W	
PROT_TYPE	WPS							
INITIAL_VALUE	000000						00	

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved					ALLOW STOPCLK	WDENPSS	WDENRUN
ACCESS_TYPE	R0,WX					R,WX	R,WX	R,WX
PROT_TYPE	WPS							
INITIAL_VALUE	00000					X	X	X

**[bit31:25] Reserved: Reserved bits**

### **[bit24] LOCK: Lock bit**

You can write data to this bit only once. This register is used to prevent rewriting of the set values of various registers. When this bit is "0", you can rewrite the setting values of various registers. When this bit is set to "1" the watchdog counter is automatically cleared. This bit is equipped with a majority circuit with 3 FFs as a measure against a bit invert caused by the effects of noise or another factor. This bit is set up by BootROM software.

bit	Description
During write operation	When "0" is written: Invalid When "1" is written: The set values of the registers are locked.
During read operation	When "0" is read: The lock for the register set values is disabled. When "1" is read: The lock for the register set values is enabled.

**[bit23:21] Reserved: Reserved bits**

**[bit20:16] OBSSEL[4:0]: Watchdog counter monitor bit output selection bits**

These bits are used for selecting any 1 bit in the watchdog counter (32 bits) as the output of the watchdog counter monitor bit.

bit20:16	Description
During write operation	<p>When "0b00000" is written: Bit 0 is selected for the monitor output.</p> <p>When "0b00001" is written: Bit 1 is selected for the monitor output.</p> <p>When "0b00010" is written: Bit 2 is selected for the monitor output.</p> <p style="text-align: center;">•</p> <p style="text-align: center;">•</p> <p style="text-align: center;">•</p> <p>When "0b11111" is written: Bit 31 is selected for the monitor output.</p>
During read operation	Set value read

**Note:**

- Output of watchdog counter monitor bit for MCU output pin is unsupported. For this product, any data written to these bits does not affect operation.

**[bit15:10] Reserved: Reserved bits**

**[bit9:8] CLKSEL[1:0]: Clock selection bits**

These bits are used to select the source clock for the watchdog counter. When activated, the watchdog counter starts its operation as the fast-CR clock. For details on clock switching, see Section "(2) Switching of Watchdog Counter Source Clock" in Section "7 Precautions for Using".

bit[9:8]	Description
During write operation	<p>When "0bX0" is written: The fast-CR clock is selected.</p> <p>When "0bX1" is written: The slow-CR clock is selected.</p> <p>Writing data in CLKSEL[1] bits does not affect operation.</p>
During read operation	Set value read

**[bit7:3] Reserved: Reserved bits**

**[bit2] ALLOWSTOPCLK: Clock stop for PSS enable bit**

This bit is used to enable transition to PSS that involves the source clock stop of the watchdog counter. This bit is valid only when the WDENPSS bit in the hardware watchdog configuration register (HWDG\_CFG) is "1". (This bit is fixed by hardware, "1" are always read.)

bit	Description
During write operation	Invalid
During read operation	<p>When "0" is read: Watchdog clock stop in PSS is disabled.</p> <p>When "1" is read: Watchdog clock stop in PSS is enabled.</p>



**[bit1] WDENPSS: Watchdog counter for PSS enable bit**

This bit is used to enable the watchdog counter in PSS. This bit is enabled when the LOCK bit in the hardware watchdog configuration register (HWDG\_CFG) is set to "1". (This bit is fixed by hardware, "0" are always read.)

bit	Description
During write operation	Invalid
During read operation	When "0" is read: The watchdog counter in PSS is disabled. When "1" is read: The watchdog counter in PSS is enabled.

**[bit0] WDENRUN: Watchdog counter for RUN enable bit**

This bit is used to enable the watchdog counter in RUN. This bit is enabled when the LOCK bit in the hardware watchdog configuration register (HWDG\_CFG) is set to "1". (This bit indicates the value of the entire system enable signal.)

bit	Description
During write operation	Invalid
During read operation	When "0" is read: The watchdog counter in RUN is disabled. When "1" is read: The watchdog counter in RUN is enabled.

**Note:**

- When the ALLOWSTOPCLK bit in the hardware watchdog configuration register (HWDG\_CFG) is "1", transition to PSS that involves clock stop is enabled. However, in case of transition from RUN to PSS during reset delay counter operation, the reset delay counter is stopped until a return from PSS to RUN occurs.

## 6.16. Hardware Watchdog Lower Limit RUN Current Register (HWDG\_RUNLLC)

This register is used to read the window lower limit value for RUN that is actually used. This register fetches the set value in the hardware watchdog lower limit RUN setting register (HWDG\_RUNLLS) when the LOCK bit in the hardware watchdog configuration register (HWDG\_CFG) is set. You cannot change the set value of this register after the LOCK bit in the hardware watchdog configuration register (HWDG\_CFG) is set.

BIT_OFFSET	31-0
BIT_NAME	WDGRUNLLC
ACCESS_TYPE	R,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000_00000000

### [bit31:0] WDGRUNLLC[31:0]: Window lower limit for RUN current bits

These bits are used to define the window lower limit value for RUN. The reading of these bits returns the initial value, until the hardware watchdog configuration register (HWDG\_CFG) LOCK bit is set.

bit31:0	Description
During write operation	Invalid
During read operation	Set value read

**Note:**

- An AHB transfer error response is generated where there is write access to this register.





### 6.17. Hardware Watchdog Upper Limit RUN Current Register (HWDG\_RUNULC)

This register is used to read the window upper limit value for RUN that is actually used. This register fetches the set value in the hardware watchdog upper limit RUN setting register (HWDG\_RUNULS) when the LOCK bit in the hardware watchdog configuration register (HWDG\_CFG) is set. Also, you cannot change the set value of this register after the LOCK bit in the hardware watchdog configuration register (HWDG\_CFG) is set.

BIT_OFFSET	31-0
BIT_NAME	WDGRUNULC
ACCESS_TYPE	R,WX
PROT_TYPE	-
INITIAL_VALUE	00000001_00000000_00000000_00000000

#### [bit31:0] WDGRUNULC[31:0]: Window upper limit for RUN current bits

These bits are used to define the window upper limit value for RUN. The reading of these bits returns the initial value, until the hardware watchdog configuration register (HWDG\_CFG) LOCK bit is set.

bit31:0	Description
During write operation	Invalid
During read operation	Set value read

**Note:**

- An AHB transfer error response is generated where there is write access to this register.

## 6.18. Hardware Watchdog Lower Limit PSS Current Register (HWDG\_PSSLLC)

This register is used to read the window lower limit value for PSS that is actually used. This register fetches the set value in the hardware watchdog lower limit PSS setting register (HWDG\_PSSLLS) when the LOCK bit in the hardware watchdog configuration register (HWDG\_CFG) is set. You cannot change the set value of this register after the LOCK bit in the hardware watchdog configuration register (HWDG\_CFG) is set.

BIT_OFFSET	31-0
BIT_NAME	WDGPSSLLC
ACCESS_TYPE	R,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000_00000000

### [bit31:0] WDGPSSLLC[31:0]: Window lower limit for PSS current bits

These bits are used to define the window lower limit value for PSS. The reading of these bits returns the initial value, until the hardware watchdog configuration register (HWDG\_CFG) LOCK bit is set.

bit31:0	Description
During write operation	Invalid
During read operation	Set value read

**Note:**

- An AHB transfer error response is generated where there is write access to this register.



## 6.19. Hardware Watchdog Upper Limit PSS Current Register (HWDG\_PSSULC)

This register is used to read the window upper limit value for PSS that is actually used. This register fetches the set value in the hardware watchdog upper limit PSS setting register (HWDG\_PSSULS) when the LOCK bit in the hardware watchdog configuration register (HWDG\_CFG) is set. You cannot change the set value of this register after the LOCK bit in the hardware watchdog configuration register (HWDG\_CFG) is set.

BIT_OFFSET	31-0
BIT_NAME	WDGPSSULC
ACCESS_TYPE	R,WX
PROT_TYPE	-
INITIAL_VALUE	10000000_00000000_00000000_00000000

### [bit31:0] WDPSSULC[31:0]: Window upper limit for PSS current bits

These bits are used to define the window upper limit value for PSS. The reading of these bits returns the initial value, until the hardware watchdog configuration register (HWDG\_CFG) LOCK bit is set.

bit31:0	Description
During write operation	Invalid
During read operation	Set value read

**Note:**

- Write access to this register generates an AHB transfer error response.

## 7. Precautions for Using

This section describes precautions on the use of the hardware watchdog timer.

### (1) Watchdog Window Setting

For the hardware watchdog timer, you can make 2 types of window setting, the setting for RUN and setting for PSS. The user must make an appropriate window setting for each device state.

When the device transitions from PSS to RUN, the hardware watchdog timer continues operation using the PSS window setting until the watchdog counter clear protection trigger sequence is executed. The watchdog counter clear protection trigger sequence switches the window setting to the RUN window setting. You can control the watchdog function in each device state with the WDENRUN bit and WDENPSS bit in the hardware watchdog configuration register (HWDG\_CFG).

### (2) Switching of Watchdog Counter Source Clock

The system controller controls the switching of the watchdog counter source clock since the source clock control of the watchdog counter is a part of the system setting information (profile). To change the watchdog counter source clock, follow the sequence below.

1. Setting the CLKSEL bit in the hardware watchdog configuration register (HWDG\_CFG) sets the new source clock of the watchdog counter. Then, writing data to the following registers finalizes the window setting.
2. Hardware watchdog lower limit RUN setting register (HWDG\_RUNLLS)
3. Hardware watchdog upper limit RUN setting register (HWDG\_RUNULS)
4. Hardware watchdog lower limit PSS setting register (HWDG\_PSSLLS)
5. Hardware watchdog upper limit PSS setting register (HWDG\_PSSULS)
6. Setting the LOCK bit in the hardware watchdog configuration register (HWDG\_CFG) to "1" locks rewriting of the register set values.
7. Writing "0xAB" to the RUN profile update trigger register of the system controller (SYSC\_TRGRUNCNTR) executes RUN profile update. At this point in time, the source clock of the watchdog counter is switched to the new source clock selected by the CLKSEL bit of the hardware watchdog configuration register (HWDG\_CFG).

### (3) Issuing Software Reset with Slow-CR Clock Selected

If issuing software reset with slow-CR clock selected by the clock of the watchdog timer, perform watchdog clear sequence beforehand.



#### (4) Watchdog Error

A watchdog error is generated under the following conditions. This watchdog error generates the watchdog reset request/interrupt request (NMI).

1. An incorrect value is written to the hardware watchdog trigger 0/1 register (HWDG\_TRG0 or HWDG\_TRG1).
2. The watchdog counter clear protection trigger sequence is violated.
3. The watchdog counter clear protection trigger sequence is not executed before the watchdog counter reaches the window upper limit value.
4. The watchdog counter clear protection trigger sequence is executed before the watchdog counter reaches the window lower limit value.
5. The watchdog counter clear protection trigger sequence is executed while the LOCK bit in the hardware watchdog configuration register (HWDG\_CFG) is still "0".

#### (5) Factors for Bus Error Responses (Data Abort)

A bus error response is generated in the following cases.

- Access (read or write) to an address where no register exists
- Violation of the watchdog register write protection sequence
- Write access to a register with the read-only attribute (indicated as R, WX)
- Write access to a register after the LOCK bit in the hardware watchdog configuration register (HWDG\_CFG) is set to "1"
- Write access with an incorrect value to the hardware watchdog protection register (HWDG\_PROT)
- Byte/Half-word write access to the hardware watchdog protection register (HWDG\_PROT)
- Write access to a register in non-privileged mode
- (Except for the hardware watchdog trigger 0/1 register (HWDG\_TRG0 and HWDG\_TRG1))

#### (6) Watchdog Timer Operation in Standby State of Processor

The hardware watchdog timer operates using the RUN window setting in the following case.

- Without a valid key ("0xBA" or "0xBB") being written to the PSS profile update enable register (SYSC\_PSSSEN.PSSSEN), the processor transitions to the standby state by executing a wait for interrupt (WFI) instruction.

#### (7) Watchdog Timer Operation in PSS

The products stop the hardware watchdog timer operation in PSS by the fixed hardware.



## CHAPTER 20: Software Watchdog Timer

This chapter describes the software watchdog timer.

---

1. Overview
2. Configuration
3. Operation
4. Setting Procedure Example
5. Operation Examples
6. Registers
7. Precautions for Using



## 1. Overview

This section provides an overview of the software watchdog timer.

The software watchdog timer is positioned in the MCU configuration group, used for detecting a runaway state caused by a user program, also the software watchdog timer assigned to the monitored CPU 0 to 1, monitors user program execution for each CPU.

### Software Watchdog Timer Features

- Detecting a runaway state of the software generates a reset request. (\*a)
- A user program starts the software watchdog timer. (\*a)
- The watchdog counter source clock is selected from 3 types. (\*a)
- (Fast-CR clock (default), slow-CR clock, or main clock)
- A user program is used to set registers. (\*a)
- It operates as a 32-bit watchdog counter.
- Different initializations between soft reset and hard reset can be applied.
- Soft reset clears only the watchdog counter.
- Hard reset initializes the watchdog counter, register setting, and internal circuits.
- It monitors the watchdog counter clear protection trigger sequence.
- The window watchdog function is implemented.
- Different operation settings are possible for each device state (RUN and PSS).
- It monitors the watchdog register write protection sequence.
- The majority circuits for bits that affect important functions are implemented.
- It can generate the watchdog reset request or watchdog interrupt request (NMI) in response to a watchdog error.
- It can generate a prior warning interrupt request before the reset request or interrupt request (NMI).
- The watchdog counter can be stopped in the debugging state of the processor.

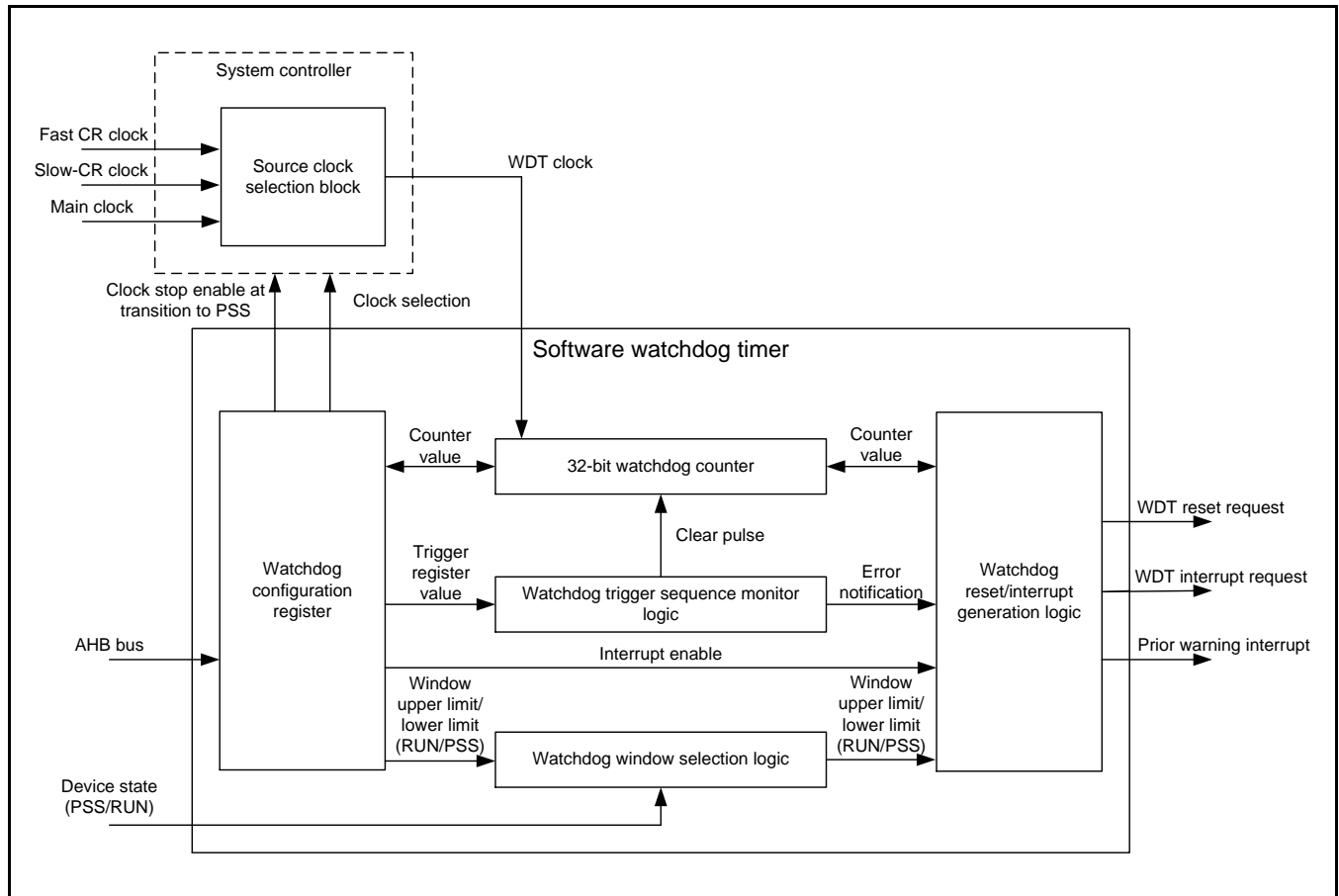
### Notes:

- The notation (\*a) indicates a feature that is specific to the software watchdog timer compared with the hardware watchdog.
- A watchdog error is a factor that causes the watchdog reset request or watchdog interrupt request (NMI).

## 2. Configuration

This section describes the block diagrams of the software watchdog timer.

**Figure 2-1 Software Watchdog Timer Block Diagram**



- Watchdog configuration register:  
In this block, the set value of each register is stored. I/O operations for information including the counter value for read operation, trigger register value necessary for clearing the counter, and window upper limit/lower limit values are performed.
- 32-bit watchdog counter:  
This is the block of the 32-bit up counter.
- Watchdog window selection logic:  
This block fetches the window upper limit value and lower limit value for each device state (RUN and PSS) and selects a window setting according to the device state transition.
- Watchdog reset/interrupt generation logic:  
This block generates the watchdog reset request, watchdog interrupt request (NMI), and prior warning interrupt request.





### 3. Operation

This section describes operation of the software watchdog timer.

#### (1) Software Watchdog Timer Functions

The software watchdog timer implements the following functions.

##### a) Reset Request Generation by Detection of Runaway State of Software

The software watchdog timer monitors user program execution by the CPU. The timer starts the monitoring when it is activated by a user program. If the watchdog counter is not cleared within the range between the upper limit value and the lower limit value of the window specified beforehand, this state is judged to be a runaway state of a user program. In this case, a reset request or an interrupt request (NMI) is generated.

##### Note:

- For example, after the watchdog timer is activated by a user program, it can detect a software runaway state in which a user program goes out of control as a result of an abnormal change in the program counter (PC).

##### b) Software Watchdog Timer Start by User Program

A user program starts the software watchdog timer. Specifically, operation is started by setting the LOCK bit in the software watchdog configuration register (SWDGN\_CFG).

##### c) Corresponding Software Watchdog Timer Control by Monitored CPU 0 to 1

The software watchdog timer is assigned to the monitored CPU 0 to 1, and user program execution is monitored for each CPU. The number of CPUs to be mounted in the MCU and the number of software watchdog timers are the same.

##### d) Watchdog Counter Source Clock Selection (3 types)

The software watchdog timer sets values in the CLKSEL[1:0] bits in the software watchdog configuration register (SWDGN\_CFG) to select a source clock for the watchdog counter from among 3 types of source clocks. Specifically, select a source clock from the fast-CR clock, slow-CR clock, or main clock. (The initial setting is for the fast-CR clock.)

Selection of a source clock is a clock request to the system controller. Source clock switching is performed on the clock system side of the system controller. For details on the procedure, see "(2) Switching of Watchdog Counter Source Clock" in Section "7. Precautions for Using."

##### e) Register Setting by User Program

Use a user program to set software watchdog timer registers.

#### f) 32-bit Watchdog Counter

The software watchdog timer operates by using the 32-bit watchdog counter (up counter) (the initial value is "0x0000\_0000").

The watchdog counter operates when all the following conditions are met.

1. The device is operating in user mode.
2. The watchdog counter has not reached the window upper limit value.
3. The device state is RUN and the WDENRUN bit is "1".

(See Section "6.15. Software Watchdog Configuration Register (SWDGn\_CFG)".)

4. The device state is PSS and the WDENPSS bit is "1".

(See Section "6.15. Software Watchdog Configuration Register (SWDGn\_CFG)".)

**Note:**

- When the watchdog counter reaches the window upper limit value, the count operation stops.

The following Table 3-1 shows the relationship between the window upper limit value and count time.

**Table 3-1 Relationship between Software Window Upper Limit Value of Watchdog Timer and Count Time**

Input Clock Frequency	Window Upper Limit Value	Count Time	Remarks
8MHz	"0x0100_0000"	About 2.1 s	It operates at the initial value of the software watchdog upper RUN setting register (SWDGn_RUNULS).
12MHz	"0x0100_0000"	About 1.4 s	
8MHz	"0x8000_0000"	About 268 s	It operates at the initial value of the software watchdog upper PSS setting register (SWDGn_PSSULS).
12MHz	"0x8000_0000"	About 178 s	

**Note:**

- The above table is an example involving fast-CR clock frequencies of 8MHz and 12MHz.

#### g) Different Initializations between Soft Reset and Hard Reset

For the software watchdog timer, different initializations between soft reset and hard reset can be applied. The following Table 3-2 shows the ranges of initialization for soft reset and hard reset.

**Table 3-2 Ranges of Initialization for Soft Reset and Hard Reset**

Conditions	Reset Name	Range of Initialization
1	Soft reset	Watchdog counter
2	Hard reset	Watchdog counter and settings for all registers



#### h) Watchdog Counter Clear Protection Trigger Sequence

The software watchdog timer monitors the watchdog counter clear protection trigger sequence to clear the watchdog counter.

The watchdog counter clear protection trigger sequence must meet the following conditions.

- Write data to the software watchdog trigger 1 register (SWDGn\_TRG1) after writing data to the software watchdog trigger 0 register (SWDGn\_TRG0).
- Do not execute the watchdog counter clear protection trigger sequence when the LOCK bit in the software watchdog configuration register (SWDGn\_CFG) is "0".
- The value to be written to the software watchdog trigger 0 register (SWDGn\_TRG0) must match the software watchdog trigger 0 configuration register (SWDGn\_TRG0CFG).
- The value to be written to the software watchdog trigger 1 register (SWDGn\_TRG1) must match the software watchdog trigger 1 configuration register (SWDGn\_TRG1CFG).
- The watchdog counter must already have reached the window lower limit value when the watchdog counter clear protection trigger sequence is completed.

**Note:**

- For details on the watchdog counter clear protection trigger sequence, see Figure 3-2.

#### i) Window Watchdog Function

The window watchdog function is a function for setting a range in which the watchdog counter value can be cleared with upper limit and lower limit values. The following shows the range in which the watchdog counter can be cleared for 2 set values.

$$(\text{Window lower limit value}) \leq (\text{watchdog counter}) < (\text{window upper limit value})$$

For example, suppose that a runaway state of a user program occurs and the watchdog counter clear protection trigger sequence is executed continuously. In this case, you can set a value other than "0x00000000" as a window lower limit value to detect the continuous watchdog counter clear as an abnormal operation.

The window setting for RUN is defined with the following 2 registers.

Software watchdog lower limit RUN current register (SWDGn\_RUNLLC)

Software watchdog upper limit RUN current register (SWDGn\_RUNULC)

The window setting for PSS is defined with the following 2 registers.

Software watchdog lower limit PSS current register (SWDGn\_PSSLLC)

Software watchdog upper limit PSS current register (SWDGn\_PSSULC)

**j) Different Operation Settings for each Device State (RUN and PSS)**

The software watchdog timer can make different operation settings for each device state (RUN and PSS). The operation setting is switched to the RUN or PSS operation setting according to the device state transition.

**RUN operation setting**

- When the LOCK bit and WDENRUN bit in the software watchdog configuration register (SWDGN\_CFG) are set to "1", operation is performed by using the window setting for RUN.
- When the watchdog counter is not cleared in the range between the window upper limit value and lower limit value, or there is a violation in the watchdog counter clear protection trigger sequence, a watchdog error is detected. The watchdog reset request or watchdog interrupt request (NMI) is generated for this watchdog error. (For details on watchdog errors, see "(4) Watchdog Error(4) Watchdog Error

**PSS operation setting**

- The window setting is switched to the window setting for PSS when the device state transitions from RUN to PSS. When the WDENPSS bit in the software watchdog configuration register (SWDGN\_CFG) is set to "1", operation is performed by using the window setting for PSS.
- When the device state transitions from PSS to RUN, the enable bit is switched from the WDENPSS bit in the software watchdog configuration register (SWDGN\_CFG) to the WDENRUN bit in the same register. However, the window setting is not switched immediately. Use of the window setting for PSS continues until the watchdog counter clear protection trigger sequence is executed.

See Figure 3-4 and Figure 3-5 for the details on the watchdog operation in RUN and PSS, respectively.



#### k) Watchdog Register Write Protection Sequence

The software watchdog timer monitors the watchdog register write protection sequence to write a setting value in the register.

The watchdog register write protection sequence must meet the following conditions.

- It writes a setting value in the register after writing data in the software watchdog protection register (SWDGn\_PROT).
- It writes data in the software watchdog protection register (SWDGn\_PROT) in privileged mode.
- The value written in the software watchdog protection register (SWDGn\_PROT) must match the lock release key for register write protection "0xEDAC\_CE55".
- It does not write a setting value in any register in another module after writing data in the software watchdog protection register (SWDGn\_PROT).
- It does not write data in the software watchdog protection register (SWDGn\_PROT) twice in a row.
- It writes a value in the register when the LOCK bit in the software watchdog configuration register (SWDGn\_CFG) is "0" and the mode is privileged mode.

#### Notes:

- For details on the watchdog register write protection sequence, see Figure 3-1.
- Write the following registers when LOCK bit=0. Write the following registers when LOCK bit is 1 and register write protection is valid, bus error response is generated.
  - Software watchdog reset factor register
  - Software watchdog interrupt configuration register
  - Software watchdog interrupt clear register
  - Software watchdog trigger 0 configuration register
  - Software watchdog trigger 1 configuration register
  - Software watchdog lower limit RUN setting register
  - Software watchdog upper limit RUN setting register
  - Software watchdog lower limit PSS setting register
  - Software watchdog upper limit RUN setting register
  - Software watchdog reset delay counter register
  - Software watchdog configuration register

**Table 3-3 Key Value for Register Programming**

Key Value	Target Register	Purpose
"0xEDAC_CE55"	SWDGn_PROT:KEY	Releasing protection lock for register write

#### Note:

- Setting the LOCK bit in the software watchdog configuration register (SWDGn\_CFG) to "1" prevents the following registers from being rewritten.
  - Software watchdog interrupt configuration register (SWDGn\_INT)
  - Software watchdog trigger 0 configuration register (SWDGn\_TRG0CFG)
  - Software watchdog trigger 1 configuration register (SWDGn\_TRG1CFG)
  - Software watchdog reset delay counter register (SWDGn\_RSTDLY)
  - Software watchdog configuration register (SWDGn\_CFG)

### I) Majority Circuits for Bits that Affect Important Functions

The software watchdog timer has majority circuits for bits that affect important functions. The majority circuit consists of 3 FFs. When the bit in an FF bit is inverted due to noise or because of another factor, the correct value can be selected by majority decision among the 3 FFs.

The bits that are equipped with majority circuits are as follows.

- RSTEN bit                      \*1
- LOCK bit                        \*2
- WDENPSS bit                  \*2
- WDENRUN bit                  \*2

#### Notes:

- The notation \*1 indicates a bit in the software watchdog interrupt configuration register (SWDGn\_INT).
- The notation \*2 indicates a bit in the software watchdog configuration register (SWDGn\_CFG).



#### m) Generation of Watchdog Reset Request or Watchdog Interrupt Request (NMI)

The software watchdog timer generates the watchdog reset request or watchdog interrupt request (NMI) when detecting a watchdog error. Use the watchdog interrupt request (NMI) as a test function. (For example, you can use the watchdog interrupt request (NMI) to set a break point at the beginning of an interrupt handler.)

Also, the software watchdog timer can generate the prior warning interrupt request before the watchdog reset request (or watchdog interrupt request (NMI)). (For example, you can use the prior warning interrupt request so that the processor can save important data to the memory before the watchdog reset request is generated.)

The software watchdog timer generates these requests by using the following procedure.

- When a watchdog error occurs, the IRQFLAG bit in the software watchdog interrupt configuration register (SWDGN\_INT) is set. At this time, if the IRQEN bit in the software watchdog interrupt configuration register (SWDGN\_INT) is "1", the prior warning interrupt request is generated.
- As many cycles as are set in the software watchdog reset delay counter register (SWDGN\_RSTDLY) are inserted between the prior warning interrupt request and the watchdog reset interrupt request or watchdog interrupt request (NMI) as a delay time.
- Once the prior warning interrupt request is generated, the watchdog counter cannot be cleared with the watchdog counter clear protection trigger sequence. Also, if a new watchdog error occurs, it is ignored.
- After the elapse of the time corresponding to the number of cycles for delay time set in the software watchdog reset delay counter register (SWDGN\_RSTDLY), the watchdog reset request or watchdog interrupt request (NMI) is generated. If the software watchdog reset delay counter register (SWDGN\_RSTDLY) is "0x0000", no delay time occurs.

#### Notes:

- *When you generate the prior warning interrupt request, set the software watchdog reset delay counter register (SWDGN\_RSTDLY) to a value other than "0x0000".*
- *The use of the watchdog interrupt request (NMI) is prohibited except when used as a test function. (This is prohibited because if such use is allowed, security will be compromised during the execution of an application.)*
- *Insertion of a number of cycles equal to that for the delay time set in the software watchdog reset delay counter register (SWDGN\_RSTDLY) is valid only for 1 watchdog reset request or watchdog interrupt request (NMI). (It remains invalid until the next hard reset occurs.)*

#### n) Watchdog Counter Stop in Debugging State of Processor

The watchdog counter stops when the processor enters the debugging state. When the processor returns from the debugging state, the watchdog counter resumes operation from the point where it had stopped.

#### Note:

- *For the definition of the debugging state of the processor, see the material from ARM® (Cortex™-R5 Revision:r1p2 Technical Reference Manual (ARM DDI 0460D)).*

## (2) Differences between Software Watchdog and Hardware Watchdog

The following Table 3-4 shows the main differences between the software watchdog and hardware watchdog.

**Table 3-4 Differences between Software Watchdog and Hardware Watchdog**

Item	Software Watchdog	Hardware Watchdog
Watchdog Counter	32-bit up counter	Same as on left
Different Initializations between Soft Reset and Hard Reset	Supported	Same as on left
Trigger Sequence for Watchdog Counter Clear	Supported	Same as on left
Window Watchdog Function	Supported	Same as on left
Different Operation Settings for Each Device State (RUN and PSS)	Supported	Same as on left
Watchdog Counter Source Clock Selection	3 types - Fast-CR clock - Slow-CR clock - Main clock	2 types - Fast-CR clock - Slow-CR clock
Watchdog Register Write Protection Sequence	Supported (However, prevention of rewriting of the window setting by the LOCK bit is not supported.)	Same as on left (However, prevention of rewriting of the window setting by the LOCK bit is supported.)
Majority Circuits for Bits That Affect Important Functions	Supported (4 target bits)	Same as on left (2 target bits)
Reset Request or Interrupt Request (NMI) Generation	Supported	Same as on left
Prior Warning Interrupt Request Generation	Supported	Same as on left
Watchdog Counter Stop in the Debugging State of the Processor	Supported	Same as on left
Watchdog Error Conditions	5 types	Same as on left
Bus Error Response Conditions	8 types	Same as on left
Watchdog Timer Start	Register write by a user program	Hard reset clear
Register Setting Value Write	User program	BootROM program
Watchdog Counter Enable Bit Control	Register write by a user program	Stop only in PSS (Fixed hardware)

**Note:**

- Unlike the case of the software watchdog timer, for the hardware watchdog timer, the BootROM program sets registers during the initial operation setting. (Write and read operation from the CPU can also be performed.)



Figure 3-1 Setting Procedure for Watchdog Register Write Protection Sequence

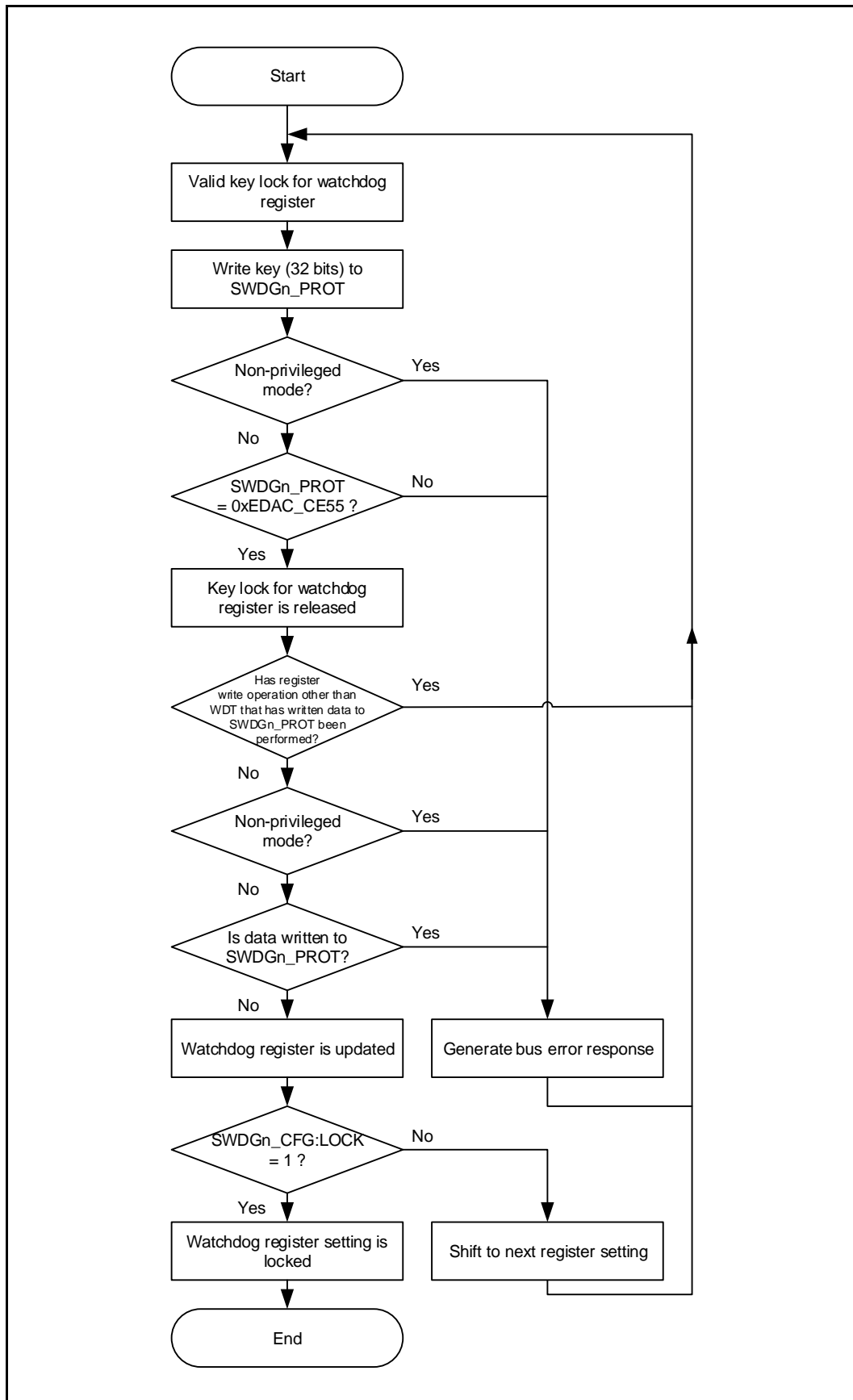
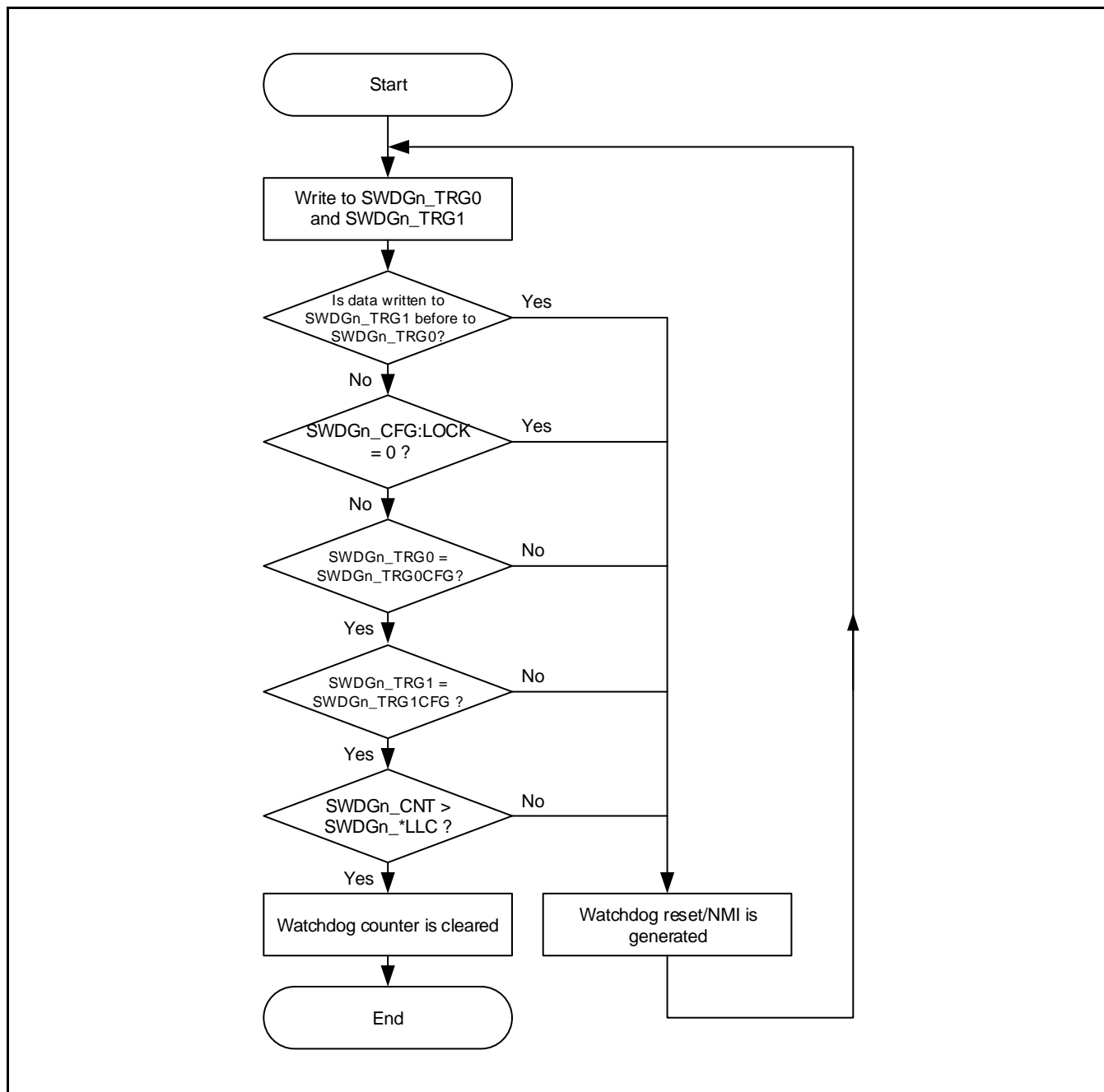


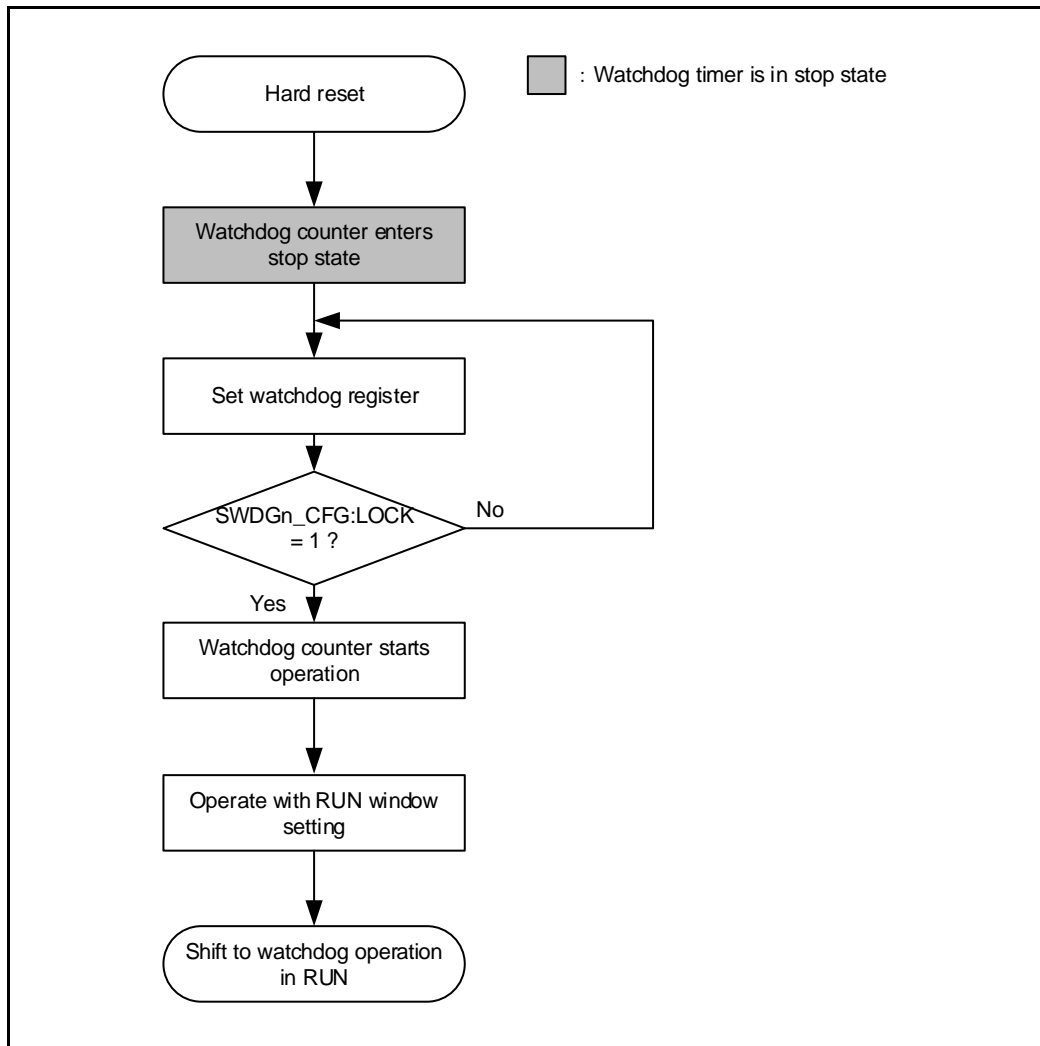
Figure 3-2 Setting Procedure for Watchdog Counter Clear Protection Trigger Sequence



**Note:**

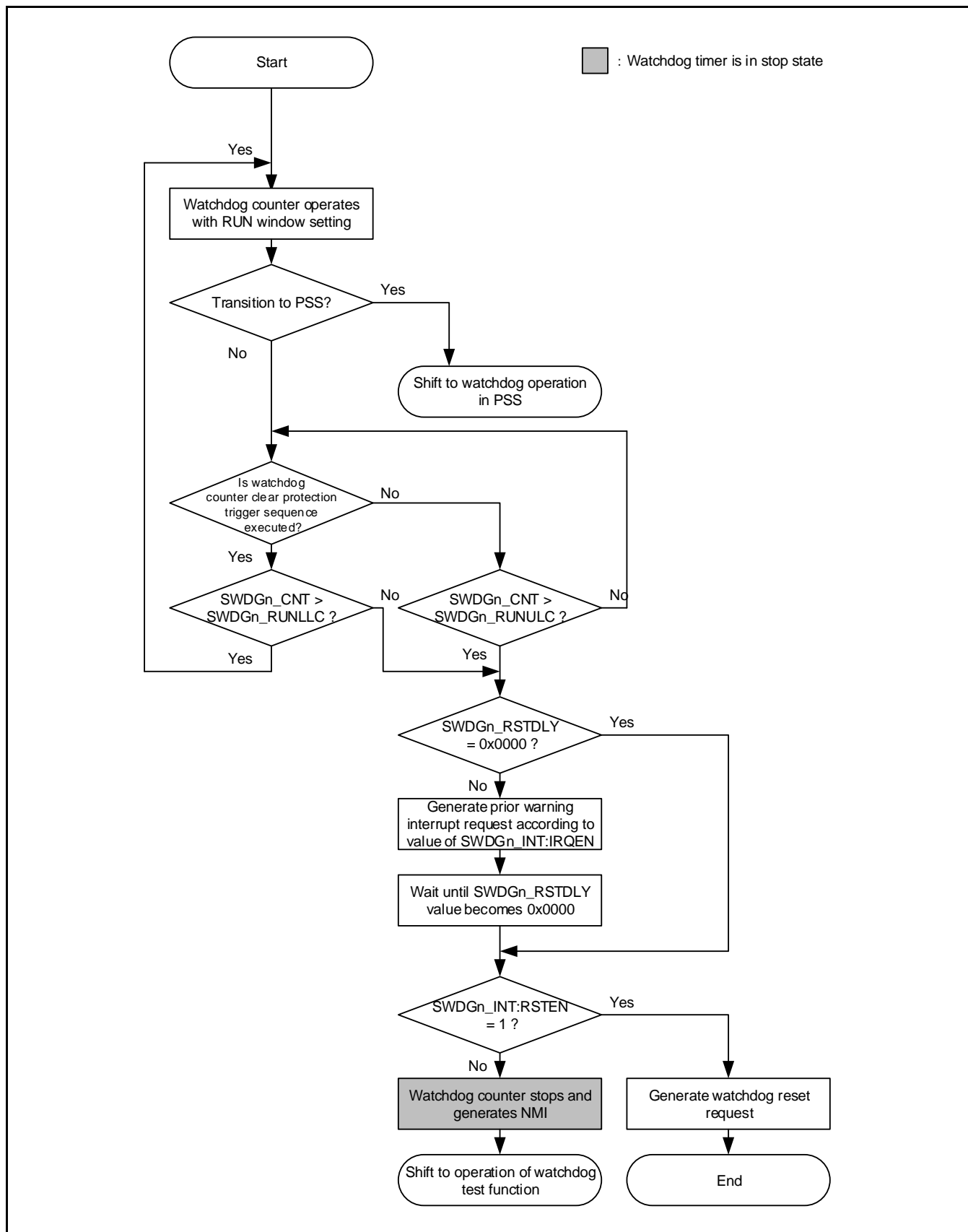
- The notation \* in Figure 3-2 is replaced with device state RUN or PSS.

Figure 3-3 Startup of Software Watchdog Timer



1. The hard reset initializes the software watchdog timer. Then, immediately after the hard reset is cleared, the stop state is indicated.
2. You can change the register setting as long as the LOCK bit in the software watchdog configuration register (SWDGn\_CFG) is "0".
3. Setting "1" in the LOCK bit in the software watchdog configuration register (SWDGn\_CFG) causes the watchdog counter to start up-counting.

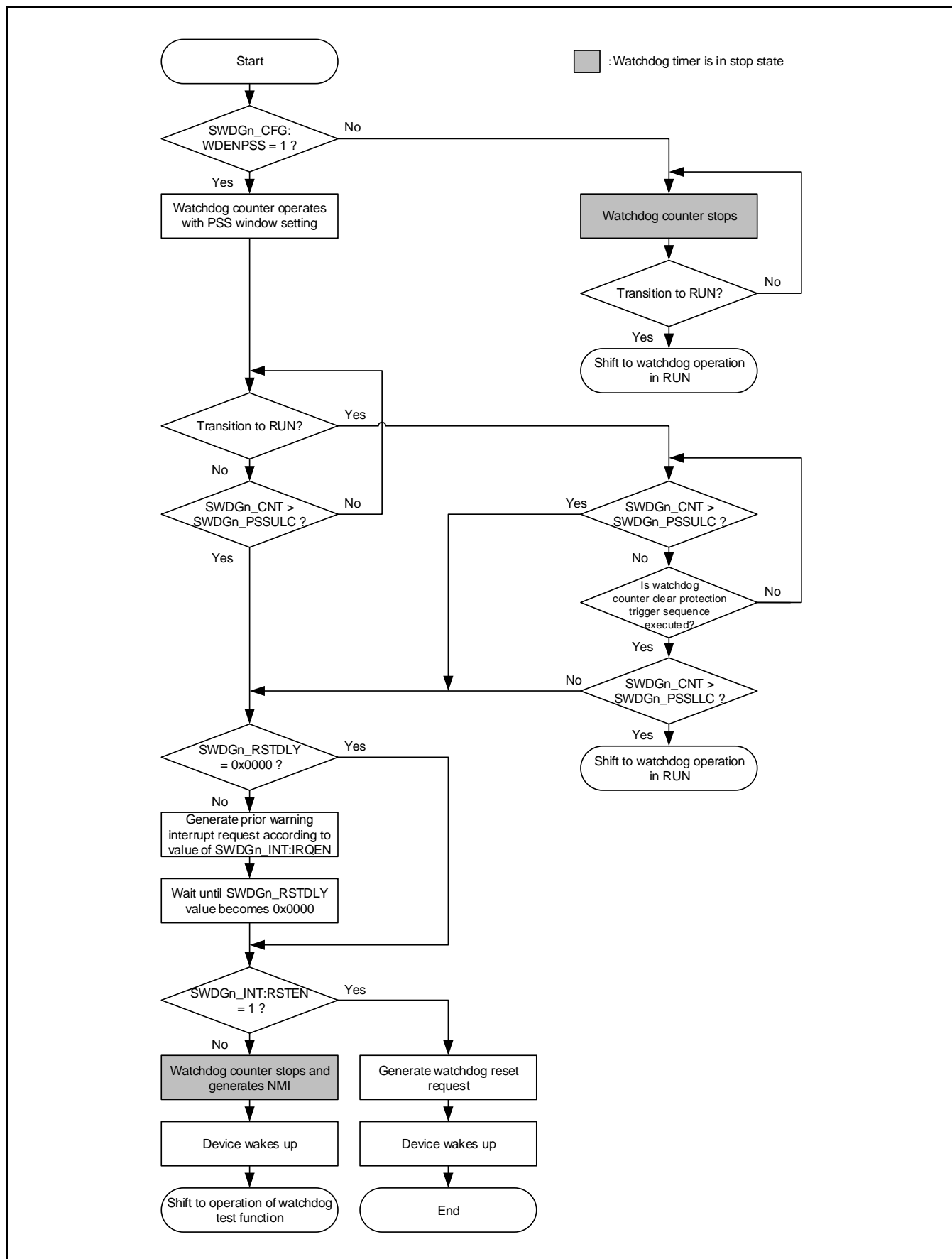
Figure 3-4 Software Watchdog Operation in RUN





1. The watchdog counter in RUN always operates.
2. When the device state transitions from RUN to PSS, the processing transitions to the operation of the watchdog in PSS. For details, see Figure 3-5.
3. During the operation of the watchdog counter, execute the watchdog counter clear protection trigger sequence within the range between the window upper limit value and lower limit value that have been specified beforehand. Do this to clear the watchdog counter periodically.
4. If a runaway state of the software occurs and prevents periodic clearing, the watchdog counter reaches the window upper limit value. Then, processing transitions to the flow that generates the watchdog reset request or watchdog interrupt request (NMI).
5. According to the value of the IRQEN bit in the software watchdog interrupt configuration register (SWDGn\_INT), the prior warning interrupt request is generated. At the same time, as many cycles as are set for the delay time in the software watchdog reset delay counter register (SWDGn\_RSTDLY) are inserted.
6. After the elapse of the time corresponding to the number of cycles for the delay time, the watchdog reset request or watchdog interrupt request (NMI) is generated according to the value of the RSTEN bit in the software watchdog interrupt configuration register (SWDGn\_INT). When the watchdog interrupt request (NMI) is generated, the processing transitions to the operation of the test function of the watchdog. For details, see Figure 3-6.

Figure 3-5 Software Watchdog Operation in PSS



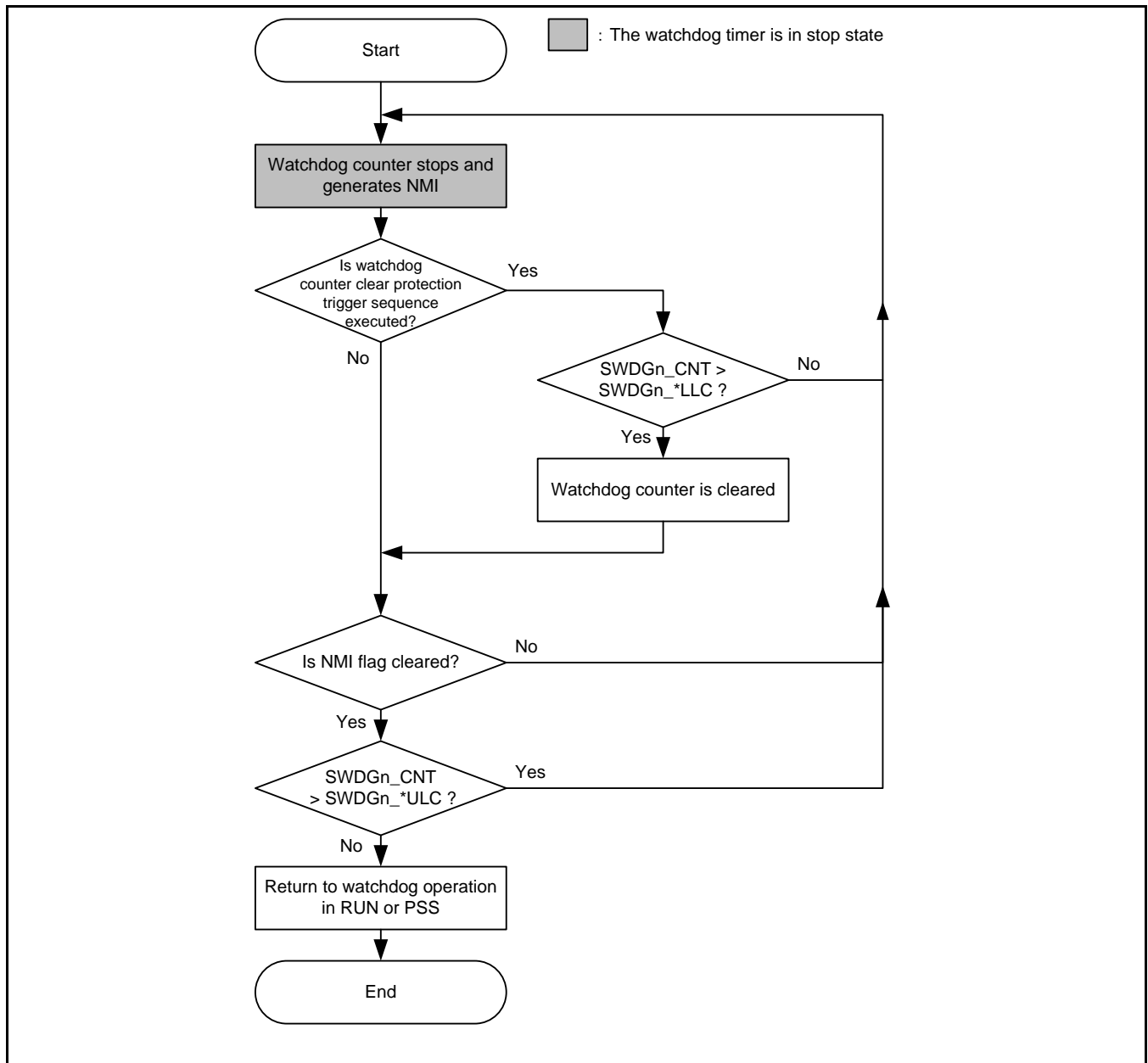


1. The watchdog counter in PSS operates or stops according to the value of the WDENPSS bit in the software watchdog configuration register (SWDGn\_CFG).
2. When the device state transitions from PSS to RUN, the watchdog counter operation is started immediately. However, the PSS window setting continues to be used until the watchdog counter clear protection trigger sequence is executed.
3. During the operation of the watchdog counter, execute the watchdog counter clear protection trigger sequence within the range between the window upper limit value and lower limit value that have been specified beforehand. Continuously clearing the watchdog counter needs to be done periodically.
4. If a runaway state of the software occurs and prevents periodic clearing, the watchdog counter reaches the window upper limit value. Then, processing transitions to the flow that generates the watchdog reset request or watchdog interrupt request (NMI).
5. According to the value of the IRQEN bit in the software watchdog interrupt configuration register (SWDGn\_INT), the prior warning interrupt request is generated. At the same time, as many cycles as are set for the delay time in the software watchdog reset delay counter register (SWDGn\_RSTDLY) are inserted.
6. After the elapse of the time corresponding to the number of cycles for the delay time, the watchdog reset request or watchdog interrupt request (NMI) is generated according to the value of the RSTEN bit in the software watchdog interrupt configuration register (SWDGn\_INT). When the watchdog interrupt request (NMI) is generated, the processing transitions to the operation of the test function of the watchdog. For details, see Figure 3-6.

**Note:**

- *If the watchdog counter clear protection trigger sequence is executed during operation with the PSS window setting, this means that the device state has already returned from PSS to RUN.*

Figure 3-6 Operation of Software Watchdog Test Function



1. The watchdog counter is in the stop state.
2. If the watchdog counter clear protection trigger sequence is executed in the stop state of the watchdog counter and the window lower limit value has already been reached at this time, the watchdog counter is cleared. (If this occurs outside the counter range, no special change occurs.)
3. When the NMIFLAG bit in the software watchdog interrupt configuration register (SWDGn\_INT) is cleared, the watchdog interrupt request (NMI) is cleared.
4. When the watchdog counter has not reached the window upper limit value, processing returns to operation in RUN or PSS. When it has already reached the window upper limit value, no special change occurs.

**Note:**

- The notation \* in Figure 3-6 is replaced with device state RUN or PSS.

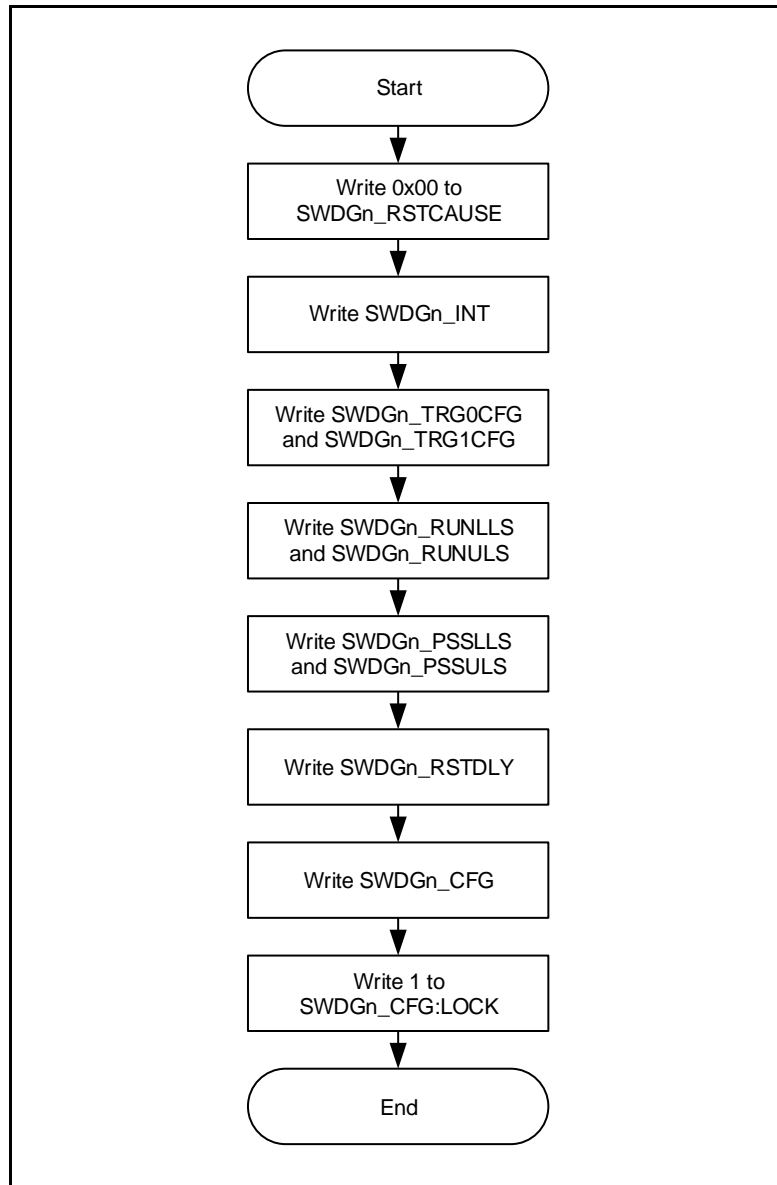




## 4. Setting Procedure Example

This section explains an example of a procedure for setting the software watchdog timer.

**Figure 4-1 Example of Software Watchdog Timer Register Setting Procedure**



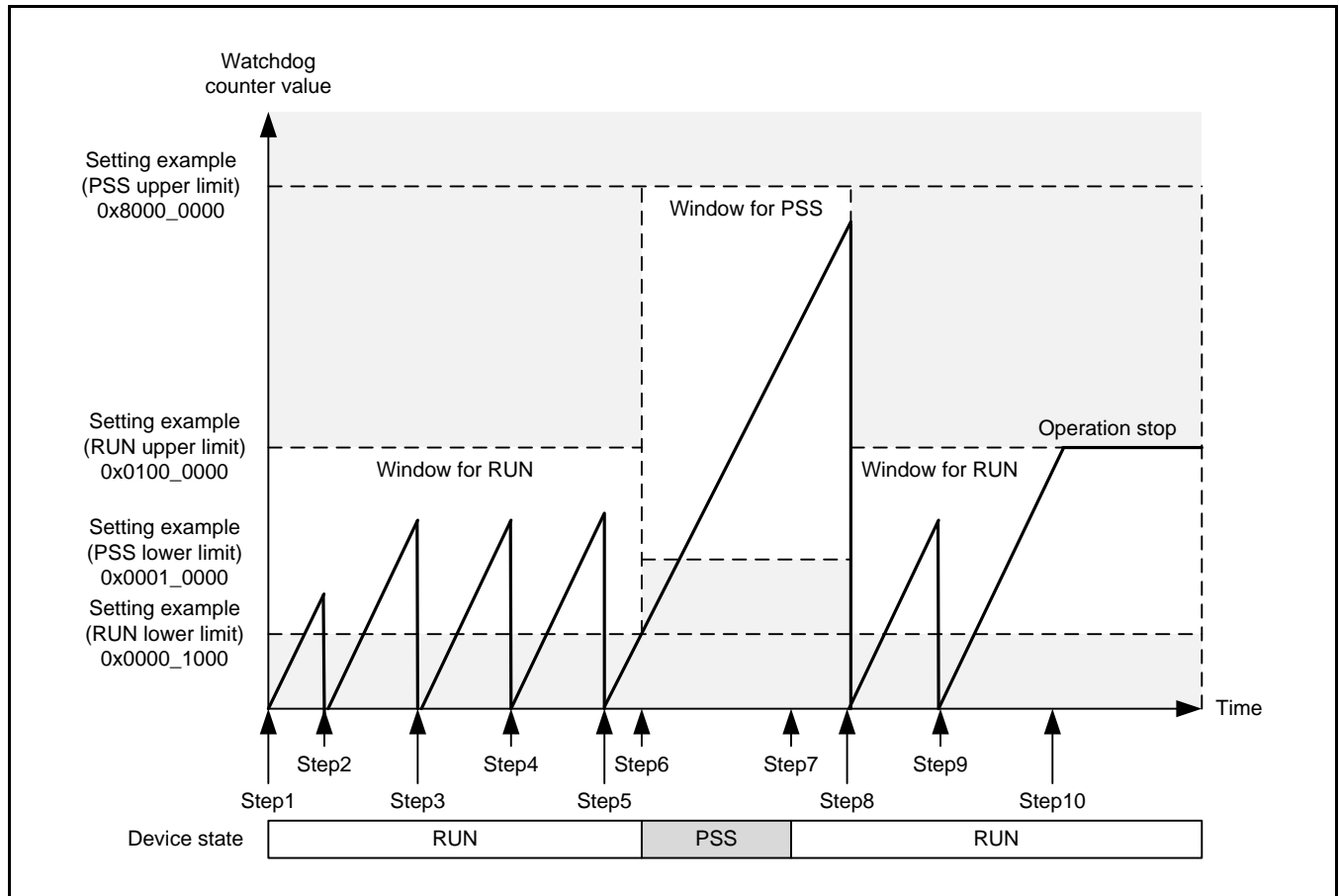
**Note:**

- In Figure 4-1, the setting order can be changed except for the step of setting the LOCK bit in the software watchdog configuration register (SWDGn\_CFG) to "1".

## 5. Operation Examples

This section describes examples of operating the software watchdog timer.

**Figure 5-1 Example of Operating Software Watchdog Timer (When Operation in PSS is Enabled)**



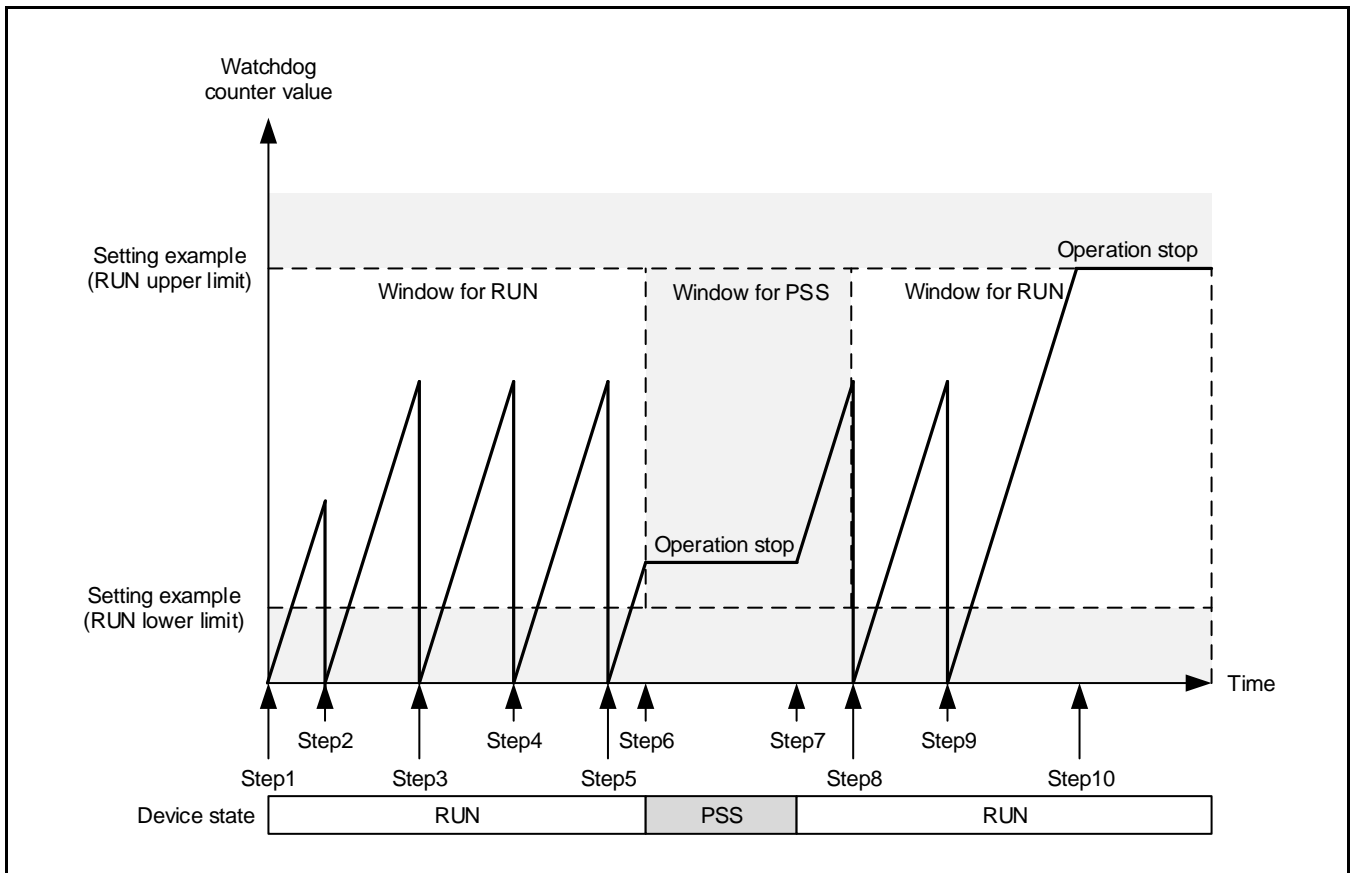
1. Before starting the software watchdog timer, the setting values of the registers are written.
2. The LOCK bit in the software watchdog configuration register (SWDGN\_CFG) is set to "1" to start the software watchdog timer. At this time, it starts up-counting from "0x00000000".
3. The watchdog counter clear protection trigger sequence is executed, and the watchdog counter is cleared.
4. The watchdog counter clear protection trigger sequence is executed, and the watchdog counter is cleared.
5. The watchdog counter clear protection trigger sequence is executed, and the watchdog counter is cleared.
6. The device state transitions from RUN to PSS. At this time, switching to the PSS window setting occurs.
7. The device state returns from PSS to RUN. At this time, the switching to the RUN window setting does not occur immediately.
8. The watchdog counter clear protection trigger sequence is executed, and the watchdog counter is cleared. At this time, the switching to the RUN window setting occurs.
9. The watchdog counter clear protection trigger sequence is executed, and the watchdog counter is cleared.
10. When the watchdog counter reaches the window upper limit value, the watchdog reset request or watchdog interrupt request (NMI) is generated.



**Notes:**

- *The window setting is not immediately switched to the window setting for RUN when the device state is returned from PSS to RUN.*
- *Before the transition from RUN to PSS, be sure to clear the watchdog counter.*

Figure 5-2 Example of Operating Software Watchdog Timer (When Operation in PSS is Disabled)



1. Before starting the software watchdog timer, the setting values of the registers are written.
2. The LOCK bit in the software watchdog configuration register (SWDGN\_CFG) is set to "1" to start the software watchdog timer. At this time, it starts up-counting from "0x00000000".
3. The watchdog counter clear protection trigger sequence is executed, and the watchdog counter is cleared.
4. The watchdog counter clear protection trigger sequence is executed, and the watchdog counter is cleared.
5. The watchdog counter clear protection trigger sequence is executed, and the watchdog counter is cleared.
6. The device state transitions from RUN to PSS. At this time, the software watchdog timer stops operation.
7. The device state returns from PSS to RUN. At this time, the software watchdog timer immediately starts operation.
8. The watchdog counter clear protection trigger sequence is executed, and the watchdog counter is cleared.
9. The watchdog counter clear protection trigger sequence is executed, and the watchdog counter is cleared.
10. When the watchdog counter reaches the window upper limit value, the watchdog reset request or watchdog interrupt request (NMI) is generated.



## 6. Registers

This section explains the registers of the software watchdog timer.

**Table 6-1 List of Software Watchdog Timer Registers**

Abbreviated Register Name	Register Name	See
SWDGn_PROT	Software Watchdog Protection Register	6.1
SWDGn_CNT	Software Watchdog counter Register	6.2
SWDGn_RSTCAUSE	Software Watchdog Reset Source Register	6.3
SWDGn_TRG0	Software Watchdog Trigger 0 Register	6.4
SWDGn_TRG1	Software Watchdog Trigger 1 Register	6.5
SWDGn_INT	Software Watchdog Interrupt Configuration Register	6.6
SWDGn_INTCLR	Software Watchdog Interrupt Clear Register	6.7
SWDGn_TRG0CFG	Software Watchdog Trigger 0 Configuration Register	6.8
SWDGn_TRG1CFG	Software Watchdog Trigger 1 Configuration Register	6.9
SWDGn_RUNLLS	Software Watchdog Lower Limit RUN Setting Register	6.10
SWDGn_RUNULS	Software Watchdog Upper Limit RUN Setting Register	6.11
SWDGn_PSSLLS	Software Watchdog Lower Limit PSS Setting Register	6.12
SWDGn_PSSULS	Software Watchdog Upper Limit PSS Setting Register	6.13
SWDGn_RSTDLY	Software Watchdog Reset Delay Counter Register	6.14
SWDGn_CFG	Software Watchdog Configuration Register	6.15
SWDGn_RUNLLC	Software Watchdog Lower Limit RUN Current Register	6.16
SWDGn_RUNULC	Software Watchdog Upper Limit RUN Current Register	6.17
SWDGn_PSSLLC	Software Watchdog Lower Limit PSS Current Register	6.18
SWDGn_PSSULC	Software Watchdog Upper Limit PSS Current Register	6.19

**Note:**

- The notation *n* indicates the CPU number of a monitored (0 to 1). Since the software watchdog timer is assigned to each of the monitored CPUs, for each type of these registers, there are as many registers as there are CPUs.

## 6.1. Software Watchdog Protection Register (SWDGn\_PROT)

This register is used to execute the watchdog register write protection sequence. Write the correct key ("0xEDAC\_CE55") to this register before performing write access to each register. (However, the software watchdog trigger 0/1 register (SWDGn\_TRG0 and SWDGn\_TRG1) are excluded. ) Writing the correct key releases the write protection lock for subsequent register access. Write access to each register enables write protection lock for the registers. (Read access to each register does not affect the lock. ) For write access to this register, you must write 32-bit data.

BIT_OFFSET	31-0
BIT_NAME	KEY
ACCESS_TYPE	R,W
PROT_TYPE	WP
INITIAL_VALUE	00000000_00000000_00000000_00000000

### [bit31:0] KEY[31:0]: Protection bits

bit31:0	Description
During write operation	<p>When "0xEDAC_CE55" is written: Protection lock for register write is released.</p> <p>When a value other than "0xEDAC_CE55" is written: Protection lock for register write is not released.</p>
During read operation	<p>When "0xFFFF_FFFF" is read: Protection lock for register write has been released.</p> <p>When "0x0000_0000" is read: Protection lock for register write has been enabled.</p>

#### Notes:

- An AHB transfer error response is generated when write access to this register is performed under any of the conditions below. (This error response invokes the CPU exception handler.)
  - A wrong key is written to this register.
  - Non-32-bit data is written to this register.
  - Data is written to this register twice in a row.

For details on the watchdog register write protection sequence, see Figure 3-1.

A protect key will be locked again by writing to the address where is in the same group area (MCU Config Group), however protect key will not be locked by writing to no protect target register.



## 6.2. Software Watchdog Counter Register (SWDGN\_CNT)

This register indicates the current up count value of the watchdog counter.

BIT_OFFSET	31-0
BIT_NAME	WDGCNT
ACCESS_TYPE	R,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000_00000000

### [bit31:0] WDGCNT[31:0]: Watchdog counter bits

bit31:0	Description
During write operation	Invalid
During read operation	Returns the current watchdog counter value. The watchdog counter value is sampled by using the bus clock.

**Note:**

- An AHB transfer error response is generated where there is write access to this register.

### 6.3. Software Watchdog Reset Factor Register (SWDGn\_RSTCAUSE)

When performing write access to this register, follow the watchdog register write protection sequence. This register is the status register that indicates the factor of the watchdog reset request or watchdog interrupt request (NMI). This register is not initialized by reset. To check the factor of the watchdog reset request, read this register.

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	WPS
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved			RST CAUSE4	RST CAUSE3	RST CAUSE2	RST CAUSE1	RST CAUSE0
ACCESS_TYPE	R0,WX			R,W	R,W	R,W	R,W	R,W
PROT_TYPE	WPS							
INITIAL_VALUE	000			X	X	X	X	X

**[bit31:5] Reserved: Reserved bits**

#### **[bit4] RSTCAUSE4: Reset factor bit 4**

If the watchdog counter clear protection trigger sequence is executed when the LOCK bit in the software watchdog configuration register (SWDGn\_CFG) is "0", this bit is set to "1".

bit	Description
During write operation	When "0" is written: This bit cleared When "1" is written: Invalid
During read operation	When "0" is read: Indicating no detection of a reset/NMI factor When "1" is read: Indicating detection of a reset/NMI factor

#### **[bit3] RSTCAUSE3: Reset factor bit 3**

If the watchdog counter clear protection trigger sequence is executed before the watchdog counter reaches the window lower limit value, this bit is set to "1".

bit	Description
During write operation	When "0" is written: This bit cleared When "1" is written: Invalid
During read operation	When "0" is read: Indicating no detection of a reset/NMI factor When "1" is read: Indicating detection of a reset/NMI factor





**[bit2] RSTCAUSE2: Reset factor bit 2**

When the watchdog counter reaches the window upper limit value, this bit is set to "1".

bit	Description
During write operation	When "0" is written: This bit cleared When "1" is written: Invalid
During read operation	When "0" is read: Indicating no detection of a reset/NMI factor When "1" is read: Indicating detection of a reset/NMI factor

When the prior warning interrupt request is used, RSTCAUSE2 cannot be cleared during the period from the occurrence of a prior warning interrupt until a watchdog reset or watchdog interrupt (NMI).

**[bit1] RSTCAUSE1: Reset factor bit 1**

When there is a violation of the watchdog counter clear protection trigger sequence, this bit is set to "1".  
For details, see Figure 3-2.

bit	Description
During write operation	When "0" is written: This bit cleared When "1" is written: Invalid
During read operation	When "0" is read: Indicating no detection of a reset/NMI factor When "1" is read: Indicating detection of a reset/NMI factor

**[bit0] RSTCAUSE0: Reset factor bit 0**

If the value written in the software watchdog trigger 0/1 register (SWDGn\_TRG0 or SWDGn\_TRG1) does not match a proper value, this bit is set to "1".

bit	Description
During write operation	When "0" is written: This bit cleared When "1" is written: Invalid
During read operation	When "0" is read: Indicating no detection of a reset/NMI factor When "1" is read: Indicating detection of a reset/NMI factor

**Notes:**

- When using the software watchdog timer, be sure to clear this register.
- The value in this register becomes valid when the watchdog reset request is generated.

## 6.4. Software Watchdog Trigger 0 Register (SWDGn\_TRG0)

When performing write access to this register, you do not need to follow the watchdog register write protection sequence. This register is used to execute the watchdog counter clear protection trigger sequence. Write the value defined in the software watchdog trigger 0 configuration register (SWDGn\_TRG0CFG) to this register. When a value other than the one in the software watchdog trigger 0 configuration register (SWDGn\_TRG0CFG) is written, the watchdog reset request or watchdog interrupt request (NMI) is generated.

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	WDGTRG0							
ACCESS_TYPE	R0,W							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

**[bit31:8] Reserved: Reserved bits**

### **[bit7:0] WDGTRG0[7:0]: Watchdog trigger 0 bits**

These bits are used to execute the watchdog counter clear protection trigger sequence to clear the watchdog counter.

bit7:0	Description
During write operation	When the SWDGn_TRG0CFG value is written: 1 condition for executing the watchdog counter clear protection trigger sequence is met. When a value other than the SWDGn_TRG0CFG value is written: A watchdog error is generated.
During read operation	Reads "0b00000000".

**Note:**

- Figure 3-2 shows the flow of watchdog counter clear protection trigger sequence.



## 6.5. Software Watchdog Trigger 1 Register (SWDGn\_TRG1)

When performing write access to this register, you do not need to follow the watchdog register write protection sequence. This register is used to execute the watchdog counter clear protection trigger sequence. Write the value defined in the software watchdog trigger 1 configuration register (SWDGn\_TRG1CFG) to this register. When a value other than the one in the software watchdog trigger 1 configuration register (SWDGn\_TRG1CFG) is written, the watchdog reset request or watchdog interrupt request (NMI) is generated.

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	WDGTRG1							
ACCESS_TYPE	R0,W							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

**[bit31:8] Reserved: Reserved bits**

### **[bit7:0] WDGTRG1[7:0]: Watchdog trigger 1 bits**

These bits are used to execute the watchdog counter clear protection trigger sequence to clear the watchdog counter.

bit7:0	Description
During write operation	When the SWDGn_TRG1CFG value is written: 1 condition for executing the watchdog counter clear protection trigger sequence is met. When a value other than the SWDGn_TRG1CFG value is written: A watchdog error is generated.
During read operation	Reads "0b00000000".

**Note:**

- Figure 3-2 shows the flow of watchdog counter clear protection trigger sequence.

## 6.6. Software Watchdog Interrupt Configuration Register (SWDGn\_INT)

When performing write access to this register, follow the watchdog register write protection sequence. This register is used to make settings related to the watchdog interrupt request (NMI) and prior warning interrupt request. This register also includes interrupt status flags.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved						RSTEN	IRQEN
ACCESS_TYPE	R0,WX						R/W	R/W
PROT_TYPE	WPS							
INITIAL_VALUE	0000000						1	0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved						NMIFLAG	IRQFLAG
ACCESS_TYPE	R0,WX						R,WX	R,WX
PROT_TYPE	WPS							
INITIAL_VALUE	0000000						0	0

**[bit31:18] Reserved: Reserved bits**

### **[bit17] RSTEN: Reset/NMI enable bit**

This bit is used to generate either the watchdog reset request or watchdog interrupt request (NMI) in response to a watchdog error. Prohibiting to set this bit to "0" except for the purpose of using it as a test function. (If this bit is set to "0" during application execution, security is compromised.) This bit is equipped with a majority circuit with 3 FFs as a measure against a bit invert caused by the effects of noise or another factor.

bit	Description
During write operation	When "0" is written: The watchdog interrupt request (NMI) is generated. When "1" is written: The watchdog interrupt request is generated.
During read operation	Set value read



**[bit16] IRQEN: Prior warning interrupt enable bit**

This bit is used to enable generation of the prior warning interrupt request.

bit	Description
During write operation	When "0" is written: The prior warning interrupt request is not generated. When "1" is written: The prior warning interrupt request is generated.
During read operation	Set value read

**[bit15:2] Reserved: Reserved bits**

**[bit1] NMIFLAG: NMI flag**

This bit is set by a watchdog error when the RSTEN bit in the software watchdog interrupt configuration register (SWDGN\_INT) is "0". When the RSTEN bit in the software watchdog interrupt clear register (SWDGN\_INT) is "1", the watchdog reset request is generated. You can clear this bit by writing "0" to the NMICLR bit in the software watchdog interrupt configuration register (SWDGN\_INTCLR).

bit	Description
During write operation	Invalid
During read operation	When "0" is read: No detection of a watchdog error (NMI) is indicated. When "1" is read: Detection of a watchdog error (NMI) is indicated.

**[bit0] IRQFLAG: IRQ flag**

This bit is set by a watchdog error. The prior warning interrupt is generated when the IRQEN bit in the software watchdog interrupt clear register (SWDGN\_INT) is "1". You can clear this bit by writing "1" to the IRQCLR bit in the software watchdog interrupt configuration register (SWDGN\_INTCLR).

bit	Description
During write operation	Invalid
During read operation	When "0" is read: No detection of a watchdog error (IRQ) is indicated. When "1" is read: Detection of a watchdog error (IRQ) is indicated.

**Notes:**

- For details on the watchdog interrupt request (NMI), see "(m) Generation of Watchdog Reset Request or Watchdog Interrupt Request (NMI)" in Section "CHAPTER 19:3. Operation".
- For details on watchdog errors, see "(4) Watchdog Error" in Section "7. Precautions for Using".

## 6.7. Software Watchdog Interrupt Clear Register (SWDGN\_INTCLR)

When performing write access to this register, follow the watchdog register write protection sequence. This register is used to clear the NMI flag and IRQ flag in the software watchdog interrupt configuration register (SWDGN\_INT). You can write data to this register even after the LOCK bit in the software watchdog configuration register (SWDGN\_CFG) is set.

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	WPS
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved						NMICLR	IRQCLR
ACCESS_TYPE	R0,WX						R0,W	R0,W
PROT_TYPE	WPS							
INITIAL_VALUE	000000						0	0

**[bit31:2] Reserved: Reserved bits**

### **[bit1] NMICLR: NMI clear bit**

This bit is used to clear the NMIFLAG bit in the software watchdog interrupt configuration register (SWDGN\_INT).

bit	Description
During write operation	When "0" is written: Invalid When "1" is written: The NMI flag is cleared.
During read operation	"0" is read.

You must clear the NMI after waiting for watchdog counter clear (0x0000\_0000). Check the watchdog counter value with SWDGN\_CNT.

### **[bit0] IRQCLR: Prior warning interrupt clear bit**

This bit is used to clear the IRQFLAG bit in the software watchdog interrupt configuration register (SWDGN\_INT).

bit	Description
During write operation	When "0" is written: Invalid When "1" is written: The IRQ flag is cleared.
During read operation	"0" is read.



## 6.8. Software Watchdog Trigger 0 Configuration Register (SWDGN\_TRG0CFG)

When performing write access to this register, follow the watchdog register write protection sequence. This register is used to define a valid value to be written to the software watchdog trigger 0 register (SWDGN\_TRG0) when the watchdog counter clear protection trigger sequence is performed.

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	WPS
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	WDGTRG0CFG							
ACCESS_TYPE	R/W							
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							

**[bit31:8] Reserved: Reserved bits**

### **[bit7:0] WDGTRG0CFG[7:0]: Watchdog trigger 0 configuration bits**

These bits are used to define a value to be written to the software watchdog trigger 0 register (SWDGN\_TRG0) for execution of the watchdog counter clear protection trigger sequence.

bit7:0	Description
During write operation	Set value written
During read operation	Set value read

## 6.9. Software Watchdog Trigger 1 Configuration Register (SWDGN\_TRG1CFG)

When performing write access to this register, follow the watchdog register write protection sequence. This register is used to define a valid value to be written to the software watchdog trigger 1 register (SWDGN\_TRG1) when the watchdog counter clear protection trigger sequence is performed.

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	WPS
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	WDGTRG1CFG							
ACCESS_TYPE	R/W							
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							

**[bit31:8] Reserved: Reserved bits**

### **[bit7:0] WDGTRG1CFG[7:0]: Watchdog trigger 1 configuration bits**

These bits are used to define a value to be written to the software watchdog trigger 1 register (SWDGN\_TRG1) for execution of the watchdog counter clear protection trigger sequence.

bit7:0	Description
During write operation	Set value written
During read operation	Set value read





## 6.10. Software Watchdog Lower Limit RUN Setting Register (SWDGn\_RUNLLS)

When performing write access to this register, follow the watchdog register write protection sequence. A setting value of the window lower limit value for RUN is written to this register. However, this register value is different from the window lower limit value for RUN that is actually used. For details on how to reflect this register value in the window lower limit value for RUN actually used, see Section "6.16 Software Watchdog Lower Limit RUN Current Register (SWDGn\_RUNLLC)". You can change the set value of this register even after the LOCK bit in the software watchdog configuration register (SWDGn\_CFG) is set.

BIT_OFFSET	31-0
BIT_NAME	WDGRUNLLS
ACCESS_TYPE	R/W
PROT_TYPE	WPS
INITIAL_VALUE	00000000_00000000_00000000_00000000

### [bit31:0] WDGRUNLLS[31:0]: Window lower limit value for RUN set bits

These bits are used to define the window lower limit value for RUN. The reading of these bits returns the set value regardless of the software watchdog configuration register (SWDGn\_CFG) LOCK bit value.

bit31:0	Description
During write operation	When "0" is written to all 32 bits: The window function is disabled. When any value except "0" is written: The window function is enabled.
During read operation	Set value read

**Note:**

- This register value is different from the window lower limit value that is actually used. The window lower limit value actually used is the value in the software watchdog lower limit RUN current register (SWDGn\_RUNLLC).

## 6.11. Software Watchdog Upper Limit RUN Setting Register (SWDGn\_RUNULS)

When performing write access to this register, follow the watchdog register write protection sequence. A setting value of the window upper limit value for RUN is written to this register. However, this register value is different from the window upper limit value for RUN that is actually used. For details on how to reflect this register value in the window upper limit value for RUN actually used, see Section "6.17 Software Watchdog Upper Limit RUN Current Register (SWDGn\_RUNULC)". You can change the set value of this register even after the LOCK bit in the software watchdog configuration register (SWDGn\_CFG) is set.

BIT_OFFSET	31-0
BIT_NAME	WDGRUNULS
ACCESS_TYPE	R/W
PROT_TYPE	WPS
INITIAL_VALUE	00000001_00000000_00000000_00000000

### [bit31:0] WDGRUNULS[31:0]: Window upper limit value for RUN set bits

These bits are used to define the window upper limit value for RUN. The reading of these bits returns the set value regardless of the software watchdog configuration register (SWDGn\_CFG) LOCK bit value.

bit31:0	Description
During write operation	Set value written
During read operation	Set value read

#### Note:

- This register value is different from the window upper limit value that is actually used. The window lower limit value actually used is the value in the software watchdog lower limit RUN current register (SWDGn\_RUNLLC).



## 6.12. Software Watchdog Lower Limit PSS Setting Register (SWDGn\_PSSLLS)

When performing write access to this register, follow the watchdog register write protection sequence. A setting value of the window lower limit value for PSS is written to this register. However, this register value is different from the window lower limit value for PSS that is actually used. For details on how to reflect this register value in the window lower limit value for PSS actually used, see Section "6.18 Software Watchdog Lower Limit PSS Current Register (SWDGn\_PSSLLC)". You can change the set value of this register even after the LOCK bit in the software watchdog configuration register (SWDGn\_CFG) is set.

BIT_OFFSET	31-0
BIT_NAME	WDGPSSLLS
ACCESS_TYPE	R/W
PROT_TYPE	WPS
INITIAL_VALUE	00000000_00000000_00000000_00000000

### [bit31:0] WDGPSSLLS[31:0]: Window lower limit value for PSS set bits

These bits are used to define the window lower limit value for PSS. The reading of these bits returns the set value regardless of the software watchdog configuration register (SWDGn\_CFG) LOCK bit value.

bit31:0	Description
During write operation	When "0" is written to all 32 bits: The window function is disabled. When any value except "0" is written: The window function is enabled.
During read operation	Set value read

#### Note:

- This register value is different from the window lower limit value that is actually used. The window lower limit value actually used is the value in the software watchdog lower limit PSS current register (SWDGn\_PSSLLC).

### 6.13. Software Watchdog Upper Limit PSS Setting Register (SWDGn\_PSSULS)

When performing write access to this register, follow the watchdog register write protection sequence. A setting value of the window upper limit value for PSS is written to this register. However, this register value is different from the window upper limit value for PSS that is actually used. For details on how to reflect this register value in the window upper limit value for PSS actually used, see Section "6.19 Software Watchdog Upper Limit PSS Current Register (SWDGn\_PSSULC)". You can change the set value of this register even after the LOCK bit in the software watchdog configuration register (SWDGn\_CFG) is set.

BIT_OFFSET	31-0
BIT_NAME	WDGPSSULS
ACCESS_TYPE	R/W
PROT_TYPE	WPS
INITIAL_VALUE	10000000_00000000_00000000_00000000

#### [bit31:0] WDGPSSULS[31:0]: Window upper limit value for PSS set bits

These bits are used to define the window upper limit value for PSS. The reading of these bits returns the set value regardless of the software watchdog configuration register (SWDGn\_CFG) LOCK bit value.

bit31:0	Description
During write operation	Set value written
During read operation	Set value read

**Note:**

- This register value is different from the window upper limit value that is actually used. The window upper limit value actually used is the value in the software watchdog upper limit PSS current register (SWDGn\_PSSULC).



## 6.14. Software Watchdog Reset Delay Counter Register (SWDGN\_RSTDLY)

When performing write access to this register, follow the watchdog register write protection sequence. This register is used to set the number of cycles for the delay time from the occurrence of a watchdog error to the watchdog reset request or watchdog interrupt request (NMI). The reference clock for the delay time is the source clock of the watchdog counter. When the IRQEN bit in the software watchdog interrupt configuration register (SWDGN\_INT) is "1", this register indicates the number of cycles for the delay time from the prior warning interrupt request to the watchdog reset request or watchdog interrupt request (NMI).

BIT_OFFSET	31-16
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	WPS
INITIAL_VALUE	00000000_00000000

BIT_OFFSET	15-0
BIT_NAME	WDGRSTDLY
ACCESS_TYPE	R0,W
PROT_TYPE	WPS
INITIAL_VALUE	00000000_00000000

[bit31:16] Reserved: Reserved bits

### [bit15:0] WDGRSTDLY[15:0]: Reset/NMI delay counter bits

These bits define the number of cycles for the delay time that is to be inserted before generation of the watchdog reset request or watchdog interrupt request (NMI). The reference clock for this delay time is the source clock of the watchdog counter.

bit15:0	Description
During write operation	Set value written
During read operation	"0x0000" is read.

## 6.15. Software Watchdog Configuration Register (SWDGn\_CFG)

When performing write access to this register, follow the watchdog register write protection sequence. This register is used to make the operation settings of the software watchdog timer.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							LOCK
ACCESS_TYPE	R0,WX							R,W
PROT_TYPE	WPS							
INITIAL_VALUE	0000000							0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved			OBSSEL				
ACCESS_TYPE	R0,WX			R/W				
PROT_TYPE	WPS							
INITIAL_VALUE	000			00000				

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved						CLKSEL	
ACCESS_TYPE	R0,WX						R/W	
PROT_TYPE	WPS							
INITIAL_VALUE	000000						00	

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved					ALLOW STOPCLK	WDENPSS	WDENRUN
ACCESS_TYPE	R0,WX					R/W	R/W	R/W
PROT_TYPE	WPS							
INITIAL_VALUE	00000					0	1	1

**[bit31:25] Reserved: Reserved bits**

### **[bit24] LOCK: Lock bit**

You can write data to this bit only once. This register is used to prevent rewriting of the set values of various registers. When this bit is "0", you can rewrite the setting values of various registers. When this bit is set to "1" the watchdog counter is automatically cleared. This bit is equipped with a majority circuit with 3 FFs as a measure against a bit invert caused by the effects of noise or another factor.

bit	Description
During write operation	When "0" is written: Invalid When "1" is written: The set values of the registers are locked.
During read operation	When "0" is read: The lock for the register set values is disabled. When "1" is read: The lock for the register set values is enabled.

**[bit23:21] Reserved: Reserved bits**



**[bit20:16] OBSSEL[4:0]: Watchdog counter monitor bit output selection bits**

These bits are used for selecting any 1 bit in the watchdog counter (32 bits) as the output of the watchdog counter monitor bit.

bit20:16	Description
During write operation	<p>When "0b00000" is written: Bit 0 is selected for the monitor output.</p> <p>When "0b00001" is written: Bit 1 is selected for the monitor output.</p> <p>When "0b00010" is written: Bit 2 is selected for the monitor output.</p> <p style="text-align: center;">•</p> <p style="text-align: center;">•</p> <p style="text-align: center;">•</p> <p>When "0b11111" is written: Bit 31 is selected for the monitor output.</p>
During read operation	Set value read

**Note:**

- Output of watchdog counter monitor bit for MCU output pin is unsupported. For this product, any data written to these bits does not affect operation.

**[bit15:10] Reserved: Reserved bits**

**[bit9:8] CLKSEL[1:0]: Clock selection bits**

These bits are used to select the source clock for the watchdog counter. When activated, the watchdog counter starts its operation as the fast-CR clock. For details on clock switching, see "(2) Switching of Watchdog Counter Source Clock" in Section "7. Precautions for Using".

bit9:8	Description
During write operation	<p>When "0b00" is written: The fast-CR clock is selected.</p> <p>When "0b01" is written: The slow-CR clock is selected.</p> <p>When "0b10" is written: prohibit to setting.</p> <p>When "0b11" is written: The main clock is selected.</p>
During read operation	Set value read

**[bit7:3] Reserved: Reserved bits**

**[bit2] ALLOWSTOPCLK: Clock stop for PSS enable bit**

This bit is used to enable transition to PSS that involves the source clock stop of the watchdog counter. This bit is valid only when the WDENPSS bit in the software watchdog configuration register (SWDGN\_CFG) is "1".

bit	Description
During write operation	<p>When "0" is written: Watchdog clock stop in PSS is disabled.</p> <p>When "1" is written: Watchdog clock stop in PSS is enabled.</p>
During read operation	Set value read

**[bit1] WDENPSS: Watchdog counter for PSS enable bit**

This bit is used to enable the watchdog counter in PSS. This bit is enabled when the LOCK bit in the software watchdog configuration register (SWDGn\_CFG) is set to "1". This bit is equipped with a majority circuit with 3 FFs as a measure against a bit invert caused by the effects of noise or another factor.

bit	Description
During write operation	When "0" is written: The watchdog counter in PSS is disabled. When "1" is written: The watchdog counter in PSS is enabled.
During read operation	Set value read

**[bit0] WDENRUN: Watchdog counter for RUN enable bit**

This bit is used to enable the watchdog counter in RUN. This bit is enabled when the LOCK bit in the software watchdog configuration register (SWDGn\_CFG) is set to "1". This bit is equipped with a majority circuit with 3 FFs as a measure against a bit invert caused by the effects of noise or another factor.

bit	Description
During write operation	When "0" is written: The watchdog counter in RUN is disabled. When "1" is written: The watchdog counter in RUN is enabled.
During read operation	Set value read

**Note:**

- When the ALLOWSTOPCLK bit in the software watchdog configuration register (SWDGn\_CFG) is "1", transition to PSS that involves clock stop is enabled. However, in case of transition from RUN to PSS during reset delay counter operation, the reset delay counter is stopped until a return from PSS to RUN occurs.





## 6.16. Software Watchdog Lower Limit RUN Current Register (SWDGn\_RUNLLC)

This register is used to read the window lower limit value for RUN that is actually used. This register fetches the set value in the software watchdog lower limit RUN setting register (SWDGn\_RUNLLS) when the LOCK bit in the software watchdog configuration register (SWDGn\_CFG) is set. You can change the set value of this register even after the LOCK bit in the software watchdog configuration register (SWDGn\_CFG) is set. In this case, this register fetches the set value of the software watchdog lower limit RUN setting register (SWDGn\_RUNLLS) when the watchdog counter clear protection sequence is executed.

BIT_OFFSET	31-0
BIT_NAME	WDGRUNLLC
ACCESS_TYPE	R,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000_00000000

### [bit31:0] WDGRUNLLC[31:0]: Window lower limit for RUN current bits

These bits are used to define the window lower limit value for RUN. The reading of these bits returns the initial value, until the software watchdog configuration register (SWDGn\_CFG) LOCK bit is set.

bit31:0	Description
During write operation	Invalid
During read operation	Set value read

**Note:**

- An AHB transfer error response is generated where there is write access to this register.

### 6.17. Software Watchdog Upper Limit RUN Current Register (SWDGn\_RUNULC)

This register is used to read the window upper limit value for RUN that is actually used. This register fetches the set value in the software watchdog upper limit RUN setting register (SWDGn\_RUNULS) when the LOCK bit in the software watchdog configuration register (SWDGn\_CFG) is set. You can change the set value of this register even after the LOCK bit in the software watchdog configuration register (SWDGn\_CFG) is set. In this case, this register fetches the set value of the software watchdog upper limit RUN setting register (SWDGn\_RUNULS) when the watchdog counter clear protection trigger sequence is executed.

BIT_OFFSET	31-0
BIT_NAME	WDGRUNULC
ACCESS_TYPE	R,WX
PROT_TYPE	-
INITIAL_VALUE	00000001_00000000_00000000_00000000

#### [bit31:0] WDGRUNULC[31:0]: Window upper limit for RUN current bits

These bits are used to define the window upper limit value for RUN. The reading of these bits returns the initial value, until the software watchdog configuration register (SWDGn\_CFG) LOCK bit is set.

bit31:0	Description
During write operation	Invalid
During read operation	Set value read

**Note:**

- An AHB transfer error response is generated where there is write access to this register.



## 6.18. Software Watchdog Lower Limit PSS Current Register (SWDGn\_PSSLLC)

This register is used to read the window lower limit value for PSS that is actually used. This register fetches the set value in the software watchdog lower limit PSS setting register (SWDGn\_PSSLLS) when the LOCK bit in the software watchdog configuration register (SWDGn\_CFG) is set. You can change the set value of this register even after the LOCK bit in the software watchdog configuration register (SWDGn\_CFG) is set. In this case, this register fetches the set value of the software watchdog lower limit PSS setting register (SWDGn\_PSSLLS) when the watchdog counter clear protection trigger sequence is executed.

BIT_OFFSET	31-0
BIT_NAME	WDGPSSLLC
ACCESS_TYPE	R,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000_00000000

### [bit31:0] WDGPPSSLLC[31:0]: Window lower limit for PSS current bits

These bits are used to define the window lower limit value for PSS. The reading of these bits returns the initial value, until the software watchdog configuration register (SWDGn\_CFG) LOCK bit is set.

bit31:0	Description
During write operation	Invalid
During read operation	Set value read

**Note:**

- An AHB transfer error response is generated where there is write access to this register.

## 6.19. Software Watchdog Upper Limit PSS Current Register (SWDGn\_PSSULC)

This register is used to read the window upper limit value for PSS that is actually used. This register fetches the set value in the software watchdog upper limit PSS setting register (SWDGn\_PSSULS) when the LOCK bit in the software watchdog configuration register (SWDGn\_CFG) is set. You can change the set value of this register even after the LOCK bit in the software watchdog configuration register (SWDGn\_CFG) is set. In this case, this register fetches the set value of the software watchdog upper limit PSS setting register (SWDGn\_PSSULS) when the watchdog counter clear protection trigger sequence is executed.

BIT_OFFSET	31-0
BIT_NAME	WDGPSSULC
ACCESS_TYPE	R,WX
PROT_TYPE	-
INITIAL_VALUE	10000000_00000000_00000000_00000000

### [bit31:0] WDGPSSULC[31:0]: Window upper limit for PSS current bits

These bits are used to define the window upper limit value for PSS. The reading of these bits returns the initial value, until the software watchdog configuration register (SWDGn\_CFG) LOCK bit is set.

bit31:0	Description
During write operation	Invalid
During read operation	Set value read

**Note:**

- Write access to this register generates an AHB transfer error response.



## 7. Precautions for Using

This section describes precautions on the use of the software watchdog timer.

### (1) Watchdog Window Setting

For the software watchdog timer, you can make 2 types of window setting, the setting for RUN and setting for PSS. The user must make an appropriate window setting for each device state.

When the device transitions from PSS to RUN, the software watchdog timer continues operation using the PSS window setting until the watchdog counter clear protection trigger sequence is executed. The watchdog counter clear protection trigger sequence switches the window setting to the RUN window setting. You can control the watchdog function in each device state with the WDENRUN bit and WDENPSS bit in the software watchdog configuration register (SWDGn\_CFG).

### (2) Switching of Watchdog Counter Source Clock

The system controller controls the switching of the watchdog counter source clock since the source clock control of the watchdog counter is a part of the system setting information (profile). To change the watchdog counter source clock, follow the sequence below.

1. Writing data to the RUN profile source clock enable register of the system controller (SYSC\_RUNCKSRER) enables oscillation of the new source clock of the watchdog. Then, writing "0xAB" to the RUN profile update trigger register of the system controller (SYSC\_TRGRUNCNTR) executes RUN profile update.
2. Setting the CLKSEL bit in the software watchdog configuration register (SWDGn\_CFG) sets the new source clock of the watchdog counter. Then, writing data to the following registers finalizes the window setting.
3. Software watchdog lower limit RUN setting register (SWDGn\_RUNLLS)
4. Software watchdog upper limit RUN setting register (SWDGn\_RUNULS)
5. Software watchdog lower limit PSS setting register (SWDGn\_PSSLSS)
6. Software watchdog upper limit PSS setting register (SWDGn\_PSSULS)
7. Setting the LOCK bit in the software watchdog configuration register (SWDGn\_CFG) to "1" locks rewriting of the register set values.
8. Writing "0xAB" to the RUN profile update trigger register of the system controller (SYSC\_TRGRUNCNTR) executes RUN profile update. At this point in time, the source clock of the watchdog counter is switched to the new source clock selected by the CLKSEL bit of the software watchdog configuration register (SWDGn\_CFG).

**Note:**

- When the oscillation of the new source clock is stable, Step 1 is unnecessary.

### (3) Issuing Software Reset with Slow-CR Clock Selected

If issuing software reset with slow-CR clock selected by the clock of the watchdog timer, perform watchdog clear sequence beforehand.

#### (4) Watchdog Error

A watchdog error is generated under the following conditions. This watchdog error generates the watchdog reset request/interrupt request (NMI).

1. An incorrect value is written to the software watchdog trigger 0/1 register (SWDGn\_TRG0 or SWDGn\_TRG1).
2. The watchdog counter clear protection trigger sequence is violated.
3. The watchdog counter clear protection trigger sequence is not executed before the watchdog counter reaches the window upper limit value.
4. The watchdog counter clear protection trigger sequence is executed before the watchdog counter reaches the window lower limit value.
5. The watchdog counter clear protection trigger sequence is executed while the LOCK bit in the software watchdog configuration register (SWDGn\_CFG) is still "0".

#### (5) Factors for Bus Error Responses (Data Abort)

A bus error response is generated in the following cases.

- Access (read or write) to an address where no register exists
- Violation of the watchdog register write protection sequence
- Write access to a register with the read-only attribute (indicated as R,WX)
- Write access to a register after the LOCK bit in the software watchdog configuration register (SWDGn\_CFG) is set to "1"
- Write access with an incorrect value to the software watchdog protection register (SWDGn\_PROT)
- Byte/Half-word write access to the software watchdog protection register (SWDGn\_PROT)
- Write access to a register in non-privileged mode
- (Except for the software watchdog trigger 0/1 register (SWDGn\_TRG0 and SWDGn\_TRG1))

#### (6) Watchdog Timer Operation in Standby State of Processor

The software watchdog timer operates using the RUN window setting in the following case.

- Without a valid key ("0xBA" or "0xBB") being written to the PSS profile update enable register (SYSC\_PSSSEN.PSSSEN), the processor transitions to the standby state by executing a wait for interrupt (WFI) instruction.





## CHAPTER 21: Bit-band Unit

This chapter explains bit-band unit.

---

1. Overview
2. Operation





## 1. Overview

This section provides an overview of the bit-band unit.

The memory map of this device has an address area called "bit-band alias area" and an address area named "bit-band area". The function of the bit-band unit is to access 1 byte in the bit-band alias area, to set or clear the 1 corresponding bit in the bit-band area. The bit-band area is mapped over a resource area.



2. Operation

This section describes the operation of bit-band unit.

(1) Bit-band Area and Bit-band Alias Area

The relationship between bit-band alias area and bit-band area is shown in Figure 2-1. 1 byte of bit-band alias area corresponds to 1 bit of bit-band area. Only the corresponding bit of bit-band in the bit-band area can be accessed from the bit-band alias area.

Figure 2-1 Bit-band Alias Area and Bit-band Area

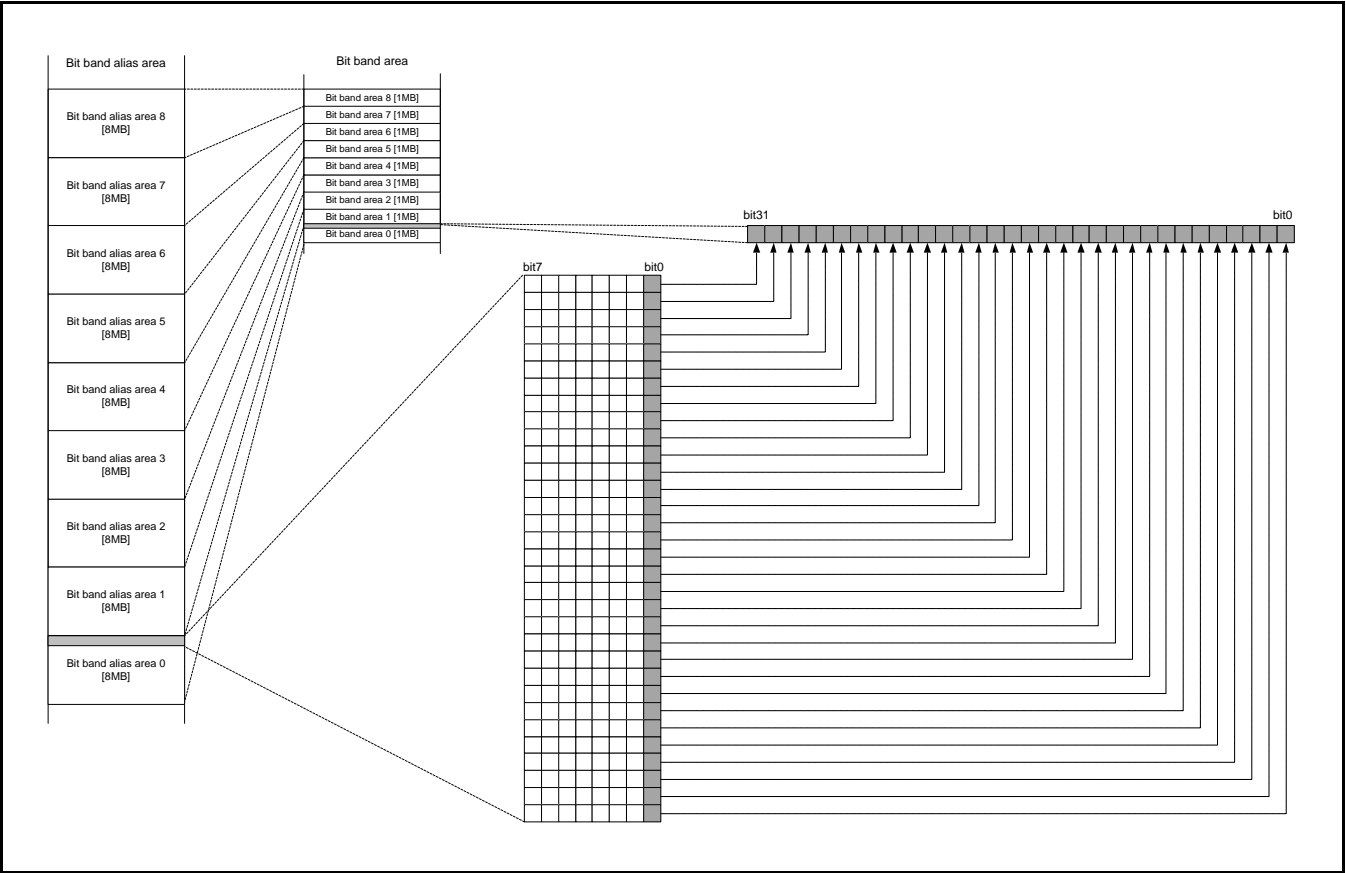


Table 2-1 Bit-band Area and Bit-band Alias Area

Bit-band Area	Groups	Bit-band Alias Area
0xB040_0000 to 0xB04F_FFFF	Memory & Config Group	0xA200_0000 to 0xA27F_FFFF
0xB060_0000 to 0xB06F_FFFF	MCU Config Group	0xA300_0000 to 0xA37F_FFFF
0xB070_0000 to 0xB07F_FFFF	Common Peripheral Group(AHB)	0xA380_0000 to 0xA3FF_FFFF
0xB080_0000 to 0xB08F_FFFF	Common Peripheral Group(APB)	0xA400_0000 to 0xA47F_FFFF
0xB100_0000 to 0xB10F_FFFF	Application Specific Peripheral Group(Slave-A)	0xA800_0000 to 0xA87F_FFFF
0xB200_0000 to 0xB20F_FFFF	Application Specific Peripheral Group(Slave-B)	0xA880_0000 to 0xA8FF_FFFF



### (2) Bit Set and Bit Clear by Bit-band Unit

If "1" is written to bit[0] in the corresponding address of bit-band in the bit-band alias area, the corresponding bit in the bit-band area is set to "1". Also, if "0" is written, corresponding bit in bit-band area is cleared to "0".

The value of bit[7:1] is not affected.

#### **Notes:**

- *Writing to a bit-band alias area that does not correspond to bit-band is prohibited.*
- *Writing to a bit-band alias area in other than byte size is prohibited.*

### (3) Reading Bit by Bit-band Unit

- The corresponding bit of bit-band area is read in bit[0] by reading using the corresponding address of bit-band in bit-band alias domain. Bit[7:1] always reads "0b0000000".
- If byte read access to a bit-band alias area that does not correspond to bit-band is performed, "0b00000000" is always read.

#### **Notes:**

- *Reading a bit-band alias area that does not correspond to bit-band is prohibited.*
- *Reading a bit-band alias area in other than byte size is prohibited.*

#### (4) Corresponding Bit of Bit-band

This shows a list of corresponding registers and bits of bit-band. The register attribute is the same even when you perform access using the bit-band function.

##### 1) System Controller

Register Name	Bit Name	Bit-band Alias Address	Bit Set	Bit Clear
System status register (SYSC_SYSSTSR)	RUN profile update completion (main status control) flag bit (RUNDF0)	0xA300_1444	-	✓
	PSS profile update completion (main status control) flag bit (PSSDF0)	0xA300_1445	-	✓
	PSS profile update completion (sub status control 1) flag bit (PSSDF1)	0xA300_144D	-	✓
System error interrupt factor register 0 (SYSC_SYSERRIR0)	Main oscillation abnormality detection error interrupt request bit (MOMIF)	0xA300_14A0	-	✓
	PLL abnormality detection error interrupt request bit (PMIF)	0xA300_14A2	-	✓
	Sub-system PLL abnormality detection error interrupt request bit (SSPMIF)	0xA300_14A4	-	✓
	5.0 V low-voltage detection interrupt request bit (LVD50IF)	0xA300_14B8	-	✓
System error interrupt factor register 1 (SYSC_SYSERRIR1)	Trigger error interrupt request bit (TRGERRIF)	0xA300_14C0	-	✓
	RUN profile update enable write error interrupt request bit (RUNTRGERRIF)	0xA300_14C1	-	✓
	PSS trigger cancel interrupt request bit (PSSTRGCIF0)	0xA300_14C2	-	✓
	PSS profile update enable write error interrupt request bit (PSENERRIF0)	0xA300_14C3	-	✓
	RUN profile error interrupt request bit (RUNERRIF0)	0xA300_14C4	-	✓
	RUN profile (PSS recovery) error interrupt request bit (RUNWKERRIF0)	0xA300_14C5	-	✓
	PSS profile error interrupt request bit (PSSERRIF0)	0xA300_14C6	-	✓
	PSS trigger (sub status control 1) cancel interrupt request bit (PSSTRGCIF1)	0xA300_14CA	-	✓
	PSS profile update enable write error interrupt request bit (PSENERRIF1)	0xA300_14CB	-	✓



## 2) Software Watchdog Timer

Register Name	Bit Name	Bit-band Alias Address	Bit Set	Bit Clear
Software watchdog interrupt configuration register (SWDG0_INT)	IRQ flag (IRQFLAG)	0xA304_0100	-	✓
	NMI flag (NMIFLAG)	0xA304_0101	-	✓
Software watchdog interrupt configuration register (SWDG1_INT)	IRQ flag (IRQFLAG)	0xA304_8100	-	✓
	NMI flag (NMIFLAG)	0xA304_8101	-	✓

## 3) Hardware Watchdog Timer

Register Name	Bit Name	Bit-band Alias Address	Bit Set	Bit Clear
Hardware watchdog interrupt configuration register (HWDG_INT)	IRQ flag (IRQFLAG)	0xA306_0100	-	✓
	NMI flag (NMIFLAG)	0xA306_0101	-	✓

## 4) External Interrupt

Register Name	Bit Name	Bit-band Alias Address	Bit Set	Bit Clear
External interrupt enable register (EIC00_ENIR)	External interrupt enable bit (EN0 to EN7)	0xA310_0000 to 0xA310_0007	✓	✓
Reserved	-	0xA310_0008 to 0xA310_001F	-	-
External interrupt factor register (EIC00_EIRR)	External interrupt factor detection bit (ER0 to ER7)	0xA310_0060 to 0xA310_0067	-	✓
Reserved	-	0xA310_0068 to 0xA310_007F	-	-
Noise filter enable register (EIC00_NFER)	Noise filter enable set bit (NFE0 to NFE7)	0xA310_00A0 to 0xA310_00A7	✓	✓
Reserved	-	0xA310_00A8 to 0xA310_00BF	-	-
DMA request enable register (EIC00_DRER)	DMA request enable bit (DRE0 to DRE7)	0xA310_01A0 to 0xA310_01A7	✓	✓
Reserved	-	0xA310_01A8 to 0xA310_01BF	-	-

### Note:

- Do not access the bit-band alias address of the reservation.

### 5) Interrupt Controller

Register Name	Bit Name	Bit-band Alias Address	Bit Set	Bit Clear
NMI software interrupt status register (IRC0_NMISIS)	NMI software interrupt status bit (NMISIS0 to NMISIS31)	0xA200_55C0 to 0xA200_55DF	✓	✓
IRQ software interrupt status register 0 (IRC0_IRQSIS0)	IRQ software interrupt status bit (IRQSIS0 to IRQSIS31)	0xA200_5A00 to 0xA200_5A1F	✓	✓
IRQ software interrupt status register 1 (IRC0_IRQSIS1)	IRQ software interrupt status bit (IRQSIS32 to IRQSIS63)	0xA200_5A20 to 0xA200_5A3F	✓	✓
IRQ software interrupt status register 2 (IRC0_IRQSIS2)	IRQ software interrupt status bit (IRQSIS64 to IRQSIS95)	0xA200_5A40 to 0xA200_5A5F	✓	✓
IRQ software interrupt status register 3 (IRC0_IRQSIS3)	IRQ software interrupt status bit (IRQSIS96 to IRQSIS127)	0xA200_5A60 to 0xA200_5A7F	✓	✓
IRQ software interrupt status register 4 (IRC0_IRQSIS4)	IRQ software interrupt status bit (IRQSIS128 to IRQSIS159)	0xA200_5A80 to 0xA200_5A9F	✓	✓
IRQ software interrupt status register 5 (IRC0_IRQSIS5)	IRQ software interrupt status bit (IRQSIS160 to IRQSIS191)	0xA200_5AA0 to 0xA200_5ABF	✓	✓
IRQ software interrupt status register 6 (IRC0_IRQSIS6)	IRQ software interrupt status bit (IRQSIS192 to IRQSIS223)	0xA200_5AC0 to 0xA200_5ADF	✓	✓
IRQ software interrupt status register 7 (IRC0_IRQSIS7)	IRQ software interrupt status bit (IRQSIS224 to IRQSIS255)	0xA200_5AE0 to 0xA200_5AFF	✓	✓
IRQ software interrupt status register 8 (IRC0_IRQSIS8)	IRQ software interrupt status bit (IRQSIS256 to IRQSIS287)	0xA200_5B00 to 0xA200_5B1F	✓	✓
IRQ software interrupt status register 9 (IRC0_IRQSIS9)	IRQ software interrupt status bit (IRQSIS288 to IRQSIS319)	0xA200_5B20 to 0xA200_5B3F	✓	✓
IRQ software interrupt status register 10 (IRC0_IRQSIS10)	IRQ software interrupt status bit (IRQSIS320 to IRQSIS351)	0xA200_5B40 to 0xA200_5B5F	✓	✓
IRQ software interrupt status register 11 (IRC0_IRQSIS11)	IRQ software interrupt status bit (IRQSIS352 to IRQSIS383)	0xA200_5B60 to 0xA200_5B7F	✓	✓
IRQ software interrupt status register 12 (IRC0_IRQSIS12)	IRQ software interrupt status bit (IRQSIS384 to IRQSIS415)	0xA200_5B80 to 0xA200_5B9F	✓	✓
IRQ software interrupt status register 13 (IRC0_IRQSIS13)	IRQ software interrupt status bit (IRQSIS416 to IRQSIS447)	0xA200_5BA0 to 0xA200_5BBF	✓	✓
IRQ software interrupt status register 14 (IRC0_IRQSIS14)	IRQ software interrupt status bit (IRQSIS448 to IRQSIS479)	0xA200_5BC0 to 0xA200_5BDF	✓	✓



Register Name	Bit Name	Bit-band Alias Address	Bit Set	Bit Clear
IRQ software interrupt status register 15 (IRC0_IRQSIS15)	IRQ software interrupt status bit (IRQSIS480 to IRQSIS511)	0xA200_5BE0 to 0xA200_5BFF	✓	✓
NMI software interrupt status register (IRC1_NMISIS)	NMI software interrupt status bit (NMISIS0 to NMISIS31)	0xA200_6000 to 0xA200_601F	✓	✓
IRQ software interrupt status register 0 (IRC1_IRQSIS0)	IRQ software interrupt status bit (IRQSIS0 to IRQSIS31)	0xA200_6020 to 0xA200_603F	✓	✓
IRQ software interrupt status register 1 (IRC1_IRQSIS1)	IRQ software interrupt status bit (IRQSIS32 to IRQSIS63)	0xA200_6040 to 0xA200_605F	✓	✓
IRQ software interrupt status register 2 (IRC1_IRQSIS2)	IRQ software interrupt status bit (IRQSIS64 to IRQSIS95)	0xA200_6060 to 0xA200_607F	✓	✓
IRQ software interrupt status register 3 (IRC1_IRQSIS3)	IRQ software interrupt status bit (IRQSIS96 to IRQSIS127)	0xA200_6080 to 0xA200_609F	✓	✓
IRQ software interrupt status register 4 (IRC1_IRQSIS4)	IRQ software interrupt status bit (IRQSIS128 to IRQSIS159)	0xA200_E080 to 0xA200_E09F	✓	✓
IRQ software interrupt status register 5 (IRC1_IRQSIS5)	IRQ software interrupt status bit (IRQSIS160 to IRQSIS191)	0xA200_E0A0 to 0xA200_E0BF	✓	✓
IRQ software interrupt status register 6 (IRC1_IRQSIS6)	IRQ software interrupt status bit (IRQSIS192 to IRQSIS223)	0xA200_E0C0 to 0xA200_E0DF	✓	✓
IRQ software interrupt status register 7 (IRC1_IRQSIS7)	IRQ software interrupt status bit (IRQSIS224 to IRQSIS255)	0xA200_E0E0 to 0xA200_E0FF	✓	✓
IRQ software interrupt status register 8 (IRC1_IRQSIS8)	IRQ software interrupt status bit (IRQSIS256 to IRQSIS287)	0xA200_E100 to 0xA200_E11F	✓	✓
IRQ software interrupt status register 9 (IRC1_IRQSIS9)	IRQ software interrupt status bit (IRQSIS288 to IRQSIS319)	0xA200_E120 to 0xA200_E13F	✓	✓
IRQ software interrupt status register 10 (IRC1_IRQSIS10)	IRQ software interrupt status bit (IRQSIS320 to IRQSIS351)	0xA200_E140 to 0xA200_E15F	✓	✓
IRQ software interrupt status register 11 (IRC1_IRQSIS11)	IRQ software interrupt status bit (IRQSIS352 to IRQSIS383)	0xA200_E160 to 0xA200_E17F	✓	✓
IRQ software interrupt status register 12 (IRC1_IRQSIS12)	IRQ software interrupt status bit (IRQSIS384 to IRQSIS415)	0xA200_E180 to 0xA200_E19F	✓	✓
IRQ software interrupt status register 13 (IRC1_IRQSIS13)	IRQ software interrupt status bit (IRQSIS416 to IRQSIS447)	0xA200_E1A0 to 0xA200_E1BF	✓	✓
IRQ software interrupt status register 14 (IRC1_IRQSIS14)	IRQ software interrupt status bit (IRQSIS448 to IRQSIS479)	0xA200_E1C0 to 0xA200_E1DF	✓	✓



Register Name	Bit Name	Bit-band Alias Address	Bit Set	Bit Clear
IRQ software interrupt status register 15 (IRC1_IRQSIS15)	IRQ software interrupt status bit (IRQSIS480 to IRQSIS511)	0xA200_E1E0 to 0xA200_E1FF	✓	✓





## 6) Inter-processor Communication

Register Name	Bit Name	Bit-band Alias Address	Bit Set	Bit Clear
Mailbox 0 request recipient status register (IPCU_MB0DSTR)	CPU0 request recipient status bit (DSTST0)	0xA20A_88C0	✓	✓
	CPU1 request recipient status bit (DSTST1)	0xA20A_88C1	✓	✓
	Reserved	0xA20A_88C2 to 0xA20A_88CF	-	-
Mailbox 0 request transmission mask status register (IPCU_MB0MSTR)	CPU0 request transmission mask status bit (MSKST0)	0xA20A_8940	✓	✓
	CPU1 request transmission mask status bit (MSKST1)	0xA20A_8941	✓	✓
	Reserved	0xA20A_8942 to 0xA20A_894F	-	-
Mailbox 0 acknowledgment status register (IPCU_MB0ASTR)	CPU0 acknowledgment status bit (ACKST0)	0xA20A_89C0	✓	✓
	CPU1 acknowledgment status bit (ACKST1)	0xA20A_89C1	✓	✓
	Reserved	0xA20A_89C2 to 0xA20A_89CF	-	-
Mailbox 1 request recipient status register (IPCU_MB1DSTR)	CPU0 request recipient status bit (DSTST0)	0xA20A_8CC0	✓	✓
	CPU1 request recipient status bit (DSTST1)	0xA20A_8CC1	✓	✓
	Reserved	0xA20A_8CC2 to 0xA20A_8CCF	-	-
Mailbox 1 request transmission mask status register (IPCU_MB1MSTR)	CPU0 request transmission mask status bit (MSKST0)	0xA20A_8D40	✓	✓
	CPU1 request transmission mask status bit (MSKST1)	0xA20A_8D41	✓	✓
	Reserved	0xA20A_8D42 to 0xA20A_8D4F	-	-
Mailbox 1 acknowledgment status register (IPCU_MB1ASTR)	CPU0 acknowledgment status bit (ACKST0)	0xA20A_8DC0	✓	✓
	CPU1 acknowledgment status bit (ACKST1)	0xA20A_8DC1	✓	✓
	Reserved	0xA20A_8DC2 to 0xA20A_8DCF	-	-
Mailbox 2 request recipient status register (IPCU_MB2DSTR)	CPU0 request recipient status bit (DSTST0)	0xA20A_90C0	✓	✓
	CPU1 request recipient status bit (DSTST1)	0xA20A_90C1	✓	✓
	Reserved	0xA20A_90C2 to 0xA20A_90CF	-	-
Mailbox 2 request transmission mask status register (IPCU_MB2MSTR)	CPU0 request transmission mask status bit (MSKST0)	0xA20A_9140	✓	✓
	CPU1 request transmission mask status bit (MSKST1)	0xA20A_9141	✓	✓
	Reserved	0xA20A_9142 to 0xA20A_914F	-	-
Mailbox 2 acknowledgment status register (IPCU_MB2ASTR)	CPU0 acknowledgment status bit (ACKST0)	0xA20A_91C0	✓	✓
	CPU1 acknowledgment status bit (ACKST1)	0xA20A_91C1	✓	✓
	Reserved	0xA20A_91C2 to 0xA20A_91CF	-	-
Mailbox 3 request recipient status register (IPCU_MB3DSTR)	CPU0 request recipient status bit (DSTST0)	0xA20A_94C0	✓	✓
	CPU1 request recipient status bit (DSTST1)	0xA20A_94C1	✓	✓
	Reserved	0xA20A_94C2 to 0xA20A_94CF	-	-
Mailbox 3 request transmission mask status register (IPCU_MB3MSTR)	CPU0 request transmission mask status bit (MSKST0)	0xA20A_9540	✓	✓
	CPU1 request transmission mask status bit (MSKST1)	0xA20A_9541	✓	✓
	Reserved	0xA20A_9542 to 0xA20A_954F	-	-
Mailbox 3 acknowledgment status register (IPCU_MB3ASTR)	CPU0 acknowledgment status bit (ACKST0)	0xA20A_95C0	✓	✓
	CPU1 acknowledgment status bit (ACKST1)	0xA20A_95C1	✓	✓
	Reserved	0xA20A_95C2 to 0xA20A_95CF	-	-

Register Name	Bit Name	Bit-band Alias Address	Bit Set	Bit Clear
Mailbox 4 request recipient status register (IPCU_MB4DSTR)	CPU0 request recipient status bit (DSTST0)	0xA20A_98C0	✓	✓
	CPU1 request recipient status bit (DSTST1)	0xA20A_98C1	✓	✓
	Reserved	0xA20A_98C2 to 0xA20A_98CF	-	-
Mailbox 4 request transmission mask status register (IPCU_MB4MSTR)	CPU0 request transmission mask status bit (MSKST0)	0xA20A_9940	✓	✓
	CPU1 request transmission mask status bit (MSKST1)	0xA20A_9941	✓	✓
	Reserved	0xA20A_9942 to 0xA20A_994F	-	-
Mailbox 4 acknowledgment status register (IPCU_MB4ASTR)	CPU0 acknowledgment status bit (ACKST0)	0xA20A_99C0	✓	✓
	CPU1 acknowledgment status bit (ACKST1)	0xA20A_99C1	✓	✓
	Reserved	0xA20A_99C2 to 0xA20A_99CF	-	-
Mailbox 5 request recipient status register (IPCU_MB5DSTR)	CPU0 request recipient status bit (DSTST0)	0xA20A_9CC0	✓	✓
	CPU1 request recipient status bit (DSTST1)	0xA20A_9CC1	✓	✓
	Reserved	0xA20A_9CC2 to xA20A_9CCF	-	-
Mailbox 5 request transmission mask status register (IPCU_MB5MSTR)	CPU0 request transmission mask status bit (MSKST0)	0xA20A_9D40	✓	✓
	CPU1 request transmission mask status bit (MSKST1)	0xA20A_9D41	✓	✓
	Reserved	0xA20A_9D42 to 0xA20A_9D4F	-	-
Mailbox 5 acknowledgment status register (IPCU_MB5ASTR)	CPU0 acknowledgment status bit (ACKST0)	0xA20A_9DC0	✓	✓
	CPU1 acknowledgment status bit (ACKST1)	0xA20A_9DC1	✓	✓
	Reserved	0xA20A_9DC2 to xA20A_9DCF	-	-
Mailbox 6 request recipient status register (IPCU_MB6DSTR)	CPU0 request recipient status bit (DSTST0)	0xA20A_A0C0	✓	✓
	CPU1 request recipient status bit (DSTST1)	0xA20A_A0C1	✓	✓
	Reserved	0xA20A_A0C2 to 0xA20A_A0CF	-	-
Mailbox 6 request transmission mask status register (IPCU_MB6MSTR)	CPU0 request transmission mask status bit (MSKST0)	0xA20A_A140	✓	✓
	CPU1 request transmission mask status bit (MSKST1)	0xA20A_A141	✓	✓
	Reserved	0xA20A_A142 to 0xA20A_A14F	-	-
Mailbox 6 acknowledgment status register (IPCU_MB6ASTR)	CPU0 acknowledgment status bit (ACKST0)	0xA20A_A1C0	✓	✓
	CPU1 acknowledgment status bit (ACKST1)	0xA20A_A1C1	✓	✓
	Reserved	0xA20A_A1C2 to 0xA20A_A1CF	-	-
Mailbox 7 request recipient status register (IPCU_MB7DSTR)	CPU0 request recipient status bit (DSTST0)	0xA20A_A4C0	✓	✓
	CPU1 request recipient status bit (DSTST1)	0xA20A_A4C1	✓	✓
	Reserved	0xA20A_A4C2 to 0xA20A_A4CF	-	-
Mailbox 7 request transmission mask status register (IPCU_MB7MSTR)	CPU0 request transmission mask status bit (MSKST0)	0xA20A_A540	✓	✓
	CPU1 request transmission mask status bit (MSKST1)	0xA20A_A541	✓	✓
	Reserved	0xA20A_A542 to 0xA20A_A54F	-	-
Mailbox 7 acknowledgment status register (IPCU_MB7ASTR)	CPU0 acknowledgment status bit (ACKST0)	0xA20A_A5C0	✓	✓
	CPU1 acknowledgment status bit (ACKST1)	0xA20A_A5C1	✓	✓
	Reserved	0xA20A_A5C2 to 0xA20A_A5CF	-	-

**Note:**

- Do not access the bit-band alias address of the reservation.



## 7) TCFLASH

Register Name	Bit Name	Bit-band Alias Address	Bit Set	Bit Clear
TCFLASH0 status register 0 (TCFCFG0_FSTAT0)	Programming/Erasing ready interrupt (RDYINT)	0xA208_81C8	-	✓
	Hang-up interrupt (HANGINT)	0xA208_81C9	-	✓
TCFLASH0 status register 1 (TCFCFG0_FSTAT1)	Programming/Erasing ready interrupt (RDYINT)	0xA208_81E8	-	✓
	Hang-up interrupt (HANGINT)	0xA208_81E9	-	✓
TCFLASH1 status register 0 (TCFCFG1_FSTAT0)	Programming/Erasing ready interrupt (RDYINT)	0xA208_A1C8	-	✓
	Hang-up interrupt (HANGINT)	0xA208_A1C9	-	✓
TCFLASH1 status register 1 (TCFCFG1_FSTAT1)	Programming/Erasing ready interrupt (RDYINT)	0xA208_A1E8	-	✓
	Hang-up interrupt (HANGINT)	0xA208_A1E9	-	✓

## 8) WorkFLASH

Register Name	Bit Name	Bit-band Alias Address	Bit Set	Bit Clear
WorkFLASH00 bus error response factor register (WFCFG00_BERR)	Uncorrectable error detection (DED)	0xA209_0300	-	✓
	Writing prohibition violation (CRWE)	0xA209_0301	-	✓
	Access size violation (SIZE)	0xA209_0302	-	✓
	Reserved area access (RESA)	0xA209_0305	-	✓
	Unprivileged writing (UNACC)	0xA209_0306	-	✓
	Protection sequence violation (ECRWL)	0xA209_0307	-	✓
	Command overrun (ACCIGN)	0xA209_0308	-	✓
	Writing to the corresponding reserved area in mirror area 2 (WTTM)	0xA209_0309	-	✓
WorkFLASH01 bus error response factor register (WFCFG01_BERR)	Uncorrectable error detection (DED)	0xA209_2300	-	✓
	Writing prohibition violation (CRWE)	0xA209_2301	-	✓
	Access size violation (SIZE)	0xA209_2302	-	✓
	Reserved area access (RESA)	0xA209_2305	-	✓
	Unprivileged writing (UNACC)	0xA209_2306	-	✓
	Protection sequence violation (ECRWL)	0xA209_2307	-	✓
	Command overrun (ACCIGN)	0xA209_2308	-	✓
	Writing to the corresponding reserved area in mirror area 2 (WTTM)	0xA209_2309	-	✓

## 9) CR Calibration

Register Name	Bit Name	Bit-band Alias Address	Bit Set	Bit Clear
Correction unit control register 1 (CU_CUCR1)	Interrupt (INT)	0xA398_0001	-	✓

### 10) GPIO

Register Name	Bit Name	Bit-band Alias Address	Bit Set	Bit Clear
Port output data register (GPIO_PODR0)	Port output data bit (POD[0] to POD[31])	0xA39C_1000 to 0xA39C_101F	✓	✓
Data direction register (GPIO_DDR0)	Data direction selection bit (DD[0] to DD[31])	0xA39C_1020 to 0xA39C_103F	✓	✓
Port output data register (GPIO_PODR1)	Port output data bit (POD[0] to POD[29])	0xA39C_1040 to 0xA39C_105D	✓	✓
Reserved	-	0xA39C_105E	-	-
Port output data register (GPIO_PODR1)	Port output data bit (POD[31])	0xA39C_105F	✓	✓
Data direction register (GPIO_DDR1)	Data direction selection bit (DD[0] to DD[29])	0xA39C_1060 to 0xA39C_107D	✓	✓
Reserved	-	0xA39C_107E	-	-
Data direction register (GPIO_DDR1)	Data direction selection bit (DD[31])	0xA39C_107F	✓	✓
Port output data register (GPIO_PODR2)	Port output data bit (POD[0] to POD[31])	0xA39C_1080 to 0xA39C_109F	✓	✓
Data direction register (GPIO_DDR2)	Data direction selection bit (DD[0] to DD[31])	0xA39C_10A0 to 0xA39C_10BF	✓	✓
Port output data register (GPIO_PODR3)	Port output data bit (POD[0] to POD[6])	0xA39C_10C0 to 0xA39C_10C6	✓	✓
Reserved	-	0xA39C_10C7, 0xA39C_10C8	-	-
Port output data register (GPIO_PODR3)	Port output data bit (POD[9] to POD[30])	0xA39C_10C9 to 0xA39C_10DE	✓	✓
Reserved	-	0xA39C_10DF	-	-
Data direction register (GPIO_DDR3)	Data direction selection bit (DD[0] to DD[6])	0xA39C_10E0 to 0xA39C_10E6	✓	✓
Reserved	-	0xA39C_10E7, 0xA39C_10E8	-	-
Data direction register (GPIO_DDR3)	Data direction selection bit (DD[9] to DD[30])	0xA39C_10E9 to 0xA39C_10FE	✓	✓
Reserved	-	0xA39C_10FF	-	-
Reserved	-	0xA39C_1110 to 0xA39C_1105	-	-
Port output data register (GPIO_PODR4)	Port output data bit (POD[6] to POD[23])	0xA39C_1106 to 0xA39C_1117	✓	✓
Reserved	-	0xA39C_1118	-	-
Port output data register (GPIO_PODR4)	Port output data bit (POD[25] to POD[31])	0xA39C_1119 to 0xA39C_111F	✓	✓
Reserved	-	0xA39C_1120 to 0xA39C_1125	-	-
Data direction register (GPIO_DDR4)	Data direction selection bit (DD[6] to DD[23])	0xA39C_1126 to 0xA39C_1137	✓	✓
Reserved	-	0xA39C_1138	-	-
Data direction register (GPIO_DDR4)	Data direction selection bit (DD[25] to DD[31])	0xA39C_1139 to 0xA39C_113F	✓	✓
Reserved	-	0xA39C_1140 to 0xA39C_13FF	-	-

**Notes:**

- Do not access the bit-band alias address of the reservation.



### 11) Multi-function Serial Interface (UART)

Register Name	Bit Name	Bit-band Alias Address	Bit Set	Bit Clear
Reserved	-	0xA400_0000	-	-
	-	0xA400_0002 to 0xA400_0004	-	-
	-	0xA400_0008 to 0xA400_000F	-	-
	-	0xA400_0011, 0xA400_0012	-	-
	-	0xA400_0014	-	-
	-	0xA400_0016	-	-
	-	0xA400_001C to 0xA400_001F	-	-
Serial auxiliary control status register (MFS00_SACSR0)	Serial timer enable bit (TMRE)	0xA400_0040	✓	✓
Reserved	-	0xA400_0045	-	-
Serial auxiliary control status register (MFS00_SACSR0)	Synchronous transmission enable bit (TSYNE)	0xA400_0046	✓	✓
	Timer interrupt enable bit (TINTE)	0xA400_0047	✓	✓
Serial auxiliary control status register (MFS00_SACSR1)	Timer interrupt flag (TINT)	0xA400_0048	-	✓
Reserved	-	0xA400_004B to 0xA400_004D	-	-
Serial auxiliary control status register (MFS00_SACSR1)	Serial test bit (STST)	0xA400_004F	✓	✓
Reserved	-	0xA400_0090 to 0xA400_0093	-	-
	-	0xA400_0098	-	-
	-	0xA400_009A	-	-
	-	0xA400_00C8	-	-
	-	0xA400_00CA to 0xA400_00CE	-	-
	-	0xA400_00DB to 0xA400_00DE	-	-
FIFO control register 0 (MFS00_FCR0)	FIFO1 operation enable bit (FE1)	0xA400_0100	✓	✓
	FIFO2 operation enable bit (FE2)	0xA400_0101	✓	✓
	FIFO1 reset bit (FCL1)	0xA400_0102	✓	-
	FIFO2 reset bit (FCL2)	0xA400_0103	✓	-
	FIFO pointer saving bit (FSET)	0xA400_0104	✓	-
	FIFO pointer reload bit (FLD)	0xA400_0105	✓	-
FIFO control register 1 (MFS00_FCR1)	FIFO selection bit (FSEL)	0xA400_0108	✓	✓
	Transmission FIFO interrupt enable bit (FTIE)	0xA400_0109	✓	✓
	Transmission FIFO data request bit (FDRQ)	0xA400_010A	✓	✓
	Reception FIFO idle detection enable bit (FRIIE)	0xA400_010B	✓	✓
	Retransmission data lost detection enable bit (FLSTE)	0xA400_010C	✓	✓
Reserved	-	0xA400_2000	-	-
	-	0xA400_2002 to 0xA400_2004	-	-
	-	0xA400_2008 to 0xA400_200F	-	-
	-	0xA400_2011, 0xA400_2012	-	-
	-	0xA400_2014	-	-
	-	0xA400_2016	-	-
	-	0xA400_201C to 0xA400_201F	-	-

Register Name	Bit Name	Bit-band Alias Address	Bit Set	Bit Clear
Serial auxiliary control status register (MFS01_SACSR0)	Serial timer enable bit (TMRE)	0xA400_2040	✓	✓
Reserved	-	0xA400_2045	-	-
Serial auxiliary control status register (MFS01_SACSR0)	Synchronous transmission enable bit (TSYNE)	0xA400_2046	✓	✓
	Timer interrupt enable bit (TINTE)	0xA400_2047	✓	✓
Serial auxiliary control status register (MFS01_SACSR1)	Timer interrupt flag (TINT)	0xA400_2048	-	✓
Reserved	-	0xA400_204B to 0xA400_204D	-	-
Serial auxiliary control status register (MFS01_SACSR1)	Serial test bit (STST)	0xA400_204F	✓	✓
Reserved	-	0xA400_2090 to 0xA400_2093	-	-
	-	0xA400_2098	-	-
	-	0xA400_209A	-	-
	-	0xA400_20C8	-	-
	-	0xA400_20CA to 0xA400_20CE	-	-
	-	0xA400_20DB to 0xA400_20DE	-	-
FIFO control register 0 (MFS01_FCR0)	FIFO1 operation enable bit (FE1)	0xA400_2100	✓	✓
	FIFO2 operation enable bit (FE2)	0xA400_2101	✓	✓
	FIFO1 reset bit (FCL1)	0xA400_2102	✓	-
	FIFO2 reset bit (FCL2)	0xA400_2103	✓	-
	FIFO pointer saving bit (FSET)	0xA400_2104	✓	-
	FIFO pointer reload bit (FLD)	0xA400_2105	✓	-
FIFO control register 1 (MFS01_FCR1)	FIFO selection bit (FSEL)	0xA400_2108	✓	✓
	Transmission FIFO interrupt enable bit (FTIE)	0xA400_2109	✓	✓
	Transmission FIFO data request bit (FDRQ)	0xA400_210A	✓	✓
	Reception FIFO idle detection enable bit (FRIIE)	0xA400_210B	✓	✓
	Retransmission data lost detection enable bit (FLSTE)	0xA400_210C	✓	✓
Reserved	-	0xA400_4000	-	-
	-	0xA400_4002 to 0xA400_4004	-	-
	-	0xA400_4008 to 0xA400_400F	-	-
	-	0xA400_4011, 0xA400_4012	-	-
	-	0xA400_4014	-	-
	-	0xA400_4016	-	-
	-	0xA400_401C to 0xA400_401F	-	-
Serial auxiliary control status register (MFS02_SACSR0)	Serial timer enable bit (TMRE)	0xA400_4040	✓	✓
Reserved	-	0xA400_4045	-	-
Serial auxiliary control status register (MFS02_SACSR0)	Synchronous transmission enable bit (TSYNE)	0xA400_4046	✓	✓
	Timer interrupt enable bit (TINTE)	0xA400_4047	✓	✓
Serial auxiliary control status register (MFS02_SACSR1)	Timer interrupt flag (TINT)	0xA400_4048	-	✓



Register Name	Bit Name	Bit-band Alias Address	Bit Set	Bit Clear
Reserved	-	0xA400_404B to 0xA400_404D	-	-
Serial auxiliary control status register (MFS02_SACSR1)	Serial test bit (STST)	0xA400_404F	✓	✓
Reserved	-	0xA400_4090 to 0xA400_4093	-	-
	-	0xA400_4098	-	-
	-	0xA400_409A	-	-
	-	0xA400_40C8	-	-
	-	0xA400_40CA to 0xA400_40CE	-	-
	-	0xA400_40DB to 0xA400_40DE	-	-
FIFO control register 0 (MFS02_FCR0)	FIFO1 operation enable bit (FE1)	0xA400_4100	✓	✓
	FIFO2 operation enable bit (FE2)	0xA400_4101	✓	✓
	FIFO1 reset bit (FCL1)	0xA400_4102	✓	-
	FIFO2 reset bit (FCL2)	0xA400_4103	✓	-
	FIFO pointer saving bit (FSET)	0xA400_4104	✓	-
	FIFO pointer reload bit (FLD)	0xA400_4105	✓	-
FIFO control register 1 (MFS02_FCR1)	FIFO selection bit (FSEL)	0xA400_4108	✓	✓
	Transmission FIFO interrupt enable bit (FTIE)	0xA400_4109	✓	✓
	Transmission FIFO data request bit (FDRQ)	0xA400_410A	-	✓
	Reception FIFO idle detection enable bit (FRIIE)	0xA400_410B	✓	✓
	Retransmission data lost detection enable bit (FLSTE)	0xA400_410C	✓	✓
Reserved	-	0xA400_6000	-	-
	-	0xA400_6002 to 0xA400_6004	-	-
	-	0xA400_6008 to 0xA400_600F	-	-
	-	0xA400_6011, 0xA400_6012	-	-
	-	0xA400_6014	-	-
	-	0xA400_6016	-	-
Serial auxiliary control status register (MFS03_SACSR0)	-	0xA400_601C to 0xA400_601F	-	-
	Serial timer enable bit (TMRE)	0xA400_6040	✓	✓
	Reserved	0xA400_6045	-	-
	Synchronous transmission enable bit (TSYNE)	0xA400_6046	✓	✓
	Timer interrupt enable bit (TINTE)	0xA400_6047	✓	✓
	Timer interrupt flag (TINT)	0xA400_6048	-	✓
Reserved	-	0xA400_604B to 0xA400_604D	-	-
Serial auxiliary control status register (MFS03_SACSR1)	Serial test bit (STST)	0xA400_604F	✓	✓
Reserved	-	0xA400_6090 to 0xA400_6093	-	-
	-	0xA400_6098	-	-
	-	0xA400_609A	-	-
	-	0xA400_60C8	-	-
	-	0xA400_60CA to 0xA400_60CE	-	-
	-	0xA400_60DB to 0xA400_60DE	-	-

Register Name	Bit Name	Bit-band Alias Address	Bit Set	Bit Clear
FIFO control register 0 (MFS03_FCR0)	FIFO1 operation enable bit (FE1)	0xA400_6100	✓	✓
	FIFO2 operation enable bit (FE2)	0xA400_6101	✓	✓
	FIFO1 reset bit (FCL1)	0xA400_6102	✓	-
	FIFO2 reset bit (FCL2)	0xA400_6103	✓	-
	FIFO pointer saving bit (FSET)	0xA400_6104	✓	-
	FIFO pointer reload bit (FLD)	0xA400_6105	✓	-
FIFO control register 1 (MFS03_FCR1)	FIFO selection bit (FSEL)	0xA400_6108	✓	✓
	Transmission FIFO interrupt enable bit (FTIE)	0xA400_6109	✓	✓
	Transmission FIFO data request bit (FDRQ)	0xA400_610A	-	✓
	Reception FIFO idle detection enable bit (FRIIE)	0xA400_610B	✓	✓
	Retransmission data lost detection enable bit (FLSTE)	0xA400_610C	✓	✓
Reserved	-	0xA400_8000	-	-
	-	0xA400_8002 to 0xA400_8004	-	-
	-	0xA400_8008 to 0xA400_800F	-	-
	-	0xA400_8011, 0xA400_8012	-	-
	-	0xA400_8014	-	-
	-	0xA400_8016	-	-
	-	0xA400_801C to 0xA400_801F	-	-
Serial auxiliary control status register (MFS04_SACSR0)	Serial timer enable bit (TMRE)	0xA400_8040	✓	✓
Reserved	-	0xA400_8045	-	-
Serial auxiliary control status register (MFS04_SACSR0)	Synchronous transmission enable bit (TSYNE)	0xA400_8046	✓	✓
	Timer interrupt enable bit (TINTE)	0xA400_8047	✓	✓
Serial auxiliary control status register (MFS04_SACSR1)	Timer interrupt flag (TINT)	0xA400_8048	-	✓
Reserved	-	0xA400_804B to 0xA400_804D	-	-
Serial auxiliary control status register (MFS04_SACSR1)	Serial test bit (STST)	0xA400_804F	✓	✓
Reserved	-	0xA400_8090 to 0xA400_8093	-	-
	-	0xA400_8098	-	-
	-	0xA400_809A	-	-
	-	0xA400_80C8	-	-
	-	0xA400_80CA to 0xA400_80CE	-	-
	-	0xA400_80DB to 0xA400_80DE	-	-
FIFO control register 0 (MFS04_FCR0)	FIFO1 operation enable bit (FE1)	0xA400_8100	✓	✓
	FIFO2 operation enable bit (FE2)	0xA400_8101	✓	✓
	FIFO1 reset bit (FCL1)	0xA400_8102	✓	-
	FIFO2 reset bit (FCL2)	0xA400_8103	✓	-
	FIFO pointer saving bit (FSET)	0xA400_8104	✓	-
	FIFO pointer reload bit (FLD)	0xA400_8105	✓	-





Register Name	Bit Name	Bit-band Alias Address	Bit Set	Bit Clear
FIFO control register 1 (MFS04_FCR1)	FIFO selection bit (FSEL)	0xA400_8108	✓	✓
	Transmission FIFO interrupt enable bit (FTIE)	0xA400_8109	✓	✓
	Transmission FIFO data request bit (FDRQ)	0xA400_810A	-	✓
	Reception FIFO idle detection enable bit (FRIIE)	0xA400_810B	✓	✓
	Retransmission data lost detection enable bit (FLSTE)	0xA400_810C	✓	✓

**Note:**

- Do not access the bit-band alias address of the reservation.

**12) Multi-function serial interface (CSIO)**

Register Name	Bit Name	Bit-band Alias Address	Bit Set	Bit Clear
Reserved	-	0xA400_0000	-	-
	-	0xA400_0002 to 0xA400_0004	-	-
	-	0xA400_0008 to 0xA400_000F	-	-
	-	0xA400_0011, 0xA400_0012	-	-
	-	0xA400_0014	-	-
	-	0xA400_0016	-	-
	-	0xA400_001C to 0xA400_001F	-	-
Serial auxiliary control status register (MFS00_SACSR0)	Serial timer enable bit (TMRE)	0xA400_0040	✓	✓
Reserved	-	0xA400_0045	-	-
Serial auxiliary control status register (MFS00_SACSR0)	Synchronous transmission enable bit (TSYNE)	0xA400_0046	✓	✓
	Timer interrupt enable bit (TINTE)	0xA400_0047	✓	✓
Serial auxiliary control status register (MFS00_SACSR1)	Timer interrupt flag (TINT)	0xA400_0048	-	✓
	Chip select error flag bit (CSE)	0xA400_004B	-	✓
	Chip select error interrupt enable bit (CSEIE)	0xA400_004C	✓	✓
	Transfer byte error enable bit (TBEEN)	0xA400_004D	✓	✓
	Serial test bit (STST)	0xA400_004F	✓	✓
Reserved	-	0xA400_0090 to 0xA400_0093	-	-
	-	0xA400_0098	-	-
	-	0xA400_009A	-	-
	-	0xA400_00C8	-	-
	-	0xA400_00CA to 0xA400_00CE	-	-
	-	0xA400_00DB to 0xA400_00DE	-	-
FIFO control register 0 (MFS00_FCR0)	FIFO1 operation enable bit (FE1)	0xA400_0100	✓	✓
	FIFO2 operation enable bit (FE2)	0xA400_0101	✓	✓
	FIFO1 reset bit (FCL1)	0xA400_0102	✓	-
	FIFO2 reset bit (FCL2)	0xA400_0103	✓	-
	FIFO pointer saving bit (FSET)	0xA400_0104	✓	-
	FIFO pointer reload bit (FLD)	0xA400_0105	✓	-
FIFO control register 1 (MFS00_FCR1)	FIFO selection bit (FSEL)	0xA400_0108	✓	✓
	Transmission FIFO interrupt enable bit (FTIE)	0xA400_0109	✓	✓
	Transmission FIFO data request bit (FDRQ)	0xA400_010A	-	✓
	Reception FIFO idle detection enable bit (FRIIE)	0xA400_010B	✓	✓
	Retransmission data lost detection enable bit (FLSTE)	0xA400_010C	✓	✓

Register Name	Bit Name	Bit-band Alias Address	Bit Set	Bit Clear
Reserved	-	0xA400_2000	-	-
	-	0xA400_2002 to 0xA400_2004	-	-
	-	0xA400_2008 to 0xA400_200F	-	-
	-	0xA400_2011, 0xA400_2012	-	-
	-	0xA400_2014	-	-
	-	0xA400_2016	-	-
	-	0xA400_201C to 0xA400_201F	-	-
Serial auxiliary control status register (MFS01_SACSR0)	Serial timer enable bit (TMRE)	0xA400_2040	✓	✓
Reserved	-	0xA400_2045	-	-
Serial auxiliary control status register (MFS01_SACSR0)	Synchronous transmission enable bit (TSYNE)	0xA400_2046	✓	✓
	Timer interrupt enable bit (TINTE)	0xA400_2047	✓	✓
Serial auxiliary control status register (MFS01_SACSR1)	Timer interrupt flag (TINT)	0xA400_2048	-	✓
	Chip select error flag bit (CSE)	0xA400_204B	-	✓
	Chip select error interrupt enable bit (CSEIE)	0xA400_204C	✓	✓
	Transfer byte error enable bit (TBEEN)	0xA400_204D	✓	✓
	Serial test bit (STST)	0xA400_204F	✓	✓
Reserved	-	0xA400_2090 to 0xA400_2093	-	-
	-	0xA400_2098	-	-
	-	0xA400_209A	-	-
	-	0xA400_20C8	-	-
	-	0xA400_20CA to 0xA400_20CE	-	-
	-	0xA400_20DB to 0xA400_20DE	-	-
FIFO control register 0 (MFS01_FCR0)	FIFO1 operation enable bit (FE1)	0xA400_2100	✓	✓
	FIFO2 operation enable bit (FE2)	0xA400_2101	✓	✓
	FIFO1 reset bit (FCL1)	0xA400_2102	✓	-
	FIFO2 reset bit (FCL2)	0xA400_2103	✓	-
	FIFO pointer saving bit (FSET)	0xA400_2104	✓	-
	FIFO pointer reload bit (FLD)	0xA400_2105	✓	-
FIFO control register 1 (MFS01_FCR1)	FIFO selection bit (FSEL)	0xA400_2108	✓	✓
	Transmission FIFO interrupt enable bit (FTIE)	0xA400_2109	✓	✓
	Transmission FIFO data request bit (FDRQ)	0xA400_210A	-	✓
	Reception FIFO idle detection enable bit (FRIIE)	0xA400_210B	✓	✓
	Retransmission data lost detection enable bit (FLSTE)	0xA400_210C	✓	✓
Reserved	-	0xA400_4000	-	-
	-	0xA400_4002 to 0xA400_4004	-	-
	-	0xA400_4008 to 0xA400_400F	-	-
	-	0xA400_4011, 0xA400_4012	-	-
	-	0xA400_4014	-	-
	-	0xA400_4016	-	-
	-	0xA400_401C to 0xA400_401F	-	-
Serial auxiliary control status register (MFS02_SACSR0)	Serial timer enable bit (TMRE)	0xA400_4040	✓	✓



Register Name	Bit Name	Bit-band Alias Address	Bit Set	Bit Clear
Reserved	-	0xA400_4045	-	-
Serial auxiliary control status register (MFS02_SACSR0)	Synchronous transmission enable bit (TSYNE)	0xA400_4046	✓	✓
	Timer interrupt enable bit (TINTE)	0xA400_4047	✓	✓
Serial auxiliary control status register (MFS02_SACSR1)	Timer interrupt flag (TINT)	0xA400_4048	-	✓
	Chip select error flag bit (CSE)	0xA400_404B	-	✓
	Chip select error interrupt enable bit (CSEIE)	0xA400_404C	✓	✓
	Transfer byte error enable bit (TBEEN)	0xA400_404D	✓	✓
	Serial test bit (STST)	0xA400_404F	✓	✓
Reserved	-	0xA400_4090 to 0xA400_4093	-	-
	-	0xA400_4098	-	-
	-	0xA400_409A	-	-
	-	0xA400_40C8	-	-
	-	0xA400_40CA to 0xA400_40CE	-	-
	-	0xA400_40DB to 0xA400_40DE	-	-
FIFO control register 0 (MFS02_FCR0)	FIFO1 operation enable bit (FE1)	0xA400_4100	✓	✓
	FIFO2 operation enable bit (FE2)	0xA400_4101	✓	✓
	FIFO1 reset bit (FCL1)	0xA400_4102	✓	-
	FIFO2 reset bit (FCL2)	0xA400_4103	✓	-
	FIFO pointer saving bit (FSET)	0xA400_4104	✓	-
	FIFO pointer reload bit (FLD)	0xA400_4105	✓	-
FIFO control register 1 (MFS02_FCR1)	FIFO selection bit (FSEL)	0xA400_4108	✓	✓
	Transmission FIFO interrupt enable bit (FTIE)	0xA400_4109	✓	✓
	Transmission FIFO data request bit (FDRQ)	0xA400_410A	-	✓
	Reception FIFO idle detection enable bit (FRIIE)	0xA400_410B	✓	✓
	Retransmission data lost detection enable bit (FLSTE)	0xA400_410C	✓	✓
Reserved	-	0xA400_6000	-	-
	-	0xA400_6002 to 0xA400_6004	-	-
	-	0xA400_6008 to 0xA400_600F	-	-
	-	0xA400_6011, 0xA400_6012	-	-
	-	0xA400_6014	-	-
	-	0xA400_6016	-	-
	-	0xA400_601C to 0xA400_601F	-	-
Serial auxiliary control status register (MFS03_SACSR0)	Serial timer enable bit (TMRE)	0xA400_6040	✓	✓
Reserved	-	0xA400_6045	-	-
Serial auxiliary control status register (MFS03_SACSR0)	Synchronous transmission enable bit (TSYNE)	0xA400_6046	✓	✓
	Timer interrupt enable bit (TINTE)	0xA400_6047	✓	✓
Serial auxiliary control status register (MFS03_SACSR1)	Timer interrupt flag (TINT)	0xA400_6048	-	✓
	Chip select error flag bit (CSE)	0xA400_604B	-	✓
	Chip select error interrupt enable bit (CSEIE)	0xA400_604C	✓	✓
	Transfer byte error enable bit (TBEEN)	0xA400_604D	✓	✓
	Serial test bit (STST)	0xA400_604F	✓	✓

Register Name	Bit Name	Bit-band Alias Address	Bit Set	Bit Clear
Reserved	-	0xA400_6090 to 0xA400_6093	-	-
	-	0xA400_6098	-	-
	-	0xA400_609A	-	-
	-	0xA400_60C8	-	-
	-	0xA400_60CA to 0xA400_60CE	-	-
	-	0xA400_60DB to 0xA400_60DE	-	-
FIFO control register 0 (MFS03_FCR0)	FIFO1 operation enable bit (FE1)	0xA400_6100	✓	✓
	FIFO2 operation enable bit (FE2)	0xA400_6101	✓	✓
	FIFO1 reset bit (FCL1)	0xA400_6102	✓	-
	FIFO2 reset bit (FCL2)	0xA400_6103	✓	-
	FIFO pointer saving bit (FSET)	0xA400_6104	✓	-
	FIFO pointer reload bit (FLD)	0xA400_6105	✓	-
FIFO control register 1 (MFS03_FCR1)	FIFO selection bit (FSEL)	0xA400_6108	✓	✓
	Transmission FIFO interrupt enable bit (FTIE)	0xA400_6109	✓	✓
	Transmission FIFO data request bit (FDRQ)	0xA400_610A	-	✓
	Reception FIFO idle detection enable bit (FRIIE)	0xA400_610B	✓	✓
	Retransmission data lost detection enable bit (FLSTE)	0xA400_610C	✓	✓
Reserved	-	0xA400_8000	-	-
	-	0xA400_8002 to 0xA400_8004	-	-
	-	0xA400_8008 to 0xA400_800F	-	-
	-	0xA400_8011, 0xA400_8012	-	-
	-	0xA400_8014	-	-
	-	0xA400_8016	-	-
	-	0xA400_801C to 0xA400_801F	-	-
Serial auxiliary control status register (MFS04_SACSR0)	Serial timer enable bit (TMRE)	0xA400_8040	✓	✓
Reserved	-	0xA400_8045	-	-
Serial auxiliary control status register (MFS04_SACSR0)	Synchronous transmission enable bit (TSYNE)	0xA400_8046	✓	✓
	Timer interrupt enable bit (TINTE)	0xA400_8047	✓	✓
Serial auxiliary control status register (MFS04_SACSR1)	Timer interrupt flag (TINT)	0xA400_8048	-	✓
	Chip select error flag bit (CSE)	0xA400_804B	-	✓
	Chip select error interrupt enable bit (CSEIE)	0xA400_804C	✓	✓
	Transfer byte error enable bit (TBEEN)	0xA400_804D	✓	✓
	Serial test bit (STST)	0xA400_804F	✓	✓
Reserved	-	0xA400_8090 to 0xA400_8093	-	-
	-	0xA400_8098	-	-
	-	0xA400_809A	-	-
	-	0xA400_80C8	-	-
	-	0xA400_80CA to 0xA400_80CE	-	-
	-	0xA400_80DB to 0xA400_80DE	-	-
FIFO control register 0 (MFS04_FCR0)	FIFO1 operation enable bit (FE1)	0xA400_8100	✓	✓
	FIFO2 operation enable bit (FE2)	0xA400_8101	✓	✓
	FIFO1 reset bit (FCL1)	0xA400_8102	✓	-
	FIFO2 reset bit (FCL2)	0xA400_8103	✓	-
	FIFO pointer saving bit (FSET)	0xA400_8104	✓	-
	FIFO pointer reload bit (FLD)	0xA400_8105	✓	-



Register Name	Bit Name	Bit-band Alias Address	Bit Set	Bit Clear
FIFO control register 1 (MFS04_FCR1)	FIFO selection bit (FSEL)	0xA400_8108	✓	✓
	Transmission FIFO interrupt enable bit (FTIE)	0xA400_8109	✓	✓
	Transmission FIFO data request bit (FDRQ)	0xA400_810A	-	✓
	Reception FIFO idle detection enable bit (FRIIE)	0xA400_810B	✓	✓
	Retransmission data lost detection enable bit (FLSTE)	0xA400_810C	✓	✓

**Note:**

- Do not access the bit-band alias address of the reservation.

### 13) Multi-function serial interface (LIN interface)

Register Name	Bit Name	Bit-band Alias Address	Bit Set	Bit Clear
Serial mode register (MFS00_SMR)	Serial data output enable bit (SOE)	0xA400_0000	✓	✓
Reserved	-	0xA400_0002	-	-
Serial mode register (MFS00_SMR)	Stop bit length selection bit (SBL)	0xA400_0003	✓	✓
	Reserved	0xA400_0004	-	-
Serial control register (MFS00_SCR)	Transmission operation enable bit (TXE)	0xA400_0008	✓	✓
	Reception operation enable bit (RXE)	0xA400_0009	✓	✓
	Transmission bus idle interrupt enable bit (TBIE)	0xA400_000A	✓	✓
	Transmission interrupt enable bit (TIE)	0xA400_000B	✓	✓
	Reception interrupt enable bit (RIE)	0xA400_000C	✓	✓
	LIN break field setting bit (LBR)	0xA400_000D	✓	-
	Master/Slave function select bit (MS)	0xA400_000E	✓	✓
	Programmable clear bit (UPCL)	0xA400_000F	✓	-
Reserved	-	0xA400_0011, 0xA400_0012	-	-
Extended communication control register (MFS00_ESCR)	LIN break field detection interrupt enable bit (LBIE)	0xA400_0014	✓	✓
	Extended stop bit length selection bit (ESBL)	0xA400_0016	✓	✓
Reserved	-	0xA400_001C	-	-
Serial status register (MFS00_SSR)	LIN break field detection flag bit (LBD)	0xA400_001D	-	✓
Reserved	-	0xA400_001E	-	-
Serial status register (MFS00_SSR)	Reception error flag clear bit (REC)	0xA400_001F	✓	-
Serial auxiliary control status register (MFS00_SACSR0)	Serial timer enable bit (TMRE)	0xA400_0040	✓	✓
Reserved	-	0xA400_0045, 0xA400_0046	-	-
Serial auxiliary control status register (MFS00_SACSR0)	Timer interrupt enable bit (TINTE)	0xA400_0047	✓	✓
Serial auxiliary control status register (MFS00_SACSR1)	Timer interrupt flag (TINT)	0xA400_0048	-	✓
	Chip select error flag bit (CSE)	0xA400_004B	-	✓
	Chip select error interrupt enable bit (CSEIE)	0xA400_004C	✓	✓
	Transfer byte error enable bit (TBEEN)	0xA400_004D	✓	✓
	Serial test bit (STST)	0xA400_004F	✓	✓
LIN assist mode control register (MFS00_LAMCR)	LIN assist mode processing enable bit (LAMEN)	0xA400_0090	✓	✓
	LIN ID register use enable bit (LIDEN)	0xA400_0091	✓	✓
	LIN checksum type selection bit (LCSTYP)	0xA400_0092	✓	✓
	LIN transmission data register clear bit (LTDRCL)	0xA400_0093	✓	-
LIN assist mode status register (MFS00_LAMSR)	LIN auto header completion flag bit (LAHC)	0xA400_0098	-	✓
	LIN checksum calculation completion flag bit (LCSC)	0xA400_009A	-	✓



Register Name	Bit Name	Bit-band Alias Address	Bit Set	Bit Clear
LIN assist mode interrupt enable register (MFS00_LAMIER)	LIN auto header completion interrupt enable bit (LAHCIE)	0xA400_00C8	✓	✓
	LIN checksum calculation completion interrupt enable bit (LCSCIE)	0xA400_00CA	✓	✓
	LIN bus error interrupt enable bit (LBSERIE)	0xA400_00CB	✓	✓
	LIN sync data error interrupt enable bit (LSFERIE)	0xA400_00CC	✓	✓
	LIN ID parity error interrupt enable bit (LPTERIE)	0xA400_00CD	✓	✓
	LIN checksum error interrupt enable bit (LCSERIE)	0xA400_00CE	✓	✓
LIN assist mode error status register (MFS00_LAMESR)	LIN bus error flag bit (LBSER)	0xA400_00DB	-	✓
	LIN sync data error flag bit (LSFER)	0xA400_00DC	-	✓
	LIN ID parity error flag bit (LPTER)	0xA400_00DD	-	✓
	LIN checksum error flag bit (LCSER)	0xA400_00DE	-	✓
FIFO control register 0 (MFS00_FCR0)	FIFO1 operation enable bit (FE1)	0xA400_0100	✓	✓
	FIFO2 operation enable bit (FE2)	0xA400_0101	✓	✓
	FIFO1 reset bit (FCL1)	0xA400_0102	✓	-
	FIFO2 reset bit (FCL2)	0xA400_0103	✓	-
	FIFO pointer saving bit (FSET)	0xA400_0104	✓	-
	FIFO pointer reload bit (FLD)	0xA400_0105	✓	-
FIFO control register 1 (MFS00_FCR1)	FIFO selection bit (FSEL)	0xA400_0108	✓	✓
	Transmission FIFO interrupt enable bit (FTIE)	0xA400_0109	✓	✓
	Transmission FIFO data request bit (FDRQ)	0xA400_010A	-	✓
	Reception FIFO idle detection enable bit (FRIIE)	0xA400_010B	✓	✓
	Retransmission data lost detection enable bit (FLSTE)	0xA400_010C	✓	✓
Serial mode register (MFS01_SMR)	Serial data output enable bit (SOE)	0xA400_2000	✓	✓
Reserved	-	0xA400_2002	-	-
Serial mode register (MFS01_SMR)	Stop bit length selection bit (SBL)	0xA400_2003	✓	✓
	Reserved	0xA400_2004	-	-
Serial control register (MFS01_SCR)	Transmission operation enable bit (TXE)	0xA400_2008	✓	✓
	Reception operation enable bit (RXE)	0xA400_2009	✓	✓
	Transmission bus idle interrupt enable bit (TBIE)	0xA400_200A	✓	✓
	Transmission interrupt enable bit (TIE)	0xA400_200B	✓	✓
	Reception interrupt enable bit (RIE)	0xA400_200C	✓	✓
	LIN break field setting bit (LBR)	0xA400_200D	✓	-
	Master/Slave function select bit (MS)	0xA400_200E	✓	✓
	Programmable clear bit (UPCL)	0xA400_200F	✓	-
Reserved	-	0xA400_2011, 0xA400_2012	-	-
Extended communication control register (MFS01_ESCR)	LIN break field detection interrupt enable bit (LBIE)	0xA400_2014	✓	✓
	Extended stop bit length selection bit (ESBL)	0xA400_2016	✓	✓
Reserved	-	0xA400_201C	-	-
Serial status register (MFS01_SSR)	LIN break field detection flag bit (LBD)	0xA400_201D	-	✓
Reserved	-	0xA400_201E	-	-
Serial status register (MFS01_SSR)	Reception error flag clear bit (REC)	0xA400_201F	✓	-

Register Name	Bit Name	Bit-band Alias Address	Bit Set	Bit Clear
Serial auxiliary control status register (MFS01_SACSR0)	Serial timer enable bit (TMRE)	0xA400_2040	✓	✓
Reserved	-	0xA400_2045, 0xA400_2046	-	-
Serial auxiliary control status register (MFS01_SACSR0)	Timer interrupt enable bit (TINTE)	0xA400_2047	✓	✓
Serial auxiliary control status register (MFS01_SACSR1)	Timer interrupt flag (TINT)	0xA400_2048	-	✓
	Chip select error flag bit (CSE)	0xA400_204B	-	✓
	Chip select error interrupt enable bit (CSEIE)	0xA400_204C	✓	✓
	Transfer byte error enable bit (TBEEN)	0xA400_204D	✓	✓
	Serial test bit (STST)	0xA400_204F	✓	✓
LIN assist mode control register (MFS01_LAMCR)	LIN assist mode processing enable bit (LAMEN)	0xA400_2090	✓	✓
	LIN ID register use enable bit (LIDEN)	0xA400_2091	✓	✓
	LIN checksum type selection bit (LCSTYP)	0xA400_2092	✓	✓
	LIN transmission data register clear bit (LTDRCL)	0xA400_2093	✓	-
LIN assist mode status register (MFS01_LAMSR)	LIN auto header completion flag bit (LAHC)	0xA400_2098	-	✓
	LIN checksum calculation completion flag bit (LCSC)	0xA400_209A	-	✓
LIN assist mode interrupt enable register (MFS01_LAMIER)	LIN auto header completion interrupt enable bit (LAHCIE)	0xA400_20C8	✓	✓
	LIN checksum calculation completion interrupt enable bit (LCSCIE)	0xA400_20CA	✓	✓
	LIN bus error interrupt enable bit (LBSERIE)	0xA400_20CB	✓	✓
	LIN sync data error interrupt enable bit (LSFERIE)	0xA400_20CC	✓	✓
	LIN ID parity error interrupt enable bit (LPTERIE)	0xA400_20CD	✓	✓
	LIN checksum error interrupt enable bit (LCSERIE)	0xA400_20CE	✓	✓
LIN assist mode error status register (MFS01_LAMESR)	LIN bus error flag bit (LBSER)	0xA400_20DB	-	✓
	LIN sync data error flag bit (LSFER)	0xA400_20DC	-	✓
	LIN ID parity error flag bit (LPTER)	0xA400_20DD	-	✓
	LIN checksum error flag bit (LCSER)	0xA400_20DE	-	✓
FIFO control register 0 (MFS01_FCR0)	FIFO1 operation enable bit (FE1)	0xA400_2100	✓	✓
	FIFO2 operation enable bit (FE2)	0xA400_2101	✓	✓
	FIFO1 reset bit (FCL1)	0xA400_2102	✓	-
	FIFO2 reset bit (FCL2)	0xA400_2103	✓	-
	FIFO pointer saving bit (FSET)	0xA400_2104	✓	-
	FIFO pointer reload bit (FLD)	0xA400_2105	✓	-
FIFO control register 1 (MFS01_FCR1)	FIFO selection bit (FSEL)	0xA400_2108	✓	✓
	Transmission FIFO interrupt enable bit (FTIE)	0xA400_2109	✓	✓
	Transmission FIFO data request bit (FDRQ)	0xA400_210A	-	✓
	Reception FIFO idle detection enable bit (FRIIE)	0xA400_210B	✓	✓
	Retransmission data lost detection enable bit (FLSTE)	0xA400_210C	✓	✓
Serial mode register (MFS02_SMR)	Serial data output enable bit (SOE)	0xA400_4000	✓	✓
Reserved	-	0xA400_4002	-	-
Serial mode register (MFS02_SMR)	Stop bit length selection bit (SBL)	0xA400_4003	✓	✓
	Reserved	0xA400_4004	-	-





Register Name	Bit Name	Bit-band Alias Address	Bit Set	Bit Clear
Serial control register (MFS02_SCR)	Transmission operation enable bit (TXE)	0xA400_4008	✓	✓
	Reception operation enable bit (RXE)	0xA400_4009	✓	✓
	Transmission bus idle interrupt enable bit (TBIE)	0xA400_400A	✓	✓
	Transmission interrupt enable bit (TIE)	0xA400_400B	✓	✓
	Reception interrupt enable bit (RIE)	0xA400_400C	✓	✓
	LIN break field setting bit (LBR)	0xA400_400D	✓	-
	Master/Slave function select bit (MS)	0xA400_400E	✓	✓
	Programmable clear bit (UPCL)	0xA400_400F	✓	-
Reserved	-	0xA400_4011, 0xA400_4012	-	-
Extended communication control register (MFS02_ESCR)	LIN break field detection interrupt enable bit (LBIE)	0xA400_4014	✓	✓
	Extended stop bit length selection bit (ESBL)	0xA400_4016	✓	✓
Reserved	-	0xA400_401C	-	-
Serial status register (MFS02_SSR)	LIN break field detection flag bit (LBD)	0xA400_401D	-	✓
Reserved	-	0xA400_401E	-	-
Serial status register (MFS02_SSR)	Reception error flag clear bit (REC)	0xA400_401F	✓	-
Serial auxiliary control status register (MFS02_SACSR0)	Serial timer enable bit (TMRE)	0xA400_4040	✓	✓
Reserved	-	0xA400_4045, 0xA400_4046	-	-
Serial auxiliary control status register (MFS02_SACSR0)	Timer interrupt enable bit (TINTE)	0xA400_4047	✓	✓
Serial auxiliary control status register (MFS02_SACSR1)	Timer interrupt flag (TINT)	0xA400_4048	-	✓
	Chip select error flag bit (CSE)	0xA400_404B	-	✓
	Chip select error interrupt enable bit (CSEIE)	0xA400_404C	✓	✓
	Transfer byte error enable bit (TBEEN)	0xA400_404D	✓	✓
	Serial test bit (STST)	0xA400_404F	✓	✓
LIN assist mode control register (MFS02_LAMCR)	LIN assist mode processing enable bit (LAMEN)	0xA400_4090	✓	✓
	LIN ID register use enable bit (LIDEN)	0xA400_4091	✓	✓
	LIN checksum type selection bit (LCSTYP)	0xA400_4092	✓	✓
	LIN transmission data register clear bit (LTDRCL)	0xA400_4093	✓	-
LIN assist mode status register (MFS02_LAMSR)	LIN auto header completion flag bit (LAHC)	0xA400_4098	-	✓
	LIN checksum calculation completion flag bit (LCSC)	0xA400_409A	-	✓
LIN assist mode interrupt enable register (MFS02_LAMIER)	LIN auto header completion interrupt enable bit (LAHCIE)	0xA400_40C8	✓	✓
	LIN checksum calculation completion interrupt enable bit (LCSCIE)	0xA400_40CA	✓	✓
	LIN bus error interrupt enable bit (LBSERIE)	0xA400_40CB	✓	✓
	LIN sync data error interrupt enable bit (LSFERIE)	0xA400_40CC	✓	✓
	LIN ID parity error interrupt enable bit (LPTERIE)	0xA400_40CD	✓	✓
	LIN checksum error interrupt enable bit (LCSERIE)	0xA400_40CE	✓	✓
LIN assist mode error status register (MFS02_LAMESR)	LIN bus error flag bit (LBSEIR)	0xA400_40DB	-	✓
	LIN sync data error flag bit (LSFER)	0xA400_40DC	-	✓
	LIN ID parity error flag bit (LPTER)	0xA400_40DD	-	✓
	LIN checksum error flag bit (LCSEIR)	0xA400_40DE	-	✓

Register Name	Bit Name	Bit-band Alias Address	Bit Set	Bit Clear
FIFO control register 0 (MFS02_FCR0)	FIFO1 operation enable bit (FE1)	0xA400_4100	✓	✓
	FIFO2 operation enable bit (FE2)	0xA400_4101	✓	✓
	FIFO1 reset bit (FCL1)	0xA400_4102	✓	-
	FIFO2 reset bit (FCL2)	0xA400_4103	✓	-
	FIFO pointer saving bit (FSET)	0xA400_4104	✓	-
	FIFO pointer reload bit (FLD)	0xA400_4105	✓	-
FIFO control register 1 (MFS02_FCR1)	FIFO selection bit (FSEL)	0xA400_4108	✓	✓
	Transmission FIFO interrupt enable bit (FTIE)	0xA400_4109	✓	✓
	Transmission FIFO data request bit (FDRQ)	0xA400_410A	-	✓
	Reception FIFO idle detection enable bit (FRIIE)	0xA400_410B	✓	✓
	Retransmission data lost detection enable bit (FLSTE)	0xA400_410C	✓	✓
Serial mode register (MFS03_SMR)	Serial data output enable bit (SOE)	0xA400_6000	✓	✓
Reserved	-	0xA400_6002	-	-
Serial mode register (MFS03_SMR)	Stop bit length selection bit (SBL)	0xA400_6003	✓	✓
	Reserved	0xA400_6004	-	-
Serial control register (MFS03_SCR)	Transmission operation enable bit (TXE)	0xA400_6008	✓	✓
	Reception operation enable bit (RXE)	0xA400_6009	✓	✓
	Transmission bus idle interrupt enable bit (TBIE)	0xA400_600A	✓	✓
	Transmission interrupt enable bit (TIE)	0xA400_600B	✓	✓
	Reception interrupt enable bit (RIE)	0xA400_600C	✓	✓
	LIN break field setting bit (LBR)	0xA400_600D	✓	-
	Master/Slave function select bit (MS)	0xA400_600E	✓	✓
	Programmable clear bit (UPCL)	0xA400_600F	✓	-
Reserved	-	0xA400_6011, 0xA400_6012	-	-
Extended communication control register (MFS03_ESCR)	LIN break field detection interrupt enable bit (LBIE)	0xA400_6014	✓	✓
	Extended stop bit length selection bit (ESBL)	0xA400_6016	✓	✓
Reserved	-	0xA400_601C	-	-
Serial status register (MFS03_SSR)	LIN break field detection flag bit (LBD)	0xA400_601D	-	✓
Reserved	-	0xA400_601E	-	-
Serial status register (MFS03_SSR)	Reception error flag clear bit (REC)	0xA400_601F	✓	-
Serial auxiliary control status register (MFS03_SACSR0)	Serial timer enable bit (TMRE)	0xA400_6040	✓	✓
Reserved	-	0xA400_6045, 0xA400_6046	-	-
Serial auxiliary control status register (MFS03_SACSR0)	Timer interrupt enable bit (TINTE)	0xA400_6047	✓	✓
Serial auxiliary control status register (MFS03_SACSR1)	Timer interrupt flag (TINT)	0xA400_6048	-	✓
	Chip select error flag bit (CSE)	0xA400_604B	-	✓
	Chip select error interrupt enable bit (CSEIE)	0xA400_604C	✓	✓
	Transfer byte error enable bit (TBEEN)	0xA400_604D	✓	✓
	Serial test bit (STST)	0xA400_604F	✓	✓



Register Name	Bit Name	Bit-band Alias Address	Bit Set	Bit Clear
LIN assist mode control register (MFS03_LAMCR)	LIN assist mode processing enable bit (LAMEN)	0xA400_6090	✓	✓
	LIN ID register use enable bit (LIDEN)	0xA400_6091	✓	✓
	LIN checksum type selection bit (LCSTYP)	0xA400_6092	✓	✓
	LIN transmission data register clear bit (LTDRCL)	0xA400_6093	✓	-
LIN assist mode status register (MFS03_LAMSR)	LIN auto header completion flag bit (LAHC)	0xA400_6098	-	✓
	LIN checksum calculation completion flag bit (LCSC)	0xA400_609A	-	✓
LIN assist mode interrupt enable register (MFS03_LAMIER)	LIN auto header completion interrupt enable bit (LAHCIE)	0xA400_60C8	✓	✓
	LIN checksum calculation completion interrupt enable bit (LCSCIE)	0xA400_60CA	✓	✓
	LIN bus error interrupt enable bit (LBSERIE)	0xA400_60CB	✓	✓
	LIN sync data error interrupt enable bit (LSFERIE)	0xA400_60CC	✓	✓
	LIN ID parity error interrupt enable bit (LPTERIE)	0xA400_60CD	✓	✓
	LIN checksum error interrupt enable bit (LCSERIE)	0xA400_60CE	✓	✓
LIN assist mode error status register (MFS03_LAMESR)	LIN bus error flag bit (LBSER)	0xA400_60DB	-	✓
	LIN sync data error flag bit (LSFER)	0xA400_60DC	-	✓
	LIN ID parity error flag bit (LPTER)	0xA400_60DD	-	✓
	LIN checksum error flag bit (LCSER)	0xA400_60DE	-	✓
FIFO control register 0 (MFS03_FCR0)	FIFO1 operation enable bit (FE1)	0xA400_6100	✓	✓
	FIFO2 operation enable bit (FE2)	0xA400_6101	✓	✓
	FIFO1 reset bit (FCL1)	0xA400_6102	✓	-
	FIFO2 reset bit (FCL2)	0xA400_6103	✓	-
	FIFO pointer saving bit (FSET)	0xA400_6104	✓	-
	FIFO pointer reload bit (FLD)	0xA400_6105	✓	-
FIFO control register 1 (MFS03_FCR1)	FIFO selection bit (FSEL)	0xA400_6108	✓	✓
	Transmission FIFO interrupt enable bit (FTIE)	0xA400_6109	✓	✓
	Transmission FIFO data request bit (FDRQ)	0xA400_610A	-	✓
	Reception FIFO idle detection enable bit (FRIIE)	0xA400_610B	✓	✓
	Retransmission data lost detection enable bit (FLSTE)	0xA400_610C	✓	✓
Serial mode register (MFS04_SMR)	Serial data output enable bit (SOE)	0xA400_8000	✓	✓
Reserved	-	0xA400_8002	-	-
Serial mode register (MFS04_SMR)	Stop bit length selection bit (SBL)	0xA400_8003	✓	✓
	Reserved	0xA400_8004	-	-
Serial control register (MFS04_SCR)	Transmission operation enable bit (TXE)	0xA400_8008	✓	✓
	Reception operation enable bit (RXE)	0xA400_8009	✓	✓
	Transmission bus idle interrupt enable bit (TBIE)	0xA400_800A	✓	✓
	Transmission interrupt enable bit (TIE)	0xA400_800B	✓	✓
	Reception interrupt enable bit (RIE)	0xA400_800C	✓	✓
	LIN break field setting bit (LBR)	0xA400_800D	✓	-
	Master/Slave function select bit (MS)	0xA400_800E	✓	✓
	Programmable clear bit (UPCL)	0xA400_800F	✓	-
Reserved	-	0xA400_8011, 0xA400_8012	-	-
Extended communication control register (MFS04_ESCR)	LIN break field detection interrupt enable bit (LBIE)	0xA400_8014	✓	✓
	Extended stop bit length selection bit (ESBL)	0xA400_8016	✓	✓
Reserved	-	0xA400_801C	-	-

Register Name	Bit Name	Bit-band Alias Address	Bit Set	Bit Clear
Serial status register (MFS04_SSR)	LIN break field detection flag bit (LBD)	0xA400_801D	-	✓
Reserved	-	0xA400_801E	-	-
Serial status register (MFS04_SSR)	Reception error flag clear bit (REC)	0xA400_801F	✓	-
Serial auxiliary control status register (MFS04_SACSR0)	Serial timer enable bit (TMRE)	0xA400_8040	✓	✓
Reserved	-	0xA400_8045, 0xA400_8046	-	-
Serial auxiliary control status register (MFS04_SACSR0)	Timer interrupt enable bit (TINTE)	0xA400_8047	✓	✓
Serial auxiliary control status register (MFS04_SACSR1)	Timer interrupt flag (TINT)	0xA400_8048	-	✓
	Chip select error flag bit (CSE)	0xA400_804B	-	✓
	Chip select error interrupt enable bit (CSEIE)	0xA400_804C	✓	✓
	Transfer byte error enable bit (TBEEN)	0xA400_804D	✓	✓
	Serial test bit (STST)	0xA400_804F	✓	✓
LIN assist mode control register (MFS04_LAMCR)	LIN assist mode processing enable bit (LAMEN)	0xA400_8090	✓	✓
	LIN ID register use enable bit (LIDEN)	0xA400_8091	✓	✓
	LIN checksum type selection bit (LCSTYP)	0xA400_8092	✓	✓
	LIN transmission data register clear bit (LTDRCL)	0xA400_8093	✓	-
LIN assist mode status register (MFS04_LAMSR)	LIN auto header completion flag bit (LAHC)	0xA400_8098	-	✓
	LIN checksum calculation completion flag bit (LCSC)	0xA400_809A	-	✓
LIN assist mode interrupt enable register (MFS04_LAMIER)	LIN auto header completion interrupt enable bit (LAHCIE)	0xA400_80C8	✓	✓
	LIN checksum calculation completion interrupt enable bit (LCSCIE)	0xA400_80CA	✓	✓
	LIN bus error interrupt enable bit (LBSERIE)	0xA400_80CB	✓	✓
	LIN sync data error interrupt enable bit (LSFERIE)	0xA400_80CC	✓	✓
	LIN ID parity error interrupt enable bit (LPTERIE)	0xA400_80CD	✓	✓
	LIN checksum error interrupt enable bit (LCSERIE)	0xA400_80CE	✓	✓
LIN assist mode error status register (MFS04_LAMESR)	LIN bus error flag bit (LBSEF)	0xA400_80DB	-	✓
	LIN sync data error flag bit (LSFER)	0xA400_80DC	-	✓
	LIN ID parity error flag bit (LPTER)	0xA400_80DD	-	✓
	LIN checksum error flag bit (LCSEF)	0xA400_80DE	-	✓
FIFO control register 0 (MFS04_FCR0)	FIFO1 operation enable bit (FE1)	0xA400_8100	✓	✓
	FIFO2 operation enable bit (FE2)	0xA400_8101	✓	✓
	FIFO1 reset bit (FCL1)	0xA400_8102	✓	-
	FIFO2 reset bit (FCL2)	0xA400_8103	✓	-
	FIFO pointer saving bit (FSET)	0xA400_8104	✓	-
	FIFO pointer reload bit (FLD)	0xA400_8105	✓	-
FIFO control register 1 (MFS04_FCR1)	FIFO selection bit (FSEL)	0xA400_8108	✓	✓
	Transmission FIFO interrupt enable bit (FTIE)	0xA400_8109	✓	✓
	Transmission FIFO data request bit (FDRQ)	0xA400_810A	-	✓
	Reception FIFO idle detection enable bit (FRIIE)	0xA400_810B	✓	✓
	Retransmission data lost detection enable bit (FLSTE)	0xA400_810C	✓	✓



**Notes:**

- *In slave mode, do not read the bit in LAMESR/LAMIER register via bit-band unit for the period from LIN Break detection to completion of LIN auto header reception (read LAHC bit).*
- *In master mode, do not read the bit in LAMESR/LAMIER register via bit-band unit for the period from LIN Break Field setting to completion of LIN auto header reception (read LAHC bit).*
- *Do not access the bit-band alias address of the reservation.*

#### 14) Base Timer (PWM Timer)

Register Name	Bit Name	Bit-band Alias Address	Bit Set	Bit Clear
Status control register (BT00_STC)	Underflow interrupt request bit(UDIR)	0xA404_0080	-	✓
	Duty match interrupt request bit (DTIR)	0xA404_0081	-	✓
	Trigger interrupt request bit (TGIR)	0xA404_0082	-	✓
	Underflow interrupt request enable bit (UDIE)	0xA404_0084	✓	✓
	Duty match interrupt request enable bit (DTIE)	0xA404_0085	✓	✓
	Trigger interrupt request enable bit(TGIE)	0xA404_0086	✓	✓
Status control register (BT01_STC)	Underflow interrupt request bit(UDIR)	0xA404_2080	-	✓
	Duty match interrupt request bit (DTIR)	0xA404_2081	-	✓
	Trigger interrupt request bit (TGIR)	0xA404_2082	-	✓
	Underflow interrupt request enable bit (UDIE)	0xA404_2084	✓	✓
	Duty match interrupt request enable bit (DTIE)	0xA404_2085	✓	✓
	Trigger interrupt request enable bit(TGIE)	0xA404_2086	✓	✓
Status control register (BT02_STC)	Underflow interrupt request bit(UDIR)	0xA404_4080	-	✓
	Duty match interrupt request bit (DTIR)	0xA404_4081	-	✓
	Trigger interrupt request bit (TGIR)	0xA404_4082	-	✓
	Underflow interrupt request enable bit (UDIE)	0xA404_4084	✓	✓
	Duty match interrupt request enable bit (DTIE)	0xA404_4085	✓	✓
	Trigger interrupt request enable bit(TGIE)	0xA404_4086	✓	✓
Status control register (BT03_STC)	Underflow interrupt request bit(UDIR)	0xA404_6080	-	✓
	Duty match interrupt request bit (DTIR)	0xA404_6081	-	✓
	Trigger interrupt request bit (TGIR)	0xA404_6082	-	✓
	Underflow interrupt request enable bit (UDIE)	0xA404_6084	✓	✓
	Duty match interrupt request enable bit (DTIE)	0xA404_6085	✓	✓
	Trigger interrupt request enable bit(TGIE)	0xA404_6086	✓	✓
Status control register (BT04_STC)	Underflow interrupt request bit(UDIR)	0xA404_8080	-	✓
	Duty match interrupt request bit (DTIR)	0xA404_8081	-	✓
	Trigger interrupt request bit (TGIR)	0xA404_8082	-	✓
	Underflow interrupt request enable bit (UDIE)	0xA404_8084	✓	✓
	Duty match interrupt request enable bit (DTIE)	0xA404_8085	✓	✓
	Trigger interrupt request enable bit(TGIE)	0xA404_8086	✓	✓
Status control register (BT05_STC)	Underflow interrupt request bit(UDIR)	0xA404_A080	-	✓
	Duty match interrupt request bit (DTIR)	0xA404_A081	-	✓
	Trigger interrupt request bit (TGIR)	0xA404_A082	-	✓
	Underflow interrupt request enable bit (UDIE)	0xA404_A084	✓	✓
	Duty match interrupt request enable bit (DTIE)	0xA404_A085	✓	✓
	Trigger interrupt request enable bit(TGIE)	0xA404_A086	✓	✓
Status control register (BT06_STC)	Underflow interrupt request bit(UDIR)	0xA404_C080	-	✓
	Duty match interrupt request bit (DTIR)	0xA404_C081	-	✓
	Trigger interrupt request bit (TGIR)	0xA404_C082	-	✓
	Underflow interrupt request enable bit (UDIE)	0xA404_C084	✓	✓
	Duty match interrupt request enable bit (DTIE)	0xA404_C085	✓	✓
	Trigger interrupt request enable bit(TGIE)	0xA404_C086	✓	✓
Status control register (BT07_STC)	Underflow interrupt request bit(UDIR)	0xA404_E080	-	✓
	Duty match interrupt request bit (DTIR)	0xA404_E081	-	✓
	Trigger interrupt request bit (TGIR)	0xA404_E082	-	✓
	Underflow interrupt request enable bit (UDIE)	0xA404_E084	✓	✓
	Duty match interrupt request enable bit (DTIE)	0xA404_E085	✓	✓
	Trigger interrupt request enable bit(TGIE)	0xA404_E086	✓	✓



Register Name	Bit Name	Bit-band Alias Address	Bit Set	Bit Clear
Status control register (BT08_STC)	Underflow interrupt request bit(UDIR)	0xA405_0080	-	✓
	Duty match interrupt request bit (DTIR)	0xA405_0081	-	✓
	Trigger interrupt request bit (TGIR)	0xA405_0082	-	✓
	Underflow interrupt request enable bit (UDIE)	0xA405_0084	✓	✓
	Duty match interrupt request enable bit (DTIE)	0xA405_0085	✓	✓
	Trigger interrupt request enable bit(TGIE)	0xA405_0086	✓	✓
Status control register (BT09_STC)	Underflow interrupt request bit(UDIR)	0xA405_2080	-	✓
	Duty match interrupt request bit (DTIR)	0xA405_2081	-	✓
	Trigger interrupt request bit (TGIR)	0xA405_2082	-	✓
	Underflow interrupt request enable bit (UDIE)	0xA405_2084	✓	✓
	Duty match interrupt request enable bit (DTIE)	0xA405_2085	✓	✓
	Trigger interrupt request enable bit(TGIE)	0xA405_2086	✓	✓
Status control register (BT10_STC)	Underflow interrupt request bit(UDIR)	0xA405_4080	-	✓
	Duty match interrupt request bit (DTIR)	0xA405_4081	-	✓
	Trigger interrupt request bit (TGIR)	0xA405_4082	-	✓
	Underflow interrupt request enable bit (UDIE)	0xA405_4084	✓	✓
	Duty match interrupt request enable bit (DTIE)	0xA405_4085	✓	✓
	Trigger interrupt request enable bit(TGIE)	0xA405_4086	✓	✓
Status control register (BT11_STC)	Underflow interrupt request bit(UDIR)	0xA405_6080	-	✓
	Duty match interrupt request bit (DTIR)	0xA405_6081	-	✓
	Trigger interrupt request bit (TGIR)	0xA405_6082	-	✓
	Underflow interrupt request enable bit (UDIE)	0xA405_6084	✓	✓
	Duty match interrupt request enable bit (DTIE)	0xA405_6085	✓	✓
	Trigger interrupt request enable bit(TGIE)	0xA405_6086	✓	✓

### 15) Base Timer (PPG Timer)

Register Name	Bit Name	Bit-band Alias Address	Bit Set	Bit Clear
Status control register (BT00_STC)	Underflow interrupt request bit(UDIR)	0xA404_0080	-	✓
Reserved	-	0xA404_0081	-	-
Status control register (BT00_STC)	Trigger interrupt request bit (TGIR)	0xA404_0082	-	✓
	Underflow interrupt request enable bit (UDIE)	0xA404_0084	✓	✓
Reserved	-	0xA404_0085	-	-
Status control register (BT00_STC)	Trigger interrupt request enable bit(TGIE)	0xA404_0086	✓	✓
Status control register (BT01_STC)	Underflow interrupt request bit(UDIR)	0xA404_2080	-	✓
Reserved	-	0xA404_2081	-	-
Status control register (BT01_STC)	Trigger interrupt request bit (TGIR)	0xA404_2082	-	✓
	Underflow interrupt request enable bit (UDIE)	0xA404_2084	✓	✓
Reserved	-	0xA404_2085	-	-
Status control register (BT01_STC)	Trigger interrupt request enable bit(TGIE)	0xA404_2086	✓	✓
Status control register (BT02_STC)	Underflow interrupt request bit(UDIR)	0xA404_4080	-	✓
Reserved	-	0xA404_4081	-	-
Status control register (BT02_STC)	Trigger interrupt request bit (TGIR)	0xA404_4082	-	✓
	Underflow interrupt request enable bit (UDIE)	0xA404_4084	✓	✓
Reserved	-	0xA404_4085	-	-
Status control register (BT02_STC)	Trigger interrupt request enable bit(TGIE)	0xA404_4086	✓	✓
Status control register (BT03_STC)	Underflow interrupt request bit(UDIR)	0xA404_6080	-	✓
Reserved	-	0xA404_6081	-	-
Status control register (BT03_STC)	Trigger interrupt request bit (TGIR)	0xA404_6082	-	✓
	Underflow interrupt request enable bit (UDIE)	0xA404_6084	✓	✓
Reserved	-	0xA404_6085	-	-
Status control register (BT03_STC)	Trigger interrupt request enable bit(TGIE)	0xA404_6086	✓	✓
Status control register (BT04_STC)	Underflow interrupt request bit(UDIR)	0xA404_8080	-	✓
Reserved	-	0xA404_8081	-	-
Status control register (BT04_STC)	Trigger interrupt request bit (TGIR)	0xA404_8082	-	✓
	Underflow interrupt request enable bit (UDIE)	0xA404_8084	✓	✓
Reserved	-	0xA404_8085	-	-
Status control register (BT04_STC)	Trigger interrupt request enable bit(TGIE)	0xA404_8086	✓	✓
Status control register (BT05_STC)	Underflow interrupt request bit(UDIR)	0xA404_A080	-	✓
Reserved	-	0xA404_A081	-	-
Status control register (BT05_STC)	Trigger interrupt request bit (TGIR)	0xA404_A082	-	✓
	Underflow interrupt request enable bit (UDIE)	0xA404_A084	✓	✓
Reserved	-	0xA404_A085	-	-
Status control register (BT05_STC)	Trigger interrupt request enable bit(TGIE)	0xA404_A086	✓	✓





Register Name	Bit Name	Bit-band Alias Address	Bit Set	Bit Clear
Status control register (BT06_STC)	Underflow interrupt request bit(UDIR)	0xA404_C080	-	✓
Reserved	-	0xA404_C081	-	✓
Status control register (BT06_STC)	Trigger interrupt request bit (TGIR)	0xA404_C082	-	✓
	Underflow interrupt request enable bit (UDIE)	0xA404_C084	✓	✓
Reserved	-	0xA404_C085	-	-
Status control register (BT06_STC)	Trigger interrupt request enable bit(TGIE)	0xA404_C086	✓	✓
Status control register (BT07_STC)	Underflow interrupt request bit(UDIR)	0xA404_E080	-	✓
Reserved	-	0xA404_E081	-	✓
Status control register (BT07_STC)	Trigger interrupt request bit (TGIR)	0xA404_E082	-	✓
	Underflow interrupt request enable bit (UDIE)	0xA404_E084	✓	✓
Reserved	-	0xA404_E085	-	-
Status control register (BT07_STC)	Trigger interrupt request enable bit(TGIE)	0xA404_E086	✓	✓
Status control register (BT08_STC)	Underflow interrupt request bit(UDIR)	0xA405_0080	-	✓
Reserved	-	0xA405_0081	-	✓
Status control register (BT08_STC)	Trigger interrupt request bit (TGIR)	0xA405_0082	-	✓
	Underflow interrupt request enable bit (UDIE)	0xA405_0084	✓	✓
Reserved	-	0xA405_0085	-	-
Status control register (BT08_STC)	Trigger interrupt request enable bit(TGIE)	0xA405_0086	✓	✓
Status control register (BT09_STC)	Underflow interrupt request bit(UDIR)	0xA405_2080	-	✓
Reserved	-	0xA405_2081	-	✓
Status control register (BT09_STC)	Trigger interrupt request bit (TGIR)	0xA405_2082	-	✓
	Underflow interrupt request enable bit (UDIE)	0xA405_2084	✓	✓
Reserved	-	0xA405_2085	-	-
Status control register (BT09_STC)	Trigger interrupt request enable bit(TGIE)	0xA405_2086	✓	✓
Status control register (BT10_STC)	Underflow interrupt request bit(UDIR)	0xA405_4080	-	✓
Reserved	-	0xA405_4081	-	-
Status control register (BT10_STC)	Trigger interrupt request bit (TGIR)	0xA405_4082	-	✓
	Underflow interrupt request enable bit (UDIE)	0xA405_4084	✓	✓
Reserved	-	0xA405_4085	-	-
Status control register (BT10_STC)	Trigger interrupt request enable bit(TGIE)	0xA405_4086	✓	✓
Status control register (BT11_STC)	Underflow interrupt request bit(UDIR)	0xA405_6080	-	✓
Reserved	-	0xA405_6081	-	-
Status control register (BT11_STC)	Trigger interrupt request bit (TGIR)	0xA405_6082	-	✓
	Underflow interrupt request enable bit (UDIE)	0xA405_6084	✓	✓
Reserved	-	0xA405_6085	-	-
Status control register (BT11_STC)	Trigger interrupt request enable bit(TGIE)	0xA405_6086	✓	✓

### 16) Base Timer (Reload Timer)

Register Name	Bit Name	Bit-band Alias Address	Bit Set	Bit Clear
Status control register (BT00_STC)	Underflow interrupt request bit(UDIR)	0xA404_0080	-	✓
Reserved	-	0xA404_0081	-	-
Status control register (BT00_STC)	Trigger interrupt request bit (TGIR)	0xA404_0082	-	✓
	Underflow interrupt request enable bit (UDIE)	0xA404_0084	✓	✓
Reserved	-	0xA404_0085	-	-
Status control register (BT00_STC)	Trigger interrupt request enable bit(TGIE)	0xA404_0086	✓	✓
Status control register (BT01_STC)	Underflow interrupt request bit(UDIR)	0xA404_2080	-	✓
Reserved	-	0xA404_2081	-	-
Status control register (BT01_STC)	Trigger interrupt request bit (TGIR)	0xA404_2082	-	✓
	Underflow interrupt request enable bit (UDIE)	0xA404_2084	✓	✓
Reserved	-	0xA404_2085	-	-
Status control register (BT01_STC)	Trigger interrupt request enable bit(TGIE)	0xA404_2086	✓	✓
Status control register (BT02_STC)	Underflow interrupt request bit(UDIR)	0xA404_4080	-	✓
Reserved	-	0xA404_4081	-	-
Status control register (BT02_STC)	Trigger interrupt request bit (TGIR)	0xA404_4082	-	✓
	Underflow interrupt request enable bit (UDIE)	0xA404_4084	✓	✓
Reserved	-	0xA404_4085	-	-
Status control register (BT02_STC)	Trigger interrupt request enable bit(TGIE)	0xA404_4086	✓	✓
Status control register (BT03_STC)	Underflow interrupt request bit(UDIR)	0xA404_6080	-	✓
Reserved	-	0xA404_6081	-	-
Status control register (BT03_STC)	Trigger interrupt request bit (TGIR)	0xA404_6082	-	✓
	Underflow interrupt request enable bit (UDIE)	0xA404_6084	✓	✓
Reserved	-	0xA404_6085	-	-
Status control register (BT03_STC)	Trigger interrupt request enable bit(TGIE)	0xA404_6086	✓	✓
Status control register (BT04_STC)	Underflow interrupt request bit(UDIR)	0xA404_8080	-	✓
Reserved	-	0xA404_8081	-	-
Status control register (BT04_STC)	Trigger interrupt request bit (TGIR)	0xA404_8082	-	✓
	Underflow interrupt request enable bit (UDIE)	0xA404_8084	✓	✓
Reserved	-	0xA404_8085	-	-
Status control register (BT04_STC)	Trigger interrupt request enable bit(TGIE)	0xA404_8086	✓	✓
Status control register (BT05_STC)	Underflow interrupt request bit(UDIR)	0xA404_A080	-	✓
Reserved	-	0xA404_A081	-	-
Status control register (BT05_STC)	Trigger interrupt request bit (TGIR)	0xA404_A082	-	✓
	Underflow interrupt request enable bit (UDIE)	0xA404_A084	✓	✓
Reserved	-	0xA404_A085	-	-
Status control register (BT05_STC)	Trigger interrupt request enable bit(TGIE)	0xA404_A086	✓	✓



Register Name	Bit Name	Bit-band Alias Address	Bit Set	Bit Clear
Status control register (BT06_STC)	Underflow interrupt request bit(UDIR)	0xA404_C080	-	✓
Reserved	-	0xA404_C081	-	-
Status control register (BT06_STC)	Trigger interrupt request bit (TGIR)	0xA404_C082	-	✓
	Underflow interrupt request enable bit (UDIE)	0xA404_C084	✓	✓
Reserved	-	0xA404_C085	-	-
Status control register (BT06_STC)	Trigger interrupt request enable bit(TGIE)	0xA404_C086	✓	✓
Status control register (BT07_STC)	Underflow interrupt request bit(UDIR)	0xA404_E080	-	✓
Reserved	-	0xA404_E081	-	-
Status control register (BT07_STC)	Trigger interrupt request bit (TGIR)	0xA404_E082	-	✓
	Underflow interrupt request enable bit (UDIE)	0xA404_E084	✓	✓
Reserved	-	0xA404_E085	-	-
Status control register (BT07_STC)	Trigger interrupt request enable bit(TGIE)	0xA404_E086	✓	✓
Status control register (BT08_STC)	Underflow interrupt request bit(UDIR)	0xA405_0080	-	✓
Reserved	-	0xA405_0081	-	-
Status control register (BT08_STC)	Trigger interrupt request bit (TGIR)	0xA405_0082	-	✓
	Underflow interrupt request enable bit (UDIE)	0xA405_0084	✓	✓
Reserved	-	0xA405_0085	-	-
Status control register (BT08_STC)	Trigger interrupt request enable bit(TGIE)	0xA405_0086	✓	✓
Status control register (BT09_STC)	Underflow interrupt request bit(UDIR)	0xA405_2080	-	✓
Reserved	-	0xA405_2081	-	-
Status control register (BT09_STC)	Trigger interrupt request bit (TGIR)	0xA405_2082	-	✓
	Underflow interrupt request enable bit (UDIE)	0xA405_2084	✓	✓
Reserved	-	0xA405_2085	-	-
Status control register (BT09_STC)	Trigger interrupt request enable bit(TGIE)	0xA405_2086	✓	✓
Status control register (BT10_STC)	Underflow interrupt request bit(UDIR)	0xA405_4080	-	✓
Reserved	-	0xA405_4081	-	-
Status control register (BT10_STC)	Trigger interrupt request bit (TGIR)	0xA405_4082	-	✓
	Underflow interrupt request enable bit (UDIE)	0xA405_4084	✓	✓
Reserved	-	0xA405_4085	-	-
Status control register (BT10_STC)	Trigger interrupt request enable bit(TGIE)	0xA405_4086	✓	✓
Status control register (BT11_STC)	Underflow interrupt request bit(UDIR)	0xA405_6080	-	✓
Reserved	-	0xA405_6081	-	-
Status control register (BT11_STC)	Trigger interrupt request bit (TGIR)	0xA405_6082	-	✓
	Underflow interrupt request enable bit (UDIE)	0xA405_6084	✓	✓
Reserved	-	0xA405_6085	-	-
Status control register (BT11_STC)	Trigger interrupt request enable bit(TGIE)	0xA405_6086	✓	✓

### 17) Base Timer (PWC Timer)

Register Name	Bit Name	Bit-band Alias Address	Bit Set	Bit Clear
Status control register (BT00_STC)	Overflow interrupt request bit(OVIR)	0xA404_0080	-	✓
Reserved	-	0xA404_0081, 0xA404_0082	-	-
Status control register (BT00_STC)	Overflow interrupt request enable bit (OVIE)	0xA404_0084	✓	✓
Reserved	-	0xA404_0085	-	-
Status control register (BT00_STC)	Measurement end interrupt request enable bit (EDIE)	0xA404_0086	✓	✓
Status control register (BT01_STC)	Overflow interrupt request bit(OVIR)	0xA404_2080	-	✓
Reserved	-	0xA404_2081, 0xA404_2082	-	-
Status control register (BT01_STC)	Overflow interrupt request enable bit (OVIE)	0xA404_2084	✓	✓
Reserved	-	0xA404_2085	-	-
Status control register (BT01_STC)	Measurement end interrupt request enable bit (EDIE)	0xA404_2086	✓	✓
Status control register (BT02_STC)	Overflow interrupt request bit(OVIR)	0xA404_4080	-	✓
Reserved	-	0xA404_4081, 0xA404_4082	-	-
Status control register (BT02_STC)	Overflow interrupt request enable bit (OVIE)	0xA404_4084	✓	✓
Reserved	-	0xA404_4085	-	-
Status control register (BT02_STC)	Measurement end interrupt request enable bit (EDIE)	0xA404_4086	✓	✓
Status control register (BT03_STC)	Overflow interrupt request bit(OVIR)	0xA404_6080	-	✓
Reserved	-	0xA404_6081, 0xA404_6082	-	-
Status control register (BT03_STC)	Overflow interrupt request enable bit (OVIE)	0xA404_6084	✓	✓
Reserved	-	0xA404_6085	-	-
Status control register (BT03_STC)	Measurement end interrupt request enable bit (EDIE)	0xA404_6086	✓	✓
Status control register (BT04_STC)	Overflow interrupt request bit(OVIR)	0xA404_8080	-	✓
Reserved	-	0xA404_8081, 0xA404_8082	-	-
Status control register (BT04_STC)	Overflow interrupt request enable bit (OVIE)	0xA404_8084	✓	✓
Reserved	-	0xA404_8085	-	-
Status control register (BT04_STC)	Measurement end interrupt request enable bit (EDIE)	0xA404_8086	✓	✓
Status control register (BT05_STC)	Overflow interrupt request bit(OVIR)	0xA404_A080	-	✓
Reserved	-	0xA404_A081, 0xA404_A082	-	-
Status control register (BT05_STC)	Overflow interrupt request enable bit (OVIE)	0xA404_A084	✓	✓
Reserved	-	0xA404_A085	-	-
Status control register (BT05_STC)	Measurement end interrupt request enable bit (EDIE)	0xA404_A086	✓	✓



Register Name	Bit Name	Bit-band Alias Address	Bit Set	Bit Clear
Status control register (BT06_STC)	Overflow interrupt request bit(OVIR)	0xA404_C080	-	✓
Reserved	-	0xA404_C081, 0xA404_C082	-	-
Status control register (BT06_STC)	Overflow interrupt request enable bit (OVIE)	0xA404_C084	✓	✓
Reserved	-	0xA404_C085	-	-
Status control register (BT06_STC)	Measurement end interrupt request enable bit (EDIE)	0xA404_C086	✓	✓
Status control register (BT07_STC)	Overflow interrupt request bit(OVIR)	0xA404_E080	-	✓
Reserved	-	0xA404_E081, 0xA404_E082	-	-
Status control register (BT07_STC)	Overflow interrupt request enable bit (OVIE)	0xA404_E084	✓	✓
Reserved	-	0xA404_E085	-	-
Status control register (BT07_STC)	Measurement end interrupt request enable bit (EDIE)	0xA404_E086	✓	✓
Status control register (BT08_STC)	Overflow interrupt request bit(OVIR)	0xA405_0080	-	✓
Reserved	-	0xA405_0081, 0xA405_0082	-	-
Status control register (BT08_STC)	Overflow interrupt request enable bit (OVIE)	0xA405_0084	✓	✓
Reserved	-	0xA405_0085	-	-
Status control register (BT08_STC)	Measurement end interrupt request enable bit (EDIE)	0xA405_0086	✓	✓
Status control register (BT09_STC)	Overflow interrupt request bit(OVIR)	0xA405_2080	-	✓
Reserved	-	0xA405_2081, 0xA405_2082	-	-
Status control register (BT09_STC)	Overflow interrupt request enable bit (OVIE)	0xA405_2084	✓	✓
Reserved	-	0xA405_2085	-	-
Status control register (BT09_STC)	Measurement end interrupt request enable bit (EDIE)	0xA405_2086	✓	✓
Status control register (BT10_STC)	Overflow interrupt request bit(OVIR)	0xA405_4080	-	✓
Reserved	-	0xA405_4081, 0xA405_4082	-	-
Status control register (BT10_STC)	Overflow interrupt request enable bit (OVIE)	0xA405_4084	✓	✓
Reserved	-	0xA405_4085	-	-
Status control register (BT10_STC)	Measurement end interrupt request enable bit (EDIE)	0xA405_4086	✓	✓
Status control register (BT11_STC)	Overflow interrupt request bit(OVIR)	0xA405_6080	-	✓
Reserved	-	0xA405_6081, 0xA405_6082	-	-
Status control register (BT11_STC)	Overflow interrupt request enable bit (OVIE)	0xA405_6084	✓	✓
Reserved	-	0xA405_6085	-	-
Status control register (BT11_STC)	Measurement end interrupt request enable bit (EDIE)	0xA405_6086	✓	✓

### 18) 32-bit Free-run Timer

Register Name	Bit Name	Bit-band Alias Address	Bit Set	Bit Clear
Timer state control register (FRT00_TCCS)	Timer clear bit (SCLR)	0xA410_0044	✓	-
	Timer Count Mode bit (MODE)	0xA410_0045	✓	✓
	Timer enable bit (STOP)	0xA410_0046	✓	✓
	Compare clear buffer enable bit (BFE)	0xA410_0047	✓	✓
	Compare clear interrupt request enable bit (ICRE)	0xA410_0048	✓	✓
	Compare clear interrupt flag bit (ICLR)	0xA410_0049	-	✓
	0 detection request enable bit (IRQZE)	0xA410_004D	✓	✓
	0 detection interrupt flag bit (IRQZF)	0xA410_004E	-	✓
Reserved	-	0xA410_004F	-	-
Timer state control register (FRT01_TCCS)	Timer clear bit (SCLR)	0xA410_2044	✓	-
	Timer Count Mode bit (MODE)	0xA410_2045	✓	✓
	Timer enable bit (STOP)	0xA410_2046	✓	✓
	Compare clear buffer enable bit (BFE)	0xA410_2047	✓	✓
	Compare clear interrupt request enable bit (ICRE)	0xA410_2048	✓	✓
	Compare clear interrupt flag bit (ICLR)	0xA410_2049	-	✓
	0 detection request enable bit (IRQZE)	0xA410_204D	✓	✓
	0 detection interrupt flag bit (IRQZF)	0xA410_204E	-	✓
Reserved	-	0xA410_204F	-	-
Timer state control register (FRT02_TCCS)	Timer clear bit (SCLR)	0xA410_4044	✓	-
	Timer Count Mode bit (MODE)	0xA410_4045	✓	✓
	Timer enable bit (STOP)	0xA410_4046	✓	✓
	Compare clear buffer enable bit (BFE)	0xA410_4047	✓	✓
	Compare clear interrupt request enable bit (ICRE)	0xA410_4048	✓	✓
	Compare clear interrupt flag bit (ICLR)	0xA410_4049	-	✓
	0 detection request enable bit (IRQZE)	0xA410_404D	✓	✓
	0 detection interrupt flag bit (IRQZF)	0xA410_404E	-	✓
Reserved	-	0xA410_404F	-	-
Timer state control register (FRT03_TCCS)	Timer clear bit (SCLR)	0xA410_6044	✓	-
	Timer Count Mode bit (MODE)	0xA410_6045	✓	✓
	Timer enable bit (STOP)	0xA410_6046	✓	✓
	Compare clear buffer enable bit (BFE)	0xA410_6047	✓	✓
	Compare clear interrupt request enable bit (ICRE)	0xA410_6048	✓	✓
	Compare clear interrupt flag bit (ICLR)	0xA410_6049	-	✓
	0 detection request enable bit (IRQZE)	0xA410_604D	✓	✓
	0 detection interrupt flag bit (IRQZF)	0xA410_604E	-	✓
Reserved	-	0xA410_604F	-	-
Timer state control register (FRT04_TCCS)	Timer clear bit (SCLR)	0xA410_8044	✓	-
	Timer Count Mode bit (MODE)	0xA410_8045	✓	✓
	Timer enable bit (STOP)	0xA410_8046	✓	✓
	Compare clear buffer enable bit (BFE)	0xA410_8047	✓	✓
	Compare clear interrupt request enable bit (ICRE)	0xA410_8048	✓	✓
	Compare clear interrupt flag bit (ICLR)	0xA410_8049	-	✓
	0 detection request enable bit (IRQZE)	0xA410_804D	✓	✓
	0 detection interrupt flag bit (IRQZF)	0xA410_804E	-	✓
Reserved	-	0xA410_804F	-	-

**Note:**

- Do not access the bit-band alias address of the reservation.



### 19) 32-bit Input Capture

Register Name	Bit Name	Bit-band Alias Address	Bit Set	Bit Clear
Input capture state control Register (ICU00_ICS)	Input capture 0 interrupt request enable bit (ICE0)	0xA414_0044	✓	✓
	Input capture 1 interrupt request enable bit (ICE1)	0xA414_0045	✓	✓
	Input capture 0 interrupt request flag bit (ICP0)	0xA414_0046	-	✓
	Input capture 1 interrupt request flag bit (ICP1)	0xA414_0047	-	✓
Input capture state control Register (ICU02_ICS)	Input capture 0 interrupt request enable bit (ICE0)	0xA414_2044	✓	✓
	Input capture 1 interrupt request enable bit (ICE1)	0xA414_2045	✓	✓
	Input capture 0 interrupt request flag bit (ICP0)	0xA414_2046	-	✓
	Input capture 1 interrupt request flag bit (ICP1)	0xA414_2047	-	✓
Input capture state control Register (ICU04_ICS)	Input capture 0 interrupt request enable bit (ICE0)	0xA414_4044	✓	✓
	Input capture 1 interrupt request enable bit (ICE1)	0xA414_4045	✓	✓
	Input capture 0 interrupt request flag bit (ICP0)	0xA414_4046	-	✓
	Input capture 1 interrupt request flag bit (ICP1)	0xA414_4047	-	✓

### 20) 16-bit Free-run Timer

Register Name	Bit Name	Bit-band Alias Address	Bit Set	Bit Clear
Timer state control register (FRT16B00_TCCS)	Interrupt mask mode bit 2 (MODE2)	0xA880_002B	✓	✓
	Timer clear bit (SCLR)	0xA880_0034	✓	-
	Timer Count Mode bit (MODE)	0xA880_0035	✓	✓
	Timer enable bit (STOP)	0xA880_0036	✓	✓
	Compare clear buffer enable bit (BFE)	0xA880_0037	✓	✓
	Compare clear interrupt request enable bit (ICRE)	0xA880_0038	✓	✓
	Compare clear interrupt flag bit (ICLR)	0xA880_0039	-	✓
	0 detection request enable bit (IRQZE)	0xA880_003D	✓	✓
	0 detection interrupt flag bit (IRQZF)	0xA880_003E	-	✓
	Clock selection bit (ECKE)	0xA880_003F	✓	✓
Timer state control register (FRT16B01_TCCS)	Interrupt mask mode bit 2 (MODE2)	0xA880_00AB	✓	✓
	Timer clear bit (SCLR)	0xA880_00B4	✓	-
	Timer Count Mode bit (MODE)	0xA880_00B5	✓	✓
	Timer enable bit (STOP)	0xA880_00B6	✓	✓
	Compare clear buffer enable bit (BFE)	0xA880_00B7	✓	✓
	Compare clear interrupt request enable bit (ICRE)	0xA880_00B8	✓	✓
	Compare clear interrupt flag bit (ICLR)	0xA880_00B9	-	✓
	0 detection request enable bit (IRQZE)	0xA880_00BD	✓	✓
	0 detection interrupt flag bit (IRQZF)	0xA880_00BE	-	✓
	Clock selection bit (ECKE)	0xA880_00BF	✓	✓
Timer state control register (FRT16B02_TCCS)	Interrupt mask mode bit 2 (MODE2)	0xA880_012B	✓	✓
	Timer clear bit (SCLR)	0xA880_0134	✓	-
	Timer Count Mode bit (MODE)	0xA880_0135	✓	✓
	Timer enable bit (STOP)	0xA880_0136	✓	✓
	Compare clear buffer enable bit (BFE)	0xA880_0137	✓	✓
	Compare clear interrupt request enable bit (ICRE)	0xA880_0138	✓	✓
	Compare clear interrupt flag bit (ICLR)	0xA880_0139	-	✓
	0 detection request enable bit (IRQZE)	0xA880_013D	✓	✓
	0 detection interrupt flag bit (IRQZF)	0xA880_013E	-	✓
	Clock selection bit (ECKE)	0xA880_013F	✓	✓



Register Name	Bit Name	Bit-band Alias Address	Bit Set	Bit Clear
Timer state control register (FRT16B03_TCCS)	Interrupt mask mode bit 2 (MODE2)	0xA880_01AB	✓	✓
	Timer clear bit (SCLR)	0xA880_01B4	✓	-
	Timer Count Mode bit (MODE)	0xA880_01B5	✓	✓
	Timer enable bit (STOP)	0xA880_01B6	✓	✓
	Compare clear buffer enable bit (BFE)	0xA880_01B7	✓	✓
	Compare clear interrupt request enable bit (ICRE)	0xA880_01B8	✓	✓
	Compare clear interrupt flag bit (ICLR)	0xA880_01B9	-	✓
	0 detection request enable bit (IRQZE)	0xA880_01BD	✓	✓
	0 detection interrupt flag bit (IRQZF)	0xA880_01BE	-	✓
	Clock selection bit (ECKE)	0xA880_01BF	✓	✓
Timer state control register (FRT16B04_TCCS)	Interrupt mask mode bit 2 (MODE2)	0xA880_022B	✓	✓
	Timer clear bit (SCLR)	0xA880_0234	✓	-
	Timer Count Mode bit (MODE)	0xA880_0235	✓	✓
	Timer enable bit (STOP)	0xA880_0236	✓	✓
	Compare clear buffer enable bit (BFE)	0xA880_0237	✓	✓
	Compare clear interrupt request enable bit (ICRE)	0xA880_0238	✓	✓
	Compare clear interrupt flag bit (ICLR)	0xA880_0239	-	✓
	0 detection request enable bit (IRQZE)	0xA880_023D	✓	✓
	0 detection interrupt flag bit (IRQZF)	0xA880_023E	-	✓
	Clock selection bit (ECKE)	0xA880_023F	✓	✓
Timer state control register (FRT16B05_TCCS)	Interrupt mask mode bit 2 (MODE2)	0xA880_02AB	✓	✓
	Timer clear bit (SCLR)	0xA880_02B4	✓	-
	Timer Count Mode bit (MODE)	0xA880_02B5	✓	✓
	Timer enable bit (STOP)	0xA880_02B6	✓	✓
	Compare clear buffer enable bit (BFE)	0xA880_02B7	✓	✓
	Compare clear interrupt request enable bit (ICRE)	0xA880_02B8	✓	✓
	Compare clear interrupt flag bit (ICLR)	0xA880_02B9	-	✓
	0 detection request enable bit (IRQZE)	0xA880_02BD	✓	✓
	0 detection interrupt flag bit (IRQZF)	0xA880_02BE	-	✓
	Clock selection bit (ECKE)	0xA880_02BF	✓	✓
Timer state control register (FRT16B06_TCCS)	Interrupt mask mode bit 2 (MODE2)	0xA800_002B	✓	✓
	Timer clear bit (SCLR)	0xA800_0034	✓	-
	Timer Count Mode bit (MODE)	0xA800_0035	✓	✓
	Timer enable bit (STOP)	0xA800_0036	✓	✓
	Compare clear buffer enable bit (BFE)	0xA800_0037	✓	✓
	Compare clear interrupt request enable bit (ICRE)	0xA800_0038	✓	✓
	Compare clear interrupt flag bit (ICLR)	0xA800_0039	-	✓
	0 detection request enable bit (IRQZE)	0xA800_003D	✓	✓
	0 detection interrupt flag bit (IRQZF)	0xA800_003E	-	✓
	Clock selection bit (ECKE)	0xA800_003F	✓	✓
Timer state control register (FRT16B07_TCCS)	Interrupt mask mode bit 2 (MODE2)	0xA800_00AB	✓	✓
	Timer clear bit (SCLR)	0xA800_00B4	✓	-
	Timer Count Mode bit (MODE)	0xA800_00B5	✓	✓
	Timer enable bit (STOP)	0xA800_00B6	✓	✓
	Compare clear buffer enable bit (BFE)	0xA800_00B7	✓	✓
	Compare clear interrupt request enable bit (ICRE)	0xA800_00B8	✓	✓
	Compare clear interrupt flag bit (ICLR)	0xA800_00B9	-	✓
	0 detection request enable bit (IRQZE)	0xA800_00BD	✓	✓
	0 detection interrupt flag bit (IRQZF)	0xA800_00BE	-	✓
	Clock selection bit (ECKE)	0xA800_00BF	✓	✓





Register Name	Bit Name	Bit-band Alias Address	Bit Set	Bit Clear
Timer state control register (FRT16B08_TCCS)	Interrupt mask mode bit 2 (MODE2)	0xA800_012B	✓	✓
	Timer clear bit (SCLR)	0xA800_0134	✓	-
	Timer Count Mode bit (MODE)	0xA800_0135	✓	✓
	Timer enable bit (STOP)	0xA800_0136	✓	✓
	Compare clear buffer enable bit (BFE)	0xA800_0137	✓	✓
	Compare clear interrupt request enable bit (ICRE)	0xA800_0138	✓	✓
	Compare clear interrupt flag bit (ICLR)	0xA800_0139	-	✓
	0 detection request enable bit (IRQZE)	0xA800_013D	✓	✓
	0 detection interrupt flag bit (IRQZF)	0xA800_013E	-	✓
	Clock selection bit (ECKE)	0xA800_013F	✓	✓
Timer state control register (FRT16B09_TCCS)	Interrupt mask mode bit 2 (MODE2)	0xA800_01AB	✓	✓
	Timer clear bit (SCLR)	0xA800_01B4	✓	-
	Timer Count Mode bit (MODE)	0xA800_01B5	✓	✓
	Timer enable bit (STOP)	0xA800_01B6	✓	✓
	Compare clear buffer enable bit (BFE)	0xA800_01B7	✓	✓
	Compare clear interrupt request enable bit (ICRE)	0xA800_01B8	✓	✓
	Compare clear interrupt flag bit (ICLR)	0xA800_01B9	-	✓
	0 detection request enable bit (IRQZE)	0xA800_01BD	✓	✓
	0 detection interrupt flag bit (IRQZF)	0xA800_01BE	-	✓
	Clock selection bit (ECKE)	0xA800_01BF	✓	✓
Timer state control register (FRT16B10_TCCS)	Interrupt mask mode bit 2 (MODE2)	0xA800_022B	✓	✓
	Timer clear bit (SCLR)	0xA800_0234	✓	-
	Timer Count Mode bit (MODE)	0xA800_0235	✓	✓
	Timer enable bit (STOP)	0xA800_0236	✓	✓
	Compare clear buffer enable bit (BFE)	0xA800_0237	✓	✓
	Compare clear interrupt request enable bit (ICRE)	0xA800_0238	✓	✓
	Compare clear interrupt flag bit (ICLR)	0xA800_0239	-	✓
	0 detection request enable bit (IRQZE)	0xA800_023D	✓	✓
	0 detection interrupt flag bit (IRQZF)	0xA800_023E	-	✓
	Clock selection bit (ECKE)	0xA800_023F	✓	✓
Timer state control register (FRT16B11_TCCS)	Interrupt mask mode bit 2 (MODE2)	0xA800_02AB	✓	✓
	Timer clear bit (SCLR)	0xA800_02B4	✓	-
	Timer Count Mode bit (MODE)	0xA800_02B5	✓	✓
	Timer enable bit (STOP)	0xA800_02B6	✓	✓
	Compare clear buffer enable bit (BFE)	0xA800_02B7	✓	✓
	Compare clear interrupt request enable bit (ICRE)	0xA800_02B8	✓	✓
	Compare clear interrupt flag bit (ICLR)	0xA800_02B9	-	✓
	0 detection request enable bit (IRQZE)	0xA800_02BD	✓	✓
	0 detection interrupt flag bit (IRQZF)	0xA800_02BE	-	✓
	Clock selection bit (ECKE)	0xA800_02BF	✓	✓
Timer state control register (FRT16B12_TCCS)	Interrupt mask mode bit 2 (MODE2)	0xA808_002B	✓	✓
	Timer clear bit (SCLR)	0xA808_0034	✓	-
	Timer Count Mode bit (MODE)	0xA808_0035	✓	✓
	Timer enable bit (STOP)	0xA808_0036	✓	✓
	Compare clear buffer enable bit (BFE)	0xA808_0037	✓	✓
	Compare clear interrupt request enable bit (ICRE)	0xA808_0038	✓	✓
	Compare clear interrupt flag bit (ICLR)	0xA808_0039	-	✓
	0 detection request enable bit (IRQZE)	0xA808_003D	✓	✓
	0 detection interrupt flag bit (IRQZF)	0xA808_003E	-	✓
	Clock selection bit (ECKE)	0xA808_003F	✓	✓

Register Name	Bit Name	Bit-band Alias Address	Bit Set	Bit Clear
Timer state control register (FRT16B13_TCCS)	Interrupt mask mode bit 2 (MODE2)	0xA808_00AB	✓	✓
	Timer clear bit (SCLR)	0xA808_00B4	✓	-
	Timer Count Mode bit (MODE)	0xA808_00B5	✓	✓
	Timer enable bit (STOP)	0xA808_00B6	✓	✓
	Compare clear buffer enable bit (BFE)	0xA808_00B7	✓	✓
	Compare clear interrupt request enable bit (ICRE)	0xA808_00B8	✓	✓
	Compare clear interrupt flag bit (ICLR)	0xA808_00B9	-	✓
	0 detection request enable bit (IRQZE)	0xA808_00BD	✓	✓
	0 detection interrupt flag bit (IRQZF)	0xA808_00BE	-	✓
	Clock selection bit (ECKE)	0xA808_00BF	✓	✓
Timer state control register (FRT16B14_TCCS)	Interrupt mask mode bit 2 (MODE2)	0xA808_012B	✓	✓
	Timer clear bit (SCLR)	0xA808_0134	✓	-
	Timer Count Mode bit (MODE)	0xA808_0135	✓	✓
	Timer enable bit (STOP)	0xA808_0136	✓	✓
	Compare clear buffer enable bit (BFE)	0xA808_0137	✓	✓
	Compare clear interrupt request enable bit (ICRE)	0xA808_0138	✓	✓
	Compare clear interrupt flag bit (ICLR)	0xA808_0139	-	✓
	0 detection request enable bit (IRQZE)	0xA808_013D	✓	✓
	0 detection interrupt flag bit (IRQZF)	0xA808_013E	-	✓
	Clock selection bit (ECKE)	0xA808_013F	✓	✓
Timer state control register (FRT16B15_TCCS)	Interrupt mask mode bit 2 (MODE2)	0xA808_01AB	✓	✓
	Timer clear bit (SCLR)	0xA808_01B4	✓	-
	Timer Count Mode bit (MODE)	0xA808_01B5	✓	✓
	Timer enable bit (STOP)	0xA808_01B6	✓	✓
	Compare clear buffer enable bit (BFE)	0xA808_01B7	✓	✓
	Compare clear interrupt request enable bit (ICRE)	0xA808_01B8	✓	✓
	Compare clear interrupt flag bit (ICLR)	0xA808_01B9	-	✓
	0 detection request enable bit (IRQZE)	0xA808_01BD	✓	✓
	0 detection interrupt flag bit (IRQZF)	0xA808_01BE	-	✓
	Clock selection bit (ECKE)	0xA808_01BF	✓	✓
Timer state control register (FRT16B16_TCCS)	Interrupt mask mode bit 2 (MODE2)	0xA808_022B	✓	✓
	Timer clear bit (SCLR)	0xA808_0234	✓	-
	Timer Count Mode bit (MODE)	0xA808_0235	✓	✓
	Timer enable bit (STOP)	0xA808_0236	✓	✓
	Compare clear buffer enable bit (BFE)	0xA808_0237	✓	✓
	Compare clear interrupt request enable bit (ICRE)	0xA808_0238	✓	✓
	Compare clear interrupt flag bit (ICLR)	0xA808_0239	-	✓
	0 detection request enable bit (IRQZE)	0xA808_023D	✓	✓
	0 detection interrupt flag bit (IRQZF)	0xA808_023E	-	✓
	Clock selection bit (ECKE)	0xA808_023F	✓	✓
Timer state control register (FRT16B17_TCCS)	Interrupt mask mode bit 2 (MODE2)	0xA808_02AB	✓	✓
	Timer clear bit (SCLR)	0xA808_02B4	✓	-
	Timer Count Mode bit (MODE)	0xA808_02B5	✓	✓
	Timer enable bit (STOP)	0xA808_02B6	✓	✓
	Compare clear buffer enable bit (BFE)	0xA808_02B7	✓	✓
	Compare clear interrupt request enable bit (ICRE)	0xA808_02B8	✓	✓
	Compare clear interrupt flag bit (ICLR)	0xA808_02B9	-	✓
	0 detection request enable bit (IRQZE)	0xA808_02BD	✓	✓
	0 detection interrupt flag bit (IRQZF)	0xA808_02BE	-	✓
	Clock selection bit (ECKE)	0xA808_02BF	✓	✓



Register Name	Bit Name	Bit-band Alias Address	Bit Set	Bit Clear
Timer state control register (FRT16B18_TCCS)	Interrupt mask mode bit 2 (MODE2)	0xA880_032B	✓	✓
	Timer clear bit (SCLR)	0xA880_0334	✓	-
	Timer Count Mode bit (MODE)	0xA880_0335	✓	✓
	Timer enable bit (STOP)	0xA880_0336	✓	✓
	Compare clear buffer enable bit (BFE)	0xA880_0337	✓	✓
	Compare clear interrupt request enable bit (ICRE)	0xA880_0338	✓	✓
	Compare clear interrupt flag bit (ICLR)	0xA880_0339	-	✓
	0 detection request enable bit (IRQZE)	0xA880_033D	✓	✓
	0 detection interrupt flag bit (IRQZF)	0xA880_033E	-	✓
	Clock selection bit (ECKE)	0xA880_033F	✓	✓
Timer state control register (FRT16B19_TCCS)	Interrupt mask mode bit 2 (MODE2)	0xA800_032B	✓	✓
	Timer clear bit (SCLR)	0xA800_0334	✓	-
	Timer Count Mode bit (MODE)	0xA800_0335	✓	✓
	Timer enable bit (STOP)	0xA800_0336	✓	✓
	Compare clear buffer enable bit (BFE)	0xA800_0337	✓	✓
	Compare clear interrupt request enable bit (ICRE)	0xA800_0338	✓	✓
	Compare clear interrupt flag bit (ICLR)	0xA800_0339	-	✓
	0 detection request enable bit (IRQZE)	0xA800_033D	✓	✓
	0 detection interrupt flag bit (IRQZF)	0xA800_033E	-	✓
	Clock selection bit (ECKE)	0xA800_033F	✓	✓

## 21) 16-bit Output Compare

Register Name	Bit Name	Bit-band Alias Address	Bit Set	Bit Clear
Compare Control Register (OCU16B00_OCS)	Compare operation enable bit (CST0)	0xA880_0830	✓	✓
	Compare operation enable bit (CST1)	0xA880_0831	✓	✓
	Output compare buffer invalidating bit (BUF0)	0xA880_0832	✓	✓
	Output compare buffer invalidating bit (BUF1)	0xA880_0833	✓	✓
	Compare match interrupt enable bit (IOE0)	0xA880_0834	✓	✓
	Compare match interrupt enable bit (IOE1)	0xA880_0835	✓	✓
	Compare match interrupt flag bit (IOP0)	0xA880_0836	-	✓
	Compare match interrupt flag bit (IOP1)	0xA880_0837	-	✓
	Output level bit (OTD0)	0xA880_0838	✓	✓
	Output level bit (OTD1)	0xA880_0839	✓	✓
	Output level Inversion mode bit (CMOD)	0xA880_083C	✓	✓
	Buffer transfer selection bit (BTS0)	0xA880_083D	✓	✓
	Buffer transfer selection bit (BTS1)	0xA880_083E	✓	✓
Compare Control Register (OCU16B02_OCS)	Compare operation enable bit (CST0)	0xA880_08B0	✓	✓
	Compare operation enable bit (CST1)	0xA880_08B1	✓	✓
	Output compare buffer invalidating bit (BUF0)	0xA880_08B2	✓	✓
	Output compare buffer invalidating bit (BUF1)	0xA880_08B3	✓	✓
	Compare match interrupt enable bit (IOE0)	0xA880_08B4	✓	✓
	Compare match interrupt enable bit (IOE1)	0xA880_08B5	✓	✓
	Compare match interrupt flag bit (IOP0)	0xA880_08B6	-	✓
	Compare match interrupt flag bit (IOP1)	0xA880_08B7	-	✓
	Output level bit (OTD0)	0xA880_08B8	✓	✓
	Output level bit (OTD1)	0xA880_08B9	✓	✓
	Output level Inversion mode bit (CMOD)	0xA880_08BC	✓	✓
	Buffer transfer selection bit (BTS0)	0xA880_08BD	✓	✓
	Buffer transfer selection bit (BTS1)	0xA880_08BE	✓	✓

Register Name	Bit Name	Bit-band Alias Address	Bit Set	Bit Clear
Compare Control Register (OCU16B04_OCS)	Compare operation enable bit (CST0)	0xA880_0930	✓	✓
	Compare operation enable bit (CST1)	0xA880_0931	✓	✓
	Output compare buffer invalidating bit (BUF0)	0xA880_0932	✓	✓
	Output compare buffer invalidating bit (BUF1)	0xA880_0933	✓	✓
	Compare match interrupt enable bit (IOE0)	0xA880_0934	✓	✓
	Compare match interrupt enable bit (IOE1)	0xA880_0935	✓	✓
	Compare match interrupt flag bit (IOP0)	0xA880_0936	-	✓
	Compare match interrupt flag bit (IOP1)	0xA880_0937	-	✓
	Output level bit (OTD0)	0xA880_0938	✓	✓
	Output level bit (OTD1)	0xA880_0939	✓	✓
	Output level Inversion mode bit (CMOD)	0xA880_093C	✓	✓
	Buffer transfer selection bit (BTS0)	0xA880_093D	✓	✓
	Buffer transfer selection bit (BTS1)	0xA880_093E	✓	✓
Compare Control Register (OCU16B06_OCS)	Compare operation enable bit (CST0)	0xA800_0830	✓	✓
	Compare operation enable bit (CST1)	0xA800_0831	✓	✓
	Output compare buffer invalidating bit (BUF0)	0xA800_0832	✓	✓
	Output compare buffer invalidating bit (BUF1)	0xA800_0833	✓	✓
	Compare match interrupt enable bit (IOE0)	0xA800_0834	✓	✓
	Compare match interrupt enable bit (IOE1)	0xA800_0835	✓	✓
	Compare match interrupt flag bit (IOP0)	0xA800_0836	-	✓
	Compare match interrupt flag bit (IOP1)	0xA800_0837	-	✓
	Output level bit (OTD0)	0xA800_0838	✓	✓
	Output level bit (OTD1)	0xA800_0839	✓	✓
	Output level Inversion mode bit (CMOD)	0xA800_083C	✓	✓
	Buffer transfer selection bit (BTS0)	0xA800_083D	✓	✓
	Buffer transfer selection bit (BTS1)	0xA800_083E	✓	✓
Compare Control Register (OCU16B08_OCS)	Compare operation enable bit (CST0)	0xA800_08B0	✓	✓
	Compare operation enable bit (CST1)	0xA800_08B1	✓	✓
	Output compare buffer invalidating bit (BUF0)	0xA800_08B2	✓	✓
	Output compare buffer invalidating bit (BUF1)	0xA800_08B3	✓	✓
	Compare match interrupt enable bit (IOE0)	0xA800_08B4	✓	✓
	Compare match interrupt enable bit (IOE1)	0xA800_08B5	✓	✓
	Compare match interrupt flag bit (IOP0)	0xA800_08B6	-	✓
	Compare match interrupt flag bit (IOP1)	0xA800_08B7	-	✓
	Output level bit (OTD0)	0xA800_08B8	✓	✓
	Output level bit (OTD1)	0xA800_08B9	✓	✓
	Output level Inversion mode bit (CMOD)	0xA800_08BC	✓	✓
	Buffer transfer selection bit (BTS0)	0xA800_08BD	✓	✓
	Buffer transfer selection bit (BTS1)	0xA800_08BE	✓	✓



Register Name	Bit Name	Bit-band Alias Address	Bit Set	Bit Clear
Compare Control Register (OCU16B10_OCS)	Compare operation enable bit (CST0)	0xA800_0930	✓	✓
	Compare operation enable bit (CST1)	0xA800_0931	✓	✓
	Output compare buffer invalidating bit (BUF0)	0xA800_0932	✓	✓
	Output compare buffer invalidating bit (BUF1)	0xA800_0933	✓	✓
	Compare match interrupt enable bit (IOE0)	0xA800_0934	✓	✓
	Compare match interrupt enable bit (IOE1)	0xA800_0935	✓	✓
	Compare match interrupt flag bit (IOP0)	0xA800_0936	-	✓
	Compare match interrupt flag bit (IOP1)	0xA800_0937	-	✓
	Output level bit (OTD0)	0xA800_0938	✓	✓
	Output level bit (OTD1)	0xA800_0939	✓	✓
	Output level Inversion mode bit (CMOD)	0xA800_093C	✓	✓
	Buffer transfer selection bit (BTS0)	0xA800_093D	✓	✓
	Buffer transfer selection bit (BTS1)	0xA800_093E	✓	✓
Compare Control Register (OCU16B12_OCS)	Compare operation enable bit (CST0)	0xA808_0830	✓	✓
	Compare operation enable bit (CST1)	0xA808_0831	✓	✓
	Output compare buffer invalidating bit (BUF0)	0xA808_0832	✓	✓
	Output compare buffer invalidating bit (BUF1)	0xA808_0833	✓	✓
	Compare match interrupt enable bit (IOE0)	0xA808_0834	✓	✓
	Compare match interrupt enable bit (IOE1)	0xA808_0835	✓	✓
	Compare match interrupt flag bit (IOP0)	0xA808_0836	-	✓
	Compare match interrupt flag bit (IOP1)	0xA808_0837	-	✓
	Output level bit (OTD0)	0xA808_0838	✓	✓
	Output level bit (OTD1)	0xA808_0839	✓	✓
	Output level Inversion mode bit (CMOD)	0xA808_083C	✓	✓
	Buffer transfer selection bit (BTS0)	0xA808_083D	✓	✓
	Buffer transfer selection bit (BTS1)	0xA808_083E	✓	✓
Compare Control Register (OCU16B14_OCS)	Compare operation enable bit (CST0)	0xA808_08B0	✓	✓
	Compare operation enable bit (CST1)	0xA808_08B1	✓	✓
	Output compare buffer invalidating bit (BUF0)	0xA808_08B2	✓	✓
	Output compare buffer invalidating bit (BUF1)	0xA808_08B3	✓	✓
	Compare match interrupt enable bit (IOE0)	0xA808_08B4	✓	✓
	Compare match interrupt enable bit (IOE1)	0xA808_08B5	✓	✓
	Compare match interrupt flag bit (IOP0)	0xA808_08B6	-	✓
	Compare match interrupt flag bit (IOP1)	0xA808_08B7	-	✓
	Output level bit (OTD0)	0xA808_08B8	✓	✓
	Output level bit (OTD1)	0xA808_08B9	✓	✓
	Output level Inversion mode bit (CMOD)	0xA808_08BC	✓	✓
	Buffer transfer selection bit (BTS0)	0xA808_08BD	✓	✓
	Buffer transfer selection bit (BTS1)	0xA808_08BE	✓	✓

Register Name	Bit Name	Bit-band Alias Address	Bit Set	Bit Clear
Compare Control Register (OCU16B16_OCS)	Compare operation enable bit (CST0)	0xA808_0930	✓	✓
	Compare operation enable bit (CST1)	0xA808_0931	✓	✓
	Output compare buffer invalidating bit (BUF0)	0xA808_0932	✓	✓
	Output compare buffer invalidating bit (BUF1)	0xA808_0933	✓	✓
	Compare match interrupt enable bit (IOE0)	0xA808_0934	✓	✓
	Compare match interrupt enable bit (IOE1)	0xA808_0935	✓	✓
	Compare match interrupt flag bit (IOP0)	0xA808_0936	-	✓
	Compare match interrupt flag bit (IOP1)	0xA808_0937	-	✓
	Output level bit (OTD0)	0xA808_0938	✓	✓
	Output level bit (OTD1)	0xA808_0939	✓	✓
	Output level Inversion mode bit (CMOD)	0xA808_093C	✓	✓
	Buffer transfer selection bit (BTS0)	0xA808_093D	✓	✓
	Buffer transfer selection bit (BTS1)	0xA808_093E	✓	✓
Compare Control Register (OCU16B18_OCS)	Compare operation enable bit (CST0)	0xA808_09B0	✓	✓
	Compare operation enable bit (CST1)	0xA808_09B1	✓	✓
	Output compare buffer invalidating bit (BUF0)	0xA808_09B2	✓	✓
	Output compare buffer invalidating bit (BUF1)	0xA808_09B3	✓	✓
	Compare match interrupt enable bit (IOE0)	0xA808_09B4	✓	✓
	Compare match interrupt enable bit (IOE1)	0xA808_09B5	✓	✓
	Compare match interrupt flag bit (IOP0)	0xA808_09B6	-	✓
	Compare match interrupt flag bit (IOP1)	0xA808_09B7	-	✓
	Output level bit (OTD0)	0xA808_09B8	✓	✓
	Output level bit (OTD1)	0xA808_09B9	✓	✓
	Output level Inversion mode bit (CMOD)	0xA808_09BC	✓	✓
	Buffer transfer selection bit (BTS0)	0xA808_09BD	✓	✓
	Buffer transfer selection bit (BTS1)	0xA808_09BE	✓	✓
Compare Control Register (OCU16B20_OCS)	Compare operation enable bit (CST0)	0xA808_0A30	✓	✓
	Compare operation enable bit (CST1)	0xA808_0A31	✓	✓
	Output compare buffer invalidating bit (BUF0)	0xA808_0A32	✓	✓
	Output compare buffer invalidating bit (BUF1)	0xA808_0A33	✓	✓
	Compare match interrupt enable bit (IOE0)	0xA808_0A34	✓	✓
	Compare match interrupt enable bit (IOE1)	0xA808_0A35	✓	✓
	Compare match interrupt flag bit (IOP0)	0xA808_0A36	-	✓
	Compare match interrupt flag bit (IOP1)	0xA808_0A37	-	✓
	Output level bit (OTD0)	0xA808_0A38	✓	✓
	Output level bit (OTD1)	0xA808_0A39	✓	✓
	Output level Inversion mode bit (CMOD)	0xA808_0A3C	✓	✓
	Buffer transfer selection bit (BTS0)	0xA808_0A3D	✓	✓
	Buffer transfer selection bit (BTS1)	0xA808_0A3E	✓	✓



Register Name	Bit Name	Bit-band Alias Address	Bit Set	Bit Clear
Compare Control Register (OCU16B22_OCS)	Compare operation enable bit (CST0)	0xA808_0AB0	✓	✓
	Compare operation enable bit (CST1)	0xA808_0AB1	✓	✓
	Output compare buffer invalidating bit (BUF0)	0xA808_0AB2	✓	✓
	Output compare buffer invalidating bit (BUF1)	0xA808_0AB3	✓	✓
	Compare match interrupt enable bit (IOE0)	0xA808_0AB4	✓	✓
	Compare match interrupt enable bit (IOE1)	0xA808_0AB5	✓	✓
	Compare match interrupt flag bit (IOP0)	0xA808_0AB6	-	✓
	Compare match interrupt flag bit (IOP1)	0xA808_0AB7	-	✓
	Output level bit (OTD0)	0xA808_0AB8	✓	✓
	Output level bit (OTD1)	0xA808_0AB9	✓	✓
	Output level Inversion mode bit (CMOD)	0xA808_0ABC	✓	✓
	Buffer transfer selection bit (BTS0)	0xA808_0ABD	✓	✓
	Buffer transfer selection bit (BTS1)	0xA808_0ABE	✓	✓



## 22) 16-bit Input Capture

Register Name	Bit Name	Bit-band Alias Address	Bit Set	Bit Clear
Input capture state control Register (ICU16B00_ICS)	Input capture 0 interrupt request enable bit (ICE0)	0xA880_1034	✓	✓
	Input capture 1 interrupt request enable bit (ICE1)	0xA880_1035	✓	✓
	Input capture 0 interrupt request flag bit (ICP0)	0xA880_1036	-	✓
	Input capture 1 interrupt request flag bit (ICP1)	0xA880_1037	-	✓
Input capture state control Register (ICU16B02_ICS)	Input capture 0 interrupt request enable bit (ICE0)	0xA880_10B4	✓	✓
	Input capture 1 interrupt request enable bit (ICE1)	0xA880_10B5	✓	✓
	Input capture 0 interrupt request flag bit (ICP0)	0xA880_10B6	-	✓
	Input capture 1 interrupt request flag bit (ICP1)	0xA880_10B7	-	✓
Input capture state control Register (ICU16B04_ICS)	Input capture 0 interrupt request enable bit (ICE0)	0xA800_1034	✓	✓
	Input capture 1 interrupt request enable bit (ICE1)	0xA800_1035	✓	✓
	Input capture 0 interrupt request flag bit (ICP0)	0xA800_1036	-	✓
	Input capture 1 interrupt request flag bit (ICP1)	0xA800_1037	-	✓
Input capture state control Register (ICU16B06_ICS)	Input capture 0 interrupt request enable bit (ICE0)	0xA800_10B4	✓	✓
	Input capture 1 interrupt request enable bit (ICE1)	0xA800_10B5	✓	✓
	Input capture 0 interrupt request flag bit (ICP0)	0xA800_10B6	-	✓
	Input capture 1 interrupt request flag bit (ICP1)	0xA800_10B7	-	✓
Input capture state control Register (ICU16B08_ICS)	Input capture 0 interrupt request enable bit (ICE0)	0xA808_1034	✓	✓
	Input capture 1 interrupt request enable bit (ICE1)	0xA808_1035	✓	✓
	Input capture 0 interrupt request flag bit (ICP0)	0xA808_1036	-	✓
	Input capture 1 interrupt request flag bit (ICP1)	0xA808_1037	-	✓
Input capture state control Register (ICU16B10_ICS)	Input capture 0 interrupt request enable bit (ICE0)	0xA808_10B4	✓	✓
	Input capture 1 interrupt request enable bit (ICE1)	0xA808_10B5	✓	✓
	Input capture 0 interrupt request flag bit (ICP0)	0xA808_10B6	-	✓
	Input capture 1 interrupt request flag bit (ICP1)	0xA808_10B7	-	✓
Input capture state control Register (ICU16B12_ICS)	Input capture 0 interrupt request enable bit (ICE0)	0xA808_1134	✓	✓
	Input capture 1 interrupt request enable bit (ICE1)	0xA808_1135	✓	✓
	Input capture 0 interrupt request flag bit (ICP0)	0xA808_1136	-	✓
	Input capture 1 interrupt request flag bit (ICP1)	0xA808_1137	-	✓
Input capture state control Register (ICU16B14_ICS)	Input capture 0 interrupt request enable bit (ICE0)	0xA808_11B4	✓	✓
	Reserved	0xA808_11B5	-	-
	Input capture 0 interrupt request flag bit (ICP0)	0xA808_11B6	-	✓
	Reserved	0xA808_11B7	-	-

**Note:**

- Do not access the bit-band alias address of the reservation.





23) 12-bit 4ch A/D Converter with a Sample Hold

Register Name	Bit Name	Bit-band Alias Address	Bit Set	Bit Clear
Range comparison flag register (ADC4SH00_ADRCIF)	Range compare interrupt factor flag bit (RCINT0)	0xA880_1AA0	-	✓
	Range compare interrupt factor flag bit (RCINT1)	0xA880_1AA1	-	✓
	Range compare interrupt factor flag bit (RCINT2)	0xA880_1AA2	-	✓
	Range compare interrupt factor flag bit (RCINT3)	0xA880_1AA3	-	✓
	Range compare interrupt factor flag bit (RCINT4)	0xA880_1AA4	-	✓
	Range compare interrupt factor flag bit (RCINT5)	0xA880_1AA5	-	✓
	Range compare interrupt factor flag bit (RCINT6)	0xA880_1AA6	-	✓
	Range compare interrupt factor flag bit (RCINT7)	0xA880_1AA7	-	✓
A/D activation request/interrupt status register (ADC4SH00_ADTS1)	Interrupt request flag bit (INT)	0xA880_1AEE	-	✓
	A/D activation request busy bit (BUSY)	0xA880_1AEF	-	✓
A/D activation request/interrupt status register (ADC4SH00_ADTS0)	Interrupt request flag bit (INT)	0xA880_1AFE	-	✓
	A/D activation request busy bit (BUSY)	0xA880_1AFF	-	✓
A/D activation request/interrupt status register (ADC4SH00_ADTS3)	Interrupt request flag bit (INT)	0xA880_1B0E	-	✓
	A/D activation request busy bit (BUSY)	0xA880_1B0F	-	✓
A/D activation request/interrupt status register (ADC4SH00_ADTS2)	Interrupt request flag bit (INT)	0xA880_1B1E	-	✓
	A/D activation request busy bit (BUSY)	0xA880_1B1F	-	✓
A/D activation request/interrupt status register (ADC4SH00_ADTS5)	Interrupt request flag bit (INT)	0xA880_1B2E	-	✓
	A/D activation request busy bit (BUSY)	0xA880_1B2F	-	✓
A/D activation request/interrupt status register (ADC4SH00_ADTS4)	Interrupt request flag bit (INT)	0xA880_1B3E	-	✓
	A/D activation request busy bit (BUSY)	0xA880_1B3F	-	✓
A/D activation request/interrupt status register (ADC4SH00_ADTS7)	Interrupt request flag bit (INT)	0xA880_1B3E	-	✓
	A/D activation request busy bit (BUSY)	0xA880_1B4F	-	✓
A/D activation request/interrupt status register (ADC4SH00_ADTS6)	Interrupt request flag bit (INT)	0xA880_1B5E	-	✓
	A/D activation request busy bit (BUSY)	0xA880_1B5F	-	✓

Register Name	Bit Name	Bit-band Alias Address	Bit Set	Bit Clear
Range comparison flag register (ADC4SH01_ADRCIF)	Range compare interrupt factor flag bit (RCINT0)	0xA800_1AA0	-	✓
	Range compare interrupt factor flag bit (RCINT1)	0xA800_1AA1	-	✓
	Range compare interrupt factor flag bit (RCINT2)	0xA800_1AA2	-	✓
	Range compare interrupt factor flag bit (RCINT3)	0xA800_1AA3	-	✓
	Range compare interrupt factor flag bit (RCINT4)	0xA800_1AA4	-	✓
	Range compare interrupt factor flag bit (RCINT5)	0xA800_1AA5	-	✓
	Range compare interrupt factor flag bit (RCINT6)	0xA800_1AA6	-	✓
	Range compare interrupt factor flag bit (RCINT7)	0xA800_1AA7	-	✓
A/D activation request/interrupt status register (ADC4SH01_ADTS1)	Interrupt request flag bit (INT)	0xA800_1AEE	-	✓
	A/D activation request busy bit (BUSY)	0xA800_1AEF	-	✓
A/D activation request/interrupt status register (ADC4SH01_ADTS0)	Interrupt request flag bit (INT)	0xA800_1AFE	-	✓
	A/D activation request busy bit (BUSY)	0xA800_1AFF	-	✓
A/D activation request/interrupt status register (ADC4SH01_ADTS3)	Interrupt request flag bit (INT)	0xA800_1B0E	-	✓
	A/D activation request busy bit (BUSY)	0xA800_1B0F	-	✓
A/D activation request/interrupt status register (ADC4SH01_ADTS2)	Interrupt request flag bit (INT)	0xA800_1B1E	-	✓
	A/D activation request busy bit (BUSY)	0xA800_1B1F	-	✓
A/D activation request/interrupt status register (ADC4SH01_ADTS5)	Interrupt request flag bit (INT)	0xA800_1B2E	-	✓
	A/D activation request busy bit (BUSY)	0xA800_1B2F	-	✓
A/D activation request/interrupt status register (ADC4SH01_ADTS4)	Interrupt request flag bit (INT)	0xA800_1B3E	-	✓
	A/D activation request busy bit (BUSY)	0xA800_1B3F	-	✓
A/D activation request/interrupt status register (ADC4SH01_ADTS7)	Interrupt request flag bit (INT)	0xA800_1B3E	-	✓
	A/D activation request busy bit (BUSY)	0xA800_1B4F	-	✓
A/D activation request/interrupt status register (ADC4SH01_ADTS6)	Interrupt request flag bit (INT)	0xA800_1B5E	-	✓
	A/D activation request busy bit (BUSY)	0xA800_1B5F	-	✓



## 24) Waveform Generator

Register Name	Bit Name	Bit-band Alias Address	Bit Set	Bit Clear
16-bit Dead Timer State Control Register 2 (WFG00_DTCR2)	Interrupt request enable bit, software trigger bit (TMIE2)	0xA880_206B	✓	✓
	Interrupt request flag bit (TMIF2)	0xA880_206C	-	✓
	GATE signal control bit 4 (GTEN4)	0xA880_206D	✓	✓
	GATE signal control bit 5 (GTEN5)	0xA880_206E	✓	✓
	Output polarity control bit (DMOD2)	0xA880_206F	✓	✓
16-bit Dead Timer State Control Register 1 (WFG00_DTCR1)	Interrupt request enable bit, software trigger bit (TMIE1)	0xA880_2073	✓	✓
	Interrupt request flag bit (TMIF1)	0xA880_2074	-	✓
	GATE signal control bit 2 (GTEN2)	0xA880_2075	✓	✓
	GATE signal control bit 3 (GTEN3)	0xA880_2076	✓	✓
	Output polarity control bit (DMOD1)	0xA880_2077	✓	✓
16-bit Dead Timer State Control Register 0 (WFG00_DTCR0)	Interrupt request enable bit, software trigger bit (TMIE0)	0xA880_207B	✓	✓
	Interrupt request flag bit (TMIF0)	0xA880_207C	-	✓
	GATE signal control bit 0 (GTEN0)	0xA880_207D	✓	✓
	GATE signal control bit 1 (GTEN1)	0xA880_207E	✓	✓
	Output polarity control bit (DMOD0)	0xA880_207F	✓	✓
16-bit Dead Timer Reload Interrupt Register (WFG00_DTIR)	16-bit dead timer 0 reload interrupt enable bit (DTRIE0)	0xA880_2092	✓	✓
	16-bit dead timer 0 reload interrupt flag bit (DTRIF0)	0xA880_2093	-	✓
	16-bit dead timer 1 reload interrupt enable bit (DTRIE1)	0xA880_2094	✓	✓
	16-bit dead timer 1 reload interrupt flag bit (DTRIF1)	0xA880_2095	-	✓
	16-bit dead timer 2 reload interrupt enable bit (DTRIE2)	0xA880_2096	✓	✓
	16-bit dead timer 2 reload interrupt flag bit (DTRIF2)	0xA880_2097	-	✓
Waveform Control Register 1 (WFG00_SIGCR1)	Noise cancel function validating bit (NRSL)	0xA880_20B5	✓	✓
	DTTI Interrupt flag bit (DTIF)	0xA880_20B6	-	✓
	DTTI input validating bit (DTIE)	0xA880_20B7	✓	✓
16-bit Dead Timer State Control Register 2 (WFG01_DTCR2)	Interrupt request enable bit, software trigger bit (TMIE2)	0xA800_206B	✓	✓
	Interrupt request flag bit (TMIF2)	0xA800_206C	-	✓
	GATE signal control bit 4 (GTEN4)	0xA800_206D	✓	✓
	GATE signal control bit 5 (GTEN5)	0xA800_206E	✓	✓
	Output polarity control bit (DMOD2)	0xA800_206F	✓	✓
16-bit Dead Timer State Control Register 1 (WFG01_DTCR1)	Interrupt request enable bit, software trigger bit (TMIE1)	0xA800_2073	✓	✓
	Interrupt request flag bit (TMIF1)	0xA800_2074	-	✓
	GATE signal control bit 2 (GTEN2)	0xA800_2075	✓	✓
	GATE signal control bit 3 (GTEN3)	0xA800_2076	✓	✓
	Output polarity control bit (DMOD1)	0xA800_2077	✓	✓
16-bit Dead Timer State Control Register 0 (WFG01_DTCR0)	Interrupt request enable bit, software trigger bit (TMIE0)	0xA800_207B	✓	✓
	Interrupt request flag bit (TMIF0)	0xA800_207C	-	✓
	GATE signal control bit 0 (GTEN0)	0xA800_207D	✓	✓
	GATE signal control bit 1 (GTEN1)	0xA800_207E	✓	✓
	Output polarity control bit (DMOD0)	0xA800_207F	✓	✓
16-bit Dead Timer Reload Interrupt Register (WFG01_DTIR)	16-bit dead timer 0 reload interrupt enable bit (DTRIE0)	0xA800_2092	✓	✓
	16-bit dead timer 0 reload interrupt flag bit (DTRIF0)	0xA800_2093	-	✓
	16-bit dead timer 1 reload interrupt enable bit (DTRIE1)	0xA800_2094	✓	✓
	16-bit dead timer 1 reload interrupt flag bit (DTRIF1)	0xA800_2095	-	✓
	16-bit dead timer 2 reload interrupt enable bit (DTRIE2)	0xA800_2096	✓	✓
	16-bit dead timer 2 reload interrupt flag bit (DTRIF2)	0xA800_2097	-	✓
Waveform Control Register 1 (WFG01_SIGCR1)	Noise cancel function validating bit (NRSL)	0xA800_20B5	✓	✓
	DTTI Interrupt flag bit (DTIF)	0xA800_20B6	-	✓
	DTTI input validating bit (DTIE)	0xA800_20B7	✓	✓

Register Name	Bit Name	Bit-band Alias Address	Bit Set	Bit Clear
16-bit Dead Timer State Control Register 2 (WFG02_DTCR2)	Interrupt request enable bit, software trigger bit (TMIE2)	0xA808_306B	✓	✓
	Interrupt request flag bit (TMIF2)	0xA808_306C	-	✓
	GATE signal control bit 4 (GTEN4)	0xA808_306D	✓	✓
	GATE signal control bit 5 (GTEN5)	0xA808_306E	✓	✓
	Output polarity control bit (DMOD2)	0xA808_306F	✓	✓
16-bit Dead Timer State Control Register 1 (WFG02_DTCR1)	Interrupt request enable bit, software trigger bit (TMIE1)	0xA808_3073	✓	✓
	Interrupt request flag bit (TMIF1)	0xA808_3074	-	✓
	GATE signal control bit 2 (GTEN2)	0xA808_3075	✓	✓
	GATE signal control bit 3 (GTEN3)	0xA808_3076	✓	✓
	Output polarity control bit (DMOD1)	0xA808_3077	✓	✓
16-bit Dead Timer State Control Register 0 (WFG02_DTCR0)	Interrupt request enable bit, software trigger bit (TMIE0)	0xA808_307B	✓	✓
	Interrupt request flag bit (TMIF0)	0xA808_307C	-	✓
	GATE signal control bit 0 (GTEN0)	0xA808_307D	✓	✓
	GATE signal control bit 1 (GTEN1)	0xA808_307E	✓	✓
	Output polarity control bit (DMOD0)	0xA808_307F	✓	✓
16-bit Dead Timer Reload Interrupt Register (WFG02_DTIR)	16-bit dead timer 0 reload interrupt enable bit (DTRIE0)	0xA808_3092	✓	✓
	16-bit dead timer 0 reload interrupt flag bit (DTRIF0)	0xA808_3093	-	✓
	16-bit dead timer 1 reload interrupt enable bit (DTRIE1)	0xA808_3094	✓	✓
	16-bit dead timer 1 reload interrupt flag bit (DTRIF1)	0xA808_3095	-	✓
	16-bit dead timer 2 reload interrupt enable bit (DTRIE2)	0xA808_3096	✓	✓
	16-bit dead timer 2 reload interrupt flag bit (DTRIF2)	0xA808_3097	-	✓
Waveform Control Register 1 (WFG02_SIGCR1)	Noise cancel function validating bit (NRSL)	0xA808_30B5	✓	✓
	DTTI Interrupt flag bit (DTIF)	0xA808_30B6	-	✓
	DTTI input validating bit (DTIE)	0xA808_30B7	✓	✓
16-bit Dead Timer State Control Register 2 (WFG03_DTCR2)	Interrupt request enable bit, software trigger bit (TMIE2)	0xA808_326B	✓	✓
	Interrupt request flag bit (TMIF2)	0xA808_326C	-	✓
	GATE signal control bit 4 (GTEN4)	0xA808_326D	✓	✓
	GATE signal control bit 5 (GTEN5)	0xA808_326E	✓	✓
	Output polarity control bit (DMOD2)	0xA808_326F	✓	✓
16-bit Dead Timer State Control Register 1 (WFG03_DTCR1)	Interrupt request enable bit, software trigger bit (TMIE1)	0xA808_3273	✓	✓
	Interrupt request flag bit (TMIF1)	0xA808_3274	-	✓
	GATE signal control bit 2 (GTEN2)	0xA808_3275	✓	✓
	GATE signal control bit 3 (GTEN3)	0xA808_3276	✓	✓
	Output polarity control bit (DMOD1)	0xA808_3277	✓	✓
16-bit Dead Timer State Control Register 0 (WFG03_DTCR0)	Interrupt request enable bit, software trigger bit (TMIE0)	0xA808_327B	✓	✓
	Interrupt request flag bit (TMIF0)	0xA808_327C	-	✓
	GATE signal control bit 0 (GTEN0)	0xA808_327D	✓	✓
	GATE signal control bit 1 (GTEN1)	0xA808_327E	✓	✓
	Output polarity control bit (DMOD0)	0xA808_327F	✓	✓
16-bit Dead Timer Reload Interrupt Register (WFG03_DTIR)	16-bit dead timer 0 reload interrupt enable bit (DTRIE0)	0xA808_3292	✓	✓
	16-bit dead timer 0 reload interrupt flag bit (DTRIF0)	0xA808_3293	-	✓
	16-bit dead timer 1 reload interrupt enable bit (DTRIE1)	0xA808_3294	✓	✓
	16-bit dead timer 1 reload interrupt flag bit (DTRIF1)	0xA808_3295	-	✓
	16-bit dead timer 2 reload interrupt enable bit (DTRIE2)	0xA808_3296	✓	✓
	16-bit dead timer 2 reload interrupt flag bit (DTRIF2)	0xA808_3297	-	✓
Waveform Control Register 1 (WFG03_SIGCR1)	Noise cancel function validating bit (NRSL)	0xA808_32B5	✓	✓
	DTTI Interrupt flag bit (DTIF)	0xA808_32B6	-	✓
	DTTI input validating bit (DTIE)	0xA808_32B7	✓	✓



25) Up/Down Counter

Register Name	Bit Name	Bit-band Alias Address	Bit Set	Bit Clear
Counter control register (UDC16B00_CCRL)	Counter clear/gate selection bit (CGSC)	0xA880_2842	✓	✓
	Counter clear bit (UDCC)	0xA880_2843	-	✓
	Reload enable bit (RLDE)	0xA880_2844	✓	✓
	Compare clear enable bit (UCRE)	0xA880_2845	✓	✓
	Counter write bit (CTUT)	0xA880_2846	✓	-
Counter control register (UDC16B00_CCRH)	Internal clock division selection bit (CLKS)	0xA880_284C	✓	✓
	Counting direction change interrupt enable bit (CFIE)	0xA880_284D	✓	✓
	Counter direction change flag bit (CDCF)	0xA880_284E	-	✓
	16-bit mode selection bit (M16E)	0xA880_284F	✓	✓
Counter status register (UDC16B00_CSRL)	Underflow detection flag bit (UDFF)	0xA880_2862	-	✓
	Overflow detection flag bit (OVFF)	0xA880_2863	-	✓
	Compare result match detection flag bit (CMPF)	0xA880_2864	-	✓
	Overflow/underflow interrupt enable bit (UDIE)	0xA880_2865	✓	✓
	Compare result match interrupt enable bit (CITE)	0xA880_2866	✓	✓
	Count activation bit (CSTR)	0xA880_2867	✓	✓
Compare match detection flag register (UDC16B00_CMPFR)	Compare match detection flag bit (CMPF0)	0xA880_2AC0	-	✓
	Compare match detection flag bit (CMPF1)	0xA880_2AC1	-	✓
	Compare match detection flag bit (CMPF2)	0xA880_2AC2	-	✓
	Compare match detection flag bit (CMPF3)	0xA880_2AC3	-	✓
	Compare match detection flag bit (CMPF4)	0xA880_2AC4	-	✓
	Compare match detection flag bit (CMPF5)	0xA880_2AC5	-	✓
Counter control register (UDC16B01_CCRL)	Counter clear/gate selection bit (CGSC)	0xA880_2C42	✓	✓
	Counter clear bit (UDCC)	0xA880_2C43	-	✓
	Reload enable bit (RLDE)	0xA880_2C44	✓	✓
	Compare clear enable bit (UCRE)	0xA880_2C45	✓	✓
	Counter write bit (CTUT)	0xA880_2C46	✓	-
Counter control register (UDC16B01_CCRH)	Internal clock division selection bit (CLKS)	0xA880_2C4C	✓	✓
	Counting direction change interrupt enable bit (CFIE)	0xA880_2C4D	✓	✓
	Counter direction change flag bit (CDCF)	0xA880_2C4E	-	✓
	16-bit mode selection bit (M16E)	0xA880_2C4F	✓	✓
Counter status register (UDC16B01_CSRL)	Underflow detection flag bit (UDFF)	0xA880_2C62	-	✓
	Overflow detection flag bit (OVFF)	0xA880_2C63	-	✓
	Compare result match detection flag bit (CMPF)	0xA880_2C64	-	✓
	Overflow/underflow interrupt enable bit (UDIE)	0xA880_2C65	✓	✓
	Compare result match interrupt enable bit (CITE)	0xA880_2C66	✓	✓
	Count activation bit (CSTR)	0xA880_2C67	✓	✓
Compare match detection flag register (UDC16B01_CMPFR)	Compare match detection flag bit (CMPF0)	0xA880_2EC0	-	✓
	Compare match detection flag bit (CMPF1)	0xA880_2EC1	-	✓
	Compare match detection flag bit (CMPF2)	0xA880_2EC2	-	✓
	Compare match detection flag bit (CMPF3)	0xA880_2EC3	-	✓
	Compare match detection flag bit (CMPF4)	0xA880_2EC4	-	✓
	Compare match detection flag bit (CMPF5)	0xA880_2EC5	-	✓

Register Name	Bit Name	Bit-band Alias Address	Bit Set	Bit Clear
Counter control register (UDC16B02_CCRL)	Counter clear/gate selection bit (CGSC)	0xA800_2842	✓	✓
	Counter clear bit (UDCC)	0xA800_2843	-	✓
	Reload enable bit (RLDE)	0xA800_2844	✓	✓
	Compare clear enable bit (UCRE)	0xA800_2845	✓	✓
	Counter write bit (CTUT)	0xA800_2846	✓	-
Counter control register (UDC16B02_CCRH)	Internal clock division selection bit (CLKS)	0xA800_284C	✓	✓
	Counting direction change interrupt enable bit (CFIE)	0xA800_284D	✓	✓
	Counter direction change flag bit (CDCF)	0xA800_284E	-	✓
	16-bit mode selection bit (M16E)	0xA800_284F	✓	✓
Counter status register (UDC16B02_CSRL)	Underflow detection flag bit (UDFF)	0xA800_2862	-	✓
	Overflow detection flag bit (OVFF)	0xA800_2863	-	✓
	Compare result match detection flag bit (CMPF)	0xA800_2864	-	✓
	Overflow/underflow interrupt enable bit (UDIE)	0xA800_2865	✓	✓
	Compare result match interrupt enable bit (CITE)	0xA800_2866	✓	✓
	Count activation bit (CSTR)	0xA800_2867	✓	✓
Compare match detection flag register (UDC16B02_CMPFR)	Compare match detection flag bit (CMPF0)	0xA800_2AC0	-	✓
	Compare match detection flag bit (CMPF1)	0xA800_2AC1	-	✓
	Compare match detection flag bit (CMPF2)	0xA800_2AC2	-	✓
	Compare match detection flag bit (CMPF3)	0xA800_2AC3	-	✓
	Compare match detection flag bit (CMPF4)	0xA800_2AC4	-	✓
	Compare match detection flag bit (CMPF5)	0xA800_2AC5	-	✓
Counter control register (UDC16B03_CCRL)	Counter clear/gate selection bit (CGSC)	0xA800_2C42	✓	✓
	Counter clear bit (UDCC)	0xA800_2C43	-	✓
	Reload enable bit (RLDE)	0xA800_2C44	✓	✓
	Compare clear enable bit (UCRE)	0xA800_2C45	✓	✓
	Counter write bit (CTUT)	0xA800_2C46	✓	-
Counter control register (UDC16B03_CCRH)	Internal clock division selection bit (CLKS)	0xA800_2C4C	✓	✓
	Counting direction change interrupt enable bit (CFIE)	0xA800_2C4D	✓	✓
	Counter direction change flag bit (CDCF)	0xA800_2C4E	-	✓
	16-bit mode selection bit (M16E)	0xA800_2C4F	✓	✓
Counter status register (UDC16B03_CSRL)	Underflow detection flag bit (UDFF)	0xA800_2C62	-	✓
	Overflow detection flag bit (OVFF)	0xA800_2C63	-	✓
	Compare result match detection flag bit (CMPF)	0xA800_2C64	-	✓
	Overflow/underflow interrupt enable bit (UDIE)	0xA800_2C65	✓	✓
	Compare result match interrupt enable bit (CITE)	0xA800_2C66	✓	✓
	Count activation bit (CSTR)	0xA800_2C67	✓	✓
Compare match detection flag register (UDC16B03_CMPFR)	Compare match detection flag bit (CMPF0)	0xA800_2EC0	-	✓
	Compare match detection flag bit (CMPF1)	0xA800_2EC1	-	✓
	Compare match detection flag bit (CMPF2)	0xA800_2EC2	-	✓
	Compare match detection flag bit (CMPF3)	0xA800_2EC3	-	✓
	Compare match detection flag bit (CMPF4)	0xA800_2EC4	-	✓
	Compare match detection flag bit (CMPF5)	0xA800_2EC5	-	✓





## 26) Motor Vector Operation Accelerator

Register Name	Bit Name	Bit-band Alias Address	Bit Set	Bit Clear
Representative State Display Register (MVA00_MVARS)	Calculation end representative display bit (CED)	0xA880_4060	-	✓
	Floating-point non-normalized error display bit (FDDEF)	0xA880_4061	-	✓
	Failure detection error display bit (FDEF)	0xA880_4062	-	✓
	Calculation data update error display bit (UDEF)	0xA880_4063	-	✓
	Cumulative three-phase current abnormality detection error display bit (UACEF)	0xA880_4064	-	✓
	Reserved	0xA880_4065	-	-
	Three-phase to two-phase DC current value abnormality detection error display bit (PCVCEF)	0xA880_4066	-	✓
	Overflow display bit (OFLF)	0xA880_4068	-	✓
	Underflow display bit (UFLF)	0xA880_4069	-	✓
	Calculation overtime error display bit (COTEF)	0xA880_406A	-	✓
	R/D converter diagnosis error display bit (RDCCEF)	0xA880_406B	-	✓
End state display register (MVA00_MVAES)	Angular calculation end display bit (AGCED)	0xA880_4070	-	✓
	Three-phase current normalization end display bit (CNED)	0xA880_4071	-	✓
	Three-phase to two-phase DC conversion end display bit (DCCED)	0xA880_4072	-	✓
	PID control end display bit (PIDED)	0xA880_4073	-	✓
	Current to voltage conversion end display bit (CVCED)	0xA880_4074	-	✓
	Two-phase to three-phase AC conversion end display bit (ACCED)	0xA880_4075	-	✓
Representative State Display Register (MVA01_MVARS)	Calculation end representative display bit (CED)	0xA800_4060	-	✓
	Floating-point non-normalized error display bit (FDDEF)	0xA800_4061	-	✓
	Failure detection error display bit (FDEF)	0xA800_4062	-	✓
	Calculation data update error display bit (UDEF)	0xA800_4063	-	✓
	Cumulative three-phase current abnormality detection error display bit (UACEF)	0xA800_4064	-	✓
	Reserved	0xA800_4065	-	-
	Three-phase to two-phase DC current value abnormality detection error display bit (PCVCEF)	0xA800_4066	-	✓
	Overflow display bit (OFLF)	0xA800_4068	-	✓
	Underflow display bit (UFLF)	0xA800_4069	-	✓
	Calculation overtime error display bit (COTEF)	0xA800_406A	-	✓
	R/D converter diagnosis error display bit (RDCCEF)	0xA800_406B	-	✓
End state display register (MVA01_MVAES)	Angular calculation end display bit (AGCED)	0xA800_4070	-	✓
	Three-phase current normalization end display bit (CNED)	0xA800_4071	-	✓
	Three-phase to two-phase DC conversion end display bit (DCCED)	0xA800_4072	-	✓
	PID control end display bit (PIDED)	0xA800_4073	-	✓
	Current to voltage conversion end display bit (CVCED)	0xA800_4074	-	✓
	Two-phase to three-phase AC conversion end display bit (ACCED)	0xA800_4075	-	✓

**Note:**

- Note: Do not access the bit-band alias address of the reservation.

### 27) R/D converter

Register Name	Bit Name	Bit-band Alias Address	Bit Set	Bit Clear
R/D converter failure interrupt request enable register (RDC00_RDCICER)	Interrupt enable bit (INTE)	0xA880_6000	✓	✓
	Interrupt request flag bit (RDCEF)	0xA880_6001	-	✓
Sine cosine conversion input register (RDC00_SCCIR)	SIN/COS update flag bit (RDCUF)	0xA880_60DF	-	✓
R/D converter failure interrupt request enable register (RDC01_RDCICER)	Interrupt enable bit (INTE)	0xA800_6000	✓	✓
	Interrupt request flag bit (RDCEF)	0xA800_6001	-	✓
Sine cosine conversion input register (RDC01_SCCIR)	SIN/COS update flag bit (RDCUF)	0xA800_60DF	-	✓

### 28) 12-bit A/D Converter

Register Name	Bit Name	Bit-band Alias Address	Bit Set	Bit Clear
A/D activation trigger control status register (ADC12B_ADTCS1)	Compare register buffer transfer control bit (BTS)	0xA808_2264	✓	✓
	Compare register buffer function control bit (BUFX)	0xA808_2265	✓	✓
	A/D data register protection clear selection bit (PRTS)	0xA808_2268	✓	✓
	A/D data register protection enable bit (PRT)	0xA808_2269	✓	✓
	Repeat conversion selection bit (RPT)	0xA808_226A	✓	✓
	Interrupt request enable bit (INTE)	0xA808_226D	✓	✓
	Interrupt request flag bit (INT)	0xA808_226E	-	✓
	A/D activation request-in-progress bit (BUSY)	0xA808_226F	-	✓
A/D activation trigger control status register (ADC12B_ADTCS0)	Compare register buffer transfer control bit (BTS)	0xA808_2274	✓	✓
	Compare register buffer function control bit (BUFX)	0xA808_2275	✓	✓
	A/D data register protection clear selection bit (PRTS)	0xA808_2278	✓	✓
	A/D data register protection enable bit (PRT)	0xA808_2279	✓	✓
	Repeat conversion selection bit (RPT)	0xA808_227A	✓	✓
	Interrupt request enable bit (INTE)	0xA808_227D	✓	✓
	Interrupt request flag bit (INT)	0xA808_227E	-	✓
	A/D activation request-in-progress bit (BUSY)	0xA808_227F	-	✓
A/D activation trigger control status register (ADC12B_ADTCS3)	Compare register buffer transfer control bit (BTS)	0xA808_2284	✓	✓
	Compare register buffer function control bit (BUFX)	0xA808_2285	✓	✓
	A/D data register protection clear selection bit (PRTS)	0xA808_2288	✓	✓
	A/D data register protection enable bit (PRT)	0xA808_2289	✓	✓
	Repeat conversion selection bit (RPT)	0xA808_228A	✓	✓
	Interrupt request enable bit (INTE)	0xA808_228D	✓	✓
	Interrupt request flag bit (INT)	0xA808_228E	-	✓
	A/D activation request-in-progress bit (BUSY)	0xA808_228F	-	✓





Register Name	Bit Name	Bit-band Alias Address	Bit Set	Bit Clear
A/D activation trigger control status register (ADC12B_ADTCS2)	Compare register buffer transfer control bit (BTS)	0xA808_2294	✓	✓
	Compare register buffer function control bit (BUFX)	0xA808_2295	✓	✓
	A/D data register protection clear selection bit (PRTS)	0xA808_2298	✓	✓
	A/D data register protection enable bit (PRT)	0xA808_2299	✓	✓
	Repeat conversion selection bit (RPT)	0xA808_229A	✓	✓
	Interrupt request enable bit (INTE)	0xA808_229D	✓	✓
	Interrupt request flag bit (INT)	0xA808_229E	-	✓
	A/D activation request-in-progress bit (BUSY)	0xA808_229F	-	✓
A/D activation trigger control status register (ADC12B_ADTCS5)	Compare register buffer transfer control bit (BTS)	0xA808_22A4	✓	✓
	Compare register buffer function control bit (BUFX)	0xA808_22A5	✓	✓
	A/D data register protection clear selection bit (PRTS)	0xA808_22A8	✓	✓
	A/D data register protection enable bit (PRT)	0xA808_22A9	✓	✓
	Repeat conversion selection bit (RPT)	0xA808_22AA	✓	✓
	Interrupt request enable bit (INTE)	0xA808_22AD	✓	✓
	Interrupt request flag bit (INT)	0xA808_22AE	-	✓
	A/D activation request-in-progress bit (BUSY)	0xA808_22AF	-	✓
A/D activation trigger control status register (ADC12B_ADTCS4)	Compare register buffer transfer control bit (BTS)	0xA808_22B4	✓	✓
	Compare register buffer function control bit (BUFX)	0xA808_22B5	✓	✓
	A/D data register protection clear selection bit (PRTS)	0xA808_22B8	✓	✓
	A/D data register protection enable bit (PRT)	0xA808_22B9	✓	✓
	Repeat conversion selection bit (RPT)	0xA808_22BA	✓	✓
	Interrupt request enable bit (INTE)	0xA808_22BD	✓	✓
	Interrupt request flag bit (INT)	0xA808_22BE	-	✓
	A/D activation request-in-progress bit (BUSY)	0xA808_22BF	-	✓
A/D activation trigger control status register (ADC12B_ADTCS7)	Compare register buffer transfer control bit (BTS)	0xA808_22C4	✓	✓
	Compare register buffer function control bit (BUFX)	0xA808_22C5	✓	✓
	A/D data register protection clear selection bit (PRTS)	0xA808_22C8	✓	✓
	A/D data register protection enable bit (PRT)	0xA808_22C9	✓	✓
	Repeat conversion selection bit (RPT)	0xA808_22CA	✓	✓
	Interrupt request enable bit (INTE)	0xA808_22CD	✓	✓
	Interrupt request flag bit (INT)	0xA808_22CE	-	✓
	A/D activation request-in-progress bit (BUSY)	0xA808_22CF	-	✓
A/D activation trigger control status register (ADC12B_ADTCS6)	Compare register buffer transfer control bit (BTS)	0xA808_22D4	✓	✓
	Compare register buffer function control bit (BUFX)	0xA808_22D5	✓	✓
	A/D data register protection clear selection bit (PRTS)	0xA808_22D8	✓	✓
	A/D data register protection enable bit (PRT)	0xA808_22D9	✓	✓
	Repeat conversion selection bit (RPT)	0xA808_22DA	✓	✓
	Interrupt request enable bit (INTE)	0xA808_22DD	✓	✓
	Interrupt request flag bit (INT)	0xA808_22DE	-	✓
	A/D activation request-in-progress bit (BUSY)	0xA808_22DF	-	✓
A/D activation trigger control status register (ADC12B_ADTCS9)	Compare register buffer transfer control bit (BTS)	0xA808_22E4	✓	✓
	Compare register buffer function control bit (BUFX)	0xA808_22E5	✓	✓
	A/D data register protection clear selection bit (PRTS)	0xA808_22E8	✓	✓
	A/D data register protection enable bit (PRT)	0xA808_22E9	✓	✓
	Repeat conversion selection bit (RPT)	0xA808_22EA	✓	✓
	Interrupt request enable bit (INTE)	0xA808_22ED	✓	✓
	Interrupt request flag bit (INT)	0xA808_22EE	-	✓
	A/D activation request-in-progress bit (BUSY)	0xA808_22EF	-	✓

Register Name	Bit Name	Bit-band Alias Address	Bit Set	Bit Clear
A/D activation trigger control status register (ADC12B_ADTCS8)	Compare register buffer transfer control bit (BTS)	0xA808_22F4	✓	✓
	Compare register buffer function control bit (BUFX)	0xA808_22F5	✓	✓
	A/D data register protection clear selection bit (PRTS)	0xA808_22F8	✓	✓
	A/D data register protection enable bit (PRT)	0xA808_22F9	✓	✓
	Repeat conversion selection bit (RPT)	0xA808_22FA	✓	✓
	Interrupt request enable bit (INTE)	0xA808_22FD	✓	✓
	Interrupt request flag bit (INT)	0xA808_22FE	-	✓
	A/D activation request-in-progress bit (BUSY)	0xA808_22FF	-	✓
A/D activation trigger control status register (ADC12B_ADTCS11)	Compare register buffer transfer control bit (BTS)	0xA808_2304	✓	✓
	Compare register buffer function control bit (BUFX)	0xA808_2305	✓	✓
	A/D data register protection clear selection bit (PRTS)	0xA808_2308	✓	✓
	A/D data register protection enable bit (PRT)	0xA808_2309	✓	✓
	Repeat conversion selection bit (RPT)	0xA808_230A	✓	✓
	Interrupt request enable bit (INTE)	0xA808_230D	✓	✓
	Interrupt request flag bit (INT)	0xA808_230E	-	✓
	A/D activation request-in-progress bit (BUSY)	0xA808_230F	-	✓
A/D activation trigger control status register (ADC12B_ADTCS10)	Compare register buffer transfer control bit (BTS)	0xA808_2314	✓	✓
	Compare register buffer function control bit (BUFX)	0xA808_2315	✓	✓
	A/D data register protection clear selection bit (PRTS)	0xA808_2318	✓	✓
	A/D data register protection enable bit (PRT)	0xA808_2319	✓	✓
	Repeat conversion selection bit (RPT)	0xA808_231A	✓	✓
	Interrupt request enable bit (INTE)	0xA808_231D	✓	✓
	Interrupt request flag bit (INT)	0xA808_231E	-	✓
	A/D activation request-in-progress bit (BUSY)	0xA808_231F	-	✓
A/D activation trigger control status register (ADC12B_ADTCS13)	Compare register buffer transfer control bit (BTS)	0xA808_2324	✓	✓
	Compare register buffer function control bit (BUFX)	0xA808_2325	✓	✓
	A/D data register protection clear selection bit (PRTS)	0xA808_2328	✓	✓
	A/D data register protection enable bit (PRT)	0xA808_2329	✓	✓
	Repeat conversion selection bit (RPT)	0xA808_232A	✓	✓
	Interrupt request enable bit (INTE)	0xA808_232D	✓	✓
	Interrupt request flag bit (INT)	0xA808_232E	-	✓
	A/D activation request-in-progress bit (BUSY)	0xA808_232F	-	✓
A/D activation trigger control status register (ADC12B_ADTCS12)	Compare register buffer transfer control bit (BTS)	0xA808_2334	✓	✓
	Compare register buffer function control bit (BUFX)	0xA808_2335	✓	✓
	A/D data register protection clear selection bit (PRTS)	0xA808_2338	✓	✓
	A/D data register protection enable bit (PRT)	0xA808_2339	✓	✓
	Repeat conversion selection bit (RPT)	0xA808_233A	✓	✓
	Interrupt request enable bit (INTE)	0xA808_233D	✓	✓
	Interrupt request flag bit (INT)	0xA808_233E	-	✓
	A/D activation request-in-progress bit (BUSY)	0xA808_233F	-	✓
A/D activation trigger control status register (ADC12B_ADTCS15)	Compare register buffer transfer control bit (BTS)	0xA808_2344	✓	✓
	Compare register buffer function control bit (BUFX)	0xA808_2345	✓	✓
	A/D data register protection clear selection bit (PRTS)	0xA808_2348	✓	✓
	A/D data register protection enable bit (PRT)	0xA808_2349	✓	✓
	Repeat conversion selection bit (RPT)	0xA808_234A	✓	✓
	Interrupt request enable bit (INTE)	0xA808_234D	✓	✓
	Interrupt request flag bit (INT)	0xA808_234E	-	✓
	A/D activation request-in-progress bit (BUSY)	0xA808_234F	-	✓



Register Name	Bit Name	Bit-band Alias Address	Bit Set	Bit Clear
A/D activation trigger control status register (ADC12B_ADTCS14)	Compare register buffer transfer control bit (BTS)	0xA808_2354	✓	✓
	Compare register buffer function control bit (BUFX)	0xA808_2355	✓	✓
	A/D data register protection clear selection bit (PRTS)	0xA808_2358	✓	✓
	A/D data register protection enable bit (PRT)	0xA808_2359	✓	✓
	Repeat conversion selection bit (RPT)	0xA808_235A	✓	✓
	Interrupt request enable bit (INTE)	0xA808_235D	✓	✓
	Interrupt request flag bit (INT)	0xA808_235E	-	✓
	A/D activation request-in-progress bit (BUSY)	0xA808_235F	-	✓
A/D activation trigger control status register (ADC12B_ADTCS17)	Compare register buffer transfer control bit (BTS)	0xA808_2364	✓	✓
	Compare register buffer function control bit (BUFX)	0xA808_2365	✓	✓
	A/D data register protection clear selection bit (PRTS)	0xA808_2368	✓	✓
	A/D data register protection enable bit (PRT)	0xA808_2369	✓	✓
	Repeat conversion selection bit (RPT)	0xA808_236A	✓	✓
	Interrupt request enable bit (INTE)	0xA808_236D	✓	✓
	Interrupt request flag bit (INT)	0xA808_236E	-	✓
	A/D activation request-in-progress bit (BUSY)	0xA808_236F	-	✓
A/D activation trigger control status register (ADC12B_ADTCS16)	Compare register buffer transfer control bit (BTS)	0xA808_2374	✓	✓
	Compare register buffer function control bit (BUFX)	0xA808_2375	✓	✓
	A/D data register protection clear selection bit (PRTS)	0xA808_2378	✓	✓
	A/D data register protection enable bit (PRT)	0xA808_2379	✓	✓
	Repeat conversion selection bit (RPT)	0xA808_237A	✓	✓
	Interrupt request enable bit (INTE)	0xA808_237D	✓	✓
	Interrupt request flag bit (INT)	0xA808_237E	-	✓
	A/D activation request-in-progress bit (BUSY)	0xA808_237F	-	✓
A/D activation trigger control status register (ADC12B_ADTCS19)	Compare register buffer transfer control bit (BTS)	0xA808_2384	✓	✓
	Compare register buffer function control bit (BUFX)	0xA808_2385	✓	✓
	A/D data register protection clear selection bit (PRTS)	0xA808_2388	✓	✓
	A/D data register protection enable bit (PRT)	0xA808_2389	✓	✓
	Repeat conversion selection bit (RPT)	0xA808_238A	✓	✓
	Interrupt request enable bit (INTE)	0xA808_238D	✓	✓
	Interrupt request flag bit (INT)	0xA808_238E	-	✓
	A/D activation request-in-progress bit (BUSY)	0xA808_238F	-	✓
A/D activation trigger control status register (ADC12B_ADTCS18)	Compare register buffer transfer control bit (BTS)	0xA808_2394	✓	✓
	Compare register buffer function control bit (BUFX)	0xA808_2395	✓	✓
	A/D data register protection clear selection bit (PRTS)	0xA808_2398	✓	✓
	A/D data register protection enable bit (PRT)	0xA808_2399	✓	✓
	Repeat conversion selection bit (RPT)	0xA808_239A	✓	✓
	Interrupt request enable bit (INTE)	0xA808_239D	✓	✓
	Interrupt request flag bit (INT)	0xA808_239E	-	✓
	A/D activation request-in-progress bit (BUSY)	0xA808_239F	-	✓
A/D activation trigger control status register (ADC12B_ADTCS21)	Compare register buffer transfer control bit (BTS)	0xA808_23A4	✓	✓
	Compare register buffer function control bit (BUFX)	0xA808_23A5	✓	✓
	A/D data register protection clear selection bit (PRTS)	0xA808_23A8	✓	✓
	A/D data register protection enable bit (PRT)	0xA808_23A9	✓	✓
	Repeat conversion selection bit (RPT)	0xA808_23AA	✓	✓
	Interrupt request enable bit (INTE)	0xA808_23AD	✓	✓
	Interrupt request flag bit (INT)	0xA808_23AE	-	✓
	A/D activation request-in-progress bit (BUSY)	0xA808_23AF	-	✓

Register Name	Bit Name	Bit-band Alias Address	Bit Set	Bit Clear
A/D activation trigger control status register (ADC12B_ADTCS20)	Compare register buffer transfer control bit (BTS)	0xA808_23B4	✓	✓
	Compare register buffer function control bit (BUFX)	0xA808_23B5	✓	✓
	A/D data register protection clear selection bit (PRTS)	0xA808_23B8	✓	✓
	A/D data register protection enable bit (PRT)	0xA808_23B9	✓	✓
	Repeat conversion selection bit (RPT)	0xA808_23BA	✓	✓
	Interrupt request enable bit (INTE)	0xA808_23BD	✓	✓
	Interrupt request flag bit (INT)	0xA808_23BE	-	✓
	A/D activation request-in-progress bit (BUSY)	0xA808_23BF	-	✓
A/D activation trigger control status register (ADC12B_ADTCS23)	Compare register buffer transfer control bit (BTS)	0xA808_23C4	✓	✓
	Compare register buffer function control bit (BUFX)	0xA808_23C5	✓	✓
	A/D data register protection clear selection bit (PRTS)	0xA808_23C8	✓	✓
	A/D data register protection enable bit (PRT)	0xA808_23C9	✓	✓
	Repeat conversion selection bit (RPT)	0xA808_23CA	✓	✓
	Interrupt request enable bit (INTE)	0xA808_23CD	✓	✓
	Interrupt request flag bit (INT)	0xA808_23CE	-	✓
	A/D activation request-in-progress bit (BUSY)	0xA808_23CF	-	✓
A/D activation trigger control status register (ADC12B_ADTCS22)	Compare register buffer transfer control bit (BTS)	0xA808_23D4	✓	✓
	Compare register buffer function control bit (BUFX)	0xA808_23D5	✓	✓
	A/D data register protection clear selection bit (PRTS)	0xA808_23D8	✓	✓
	A/D data register protection enable bit (PRT)	0xA808_23D9	✓	✓
	Repeat conversion selection bit (RPT)	0xA808_23DA	✓	✓
	Interrupt request enable bit (INTE)	0xA808_23DD	✓	✓
	Interrupt request flag bit (INT)	0xA808_23DE	-	✓
	A/D activation request-in-progress bit (BUSY)	0xA808_23DF	-	✓
A/D activation trigger control status register (ADC12B_ADTCS25)	Compare register buffer transfer control bit (BTS)	0xA808_23E4	✓	✓
	Compare register buffer function control bit (BUFX)	0xA808_23E5	✓	✓
	A/D data register protection clear selection bit (PRTS)	0xA808_23E8	✓	✓
	A/D data register protection enable bit (PRT)	0xA808_23E9	✓	✓
	Repeat conversion selection bit (RPT)	0xA808_23EA	✓	✓
	Interrupt request enable bit (INTE)	0xA808_23ED	✓	✓
	Interrupt request flag bit (INT)	0xA808_23EE	-	✓
	A/D activation request-in-progress bit (BUSY)	0xA808_23EF	-	✓
A/D activation trigger control status register (ADC12B_ADTCS24)	Compare register buffer transfer control bit (BTS)	0xA808_23F4	✓	✓
	Compare register buffer function control bit (BUFX)	0xA808_23F5	✓	✓
	A/D data register protection clear selection bit (PRTS)	0xA808_23F8	✓	✓
	A/D data register protection enable bit (PRT)	0xA808_23F9	✓	✓
	Repeat conversion selection bit (RPT)	0xA808_23FA	✓	✓
	Interrupt request enable bit (INTE)	0xA808_23FD	✓	✓
	Interrupt request flag bit (INT)	0xA808_23FE	-	✓
	A/D activation request-in-progress bit (BUSY)	0xA808_23FF	-	✓
A/D activation trigger control status register (ADC12B_ADTCS27)	Compare register buffer transfer control bit (BTS)	0xA808_2404	✓	✓
	Compare register buffer function control bit (BUFX)	0xA808_2405	✓	✓
	A/D data register protection clear selection bit (PRTS)	0xA808_2408	✓	✓
	A/D data register protection enable bit (PRT)	0xA808_2409	✓	✓
	Repeat conversion selection bit (RPT)	0xA808_240A	✓	✓
	Interrupt request enable bit (INTE)	0xA808_240D	✓	✓
	Interrupt request flag bit (INT)	0xA808_240E	-	✓
	A/D activation request-in-progress bit (BUSY)	0xA808_240F	-	✓



Register Name	Bit Name	Bit-band Alias Address	Bit Set	Bit Clear
A/D activation trigger control status register (ADC12B_ADTCS26)	Compare register buffer transfer control bit (BTS)	0xA808_2414	✓	✓
	Compare register buffer function control bit (BUFX)	0xA808_2415	✓	✓
	A/D data register protection clear selection bit (PRTS)	0xA808_2418	✓	✓
	A/D data register protection enable bit (PRT)	0xA808_2419	✓	✓
	Repeat conversion selection bit (RPT)	0xA808_241A	✓	✓
	Interrupt request enable bit (INTE)	0xA808_241D	✓	✓
	Interrupt request flag bit (INT)	0xA808_241E	-	✓
	A/D activation request-in-progress bit (BUSY)	0xA808_241F	-	✓
A/D activation trigger control status register (ADC12B_ADTCS29)	Compare register buffer transfer control bit (BTS)	0xA808_2424	✓	✓
	Compare register buffer function control bit (BUFX)	0xA808_2425	✓	✓
	A/D data register protection clear selection bit (PRTS)	0xA808_2428	✓	✓
	A/D data register protection enable bit (PRT)	0xA808_2429	✓	✓
	Repeat conversion selection bit (RPT)	0xA808_242A	✓	✓
	Interrupt request enable bit (INTE)	0xA808_242D	✓	✓
	Interrupt request flag bit (INT)	0xA808_242E	-	✓
	A/D activation request-in-progress bit (BUSY)	0xA808_242F	-	✓
A/D activation trigger control status register (ADC12B_ADTCS28)	Compare register buffer transfer control bit (BTS)	0xA808_2434	✓	✓
	Compare register buffer function control bit (BUFX)	0xA808_2435	✓	✓
	A/D data register protection clear selection bit (PRTS)	0xA808_2438	✓	✓
	A/D data register protection enable bit (PRT)	0xA808_2439	✓	✓
	Repeat conversion selection bit (RPT)	0xA808_243A	✓	✓
	Interrupt request enable bit (INTE)	0xA808_243D	✓	✓
	Interrupt request flag bit (INT)	0xA808_243E	-	✓
	A/D activation request-in-progress bit (BUSY)	0xA808_243F	-	✓
A/D activation trigger control status register (ADC12B_ADTCS31)	Compare register buffer transfer control bit (BTS)	0xA808_2444	✓	✓
	Compare register buffer function control bit (BUFX)	0xA808_2445	✓	✓
	A/D data register protection clear selection bit (PRTS)	0xA808_2448	✓	✓
	A/D data register protection enable bit (PRT)	0xA808_2449	✓	✓
	Repeat conversion selection bit (RPT)	0xA808_244A	✓	✓
	Interrupt request enable bit (INTE)	0xA808_244D	✓	✓
	Interrupt request flag bit (INT)	0xA808_244E	-	✓
	A/D activation request-in-progress bit (BUSY)	0xA808_244F	-	✓
A/D activation trigger control status register (ADC12B_ADTCS30)	Compare register buffer transfer control bit (BTS)	0xA808_2454	✓	✓
	Compare register buffer function control bit (BUFX)	0xA808_2455	✓	✓
	A/D data register protection clear selection bit (PRTS)	0xA808_2458	✓	✓
	A/D data register protection enable bit (PRT)	0xA808_2459	✓	✓
	Repeat conversion selection bit (RPT)	0xA808_245A	✓	✓
	Interrupt request enable bit (INTE)	0xA808_245D	✓	✓
	Interrupt request flag bit (INT)	0xA808_245E	-	✓
	A/D activation request-in-progress bit (BUSY)	0xA808_245F	-	✓

Register Name	Bit Name	Bit-band Alias Address	Bit Set	Bit Clear
Range comparison flag register (ADC12B_ADRCIF)	Range compare interrupt factor flag bit (RCINT0)	0xA808_2A00	-	✓
	Range compare interrupt factor flag bit (RCINT1)	0xA808_2A01	-	✓
	Range compare interrupt factor flag bit (RCINT2)	0xA808_2A02	-	✓
	Range compare interrupt factor flag bit (RCINT3)	0xA808_2A03	-	✓
	Range compare interrupt factor flag bit (RCINT4)	0xA808_2A04	-	✓
	Range compare interrupt factor flag bit (RCINT5)	0xA808_2A05	-	✓
	Range compare interrupt factor flag bit (RCINT6)	0xA808_2A06	-	✓
	Range compare interrupt factor flag bit (RCINT7)	0xA808_2A07	-	✓
	Range compare interrupt factor flag bit (RCINT8)	0xA808_2A08	-	✓
	Range compare interrupt factor flag bit (RCINT9)	0xA808_2A09	-	✓
	Range compare interrupt factor flag bit (RCINT10)	0xA808_2A0A	-	✓
	Range compare interrupt factor flag bit (RCINT11)	0xA808_2A0B	-	✓
	Range compare interrupt factor flag bit (RCINT12)	0xA808_2A0C	-	✓
	Range compare interrupt factor flag bit (RCINT13)	0xA808_2A0D	-	✓
	Range compare interrupt factor flag bit (RCINT14)	0xA808_2A0E	-	✓
	Range compare interrupt factor flag bit (RCINT15)	0xA808_2A0F	-	✓
	Range compare interrupt factor flag bit (RCINT16)	0xA808_2A10	-	✓
	Range compare interrupt factor flag bit (RCINT17)	0xA808_2A11	-	✓
	Range compare interrupt factor flag bit (RCINT18)	0xA808_2A12	-	✓
	Range compare interrupt factor flag bit (RCINT19)	0xA808_2A13	-	✓
	Range compare interrupt factor flag bit (RCINT20)	0xA808_2A14	-	✓
	Range compare interrupt factor flag bit (RCINT21)	0xA808_2A15	-	✓
	Range compare interrupt factor flag bit (RCINT22)	0xA808_2A16	-	✓
	Range compare interrupt factor flag bit (RCINT23)	0xA808_2A17	-	✓
	Range compare interrupt factor flag bit (RCINT24)	0xA808_2A18	-	✓
	Range compare interrupt factor flag bit (RCINT25)	0xA808_2A19	-	✓
	Range compare interrupt factor flag bit (RCINT26)	0xA808_2A1A	-	✓
	Range compare interrupt factor flag bit (RCINT27)	0xA808_2A1B	-	✓
	Range compare interrupt factor flag bit (RCINT28)	0xA808_2A1C	-	✓
	Range compare interrupt factor flag bit (RCINT29)	0xA808_2A1D	-	✓
	Range compare interrupt factor flag bit (RCINT30)	0xA808_2A1E	-	✓
	Range compare interrupt factor flag bit (RCINT31)	0xA808_2A1F	-	✓
Scan conversion control status register (ADC12B_ADSCANS0)	Continuous and stop scan conversion mode select bit (SCMD)	0xA808_2A3D	✓	✓
	Scan conversion completion interrupt request enable bit (SCIE)	0xA808_2A3E	✓	✓
	Scan conversion completion interrupt factor flag bit (SCINT)	0xA808_2A3F	-	✓

## 29) FlexRay

Register Name	Bit Name	Bit-band Alias Address	Bit Set	Bit Clear
Flag register (FLXRY_CIF1F)	Timer 0 interrupt request bit (TREQ0)	0xA880_8040	-	✓
	Timer 1 interrupt request bit (TREQ1)	0xA880_8041	-	✓
	DMA request flag bit of input buffer host (DREQI)	0xA880_8042	-	✓
	DMA request flag bit of output buffer (DREQO)	0xA880_8043	-	✓





### 30) Clock for FlexRay/RDC

Register Name	Bit Name	Bit-band Alias Address	Bit Set	Bit Clear
FlexRay/RDC PLL clock output control flag register (ERAYP_CLKR2F)	FlexRay/RDC PLL alarm interrupt request flag bit (FPOVIR)	0xA888_004E	-	✓
Automatic gear control flag register (ERAYP_PLL2CTRLF)	Gear up interrupt flag bit (GRUP)	0xA888_0058	-	✓
	Gear down interrupt flag bit (GRDN)	0xA888_005A	-	✓



## CHAPTER 22: DMA Controller

This section explains DMA controller.

---

1. Overview
2. Configuration
3. Operation
4. Registers





## 1. Overview

The DMA Controller (DMAC) implements Direct Memory Access (DMA) with little CPU intervention. DMAC is performing complex data transfers through N DMA channels. This section describes the features and the block diagram of DMA Controller.

### Feature of DMA Controller

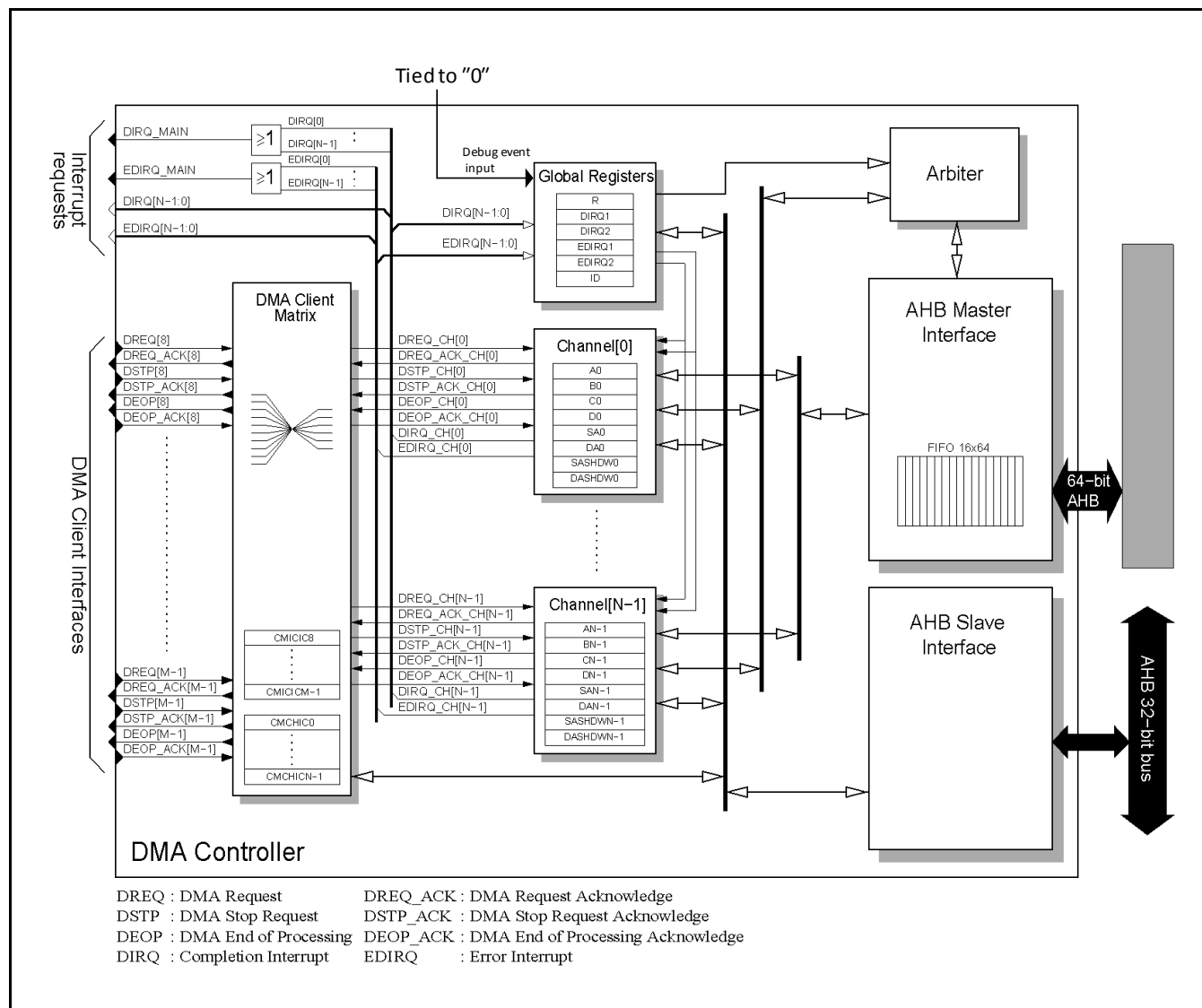
The following items are the features related to data transfer by the DMA Controller:

- Data can be transferred independently over multiple channels
- A high number of MCU internal and MCU DMA clients can be assigned to the available DMA channels
- Flexible priority between DMA channels (fixed, dynamic, or round-robin)
- DMA transfer request sources
  - Hardware request (internal clients)
  - Software request (register write)
- Transfer modes
  - Block transfer and burst transfer
- Addressing: full 32-bit (incrementing, decrementing, or fixed)
- Data types: 8-, 16-, 32-, or 64-bit wide data

## 2. Configuration

This section shows the block diagram.

Figure 2-1 Block Diagram of DMA Controller





### 3. Operation

This section describes the operation of the DMA controller

#### (1) Features of DMA Controller

- DMA Client Matrix, routing "M" DMA clients to "N" DMA channels
- DMA trigger
  - Hardware request (internal clients)
  - Software request
- Two transfer modes
  - Block transfer
  - Burst transfer
- 8-, 16-, 32-, or 64-bit wide data transfers
- Master transactions can be done in user or privileged mode
- Writing the configuration registers can only be done in privileged mode and 8-, 16-, or 32-bit wide. Illegal accesses result in an error response
- Reading the configuration registers can be done in user or privileged mode
- Incrementing, decrementing, or fixed addressing independent for source and destination
- Independent source and destination access protection
- Central interrupt flag register for completion and error interrupts
- Stop status per channel for analysis by ISRs or debugging
- Shadow registers for source and destination address
- Configurable debug event behavior (continue, halt, or stop)
- Three channel arbitration schemes
  - Fixed priority
  - Dynamic priority
  - Round-robin

#### (2) Global Functions of the DMA Controller

##### DMAC Enabling or Halting

After reset the DMA Controller is disabled. DMAC is enabled by setting DMA Enable (DMAi\_R:DE) to "1". If DMA Enable is set to "1" the individual DMA Channel Enable (DMAi\_An:EB) settings become effective. Setting DMA Enable to "0" disables the complete DMAC. Setting DMA Enable to "0" while there is a not completed transfer running on a DMA channel, the running transfer is stopped when its current block of data was transferred, and an error interrupt is issued. In case it is the last block of data of this DMA transfer a completion interrupt is issued. DMA channels which have a DMA transfer pending, but are not currently served are disabled and an error interrupt is issued. DMA channels which are enabled but have not started a DMA transfer (channel was enabled and there was not a transfer request yet) are simply disabled without issuing an error interrupt.

The DMA Controller can be halted completely (all DMA channels) by writing a single bit, DMA Halt (DMAi\_R:DH). When this bit is set to "1", all DMA channels are requested to halt and not perform DMA transfers until this bit is cleared. After DMA Halt is cleared the halted DMA transfers continue at the point they were halted. If DMA Halt is set to "1" while a transfer is running, halting takes place once the current block of data of the running transfer is transferred. Depending on the clock ratio of the DMAC/source clock domain and DMAC/destination clock domain a considerable time can elapse between the time when the halt request was set and when the DMAC is actually halted.

Global disable and halt request condition of the DMAC is indicated by DMA Stop/Halt Request Flag (DMAi\_R:DSHR). DMA Stop/Halt Request Flag is "0" if none of the below disable or halt request conditions is true:

- DMAi\_R:DE bit is set to "0"
- DMAi\_R:DH bit is set to "1"
- DMAi\_R:DBE="1", DMAi\_R:DB[1:0]="10" (stop on debug event), and a "debug event" is pending
- DMAi\_R:DBE="1", DMAi\_R:DB[1:0]="01" (halt on debug event), and a "debug event" is pending

If any of the above conditions is true, DMA Stop/Halt Request Flag is "1" indicating that DMA transfers of all channels are requested to halt or stop.

The condition that all channels are halted or have come to a stop after a global halt or stop request is indicated by DMA Stop/Halt Status Flag (DMAi\_R:DSHS). DMA Stop/Halt Status Flag is "0" if one or more channels are not yet stopped or halted and "1" if all channels are stopped or halted

### (3) DMAC Debug Behavior

The DMA Controller can be configured to react in a predefined way on a "debug event" (e.g. debugger break point). The feature to react on a "debug event" can be enabled with bit Debug Enable (DMAi\_R:DBE). If enabled the DMACs behavior depend on the setting of Debug Behavior (DMAi\_R:DB[1:0]). This feature is disabled after reset.

The behavior can be configured to stop all transfers (DMAi\_R:DB[1:0]="10"), or to halt all transfers (DMAi\_R:DB[1:0]="01"), or just to continue operation independent of debug events (DMAi\_R:DB[1:0]="00"). Initial value is to continue operation independent of debug events.

**Note:**

- *The debug feature cannot be used with this product.*



## 3.1. DMA Channels

This section describes behavior of the DMA channel.

### (1) Modes of Operation

The DMA channels can operate in two modes:

- Block transfer mode
- Burst transfer mode

The mode must be set with bits Mode Select (DMAi\_Bn:MS[1:0]). After reset, the channel is set to block transfer mode.

### (2) Block Transfer Mode

In block transfer mode the DMA client will request the transfers of a specified number of blocks of data. The number of blocks to be transferred is specified with Transfer Count (DMAi\_An:TC[15:0]). Each block of data is transferred in one arbitration phase of the DMA arbiter. For each block to transfer a DMA transfer request, either a hardware request or software request is needed. The DMA client or the software continues to give requests until the DMAC has transferred the specified number of blocks and thus the DMA transfer is completed. The DMA transfer will be successfully completed if all blocks of data were transferred without error or unsuccessfully completed if an error condition occurred.

After each transferred block of data the DMA arbiter does another arbitration and proceed with the next requesting channel with the highest priority. The arbitration depends on the selected arbitration scheme and the set priorities of the requesting channels (for details about arbitration see Section "3.3. DMA Arbiter").

### (3) DMA Transfer Requests

#### 1. Hardware request

For a channel hardware request Input Select (DMAi\_An:IS) need to be set to "01". The DMA client which is routed through the DMA Client Matrix to the channel will give the trigger by asserting DREQ. See Section "3.2. DMA Client Matrix" for the function and configuration of the DMA Client Matrix.

#### 2. Software request

For a software request Input Select need to be set to "00" and Software Trigger (DMAi\_An:ST) must be set to "1". Software Trigger shall be set to "1" if the channel is ready to receive a software trigger, which is indicated with Software Trigger Ready (DMAi\_Bn:SR) and no error condition is pending (DMAi\_Bn:SS[2:0] is "000" or "101"). If Software Trigger is tried to set to "1" while Software Trigger Ready is "0", it is ignored. Software Trigger is automatically cleared by hardware once the trigger is accepted or an error condition occurs. The error conditions can be:

- DMA channel is disabled right after setting Software Trigger
- A debug event occurred and DMAC is configured to stop all transfers on a debug event
- The master interface receives an error response and the CPU sets the Software Trigger before receiving an error interrupt caused by the AHB error

**a) Block of Data**

A block of data is determined by the setting of Block Count (DMAi\_An:BC[3:0]) and Transfer Width (DMAi\_Bn:TW[1:0]). The DMA Controller will make DMAi\_An:BC + 1 data transfers from source address range starting at Source Address (DMAi\_SAn:SA) to destination address range starting at Destination Address (DMAi\_DAn:DA). If DMAi\_An:BC is set to "0" a single data transfer from Source Address (DMAi\_SAn:SA) to Destination Address (DMAi\_DAn:DA) will be done. The settings of Block Count (DMAi\_An:BC[3:0]), Beat Limit (DMAi\_An:BL[1:0]), Alternate (DMAi\_An:AL), and Transfer Width (DMAi\_Bn:TW[1:0]) define how the AHB master interface issues the DMAi\_An:BC + 1 data transfers. The following table shows all possible combinations between DMAi\_An:BC, DMAi\_An:BL, and DMAi\_An:AL. Only these three influence the sequence of AHB transfers, whereas DMAi\_Bn:TW only affects the data size which will be transferred.



Table 3-1 Block Count/Beat Limit/Alternate Combinations

Block Count	Beat Limit	Alternate	Resulting Sequence of AHB Transfers
0	SINGLE	0	1x SINGLE RD + 1x SINGLE WR
1	SINGLE	0	2x SINGLE RD + 2x SINGLE WR
2	SINGLE	0	3x SINGLE RD + 3x SINGLE WR
3	SINGLE	0	4x SINGLE RD + 4x SINGLE WR
4	SINGLE	0	5x SINGLE RD + 5x SINGLE WR
5	SINGLE	0	6x SINGLE RD + 6x SINGLE WR
6	SINGLE	0	7x SINGLE RD + 7x SINGLE WR
7	SINGLE	0	8x SINGLE RD + 8x SINGLE WR
8	SINGLE	0	9x SINGLE RD + 9x SINGLE WR
9	SINGLE	0	10x SINGLE RD + 10x SINGLE WR
10	SINGLE	0	11x SINGLE RD + 11x SINGLE WR
11	SINGLE	0	12x SINGLE RD + 12x SINGLE WR
12	SINGLE	0	13x SINGLE RD + 13x SINGLE WR
13	SINGLE	0	14x SINGLE RD + 14x SINGLE WR
14	SINGLE	0	15x SINGLE RD + 15x SINGLE WR
15	SINGLE	0	16x SINGLE RD + 16x SINGLE WR
0	SINGLE	1	1x (1x SINGLE RD + 1x SINGLE WR)
1	SINGLE	1	2x (1x SINGLE RD + 1x SINGLE WR)
2	SINGLE	1	3x (1x SINGLE RD + 1x SINGLE WR)
3	SINGLE	1	4x (1x SINGLE RD + 1x SINGLE WR)
4	SINGLE	1	5x (1x SINGLE RD + 1x SINGLE WR)
5	SINGLE	1	6x (1x SINGLE RD + 1x SINGLE WR)
6	SINGLE	1	7x (1x SINGLE RD + 1x SINGLE WR)
7	SINGLE	1	8x (1x SINGLE RD + 1x SINGLE WR)
8	SINGLE	1	9x (1x SINGLE RD + 1x SINGLE WR)
9	SINGLE	1	10x (1x SINGLE RD + 1x SINGLE WR)
10	SINGLE	1	11x (1x SINGLE RD + 1x SINGLE WR)
11	SINGLE	1	12x (1x SINGLE RD + 1x SINGLE WR)
12	SINGLE	1	13x (1x SINGLE RD + 1x SINGLE WR)
13	SINGLE	1	14x (1x SINGLE RD + 1x SINGLE WR)
14	SINGLE	1	15x (1x SINGLE RD + 1x SINGLE WR)
15	SINGLE	1	16x (1x SINGLE RD + 1x SINGLE WR)
0	INCR4	0	1x SINGLE RD + 1x SINGLE WR
1	INCR4	0	2x SINGLE RD + 2x SINGLE WR
2	INCR4	0	3x SINGLE RD + 3x SINGLE WR
3	INCR4	0	1x 4_BEAT RD + 1x 4_BEAT WR
4	INCR4	0	1x 4_BEAT RD + 1x SINGLE RD + 1x 4_BEAT WR + 1x SINGLE WR
5	INCR4	0	1x 4_BEAT RD + 2x SINGLE RD + 1x 4_BEAT WR + 2x SINGLE WR
6	INCR4	0	1x 4_BEAT RD + 3x SINGLE RD + 1x 4_BEAT WR + 3x SINGLE WR
7	INCR4	0	2x 4_BEAT RD + 2x 4_BEAT WR
8	INCR4	0	2x 4_BEAT RD + 1x SINGLE RD + 2x 4_BEAT WR + 1x SINGLE WR
9	INCR4	0	2x 4_BEAT RD + 2x SINGLE RD + 2x 4_BEAT WR + 2x SINGLE WR
10	INCR4	0	2x 4_BEAT RD + 3x SINGLE RD + 2x 4_BEAT WR + 3x SINGLE WR
11	INCR4	0	3x 4_BEAT RD + 3x 4_BEAT WR
12	INCR4	0	3x 4_BEAT RD + 1x SINGLE RD + 3x 4_BEAT WR + 1x SINGLE WR
13	INCR4	0	3x 4_BEAT RD + 2x SINGLE RD + 3x 4_BEAT WR + 2x SINGLE WR
14	INCR4	0	3x 4_BEAT RD + 3x SINGLE RD + 3x 4_BEAT WR + 3x SINGLE WR
15	INCR4	0	4x 4_BEAT RD + 4x 4_BEAT WR

Block Count	Beat Limit	Alternate	Resulting Sequence of AHB Transfers
0	INCR4	1	1x SINGLE RD + 1x SINGLE WR
1	INCR4	1	2x (1x SINGLE RD + 1x SINGLE WR)
2	INCR4	1	3x (1x SINGLE RD + 1x SINGLE WR)
3	INCR4	1	1x (1x 4_BEAT RD + 1x 4_BEAT WR)
4	INCR4	1	1x (1x 4_BEAT RD + 1x 4_BEAT WR) + 1x SINGLE RD + 1x SINGLE WR
5	INCR4	1	1x (1x 4_BEAT RD + 1x 4_BEAT WR) + 2x (1x SINGLE RD + 1x SINGLE WR)
6	INCR4	1	1x (1x 4_BEAT RD + 1x 4_BEAT WR) + 3x (1x SINGLE RD + 1x SINGLE WR)
7	INCR4	1	2x (1x 4_BEAT RD + 1x 4_BEAT WR)
8	INCR4	1	2x (1x 4_BEAT RD + 1x 4_BEAT WR) + 1x SINGLE RD + 1x SINGLE WR
9	INCR4	1	2x (1x 4_BEAT RD + 1x 4_BEAT WR) + 2x (1x SINGLE RD + 1x SINGLE WR)
10	INCR4	1	2x (1x 4_BEAT RD + 1x 4_BEAT WR) + 3x (1x SINGLE RD + 1x SINGLE WR)
11	INCR4	1	3x (1x 4_BEAT RD + 1x 4_BEAT WR)
12	INCR4	1	3x (1x 4_BEAT RD + 1x 4_BEAT WR) + 1x SINGLE RD + 1x SINGLE WR
13	INCR4	1	3x (1x 4_BEAT RD + 1x 4_BEAT WR) + 2x (1x SINGLE RD + 1x SINGLE WR)
14	INCR4	1	3x (1x 4_BEAT RD + 1x 4_BEAT WR) + 3x (1x SINGLE RD + 1x SINGLE WR)
15	INCR4	1	4x (1x 4_BEAT RD + 1x 4_BEAT WR)
0	INCR8	0	1x SINGLE RD + 1x SINGLE WR
1	INCR8	0	2x SINGLE RD + 2x SINGLE WR
2	INCR8	0	3x SINGLE RD + 3x SINGLE WR
3	INCR8	0	1x 4_BEAT RD + 1x 4_BEAT WR
4	INCR8	0	1x 4_BEAT RD + 1x SINGLE RD + 1x 4_BEAT WR + 1x SINGLE WR
5	INCR8	0	1x 4_BEAT RD + 2x SINGLE RD + 1x 4_BEAT WR + 2x SINGLE WR
6	INCR8	0	1x 4_BEAT RD + 3x SINGLE RD + 1x 4_BEAT WR + 3x SINGLE WR
7	INCR8	0	1x 8_BEAT RD + 1x 8_BEAT WR
8	INCR8	0	1x 8_BEAT RD + 1x SINGLE RD + 1x 8_BEAT WR + 1x SINGLE WR
9	INCR8	0	1x 8_BEAT RD + 2x SINGLE RD + 1x 8_BEAT WR + 2x SINGLE WR
10	INCR8	0	1x 8_BEAT RD + 3x SINGLE RD + 1x 8_BEAT WR + 3x SINGLE WR
11	INCR8	0	1x 8_BEAT RD + 1x 4_BEAT RD + 1x 8_BEAT WR + 1x 4_BEAT WR
12	INCR8	0	1x 8_BEAT RD + 1x 4_BEAT RD + 1x SINGLE RD + 1x 8_BEAT WR + 1x 4_BEAT WR + 1x SINGLE WR
13	INCR8	0	1x 8_BEAT RD + 1x 4_BEAT RD + 2x SINGLE RD + 1x 8_BEAT WR + 1x 4_BEAT WR + 2x SINGLE WR
14	INCR8	0	1x 8_BEAT RD + 1x 4_BEAT RD + 3x SINGLE RD + 1x 8_BEAT WR + 1x 4_BEAT WR + 3x SINGLE WR
15	INCR8	0	2x 8_BEAT RD + 2x 8_BEAT WR
0	INCR8	1	1x SINGLE RD + 1x SINGLE WR
1	INCR8	1	2x (1x SINGLE RD + 1x SINGLE WR)
2	INCR8	1	3x (1x SINGLE RD + 1x SINGLE WR)
3	INCR8	1	1x (1x 4_BEAT RD + 1x 4_BEAT WR)
4	INCR8	1	1x (1x 4_BEAT RD + 1x 4_BEAT WR) + 1x SINGLE RD + 1x SINGLE WR
5	INCR8	1	1x (1x 4_BEAT RD + 1x 4_BEAT WR) + 2x (1x SINGLE RD + 1x SINGLE WR)
6	INCR8	1	1x (1x 4_BEAT RD + 1x 4_BEAT WR) + 3x (1x SINGLE RD + 1x SINGLE WR)
7	INCR8	1	1x (1x 8_BEAT RD + 1x 8_BEAT WR)
8	INCR8	1	1x (1x 8_BEAT RD + 1x 8_BEAT WR) + 1x SINGLE RD + 1x SINGLE WR
9	INCR8	1	1x (1x 8_BEAT RD + 1x 8_BEAT WR) + 2x (1x SINGLE RD + 1x SINGLE WR)
10	INCR8	1	1x (1x 8_BEAT RD + 1x 8_BEAT WR) + 3x (1x SINGLE RD + 1x SINGLE WR)
11	INCR8	1	1x (1x 8_BEAT RD + 1x 8_BEAT WR) + 1x (1x 4_BEAT RD + 1x 4_BEAT WR)
12	INCR8	1	1x (1x 8_BEAT RD + 1x 8_BEAT WR) + 1x (1x 4_BEAT RD + 1x 4_BEAT WR) + 1x SINGLE RD + 1x SINGLE WR





Block Count	Beat Limit	Alternate	Resulting Sequence of AHB Transfers
13	INCR8	1	1x (1x 8_BEAT RD + 1x 8_BEAT WR) + 1x (1x 4_BEAT RD + 1x 4_BEAT WR) + 2x (1x SINGLE RD + 1x SINGLE WR)
14	INCR8	1	1x (1x 8_BEAT RD + 1x 8_BEAT WR) + 1x (1x 4_BEAT RD + 1x 4_BEAT WR) + 3x (1x SINGLE RD + 1x SINGLE WR)
15	INCR8	1	2x (1x 8_BEAT RD + 1x 8_BEAT WR)
0	INCR16	0	1x SINGLE RD + 1x SINGLE WR
1	INCR16	0	2x SINGLE RD + 2x SINGLE WR
2	INCR16	0	3x SINGLE RD + 3x SINGLE WR
3	INCR16	0	1x 4_BEAT RD + 1x 4_BEAT WR
4	INCR16	0	1x 4_BEAT RD + 1x SINGLE RD + 1x 4_BEAT WR + 1x SINGLE WR
5	INCR16	0	1x 4_BEAT RD + 2x SINGLE RD + 1x 4_BEAT WR + 2x SINGLE WR
6	INCR16	0	1x 4_BEAT RD + 3x SINGLE RD + 1x 4_BEAT WR + 3x SINGLE WR
7	INCR16	0	1x 8_BEAT RD + 1x 8_BEAT WR
8	INCR16	0	1x 8_BEAT RD + 1x SINGLE RD + 1x 8_BEAT WR + 1x SINGLE WR
9	INCR16	0	1x 8_BEAT RD + 2x SINGLE RD + 1x 8_BEAT WR + 2x SINGLE WR
10	INCR16	0	1x 8_BEAT RD + 3x SINGLE RD + 1x 8_BEAT WR + 3x SINGLE WR
11	INCR16	0	1x 8_BEAT RD + 1x 4_BEAT RD + 1x 8_BEAT WR + 1x 4_BEAT WR
12	INCR16	0	1x 8_BEAT RD + 1x 4_BEAT RD + 1x SINGLE RD + 1x 8_BEAT WR + 1x 4_BEAT WR + 1x SINGLE WR
13	INCR16	0	1x 8_BEAT RD + 1x 4_BEAT RD + 2x SINGLE RD + 1x 8_BEAT WR + 1x 4_BEAT WR + 2x SINGLE WR
14	INCR16	0	1x 8_BEAT RD + 1x 4_BEAT RD + 3x SINGLE RD + 1x 8_BEAT WR + 1x 4_BEAT WR + 3x SINGLE WR
15	INCR16	0	1x 16_BEAT RD + 1x 16_BEAT WR
0	INCR16	1	1x SINGLE RD + 1x SINGLE WR
1	INCR16	1	2x (1x SINGLE RD + 1x SINGLE WR)
2	INCR16	1	3x (1x SINGLE RD + 1x SINGLE WR)
3	INCR16	1	1x (1x 4_BEAT RD + 1x 4_BEAT WR)
4	INCR16	1	1x (1x 4_BEAT RD + 1x 4_BEAT WR) + 1x SINGLE RD + 1x SINGLE WR
5	INCR16	1	1x (1x 4_BEAT RD + 1x 4_BEAT WR) + 2x (1x SINGLE RD + 1x SINGLE WR)
6	INCR16	1	1x (1x 4_BEAT RD + 1x 4_BEAT WR) + 3x (1x SINGLE RD + 1x SINGLE WR)
7	INCR16	1	1x (1x 8_BEAT RD + 1x 8_BEAT WR)
8	INCR16	1	1x (1x 8_BEAT RD + 1x 8_BEAT WR) + 1x SINGLE RD + 1x SINGLE WR
9	INCR16	1	1x (1x 8_BEAT RD + 1x 8_BEAT WR) + 2x (1x SINGLE RD + 1x SINGLE WR)
10	INCR16	1	1x (1x 8_BEAT RD + 1x 8_BEAT WR) + 3x (1x SINGLE RD + 1x SINGLE WR)
11	INCR16	1	1x (1x 8_BEAT RD + 1x 8_BEAT WR) + 1x (1x 4_BEAT RD + 1x 4_BEAT WR)
12	INCR16	1	1x (1x 8_BEAT RD + 1x 8_BEAT WR) + 1x (1x 4_BEAT RD + 1x 4_BEAT WR) + 1x SINGLE RD + 1x SINGLE WR
13	INCR16	1	1x (1x 8_BEAT RD + 1x 8_BEAT WR) + 1x (1x 4_BEAT RD + 1x 4_BEAT WR) + 2x (1x SINGLE RD + 1x SINGLE WR)
14	INCR16	1	1x (1x 8_BEAT RD + 1x 8_BEAT WR) + 1x (1x 4_BEAT RD + 1x 4_BEAT WR) + 3x (1x SINGLE RD + 1x SINGLE WR)
15	INCR16	1	1x 16_BEAT RD + 1x 16_BEAT WR

**Notes:**

- *n\_BEAT RD can be a "n" beat incremental burst (INCRn) or it can be "n" times a single (SINGLE) data transfer. n\_BEAT RD will be "n" times a SINGLE transfer if one or more of the following conditions is met:*
  - *Fixed Source Address (DMAi\_Dn:FS) is set to "1"*
  - *Decrement Source Address (DMAi\_Dn:DES) is set to "1"*
  - *The 1 KB AHB address boundary will be crossed by the read block transfer*
- *n\_BEAT WR can be a "n" beat incremental burst (INCRn) or it can be "n" times a single (SINGLE) data transfer. n\_BEAT WR will be "n" times a SINGLE transfer if one or more of the following conditions is met:*
  - *Fixed Destination Address (DMAi\_Dn:FD) is set to "1"*
  - *Decrement Destination Address (DMAi\_Dn:DED) is set to "1"*
  - *The 1 KB AHB address boundary will be crossed by the write block transfer.*

After each successful read data transfer Source Address Shadow (DMAi\_SASHDWn:SASHDW) will be either incremented, decremented, or remains unaltered. The behavior is determined by the settings of Fixed Source Address, Decrement Source Address, or Fixed Block Source Address. Destination Address Shadow (DMAi\_DASHDWn:DASHDW) exhibits the same behavior with respect to the settings of Fixed Destination Address, Decrement Destination Address, or Fixed Block Destination Address and will be updated after each successful write data transfer. The tables below list the possible combinations and the resulting action.

**Table 3-2 Source Address Shadow Update Behavior**

DMAi_Dn:FS	DMAi_Dn:DES	DMAi_Dn:FBS	Description of DMAi_SASHDWn:SASHDW Update Behavior
0	0	0	SASHDW is incremented at each successful read data transfer. Size of address increment depends on Transfer Width.
0	1	0	SASHDW is decremented at each successful read data transfer. Size of address decrement depends on Transfer Width.
0	X	1	SASHDW is incremented at each successful read data transfer. SASHDW is updated with the value stored in DMAi_SAn at the end of a block.
1	X	X	SASHDW remains constant.

**Table 3-3 Destination Address Shadow Update Behavior**

DMAi_Dn:FD	DMAi_Dn:DED	DMAi_Dn:FBD	Description of DMAi_DASHDWn:DASHDW Update Behavior
0	0	0	DASHDW is incremented at each successful write data transfer. Size of address increment depends on Transfer Width.
0	1	0	DASHDW is decremented at each successful write data transfer. Size of address decrement depends on Transfer Width.
0	X	1	DASHDW is incremented at each successful write data transfer. DASHDW is updated with the value stored in DMAi_DAn at the end of a block.
1	X	X	DASHDW remains constant.



Figure 3-1 illustrates this behavior exemplarily for Source Address Shadow.

**Figure 3-1 Illustration of DMAi\_SASHDWn:SASHDW Update**

SA = 00002B30												
SA_SHDW	Block 0 (BC=3)				Block 1 (BC=3)				Block 2 (BC=3)			
	1	2	3	4	1	2	3	4	1	2	3	4
	00002B30	00002B31	00002B32	00002B33	00002B34	00002B35	00002B36	00002B37	00002B38	00002B39	00002B3A	00002B3B
FS = 0 DES = 0 FBS = 0 TW = 00												
SA_SHDW	Block 0 (BC=3)				Block 1 (BC=3)				Block 2 (BC=3)			
	1	2	3	4	1	2	3	4	1	2	3	4
	00002B30	00002B2F	00002B2E	00002B2D	00002B2C	00002B2B	00002B2A	00002B29	00002B28	00002B27	00002B26	00002B25
FS = 0 DES = 1 FBS = 0 TW = 00												
SA_SHDW	Block 0 (BC=3)				Block 1 (BC=3)				Block 2 (BC=3)			
	1	2	3	4	1	2	3	4	1	2	3	4
	00002B30	00002B32	00002B34	00002B36	00002B30	00002B32	00002B34	00002B36	00002B30	00002B32	00002B34	00002B36
FS = 0 DES = X FBS = 1 TW = 01												

At the successful end of a DMA transfer Source Address or Destination Address can be updated with the value stored in Source Address Shadow or Destination Address Shadow respectively. This can be configured with the bits Update Source Address (DMAi\_Dn:US) or Update Destination Address (DMAi\_Dn:UD).

### b) DMA Transfer Size

The DMA transfer size is calculated by the following formula:

$$\begin{aligned} \text{DMA transfer size [byte]} &= \text{Number of data transfers} * (2^{\text{Transfer Width}/8-1}) \\ &= (\text{DMAi\_An:BC} + 1) * (\text{DMAi\_An:TC} + 1) * (2^{\text{DMAi\_Bn:TW}}) \end{aligned}$$

Transfer Count (DMAi\_An:TC[15:0]) determines the number of blocks to be transferred in a DMA transfer. Block Count (DMAi\_An:BC[3:0]) determines the number of data transfers in each block.

### c) DMA Transfer Completion and Error Handling

Each DMA channel issues an interrupt at the end of a DMA transfer. This can be either a completion interrupt if the DMA transfer completed successfully or an error interrupt in case of an error condition or a stop request. A completion interrupt is signaled with flag of DIRQ (DMAi\_Bn:DQ) and an error interrupt with flag of EDIRQ (DMAi\_Bn:EQ). There is a DMA transfer end code associated with each interrupt which is encoded in Stop Status (DMAi\_Bn:SS[2:0]).

If a DMA transfer is completed successfully and the completion interrupt raised, Stop Status will show "normal end" (DMAi\_Bn:SS[2:0]="101"). If it is ended in error and the error interrupt raised, Stop Status will show one of the following possibilities:

- Stop request (DMAi\_Bn:SS[2:0]="010")
- Source access error (DMAi\_Bn:SS[2:0]="011")
- Destination access error (DMAi\_Bn:SS[2:0]="100")

A "stop request" during a running DMA transfer can be caused by assertion of the stop request signal (DSTP) of the DMA transfer requesting client, if the DMA channel is disabled (DMAi\_An:EB), if the complete DMA Controller is disabled (DMAi\_R:DE), or if a debug event occurs and the DMA Controller is configured to stop on a debug event.

Both interrupts, completion as well as error interrupt can be masked with bits Completion Interrupt (DMAi\_Bn:CI) and Error Interrupt (DMAi\_Bn:EI) respectively. If these bits are set to "1" the interrupts are not masked. All unmasked completion interrupts are ORed and signalled to the Interrupt Controller. The same is done for the error interrupts.

All completion Interrupt Flags are in addition to the channel registers, available in two 32-bit registers (DMAi\_DIRQ1 and DMAi\_DIRQ2) for easier software handling. All Error Interrupts are handled in the same way and are available in register DMAi\_EDIRQ1 and DMAi\_EDIRQ2.

Completion interrupt DMAi\_Bn:DQ must be cleared by setting Clear DIRQ (DMAi\_Cn:CD). Error interrupt EQ must be cleared by setting Clear EDIRQ (DMAi\_Cn:CE). Stop Status will be cleared to "initial value" (DMAi\_Bn:SS = "000") if DMAi\_Bn:DQ or DMAi\_Bn:EQ is set to "1".

#### **d) Source and Destination Protection**

Each DMA channel has the possibility to define source and destination protection information independently. This information will be used by the AHB master and driven on the AHB protection control signals (HPROTM[3:0]) for use by peripherals which have implemented access protection as defined by the AHB protocol. Protection information must be provided by Source Protection (DMAi\_Bn:SP[3:0]) or Destination Protection (DMAi\_Bn:DP[3:0]) if used. DMA Controller does only data transfers thus DMAi\_Bn:SP[0] and DMAi\_Bn:DP[0] are statically set to "1". Initial value indicates a "Not cacheable/Not bufferable/Privileged access/Data access" source and destination protection.

#### **e) Channel Disabling and Halting**

After reset a DMA channel is disabled by default in order to properly configure it before a DMA request is serviced. DMA channel is enabled by setting Channel Enable (DMAi\_An:EB) to "1". After setting DMAi\_An:EB to "1" the channel waits for a DMA request.

Each DMA channel can be independently disabled. This is done by setting Channel Enable (DMAi\_An:EB) to "0". Setting this bit to "0" can be done at any time but has different effects when it is done. If DMAi\_An:EB is set to "0" during a running DMA transfer, the transfer is stopped at the next transfer gap, an error interrupt is raised and the channel is disabled. Transfer gap means the DMAC has transferred a block of data and the AHB master interface releases the bus request for a few cycles. When DMAi\_An:EB is set to "0" while an interrupt is pending (DMAi\_Bn:DQ="1" or DMAi\_Bn:EQ="1") or no DMA transfer is running there will be no other effect besides the channel is disabled.

Channel halting is done by setting Pause Bit (DMAi\_An:PB) to "1". If this bit is set to "1" during a running DMA transfer, it will halt after completion of the current transferred block. If it is set to "1" before receiving a transfer request the halt state is entered immediately. Clearing this bit will put the channel into run state and it will wait for the next transfer request to continue the DMA transfer or if a transfer request is already pending, it will continue immediately.

#### **f) Burst Transfer Mode**

Burst transfer mode is almost identical to the block transfer mode. The only difference is the request of a DMA transfer. Whereas in block transfer mode one request for each block of data is needed, in burst transfer mode only one request is needed at the begin of a DMA transfer for the complete transfer. The requests needed for subsequent blocks of data is generated internally by the DMA Controller itself.



## 3.2. DMA Client Matrix

This section describes configuration and behavior of the DMA Client Matrix.

### (1) Overview

The DMA Client Matrix provides the possibility to route "M" DMA clients to "N" DMA channels. "M" is greater than or equal to "N". The selection which DMA channel serves which DMA client will be set with Client Interface (DMAi\_CMCHICn:CI). The configuration of the internal DMA clients will be done with the registers DMAi\_CMICICm.

### (2) Modes of Operation

Each DMA Client Interface can work in one of the following modes:

#### 1. Disabled mode

Internal DMA clients:

An internal DMA Client Interface is disabled if it is not selected by any of the DMA Client Matrix Channel Configuration Registers (DMAi\_CMCHICn:CI). Reconfiguration of the internal DMA Client Interface shall only be done in disabled mode.

#### 2. Normal mode

In this mode a DMA channel is routed directly to the specified (DMAi\_CMCHICn:CI) DMA client. The operation of the DMA Client Matrix in this mode is fully transparent and behaves as if the DMA client would be connected directly to the DMA Channel Interface.

### (3) Functional Description

Purpose of the DMA Client Matrix is to provide flexibility in the use of available DMA channels. The configuration of the DMA Client Matrix is intended to be static and shall be done after the boot code execution when the software is setting up the system.

#### a) Structure of the DMA Client Matrix

The DMA Client Matrix will be a full matrix where each DMA Client Interface "m" can be routed to every DMA Channel Interface "n".

#### b) DMA Client Matrix Configuration

The configuration of the DMA Client Matrix is done with the following registers:

- DMAC Client Matrix Internal Client Configuration Registers (DMAi\_CMICICm)
- DMAC Client Matrix Channel Interface Configuration Registers (DMAi\_CMCHICn)

The Config bit Behavior Request Acknowledge, DMAi\_CMICICm:BEHREQACK, sets the behavior of the output signal DREQ\_ACK[m] if the internal DMA Client Interface "m" is not selected in any of the Channel Configuration Registers (DMAi\_CMCHICn). The user can choose that DREQ\_ACK[m] drives inactive level or that DREQ[m] is connected to DREQ\_ACK[m] in that case. The later can be used to reset a, due to software misbehavior, falsely set DMA request signal without violating the two-way handshake protocol.

The Config bit Behavior Stop Acknowledge, DMAi\_CMICICm:BEHSTPACK, sets the behavior of the output signal DSTP\_ACK[m] if the internal DMA Client Interface "m" is not selected in any of the channel Configuration Registers (DMAi\_CMCHICn:CI). The user can choose that DSTP\_ACK[m] drives inactive level or that DSTP[m] is connected to DSTP\_ACK[m] in that case.

The DMAC Client Matrix Channel Interface Configuration Register contains up to nine selection bits. For their function see the description below:

The Selection bits Client Interface, DMAi\_CMCHICn:CI, specify which DMA Client Interface "m" is connected to the DMA Channel Interface "n". The configuration of these bits must take place before DMAi\_R:DE and DMAi\_An:EB is set to "1". The client interface number must be programmed as binary value to DMAi\_CMCHICn:CI. Setting of CI makes the connection between DMA Client Interface defined by the value of CI and DMA Channel Interface "n". Selecting twice or more times the same DMA Client Interface in any of the DMA Client Matrix Channel Configuration Registers results in unpredictable behavior of the DMAC and must be avoided.

Availability of certain DMA clients depends on specific device. See Section "List of Interrupt Factor and DMA Activation Factor" of "CHAPTER: Appendix".

#### (4) Initialization and Application Information

##### Reset

The reset state of each DMA Client Matrix Configuration bit is shown in the register description of DMAi\_CMICICm, and DMAi\_CMCHICn. To summarize it, after hardware reset, all internal DMA Client Interfaces are disabled, all signals set to high-active level, and DMA Channel Interface 0 to "N-1" is configured to route to DMA Client Interface 0 to "N-1".

##### Note:

- *Selecting the same DMA Client Interface in two or more DMA Channel Interfaces leads to unpredictable behavior of the DMAC. Therefore DMAi\_CMCHICn:CI must be properly configured before enabling the DMAC and one or more of its channels.*



### 3.3. DMA Arbiter

This section describes configuration and behavior of the DMA arbiter which chooses a DMA channel based on the arbitration scheme.

#### (1) Overview

The DMA arbiter is responsible for choosing a DMA channel based on the arbitration scheme selected in the Global Configuration Register (DMAi\_R:PR). There are three arbitration schemes available:

- Fixed priority
- Dynamic priority
- Round-robin

The arbitration schemes are explained in detail in the following sections. The arbitration scheme can be changed any time, however it becomes effective only after the current running data transfer has been completed at the next transfer gap.

#### (2) Fixed Priority

In fixed priority arbitration scheme the DMA channels have a "fixed" priority which can be set with Priority Number (DMAi\_Bn:PN). Priority Number equal to "0" has the highest priority, whereas Priority Number equal to 127 has the lowest priority. DMA channels with equal Priority Number, the channel with the lowest channel number "n" has the highest priority. The initial value of DMAi\_Bn:PN is 127. Priority Number can be changed any time, however it becomes effective only after the current running data transfer has been completed at the next transfer gap. Fixed priority arbitration example shows an arbitration example for eight DMA channels to illustrate the behavior.

**Table 3-4 Fixed Priority Arbitration Scheme**

Arbitration Cycle	Requesting DMA Channel "n"	PN of Requesting DMA Channel "n"	Grant Given to DMA Channel "n"
1	2	0	2
	4	2	
	7	6	
2	4	2	4
	7	6	
	8	5	
3	4	2	4
	7	6	
	8	5	
4	1	5	1
	7	6	
	8	5	
5	3	9	8
	7	6	
	8	5	

### (3) Dynamic Priority

The dynamic priority arbitration scheme is an extension of the fixed priority arbitration scheme. The priority of the DMA channels is dynamically adjusted based on the criterion whether a channel got a grant or not. If a channels request was granted its dynamic Priority Number is loaded with the Priority Number stored in DMAi\_Bn:PN and if a channels request was not granted its dynamic Priority Number is decremented by "1". The arbiter is giving grant to the requesting DMA channel with the lowest dynamic Priority Number. If two or more requesting channels have equal dynamic Priority Numbers the DMA channel with the lowest channel number "n" has the highest priority and will win the arbitration process. Priority Number can be changed any time, however it becomes effective after it has been re-loaded by the channel, i.e. after the channel has won the arbitration. Dynamic priority arbitration example shows an arbitration example for four DMA channels to illustrate the behavior.

**Note:**

- In case a channel's re-programmed Priority Number should become effective immediately (not only after the re-programmed channel has won the arbitration), this behavior can be forced by changing the arbitration scheme to Fixed priority and then changing it back to dynamic priority while in halt state. However, it must be noted that the dynamic priority of all channels will be re-loaded with Priority Number (DMAi\_Bn:PN) and not only the re-programmed channels.

**Table 3-5 Dynamic Priority Arbitration Scheme**

Arbitration Cycle	Requesting DMA Channel		Dynamic PN of DMA Channel	PN of DMA Channel	Grant Given to DMA Channel
1	ch.0	Yes	1	1	0
	ch.1	Yes	2	2	
	ch.2	Yes	3	3	
	ch.3	No	3	3	
2	ch.0	No	1	1	1
	ch.1	Yes	1	2	
	ch.2	Yes	1	3	
	ch.3	No	3	3	
3	ch.0	No	1	1	2
	ch.1	No	2	2	
	ch.2	Yes	1	3	
	ch.3	Yes	3	3	
4	ch.0	No	1	1	1
	ch.1	Yes	2	2	
	ch.2	No	3	3	
	ch.3	Yes	2	3	
5	ch.0	No	1	1	3
	ch.1	No	2	2	
	ch.2	Yes	3	3	
	ch.3	Yes	1	3	





#### (4) Round-robin

In round-robin arbitration scheme the turn is rotated in directional and cyclic order from DMA channel 0 to DMA channel "n". At most one DMA channel request can be granted at any time, this is defined as a turn being given. The turn is moved forward at each transfer gap. The turn's rotation is not strictly round-robin in order not to waste an arbitration phase by giving the turn to a non-requesting DMA channel. Instead the turn is given to the next requesting DMA channel in the rotation direction. If no DMA channel was served last (only possible in initial state) the requesting DMA channel with the lowest channel number "n" has the highest priority and will win the arbitration process. Round-robin arbitration example shows an arbitration example for eight DMA channels to illustrate the behavior.

**Table 3-6 Round-robin Priority Arbitration Scheme**

Arbitration Cycle	Requesting DMA Channel	Grant Given to DMA Channel	Last Served DMA Channel
1	2	2	None
	4		
	7		
2	4	4	2
	7		
	8		
3	4	7	4
	7		
	8		
4	1	8	7
	4		
	8		
5	1	1	8
	4		
	7		

#### (5) Application Information

##### a) Fixed Priority Arbitration

With this arbitration scheme the DMA channel request from the channel with the highest priority will be selected for service. If the DMAC is programmed that channel 0 is assigned the highest priority and this channel has a higher service request rate compared to the other channels, it is possible that this channel absorbs the complete bandwidth of the DMA Controller, means that the other channels will not be serviced.

##### b) Dynamic Priority Arbitration

With this arbitration scheme starving is tried to be avoided by assigning a requesting channel which got no grant the next higher priority level. However starving cannot be avoided if the DMAC is not programmed properly i.e. if channel 0 is assigned the highest priority the other channels cannot reach a higher priority than channel 0. If this channel has in addition a higher service request rate than the other channels, it will use the complete bandwidth of the DMAC.

##### c) Round-robin Arbitration

With this arbitration scheme starving of requesting channels is not possible even if channel 0 has a service request rate, which is equal to or exceeds the arbitration rate.

### 3.4. DMA AHB Slave Interface

This section describes information about the slave interface of the DMAC.

This is the DMA Controller's system interface through which the DMACs registers are accessed.

#### (1) Supported Data Transfers

The slave interface supports 8-, 16-, and 32-bit wide AHB data transfers. 16-bit and 32-bit accesses shall be 16-bit address respective 32-bit address aligned.

Single data and fixed incremental burst accesses are supported (SINGLE, INCR4, INCR8, and INCR16).

**Note:**

- All writing to DMA Controller registers must be done in privileged mode.

#### (2) Data Transfer Response

The DMA AHB slave interface responds with the following possibilities to any kind of access:

- OKAY response
- ERROR response

The ERROR response will be given for accesses where a protection error or a register access error occurs.

##### a) Protection Error

A protection error is raised if any of following conditions is met.

- Writing registers in user mode.

##### b) Register Access Error

A register access error is raised if a read or write to a reserved address location is attempted.

A register access error is raised if a write access is attempted to read only registers.



### 3.5. Additional information

This section gives additional information for using DMAC

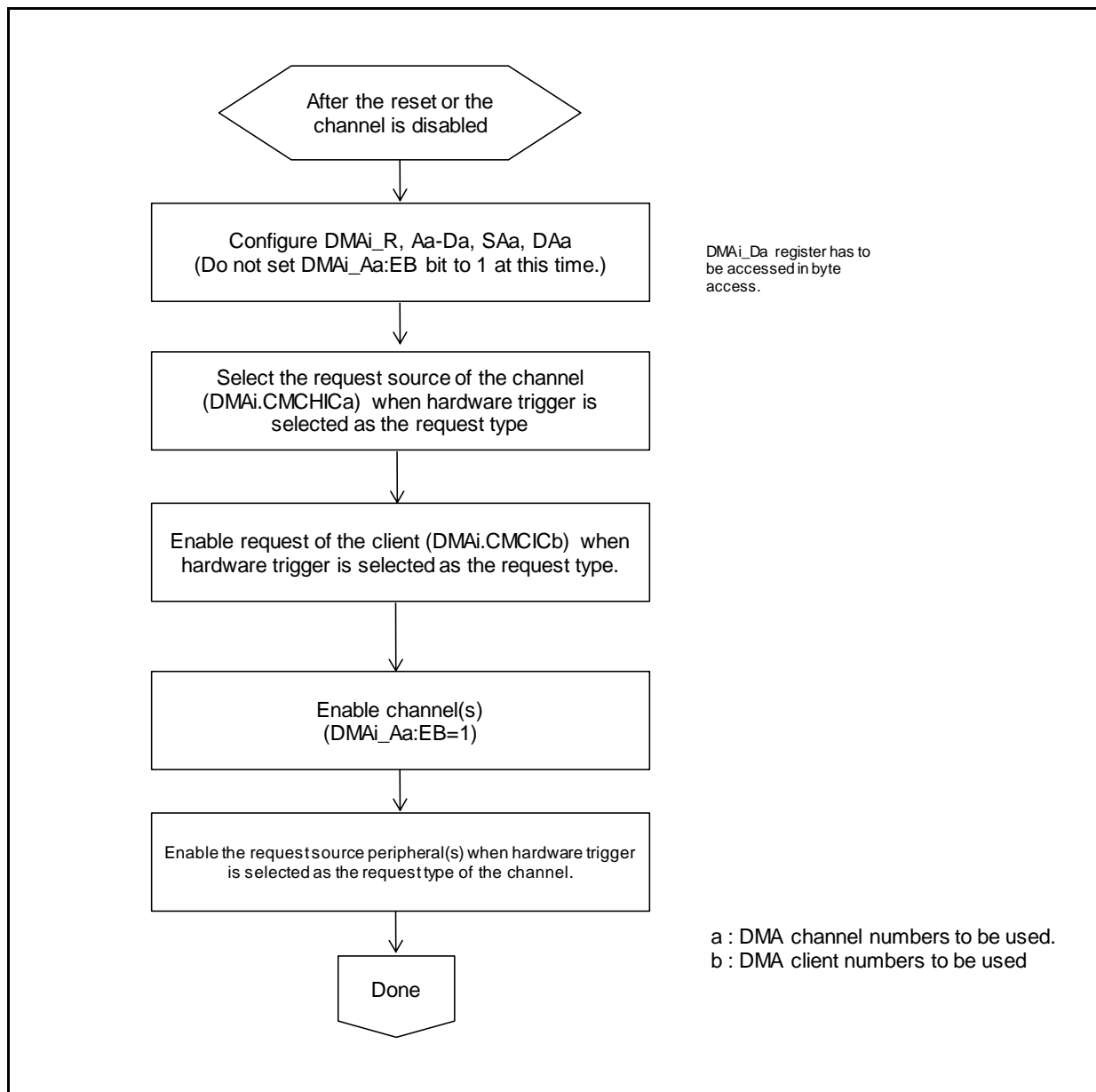
#### (1) Entering PSS(Power Saving State) Mode

- Before entering PSS(Power Saving State) mode, please make sure followings :
  - All DMAC channels are disabled and their operations are completely stopped.
  - No pending or ongoing register access to DMA controller.

#### (2) Operation Flow

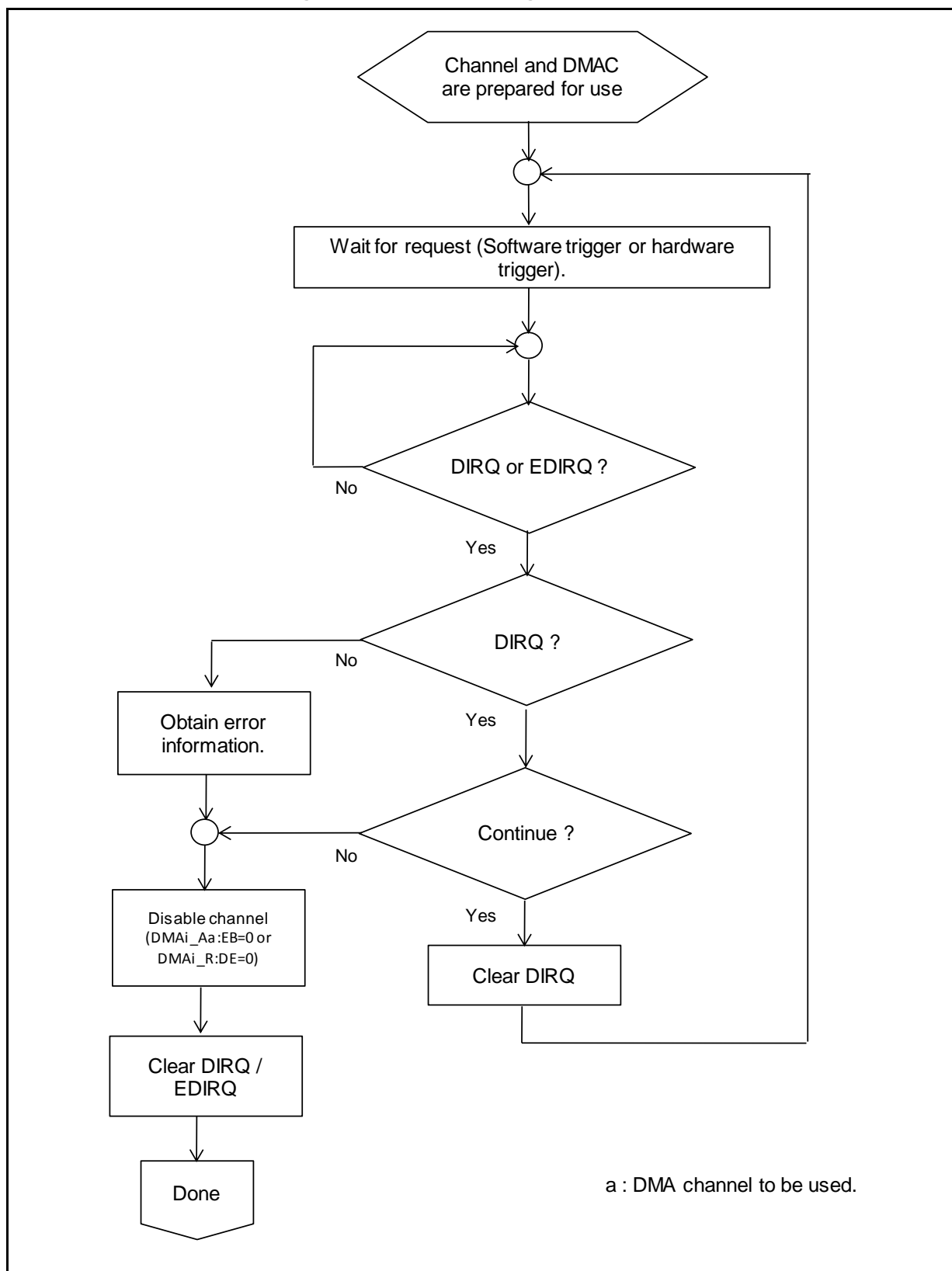
##### a) Configuring and Enabling a DMA Channel

Figure 3-2 Flow for Configuring and Enabling a DMA Channel



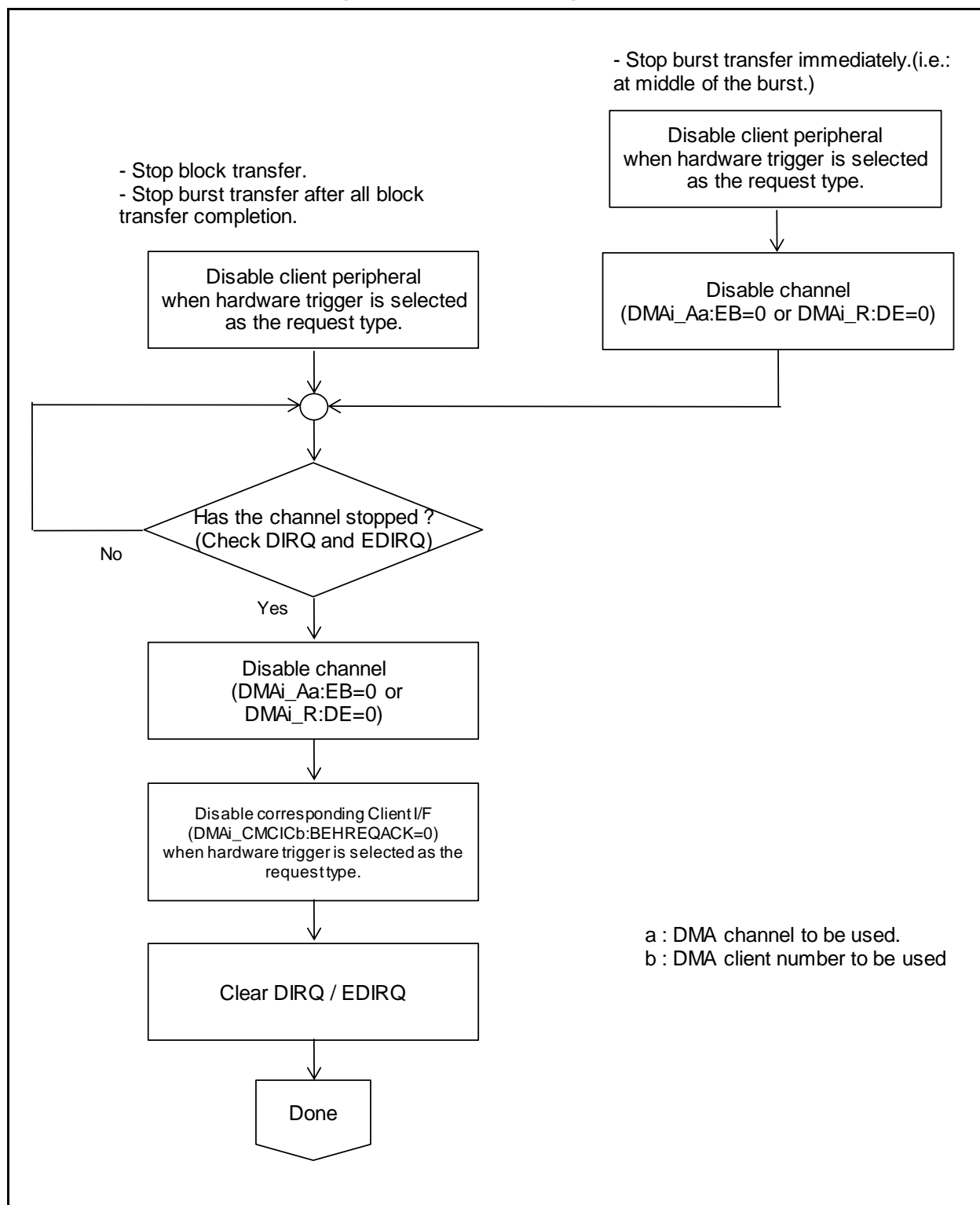
## b) Handling DMAC Requests

Figure 3-3 Flow for Handling DMAC Requests



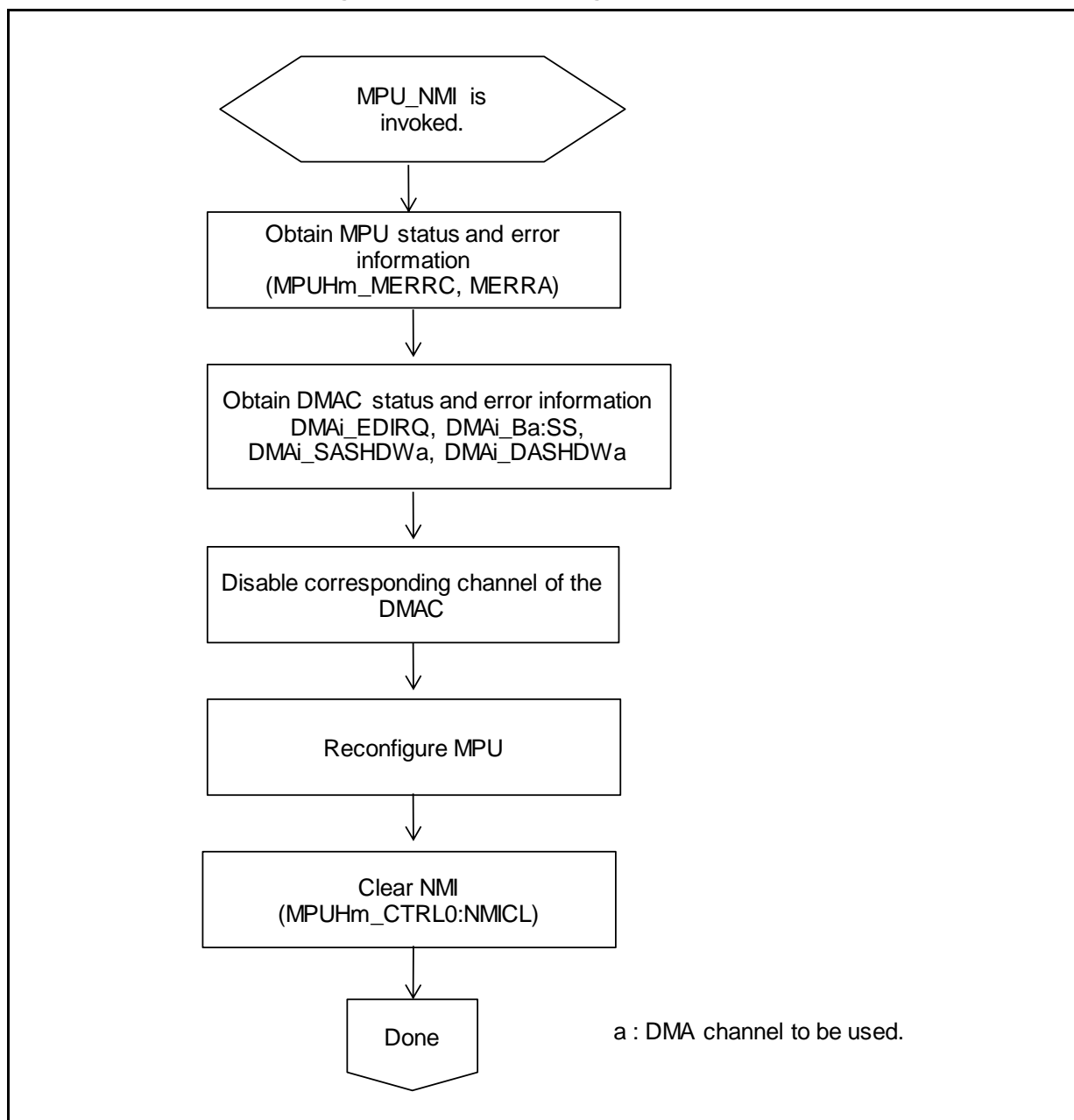
c) Stop Transfer

Figure 3-4 Flow for Stopping Transfer



d) Handling DMAC\_MPU NMI

Figure 3-5 Flow for Handling DMAC\_MPU NMI





## 4. Registers

This section gives the explanation of DMAC.

**Table 4-1 List of DMA Controller Registers**

Abbreviated Register Name	Register Name	See
DMAi_R	DMA Controller Global Configuration Register	4.1
DMAi_DIRQ1	DMA Controller Global Completion Interrupt 1 Register	4.2
DMAi_EDIRQ1	DMA Controller Global Error Interrupt 1 Register	4.3
DMAi_EDIRQ2	DMA Controller Global Error Interrupt 2 Register	4.4
DMAi_DIRQ2	DMA Controller Global Completion Interrupt 2 Register	4.5
DMAi_ID	DMA Controller ID Register	4.6
DMAi_An	DMA Controller Channel Configuration A Register Channel "n"	4.7
DMAi_Bn	DMA Controller Channel Configuration B Register Channel "n"	4.8
DMAi_SAn	DMA Controller Channel Configuration Source Address Register Channel	4.9
DMAi_DAn	DMA Controller Channel Configuration Destination Address Register Channel "n"	4.10
DMAi_Cn	DMA Controller Channel Configuration C Register Channel "n"	4.11
DMAi_Dn	DMA Controller Channel Configuration D Register Channel "n"	4.12
DMAi_En	DMA Controller Channel Configuration E Register Channel "n"	4.13
DMAi_SASHDWn	DMA Controller Channel Configuration Source Address Shadow Register Channel "n"	4.14
DMAi_DASHDWn	DMA Controller Channel Configuration Destination Address Shadow Register Channel "n"	4.15
DMAi_CMICm	DMA Controller Client Matrix Internal Client Interface Configuration Register "m"	4.16
DMAi_CMCHICn	DMA Controller Client Matrix Channel Interface Configuration Register "n"	4.17

"i" represents instance number of DMAC. i = 0 in this product.

"n" represents channel number. n = 0, 1, 2, ...,15.

"N" represents number of DMAC channels per DMAC instance. N = 16 in this product.

"m" represents client number. m = 8, 9, ...,142.

"M" represents number of client I/F. M = 143 in this product.

## 4.1. DMA Controller Global Configuration Register (DMAi\_R)

This register handles the enabling and halting of the entire DMA Controller, arbitration scheme of the DMA channel arbiter can be chosen, enabling of debug function, and the DMACs behavior in case of a "debug event" can be selected.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	DE	DSHR	DBE	PR[1]	PR[0]	DH	DB[1]	DB[0]
ACCESS_TYPE	R/W	R,WX	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	WP							
INITIAL_VALUE	0	1	0	0	0	0	0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	DSHS
ACCESS_TYPE	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R,WX
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	1

### [bit31] DE: DMA Enable(DE)

If this bit is set to "0", the DMAC is disabled. This also means that all DMA channels are disabled independent of the settings of DMAi\_An:EB.

If this bit is set to "1", the DMAC is enabled and the enabling of the channels depend on the setting of DMAi\_An:EB.

When this bit is set to "0" during a DMA transfer, the channel which is in the middle of a transfer stops at the next transfer gap.

The transfer gap means that DMAC de-asserts bus request to the bus arbiter for a short time in the middle of a DMA transfer after a block of data has been transferred. This ensures that the bus is not completely blocked by a very long DMA transfer.

Value	Description
0	DMAC globally disabled
1	DMAC globally enabled



**[bit30] DSHR: DMA Stop/Halt Request Flag**

This bit indicates that the DMA transfers of all channels have been requested to halt or disable.

This bit is set to "0" by hardware if none of the conditions below are true.

This bit is set to "1" by hardware if one or more of the conditions below are true.

Conditions for DMA Stop/Halt Request Flag:

- DMAi\_R:DE is set to "0" (all channels are disabled)
- DMAi\_R:DH is set to "1" (all channels are halted)
- DMAi\_R:DBE is set to "1", DMAi\_R:DB is set to "10" (stop on debug events), and a debug event is pending
- DMAi\_R:DBE is set to "1", DMAi\_R:DB is set to "01" (halt on debug events), and a debug event is pending

Value	Description
0	Indicates that global halt/disable condition of DMAC is removed
1	Indicates that DMA transfers of all channels are requested to halt or disable

**[bit29] DBE: Debug Enable**

This bit determines whether DMAC reacts on a debug event (i.e. debugger break point).

The behavior of the DMAC on the occurrence of a debug event depends on configuration bits, Debug Behavior (DMAi\_R:DB).

Value	Description
0	DMAC does not react on debug events
1	DMAC reacts on debug events. Reaction depends on setting of Debug Behavior (DMAi_R:DB)

**Note:**

- The debug event cannot be used with this product.

**[bit28:27] PR[1:0]: Priority Type**

These bits select the arbitration scheme of the DMAC arbiter. In case of dynamic priority, channel priority is updated at the transfer gap.

Value	Description
00	Fixed priority
01	Dynamic priority
10	Round robin
11	Reserved

**[bit26] DH: DMA Halt**

When this bit is set to "1", all DMA channels are halted and do not perform DMA transfers until this bit is set back to "0". After it is cleared the halted DMA transfers continue at the point they were halted.

If this bit is set to "1" while a DMA transfer is ongoing, DMAC halts the transfer at the next transfer gap.

About the transfer gap, refer to the description of DMAi\_R:DE bit.

**[bit25:24] DB[1:0]: Debug Behavior**

Value	Description
00	DMAC continues on debug event
01	DMAC halts all transfers on debug event
10	DMAC stops all transfers on debug event
11	Reserved

**Note:**

- The debug event cannot be used with this product.

**[bit23:1] Reserved: Reserved bits**

**[bit0] DSHS: DMA Stop/Halt Status Flag**

Value	Description
0	Indicate that DMA transfer of at least one channel is running
1	Indicate that DMA transfers of all channels are halted or disabled



## 4.2. DMA Controller Global Completion Interrupt 1 Register (DMAi\_DIRQ1)

The DMA Controller Global Completion Interrupt 1 Register combines the completion Interrupt Flags (DMAi\_Bn:DQ) from DMA channels 0 to 31.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	DIRQ[31]	DIRQ[30]	DIRQ[29]	DIRQ[28]	DIRQ[27]	DIRQ[26]	DIRQ[25]	DIRQ[24]
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	DIRQ[23]	DIRQ[22]	DIRQ[21]	DIRQ[20]	DIRQ[19]	DIRQ[18]	DIRQ[17]	DIRQ[16]
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	DIRQ[15]	DIRQ[14]	DIRQ[13]	DIRQ[12]	DIRQ[11]	DIRQ[10]	DIRQ[9]	DIRQ[8]
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	DIRQ[7]	DIRQ[6]	DIRQ[5]	DIRQ[4]	DIRQ[3]	DIRQ[2]	DIRQ[1]	DIRQ[0]
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

### [bit31:0] DIRQ: Global Completion Interrupt 1

This is a read-only register which gives the status of DIRQ of channels 0 to 31. Channels which are not available on a particular device reads "0".

### 4.3. DMA Controller Global Error Interrupt 1 Register (DMAi\_EDIRQ1)

The DMA Controller Global Error Interrupt 1 Register combines the Error Interrupt Flags (DMAi\_Bn:EQ) from DMA channels 0 to 31.

BITS_OFFSET	31	30	29	28	27	26	25	24
BITS_NAME	EDIRQ[31]	EDIRQ[30]	EDIRQ[29]	EDIRQ[28]	EDIRQ[27]	EDIRQ[26]	EDIRQ[25]	EDIRQ[24]
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BITS_OFFSET	23	22	21	20	19	18	17	16
BITS_NAME	EDIRQ[23]	EDIRQ[22]	EDIRQ[21]	EDIRQ[20]	EDIRQ[19]	EDIRQ[18]	EDIRQ[17]	EDIRQ[16]
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BITS_OFFSET	15	14	13	12	11	10	9	8
BITS_NAME	EDIRQ[15]	EDIRQ[14]	EDIRQ[13]	EDIRQ[12]	EDIRQ[11]	EDIRQ[10]	EDIRQ[9]	EDIRQ[8]
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BITS_OFFSET	7	6	5	4	3	2	1	0
BITS_NAME	EDIRQ[7]	EDIRQ[6]	EDIRQ[5]	EDIRQ[4]	EDIRQ[3]	EDIRQ[2]	EDIRQ[1]	EDIRQ[0]
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

#### [bit31:0] EDIRQ: Global Error Interrupt 1

This is a read-only register which gives the status of EDIRQ of channels 0 to 31. Channels which are not available on a particular device reads "0".



#### 4.4. DMA Controller Global Error Interrupt 2 Register (DMAi\_EDIRQ2)

The DMA Controller Global Error Interrupt 2 Register combines the Error Interrupt Flags (DMAi\_Bn:EQ) from DMA channels 32 to 63.

BITS_OFFSET	31	30	29	28	27	26	25	24
BITS_NAME	EDIRQ[63]	EDIRQ[62]	EDIRQ[61]	EDIRQ[60]	EDIRQ[59]	EDIRQ[58]	EDIRQ[57]	EDIRQ[56]
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BITS_OFFSET	23	22	21	20	19	18	17	16
BITS_NAME	EDIRQ[55]	EDIRQ[54]	EDIRQ[53]	EDIRQ[52]	EDIRQ[51]	EDIRQ[50]	EDIRQ[49]	EDIRQ[48]
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BITS_OFFSET	15	14	13	12	11	10	9	8
BITS_NAME	EDIRQ[47]	EDIRQ[46]	EDIRQ[45]	EDIRQ[44]	EDIRQ[43]	EDIRQ[42]	EDIRQ[41]	EDIRQ[40]
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BITS_OFFSET	7	6	5	4	3	2	1	0
BITS_NAME	EDIRQ[39]	EDIRQ[38]	EDIRQ[37]	EDIRQ[36]	EDIRQ[35]	EDIRQ[34]	EDIRQ[33]	EDIRQ[32]
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

##### [bit31:0] EDIRQ: Global Error Interrupt 2

This is a read-only register which gives the status of EDIRQ of channels 32 to 63. Channels which are not available on a particular device reads "0".

## 4.5. DMA Controller Global Completion Interrupt 2 Register (DMAi\_DIRQ2)

The DMA Controller Global Completion Interrupt 2 Register combines the completion Interrupt Flags (DMAi\_Bn:DQ) from DMA channels 32 to 63

BITS_OFFSET	31	30	29	28	27	26	25	24
BITS_NAME	DIRQ[63]	DIRQ[62]	DIRQ[61]	DIRQ[60]	DIRQ[59]	DIRQ[58]	DIRQ[57]	DIRQ[56]
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BITS_OFFSET	23	22	21	20	19	18	17	16
BITS_NAME	DIRQ[55]	DIRQ[54]	DIRQ[53]	DIRQ[52]	DIRQ[51]	DIRQ[50]	DIRQ[49]	DIRQ[48]
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BITS_OFFSET	15	14	13	12	11	10	9	8
BITS_NAME	DIRQ[47]	DIRQ[46]	DIRQ[45]	DIRQ[44]	DIRQ[43]	DIRQ[42]	DIRQ[41]	DIRQ[40]
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BITS_OFFSET	7	6	5	4	3	2	1	0
BITS_NAME	DIRQ[39]	DIRQ[38]	DIRQ[37]	DIRQ[36]	DIRQ[35]	DIRQ[34]	DIRQ[33]	DIRQ[32]
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

### [bit31:0] DIRQ: Global Completion Interrupt 2

This is a read-only register which gives the status of DIRQ of channels 32 to 63. Channels which are not available on a particular device reads "0".



## 4.6. DMA Controller ID Register (DMAi\_ID)

The DMA Controller ID Register always returns a constant. This register offers no function.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	MID[31]	MID[30]	MID[29]	MID[28]	MID[27]	MID[26]	MID[25]	MID[24]
ACCESS_TYPE	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	MID[23]	MID[22]	MID[21]	MID[20]	MID[19]	MID[18]	MID[17]	MID[16]
ACCESS_TYPE	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R1,WX
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	1

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	MID[15]	MID[14]	MID[13]	MID[12]	MID[11]	MID[10]	MID[9]	MID[8]
ACCESS_TYPE	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R1,WX	R0,WX	R1,WX
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	1	0	1

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	MID[7]	MID[6]	MID[5]	MID[4]	MID[3]	MID[2]	MID[1]	MID[0]
ACCESS_TYPE	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

### [bit31:0] MID: Module Number

Always return 0x00010500.

## 4.7. DMA Controller Channel Configuration A Register Channel "n" (DMAi\_An)

The register specifies whether to enable or halt a DMA channel. It specifies the source of a DMA transfer request and DMA transfer parameters Block Count, Transfer Count, Beat Limit, Alternate, and Timeout.

BITS_OFFSET	31	30	29	28	27	26	25	24
BITS_NAME	EB	PB	ST	IS[1]	IS[0]	AL	BL[1]	BL[0]
ACCESS_TYPE	R/W	R/W	R0,W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BITS_OFFSET	23	22	21	20	19	18	17	16
BITS_NAME	BC[3]	BC[2]	BC[1]	BC[0]	TO[3]	TO[2]	TO[1]	TO[0]
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W1	R/W1	R/W1	R/W1
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	1	1	1	1

BITS_OFFSET	15	14	13	12	11	10	9	8
BITS_NAME	TC[15]	TC[14]	TC[13]	TC[12]	TC[11]	TC[10]	TC[9]	TC[8]
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BITS_OFFSET	7	6	5	4	3	2	1	0
BITS_NAME	TC[7]	TC[6]	TC[5]	TC[4]	TC[3]	TC[2]	TC[1]	TC[0]
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

### [bit31] EB: Enable Bit

This bit is used to enable/disable a DMA channel. If this bit is set to "1", the channel is enabled and waits for a request to start a DMA transfer (Before that, DMAi\_R:DE bit needs to be set to "1" already). If this bit is set to "0", the channel is disabled and does not perform a DMA transfer. When this bit is set to "0" during a running DMA transfer which will not complete at the next transfer gap, the DMA transfer is terminated. This is regarded as a forced stop and an error interrupt is generated. When this bit is set to "0" while the last block of a DMA transfer is running, the DMA transfer completes at the next transfer gap and a completion interrupt is generated. About the transfer gap, refer to the description of DMAi\_R:DE bit. This bit is useful to re-configure each configuration register of the channel after a DMA transfer.

Value	Description
0	Channel is disabled
1	Channel is enabled





**[bit30] PB: Pause Bit**

This bit is used to halt the transfer of the DMA channel. If this bit is set to "1", this channel halts the transfer and does not perform a DMA transfer until this bit is cleared.

When this bit is set to "1" while no transfer is ongoing DMAC enters the halt state immediately. When it was set to "1" during a running transfer, the halt state is entered at the next transfer gap. If the DMA transfer completes at the next transfer gap a completion interrupt is issued.

When this bit is set to "0" the halt condition is cleared and DMAC waits for the next request to continue the DMA transfer.

This bit is useful to halt a DMA transfer without re-configuration of each configuration register of the channel.

Value	Description
0	Channel is not halted
1	Channel is halted

**[bit29] ST: Software Trigger**

This bit is used to generate a software request. This bit can only be used when all of following conditions are satisfied.

- DMAi\_Bn:SR = "1"
- The stop status (DMAi\_Bn:SS) is either initial value or normal end
- DIRQ is cleared.
- EDIRQ is cleared.
- In Block transfer mode, the previous transfer has been completed.

When this bit is set to "1", DMA transfer is requested because software request has been received. The DMAC sets this bit to "0" if the Software Trigger has been recognized, an internal channel request for service is set, and Software Trigger Ready (DMAi\_Bn:SR) is set to "0". The software request is successful when DMAi\_Bn:SR changes its status from "1" to "0". ST can only be read as "0".

Value	Description
0	No software request
1	Software request

**[bit28:27] IS[1:0]: Input Select**

These bits are used to select the trigger source of a DMA transfer request. When the trigger source of a DMA transfer is a software request, IS bits are set to "00". When the trigger source of a DMA transfer is a hardware request, IS bits are set to "01".

Value	Description
00	Software request
01	Hardware request
10	Reserved
11	Reserved

**[bit26] AL: Alternate**

This bit decides whether the data transfers should alternate between read and write or should be contiguous reads followed by contiguous writes. The alternation takes place after each incremental read burst and after each single data read.

Value	Description
0	Contiguous
1	Alternate

**[bit25:24] BL[1:0]: Beat Limit**

Beat Limit controls the maximum burst length the AHB master can make on the AHB bus for this DMA channel.

Value	Description
00	Single transfer (SINGLE)
01	4-beat incrementing burst (INCR4)
10	8-bit incrementing burst (INCR8)
11	16-bit incrementing burst (INCR16)

These bits are used in combination with Block Count to decide which type of burst has to be transmitted over the AHB interface.

**[bit23:20] BC[3:0]: Block Count**

These bits specify the total length of a single block in block/burst transfer mode. The maximum block count is 16. For e.g. if BC = 4, the number of data transfers is 5 (BC + 1).

Alternate, Block Count, and Beat Limit together decide the bursts generated by the AHB master.

**[bit19:16] TO[3:0]: Timeout**

The bits are reserved in this product. Write 0b1111 to the bits.

**[bit15:0] TC[15:0]: Transfer Count**

These bits are used to specify the transfer count for the block/burst transfer. The maximum transfer count is 65536. If TC is set to zero then one transfer is done and if TC is set to 65535 then 65536 transfers are done.

In burst and block transfer mode the TC represents the number of blocks of data transfers that the channel has to make before DMA "End of Processing" (DEOP) is generated. For e.g. if TC = 9 and DMAi\_An:BC = 9 then total number of transfers the DMAC does = (DMAi\_An:BC + 1) x (TC + 1) = 10 x 10 = 100 transfers before DEOP is generated.



#### 4.8. DMA Controller Channel Configuration B Register Channel "n" (DMAi\_Bn)

This register contains Error and Completion Interrupt Flags, and Interrupt Mask bits. Stop status of DMA operation is available in this register. Operation mode selection, transfer data width, source, and destination protection information are located here. Software trigger ready flag shows readiness of DMA channel to receive a Software Trigger. Priority Number for DMA channel arbiter can be set here for fixed and dynamic priority arbitration scheme.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	DQ	EQ	MS[1]	MS[0]	TW[1]	TW[0]	SR	Reserved
ACCESS_TYPE	R,WX	R,WX	R0,W	R/W	R/W	R/W	R,WX	R,WX
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved	Reserved	Reserved	EI	CI	SS[2]	SS[1]	SS[0]
ACCESS_TYPE	R0,WX	R0,WX	R0,WX	R/W	R/W	R,WX	R,WX	R,WX
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved	PN[6]	PN[5]	PN[4]	PN[3]	PN[2]	PN[1]	PN[0]
ACCESS_TYPE	R0,WX	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	WP							
INITIAL_VALUE	0	1	1	1	1	1	1	1

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	TC[7]	TC[6]	TC[5]	TC[4]	TC[3]	TC[2]	TC[1]	TC[0]
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

##### [bit31] DQ: Flag of DIRQ

DQ is set to "1" when a DMA transfer completed successfully. DQ is cleared by hardware if DMAi\_Cn:CD (Clear DIRQ) bit is set to "1". Otherwise the value is retained.

##### [bit30] EQ: Flag of EDIRQ

EQ is set to "1" when a DMA transfer is finished with an error. EQ is cleared by hardware if DMAi\_Cn:CE (Clear EDIRQ) bit is set to "1". Otherwise the value is retained.

**[bit29:28] MS[1:0]: Mode select**

MS sets the transfer mode of the channel.

Value	Description
00	Block transfer mode
01	Burst transfer mode
10	Reserved
11	Reserved

**[bit27:26] TW[1:0]: Transfer data width**

TW specifies the data width for every data transfer of the DMA transfer.

Value	Description
00	Byte
01	Half word
10	Word
11	Double word

**[bit25] SR: Software Trigger Ready**

This bit is used to indicate a condition\* that the DMA channel is ready to receive a software request. The following conditions can cause that the DMA channel is not ready to receive a software request. If one or more of the conditions below are true SR is set to "0" by hardware. If none of the conditions are true SR is set to "1" by hardware.

- DMAi\_R:DE == "0"; DMAC is disabled
- DMAi\_R:DH == "1"; All DMA channels are halted
- DMAi\_An:EB == "0"; DMA channel is disabled
- DMAi\_An:PB == "1"; DMA channel is halted
- DMAi\_An:IS != "00"; Input select is not set to software trigger
- DEBUG == "1" and DMAi\_R:DF == "1" and DMAi\_R:DB == "01" or DMAi\_R:DB == "10", debug event is pending while debug flag is set and Debug Behavior is set to HALT or STOP
- DMA transfer is active and a software request is pending

\*: See the explanation of DMAi\_An:ST bit in Section "4.7. DMA Controller Channel Configuration A Register Channel "n" (DMAi\_An)" for the rest of the conditions.

**[bit20] EI: Error Interrupt Enable**

This bit is used to control the issue of an error interrupt (EDIRQ). If this bit is set to "1", an error interrupt is issued due to any of the following transfer errors:

- Transfer stop request by signal DSTP, or disable the transfer with DMAi\_An:EB or DMAi\_R:DE, or debug event (if DMAi\_R:DBE="1" and DMAi\_R:DB[1:0]="10")
- Source access error
- Destination access error

Value	Description
0	Error interrupt issuance is disabled
1	Error interrupt issuance is enabled

**[bit19] CI: Completion interrupt enable**

This bit is used to control the issue of a completion interrupt (DIRQ). If this bit is set to "1", a completion interrupt is issued after the DMA transfer completed normally.

Value	Description
0	Completion interrupt issuance is disabled
1	Completion interrupt issuance is enabled

**[bit18:16] SS[2:0]: Stop status**

These bits are used to show the end code of DMA transfer. SS is set by hardware when an error or completion interrupt is raised and it is cleared by hardware when either DMAi\_Cn:CE or DMAi\_Cn:CD is set to "1".

Value	Description
000	Initial value. Status : None
001	Reserved
010	Stop request by: - DSTP - Channel disable (DMAi_An:EB="0") - DMA disable (DMAi_R:DE="0") - Debug event (DMAi_R:DBE="1" and DMAi_R:DB="10") Status : Stop
011	Source access error Status : Error
100	Destination access error Status : Error
101	Normal end Status : End
110	Reserved
111	Reserved

When different events occur at the same time, the end code is displayed according to the following priority:

1. Reset
2. Cleared by clearing completion/error interrupt (DIRQ/EDIRQ)
3. Source access error
4. Destination access error
5. Stop request

**Note:**

- If the interrupt bit is cleared then Stop Status is also cleared.

**[bit15:12] SP[3:0]: Source protection**

These bits are used to control source access protection. During source accesses HPROTM is driven with SP during the AHB address phase.

- SP[3]: "0": Not cacheable
- SP[3]: "1": Cacheable
- SP[2]: "0": Not bufferable
- SP[2]: "1": Bufferable
- SP[1]: "0": User access
- SP[1]: "1": Privileged access
- SP[0]: "0": Instruction access (DMAC only makes data accesses thus SP[0] is fixed to "1" and writing it to "0" has no effect.)
- SP[0]: "1": Data access

SP is considered by sources which support protection control function.

**[bit11:8] DP[3:0]: Destination protection**

These bits are used to control destination access protection. During destination accesses HPROTM is driven with DP during the AHB address phase.

- DP[3]: "0": Not cacheable
- DP[3]: "1": Cacheable
- DP[2]: "0": Not bufferable
- DP[2]: "1": Bufferable
- DP[1]: "0": User access
- DP[1]: "1": Privileged access
- DP[0]: "0": Instruction access (DMAC only makes data accesses thus DP[0] is fixed to "1" and writing it to "0" has no effect.)
- DP[0]: "1": Data access

DP is only considered by destinations which support protection control function.

**[bit6:0] PN[6:0]: Priority number**

These bits are used to specify the Priority Number. The Priority Number is needed if fixed priority or dynamic priority has been selected as arbitration scheme. It has no significance if round robin arbitration scheme is selected. The channel with lower priority number value has a higher priority.



## 4.9. DMA Controller Channel Configuration Source Address Register Channel n (DMAi\_SAn)

This register holds the source address for the DMA transfer of channel "n". There are the same N register (0 to N-1).

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	SA[31]	SA[30]	SA[29]	SA[28]	SA[27]	SA[26]	SA[25]	SA[24]
ACCESS_TYPE	R,W	R,W	R,W	R,W	R,W	R,W	R,W	R,W
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	SA[23]	SA[22]	SA[21]	SA[20]	SA[19]	SA[18]	SA[17]	SA[16]
ACCESS_TYPE	R,W	R,W	R,W	R,W	R,W	R,W	R,W	R,W
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	SA[15]	SA[14]	SA[13]	SA[12]	SA[11]	SA[10]	SA[9]	SA[8]
ACCESS_TYPE	R,W	R,W	R,W	R,W	R,W	R,W	R,W	R,W
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	SA[7]	SA[6]	SA[5]	SA[4]	SA[3]	SA[2]	SA[1]	SA[0]
ACCESS_TYPE	R,W	R,W	R,W	R,W	R,W	R,W	R,W	R,W
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

### [bit31:0] SA[31:0]: Source address

These bits are used to specify the source address to start a DMA transfer. Source Address is updated with the value of DMAi\_SASHDWn at the end of a DMA transfer if Update Source Address (DMAi\_Dn:US) is set to "1", Fixed Block Source Address (DMAi\_Dn:FBS) is set to "0", and the transfer ends successfully. Otherwise it retains its value.

#### Notes:

- DMAi\_SAn register must be loaded with aligned addresses with respect to the transfer data width. If non-aligned addresses are loaded the DMAC converts it to an aligned address according to the setting of Transfer data Width (DMAi\_Bn:TW).
- Disable the channel (i.e. DMAi\_R:DE=0 or DMAiAn:EB = 0) before configuring this register.

## 4.10. DMA Controller Channel Configuration Destination Address Register Channel n (DMAi\_DAn)

This register holds the destination address for the DMA transfer of channel "n". There are the same N register (0 to N-1).

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	DA[31]	DA[30]	DA[29]	DA[28]	DA[27]	DA[26]	DA[25]	DA[24]
ACCESS_TYPE	R,W	R,W	R,W	R,W	R,W	R,W	R,W	R,W
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	DA[23]	DA[22]	DA[21]	DA[20]	DA[19]	DA[18]	DA[17]	DA[16]
ACCESS_TYPE	R,W	R,W	R,W	R,W	R,W	R,W	R,W	R,W
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	DA[15]	DA[14]	DA[13]	DA[12]	DA[11]	DA[10]	DA[9]	DA[8]
ACCESS_TYPE	R,W	R,W	R,W	R,W	R,W	R,W	R,W	R,W
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	DA[7]	DA[6]	DA[5]	DA[4]	DA[3]	DA[2]	DA[1]	DA[0]
ACCESS_TYPE	R,W	R,W	R,W	R,W	R,W	R,W	R,W	R,W
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

### [bit31:0] DA[31:0]: Destination Address

These bits are used to specify the destination address to start a DMA transfer. Destination Address is updated with the value of DMAi\_DASHDWn at the end of a DMA transfer if Update Destination Address (DMAi\_Dn:UD) is set to "1", Fixed Block Destination Address (DMAi\_Dn:FBD) is set to "0", and the transfer ends successfully. Otherwise it retains its value.

#### Notes:

- DMAi\_DAn register must be loaded with aligned addresses with respect to the transfer data width. If non-aligned addresses are loaded the DMAC converts it to an aligned address according to the setting of Transfer data Width (DMAi\_Bn:TW).
- Disable the channel (i.e. DMAi\_R:DE=0 or DMAiAn:EB = 0) before configuring this register.





#### 4.11. DMA Controller Channel Configuration C Register Channel n (DMAi\_Cn)

The DMA Controller Channel Configuration C Register is used to clear the Interrupt Flags set in the DMAi\_Bn register. By writing a "1" to a bit in this register, the software can clear the corresponding flag in the DMAi\_Bn register. There are the same N register (0 to N-1).

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	CE
ACCESS_TYPE	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,W
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	CD
ACCESS_TYPE	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,W
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

**[bit31:9] Reserved: Reserved bits**

**[bit8] CE: Clear EDIRQ**

Setting this bit clears the EDIRQ flag bit (DMAi\_Bn:EQ). DMAC clears this bit automatically.

**[bit7:1] Reserved: Reserved bits**

**[bit0] CD: Clear DIRQ**

Setting this bit clears the DIRQ flag bit (DMAi\_Bn:DQ). DMAC clears this bit automatically.

## 4.12. DMA Controller Channel Configuration D Register Channel n (DMAi\_Dn)

This register controls the DMA channels addressing behavior for a DMA transfer. Addressing can be independently chosen for source and destination.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	FS	DES	US	FBS	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R/W	R/W	R/W	R/W	R0,WX	R0,WX	R0,WX	R0,WX
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	FD	DED	UD	FBD	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R/W	R/W	R/W	R/W	R0,WX	R0,WX	R0,WX	R0,WX
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

**Note:**

- This register must be accessed by 8-bit access to the effective bits.

### [bit31] FS: Fixed source address

This bit is used to keep the source address at a fix value.

Value	Description
0	Source address is incremented
1	Source address is kept fix. AHB master only makes single transfers to access the source



**[bit30] DES: Decrement source address**

If this bit is set the source address on the AHB interface is decremented on each AHB transfer. In this mode the AHB master makes only single transfers to access the source.

Value	Description
0	Source address is incremented
1	Source address is decremented

**Note:**

- Fixed Source Address (DMAi\_Dn:FS) has higher priority than Decrement Source Address. This bit is only effective if DMAi\_Dn:FS = "0" and DMAi\_Dn:FBS = "0".

**[bit29] US: Update source address**

Value	Description
0	DMAi_SAn is not updated after DMA transfer is successfully completed. DMAi_SAn is retained
1	DMAi_SAn is updated with the next address after the DMA transfer is successfully completed. E.g. last source address is 0x0BF000FC, DMAi_Bn:TW = 10 (word) then DMAi_SAn is updated with address 0x0BF00100.

**Note:**

- This bit is only effective if DMAi\_Dn:FBS = "0".

**[bit28] FBS: Fixed Block Source Address**

Value	Description
0	Start address of first block of DMA transfer is set to the value stored in DMAi_SAn. Start address of consecutive blocks is the address following the previous block last address (according to transfer data width (DMAi_Bn:TW))
1	Start address of each block is set to the value stored in DMAi_SAn

**Note:**

- Setting of FBS is only effective if DMAi\_Dn:FS = "0" and (DMAi\_Bn:MS = "00" (block transfer mode) or DMAi\_Bn:MS = "01" (burst transfer mode)).

**[bit15] FD: Fixed destination address**

This bit is used to keep the destination address at a fix value.

Value	Description
0	Destination address is incremented
1	Destination address is kept fixed. AHB master only makes single transfers to access the destination

**[bit14] DED: Decrement destination address**

If this bit is set the destination address on the AHB interface is decremented on each AHB transfer. In this mode the AHB master makes only single transfers to access the destination.

Value	Description
0	Destination address is incremented
1	Destination address is decremented

**Note:**

- Fixed Destination Address (DMAi\_Dn:FD) has higher priority than Decrement Destination Address. This bit is only effective if DMAi\_Dn:FD = "0" and DMAi\_Dn:FBD = "0".

**[bit13] UD: Update destination address**

Value	Description
0	DMAi_DAn is not updated after DMA transfer is successfully completed. DMAi_DAn is retained
1	DMAi_DAn is updated with the next address after the DMA transfer is successfully completed. E.g. last destination address was 0x0BF40010, DMAi_Bn:TW = 10 (word) then DMAi_DAn is updated with address 0x0BF40014.

**Note:**

- This bit is effective only if DMAi\_Dn:FBD = "0".

**[bit12] FBD: Fixed block destination address**

Value	Description
0	Start address of first block of DMA transfer is set to the value stored in DMAi_DAn. Start address of consecutive blocks is the address following the previous block last address (according to transfer data width (DMAi_Bn:TW))
1	Start address of each block is set to the value stored in DMAi_DAn

**Note:**

- Setting of FBD is only effective if FD = "0" and (DMAi\_Bn:MS = "00" (block transfer mode) or DMAi\_Bn:MS = "01" (burst transfer mode)).



### 4.13. DMA Controller Channel Configuration E Register Channel n (DMAi\_En)

This register is prepared for reserved features. Do not use this register.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	EE	DC[14]	DC[13]	DC[12]	DC[11]	DC[10]	DC[9]	DC[8]
ACCESS_TYPE	R/W0	R/W0	R/W0	R/W0	R/W0	R/W0	R/W0	R/W0
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	DC[7]	DC[6]	DC[5]	DC[4]	DC[3]	DC[2]	DC[1]	DC[0]
ACCESS_TYPE	R/W0	R/W0	R/W0	R/W0	R/W0	R/W0	R/W0	R/W0
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	IC[15]	IC[14]	IC[13]	IC[12]	IC[11]	IC[10]	IC[9]	IC[8]
ACCESS_TYPE	R/W0	R/W0	R/W0	R/W0	R/W0	R/W0	R/W0	R/W0
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	IC[7]	IC[6]	IC[5]	IC[4]	IC[3]	IC[2]	IC[1]	IC[0]
ACCESS_TYPE	R/W0	R/W0	R/W0	R/W0	R/W0	R/W0	R/W0	R/W0
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

#### [bit31] EE: Reserved

Always write "0" to the bit when writing.

#### [bit30:16] DC[14:0]: Reserved

Always write "0" to the bits when writing.

#### [bit15:0] IC[15:0]: Reserved

Always write "0" to the bits when writing.

#### 4.14. DMA Controller Channel Configuration Source Address Shadow Register Channel "n" (DMAi\_SASHDWn)

This register holds the address of the last source data transfer for the DMA transfer of channel "n" in case of a source access error. If the DMA transfer is stopped it contains the next address following the last successful data transfer.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	SASHDW [31]	SASHDW [30]	SASHDW [29]	SASHDW [28]	SASHDW [27]	SASHDW [26]	SASHDW [25]	SASHDW [24]
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	SASHDW [23]	SASHDW [22]	SASHDW [21]	SASHDW [20]	SASHDW [19]	SASHDW [18]	SASHDW [17]	SASHDW [16]
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	SASHDW [15]	SASHDW [14]	SASHDW [13]	SASHDW [12]	SASHDW [11]	SASHDW [10]	SASHDW [9]	SASHDW [8]
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	SASHDW [7]	SASHDW [6]	SASHDW [5]	SASHDW [4]	SASHDW [3]	SASHDW [2]	SASHDW [1]	SASHDW [0]
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

##### [bit31:0] SASHDW[31:0]: Source Address Shadow

At the start of a DMA transfer this register contains a copy of DMAi\_SAn. During a DMA transfer it is either incremented, decremented, or stays constant according to the settings of DMAi\_Dn:FS, DMAi\_Dn:DES, DMAi\_Dn:US, and DMAi\_Dn:FBS. In case of a source access error SASHDW shows the address of the read access which causes the error. In case of a stop of the DMA transfer it shows the next address following the last read access done.

**Note:**

- SASHDW does not show an aligned address if a non-aligned address is loaded into DMAi\_SAn initially. However the DMAC will have made the read access to address SASHDW aligned according to the setting of transfer data width (DMAi\_Bn:TW).



#### 4.15. DMA Controller Channel Configuration Destination Address Shadow Register Channel "n" (DMAi\_DASHDWn)

This register holds the address of the last destination data transfer for the DMA transfer of channel "n" in case of a destination access error. If the DMA transfer is stopped it contains the next address following the last successful data transfer.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	DASHDW [31]	DASHDW [30]	DASHDW [29]	DASHDW [28]	DASHDW [27]	DASHDW [26]	DASHDW [25]	DASHDW [24]
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	DASHDW [23]	DASHDW [22]	DASHDW [21]	DASHDW [20]	DASHDW [19]	DASHDW [18]	DASHDW [17]	DASHDW [16]
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	DASHDW [15]	DASHDW [14]	DASHDW [13]	DASHDW [12]	DASHDW [11]	DASHDW [10]	DASHDW [9]	DASHDW [8]
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	DASHDW [7]	DASHDW [6]	DASHDW [5]	DASHDW [4]	DASHDW [3]	DASHDW [2]	DASHDW [1]	DASHDW [0]
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

##### [bit31:0] DASHDW[31:0]: Destination Address Shadow

At the start of a DMA transfer this register contains a copy of DMAi\_DAn. During a DMA transfer it is either incremented, decremented, or stays constant according to the settings of DMAi\_Dn:FD, DMAi\_Dn:DED, DMAi\_Dn:UD, and DMAi\_Dn:FBD. In case of a destination access error DASHDW shows the address of the write access which causes the error. In case of a stop of the DMA transfer it shows the next address following the last write access done.

##### Note:

- DASHDW does not show an aligned address if a non-aligned address is loaded into DMAi\_DAn initially. However the DMAC will have made the write access to address DASHDW aligned according to the setting of transfer data width (DMAi\_Bn:TW).

## 4.16. DMA Controller Client Matrix Internal Client Interface Configuration Register m (DMAi\_CMICm)

This register controls internal client interface m. Behavioral configuration bits determine the behavior of the acknowledge signals in case the DMA client is not selected. Also that this register controls enables DMA request for some peripherals.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved	Reserved	Reserved	Reserved	BEHSTPACK	Reserved	BEHREQACK	Reserved
ACCESS_TYPE	R0,WX	R0,WX	R0,WX	R/W0	R/W	R0,WX	R/W	R0,WX
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

**Note:**

- Attribute of 27th bit (BEHSTPACK) is "R/W" in DMAi\_CMICIC8 and DMAi\_CMICIC9, is "R0,WX" in DMAi\_CMICIC10, DMAi\_CMICIC11, ... and DMAi\_CMICIC142.

**[bit31:29] Reserved: Reserved bits**

**[bit28] Reserved: Reserved bit**

Always write "0" to this bit.

**[bit27] BEHSTPACK: Behavior stop acknowledge**

BEHSTPACK sets the behavior of the Internal DMA Client Interface m output signal DSTP\_ACK[m] if the client interface is not selected in any of the Channel Configuration Registers (DMAi\_CMCHICn).

**Note:**

- This configuration bit is read "0" in DMAi\_CMICICx (x=10, 11, ... ,142)





Value	Description
0	DSTP_ACK[m] outputs inactive logic level
1	DSTP[m] connects directly to DSTP_ACK[m]

**[bit25] BEHREQACK: Behavior request acknowledge**

BEHREQACK sets the behavior of the Internal DMA Client Interface m output signal DREQ\_ACK[m] if the client interface is not selected in any of the Channel Configuration Registers (DMAi\_CMCHICn).

Value	Description (1)
0	DREQ_ACK[m] outputs inactive logic level.
1	DREQ[m] connects directly to DREQ_ACK[m].

This bit also works as DMA request enable signal to the corresponding peripheral which send DMA request by IRQ. The bit must be changed while the peripheral function is disabled to guarantee that its IRQ is kept to disabled.

Value	Description (2)
0	For the corresponding peripheral whose DMA request is controlled by IRQ: DMA request by IRQ is disabled.
1	For the corresponding peripheral whose DMA request is controlled by IRQ: DMA request by IRQ is enabled.

**[bit24:0] Reserved: Reserved bits**

## 4.17. DMA Controller Client Matrix Channel Interface Configuration Register "n" (DMAi\_CMCHICn)

This register controls internal client interface "n".

BITS_OFFSET	31	30	29	28	27	26	25	24
BITS_NAME	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BITS_OFFSET	23	22	21	20	19	18	17	16
BITS_NAME	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BITS_OFFSET	15	14	13	12	11	10	9	8
BITS_NAME	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	CI[8]
ACCESS_TYPE	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R/W
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BITS_OFFSET	7	6	5	4	3	2	1	0
BITS_NAME	CI[7]	CI[6]	CI[5]	CI[4]	CI[3]	CI[2]	CI[1]	CI[0]
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	*1	*1	*1	*1

\*1: Initial value of DMAi\_CMCHICn:CI[8:0] is "n".

(e.g.) Initial value of DMAi\_CMCHIC0:CI[8:0]=000000000

Initial value of DMAi\_CMCHIC5:CI[8:0]=000000101

### [bit8:0] CI[8:0]: Client interface

Client Interface specifies that DMA channel n is routed to the DMA Client given by the binary value of CI.





## CHAPTER 23: Memory Protection (AHB)

This chapter explains the function and operation of the Memory Protection Unit for the AMBA™ Advanced High Speed Bus (MPU AHB).

---

1. Overview
2. Configuration
3. Operation
4. Registers
5. Precautions for Using



## 1. Overview

This section describes the features and the block diagram of the MPU AHB.

### Features of MPU AHB

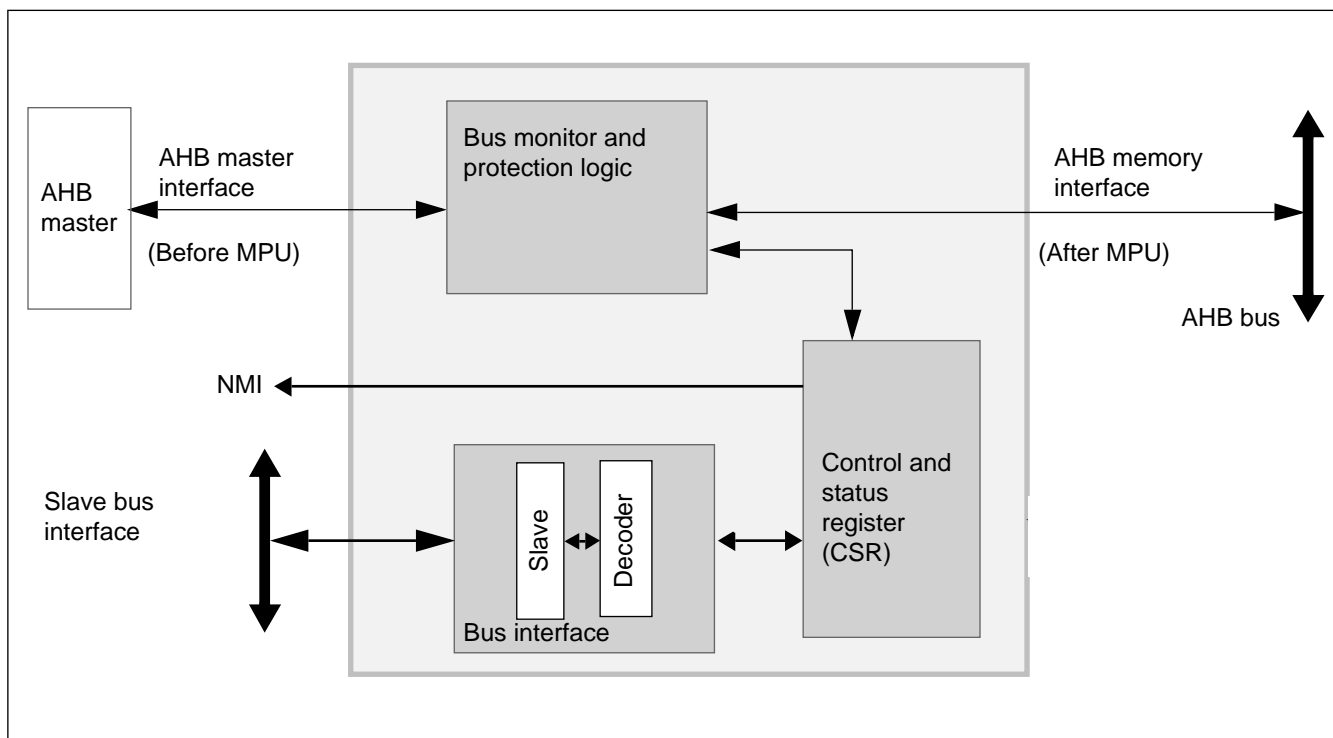
The MPU AHB module monitors the accesses from AHB masters and checks each access against an authorized set of access permissions. Access permissions (known as "permission attributes" from here onwards) are defined by access permissions bits. These bits are explained in Section MPU access permissions. The MPU AHB provides 8 regions and one background region. Each region has corresponding Access Permission bits that define the permission attributes for that particular region. Any unauthorized access to that memory space is flagged using a non-maskable interrupt. MPU AHB also collects information about unauthorized bus access and stores it in its internal registers. The features of the MPU AHB module are listed in this section:

- Each of the 8 regions in MPU AHB is specified using corresponding start address and end address
- The background region covers the entire 4GB address space
- With an unauthorized access, the MPU AHB generates an NMI to the CPU
- The MPU AHB collects information about the AHB Master bus access that caused the unauthorized access
- It supports 8-, 16- and 32-bit bus accesses for the configuration of the MPU AHB registers
- It supports lock and unlock feature to protect registers from illegal write accesses
- The modification of registers in the MPU AHB is only allowed in privileged mode
- MPU AHB registers can be written only after execution of the unlock sequence
- Supports MPU stop feature that allows blocking of all the accesses to memory space.
- It supports a privileged mode overwrite feature that allows the privilege attribute of the memory side AHB interface to be overwritten

## 2. Configuration

This section shows block diagram of the MPU AHB

Figure 2-1 Block Diagram of MPU AHB



### a) Bus Interface

The bus masters can access the MPU AHB module through its slave bus interface.

### b) Bus Monitor and Protection Logic

This logic monitors the transfers on AHB master interface bus. It finds out the region/s where the current transfer belongs to and then applies permissions based on the region match. It signals any permission violation using an NMI interrupt. All transfers on the AHB Master Interface (including the one that caused permission violation) are blocked until the NMI is cleared.

### c) Control and Status Registers

The operation of the MPU AHB can be controlled and monitored through its Control and Status Registers (CSR). For more details about the CSR see Section "4. Registers".



### 3. Operation

This section describes the operation of the MPU AHB.

#### (1) MPU AHB Region Granularity

The MPU AHB supports up to 8 regions and provides the start and end addresses for each of these 8 regions. MPU AHB regions are defined with a granularity of 32 bytes.

The start address specifies the first address of the region and is specified by registers MPUHm\_SADDR1 to MPUHm\_SADDR8 for region 1 to region 8 respectively. Since the region granularity is 32 bytes, the least significant 5 bits of the start addresses will read 0.

End Addresses are specified by registers MPUHm\_EADDR1 to MPUHm\_EADDR8 for region 1 to region 8 respectively. The least significant 5 bits of the End Address registers are read only bits and will always read 1. This ensures the granularity of 32 bytes.

#### (2) AHB Burst Monitoring

AHB protocol defines four, eight, and sixteen beat bursts. It also defines undefined length burst and single transfers.

In AHB protocol during each transfer within burst the next transfer address is always driven onto the bus.

Bus monitor and protection logic in MPU AHB monitors each address of transfer within AHB burst transaction. For each address, the region match is found and hence the corresponding permission attributes are checked against current transfer attributes.

Figure 3-1 Example of MPU AHB Timing

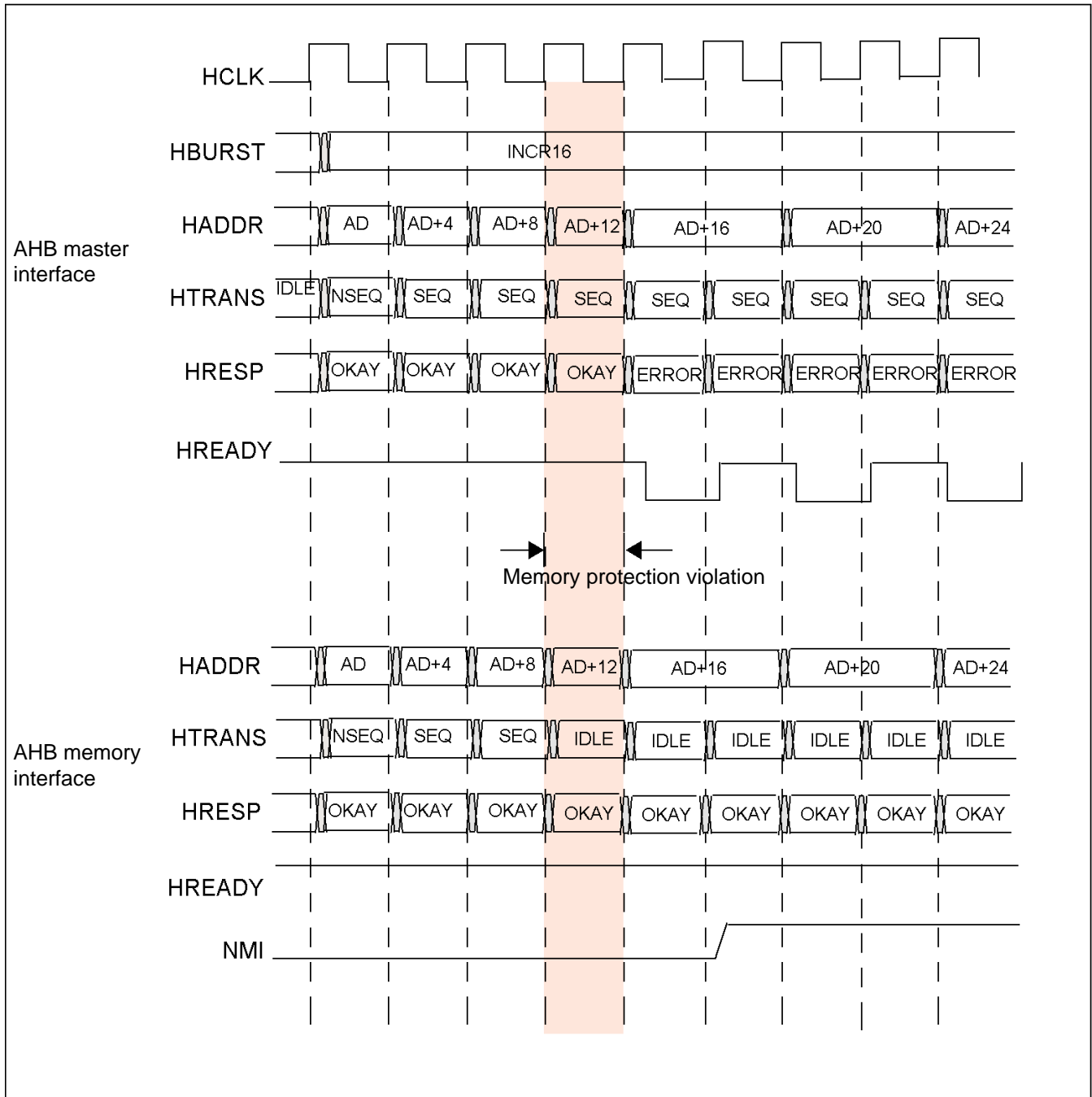


Figure 3-1 explains the operation of MPU AHB. It shows fixed length burst (INCR16) initiated on AHB master interface. MPU AHB checks each address of AHB transfer and checks for memory protection violation.

- MPU AHB detects a memory protection violation at address AD + 12
- MPU AHB starts driving IDLE transfers on HTRANS lines of AHB memory interface from that instance
- As address phase of AD + 12 is the data phase for address AD + 8, this phase is allowed to complete
- ERROR response is generated on AHB master interface for data phase of AD + 12
- MPUHm\_CTRL0:NMI flag is set

Error response on AHB master interface is generated until NMI flag is cleared by software. All AHB transfers are also blocked during this time.





### (3) Priority Decision of MPU

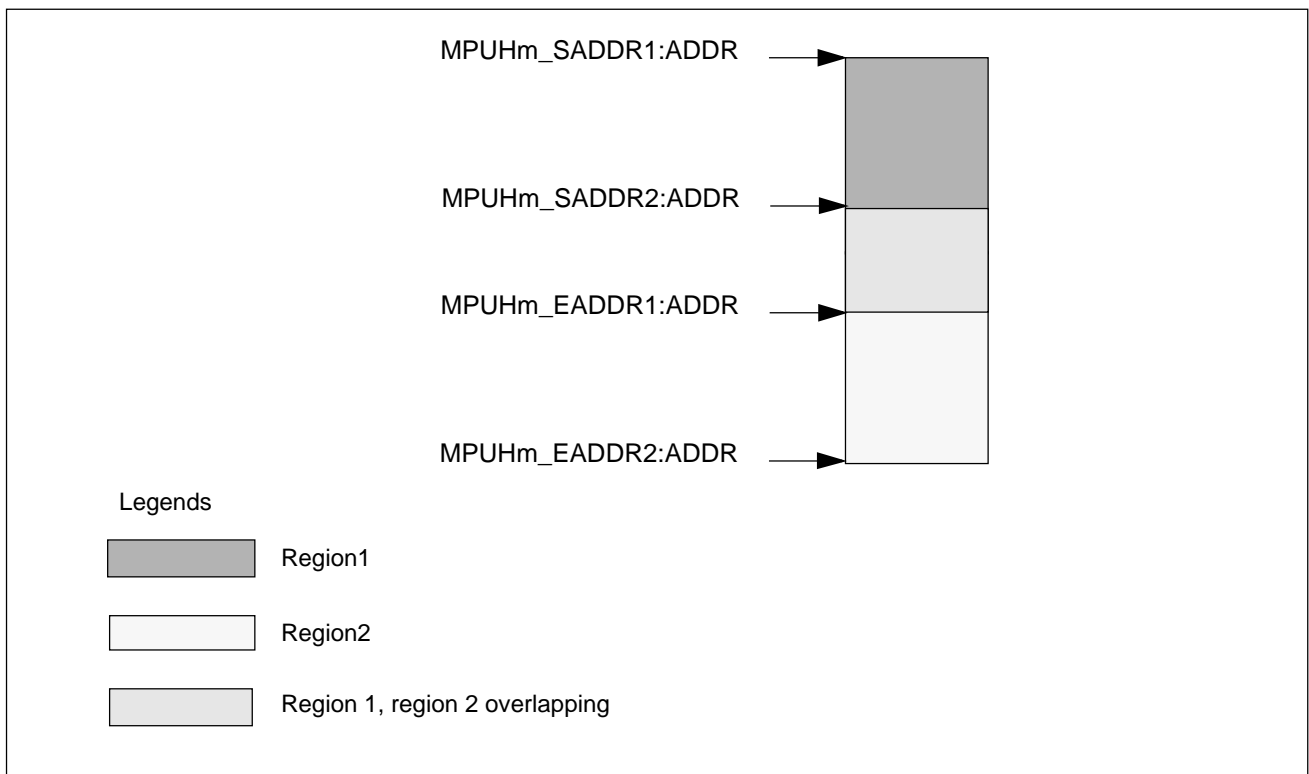
MPU AHB offers the starting address and the ending address of 8 each region. The region is permitted to overlap mutually in MPU AHB. Therefore, the AHB address of an arbitrary forwarding might be corresponding to two or more regions.

MPU supports 8 regions or less, and each region is identified as region 1, region 2, and region 3.... region 8. The priority of region 8 is the highest, and the priority of region 7 is high in the second. The priority of the backing ground region is lowest most, and the priority of region 1 does secondarily by being low.

When the address of the AHB forwarding is corresponding to two or more regions, the permission attribute corresponding to the region with the highest priority in a corresponding region is applied.

The example of the overlap in region 1 and region 2 with Figure 3-2 is shown.

**Figure 3-2 Example of Region Overlapping**



#### (4) MPU Access Permissions

Region control registers in the MPU AHB (i.e. MPUHm\_CTRL1 to MPUHm\_CTRL8) are used to control the access permission to regions 1 to 8. Also MPUHm\_CTRL0 is used to control the access permission for the background region.

Table 3-1 describes access permission bits (AP) in these registers and corresponding permission attributes.

**Table 3-1 Access Permissions**

AP Bits	Access in Privileged Mode	Access in Non-privileged Mode	Comment
000 (default)	No access	No access	All bus accesses are blocked and hence would generate a memory protection violation.
001	Read, write	No access	Reads and writes are permitted in privileged mode only. Any access in non-privileged mode would generate a memory protection violation.
010	Read, write	Read only	Reads and writes are permitted in privileged mode. Only reads are permitted in non-privileged mode. Writes in non-privileged mode would generate a memory protection violation.
011	Read, write	Read, write	All bus transfers are permitted. No memory protection violation is generated in this mode.
100	No access	No access	All bus accesses are blocked and hence generate a memory protection violation.
101	Read only	No access	Reads are permitted in privileged mode only. Writes in privileged mode and any access in non-privileged mode would generate a memory protection violation.
110	Read only	Read only	Reads are permitted in privileged as well as non-privileged mode. Any write access would generate a memory protection violation.
111	Read, write	Read, write	All bus transfers are permitted. No memory protection violation is generated in this mode.

#### (5) Bus Monitor and Protection Logic

All transfers on the AHB Master

Interface are monitored and checked for permitted access.

- Bus monitor and protection logic within the MPU AHB compares the address of the current transfer with the start and end addresses of each region to find a region match, i.e. where the current transfer matches one of the 8 defined regions.
- As explained in Section "Priority decision of MPU", the AHB transfer address may match multiple regions where the permission attributes of the region with highest priority are checked against the attributes of the currently applied transfer from the AHB master.
- If the attributes of the currently applied transfer are within the permitted attributes, the current transaction is passed on to the AHB memory interface.
- If the attributes are not within permitted attributes, the current transfer is blocked. The non-maskable interrupt flag (MPUHm\_CTRL0:NMI) is set. The address and control information of the current transfer is stored in MPUHm\_MERRA and MPUHm\_MERRC respectively.
- All further transfers are blocked until the MPUHm\_CTRL0:NMI flag is cleared by software. Further monitoring of the AHB transfer addresses is also stalled until the MPUHm\_CTRL0:NMI flag is cleared.



When a transfer is blocked, the MPU AHB performs the following actions:

- It drives idle transfers on the AHB memory interface
- It generates an error response on the AHB master interface

#### **(6) MPU Stop Feature**

MPU AHB supports MPU stop feature.

When this mode is enabled all accesses to memory space are blocked and MPU AHB does the following actions.

- Drives idle transfers on the AHB memory interface
- Generates error response on AHB master interface

For more details on MPU stop related register bits see Section "4.1. MPU AHB Control Register (MPUHm\_CTRL0)".

#### **(7) Privileged Mode Overwrite Feature**

The MPU AHB supports privileged mode overwrite feature.

When this mode is enabled, the privileged mode attribute on the AHB memory interface is set by setting the MPUHm\_CTRL0:PROT bit as explained in Section "4.1. MPU AHB Control Register (MPUHm\_CTRL0)".

**Note:**

- *Bus monitor and protection logic for detecting memory protection violation uses the privileged mode attribute on the AHB master interface and is setting MPUHm\_CTRL0:PROT bit, even when the privileged mode overwrite feature is enabled.*

## 4. Registers

The MPU AHB module contains various registers to configure its operation, to monitor its status and to read the information it has collected from the AHB master interface at the time of a memory protection violation.

The MPU AHB module is allocated 1 KB of MCU address space for mapping the Configuration and Status Registers (CSR). The address area allocated to the MPU AHB and the CSRs in the MPU AHB are explained in this section.

The following registers are available for the MPU AHB:

**Table 4-1 List of MPU AHB Registers**

Abbreviated Register Name	Register Name	See
MPUHm_CTRL0	MPU AHB Control Register	4.1
MPUHm_NMIEN	MPU AHB NMI Enable Register	4.2
MPUHm_MERRC	MPU AHB Memory Error Control Register	4.3
MPUHm_MERRA	MPU AHB Memory Error Address Register	4.4
MPUHm_CTRL1 to 8	MPU AHB Control Register	4.5
MPUHm_SADDR1 to 8	MPU AHB Start Address Register	4.6
MPUHm_EADDR1 to 8	MPU AHB End Address Register	4.7
MPUHm_UNLOCK	MPU AHB Unlock Register	4.8
MPUHm_MID	MPU AHB Module ID Register	4.9

The suffix "m" in the register name indicates that the register is an instance "m" of the module.



## 4.1. MPU AHB Control Register (MPUHm\_CTRL0)

MPU AHB control register can be used by the software to configure the MPU AHB. This register provides enable bit to enable the MPU AHB monitoring and protection function. It also provides permission attributes for background region. It also provides controls for enabling or disabling of privileged mode overwrite feature. Lastly it provides status of MPU AHB lock bit and non-maskable interrupt flag.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved	Reserved	Reserved	Reserved	Reserved	AP[2]	AP[1]	AP[0]
ACCESS_TYPE	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R/W	R/W	R/W
PROT_TYPE	WPS							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	MPUENC	MPUEN
ACCESS_TYPE	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R/W	R,WX
PROT_TYPE	WPS							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved	Reserved	Reserved	PROT	POEN	MPU STOPEN	MPUSTOP	LST
ACCESS_TYPE	R0,WX	R0,WX	R0,WX	R/W	R/W	R/W	R,WX	R,WX
PROT_TYPE	WPS							
INITIAL_VALUE	0	0	0	0	0	0	0	1

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	NMICL	NMI
ACCESS_TYPE	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,W	R,WX
PROT_TYPE	WPS							
INITIAL_VALUE	0	0	0	0	0	0	0	0

**[bit31:27] Reserved: Reserved bits**

**[bit26:24] AP[2:0]: Access permissions for background region**

These bits are used to control access permissions for background region. For detailed description of these bits, see Table 3-1.

**Note:**

- AP[2:0] bits for background region can only be written when MPU is disabled (MPUHm\_CTRL0:MPUEN = 0). Set this bit after writing "0" to MPUHm\_CTRL0:MPUENC and confirming that MPUHm\_CTRL0:MPUEN bit return 0 read.

**[bit23:18] Reserved: Reserved bits**

**[bit17] MPUENC: MPU AHB enable control**

Value	Description
0	Disables MPU AHB monitoring and protection function
1	Enables MPU AHB monitoring and protection function

Read MPUEN bit and confirm state of MPU AHB monitoring and protection function.

**[bit16] MPUEN: MPU enable status**

Value	Description
0	MPU AHB monitoring and protection function is disabled. All accesses from AHB master interface are passed on to AHB memory interface without any protection
1	MPU AHB monitoring and protection function is enabled

**[bit15:13] Reserved: Reserved bits**

**[bit12] PROT: Privileged mode attribute**

When POEN=1, the privilege attribute on AHB memory interface is controlled by this bit.

Value	Description
0	Non-privileged mode
1	Privileged mode

**[bit11] POEN: Privileged mode overwrite feature enable**

Value	Description
0	Privileged mode overwrite feature is disabled
1	Privileged mode overwrite feature is enabled

**[bit10] MPUSTOPEN: Enable for MPU stop feature**

This bit along with MPU stop input controls the stop status of MPU AHB.

Value	Description
0	MPU stop feature is disabled
1	MPU stop feature is enabled

**[bit9] MPUSTOP: MPU stop status**

Value	Description
0	MPU AHB is not in stop mode
1	MPU AHB is in stop mode (i.e. MPUSTOPEN="1" and MPU stop input is asserted). All accesses are blocked in this mode

**[bit8] LST: MPU lock status**

Value	Description
0	MPU AHB is unlocked, registers in MPU AHB can be written
1	MPU AHB is locked, no registers (other than MPUHm_UNLOCK register) in MPU AHB can be written

**[bit7:2] Reserved: Reserved bits****[bit1] NMICL: NMI interrupt clear**

Value	Description
0	No effect
1	Clear the NMI interrupt flag

The read value is always "0".

**[bit0] NMI: NMI interrupt flag**

This interrupt flag indicates that the memory protection violation was detected for an AHB transfer.

**Note:**

- The register bits MPUHm\_CTRL0:AP[2:0], MPUHm\_CTRL0:POEN and MPUHm\_CTRL0:PROT can only be written when the MPU is disabled (MPUHm\_CTRL0:MPUEN = 0). When MPU is disabled, write to bit[23:16] in byte access to avoid simultaneous writing to these bits.

## 4.2. MPU AHB NMI Enable Register (MPUHm\_NMIEN)

MPU AHB NMI Enable Register can be used by software to reset the NMI enable bit. The default value of the NMI enable bit is 1. This bit can be reset by software only once after a reset operation.

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	WPS
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	NMIEN
ACCESS_TYPE	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R,W
PROT_TYPE	WPS							
INITIAL_VALUE	0	0	0	0	0	0	0	1

**[bit31:1] Reserved: Reserved bits**

**[bit0] NMIEN: NMI interrupt enable**

This bit decides whether the NMI interrupt flag is routed to an NMI interrupt signal or not.

Value	Description
0	NMI interrupt flag does not trigger an NMI interrupt signal
1	NMI interrupt flag triggers an NMI interrupt signal

The value of this bit can be changed only once after reset.





### 4.3. MPU AHB Memory Error Control Register (MPUHm\_MERRC)

This is a read-only register that provides the control information of AHB transfer for which memory protection violation was detected. This register can be read to get the privileged mode and transfer mode (write/read) information of the AHB transfer. This register is read-only. Writing to this register returns a bus error.

BITS	31	30	29	28	27	26	25	24
BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BITS	23	22	21	20	19	18	17	16
BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BITS	15	14	13	12	11	10	9	8
BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BITS	7	6	5	4	3	2	1	0
BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	HPROT	HWRITE
ACCESS_TYPE	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

**[bit31:2] Reserved: Reserved bits**

**[bit1] HPROT: AHB transfer privileged mode**

This bit provides the status of the HPROT signal of the AHB transfer for which memory protection violation is detected.

**[bit0] HWRITE: AHB transfer mode**

This bit provides the status of the HWRITE signal of the AHB transfer for which memory protection violation is detected.

#### 4.4. MPU AHB Memory Error Address Register (MPUHm\_MERRA)

This is a read-only register that provides the address of AHB transfer for which memory protection violation was detected. This register is read-only. Writing to this register returns a bus error.

BITS_OFFSET	31	30	29	28	27	26	25	24
BITS_NAME	HADDR[31]	HADDR[30]	HADDR[29]	HADDR[28]	HADDR[27]	HADDR[26]	HADDR[25]	HADDR[24]
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BITS_OFFSET	23	22	21	20	19	18	17	16
BITS_NAME	HADDR[23]	HADDR[22]	HADDR[21]	HADDR[20]	HADDR[19]	HADDR[18]	HADDR[17]	HADDR[16]
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BITS_OFFSET	15	14	13	12	11	10	9	8
BITS_NAME	HADDR[15]	HADDR[14]	HADDR[13]	HADDR[12]	HADDR[11]	HADDR[10]	HADDR[9]	HADDR[8]
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BITS_OFFSET	7	6	5	4	3	2	1	0
BITS_NAME	HADDR[7]	HADDR[6]	HADDR[5]	HADDR[4]	HADDR[3]	HADDR[2]	HADDR[1]	HADDR[0]
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

##### [bit31:0] HADDR[31:0]: AHB address

The address of an AHB transfer for which memory protection violation was read.



## 4.5. MPU AHB Region Control Registers (MPUHm\_CTRL1 to 8)

MPU AHB Region Control Register is used to specify access permission for a particular region. Software can also use this register for enabling or disabling the particular region.

MPUHm\_CTRL1 control register for region 1 is explained below. This product can use 8 regions.

BIT_OFFSET	31-16
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	WPS
INITIAL_VALUE	00000000_00000000

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved					AP		
ACCESS_TYPE	R0,WX					R/W		
PROT_TYPE	WPS							
INITIAL_VALUE	00000					000		

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved						MPUENC	MPUEN
ACCESS_TYPE	R0,WX						R/W	R,WX
PROT_TYPE	WPS							
INITIAL_VALUE	000000						0	0

**[bit31:11] Reserved: Reserved bits**

**[bit10:8] AP[2:0]: Access permissions**

These bits are used to control access permissions for region 1. For a detailed description of these bits see Table 3-1.

**[bit7:2] Reserved: Reserved bits**

**[bit1] MPUENC: Enable control**

Value	Description
0	Memory Protection for region 1 is disabled
1	Memory Protection for region 1 is enabled

The enable status of the region can be read through the MPUHm\_CTRL1:MPUEN bit.

**[bit0] MPUEN: Enable status**

Value	Description
0	Memory Protection for region 1 is disabled
1	Memory Protection for region 1 is enabled

This is a read-only bit, writing to this bit has no effect.

**Note:**

- The access permission bits (i.e. MPUHm\_CTRL1 to 8:AP) can only be written when the corresponding region is disabled (i.e. MPUHm\_CTRL1 to 8:MPUEN = 0) or the MPU is disabled (MPUHm\_CTRL0:MPUEN = 0). When memory protection of region 1 is disabled, write to bit[7:0] in byte access to avoid simultaneous writing to these bits. The region 2 to 8 are same.



## 4.6. MPU AHB Start Address Registers (MPUHM\_SADDR1 to 8)

Each region in MPU AHB can be defined by specifying the start address and end address for that particular region. The MPUHM\_SADDR1 to 8 registers are used to specify the start address for the 8 regions. The start address indicates the first address for that region. MPUHM\_SADDR1 is the start address register for region 1 and is explained below. This product can use 8 regions.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	SADDR[31]	SADDR[30]	SADDR[29]	SADDR[28]	SADDR[27]	SADDR[26]	SADDR[25]	SADDR[24]
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	WPS							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	SADDR[23]	SADDR[22]	SADDR[21]	SADDR[20]	SADDR[19]	SADDR[18]	SADDR[17]	SADDR[16]
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	WPS							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	SADDR[15]	SADDR[14]	SADDR[13]	SADDR[12]	SADDR[11]	SADDR[10]	SADDR[9]	SADDR[8]
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	WPS							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	SADDR[7]	SADDR[6]	SADDR[5]	SADDR[4]	SADDR[3]	SADDR[2]	SADDR[1]	SADDR[0]
ACCESS_TYPE	R/W	R/W	R/W	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX
PROT_TYPE	WPS							
INITIAL_VALUE	0	0	0	0	0	0	0	0

### [bit31:0] SADDR[31:0]: Start address

These bits are used to set start address for each region.

#### Note:

- The MPUHM\_SADDR1 to 8 registers can only be written when the corresponding region is disabled (MPUHM\_CTRL1 to 8:MPUEN = 0) or the MPU is disabled (MPUHM\_CTRL0:MPUEN = 0).

## 4.7. MPU AHB End Address Registers (MPUHm\_EADDR1 to 8)

Each region in MPU AHB can be defined by specifying a start address and end address for that particular region. MPUHm\_EADDR1 to 8 registers are used to specify the end addresses for the 8 regions. The end address indicates the last address for that region. The MPUHm\_EADDR1 End Address Register for Region 1 is explained below. This product can use 8 regions.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	EADDR[31]	EADDR[30]	EADDR[29]	EADDR[28]	EADDR[27]	EADDR[26]	EADDR[25]	EADDR[24]
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	WPS							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	EADDR[23]	EADDR[22]	EADDR[21]	EADDR[20]	EADDR[19]	EADDR[18]	EADDR[17]	EADDR[16]
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	WPS							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	EADDR[15]	EADDR[14]	EADDR[13]	EADDR[12]	EADDR[11]	EADDR[10]	EADDR[9]	EADDR[8]
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	WPS							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	EADDR[7]	EADDR[6]	EADDR[5]	EADDR[4]	EADDR[3]	EADDR[2]	EADDR[1]	EADDR[0]
ACCESS_TYPE	R/W	R/W	R/W	R1,WX	R1,WX	R1,WX	R1,WX	R1,WX
PROT_TYPE	WPS							
INITIAL_VALUE	0	0	0	1	1	1	1	1

### [bit31:0] EADDR[31:0]: End address

These bit are used to set end address for each region.

#### Note:

- MPUHm\_EADDR1 to 8 registers can only be written when the corresponding region is disabled (MPUHm\_CTRL1 to 8:MPUEN = 0) or the MPU is disabled (MPUHm\_CTRL0:MPUEN = 0).



## 4.8. MPU AHB Unlock Register (MPUHM\_UNLOCK)

The software can use this register to lock or unlock the MPU AHB registers for write access.

BIT_OFFSET	31-0
BIT_NAME	UNLOCK
ACCESS_TYPE	R0,W
PROT_TYPE	WP
INITIAL_VALUE	00000000_00000000_00000000_00000000

### [bit31:0] UNLOCK[31:0]: MPU AHB unlock

The MPU AHB Unlock Register protects the MPU AHB module from being modified accidentally by software. The MPU AHB registers cannot be written until this register has been written with a specific unlock value. The correct value for unlocking can be written only in privileged mode. Reading this register always returns "0". To lock the MPU AHB again software must write another value specific to lock. A write access to the MPU AHB registers without unlocking or writing value other than the lock or unlock value to this register causes a protection error.

- Unlock value: 0x ACCABB56
- Lock value: 0x112ABB56

#### **Note:**

- *This register cannot be written by an 8- or 16-bit write access; any such access causes a protection error.*

## 4.9. MPU AHB Module ID Register (MPUHm\_MID)

This register has no function and returns a meaningless constant. This register is read-only. Writing to this register returns a bus error.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	MID[31]	MID[30]	MID[29]	MID[28]	MID[27]	MID[26]	MID[25]	MID[24]
ACCESS_TYPE	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	MID[23]	MID[22]	MID[21]	MID[20]	MID[19]	MID[18]	MID[17]	MID[16]
ACCESS_TYPE	R0,WX	R0,WX	R0,WX	R0,WX	R1,WX	R1,WX	R0,WX	R1,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	1	1	0	1

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	MID[15]	MID[14]	MID[13]	MID[12]	MID[11]	MID[10]	MID[9]	MID[8]
ACCESS_TYPE	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	MID[7]	MID[6]	MID[5]	MID[4]	MID[3]	MID[2]	MID[1]	MID[0]
ACCESS_TYPE	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

### [bit31:0] MID[31:0]: Module ID

"0x000d0000" is always retrieved.





## 5. Precautions for Using

This section is the programmer's guide, which lists the usage notes for programming the MPU AHB module. It is recommended to read these guidelines before programming the MPU AHB module.

### (1) General Usage Notes

- Reserved bits return value "0". The software programs shall be independent of the values read from the reserved register bits
- The MPU AHB supports storage of information of only first memory protection violation on the bus. Therefore, once an NMI interrupt flag is set, the further monitoring of AHB master interface is stalled until MPUHm\_CTRL0:NMI bit is cleared. This implies that any memory protection violation occurring on the bus, while the MPUHm\_CTRL0:NMI is set, is simply ignored by MPU AHB

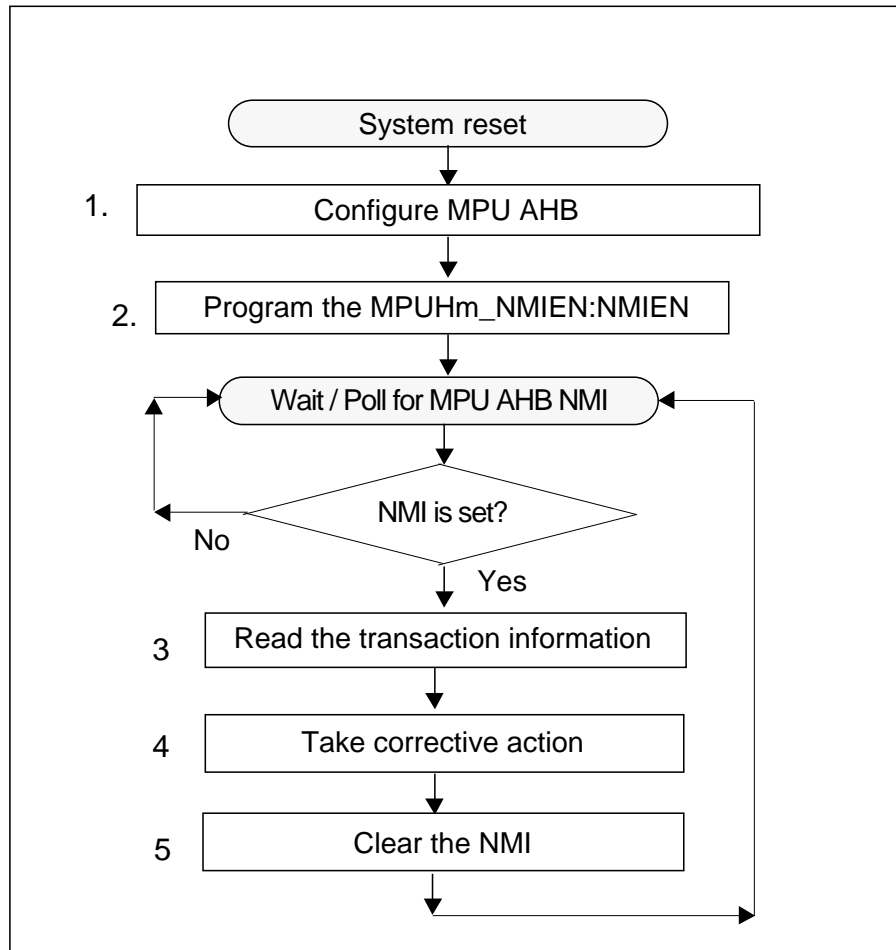
#### **Note:**

- *Software developers must try to keep the ISR size small, so that the NMI interrupt of MPU AHB does not remain unattended for long.*

## (2) Steps in Programming the MPU AHB Module

Figure 5-1 gives the general steps a programmer shall follow while using the MPU AHB module.

Figure 5-1 Block Diagram of MPU AHB



1. After the system reset, the software detects the module ID number of MPU AHB, by reading the MPUHm\_MID register. This helps it in identifying the attributes and capabilities supported by the MPU AHB module. Then the software configures MPU AHB by setting appropriate registers.
2. By default, MPU AHB propagates the MPUHm\_CTRL0:NMI flag to the CPU through the Interrupt Controller. If polling mode is desired, the software can reset the MPUHm\_NMIEN:NMIEN bit to "0".

**Note:**

- The MPUHm\_NMIEN:NMIEN can be written only once after reset. Subsequent write accesses to this bit has no visible impact on the state of this bit.



3. When the NMI is triggered (or in polling mode, if the software detects during its polling cycle that the MPUHm\_CTRL0:NMI status flag is set), the CPU is invoked. This reads the status information collected and stored by MPU AHB in its CSR.
4. The software diagnoses the information about the protection violation transaction and initiates a corrective action (if any).
5. Once the software has processed the information from the status registers, it clears the MPUHm\_CTRL0:NMI flag by writing a "1" to the MPUHm\_CTRL0:NMICL bit. Clearing MPUHm\_CTRL0:NMI flag ensures that the MPU AHB starts monitoring the AHB master interface again for checking memory protection violation.

**Note:**

- Software may clear the NMI flag before taking corrective action, hence steps 4 and 5 could be interchanged.



## CHAPTER 24: Time Protection

This chapter explains time protection.

---

1. Overview
2. Configuration
3. Operation
4. Setting Procedure Examples
5. Registers
6. Others



## 1. Overview

This section provides an overview of time protection.

Modern operating systems (especially those with time-based triggers) need to manage various time-based operations of tasks. For example, tasks involve the following types of time:

- Overall maximum execution time
- Time limit from start to end
- Maximum tolerated period during which certain-level interrupts are prohibited
- Maximum tolerated period during which all interrupts are prohibited
- Minimum rate of restart

To satisfy the requirements, each CPU is equipped with a timing protection unit to provide hardware-based support.

The timing protection unit provides the following features:

- 8 timers of the same type are provided, each of which is denoted by the timer number "m". (m = 0 to 7)  
These timers are used for the execution time, the lock period, the arrival time interval, and the protection of the time limit.
- 24-bit up-counter, for which normal mode or overflow mode can be selected
- Generation of NMI interrupts
- Setting of end counts and preload values
- Preload function that loads the preload value in the overflow mode
- Free-run function that restarts the timer automatically
- Clock division enabled by a global prescaler (division ratio: 1/1 to 1/64)  
This function divides the system clock into the frequency required as the input to each timer-specific prescaler.
- Clock division by each timer-specific prescaler (division ratio: 1/1, 1/2, 1/4, or 1/16)  
Dividing the output clock from the global prescaler into the clock used by the timer
- Reading the current count value
- Control of start, stop, and restart of each timer
- Notification of each timer status (stopped or operating)
- Equipped with a slave interface of AHB 64 bits for register access
- Support of the debug mode for stalling timers

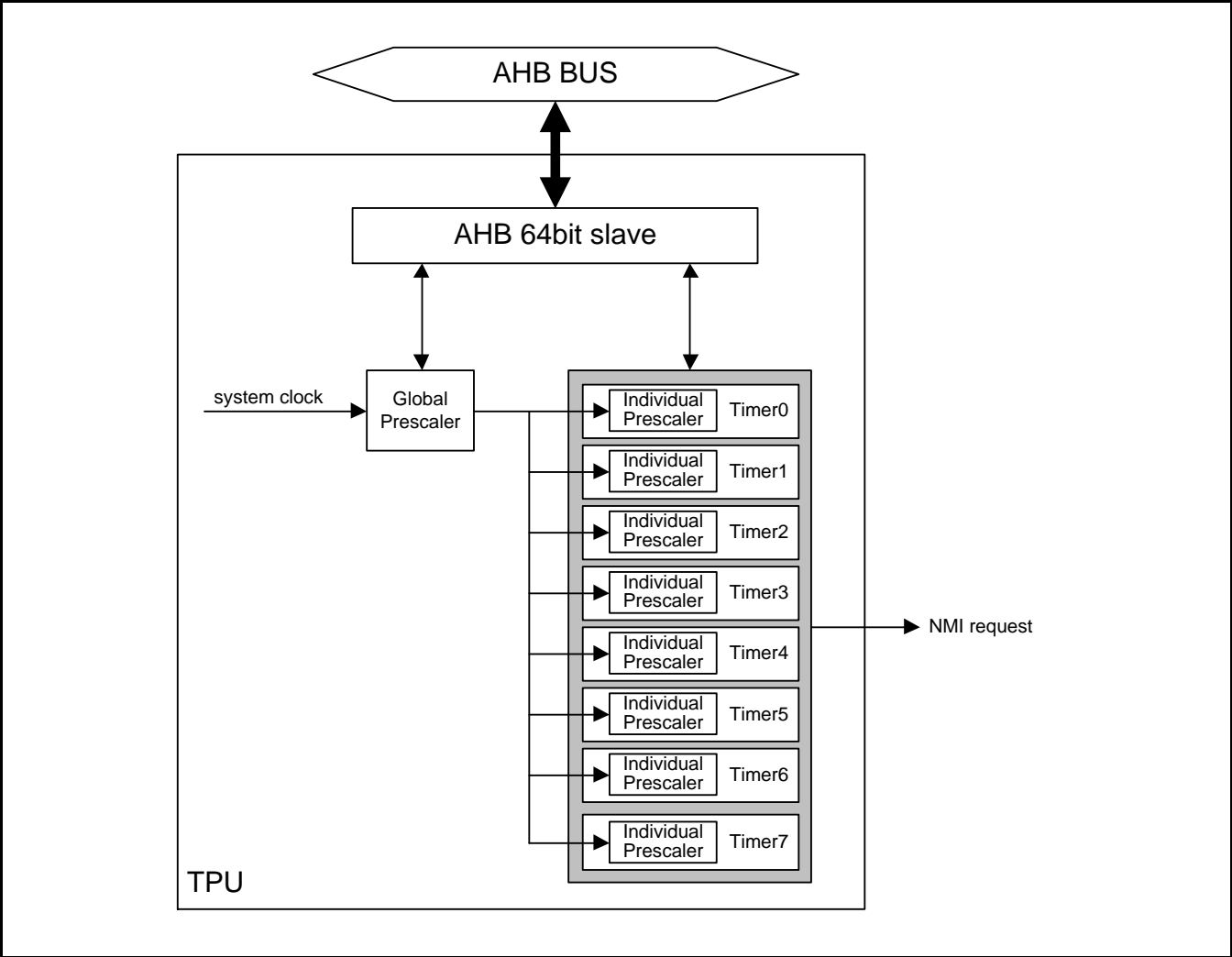
### Notes:

- This chapter sometimes refers to the timing protection unit by its abbreviation TPU (Timing Protection Unit).
- "n" in the abbreviated register name of "TPUn\_\*\*\*\*" indicates the CPU number, while "m" in "TPUn\_\*\*m" or "TPUn\_\*\*\*:\*\*m" indicates the timer number.

2. Configuration

This section explains a block diagram of the timing protection unit.

Figure 2-1 Block Diagram of Timing Protection Unit





### 3. Operation

This section explains the operation of the timing protection unit.

#### (1) Sequence Protection by the Lock Release Register

The timing protection unit provides the sequence protection feature using a lock release register (TPUn\_UNLOCK). The lock must be released by writing the lock release value in PUn\_UNLOCK before the timer setting can be updated. Writing the locking value after making a timer setting sets the locked state again.

#### (2) Timer Operation Mode

##### a) Normal Mode

The condition for generating an interrupt request in the normal mode is "the current timer count value  $\geq$  end count value". The current timer count value being counted up is compared with the end-count/preload-value setting bits (TPUn\_TCN0m:ECPL) in timer control register 0. If the current count value is equal to or larger than the end-count value, an interrupt request occurs and the corresponding bit (TPUn\_TIR:IRm) in the timer interrupt request register becomes "1". If an interrupt request occurs and the corresponding bit (TPUn\_TIE:IEm) in the timer interrupt enable register is enabled, an interrupt for CPU is generated.

To start the timer operation, write "1" in the operation start bit (TPUn\_TCN0m:START) in timer control register 0. The timer is reset and the counting starts from "0x000000". During timer operation, the corresponding bit (TPUn\_TST:STm) in the timer status register keeps the value "1".

The timer automatically stops when an interrupt request occurs.

To stop the timer operation arbitrarily, write "1" in the operation stop bit (TPUn\_TCN0m:STOP) in timer control register 0. TPUn\_TST:STm becomes "0" when the timer stops. The current timer count value of the stopped timer is stalled.

To restart the timer, write "1" in the operation restart bit (TPUn\_TCN0m:CONT) in timer control register 0. The current timer count value that has been stalled at the time of timer stop is used as the starting point of the counting. TPUn\_TST:STm keeps the value "1" during timer operation.

##### b) Overflow Mode

The condition for generating an interrupt request in the overflow mode is timer overflow. If a count up occurs when the current timer count value is "0xFFFFFFFF", an interrupt request occurs and TPUn\_TIR:IRm becomes "1". If an interrupt request occurs and TPUn\_TIE:IEm is enabled, an interrupt to CPU is generated.

Start, stop, and restart of the operation are the same as those in normal mode.

##### c) Debug Mode

All timers stop when "1" is written in the debug mode enable/disable bit (TPUn\_CFG:DBGE) in the configuration register and the CPU enters the debug state. The current timer count values of the stopped timers are stalled. The timers start operation as soon as the above condition is no longer true.

For the definition of the debug state, see "12.8 of Cortex<sup>TM</sup>-R5 Revision:r1p2 Technical Reference Manual (ARM DDI 0460D)".

### **(3) Preload Function**

This function sets the counting start value in the overflow mode to any specified value, instead of setting it to the normal "0". This function is enabled by writing "1" in the preload function enable/disable setting bit (TPUn\_TCN1m:PL) in timer control register 1. Any counting start value being set is called a preload value. The preload value is set by TPUn\_TCN0m:ECPL.

This function cannot be used in the normal mode.

### **(4) Free-run Function**

This function automatically restarts the timer, instead of stopping it, when an interrupt request occurs. This function is enabled by writing "1" in the free-run function enable/disable setting bit (TPUn\_TCN1m:FRT) in timer control register 1.

This function can be used in both normal and overflow modes.

The count starting value at restart is the same as in normal operation: "0x000000" for the normal mode, and "0x000000" or the preload value in the overflow mode.

Because the timer restarts as soon as an interrupt request occurs, the condition TPUn\_TST:STm="1" continues. When the condition for interrupt request generation is satisfied, TPUn\_TIR:IRm becomes "1". The value stays at "1" if the previous interrupt request is not cleared.





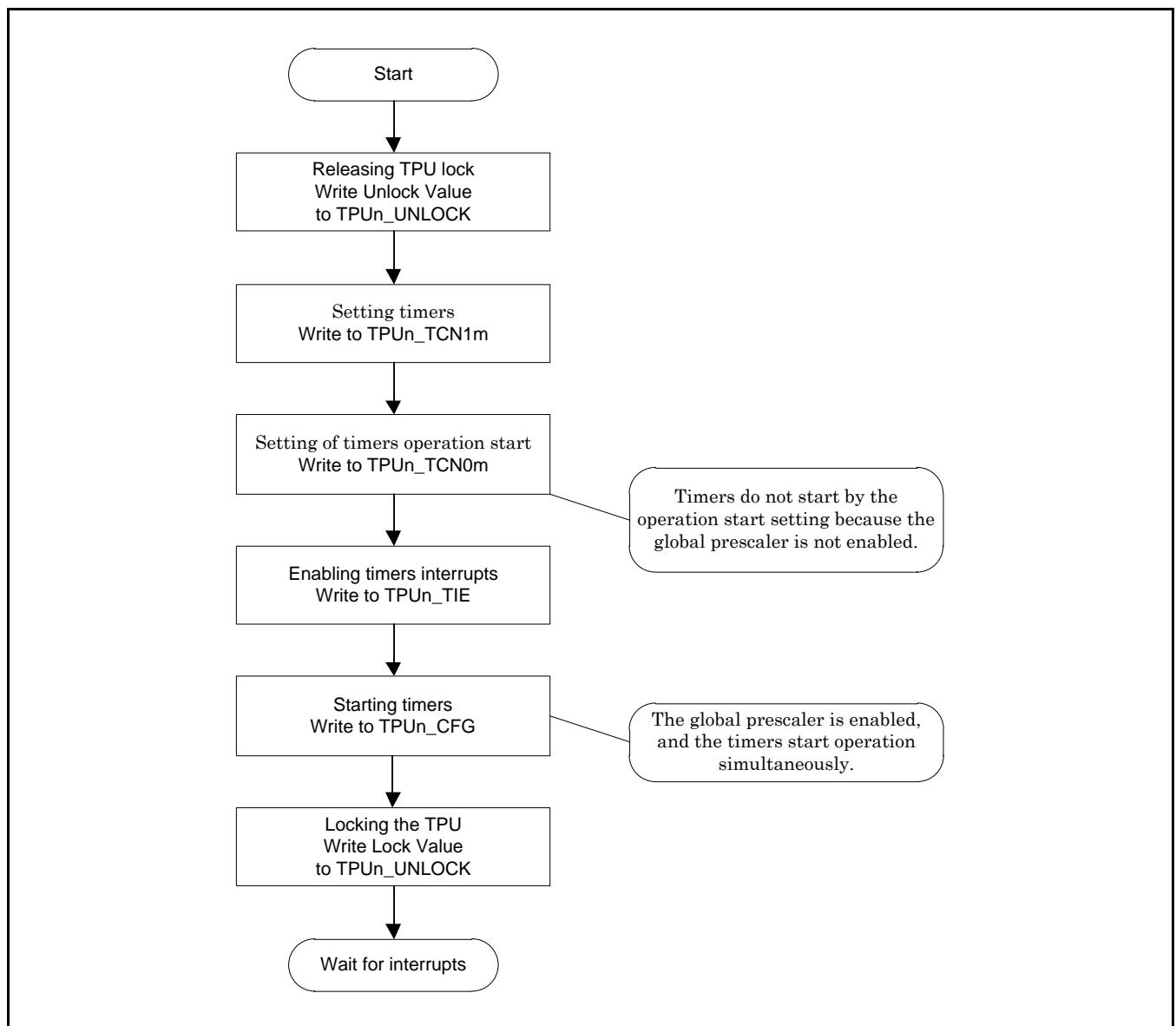
## 4. Setting Procedure Examples

This section explains provides setting procedure examples of the timing protection unit.

### (1) When Multiple Timers are Simultaneously Started from the Initial State

When multiple timers are simultaneously started from the initial state, the enable setting of the global prescaler is used as the trigger. Configure TPU<sub>n</sub>\_TCN1m and TPU<sub>n</sub>\_TCN0m of the target timers to the operational status before enabling the global prescaler. At this time the timers are not still operational because the counters are not supplied with the clock. Enabling the global prescaler simultaneously starts the timers that have previously been configured with the operation start setting.

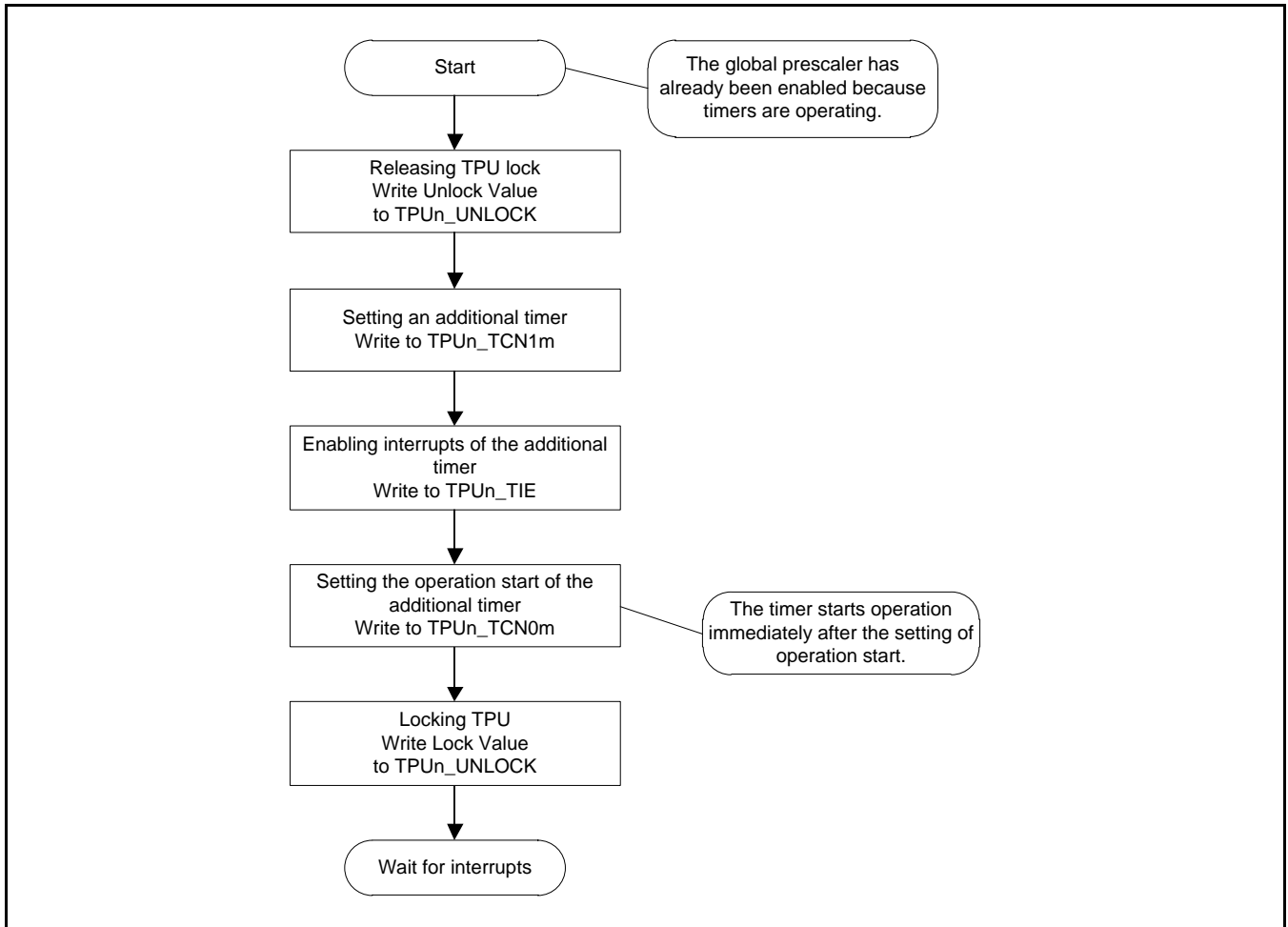
**Figure 4-1 Timing Protection Unit Setting Flow (When Multiple Timers are Simultaneously Started From the Initial State)**



## (2) Starting an Additional Timer While Some Timers are in Operation

When starting an additional timer while some timers are in operation, the setting action for starting the additional timer directly triggers the start of the timer because the global prescaler is already in operation providing timer clocks to timers.

Figure 4-2 Timing Protection Unit Setting Flow (Starting an Additional Timer While Some Timers are in Operation)



**Note:**

- Because the timer is started when the global prescaler is already in operation, the state of the global prescaler varies depending on the timing of the setting action for the timer start. For this reason, the actual time required before generation of an interrupt may have a margin of error. For example, suppose that the global prescaler has been started with a division ratio of 1/12. Also, suppose that it takes a period of 2 system clock cycles between the setting action for the timer start and the detection of the first clock edge of the global clock. In this case, there will be a time difference of 10 system clock cycles. The maximum time difference depends on the division ratio of the global prescaler.



## 5. Registers

This section explains the registers used by the timing protection unit.

**Table 5-1 List of TPU Registers**

Abbreviated Register Name	Register Name	See
TPUn_UNLOCK	TPU Lock Release Register	5.1
TPUn_LST	TPU Lock Status Register	5.2
TPUn_CFG	TPU Configuration Register	5.3
TPUn_TIR	TPU Timer Interrupt Request Register	5.4
TPUn_TST	TPU Timer Status Register	5.5
TPUn_TIE	TPU Timer Interrupt Enable Register	5.6
TPUn_TCN0m	TPU Timer m Control Register 0	5.7
TPUn_TCN1m	TPU Timer m Control Register 1	5.8
TPUn_TCCm	TPU Timer m Current Count Register	5.9

## 5.1. TPU Lock Release Register (TPUn\_UNLOCK)

This register controls the write lock of the registers of the timing protection unit.

BIT_OFFSET	31-0
BIT_NAME	UNLOCK
ACCESS_TYPE	R0,W
PROT_TYPE	WP
INITIAL_VALUE	00000000_00000000_00000000_00000000

### [bit31:0] UNLOCK[31:0]: Lock release bits of the timing protection unit

These bits control the write lock of the configuration registers of the timing protection unit.

TPUn\_CFG and TPUn\_TCN10 to TPUn\_TCN17 are subject to sequence protection by this register.

Value	Description
0xACC5A110	Unlock value (which enables writing)
0xB10CACC5	Lock value (which disables writing)
Other than above	Setting prohibited (bus error returned)

**Note:**

- The value of all 32 bits of this register is needed to determine the lock status. Therefore, writing to this register must be made with 32-bit or 64-bit access. 8-bit or 16-bit access is not allowed.



## 5.2. TPU Lock Status Register (TPUn\_LST)

This register indicates the lock status of the timing protection unit.

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved							LST
ACCESS_TYPE	R0,WX							R,WX
PROT_TYPE	-							
INITIAL_VALUE	00000000							1

**[bit31:1] Reserved: Reserved bits**

**[bit0] LST: Lock status of the timing protection unit**

This bit indicates the lock status of the timing protection unit.

Value	Description
0	Unlocked state
1	Locked state

### 5.3. TPU Configuration Register (TPUn\_CFG)

This register makes the setting of the global prescaler and enables/disables interrupts for the timing protection unit. This register also makes the setting of enable/disable of the debug mode.

BITS	31	30	29	28	27	26	25	24
BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							DBGE
ACCESS_TYPE	R0,WX							R/W
PROT_TYPE	WPS							
INITIAL_VALUE	0000000							0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	GLBPSE	Reserved	GLBPS					
ACCESS_TYPE	R/W	R0,WX	R/W					
PROT_TYPE	WPS							
INITIAL_VALUE	0	0	000000					

BITS	15	14	13	12	11	10	9	8
BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	WPS							
INITIAL_VALUE	00000000							

BITS	7	6	5	4	3	2	1	0
BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved							INTE
ACCESS_TYPE	R0,WX							R/W
PROT_TYPE	WPS							
INITIAL_VALUE	0000000							0

**[bit31:25] Reserved: Reserved bits**

**[bit24] DBGE: Debug mode enable/disable setting bit**

This bit sets enable/disable of the debug mode.

Value	Description
0	Disable debug mode
1	Enable debug mode

**[bit23] GLBPSE: Global prescaler enable/disable setting bit**

This bit sets enable/disable of the global prescaler. If this bit is set to disable, the global prescaler counter stops after the next increment pulse. All timers also stop as a result.

Value	Description
0	Disable global prescaler
1	Enable global prescaler

**[bit22] Reserved: Reserved bit**



**[bit21:16] GLBPS[5:0]: Global prescaler division setting bit**

This bit sets the division ratio of the global prescaler. The input system clock is divided according to the configured division ratio, generating a clock of a lower frequency. A divided clock is provided to each timer.

Value	Description
000000	1/1
000001	1/2
000010	1/3
-	-
-	-
-	-
111111	1/64

**[bit15:1] Reserved: Reserved bits**

**[bit0] INTE: Timing protection unit interrupt enable/disable setting bit**

This bit makes the enable/disable setting of interrupts of the timing protection unit. The enable setting allows generation of interrupts to CPU. The disable setting prohibits generation of interrupts to CPU.

Value	Description
0	Disable interrupts of the timing protection unit
1	Enable interrupts of the timing protection unit

**Notes:**

- For details of the debug mode, see Section "3. Operation".
- Interrupts to CPU are generated when  $TPUn\_CFG:INTE="1"$  and  $TPUn\_TIR:IRm="1"$  and  $TPUn\_TIE:IEm="1"$ .
- It is prohibited that carries out timer starting by  $TPU0\_CFG:GLBPS="0b000000"$  and  $TPU0\_TCN1m:PS="0b00."$

## 5.4. TPU Timer Interrupt Request Register (TPUn\_TIR)

This register indicates the interrupt request status of each timer. Each timer is assigned with 1 bit, and the bit position indicates the timer number "m".

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	IR7	IR6	IR5	IR4	IR3	IR2	IR1	IR0
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

**[bit31:8] Reserved: Reserved bits**

**[bit7] IR7: Timer 7 interrupt request bit**

**[bit6] IR6: Timer 6 interrupt request bit**

**[bit5] IR5: Timer 5 interrupt request bit**

**[bit4] IR4: Timer 4 interrupt request bit**

**[bit3] IR3: Timer 3 interrupt request bit**

**[bit2] IR2: Timer 2 interrupt request bit**

**[bit1] IR1: Timer 1 interrupt request bit**

**[bit0] IR0: Timer 0 interrupt request bit**

These bits indicate the interrupt request status of individual timers. Indication is made for each timer about whether the interrupt request flag is on or there is a retained interrupt request.

Value	Description
0	An interrupt request does not exist for timer m
1	An interrupt request exists for timer m

**Note:**

- An interrupt to CPU is generated when  $TPUn\_CFG:INTE="1"$  and  $TPUn\_TIR:IRm="1"$  and  $TPUn\_TIE:IEm="1"$ .





## 5.5. TPU Timer Status Register (TPUn\_TST)

This register indicates the operation status of each timer. Each timer is assigned 1 bit, and the bit position indicates the timer number "m".

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	ST7	ST6	ST5	ST4	ST3	ST2	ST1	ST0
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

**[bit31:8] Reserved: Reserved bits**

**[bit7] ST7: Timer 7 status bit**

**[bit6] ST6: Timer 6 status bit**

**[bit5] ST5: Timer 5 status bit**

**[bit4] ST4: Timer 4 status bit**

**[bit3] ST3: Timer 3 status bit**

**[bit2] ST2: Timer 2 status bit**

**[bit1] ST1: Timer 1 status bit**

**[bit0] ST0: Timer 0 status bit**

These bits indicate the operation status of individual timers. Their indications show whether individual timers are operating or stopped.

Value	Description
0	Timer m is stopped
1	Timer m is operating

## 5.6. TPU Timer Interrupt Enable Register (TPUn\_TIE)

This register sets enable/disable of interrupt of individual timers. Each timer is assigned 1 bit, and the bit position indicates the timer number "m".

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	WP
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	IE7	IE6	IE5	IE4	IE3	IE2	IE1	IE0
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	0	0

**[bit31:8] Reserved: Reserved bits**

**[bit7] IE7: Timer 7 interrupt enable/disable setting bit**

**[bit6] IE6: Timer 6 interrupt enable/disable setting bit**

**[bit5] IE5: Timer 5 interrupt enable/disable setting bit**

**[bit4] IE4: Timer 4 interrupt enable/disable setting bit**

**[bit3] IE3: Timer 3 interrupt enable/disable setting bit**

**[bit2] IE2: Timer 2 interrupt enable/disable setting bit**

**[bit1] IE1: Timer 1 interrupt enable/disable setting bit**

**[bit0] IE0: Timer 0 interrupt enable/disable setting bit**

These bits set enable/disable of interrupts of individual timers. If these bits are enabled, interrupt requests of corresponding timers generate interrupts to CPU. If these bits are disabled, interrupt requests of corresponding timers do not generate interrupts to CPU.

Value	Description
0	Disable interrupt of timer m
1	Enable interrupt of timer m

**Note:**

- An interrupt to CPU is generated when *TPUn\_CFG:INTE*="1" and *TPUn\_TIR:IRm*="1" and *TPUn\_TIE:IEm*="1".



## 5.7. TPU Timer m Control Register 0 (TPUn\_TCN0m)

This register controls operation of individual timers. This register controls start, stop, and restart of timers, and controls timer interrupts. This register also sets the end counts and the preload values of timers.

Registers of the same type are provided for individual timers, and "m" in the abbreviated register name indicates the timer number m (0 to 7).

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	START	STOP	CONT	IES	IEC	IRC	Reserved	
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,WX	
PROT_TYPE	WP							
INITIAL_VALUE	0	0	0	0	0	0	00	

BIT_OFFSET	23-0
BIT_NAME	ECPL
ACCESS_TYPE	R/W
PROT_TYPE	WP
INITIAL_VALUE	00000000_00000000_00000000

### [bit31] START: Timer operation start bit

Writing "1" in this bit resets the timer and starts its operation.

Value	Description
0	Invalid (no effect on operation)
1	Start timer operation

### [bit30] STOP: Timer operation stop bit

Writing "1" in this bit stops the timer operation. It does not reset the timer.

Value	Description
0	Invalid (no effect on operation)
1	Stop timer operation

### [bit29] CONT: Timer operation restart bit

Writing "1" in this bit restarts the timer from the previously stalled state.

Value	Description
0	Invalid (no effect on operation)
1	Restart timer operation

### [bit28] IES: Timer interrupt enable set bit

Writing "1" in this bit sets the corresponding bit in the timer interrupt enable register. "1" is written in TPU<sub>n</sub>\_TIE:IE<sub>m</sub>.

Value	Description
0	Invalid (no effect on operation)
1	Enable timer interrupt

**[bit27] IEC: Timer interrupt enable clear bit**

Writing "1" in this bit clears the corresponding bit in the timer interrupt enable register. "0" is written in TPU<sub>n</sub>\_TIE:IE<sub>m</sub>.

Value	Description
0	Invalid (no effect on operation)
1	Disable timer interrupt

**[bit26] IRC: Timer interrupt request clear bit**

Writing "1" in this bit clears the corresponding bit in the timer interrupt request enable register. TPU<sub>n</sub>\_TIR:IR<sub>m</sub> is cleared to "0".

Value	Description
0	Invalid (no effect on operation)
1	Clear timer interrupt flag

**[bit25:24] Reserved: Reserved bits**

**[bit23:0] ECPL[23:0]: End count / preload value setting bit**

The value of these bits is used as the end count in the normal operation mode, and as the preload value when the operation mode is overflow mode and the preload function is enabled.

**Notes:**

- When multiple bits among *START*, *STOP*, and *CONT* are set to "1" simultaneously, the priority order is *START* > *CONT* > *STOP*.
- For details on timer operation modes, end counts, and preload values, see Section "3. Operation".
- TPU<sub>n</sub>\_TCN00 to TPU<sub>n</sub>\_TCN07 share the same register bit configuration.



## 5.8. TPU Timer m Control Register 1 (TPUn\_TCN1m)

This register controls operation of individual timers. This register configures the timer operation mode and individual prescalers. Registers of the same type are provided for individual timers, and "m" in the abbreviated register name indicates the timer number m (0 to 7).

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	WPS
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved			PL	FRT	TMOD	PS	
ACCESS_TYPE	R0,WX			R/W	R/W	R/W	R/W	
PROT_TYPE	WPS							
INITIAL_VALUE	000			0	0	0	00	

**[bit31:5] Reserved: Reserved bits**

### **[bit4] PL: Preload function enable/disable setting bit**

This bit sets enable/disable of the preload function. The preload function is effective only when the timer operation mode is the overflow mode.

Value	Description
0	Disable preload function
1	Enable preload function

### **[bit3] FRT: Free-run function enable/disable setting bit**

This bit sets enable/disable of the free-run function.

Value	Description
0	Disable free-run function
1	Enable free-run function

### **[bit2] TMOD: Timer operation mode setting bit**

This bit sets the operation mode of the timer.

Value	Description
0	Normal mode
1	Overflow mode

**[bit1:0] PS[1:0]: Individual prescaler division setting bit**

This bit sets the division ratio of the individual prescaler. An individual prescaler takes the output clock of the global prescaler, and divides it according to the configured division ratio to generate a clock of a timer-specific frequency.

Value	Description
00	Divided by 1
01	Divided by 2
10	Divided by 4
11	Divided by 16

**Notes:**

- For details on timer operation modes, the preload function and the free-run function, see Section "3. Operation".
- *TPUn\_TCN10 to TPUn\_TCN17 share the same register bit configuration.*



## 5.9. TPU Timer m Current Count Value Register (TPUn\_TCCm)

This register indicates the current count value of each timer. Registers of the same type are provided for individual timers, and "m" in the abbreviated register name indicates the timer number m (0 to 7).

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

BIT_OFFSET	23-0
BIT_NAME	TCC
ACCESS_TYPE	R,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000

**[bit31:24] Reserved: Reserved bits**

**[bit23:0] TCC[23:0]: Timer current count value bits**

These bits indicate the current timer count value. They can be read at any time. However, because the timer does not stop and continues counting when the value is read, the value being read may become different from the current timer count value by the time CPU gets the value.

**Note:**

- *TPUn\_TCC0 to TPUn\_TCC7 share the same register bit configuration.*

## 6. Others

This section explains the notes when using the timing protection unit.

An access to the register of the timing protection unit returns a bus error in the following conditions:

- Sequence protection violation against TPU<sub>n</sub>\_UNLOCK
- Writing in TPU<sub>n</sub>\_UNLOCK a value other than that of lock release or locking.
- Privilege protection error
- Read or write attempt in which access is made only to a register undefined area
- Write attempt in which access is made only to bits of WX attribute (including register undefined area)







## CHAPTER 25: Inter-processor Communication

This chapter explains inter-processor communication.

---

1. Overview
2. Configuration
3. Operation
4. Setting Procedure Example
5. Registers
6. Other



## 1. Overview

This section provides an overview of inter-processor communication.

In the case of multi-CPU configuration, it is necessary for a CPU to issue an interrupt to another CPU. To meet this requirement, an inter-processor communication unit is installed to provide support by hardware.

The inter-processor communication unit performs interrupt processing between CPUs by using the mailbox. 8 mailboxes are installed and interrupt processing of 8 systems can be performed simultaneously. Since the mailboxes are shared among the CPUs, interrupt processing for 8 systems can efficiently be used.

The inter-processor communication unit supports 3 types of operation modes.

The inter-processor communication unit supports 2 types of interrupts, request and acknowledgment. An ordinary interrupt is called a request. If a CPU received a request from another CPU, and the former sends the latter an interrupt to acknowledge the receipt of the request, this interrupt is called an acknowledgment.

The inter-processor communication unit has the request transmission masking function and inter-CPU data handover function.

Each mailbox has a register set for performing settings and status management, and is installed with the AHB 64-bit slave interface for accessing the registers.

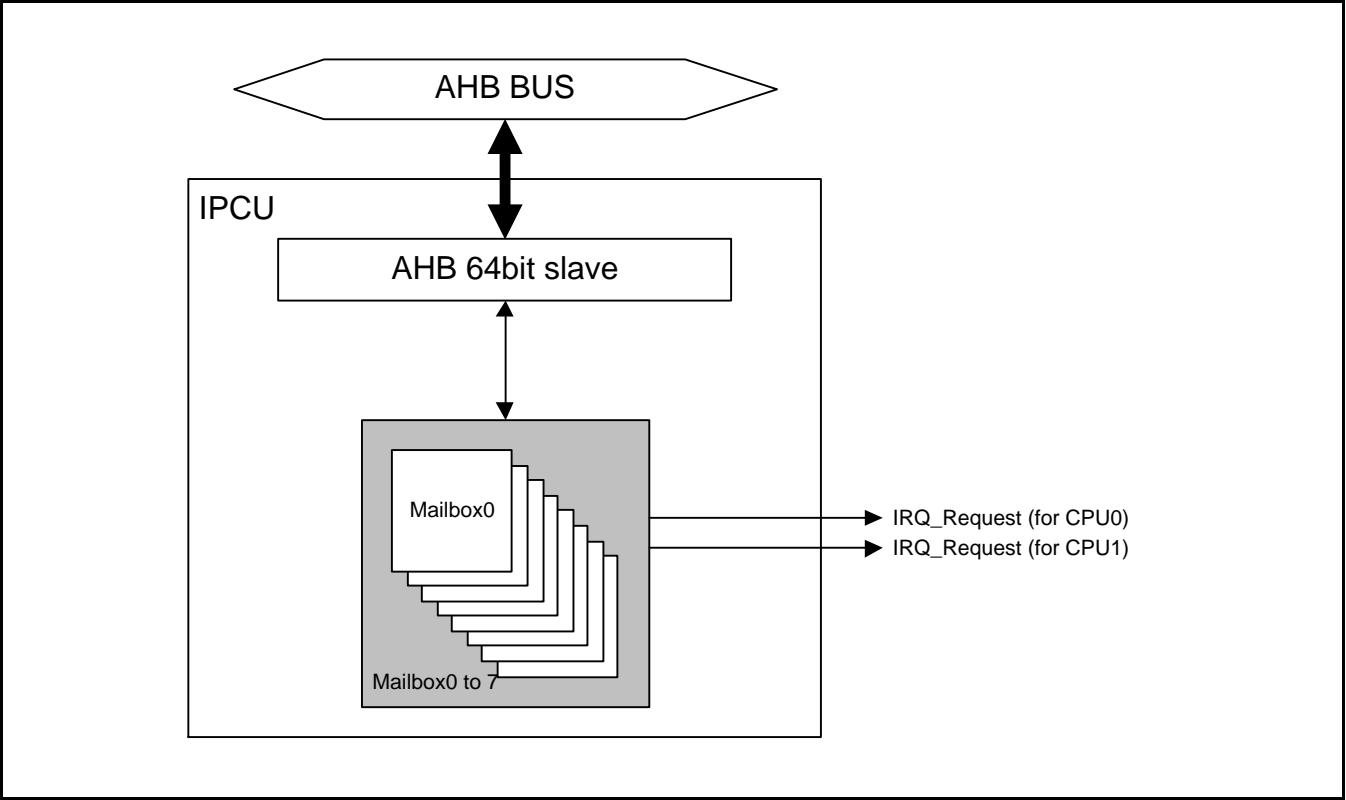
### Notes:

- This chapter refers to the inter-processor communication unit as the IPCU, which is its abbreviation, in some places.
- In the abbreviated names of the registers to be used by the inter-processor communication unit such as "IPCU\_\*\*\*n", "IPCU\_MBm\*\*\*", and "IPCU\_MBm\*\*\*.\*\*n", "n" represents the CPU number and "m" represents the mailbox number.

2. Configuration

This section explains a block diagram of the inter-processor communication unit.

Figure 2-1 Block Diagram of Inter-processor Communication Unit





### 3. Operation

This section describes the operation of inter-processor communication unit.

#### (1) Occupying and Releasing a Mailbox

A CPU that sends a request uses a mailbox to send the request. Read IPCU\_MBSTR to check the use status of the mailboxes. If a mailbox is available, write "1" in the relevant bit in the request sender setting register (IPCU\_MBmSRCR:SRCn) in the register set of the mailbox. If the writing is executed correctly, the occupation of the mailbox is completed and it becomes ready to be used. After the completion of request transmission and acknowledgment reception (if needed), writing "0x00000000" in IPCU\_MBmSRCR releases the mailbox. Access to the register set of the mailbox being occupied is restricted to the CPU that is the request sender and has occupied the mailbox and the CPU that is set as the request recipient. For access restrictions of each register, see Section "5. Registers" and "6. Other".

#### Sequence Protection of the Register Set of a Mailbox

Registers included in the register set of a mailbox cannot be written when no CPU is occupying the mailbox. In addition, when a mailbox is released, all the registers are cleared.

#### (2) Operation Mode

The inter-processor communication unit can select an operation mode from the 3 operation modes for each mailbox.

##### a) Manual Mode

In this mode, to clear a request after receiving the request, the request recipient CPU writes "1" in the relevant bit in the request recipient clear register (IPCU\_MBmDCR:DSTCn). Then, this CPU writes "1" in the relevant bit in the acknowledgment set register (IPCU\_MBmASR:ACKSn) to set an acknowledgment, thus sending the acknowledgment to the request sender CPU.

##### b) Automatic Acknowledgment Mode

In this mode, to clear a request after receiving the request, the request recipient CPU writes "1" in IPCU\_MBmDCR:DSTCn. This action of the CPU automatically sends an acknowledgment to the request sender CPU.

##### c) Automatic Clear Mode

In this mode, to clear a request after receiving the request, the request recipient CPU writes "1" in IPCU\_MBmDCR:DSTCn. This action of the CPU clears all the registers included in the register set of the mailbox being used and automatically releases the mailbox. The acknowledgment is not sent.

#### (3) Request Transmission Masking Function

This function is used to mask request transmission individually for request recipient CPUs. When the same request is to be sent to multiple CPUs, you can insert time delays between request transmission timings. To do so, mask some recipient CPUs before starting the request transmission, and later unmask them.

#### (4) Inter-CPU Data Handover Function

Every mailbox is equipped with 9 32-bit readable and writable registers, which are used for data handover between CPUs. This function is used in the following 2 cases: 1) the request sender CPU writes data to be handed over in these registers and then sends the request, and the request recipient CPU reads the data, and 2) the request recipient CPU writes data in these registers and then sends an acknowledgment, and the request sender CPU reads the data after receiving the acknowledgment.

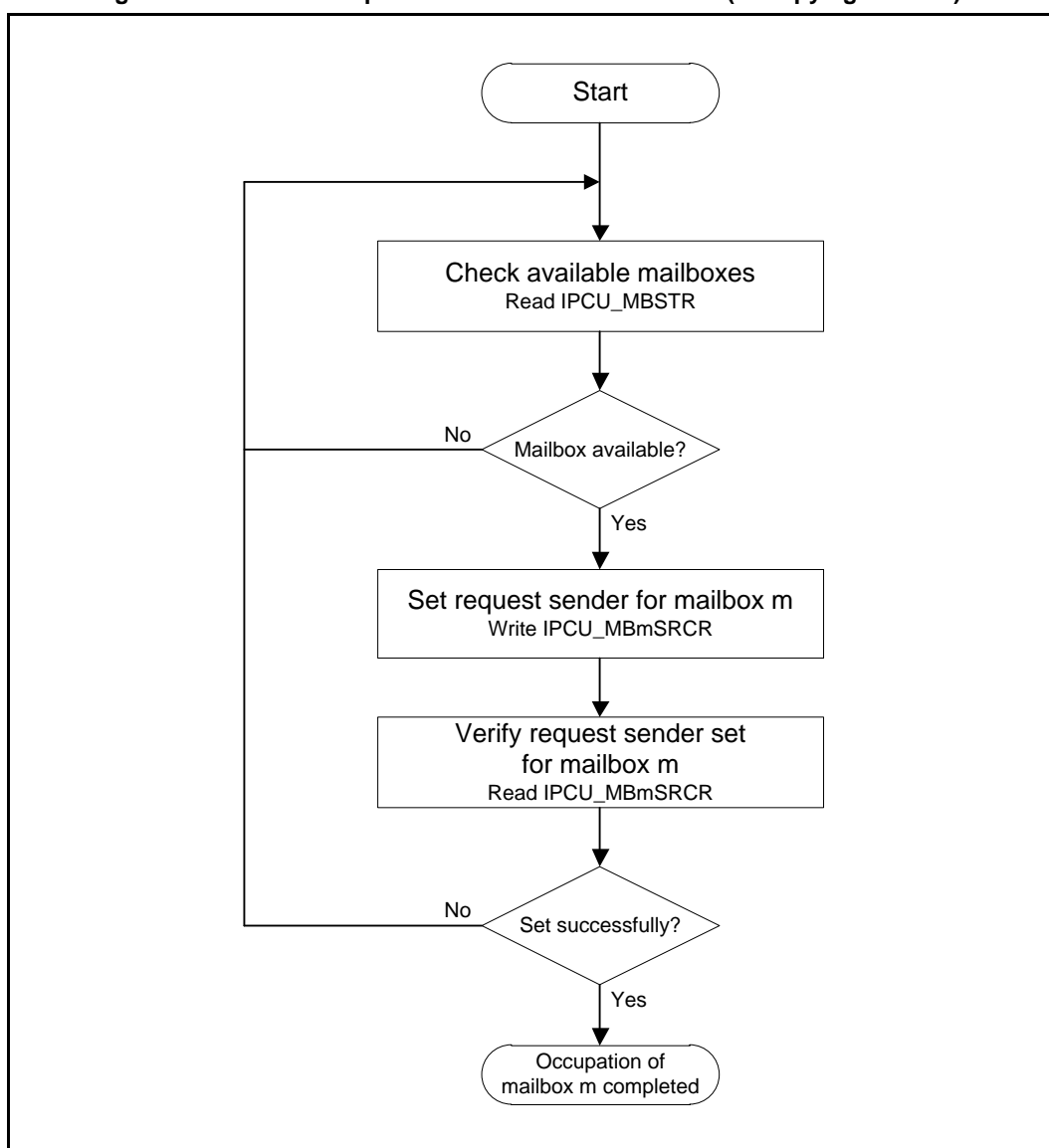
## 4. Setting Procedure Example

This section explains the setting procedure examples of the inter-processor communication unit.

### (1) Occupying a Mailbox

During the period from when the request sender CPU reads the status register of mailboxes (IPCU\_MBSTR) to when it checks available mailboxes and writes IPCU\_MBmSRCR to occupy a mailbox, another CPU may have already written IPCU\_MBmSRCR. Therefore, after write operation of IPCU\_MBmSRCR, it is necessary to read IPCU\_MBmSRCR and verify the validity of the write operation by the own CPU.

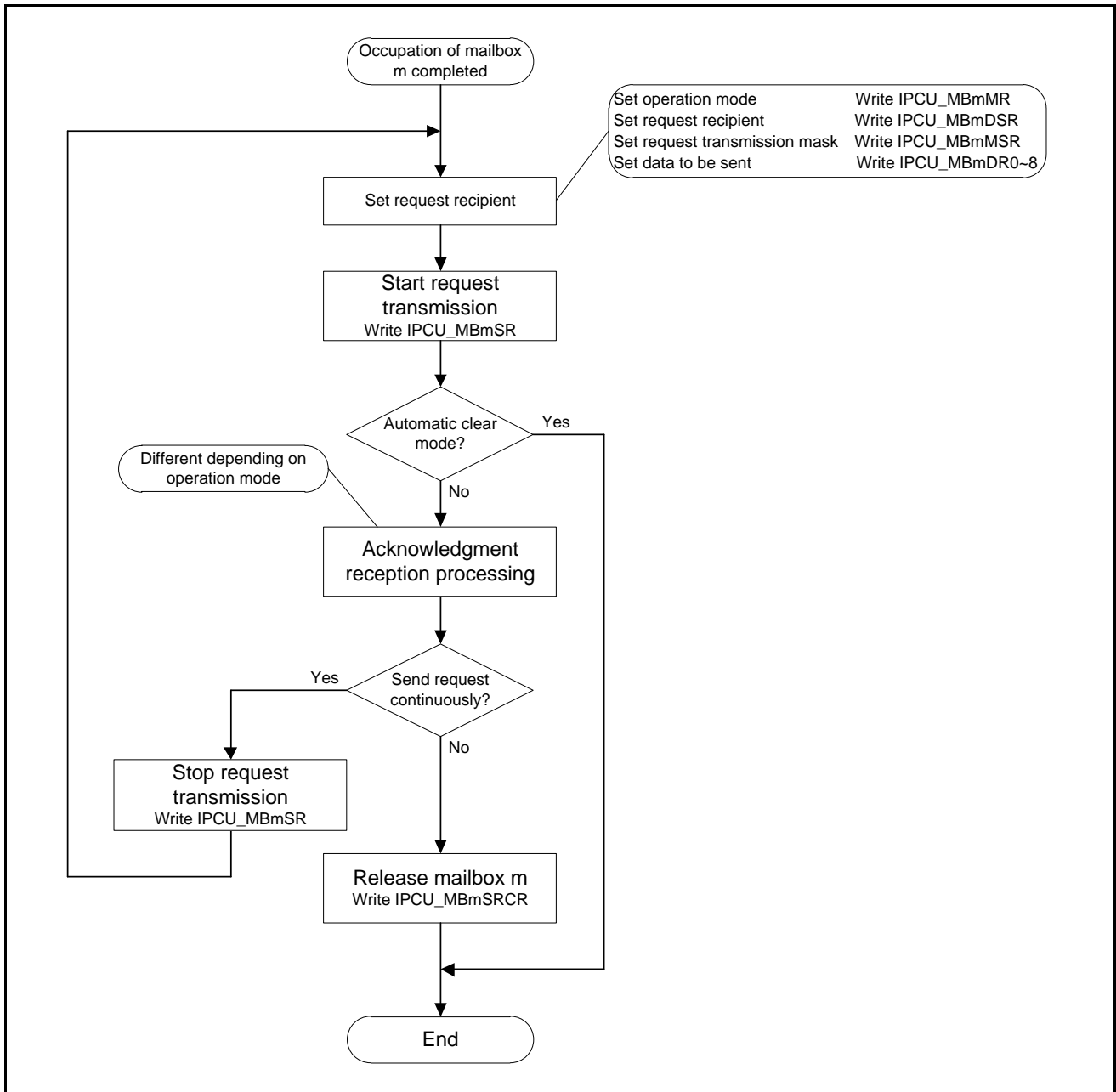
Figure 4-1 Flow of Inter-processor Communication Unit (Occupying Mailbox)



## (2) Sending Request and Releasing Mailbox

After having occupied a mailbox, the request sender CPU makes setting for various registers for request transmission and writes "1" to the request transmission bit in the request transmission register (IPCU\_MBmSR:SEND), thereby starting request transmission. In a mode other than automatic clear mode, since an acknowledgment is sent from the request recipient CPU, receive the acknowledgment and perform processing according to each operation mode. After completing the processing for receiving the acknowledgments from all of the acknowledgment senders, you complete the processing regarding the request. If you are continuously to send another request, do not release the mailbox but first write "0" in IPCU\_MBmSR:SEND to stop sending. After making setting of registers again, write "1" in IPCU\_MBmSR:SEND this time to start request transmission. To end request transmission, clear IPCU\_MBmSRCR to release the mailbox. Releasing the mailbox automatically clears all the registers included in the register set of the mailbox, and you do not need to write "0" to IPCU\_MBmSR:SEND.

Figure 4-2 Setting Flow of Inter-processor Communication Unit (Sending Request and Releasing Mailbox)



### (3) Request Reception Processing and Acknowledgment Reception Processing

A CPU that has received a request clears the request, and sends the acknowledgment according to the operation mode. The request sender CPU that has received the acknowledgment clears the acknowledgment. The setting procedure consisting of a series of steps varies depending on the operation mode.

The following explains example cases in which CPU0 uses mailbox 0 to send a request to CPU1.

#### a) Manual Mode

1. CPU1 detects an interrupt.
2. Check the number of mailbox used for sending the request detected by CPU1 and the type of the interrupt (request or acknowledgment). For this checking, there are the following 2 methods.
3. Read the interrupt status register (IPCU\_ISTR1). The fact that IPCU\_ISTR1:IST0 is "1" indicates that the mailbox 0 has generated the interrupt. Then, read IPCU\_MB0SRCR to check the request sender CPU. The fact that the own CPU has not sent the interrupt indicates that the type of the interrupt is a request.
4. Read the mailbox address register (IPCU\_MAR1). The fact that IPCU\_MAR1:MAR is "0x100" indicates that mailbox 0 has generated the interrupt. In addition, the fact that IPCU\_MAR1:REQ is "1" indicates that the type of the interrupt is a request.
5. CPU1 reads the data registers 0 to 8 (IPCU\_MB0DR0 to 8) to receive the data from CPU0.
6. CPU1 writes "1" to IPCU\_MB0DCR:DSTC1. This clears the request sent from CPU0.
7. CPU1 writes "1" to IPCU\_MB0ASR:ACKS1. This sends an acknowledgment to CPU0.
8. CPU0 detects the interrupt.
9. Check the number of mailbox used for sending the request detected by CPU0 and the type of the interrupt (request or acknowledgment). Similarly to step 2, for this checking, there are the following 2 methods.
10. Read IPCU\_ISTR0. The fact that IPCU\_ISTR0:IST0 is "1" indicates that the mailbox 0 has generated the interrupt. Then, read IPCU\_MB0SRCR to check the request sender CPU. The fact that the own CPU has sent the interrupt indicates that the type of the interrupt is an acknowledgment.
11. Read IPCU\_MAR0. The fact that IPCU\_MAR0:MAR is "0x100" indicates that mailbox 0 has generated the interrupt. In addition, the fact that IPCU\_MAR0:ACK is "1" indicates that the type of the interrupt is an acknowledgment.
12. CPU0 reads the acknowledgment status register (IPCU\_MB0ASTR). The fact that IPCU\_MB0ASTR:ACKST1 is "1" indicates that CPU1 has sent the acknowledgment.
13. CPU0 writes "1" to the relevant bit in the acknowledgment clear register (IPCU\_MB0ACR:ACKC1). This clears the acknowledgment sent from CPU1.

#### b) Automatic Acknowledgment Mode

1. CPU1 detects an interrupt.
2. Check the number of mailbox used for sending the request detected by CPU1 and the type of the interrupt (request or acknowledgment). For this checking, there are the following 2 methods.
3. Read IPCU\_ISTR1. The fact that IPCU\_ISTR1:IST0 is "1" indicates that the mailbox 0 has generated the interrupt. Then, read IPCU\_MB0SRCR to check the request sender CPU. The fact that the own CPU has not sent the interrupt indicates that the type of the interrupt is a request.
4. Read IPCU\_MAR1. The fact that IPCU\_MAR1:MAR is "0x100" indicates that mailbox 0 has generated the interrupt. In addition, the fact that IPCU\_MAR1:REQ is "1" indicates that the type of the interrupt is a request.
5. CPU1 reads IPCU\_MB0DR0 to 8 to receive the data from CPU0.





6. CPU1 writes "1" to IPCU\_MB0DCR:DSTC1. This clears the request sent from CPU0, and at the same time, IPCU\_MB0ASTR:ACKST1 becomes "1" and an acknowledgment is sent to CPU0.
7. CPU0 detects the interrupt.
8. Check the number of mailbox used for sending the request detected by CPU0 and the type of the interrupt (request or acknowledgment). Similarly to step 2, for this checking, there are the following 2 methods.
9. Read IPCU\_ISTR0. The fact that IPCU\_ISTR0:IST0 is "1" indicates that the mailbox 0 has generated the interrupt. Then, read IPCU\_SRC\_MB0 to check the request sender CPU. The fact that the own CPU has sent the interrupt indicates that the type of the interrupt is an acknowledgment.
10. Read IPCU\_MAR0. The fact that IPCU\_MAR0:MAR is "0x100" indicates that mailbox 0 has generated the interrupt. In addition, the fact that IPCU\_MAR1:ACK is "1" indicates that the type of the interrupt is an acknowledgment.
11. CPU0 reads IPCU\_MB0ASTR. The fact that IPCU\_MB0ASTR:ACKST1 is "1" indicates that CPU1 has sent the acknowledgment.
12. CPU0 writes "1" to IPCU\_MB0ACR:ACKC1. This clears the acknowledgment sent from CPU1.

#### c) Automatic Clear Mode

1. CPU1 detects an interrupt.
2. Check the number of mailbox used for sending the request detected by CPU1 and the type of the interrupt (request or acknowledgment). For this checking, there are the following 2 methods.
3. Read IPCU\_ISTR1. The fact that IPCU\_ISTR1:IST0 is "1" indicates that the mailbox 0 has generated the interrupt. Then, read IPCU\_MB0SRCR to check the request sender CPU. The fact that the own CPU has not sent the interrupt indicates that the type of the interrupt is a request.
4. Read IPCU\_MAR1. The fact that IPCU\_MAR1:MAR is "0x100" indicates that mailbox 0 has generated the interrupt. In addition, the fact that IPCU\_MAR1:REQ is "1" indicates that the type of the interrupt is a request.
5. CPU1 reads IPCU\_MB0DR0 to 8 to receive the data from CPU0.
6. CPU1 writes "1" to IPCU\_MB0DCR:DSTC1. This clears the request sent from CPU0. At the same time, all the registers included in the register set of the mailbox are cleared, and the mailbox is released.

## 5. Registers

This section explains the registers used in the inter-processor communication unit.

**Table 5-1 List of Inter-processor Communication Unit Registers**

Abbreviated Register Name	Register Name	See
IPCU_ISTRn	IPCU Interrupt Status Register	5.1
IPCU_MARn	IPCU Mailbox Address Register	5.2
IPCU_MBmSRCR	IPCU Mailbox m Request Sender Setting Register	5.3
IPCU_MBmMR	IPCU Mailbox m Operation Mode Setting Register	5.4
IPCU_MBmSR	IPCU Mailbox m Request Send Register	5.5
IPCU_MBmDSR	IPCU Mailbox m Request Recipient Set Register	5.6
IPCU_MBmDCR	IPCU Mailbox m Request Recipient clear Register	5.7
IPCU_MBmDSTR	IPCU Mailbox m Request Recipient Status Register	5.8
IPCU_MBmMSR	IPCU Mailbox m Request Transmission Mask Set Register	5.9
IPCU_MBmMCR	IPCU Mailbox m Request Transmission Mask Clear Register	5.10
IPCU_MBmMSTR	IPCU Mailbox m Request Transmission Mask Status Register	5.11
IPCU_MBmASR	IPCU Mailbox m Acknowledgment Set Register	5.12
IPCU_MBmACR	IPCU Mailbox m Acknowledgment Clear Register	5.13
IPCU_MBmASTR	IPCU Mailbox m Acknowledgment Status Register	5.14
IPCU_MBmASRCR	IPCU Mailbox m Acknowledgment Sender Status Register	5.15
IPCU_MBmDR0 to 8	IPCU Mailbox m Data Registers 0 to 8	5.16
IPCU_MBSTR	IPCU Mailbox Status Register	5.17



## 5.1. IPCU Interrupt Status Register (IPCU\_ISTRn)

This register indicates the status of the interrupt received by each CPU. The same type of register is installed for each CPU. The "n" in the abbreviated register name corresponds to the CPU number, n (0 to 1). 1 bit is assigned per mailbox, and the bit location corresponds to the mailbox number, m (0 to 7).

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	IST7	IST6	IST5	IST4	IST3	IST2	IST1	IST0
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

**[bit31:8] Reserved: Reserved bits**

**[bit7] IST7: Mailbox 7 interrupt status bit**

**[bit6] IST6: Mailbox 6 interrupt status bit**

**[bit5] IST5: Mailbox 5 interrupt status bit**

**[bit4] IST4: Mailbox 4 interrupt status bit**

**[bit3] IST3: Mailbox 3 interrupt status bit**

**[bit2] IST2: Mailbox 2 interrupt status bit**

**[bit1] IST1: Mailbox 1 interrupt status bit**

**[bit0] IST0: Mailbox 0 interrupt status bit**

Each of these bits indicates which mailbox m has sent the interrupt received by CPU<sub>n</sub>. When multiple interrupts are received, the bits corresponding to all the mailboxes m that have sent the interrupts become "1".

Value	Description
0	No interrupt has occurred from mailbox m to CPU <sub>n</sub> .
1	An interrupt has occurred from mailbox m to CPU <sub>n</sub> .

**Note:**

- The bit configuration is the same for all of registers IPCU\_ISTR0 to IPCU\_ISTR1.

## 5.2. IPCU Mailbox Address Register (IPCU\_MARn)

This register indicates the status of the interrupt received by each CPU. It indicates the address of the mailbox that has sent the interrupt and whether the interrupt is a request or acknowledgment. The same type of register is installed for each CPU. The "n" in the abbreviated register name corresponds to the CPU number, n (0 to 1).

BIT_OFFSET	31-16
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved		ACK	REQ	MAR[11:8]			
ACCESS_TYPE	R0,WX		R,WX	R,WX	R,WX			
PROT_TYPE	-							
INITIAL_VALUE	00		0	0	0000			

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	MAR[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

**[bit31:14] Reserved: Reserved bits**

**[bit13] ACK: Acknowledgment indication bit**

This bit indicates whether the type of the reported interrupt is an acknowledgment.

Value	Description
0	The interrupt type is not an acknowledgment.
1	The interrupt type is an acknowledgment.

**[bit12] REQ: Request indication bit**

This bit indicates whether the type of the reported interrupt is a request.

Value	Description
0	The interrupt type is not a request.
1	The interrupt type is a request.



**[bit11:0] MAR[11:0]: Interrupt sending mailbox address indication bits**

These bits indicate the address (offset) of the mailbox that has sent the interrupt. The address of the mailbox means the address of the register that is located at the beginning in the register set of each mailbox (IPCU\_MBmSRCR). When multiple interrupts occur, the address of the first mailbox that sends an interrupt is indicated. When all the interrupts sent by the first mailbox are cleared, these bits indicate the address of the next mailbox that sends an interrupt.

Value	Description
0x000	No interrupt has occurred to CPU <sub>n</sub> .
0x100	An interrupt has occurred from mailbox 0 to CPU <sub>n</sub> .
0x180	An interrupt has occurred from mailbox 1 to CPU <sub>n</sub> .
0x200	An interrupt has occurred from mailbox 2 to CPU <sub>n</sub> .
0x280	An interrupt has occurred from mailbox 3 to CPU <sub>n</sub> .
0x300	An interrupt has occurred from mailbox 4 to CPU <sub>n</sub> .
0x380	An interrupt has occurred from mailbox 5 to CPU <sub>n</sub> .
0x400	An interrupt has occurred from mailbox 6 to CPU <sub>n</sub> .
0x480	An interrupt has occurred from mailbox 7 to CPU <sub>n</sub> .

**Notes:**

- ACK and REQ are not set to "1" at a time.
- The bit configuration is the same for all of registers IPCU\_MAR0 to IPCU\_MAR1.

### 5.3. IPCU Mailbox m Request Sender Setting Register (IPCU\_MBmSRCR)

This register is one of those included in the register set of each mailbox. The request sender CPU writes this register. This register sets a request sender of each mailbox. The same type of register is installed for each mailbox. The "m" in the abbreviated register name corresponds to the mailbox number, m (0 to 7). 1 bit is assigned per CPU, and the bit location corresponds to the CPU number, n (0 to 1).

BIT_OFFSET	31-16
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved							
ACCESS_TYPE	R/W0							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved						SRC1	SRC0
ACCESS_TYPE	R/W0						R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	000000						0	0

**[bit31:16] Reserved: Reserved bits**

**[bit15:2] Reserved: Reserved bits**

Writing a value other than "000000\_00000000" to these bits is prohibited. Normal operation is not guaranteed when a value other than "000000\_00000000" is written.



**[bit1] SRC1: CPU1 request sender setting bit**

**[bit0] SRC0: CPU0 request sender setting bit**

These bits set request senders of mailbox m. Writing "1" to SRCn causes CPU<sub>n</sub> to occupy mailbox m. After completion of use of mailbox m, when CPU<sub>n</sub> writes "0" to SRCn, mailbox m is released. Writing "1" to any bit other than the bit corresponding to the own CPU is prohibited. Normal operation is not guaranteed in case of prohibited writing.

Value	Description	
	Write	Read
0	Clear the request sender setting for CPU <sub>n</sub> .	CPU <sub>n</sub> is not a request sender.
1	Set CPU <sub>n</sub> as a request sender.	CPU <sub>n</sub> is a request sender.

**Notes:**

- When a value other than "0x00000000" is read from this register, which means that any CPU is using mailbox m, writing a value other than "0x\*\*\*\*0000" is invalid. If a value other than "0x\*\*\*\*0000" is written, the register is not updated. It is necessary to write "0x\*\*\*\*0000" to release mailbox m.
- Releasing mailbox m by writing "0x00000000" to this register clears all the registers included in the register set of mailbox m to their initial values.
- For details on usage and restrictions of this register, see Section "6. Other".
- The bit configuration is the same for all of registers IPCU\_MB0SRCR to IPCU\_MB7SRCR.

## 5.4. IPCU Mailbox m Operation Mode Setting Register (IPCU\_MBmMR)

This register is one of those included in the register set of each mailbox. The request sender CPU writes this register. This register sets an operation mode of each mailbox. The same type of register is installed for each mailbox. The "m" in the abbreviated register name corresponds to the mailbox number, m (0 to 7).

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	WS
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved					MODE		
ACCESS_TYPE	R0,WX					R/W		
PROT_TYPE	WS							
INITIAL_VALUE	00000					000		

**[bit31:3] Reserved: Reserved bits**

**[bit2:0] MODE: Operation mode setting bits**

These bits set an operation mode of mailbox m.

Value	Description
000	Manual mode
001	
010	Automatic acknowledgment mode
011	
100	Automatic clear mode
Other than above	Setting prohibited (when any value is set, normal operation is not guaranteed.)

**Notes:**

- In any of the following conditions, writing to this register becomes invalid and the register is not updated.
- IPCU\_MBmSRCR is "0x00000000". (No CPU is using a mailbox.)  
IPCU\_MBmSR is "0x00000001". (A request is being sent.)
- For details on each operation mode, see "3. Operation" and "4. Setting Procedure Example".
- The bit configuration is the same for all of registers IPCU\_MB0MR to IPCU\_MB7MR.





## 5.5. IPCU Mailbox m Request Send Register (IPCU\_MBmSR)

This register is one of those included in the register set of each mailbox. The request sender CPU writes this register. This register controls request transmission of each mailbox. The same type of register is installed for each mailbox. The "m" in the abbreviated register name corresponds to the mailbox number, m (0 to 7).

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved							SEND
ACCESS_TYPE	R0,WX							R/W
PROT_TYPE	WS							
INITIAL_VALUE	00000000							0

**[bit31:1] Reserved: Reserved bits**

**[bit0] SEND: Request send bit**

This bit serves as a trigger for request transmission of mailbox m.

Value	Description
0	Stop request transmission.
1	Start request transmission.

**Note:**

- In the following condition, writing to this register becomes invalid and the register is not updated.
- IPCU\_MBmSRCR is "0x00000000". (No CPU is using a mailbox.)
- For details on usage and restrictions of this register, see Section "6. Other".
- The bit configuration is the same for all of registers IPCU\_MB0SR to IPCU\_MB7SR.

## 5.6. IPCU Mailbox m Request Recipient Set Register (IPCU\_MBmDSR)

This register is one of those included in the register set of each mailbox. The request sender CPU writes this register. This register sets a request recipient for each mailbox. The same type of register is installed for each mailbox. The "m" in the abbreviated register name corresponds to the mailbox number, m (0 to 7). 1 bit is assigned per CPU, and the bit location corresponds to the CPU number, n (0 to 1).

BIT_OFFSET	31-16
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved							
ACCESS_TYPE	R0,W0							
PROT_TYPE	WS							
INITIAL_VALUE	00000000							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved						DSTS1	DSTS0
ACCESS_TYPE	R0,WX						R0,W	R0,W
PROT_TYPE	WS							
INITIAL_VALUE	000000						0	0

**[bit31:16] Reserved: Reserved bits**

**[bit15:2] Reserved: Reserved bits**

Writing a value other than "000000\_00000000" to these bits is prohibited. Normal operation is not guaranteed when a value other than "000000\_00000000" is written.

**[bit1] DSTS1: CPU1 request recipient set bit**

**[bit0] DSTS0: CPU0 request recipient set bit**

These bits set request recipients of mailbox m. Writing "1" to DSTSn sets CPU<sub>n</sub> as a request recipient of mailbox m. You can use these bits only for setting, which is not cleared by writing "0" to these bits. The actual recipient is checked from IPCU\_MBmDSTR. It is prohibited to write "1" to the bit corresponding to the own CPU to specify it as a recipient. Normal operation is not guaranteed in case of prohibited writing.

Value	Description
0	Invalid (no effect on operation)
1	Set CPU <sub>n</sub> as a request recipient of mailbox m.

**Notes:**

- In any of the following conditions, writing to this register becomes invalid and does not update the IPCU\_MBmDSTR status.
- IPCU\_MBmSRCR is "0x00000000". (No CPU is using a mailbox.)  
IPCU\_MBmSR is "0x00000001". (A request is being sent.)
- The bit configuration is the same for all of registers IPCU\_MB0DSTR to IPCU\_MB7DSTR.



## 5.7. IPCU Mailbox m Request Recipient Clear Register (IPCU\_MBmDCR)

This register is one of those included in the register set of each mailbox. The request recipient CPU writes this register. This register clears the request recipient for each mailbox. The same type of register is installed for each mailbox. The "m" in the abbreviated register name corresponds to the mailbox number, m (0 to 7). 1 bit is assigned per CPU, and the bit location corresponds to the CPU number, n (0 to 1).

BIT_OFFSET	31-16
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	WS
INITIAL_VALUE	00000000_00000000

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved							
ACCESS_TYPE	R0,W0							
PROT_TYPE	WS							
INITIAL_VALUE	00000000							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved						DSTC1	DSTC0
ACCESS_TYPE	R0,W0						R0,W	R0,W
PROT_TYPE	WS							
INITIAL_VALUE	000000						0	0

**[bit31:16] Reserved: Reserved bits**

**[bit15:2] Reserved: Reserved bits**

Writing a value other than "000000\_00000000" to these bits is prohibited. Normal operation is not guaranteed when a value other than "000000\_00000000" is written.

**[bit1] DSTC1: CPU1 request recipient clear bit**

**[bit0] DSTC0: CPU0 request recipient clear bit**

These bits clear request recipients of mailbox m. Writing "1" to DSTCn clears CPU<sub>n</sub> from the request recipients of mailbox m. You can use these bits only for clearing, and writing "0" to these bits has no effect on operation. The actual recipient status is checked from IPCU\_MBmDSTR. Writing "1" to any bit other than the bit corresponding to the own CPU is prohibited. Also, writing "1" to the bit corresponding to a CPU that has not been set as a recipient in the first place is prohibited. Normal operation is not guaranteed in case of prohibited writing.

Value	Description
0	Invalid (no effect on operation)
1	Clear CPU <sub>n</sub> from the request recipients of mailbox m.

**Notes:**

- In the following condition, writing to this register becomes invalid and does not update the IPCU\_DSTST\_MBm status.
- IPCU\_MBmSR is "0x00000000". (No CPU is using a mailbox.)
- The bit configuration is the same for all of registers IPCU\_MB0DCR to IPCU\_MB7DCR.

## 5.8. IPCU Mailbox m Request Recipient Status Register (IPCU\_MBmDSTR)

This register is one of those included in the register set of each mailbox. This register indicates the status of the request recipients for each mailbox. The same type of register is installed for each mailbox. The "m" in the abbreviated register name corresponds to the mailbox number, m (0 to 7). 1 bit is assigned per CPU, and the bit location corresponds to the CPU number, n (0 to 1).

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	WS
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved						DSTST1	DSTST0
ACCESS_TYPE	R0,WX						R0,W	R0,W
PROT_TYPE	WS							
INITIAL_VALUE	000000						0	0

**[bit31:2] Reserved: Reserved bits**

**[bit1] DSTST1: CPU1 request recipient status bit**

**[bit0] DSTST0: CPU0 request recipient status bit**

These bits indicate the status of request recipients of mailbox m. "1" of DSTSTn indicates that CPU<sub>n</sub> is a request recipient of mailbox m. In addition, when the request recipient CPU clears the request, the result of the clearing is also reflected.

Value	Description
0	CPU <sub>n</sub> is not a request recipient of mailbox m.
1	CPU <sub>n</sub> is a request recipient of mailbox m.

**Note:**

- The bit configuration is the same for all of registers IPCU\_MB0DSTR to IPCU\_MB7DSTR.



## 5.9. IPCU Mailbox m Request Transmission Mask Set Register (IPCU\_MBmMSR)

This register is one of those included in the register set of each mailbox. The request sender CPU writes this register. This register sets the request transmission mask for each mailbox. The same type of register is installed for each mailbox. The "m" in the abbreviated register name corresponds to the mailbox number, m (0 to 7). 1 bit is assigned per CPU, and the bit location corresponds to the CPU number, n (0 to 1).

BIT_OFFSET	31-16
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	WS
INITIAL_VALUE	00000000_00000000

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved							
ACCESS_TYPE	R0,W0							
PROT_TYPE	WS							
INITIAL_VALUE	00000000							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved						MSKS1	MSKS0
ACCESS_TYPE	R0,W0						R0,W	R0,W
PROT_TYPE	WS							
INITIAL_VALUE	000000						0	0

**[bit31:16] Reserved: Reserved bits**

**[bit15:2] Reserved: Reserved bits**

Writing a value other than "000000\_00000000" to these bits is prohibited. Normal operation is not guaranteed when a value other than "000000\_00000000" is written.

**[bit1] MSKS1: CPU1 request transmission mask set bit**

**[bit0] MSKS0: CPU0 request transmission mask set bit**

These bits set request transmission masks of mailbox m. Writing "1" to MSKSn sets the request transmission mask of mailbox m for CPU<sub>n</sub>. You can use these bits only for setting, which is not cleared by writing "0" to these bits. The actual transmission mask status is checked from IPCU\_MBmMSTR. Writing "1" to the bit corresponding to the own CPU is prohibited. Normal operation is not guaranteed in case of prohibited writing.

Value	Description
0	Invalid (no effect on operation)
1	Mask the request transmission of mailbox m for CPU <sub>n</sub> .

**Notes:**

- In any of the following conditions, writing to this register becomes invalid and does not update the IPCU\_MBmMSTR status.
- IPCU\_MBmSRCR is "0x00000000". (No CPU is using a mailbox.)
- IPCU\_MBmSR is "0x00000001". (A request is being sent.)
- For details on the masking function, see Section "3.Operation".
- The bit configuration is the same for all of registers IPCU\_MB0MSR to IPCU\_MB7MSR.



### 5.10. IPCU Mailbox m Request Transmission Mask Clear Register (IPCU\_MBmMCR)

This register is one of those included in the register set of each mailbox. The request sender CPU writes this register. This register clears the request transmission mask for each mailbox. The same type of register is installed for each mailbox. The "m" in the abbreviated register name corresponds to the mailbox number, m (0 to 7). 1 bit is assigned per CPU, and the bit location corresponds to the CPU number, n (0 to 1).

BIT_OFFSET	31-16
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	WS
INITIAL_VALUE	00000000_00000000

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved							
ACCESS_TYPE	R0,W0							
PROT_TYPE	WS							
INITIAL_VALUE	00000000							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved						MSKS1	MSKS0
ACCESS_TYPE	R0,W0						R0,W	R0,W
PROT_TYPE	WS							
INITIAL_VALUE	000000						0	0

**[bit31:16] Reserved: Reserved bits**

**[bit15:2] Reserved: Reserved bits**

Writing a value other than "000000\_00000000" to these bits is prohibited. Normal operation is not guaranteed when a value other than "000000\_00000000" is written.

**[bit1] MSKC1: CPU1 request transmission mask clear bit**

**[bit0] MSKC0: CPU0 request transmission mask clear bit**

These bits clear request transmission masks of mailbox m. Writing "1" to MSKCn clears the request transmission mask of mailbox m for CPU<sub>n</sub>. You can use these bits only for clearing, and writing "0" to these bits has no effect on operation. The actual transmission mask status is checked from IPCU\_MBmMSTR.

Value	Description
0	Invalid (no effect on operation)
1	Clear the request transmission mask of mailbox m for CPU <sub>n</sub> .

**Note:**

- In the following condition, writing to this register becomes invalid and does not update the IPCU\_MSKST\_MBm status.
- IPCU\_MBmSRCR is "0x00000000". (No CPU is using a mailbox.)
- For details on the masking function, see Section "3. Operation".
- The bit configuration is the same for all of registers IPCU\_MB0MCR to IPCU\_MB7MCR.





## 5.11. IPCU Mailbox m Request Transmission Mask Status Register (IPCU\_MBmMSTR)

This register is one of those included in the register set of each mailbox. This register indicates the status of the request transmission masks for each mailbox. The same type of register is installed for each mailbox. The "m" in the abbreviated register name corresponds to the mailbox number, m (0 to 7). 1 bit is assigned per CPU, and the bit location corresponds to the CPU number, n (0 to 1).

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved						MSKS1	MSKS0
ACCESS_TYPE	R0,WX						R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	000000						0	0

**[bit31:2] Reserved: Reserved bits**

**[bit1] MSKST1: CPU1 request transmission mask status bit**

**[bit0] MSKST0: CPU0 request transmission mask status bit**

These bits indicate the status of the request transmission masks of mailbox m. "1" of MSKSTn indicates that the request transmission of mailbox m for CPU n is masked.

Value	Description
0	The request transmission of mailbox m for CPU n is not masked.
1	The request transmission of mailbox m for CPU n is masked.

**Notes:**

- For details on the masking function, see Section "3. Operation".
- The bit configuration is the same for all of registers IPCU\_MB0MSTR to IPCU\_MB7MSTR.

## 5.12. IPCU Mailbox m Acknowledgment Set Register (IPCU\_MBmASR)

This register is one of those included in the register set of each mailbox. The request recipient CPU writes this register. Setting this register after clearing a request sent from each mailbox sends the acknowledgment. The same type of register is installed for each mailbox. The "m" in the abbreviated register name corresponds to the mailbox number, m (0 to 7). 1 bit is assigned per CPU, and the bit location corresponds to the CPU number, n (0 to 1).

BIT_OFFSET	31-16
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	WS
INITIAL_VALUE	00000000_00000000

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved							
ACCESS_TYPE	R0,W0							
PROT_TYPE	WS							
INITIAL_VALUE	00000000							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved						ACKS1	ACKS0
ACCESS_TYPE	R0,W0						R0,W	R0,W
PROT_TYPE	WS							
INITIAL_VALUE	000000						0	0

**[bit31:16] Reserved: Reserved bits**

**[bit15:2] Reserved: Reserved bits**

Writing a value other than "000000\_00000000" to these bits is prohibited. Normal operation is not guaranteed when a value other than "000000\_00000000" is written.



**[bit1] ACKS1: CPU1 acknowledgment set bit**

**[bit0] ACKS0: CPU0 acknowledgment set bit**

These bits set the acknowledgments for requests of mailbox m. Writing "1" to ACKSn sets the acknowledgment from CPU<sub>n</sub>. You can use these bits only for setting, which is not cleared by writing "0" to these bits. The actual acknowledgment status is checked from IPCU\_MBmASTR. Writing "1" to any bit other than the bit corresponding to the own CPU is prohibited. Also, writing "1" to any bit that has already been set is prohibited. Normal operation is not guaranteed in case of prohibited writing.

Value	Description
0	Invalid (no effect on operation)
1	Set the acknowledgment from CPU <sub>n</sub> for a request of mailbox m.

**Notes:**

- You can use this register only in manual mode.
- In any of the following conditions, writing to this register becomes invalid and does not update the IPCU\_MBmASTR status.
  - IPCU\_MBmSRCR is "0x00000000". (No CPU is using a mailbox.)
  - IPCU\_MBmSR is "0x00000000". (A request is not being sent.)
  - IPCU\_MBmMR is not "0x00000000" or "0x00000001". (The mode is not manual mode.)
- For details on the acknowledgment, see Section "3.Operation".
- The bit configuration is the same for all of registers IPCU\_MB0ASR to IPCU\_MB7ASR.

### 5.13. IPCU Mailbox m Acknowledgment Clear Register (IPCU\_MBmACR)

This register is one of those included in the register set of each mailbox. The request sender CPU that has received the acknowledgment for the request writes this register. This register clears acknowledgments of each mailbox. The same type of register is installed for each mailbox. The "m" in the abbreviated register name corresponds to the mailbox number, m (0 to 7). 1 bit is assigned per CPU, and the bit location corresponds to the CPU number, n (0 to 1).

BIT_OFFSET	31-16
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	WS
INITIAL_VALUE	00000000_00000000

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved							
ACCESS_TYPE	R0,W0							
PROT_TYPE	WS							
INITIAL_VALUE	00000000							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved						ACKC1	ACKC0
ACCESS_TYPE	R0,W0						R0,W	R0,W
PROT_TYPE	WS							
INITIAL_VALUE	000000						0	0

**[bit31:16] Reserved: Reserved bits**

**[bit15:2] Reserved: Reserved bits**

Writing a value other than "000000\_00000000" to these bits is prohibited. Normal operation is not guaranteed when a value other than "000000\_00000000" is written.



**[bit1] ACKC1: CPU1 acknowledgment clear bit**

**[bit0] ACKC0: CPU0 acknowledgment clear bit**

These bits clear the acknowledgments for requests of mailbox m. Writing "1" to ACKCn clears the acknowledgment sent from CPU<sub>n</sub>. You can use these bits only for clearing, and writing "0" to these bits has no effect on operation. The actual acknowledgment status is checked from IPCU\_MBmASTR.

Value	Description
0	Invalid (no effect on operation)
1	Clear the acknowledgment sent from CPU <sub>n</sub> for a request of mailbox m.

**Notes:**

- You cannot use this register in automatic clear mode.
- In any of the following conditions, writing to this register becomes invalid and does not update the IPCU\_MBmASTR status.
  - IPCU\_MBmSRCR is "0x00000000". (No CPU is using a mailbox.)
  - IPCU\_MBmSR is "0x00000000". (A request is not being sent.)
  - IPCU\_MBmMR is "0x00000004". (Automatic clear mode.)
- For details on the acknowledgment, see Section "3.Operation".
- The bit configuration is the same for all of registers IPCU\_MB0ACR to IPCU\_MB7ACR.

## 5.14. IPCU Mailbox m Acknowledgment Status Register (IPCU\_MBmASTR)

This register is one of those included in the register set of each mailbox. This register indicates the status of the acknowledgments of each mailbox. The same type of register is installed for each mailbox. The "m" in the abbreviated register name corresponds to the mailbox number, m (0 to 7). 1 bit is assigned per CPU, and the bit location corresponds to the CPU number, n (0 to 1).

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved						ACKST1	ACKST0
ACCESS_TYPE	R0,WX						R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	000000						0	0

**[bit31:2] Reserved: Reserved bits**

**[bit1] ACKST1: CPU1 acknowledgment status bit**

**[bit0] ACKST0: CPU0 acknowledgment status bit**

These bits indicate the status of acknowledgments of mailbox m. "1" of ACKSTn indicates that CPU<sub>n</sub> is an acknowledgment sender of mailbox m. In addition, when the request sender CPU clears the acknowledgment, the result of the clearing is also reflected.

Value	Description
0	CPU <sub>n</sub> is not an acknowledgment sender of mailbox m.
1	CPU <sub>n</sub> is an acknowledgment sender of mailbox m.

**Notes:**

- You cannot use this register in automatic clear mode. In automatic clear mode, the read value is always "0x00000000".
- The bit configuration is the same for all of registers IPCU\_MB0ASTR to IPCU\_MB7ASTR.



### 5.15. IPCU Mailbox m Acknowledgment Sender Status Register (IPCU\_MBmASRCR)

This register is one of those included in the register set of each mailbox. This register indicates the status of the acknowledgments of each mailbox. The same type of register is installed for each mailbox. The "m" in the abbreviated register name corresponds to the mailbox number, m (0 to 7). 1 bit is assigned per CPU, and the bit location corresponds to the CPU number, n (0 to 1).

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved						ACKSRC1	ACKSRC0
ACCESS_TYPE	R0,WX						R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	000000						0	0

**[bit31:2] Reserved: Reserved bits**

**[bit1] ACKSRC1: CPU1 acknowledgment sender status bit**

**[bit0] ACKSRC0: CPU0 acknowledgment sender status bit**

These bits indicate the status of acknowledgments of mailbox m. "1" of ACKSTn indicates that CPU<sub>n</sub> is an acknowledgment sender of mailbox m. 2 or more of these bits are not set to "1" at a time. When multiple acknowledgments have been sent, only the bit corresponding to the first CPU that has sent an acknowledgment becomes "1". After the request sender CPU has cleared the acknowledgment, only the bit corresponding to the next CPU that has sent an acknowledgment becomes "1".

Value	Description
0	CPU <sub>n</sub> is not an acknowledgment sender of mailbox m.
1	CPU <sub>n</sub> is an acknowledgment sender of mailbox m.

**Notes:**

- You can use this register only in manual mode or automatic acknowledgment mode.
- Unlike IPCU\_MBmASTR, which indicates all the acknowledgment senders, this register indicates only 1 sender at a time in the order of time from the oldest first.
- The bit configuration is the same for all of registers IPCU\_MB0ASRCR to IPCU\_MB7ASRCR.

## 5.16. IPCU Mailbox m Data Registers 0 to 8 (IPCU\_MBmDR0 to 8)

Each of these registers is one of those included in the register set of each mailbox. These registers store data to be handed over between CPUs. The same type of 9 registers are installed for each mailbox. The "m" in the abbreviated register name corresponds to the mailbox number, m (0 to 7) and the number at the end of the abbreviated register name corresponds to the data number (0 to 8).

BIT_OFFSET	31-0
BIT_NAME	DT
ACCESS_TYPE	R/W
PROT_TYPE	WS
INITIAL_VALUE	00000000_00000000_00000000_00000000

### [bit31:0] DT[31:0]: Data bits

These bits store data to be handed over between CPUs.

**Note:**

- In the following condition, writing to this register becomes invalid and the register is not updated.
- *IPCU\_MBmSRCR* is "0x00000000". (No CPU is using a mailbox.)
- The bit configuration is the same for all of registers *IPCU\_MB0DR0* to *IPCU\_MB7DR8*.





## 5.17. IPCU Mailbox Status Register (IPCU\_MBSTR)

This register indicates the status of whether each mailbox is available. 1 bit is assigned per mailbox, and the bit location corresponds to the mailbox number, m (0 to 7).

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	MBST7	MBST6	MBST5	MBST4	MBST3	MBST2	MBST1	MBST0
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:8] Reserved: Reserved bits

[bit7] MBST7: Mailbox 7 status

[bit6] MBST6: Mailbox 6 status

[bit5] MBST5: Mailbox 5 status

[bit4] MBST4: Mailbox 4 status

[bit3] MBST3: Mailbox 3 status

[bit2] MBST2: Mailbox 2 status

[bit1] MBST1: Mailbox 1 status

[bit0] MBST0: Mailbox 0 status

These bits indicate the status of each mailbox. "1" of MBSTm indicates that mailbox m is being used.

Value	Description
0	Mailbox m is available.
1	Mailbox m is being used by another CPU.

**Note:**

- During the period from when the own CPU reads this register to when it checks available mailboxes and writes IPCU\_MBmSRCR to occupy a mailbox, another CPU may have already written IPCU\_MBmSRCR. After writing IPCU\_MBmSRCR, read IPCU\_MBmSRCR and confirm that the own CPU has successfully occupied the mailbox.

## 6. Other

This section explains precautions on use of the inter-processor communication unit.

When a mailbox is occupied by a CPU, it is prohibited for any CPU that is neither the CPU that has occupied the mailbox nor one specified as the request recipient to write to the registers in the mailbox register set. In addition, the registers that can be written by the request sender CPU and those that can be written by the request recipient CPU are decided respectively. For details, see the explanation of each register in Section "5. Registers". Normal operation is not guaranteed in case of prohibited writing.

The inter-processor communication unit does not identify a master of register access (which CPU is accessing a register). Therefore, the hardware cannot provide protection against prohibited write access.

There is no condition in which a bus error is returned in accessing a register of the inter-processor communication unit.



# CHAPTER 26: Exclusive Access Memory (EAM)

This chapter provides an overview of the exclusive access memory (EAM) and describes its configuration, operation, and memory area.

---



1. Overview
2. Configuration
3. Operation
4. Operation Examples
5. Memory Area



## 1. Overview

This section describes the features of the exclusive access memory (EAM) block.

If a conflicting access occurs, Cortex™-R5F can exclusively access the memory area of this block by using the AXI exclusive command for the memory area.

### Features

- This block is located as a slave on the HPM, and this memory area can be accessed exclusively from Cortex-R5F.
- This block has 48 bytes of memory area that can be accessed exclusively.
- Exclusion processing by this block is positioned as the global monitor. (The local monitor is located on Cortex-R5F.) For details on the local monitor and the global monitor, see the "ARM® Architecture Reference Manual ARM®v7-A and ARM®v7-R edition (ARM DDI 0406B)".
- Up to 4 ID and address range sets can be monitored for exclusive access.
- The write data width is 64 bits.
- The read data width is 64 bits.
- The ID width is 8 bits.

### Note:

- *Though EAM can observe the range of the ID and address up to 4 sets at the same time, this product is 2 sets.*

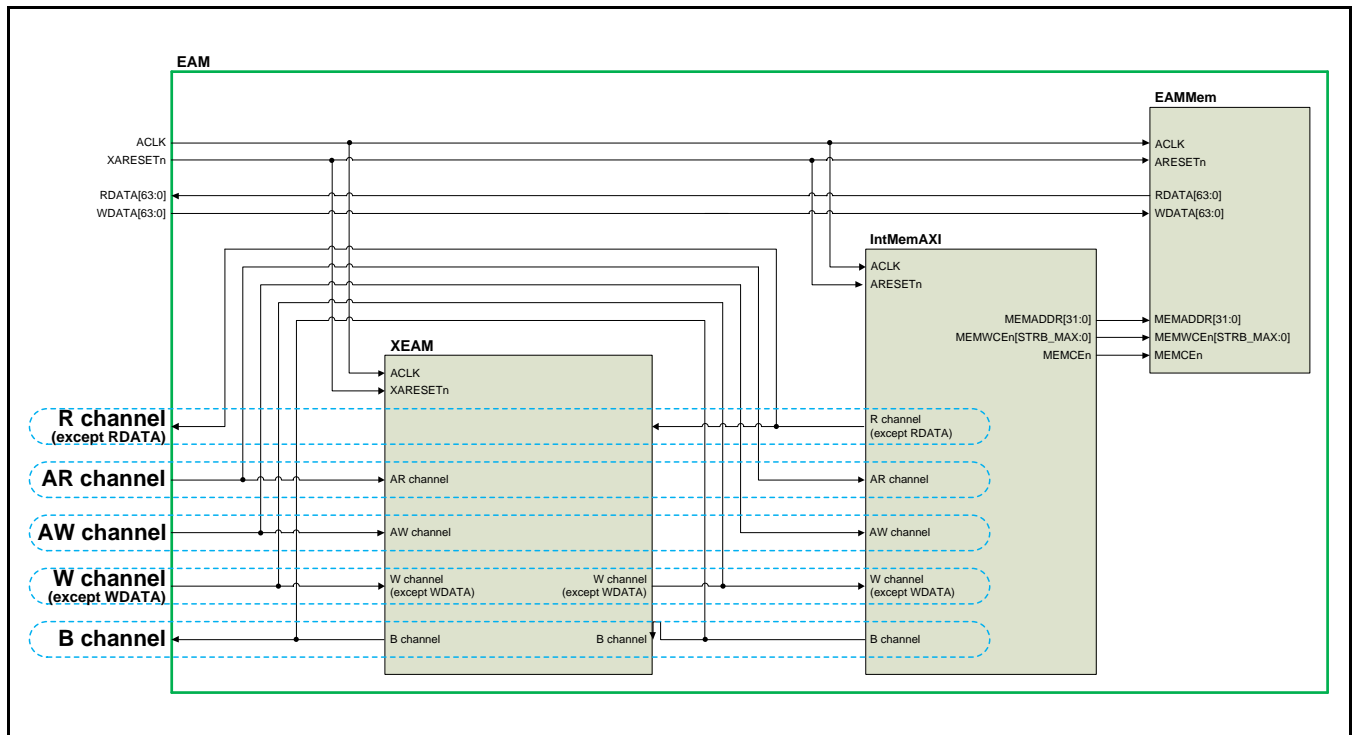
## 2. Configuration

This section shows the configuration of the exclusive access memory (EAM).

### 2.1. Block Diagram

This section shows a block diagram of the exclusive access memory (EAM).

Figure 2-1 Block Diagram of Exclusive Access Memory (EAM)



- XEAM  
Exclusive access processing is performed.  
Up to 4 ID and address range sets can be monitored simultaneously.
- IntMemAXI  
Conversion from the AXI interface to the memory interface is performed.
- EAMMem  
This block has a capacity of 48 bytes as a memory area, and reads and writes data using the memory interface.



### 3. Operation

This section describes the operation for each instruction.

The letter "x" in LDRx, STRx, LDREXx, STREXx, and so on in the commands described below represents the size of the command: "D" = Double-word, " " = Word, "H" = Half-word, and "B" = Byte. (Word has no suffix.)

### 3.1. General Aspects of Transfer

This section outlines the operation for normal instructions (such as STRx and LDRx) and exclusive instructions (such as STREXx and LDREXx).

- This block is located as a slave on the HPM, and receives and processes an exclusive access (in addition to normal access) from a master.
- The memory area in this block has a capacity of 48 bytes. This memory area supports exclusive access. Normal read (LDRx), normal write (STRx), exclusive load (LDREXx), and exclusive store (STREXx) can be performed.
- IntMemAXI in this block has no address buffer mounted. Therefore, it has no WID and processes both normal write (STRx) and exclusive store instruction (STREXx) without interleaving. It processes address and data pairs only one by one. So, consider a transfer method for a master.
- This block does not support AxCACHE and AxPROT.





## 3.2. Normal Access

This section describes the operation for normal instructions (such as STRx and LDRx).

### (1) Read Instruction (LDRx)

- This instruction reads the relevant address in EAMMem according to a read instruction (LDRx) from a master.

### (2) Write Instruction (STRx)

- This instruction writes to the relevant address in EAMMem according to a write instruction (STRx) from a master.

### 3.3. Exclusive Access

This section outlines the operation for exclusive instructions (such as STREXx and LDREXx).

- Up to 4 ID (except for WID) and address sets are monitored simultaneously for the exclusive load instruction (LDREXx) and exclusive store instruction (STREXx) from a master.
- Whether access is exclusive or normal is determined by ARLOCK[1:0] (Read) or AWLOCK[1:0] (Write). The access is handled as exclusive load when ARLOCK[1:0] is "0b01" and as exclusive store when AWLOCK[1:0] is "0b01". In cases other than these, it is handled as normal read (LDRx) or write (STRx).



### 3.3.1. Exclusive Load Instruction (LDREXx)

This section describes the operation for the exclusive load instruction (LDREXx).

This section uses the following rules in transfer example tables:

The transfer example tables are described in ascending order of time.

However, transfers that have "+" in the Simultaneous Reception column are received simultaneously.

ADD (Address) A, B, C, and D are addresses whose address ranges do not overlap in any part. ADD (Address) A and ADD (Address) @A and the like are addresses whose address ranges overlap in some part.

- When an exclusive load instruction (LDREXx) is received, 1 address is monitored for 1 ID.
- Exclusive load instruction (LDREXx) monitoring is performed on 4 sets of different IDs and addresses. (A register that holds an ID and address as a set is hereafter referred to as a monitoring register.)
- If an exclusive load instruction (LDREXx) is determined as exclusive access NG with the following conditions, the response (RRESP[1:0]) to the master is set to OKAY ("0b00"). Otherwise, it is determined as exclusive access OK and EXOKAY ("0b01") is sent.
  1. If normal write (STRx) and an exclusive load instruction (LDREXx) are received simultaneously and they have the same address or their address ranges overlap in some part, the exclusive load instruction (LDREXx) is determined as exclusive access NG. (At this time, normal write (STRx) is written normally and the information of the exclusive load instruction (LDREXx) that is determined as exclusive access NG is not held by a monitoring register.)
  2. If an exclusive store instruction (STREXx) and an exclusive load instruction (LDREXx) are received simultaneously and they have the same address or their address ranges overlap in some part, the exclusive load instruction (LDREXx) is determined as exclusive access NG. (At this time, the exclusive store instruction (STREXx) is processed normally and the information of the exclusive load instruction (LDREXx) that is determined as exclusive access NG is not held by a monitoring register.)
  3. If an exclusive load instruction (LDREXx) is sent from a master by a method other than single burst (ARLENS[3:0]="0b0000" and ARBURSTS[1:0]="0b01"), it is determined as exclusive access NG.
  4. If monitoring registers already hold 4 exclusive load instructions (LDREXx) when an exclusive load instruction (LDREXx) is received, the ID of the received exclusive load instruction (LDREXx) does not match the values in the monitoring registers, and address ranges do not overlap in any part, the instruction is determined as exclusive access NG. (At this time, the information of the exclusive load instruction (LDREXx) that is determined as exclusive access NG is not held by a monitoring register.)

The following table provides an example of operation when normal write (STRx) and an exclusive load instruction (LDREXx) are received simultaneously and they have the same address, as described in 1. above.

**Table 3-1 When Normal Write (STRx) and Exclusive Load Instruction (LDREXx) are Received Simultaneously and They Have the Same Address**

Simultaneous Reception	Transfer Type	AxID	ADD	Response	Monitoring Register 1		Monitoring Register 2		Monitoring Register 3		Monitoring Register 4	
					ID	ADD	ID	ADD	ID	ADD	ID	ADD
	LDREXx	0	A	EXOKAY	0	A						
+	STRx	1	A	OKAY								
+	LDREXx	0	A	OKAY								
	LDREXx	3	B	EXOKAY	3	B						
+	STRx	0	C	OKAY	3	B						
+	LDREXx	2	C	OKAY	3	B						

The following tables provide examples of operation when an exclusive store instruction (STREXx) and an exclusive load instruction (LDREXx) are received simultaneously, as described in 2. above.

**Table 3-2 When Exclusive Store Instruction (STREXx) and Exclusive Load Instruction (LDREXx) are Received Simultaneously and They Have the Same Address**

Simultaneous Reception	Transfer Type	AxID	ADD	Response	Monitoring Register 1		Monitoring Register 2		Monitoring Register 3		Monitoring Register 4	
					ID	ADD	ID	ADD	ID	ADD	ID	ADD
	LDREXx	0	A	EXOKAY	0	A						
+	STREXx	0	B	OKAY	0	A						
+	LDREXx	1	B	OKAY	0	A						
	STREXx	0	A	EXOKAY								

**Table 3-3 When Exclusive Store Instruction (STREXx) and Exclusive Load Instruction (LDREXx) are Received Simultaneously and They Have Different Addresses**

Simultaneous Reception	Transfer Type	AxID	ADD	Response	Monitoring Register 1		Monitoring Register 2		Monitoring Register 3		Monitoring Register 4	
					ID	ADD	ID	ADD	ID	ADD	ID	ADD
	LDREXx	0	A	EXOKAY	0	A						
+	STREXx	0	B	OKAY	0	A						
+	LDREXx	1	A	EXOKAY	1	A						
	STREXx	1	A	EXOKAY								



**Table 3-4 When Exclusive Store Instruction (STREXx) and Exclusive Load Instruction (LDREXx) are Received Simultaneously and They Have Different Addresses**

Simultaneous Reception	Transfer Type	AxID	ADD	Response	Monitoring Register 1		Monitoring Register 2		Monitoring Register 3		Monitoring Register 4	
					ID	ADD	ID	ADD	ID	ADD	ID	ADD
	LDREXx	0	A	EXOKAY	0	A						
+	STREXx	0	A	EXOKAY								
+	LDREXx	1	B	EXOKAY	1	B						
	STREXx	1	B	EXOKAY								

The following tables provide examples of operations when address ranges do not overlap in any part and when address ranges overlap in some part, as described in 4. above.

**Table 3-5 When Exclusive Load Instruction (LDREXx) Whose Address Range (of ID Different from IDs Held by Monitoring Registers) Does Not Overlap in Any Part is Received**

Simultaneous Reception	Transfer Type	AxID	ADD	Response	Monitoring Register 1		Monitoring Register 2		Monitoring Register 3		Monitoring Register 4	
					ID	ADD	ID	ADD	ID	ADD	ID	ADD
	LDREXx	0	A	EXOKAY	0	A						
	LDREXx	1	B	EXOKAY	0	A	1	B				
	LDREXx	2	C	EXOKAY	0	A	1	B	2	C		
	LDREXx	3	D	EXOKAY	0	A	1	B	2	C	3	D
	LDREXx	4	E	OKAY	0	A	1	B	2	C	3	D

**Table 3-6 When Exclusive Load Instruction (LDREXx) Whose Address Range (of the Same ID as ID Held by Monitoring Register) Overlaps in Some Part is Received**

Simultaneous Reception	Transfer Type	AxID	ADD	Response	Monitoring Register 1		Monitoring Register 2		Monitoring Register 3		Monitoring Register 4	
					ID	ADD	ID	ADD	ID	ADD	ID	ADD
	LDREXx	0	A	EXOKAY	0	A						
	LDREXx	1	B	EXOKAY	0	A	1	B				
	LDREXx	2	C	EXOKAY	0	A	1	B	2	C		
	LDREXx	3	D	EXOKAY	0	A	1	B	2	C	3	D
	LDREXx	3	@A	EXOKAY			1	B	2	C	3	@A

### 3.3.2. Exclusive Store Instruction (STREXx)

This section describes the operation for the exclusive store instruction (STREXx).

This section uses the following rules in transfer example tables:

The transfer example tables are described in ascending order of time.

However, transfers that have "+" in the Simultaneous Reception column are received simultaneously.

ADD (Address) A, B, C, and D are addresses whose address ranges do not overlap in any part. ADD (Address) A and ADD (Address) @A and the like are addresses whose address ranges overlap in some part.

- If an exclusive store instruction (STREXx) is determined as exclusive access OK, a write command is sent to IntMemAXI without particularly processing the write command.
- If an exclusive store instruction (STREXx) is under the following conditions, it is determined as exclusive access NG, write strobe is dropped to IntMemAXI (0x00), and a write command that does not update data is sent. The response (BRESP[1:0]) to the master is set to OKAY ("0b00"). If the instruction is not under the following conditions, it is determined as exclusive access OK and EXOKAY ("0b01") is sent.
  1. If an exclusive store instruction (STREXx) that is different from exclusive load instructions (LDREXx) held by monitoring registers is received, it is determined as exclusive access NG. (This includes a case where normal write (STRx) cleared an internally held exclusive load instruction (LDREXx).)
  2. If an exclusive store instruction (STREXx) is sent from a master by means other than single burst (AWLENS[3:0]="0b0000" and AWBURSTS[1:0]="0b01"), it is determined as exclusive access NG.



The following tables provide examples of operation that is described in 1. above.

**Table 3-7 When Exclusive Store Instruction (STREXx) Different from Contents of Monitoring Registers is Received**

Simultaneous Reception	Transfer Type	AxID	ADD	Response	Monitoring Register 1		Monitoring Register 2		Monitoring Register 3		Monitoring Register 4	
					ID	ADD	ID	ADD	ID	ADD	ID	ADD
	LDREXx	0	A	EXOKAY	0	A						
	LDREXx	1	B	EXOKAY	0	A	1	B				
	LDREXx	2	C	EXOKAY	0	A	1	B	2	C		
	LDREXx	3	D	EXOKAY	0	A	1	B	2	C	3	D
	STREXx	0	E	OKAY	0	A	1	B	2	C	3	D

**Table 3-8 When Normal Write (STRx) Cleared Monitoring Register**

Simultaneous Reception	Transfer Type	AxID	ADD	Response	Monitoring Register 1		Monitoring Register 2		Monitoring Register 3		Monitoring Register 4	
					ID	ADD	ID	ADD	ID	ADD	ID	ADD
	LDREXx	0	A	EXOKAY	0	A						
	LDREXx	1	B	EXOKAY	0	A	1	B				
	STRx	0	A	OKAY			1	B				
	STREXx	0	A	OKAY			1	B				

## 4. Operation Examples

This section describes the operation of this block for exclusive access from a master.

This section uses the following rules in transfer example tables:

The transfer example tables are described in ascending order of time.

However, transfers that have "+" in the Simultaneous Reception column are received simultaneously.

ADD (Address) A, B, C, and D are addresses whose address ranges do not overlap in any part. ADD (Address) A and ADD (Address) @A and the like are addresses whose address ranges overlap in some part.

The following tables provide examples of the operation of this block when exclusive access is received from a master.

**Table 4-1 When Exclusive Load Instructions (LDREXx) Have Different IDs and Address Ranges**

Simultaneous Reception	Transfer Type	AxID	ADD	Response	Monitoring Register 1		Monitoring Register 2		Monitoring Register 3		Monitoring Register 4	
					ID	ADD	ID	ADD	ID	ADD	ID	ADD
	LDREXx	0	A	EXOKAY	0	A						
	LDREXx	1	B	EXOKAY	0	A	1	B				
	LDREXx	2	C	EXOKAY	0	A	1	B	2	C		
	LDREXx	3	D	EXOKAY	0	A	1	B	2	C	3	D
	STREXx	0	A	EXOKAY			1	B	2	C	3	D
	STREXx	1	B	EXOKAY					2	C	3	D
	STREXx	2	C	EXOKAY							3	D
	STREXx	3	D	EXOKAY								

**Table 4-2 When Exclusive Load Instructions (LDREXx) Have Different IDs and the Same Address Range**

Simultaneous Reception	Transfer Type	AxID	ADD	Response	Monitoring Register 1		Monitoring Register 2		Monitoring Register 3		Monitoring Register 4	
					ID	ADD	ID	ADD	ID	ADD	ID	ADD
	LDREXx	0	A	EXOKAY	0	A						
	LDREXx	1	A	EXOKAY	1	A						
	LDREXx	2	A	EXOKAY	2	A						
	LDREXx	3	A	EXOKAY	3	A						
	STREXx	0	A	OKAY	3	A						
	STREXx	1	A	OKAY	3	A						
	STREXx	2	A	OKAY	3	A						
	STREXx	3	A	EXOKAY								





**Table 4-3 When Exclusive Store Instructions (STREXx) That Have Different IDs for the Same Address Range Arrive**

Simultaneous Reception	Transfer Type	AxID	ADD	Response	Monitoring Register 1		Monitoring Register 2		Monitoring Register 3		Monitoring Register 4	
					ID	ADD	ID	ADD	ID	ADD	ID	ADD
	LDREXx	0	A	EXOKAY	0	A						
	STREXx	1	A	OKAY	0	A						
	LDREXx	1	A	EXOKAY	1	A						
	STREXx	2	A	OKAY	1	A						
	LDREXx	2	A	EXOKAY	2	A						
	STREXx	3	A	OKAY	2	A						
	LDREXx	3	A	EXOKAY	3	A						
	STREXx	0	A	OKAY	3	A						

Row 2: If STREXx is determined as exclusive access NG, the monitoring register holds values as they are.  
Row 3: If LDREXx has the same address range, the current ID is overwritten with the ID of LDREXx.

**Table 4-4 When ID and Address Range of Exclusive Load Instruction (LDREXx) Do Not Match Those of Exclusive Store Instruction (STREXx)**

Simultaneous Reception	Transfer Type	AxID	ADD	Response	Monitoring Register 1		Monitoring Register 2		Monitoring Register 3		Monitoring Register 4	
					ID	ADD	ID	ADD	ID	ADD	ID	ADD
	LDREXx	0	A	EXOKAY	0	A						
	LDREXx	1	B	EXOKAY	0	A	1	B				
	LDREXx	2	C	EXOKAY	0	A	1	B	2	C		
	LDREXx	3	D	EXOKAY	0	A	1	B	2	C	3	D
	STREXx	0	D	OKAY			1	B	2	C	3	D
	STREXx	1	C	OKAY					2	C	3	D
	STREXx	2	B	OKAY							3	D
	STREXx	3	A	OKAY								

**Table 4-5 When Order of Exclusive Load Instructions (LDREXx) is Different from That of Exclusive Store Instructions (STREXx)**

Simultaneous Reception	Transfer Type	AxID	ADD	Response	Monitoring Register 1		Monitoring Register 2		Monitoring Register 3		Monitoring Register 4	
					ID	ADD	ID	ADD	ID	ADD	ID	ADD
	LDREXx	0	A	EXOKAY	0	A						
	LDREXx	1	B	EXOKAY	0	A	1	B				
	LDREXx	2	C	EXOKAY	0	A	1	B	2	C		
	LDREXx	3	D	EXOKAY	0	A	1	B	2	C	3	D
	STREXx	3	D	EXOKAY	0	A	1	B	2	C		
	STREXx	2	C	EXOKAY	0	A	1	B				
	STREXx	1	B	EXOKAY	0	A						
	STREXx	0	A	EXOKAY								

**Table 4-6 When 5 Exclusive Load Instructions (LDREXx) are Received**

Simultaneous Reception	Transfer Type	AxID	ADD	Response	Monitoring Register 1		Monitoring Register 2		Monitoring Register 3		Monitoring Register 4	
					ID	ADD	ID	ADD	ID	ADD	ID	ADD
	LDREXx	0	A	EXOKAY	0	A						
	LDREXx	1	B	EXOKAY	0	A	1	B				
	LDREXx	2	C	EXOKAY	0	A	1	B	2	C		
	LDREXx	3	D	EXOKAY	0	A	1	B	2	C	3	D
	LDREXx	0	E	EXOKAY	0	E	1	B	2	C	3	D
	STREXx	0	A	OKAY			1	B	2	C	3	D
	STREXx	1	B	EXOKAY					2	C	3	D
	STREXx	2	C	EXOKAY							3	D
	STREXx	3	D	EXOKAY								
	STREXx	0	E	OKAY								

Row 5: The values are overwritten by LDREXx that has the same ID and a different address range.

Row 6: The response is OKAY because a monitoring register is overwritten (in row 5).

**Table 4-7 When 5 Exclusive Load Instructions (LDREXx) are Received**

Simultaneous Reception	Transfer Type	AxID	ADD	Response	Monitoring Register 1		Monitoring Register 2		Monitoring Register 3		Monitoring Register 4	
					ID	ADD	ID	ADD	ID	ADD	ID	ADD
	LDREXx	0	A	EXOKAY	0	A						
	LDREXx	1	B	EXOKAY	0	A	1	B				
	LDREXx	2	C	EXOKAY	0	A	1	B	2	C		
	LDREXx	3	D	EXOKAY	0	A	1	B	2	C	3	D
	LDREXx	4	E	OKAY	0	A	1	B	2	C	3	D
	STREXx	0	A	EXOKAY			1	B	2	C	3	D
	STREXx	1	B	EXOKAY					2	C	3	D
	STREXx	2	C	EXOKAY							3	D
	STREXx	3	D	EXOKAY								
	STREXx	4	E	OKAY								

Row 5: The monitoring registers are not overwritten because the ID is different from the 4 IDs that are held.  
LDREXx=OKAY



**Table 4-8 When Exclusive Store Instruction (STREXx) Arrives before Exclusive Load Instruction (LDREXx)**

Simultaneous Reception	Transfer Type	AxID	ADD	Response	Monitoring Register 1		Monitoring Register 2		Monitoring Register 3		Monitoring Register 4	
					ID	ADD	ID	ADD	ID	ADD	ID	ADD
	STREXx	0	A	OKAY								
	LDREXx	0	A	EXOKAY	0	A						

Row 1: The response is OKAY because STREXx is received when there are no monitoring targets in the monitoring registers.

**Table 4-9 When There is Normal Write (STRx) That Has the Same Address between Exclusive Load Instruction (LDREXx) and Exclusive Store Instruction (STREXx) (When Normal Write is Received, Monitoring Registers are Cleared)**

Simultaneous Reception	Transfer Type	AxID	ADD	Response	Monitoring Register 1		Monitoring Register 2		Monitoring Register 3		Monitoring Register 4	
					ID	ADD	ID	ADD	ID	ADD	ID	ADD
	LDREXx	0	A	EXOKAY	0	A						
	STRx	0	A	OKAY								
	STREXx	0	A	OKAY								
	LDREXx	0	B	EXOKAY	0	B						
	STRx	1	B	OKAY								
	STREXx	0	B	OKAY								

Row 2: Normal write (STRx) to the same ID and address as those of the exclusive load instruction (LDREXx).

Row 6: Normal write (STRx) to a different ID but from the same address as that of the exclusive load instruction (LDREXx).

**Table 4-10 When Exclusive Load Instruction (LDREXx) and Exclusive Store Instruction (STREXx) Arrive Simultaneously (If It Arrives Simultaneously with STREXx, STREXx Takes Precedence)**

Simultaneous Reception	Transfer Type	AxID	ADD	Response	Monitoring Register 1		Monitoring Register 2		Monitoring Register 3		Monitoring Register 4	
					ID	ADD	ID	ADD	ID	ADD	ID	ADD
	LDREXx	1	A	EXOKAY	1	A						
+	LDREXx	2	A	OKAY	1	A						
+	STREXx	0	A	OKAY	1	A						

Row 3: The response is STREXx=OKAY and monitoring registers are not cleared because the ID does not match the ID in the monitoring register.

Row 2: The response is LDREXx=OKAY and monitoring registers are not updated because the address matches the address of STREXx received simultaneously.

**Table 4-11 When Exclusive Load Instruction (LDREXx) and Exclusive Store Instruction (STREXx) Arrive Simultaneously (If It Arrives Simultaneously with STREXx, STREXx Takes Precedence)**

Simultaneous Reception	Transfer Type	AxID	ADD	Response	Monitoring Register 1		Monitoring Register 2		Monitoring Register 3		Monitoring Register 4	
					ID	ADD	ID	ADD	ID	ADD	ID	ADD
	LDREXx	1	A	EXOKAY	1	A						
+	LDREXx	2	B	EXOKAY	2	B						
+	STREXx	1	A	EXOKAY								

Row 3: The response is STREXx=EXOKAY and monitoring registers are cleared because both the ID and address match those in the monitoring register.

Row 2: The address does not match the address of STREXx received simultaneously. Therefore, it is held as a new monitoring target in the monitoring register.

**Table 4-12 When Exclusive Load Instruction (LDREXx) and Exclusive Store Instruction (STREXx) Arrive Simultaneously (If It Arrives Simultaneously with STREXx, STREXx Takes Precedence)**

Simultaneous Reception	Transfer Type	AxID	ADD	Response	Monitoring Register 1		Monitoring Register 2		Monitoring Register 3		Monitoring Register 4	
					ID	ADD	ID	ADD	ID	ADD	ID	ADD
	LDREXx	1	A	EXOKAY	1	A						
+	LDREXx	1	B	EXOKAY	1	B						
+	STREXx	0	A	OKAY	1	A						

Row 3: The response is STREXx=OKAY and monitoring registers are not cleared because the ID does not match the ID in the monitoring register.

Row 2: The monitoring register is updated because the address does not match the address of STREXx received simultaneously and the ID is the same as the ID in the monitoring register.

**Table 4-13 When Exclusive Load Instruction (LDREXx) and Exclusive Store Instruction (STREXx) Arrive Simultaneously (If It Arrives Simultaneously with STREXx, STREXx Takes Precedence)**

Simultaneous Reception	Transfer Type	AxID	ADD	Response	Monitoring Register 1		Monitoring Register 2		Monitoring Register 3		Monitoring Register 4	
					ID	ADD	ID	ADD	ID	ADD	ID	ADD
+	LDREXx	1	A	EXOKAY	1	A						
+	STREXx	0	B	OKAY								

Row 2: The response is STREXx=OKAY because there is no monitoring target in monitoring registers (no LDREXx is received before STREXx).

Row 1: The address does not match the address of STREXx received simultaneously. Therefore, it is held as a new monitoring target in the monitoring register.



**Table 4-14 When Exclusive Load Instruction (LDREXx) is Received Simultaneously with Normal Write (STRx) and They Have the Same Address (If It is Received Simultaneously with Normal Write to the Same Address Range, OKAY is Returned to LDREXx)**

Simultaneous Reception	Transfer Type	AxID	ADD	Response	Monitoring Register 1		Monitoring Register 2		Monitoring Register 3		Monitoring Register 4	
					ID	ADD	ID	ADD	ID	ADD	ID	ADD
+	LDREXx	0	A	OKAY								
+	STRx	1	A	OKAY								

Row 2: The response is OKAY because the transfer type is normal write (STRx).

Row 1: The response is LDREXx=OKAY because the address is the same as that of normal Write (STRx).

**Table 4-15 When Exclusive Load Instruction (LDREXx) is Received Simultaneously with Normal Write (STRx) and They Have the Same Address (If It is Received Simultaneously with Normal Write to the Same Address Range, OKAY is Returned to LDREXx)**

Simultaneous Reception	Transfer Type	AxID	ADD	Response	Monitoring Register 1		Monitoring Register 2		Monitoring Register 3		Monitoring Register 4	
					ID	ADD	ID	ADD	ID	ADD	ID	ADD
	LDREXx	1	B	EXOKAY	1	B						
+	LDREXx	0	B	OKAY								
+	STRx	2	B	OKAY								
	STREXx	1	B	OKAY								

Row 3: If the address is the same as in the monitoring register, the monitoring register is cleared.

Row 2: The response is LDREXx=OKAY because the address is same as that of normal Write (STRx).

Row 4: The response is STREXx=OKAY because the monitoring register is cleared by normal write (STRx).

**Table 4-16 When Exclusive Load Instruction (LDREXx) is Received Simultaneously with Normal Write (STRx) and Their Address Ranges Do Not Match**

Simultaneous Reception	Transfer Type	AxID	ADD	Response	Monitoring Register 1		Monitoring Register 2		Monitoring Register 3		Monitoring Register 4	
					ID	ADD	ID	ADD	ID	ADD	ID	ADD
+	LDREXx	0	A	EXOKAY	0	A						
+	STRx	1	B	OKAY								

**Table 4-17 When Exclusive Store Instruction (STREXx) Does Not Arrive after Exclusive Load Instruction (LDREXx) (Monitoring Continues Until STREXx Arrives)**

Simultaneous Reception	Transfer Type	AxID	ADD	Response	Monitoring Register 1		Monitoring Register 2		Monitoring Register 3		Monitoring Register 4	
					ID	ADD	ID	ADD	ID	ADD	ID	ADD
	LDREXx	0	A	EXOKAY	0	A						
		...	...		0	A						
		...	...		0	A						
		...	...		0	A						

**Table 4-18 When Address Ranges Overlap in Some Part and Exclusive Store Instruction (STREXx) with Previous Monitoring Target Address is Received after Overwriting (When Address Ranges A and @A Overlap in Some Part, Monitoring Target is Overwritten)**

Simultaneous Reception	Transfer Type	AxID	ADD	Response	Monitoring Register 1		Monitoring Register 2		Monitoring Register 3		Monitoring Register 4	
					ID	ADD	ID	ADD	ID	ADD	ID	ADD
	LDREX	0	A	EXOKAY	0	A						
	LDREXD	0	@A	EXOKAY	0	@A						
	STREX	0	A	OKAY	0	@A						
	STREXD	0	@A	EXOKAY								

Row 2: The address range is overwritten with the later address range because the address ranges overlap in some part.

Row 3: OKAY is returned to STREX for the previous address range (that does not exactly match the monitoring target address). When the response is OKAY, WSTRB is dropped to prevent the relevant area being overwritten, and therefore the monitoring target continues to be used.

**Table 4-19 When Address Ranges Overlap in Some Part and Exclusive Store Instruction (STREXx) with Previous Monitoring Target Address is Received after Overwriting (When Address Ranges A and @A Overlap in Some Part, Monitoring Target is Overwritten)**

Simultaneous Reception	Transfer Type	AxID	ADD	Response	Monitoring Register 1		Monitoring Register 2		Monitoring Register 3		Monitoring Register 4	
					ID	ADD	ID	ADD	ID	ADD	ID	ADD
	LDREXD	0	@A	EXOKAY	0	@A						
	LDREX	0	A	EXOKAY	0	A						
	STREXD	0	@A	OKAY	0	A						
	STREX	0	A	EXOKAY								

Row 2: The address range is overwritten with the later address range because the address ranges overlap in some part.

Row 3: OKAY is returned to STREXD for the previous address range (that does not exactly match the monitoring target address). When the response is OKAY, WSTRB is dropped to prevent the relevant area being overwritten, and therefore the monitoring target continues to be used.



## 5. Memory Area

This section describes the memory area of the exclusive access memory (EAM).

**Table 5-1 List of EAM**

Abbreviated Memory Name	Memory Name	See
EAM	Exclusive Access Memory	5.1

## 5.1. Exclusive Access Memory

All 48-byte memories (exclusive access memories) mounted on the exclusive access memory (EAM) allow exclusive access. There are no special access restrictions for this memory area.

Base(0x0280\_0000)+0x0000\_0000

BIT_OFFSET	31-0
BIT_NAME	Exclusive Access Memory
ACCESS_TYPE	R/W
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000_00000000

:

:

Base(0x0280\_0000)+0x0000\_002C

BIT_OFFSET	31-0
BIT_NAME	Exclusive Access Memory
ACCESS_TYPE	R/W
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000_00000000

(Base+0x0000\_0000 to Base+0x0000\_002C)

### [bit31:0] Exclusive access memory

These bits allow writing and reading of exclusive access data.







## CHAPTER 27: I/O Port

This chapter describes I/O port.

---

1. Overview
2. Configuration
3. Setting Procedure Examples
4. Registers
5. Precautions for Using



## 1. Overview

This section provides an overview of I/O port.

I/O port has a general-purpose I/O module. The external pins can be used as I/O ports. Assignment of an external pin and input to an internal resource can be set.

### (1) General-purpose I/O Port Module (GPIO)

General-purpose I/O module enables using external pins as I/O ports. The general-purpose I/O module is composed of 5 GPIO ports. Each GPIO port has 32 channels, which correspond to external pins. For example, the external pin P216 corresponds to the channel 16 setting of the GPIO port 2.

In the part number equipped with the key code function, the key code setting is necessary for writing in a specific register. See Section "Part Number Option" of "Data Sheet" for the presence of the key code function.

### (2) Port Configuration Module (PPC)

The port configuration module sets I/O from/to an external pin. This setting can be set per external pin.

- Output enable display
- I/O status display
- Pull-up setting
- Input level setting
- Output drive capacity setting
- Input disable setting
- Output (GPIO, resource) function selection

In the part number equipped with the key code function, the key code setting is necessary for writing in a specific register. See Section "Part Number Option" of "Data Sheet" for the presence of the key code function.

### (3) Resource Input Configuration Module (RIC)

The resource input configuration module selects input from an external pin or output from another internal resource as resource input.

2. Configuration

This section describes the block diagram of I/O port.  
Figure 2-1 and Figure 2-2 show the I/O port configuration diagrams.

Figure 2-1 Configuration Diagram of GPIO and PPC

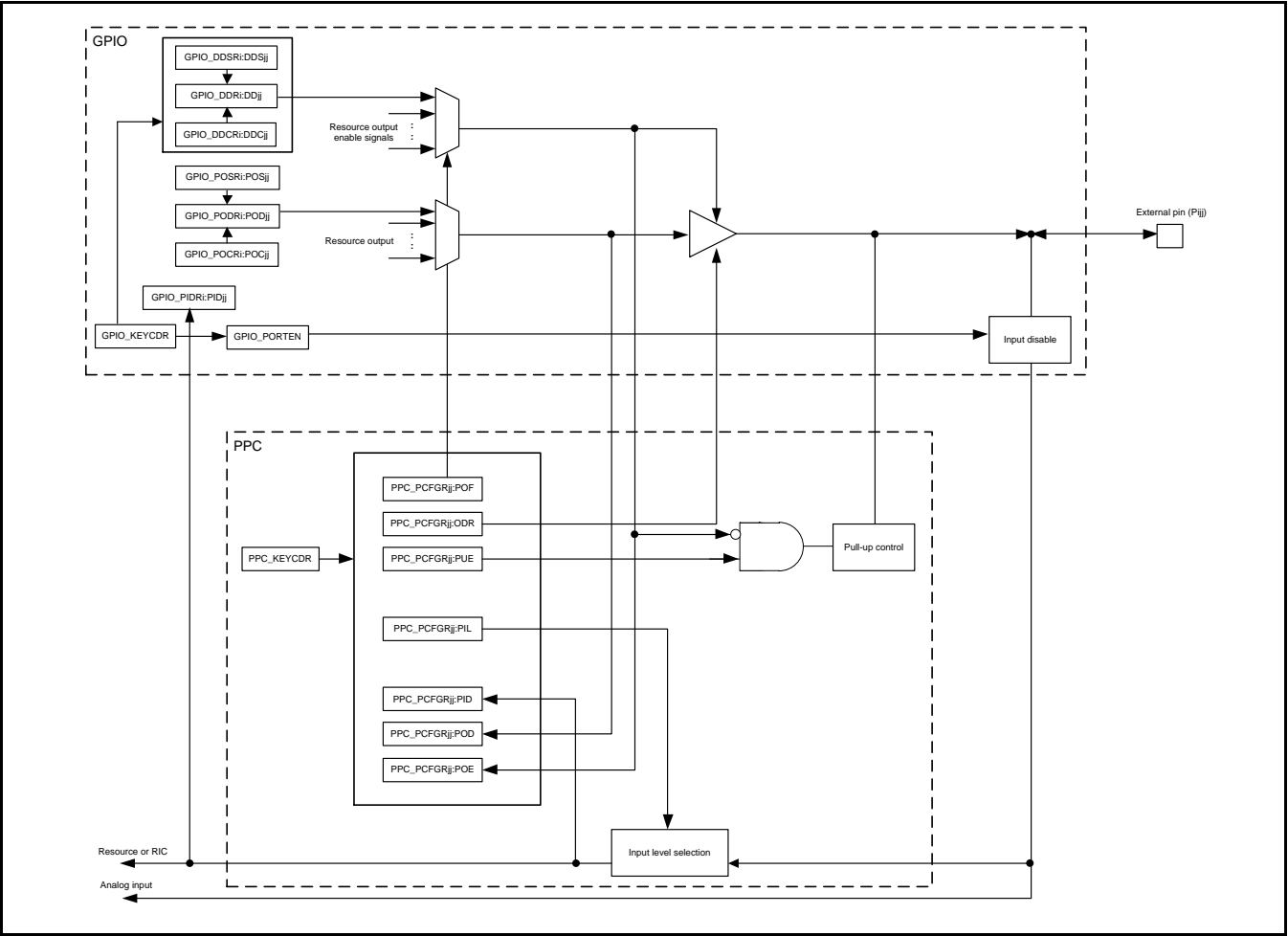
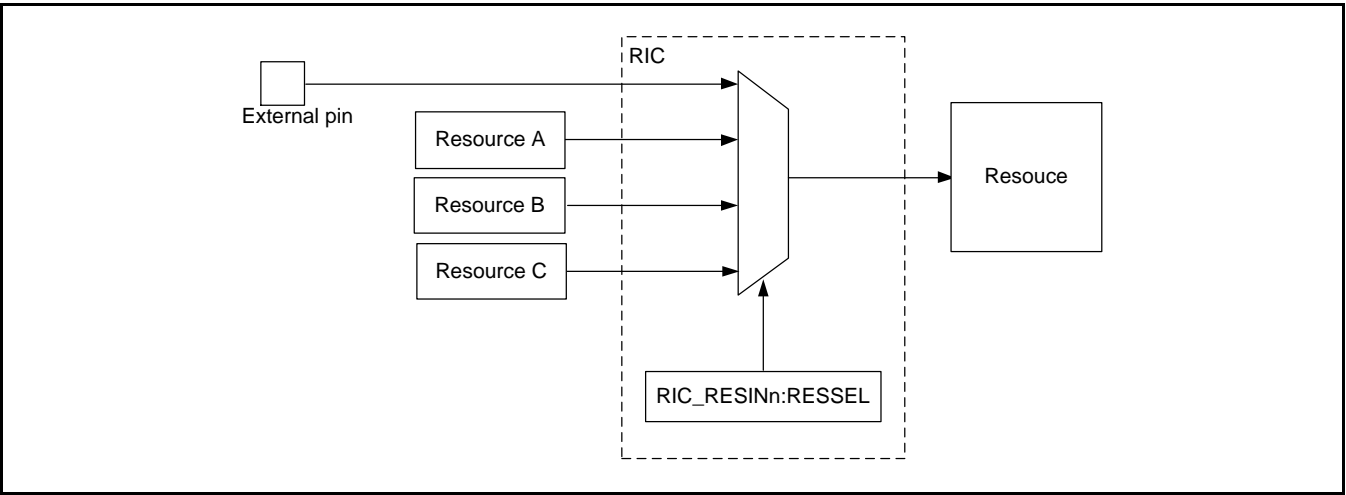


Figure 2-2 Configuration Diagram of RIC

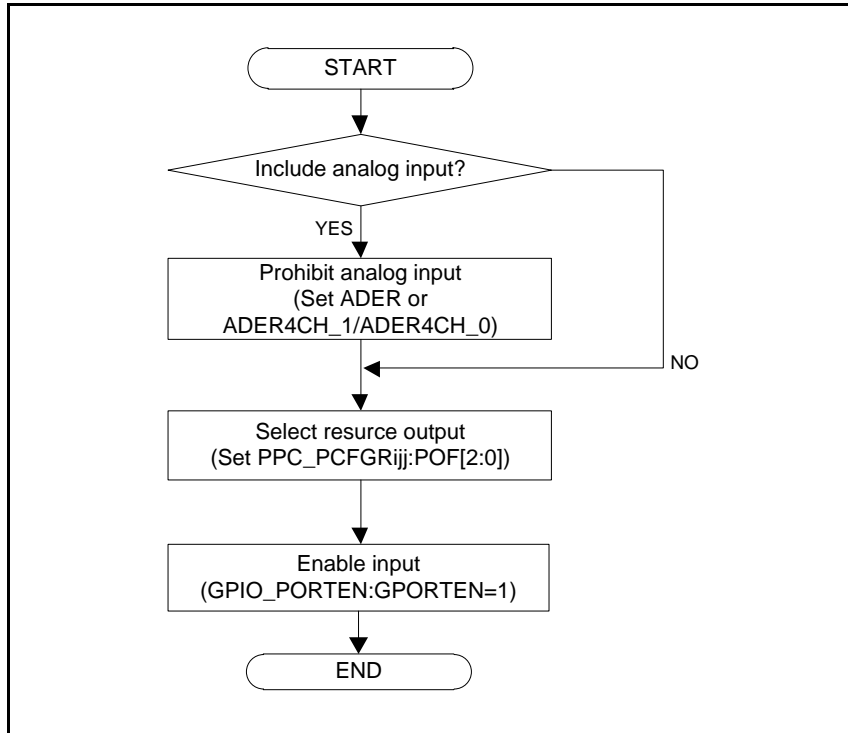


### 3. Setting Procedure Examples

This section describes the setting procedure examples of I/O port.

#### (1) Assignment of Peripheral I/O Pin (Both Directions)

Figure 3-1 Setting Procedure of Peripheral I/O Pin (Both Directions)

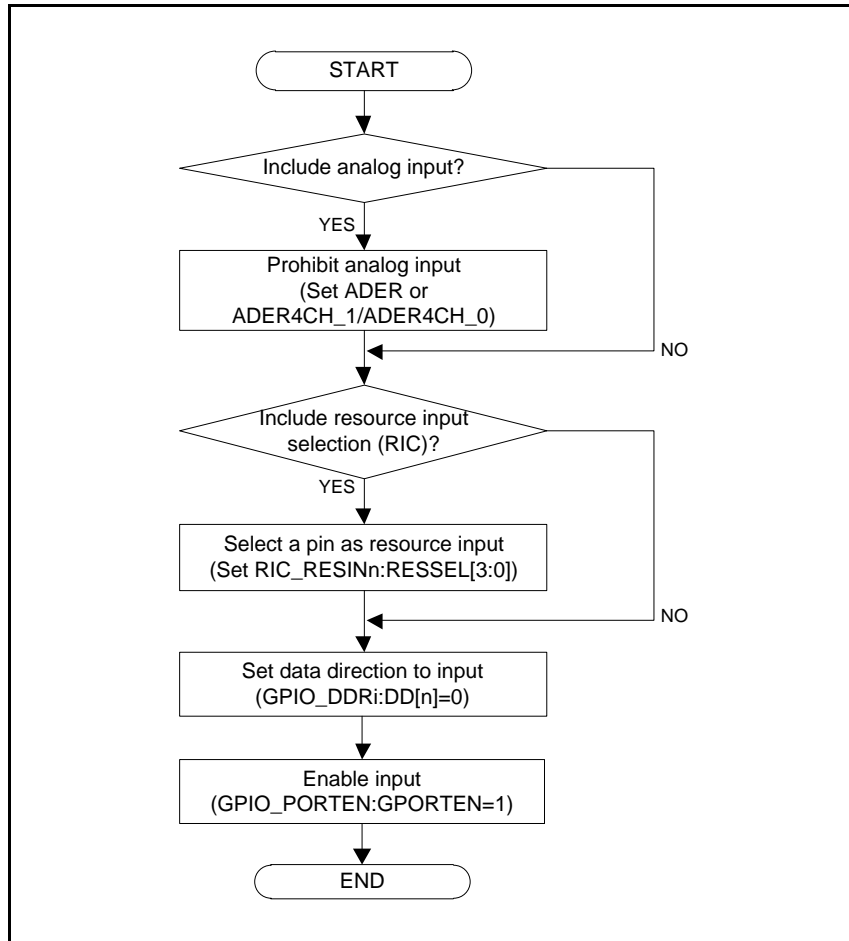


**Notes:**

- The following registers are the applicable key code registers.
  - Analog input control register (ADER)
  - Port enable register (GPIO\_PORTEN)
  - 4ch ADC analog input enable register (ADER4CH\_1, ADER4CH\_0)
- For details on the analog input disable setting, see "CHAPTER: 12-bit A/D Converter A/D Activation Compare" and "CHAPTER: 12-bit 4ch A/D Converter Interface".

(2) Assignment of Resource Input Pin (Selecting the Resource Input of an External Pin)

Figure 3-2 Setting Procedure of Resource Input Pin

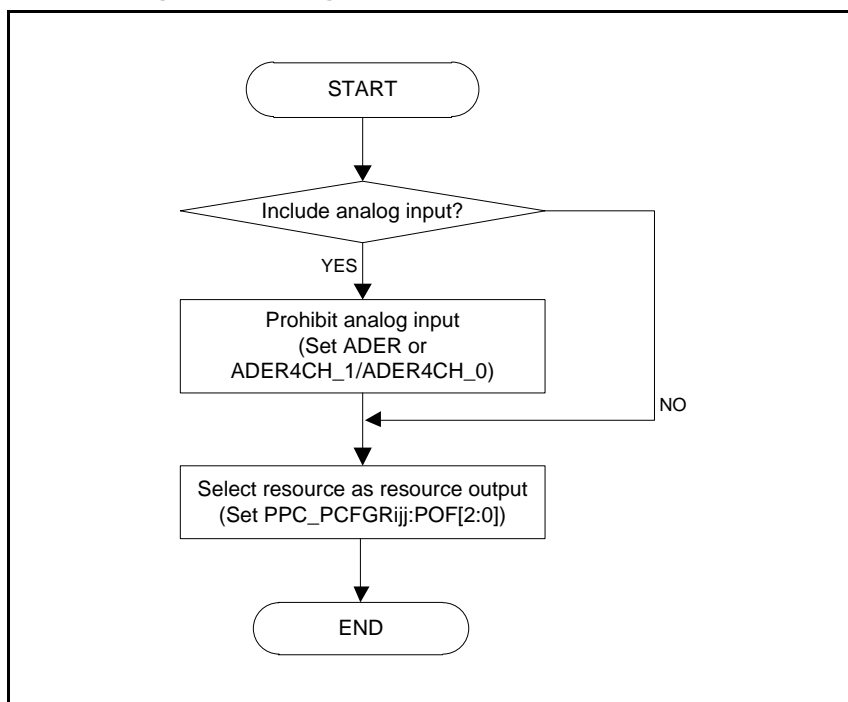


**Notes:**

- The following registers are the applicable key code registers.
  - Analog input control register (ADER)
  - Port enable register (GPIO\_PORTEN)
  - 4ch ADC analog input enable register (ADER4CH\_1, ADER4CH\_0)
- For details on the analog input disable setting, see "CHAPTER: 12-bit A/D Converter A/D Activation Compare" and "CHAPTER: 12-bit 4ch A/D Converter Interface".

### (3) Assignment of Resource Output Pin

Figure 3-3 Setting Procedure of Resource Output Pin

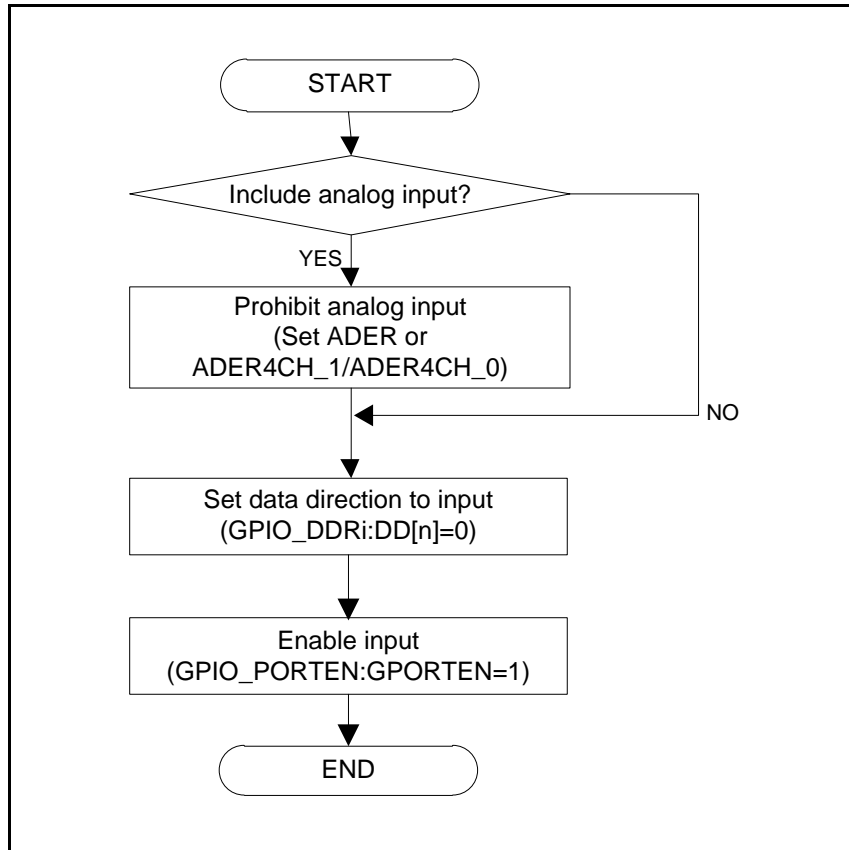


**Notes:**

- The following registers are the applicable key code registers.
  - Analog input control register (ADER)
  - Port enable register (GPIO\_PORTEN)
  - 4ch ADC analog input enable register (ADER4CH\_1, ADER4CH\_0)
- For details on the analog input disable setting, see "CHAPTER: 12-bit A/D Converter A/D Activation Compare" and "CHAPTER: 12-bit 4ch A/D Converter Interface".

#### (4) Assignment of Port Function (Input)

Figure 3-4 Setting Procedure of Port Function (Input)



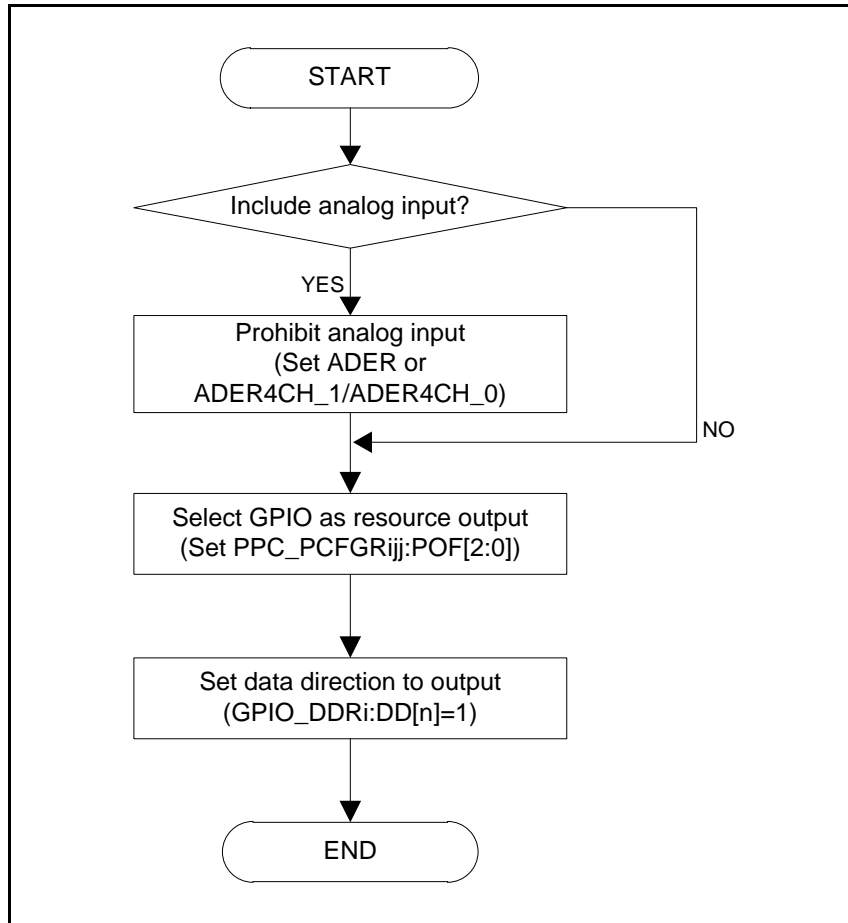
**Notes:**

- The following registers are the applicable key code registers.
  - Analog input control register (ADER)
  - Port enable register (GPIO\_PORTEN)
  - 4ch ADC analog input enable register (ADER4CH\_1, ADER4CH\_0)
- For details on the analog input disable setting, see "CHAPTER: 12-bit A/D Converter A/D Activation Compare" and "CHAPTER: 12-bit 4ch A/D Converter Interface".



### (5) Assignment of Port Function (Output)

Figure 3-5 Setting Procedure of Port Function (Output)

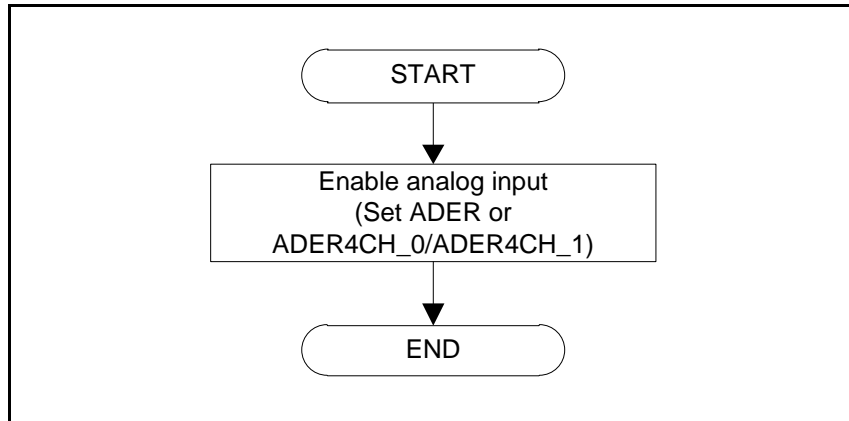


**Notes:**

- The following registers are the applicable key code registers.
  - Analog input control register (ADER)
  - Port enable register (GPIO\_PORTEN)
  - 4ch ADC analog input enable register (ADER4CH\_1, ADER4CH\_0)
- For details on the analog input disable setting, see "CHAPTER: 12-bit A/D Converter A/D Activation Compare" and "CHAPTER: 12-bit 4ch A/D Converter Interface".

### (6) Assignment of A/D Converter Input

Figure 3-6 Setting Procedure of A/D Converter Input

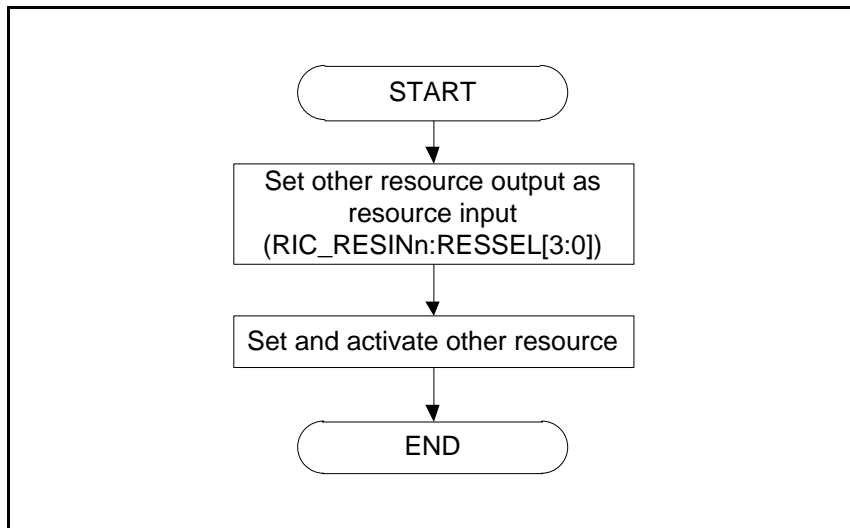


**Notes:**

- The following registers are the applicable key code registers.
  - Analog input control register (ADER)
  - Port enable register (GPIO\_PORTEN)
  - 4ch ADC analog input enable register (ADER4CH\_1, ADER4CH\_0)
- For details on the analog input disable setting, see "CHAPTER: 12-bit A/D Converter A/D Activation Compare" and "CHAPTER: 12-bit 4ch A/D Converter Interface".

### (7) Resource Input Selection (Selecting Other Resource Output)

Figure 3-7 Setting Procedure of Resource Input Selection (Selecting Other Resource Output)



**Note:**

- For details on resource input selection, see Section "Resource Input Selection" in "CHAPTER: Appendix".

## 4. Registers

This section describes the register list of the I/O port.

**Table 4-1 List of I/O Port Registers**

Abbreviated Register Name	Register Name	See
GPIO_DDRI	Data Direction Register	4.1
GPIO_DDRI	Data Direction Set Register	4.2
GPIO_DDCRI	Data Direction Clear Register	4.3
GPIO_PODRI	Port Output Data Register	4.4
GPIO_POSRI	Port Output Set Register	4.5
GPIO_POCRI	Port Output Clear Register	4.6
GPIO_PIDRI	Port Input Data Register	4.7
GPIO_PORTEN	Port Input Enable Register	4.8
GPIO_KEYCDR	GPIO Key Code Register	4.9
PPC_PCFGRijj	Port Configuration Register	4.10
PPC_KEYCDR	PPC Key Code Register	4.11
RIC_RESINn	Resource Input Setting Register	4.12

i: GPIO port number (i = 0 to 4)

jj: GPIO channel number (jj = 00 to 31)

n: Number of selectable resource inputs (n = 0 to 11)



#### 4.1. Data Direction Register (GPIO\_DDRi) (i=0 to 4)

This register sets the I/O direction of a pin. The direction is set when a GPIO is selected by the port output function selection bit (POF[2:0]) of the port setting register (PPC\_PCFGRijj). This register can be directly written or can be set/cleared using the data direction set register (GPIO\_DDSRi) and the data direction clear register (GPIO\_DDCRi).

BIT_OFFSET	31-0
BIT_NAME	DD
ACCESS_TYPE	R/W
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000_00000000

##### [bit n] DD[n]: Data direction selection bit (n=0 to 31)

Value	Description
0	Set data direction to input.
1	Set data direction to output.

**Note:**

- This is the applicable key code register. To write to this register, the GPIO key code register (GPIO\_KEYCDR) must be set. If this register is written before the key code is released, a bus error response is returned. See Section "Part Number Option" of "Data Sheet" for the presence of the key code function.

## 4.2. Data Direction Set Register (GPIO\_DDSRi) (i=0 to 4)

This register is used to set the data direction selection bit (GPIO\_DDRi: DD31 to DD0).

BIT_OFFSET	31-0
BIT_NAME	DDS
ACCESS_TYPE	R0,W
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000_00000000

### [bit n] DDS[n]: Data direction set bit (n=0 to 31)

Value	Description
0	No effect
1	Set the data direction selection bit (GPIO_DDRi: DD31 to DD0) to "1".

**Note:**

- This is the applicable key code register. To write to this register, the GPIO key code register (GPIO\_KEYCDR) must be set. If this register is written before the key code is released, a bus error response is returned. See Section "Part Number Option" of "Data Sheet" for the presence of the key code function.



### 4.3. Data Direction Clear Register (GPIO\_DDCRi) (i=0 to 4)

This register is used to clear the data direction selection bit (GPIO\_DDRi: DD31 to DD0).

BIT_OFFSET	31-0
BIT_NAME	DDC
ACCESS_TYPE	R0,W
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000_00000000

#### [bit n] DDC[n]: Data direction clear register (n=0 to 31)

Value	Description
0	No effect
1	Clear the data direction selection bit (GPIO_DDRi: DD31 to DD0) to "0".

**Note:**

- This is the applicable key code register. To write to this register, the GPIO key code register (GPIO\_KEYCDR) must be set. If this register is written before the key code is released, a bus error response is returned. See Section "Part Number Option" of "Data Sheet" for the presence of the key code function.

#### 4.4. Port Output Data Register (GPIO\_PODRi) (i=0 to 4)

This register is used to set a port output value. The setting value is enabled when a port output is selected (GPIO\_DDRi: DD31 to DD0 = 1). This register can be directly written or can be set/cleared using the port output set register (GPIO\_POSRi) and the port output clear register (GPIO\_POCRi).

BIT_OFFSET	31-0
BIT_NAME	POD
ACCESS_TYPE	R/W
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000_00000000

Bit30 of GPIO\_PODR1 is Reserved (Reserved bit).

Bit31 and bit8 to 7 of GPIO\_PODR3 are Reserved (Reserved bits).

Bit24 and bit5 to 0 of GPIO\_PODR4 are Reserved (Reserved bits).

R/W attribute and initial value of reserved bits are same to other bits.

##### [bit n] POD[n]: Port output data bit (n=0 to 31)

Value	Description
0	Output "L".
1	Output "H".





## 4.5. Port Output Set Register (GPIO\_POSRi) (i=0 to 4)

This register is used to set the port output data bit (GPIO\_PODRi: POD31 to POD0).

BIT_OFFSET	31-0
BIT_NAME	POS
ACCESS_TYPE	R0,W
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000_00000000

Bit30 of GPIO\_POSR1 is Reserved (Reserved bit).

Bit31 and bit8 to 7 of GPIO\_POSR3 are Reserved (Reserved bits).

Bit24 and bit5 to 0 of GPIO\_POSR4 are Reserved (Reserved bits)

R/W attribute and initial value of reserved bits are same to other bits.

### [bit n] POS[n]: Port output set bit (n=0 to 31)

Value	Description
0	No effect
1	Set the port output data bit (GPIO_PODRi: POD31 to POD0) to "1".

## 4.6. Port Output Clear Register (GPIO\_POCRi) (i=0 to 4)

This register is used to clear the port output data bit (GPIO\_PODRi: POD31 to POD0).

BIT_OFFSET	31-0
BIT_NAME	POC
ACCESS_TYPE	R0,W
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000_00000000

Bit30 of GPIO\_POCR1 is Reserved (Reserved bit).

Bit31 and bit8 to 7 of GPIO\_POCR 3 are Reserved (Reserved bits).

Bit24 and bit5 to 0 of GPIO\_POCR 4 are Reserved (Reserved bits).

R/W attribute and initial value of reserved bits are same to other bits.

### [bit n] POC[n]: Port output clear bit (n=0 to 31)

Value	Description
0	No effect
1	Clear the port output data bit (GPIO_PODRi: POD31 to POD0) to "0".



#### 4.7. Port Input Data Register (GPIO\_PIDRi) (i=0 to 4)

This register indicates an input data value. This register indicates input data according to the setting of the input level selection bit (PPC\_PCFGRij:PIL[1:0]) when the global input enable bit (GPIO\_PORTEN:GPORTEN) is "1". An indefinite value is read from the port input data register when the global input enable bit (GPIO\_PORTEN:GPORTEN) is "0".

BIT_OFFSET	31-0
BIT_NAME	PID
ACCESS_TYPE	R,WX
PROT_TYPE	-
INITIAL_VALUE	XXXXXXXX_XXXXXXXX_XXXXXXXX_XXXXXXXX

Bit30 of GPIO\_PIDR1 is Reserved (Reserved bit).

Bit31 and bit8 to 7 of GPIO\_PIDR3 are Reserved (Reserved bits).

Bit24 and bit5 to 0 of GPIO\_PIDR4 are Reserved (Reserved bits).

R/W attribute and initial value of reserved bits are same to other bits.

##### [bit n] PID[n]: Port input data register (n = 0 to 31)

Value	Description
0	"L" is input.
1	"H" is input.

## 4.8. Port Input Enable Register (GPIO\_PORTEN)

This register sets the input rejection of a port.

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved						CPORTEN	GPORTEN
ACCESS_TYPE	R0,WX						R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	000000						0	0

**[bit31:2] Reserved: Reserved bits**

**[bit1] CPORTEN: Serial write pin input enable bit**

Input to a serial write pin is enabled. Regardless of the global input enable bit (GPORTEN) setting, this bit can be used to enable input to the serial write pin.

Value	Description
0	No operation
1	Enable input to the serial write pin (Multi-function serial interface ch.0)

**[bit0] GPORTEN: Global input enable bit**

Value	Description
0	Disable input to the pin.
1	Enable input to the pin

**Note:**

- This is the applicable key code register. To write to this register, the GPIO key code register (GPIO\_KEYCDR) must be set. If this register is written before the key code is released, a bus error response is returned. See Section "Part Number Option" of "Data Sheet" for the presence of the key code function.



## 4.9. GPIO Key Code Register (GPIO\_KEYCDR)

This register sets register writing with a function for protection against erroneous writing. If writing to this register is not done using the prescribed method, writing to the relevant register is invalid. In the part number unequipped with the key code function, access to this register has no effect on the operation.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	KEY		SIZE		Reserved			
ACCESS_TYPE	R0,W		R0,W		R0,WX			
PROT_TYPE	-							
INITIAL_VALUE	00		00		0000			

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved	RADR[14:8]						
ACCESS_TYPE	R0,WX	R0,W						
PROT_TYPE	-							
INITIAL_VALUE	0	0000000						

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	RADR[7:0]							
ACCESS_TYPE	R0,W							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

### [bit31:30] KEY[1:0]: Key code bits

These are key code setting bits. Write "00", "01", "10", and "11" continuously to these bits in this order. The key code setting becomes invalid immediately upon any writing to these bits in a different order. In such cases, set the key code again from the beginning.

Value	Description
00	1st key code
01	2nd key code
10	3rd key code
11	4th key code

**[bit29:28] SIZE[1:0]: Access size bits**

These bits set the access size for writing to the applicable key code register. Write the same data to these bits when writing key codes "00", "01", "10", and "11" in this order.

Value	Description
00	Set byte access.
01	Set half-word access.
10	Set word access.
11	Reserved

**[bit27:15] Reserved: Reserved bits**

**[bit14:0] RADR[14:0]: Port address bits**

These bits set the lower 15 bits of the address of the applicable key code register. Write the same data to these bits when writing key codes "00", "01", "10", and "11" in this order.

Value	Description
	Set the lower 15 bits of the address of the applicable key code register.

**Notes:**

- The key code setting becomes invalid upon any writing to the KEY[1:0] in any order than "00", "01", "10", and "11". In such case, set the key code again from the beginning.
- When different data is written to the SIZE[1:0] or the RADR[14:0] while writing the key code "00", "01", "10", and "11", the key code setting will become invalid. In such case, set it again from the beginning.
- If writing to register with different access size or address after releasing the key code, the key code setting becomes invalid. Set it from the beginning again.
- The applicable key code registers are the following registers.
  - Data direction register (GPIO\_DDRI)
  - Data direction set register (GPIO\_DDSDRI)
  - Data direction clear register (GPIO\_DDCRI)
  - Port input enable register (GPIO\_PORTEN)
- This register is valid only for word access.
- There is no effect on the key code protection release process even if registers other than this register and the applicable key code register that is in the release processing (the register set by the RADA) are accessed while writing the key code "00", "01", "10", and "11".



## 4.10. Port Configuration Register (PPC\_PCFGRIj) (i=0 to 4, jj=00 to 31)

This register sets and displays I/O from/to an external pin.

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	POE	POD	PID	Reserved	PIL		PUE	Reserved
ACCESS_TYPE	R,WX	R,WX	R,WX	R0,WX	R/W		R/W	R0,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	X	0	00		0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	ODR		Reserved			POF		
ACCESS_TYPE	R/W		R0,WX			R/W		
PROT_TYPE	-							
INITIAL_VALUE	00		0000			000		

There are no port configuration registers following number.

- i=1, jj=30
- i=3, jj=7 to 8, 31
- i=4, jj=0 to 5, 24

### [bit15] POE: Port output enable bit

This bit indicates whether pin output is enabled.

Value	Description
0	Pin output is high-impedance.
1	Pin output is enabled.

### [bit14] POD: Port output data bit

This bit indicates the value outputted to a pin.

This bit is enabled only when the port output enable bit (POE) is "1".

Value	Description
0	Output "L".
1	Output "H".

### [bit13] PID: Port input data bit

This bit indicates the value inputted to the pin selected by the input level bit (PIL[1:0]).

This bit is indefinite when the global input enable bit (GPIO\_GPORTEN) is "0".

Value	Description
0	The pin input is "L".
1	The pin input is "H".

### [bit12] Reserved: Reserved bit

**[bit11:10] PIL[1:0]: Input level bits**

This bit selects a pin input level.

Value	Description
00	Type A
01	Type B
10	Type C
11	Type D

For details, see Section "Input Level Setting of I/O Port Various Settings" in "CHAPTER: Appendix".

**[bit9] PUE: Pull up enable bit**

This bit sets whether pull-up is performed or not in input status.

Value	Description
0	No pull up.
1	Pull up.

**[bit8] Reserved: Reserved bit**

**[bit7:6] ODR[1:0]: Port output drive selection bits**

This bit selects an output drive capacity of a port.

Value	Description
00	Type A
01	Type B
10	Type C
11	Type D

For details, see Section "Output Drive Setting" in "CHAPTER: Appendix"

**[bit5:3] Reserved: Reserved bits**

**[bit2:0] POF[2:0]: Port output function selection bits**

This bit selects a function to output to a port.

Value	Description
000	Resource A output
001	Resource B output
010	Resource C output
011	Resource D output
100	Resource E output
101	Resource F output
110	Resource G output
111	Resource H output

For details, see Section "Output Resource Selection" in "CHAPTER: Appendix".





**Note:**

- *This is the applicable key code register. To write to this register, the PPC key code register (PPC\_KEYCDR) must be set. If this register is written before the key code is released, a bus error response is returned. See Section "Part Number Option" of "Data Sheet" for the presence of the key code function.*

## 4.11. PPC Key Code Register (PPC\_KEYCDR)

This register sets register writing with a function for protection against erroneous writing. If writing to this register is not done using the prescribed method, writing to the relevant register is invalid. In the part number unequipped with the key code function, access to this register has no effect on the operation.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	KEY		SIZE		Reserved			
ACCESS_TYPE	R0,W		R0,W		R0,WX			
PROT_TYPE	-							
INITIAL_VALUE	00		00		0000			

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved	RADR[14:8]						
ACCESS_TYPE	R0,WX	R0,W						
PROT_TYPE	-							
INITIAL_VALUE	0	0000000						

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	RADR[7:0]							
ACCESS_TYPE	R0,W							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

### [bit31:30] KEY[1:0]: Key code bits

These are key code setting bits. Write "00", "01", "10", and "11" continuously to these bits in this order. The key code setting becomes invalid immediately upon any writing to these bits in a different order. In such cases, set the key code again from the beginning.

Value	Description
00	1st key code
01	2nd key code
10	3rd key code
11	4th key code



#### [bit29:28] SIZE[1:0]: Access size bits

These bits set the access size for writing to the applicable key code register. Write the same data to these bits when writing key codes "00", "01", "10", and "11" in this order.

Value	Description
00	Set byte access.
01	Set half-word access.
10	Set word access.
11	Reserved

#### [bit27:15] Reserved: Reserved bits

#### [bit14:0] RADR[14:0]: Port address bits

These bits set the lower 15 bits of the address of the applicable key code register. Write the same data to these bits when writing key codes "00", "01", "10", and "11" in this order.

Value	Description
	Set the lower 15 bits of the address of the applicable key code register.

#### Notes:

- The key code setting becomes invalid upon any writing to the KEY[1:0] in any order than "00", "01", "10", and "11". In such case, set the key code again from the beginning.
- When different data is written to the SIZE[1:0] or the RADR[14:0] while writing the key code "00", "01", "10", and "11", the key code setting will become invalid. In such case, set it again from the beginning.
- If writing to register with different access size or address after releasing the key code, the key code setting becomes invalid. Set it from the beginning again.
- The applicable key code register is the port setting register (PPC\_PCFGRIj).
- This register is valid only for word access.
- There is no effect on the key code protection release process even if registers other than this register and the applicable key code register that is in the release processing (the register set by the RADA) are accessed while writing the key code "00", "01", "10", and "11".

## 4.12. Resource Input Setting Register (RIC\_RESINn) (n = 0 to 11)

This register selects an external pin input or an output from another internal resource as resource input.

### (1) Resource Input Setting Register 0 to 9 (RIC\_RESINn (n=0 to 9))

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved				Reserved			
ACCESS_TYPE	R0,WX				R0,W0			
PROT_TYPE	-							
INITIAL_VALUE	0000				0000			

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved				RESSEL			
ACCESS_TYPE	R0,WX				R/W			
PROT_TYPE	-							
INITIAL_VALUE	0000				0000			

[bit15:12] Reserved: Reserved bits

[bit11:8] Reserved: Reserved bits

[bit7:4] Reserved: Reserved bits

#### [bit3:0] RESSEL[3:0]: Resource selection bits

These bits select an input to a resource.

Value	Description
0000	Source A
0001	Source B
0010	Source C
0011	Source D
0100	Source E
0101	Source F
0110	Source G
0111	Source H
1000	Source I
1001	Source J
1010	Source K
1011	Source L
1100	Source M
1101	Source N
1110	Source O
1111	Source P

For assignment of source, see Section "Resource Input Selection" in "CHAPTER: Appendix".



**(2) Resource Input Setting Register 10 (RIC\_RESIN10)**

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved				Reserved			PORTSEL
ACCESS_TYPE	R0,WX				R0,W0			R/W
PROT_TYPE	-							
INITIAL_VALUE	0000				000			0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved				RESSEL			
ACCESS_TYPE	R0,WX				R/W			
PROT_TYPE	-							
INITIAL_VALUE	0000				0000			

**[bit15:12] Reserved: Reserved bits**

**[bit11:9] Reserved: Reserved bits**

**[bit8] PORTSEL: Resource selection bit**

This bit selects external activation trigger factor of base timer ch.0, ch.2, ch.4.

Value	Description
0	External pin input corresponding to base timer ch.0, ch.2, ch.4
1	Gate output of Waveform generator 0 (ch.0, ch.2, ch.4) (Connection) Waveform generator ch.0 → Base timer ch.0 Waveform generator ch.2 → Base timer ch.2 Waveform generator ch.4 → Base timer ch.4

**[bit7:4] Reserved: Reserved bits**

**[bit3:0] RESSEL[3:0]: Resource selection bits**

These bits select an input to a resource.

Value	Description
0000	Source A
0001	Source B
0010	Source C
0011	Source D
0100	Source E
0101	Source F
0110	Source G
0111	Source H
1000	Source I
1001	Source J
1010	Source K
1011	Source L
1100	Source M
1101	Source N
1110	Source O
1111	Source P

For assignment of source, see Section "Resource Input Selection" in "CHAPTER: Appendix".



**(3) Resource Input Setting Register 11 (RIC\_RESIN11)**

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved				Reserved			PORTSEL
ACCESS_TYPE	R0,WX				R0,W0			R/W
PROT_TYPE	-							
INITIAL_VALUE	0000				000			0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved				RESSEL			
ACCESS_TYPE	R0,WX				R/W			
PROT_TYPE	-							
INITIAL_VALUE	0000				0000			

**[bit15:12] Reserved: Reserved bits**

**[bit11:9] Reserved: Reserved bits**

**[bit8] PORTSEL: Resource selection bit**

This bit selects external activation trigger factor of base timer ch.6, ch.8, ch.10.

Value	Description
0	External pin input corresponding to base timer ch.6, ch.8, ch.10
1	Gate output of Waveform generator 1 (ch.6, ch.8, ch.10) (Connection) Waveform generator ch.6 → Base timer ch.6 Waveform generator ch.8 → Base timer ch.8 Waveform generator ch.10 → Base timer ch.10

**[bit7:4] Reserved: Reserved bits**

**[bit3:0] RESSEL[3:0]: Resource selection bits**

These bits select an input to a resource.

Value	Description
0000	Source A
0001	Source B
0010	Source C
0011	Source D
0100	Source E
0101	Source F
0110	Source G
0111	Source H
1000	Source I
1001	Source J
1010	Source K
1011	Source L
1100	Source M
1101	Source N
1110	Source O
1111	Source P

For assignment of source, see Section "Resource Input Selection" in "CHAPTER: Appendix".





## 5. Precautions for Using

This section describes the precautions necessary when using the I/O port.

A glitch may occur for a brief second (2 or 3 ns) when a general-purpose I/O port is switched (from input to output or from output to input).



## CHAPTER 28: CR Calibration

This chapter explains CR calibration.

---

1. Overview
2. Configuration
3. Operation
4. Registers



1. Overview

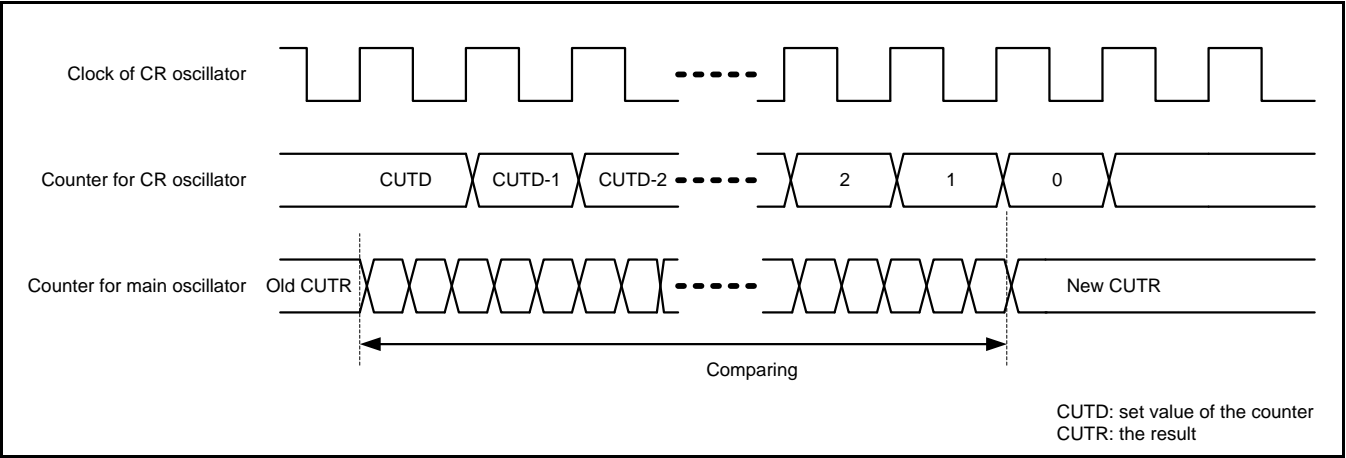
This section provides an overview of CR calibration.

The fast-CR oscillator can correct frequencies by configuring trimming. A trimming value can be determined from the calculation of a count value for correcting the frequency of the fast-CR oscillation circuit.

CR Oscillation Correction

Measurement of a clock error can be done by driving the counter driven by the main clock and measuring it for the set period of the counter driven by the CR clock (Figure 1-1).

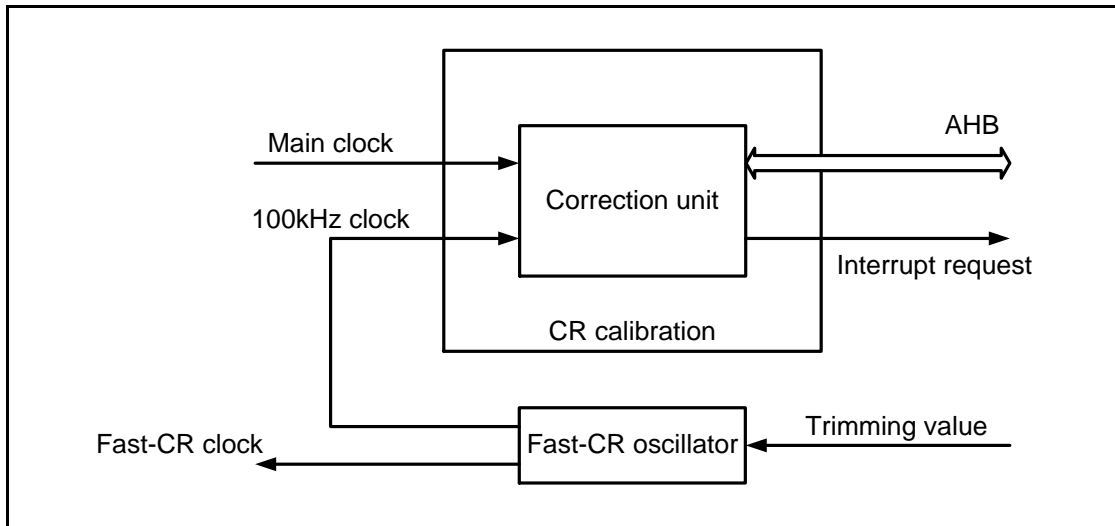
Figure 1-1 Comparison of Counters Driven by Each Clock



## 2. Configuration

This section explains the configuration for CR calibration.

Figure 2-1 CR Calibration Configuration Diagram



**Note:**

- For details on the setting of a trimming value, see the description of the fast-CR oscillation control register (SYSC\_CRCNTR) in the following "CHAPTER: Clock System".



### 3. Operation

This section explains the operation for CR calibration.

This explanation shows measurement of the CR clock frequency (error measurement). The measurement uses the following registers. For details, see Section "4. Registers".

- Correction unit control register (CUCR1)  
Controls the correction unit.
- CR clock timer data register 1 (CUTD1)  
Stores the set value of the measurement period. (CR clock driving)
- Main oscillation timer data register (CUTR1)  
Stores the main oscillation timer count results.

#### 3.1. CR Clock Frequency Calculation

##### Frequency Calculation Procedure

1. Set the CR clock timer data register 1 (CUTD1).
2. Set "1" in the interrupt enable bit (CUCR1:INTEN).
3. Set "1" in the correction start bit (CUCR1:STRT).
4. Run a loop to wait for an interrupt.
5. Generate an interrupt.
6. Read the main oscillation timer data register 1 (CUTR1).
7. The ratio of "main clock frequency:CR clock frequency" can be calculated with "CUTR1:CUTD1".

## 3.2. CR Clock Frequency Correction

This section shows frequency correction procedures.

### (1) Measurement of Maximum Frequency (Fmax)

With the CRTRM[7:0] bits as "0b1111\_1111" in the fast-CR oscillation control register (SYSC\_CRCNTR) of the clock system, calculate the maximum frequency (Fmax) of the CR clock from the error measurement by the correction unit.

### (2) Measurement of Minimum Frequency (Fmin)

With the CRTRM[7:0] bits as "0b0000\_0000" in the fast-CR oscillation control register (SYSC\_CRCNTR) of the clock system, calculate the minimum frequency (Fmin) of the CR clock from the error measurement by the correction unit.

### (3) Calculation of Trimming Value

For n that is next to the frequency degree per step (Fstep) in the following expression, assign a value from 0 to 255. Then, calculate the value of n where the frequency Ftrm after trimming is closest to 100kHz.

$$Fstep = (Fmax - Fmin) / 255 \quad (Fstep: \text{Frequency per step of the trimming value})$$

$$Ftrm = Fmin + Fstep * n$$

### (4) Setting of Trimming Value

Set the calculated trimming value (n) in the CRTRM[7:0] bits in the fast-CR oscillation control register (SYSC\_CRCNTR) of the clock system.

**Note:**

- A profile update overrides the counter value when the main and fast-CR clocks have stopped. Write "0" in the STRT bit to stop comparison. Then, rewrite "1" and start over.



## 4. Registers

This section explains the CR calibration registers.

Table 4-1 lists the registers.

**Table 4-1 List of CR Calibration Registers**

Abbreviated Register Name	Register Name	See
CU_CUCR1	Correction Unit Control Register 1	4.1
CU_CUTD1	CR Clock Timer Data Register 1	4.2
CU_CUTR1	Main Oscillation Timer Register 1	4.3
CU_CUCRC1	Correction Unit Control Clear Register 1	4.4

## 4.1. Correction Unit Control Register 1 (CU\_CUCR1)

This register sets the start of correction of the CR oscillation correction unit, and the clearing and enabling of interrupts.

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved							
ACCESS_TYPE	R1,WX							
PROT_TYPE	-							
INITIAL_VALUE	11111111							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved	Reserved		STRT	Reserved		INT	INTEN
ACCESS_TYPE	R/W0	R0,WX		R,W	R0,WX		R,W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	00		0	0		0	0

**[bit15:8] Reserved: Reserved bits**

**[bit7] Reserved: Reserved bit**

This bit disables writing of "1". Normal operation is not guaranteed when "1" is written.

**[bit6:5] Reserved: Reserved bits**

**[bit4] STRT (calibration STaRT): Correction start**

This bit starts the 100 kHz drive counter generated from the main clock and CR clock. The INT bit is set to "1" at the completion of comparison.

If "0" is set, the comparison is aborted. Writing "1" during the comparison does not have any effect. The bit is cleared to "0" when the comparison is completed.

Value	Description
0	Abort comparison.
1	Start comparison (during the comparison).

**[bit3:2] Reserved: Reserved bits**

**[bit1] INT (calibration INTerrupt): Interrupt**

This bit is set to "1" at the completion of comparison. If the INTEN bit setting is "1", an interrupt is generated. Writing "0" clears it. If the INTC bit setting for CU\_CUCRC1 register is "1", this bit will be cleared by "0".





**[bit0] INTEN (calibration INTerrupt ENable): Enabling interrupt**

This bit sets whether to generate an interrupt for an INT bit setting.

Value	Description
0	Disable interrupt.
1	Enable interrupt.

## 4.2. CR Clock Timer Data Register 1 (CU\_CUTD1)

This register sets the drive period of the counter driven by the CR clock.

BIT_OFFSET	15-0
BIT_NAME	TDD
ACCESS_TYPE	R/W
PROT_TYPE	-
INITIAL_VALUE	11000011_01010000

### [bit15:0] TDD[15:0] (Timer Data Data): Timer data

These bits set the comparison period with a number of CR clocks.

Value	Description
0x0000	65536
0x0001	1
0x0002	2
0x0003	3
...	...
0xC350	50000
...	...
0xFFFD	65533
0xFFFE	65534
0xFFFF	65535



### 4.3. Main Oscillation Timer Data Register 1 (CU\_CUTR1)

This register indicates the number of counters driven by the main clock during the set period of CUTD1.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

BIT_OFFSET	23-0							
BIT_NAME	TDR							
ACCESS_TYPE	R,WX							
PROT_TYPE	-							
INITIAL_VALUE	00000000_00000000_00000000							

**[bit31:24] Reserved: Reserved bits**

#### **[bit23:0] TDR[23:0] (Timer Data Register): Timer data**

These bits indicate the number of counts that were counted in the comparison period. Read them after the comparison is completed.

The value read during a comparison is undefined and writing is invalid.

Value	Description
0x00_0000	0
0x00_0001	1
0x00_0002	2
0x00_0003	3
...	...
0xFF_FFFD	16777213
0xFF_FFFE	16777214
0xFF_FFFF	16777215

## 4.4. Correction Unit Control Clear Register 1 (CU\_CUCRC1)

This register clears interrupt of the CR oscillation correction unit.

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved						INTC	Reserved
ACCESS_TYPE	R0,WX						R0,W	R0,WX
PROT_TYPE	-							
INITIAL_VALUE	000000						0	0

**[bit31:2] Reserved: Reserved bits**

**[bit1] INTC (INT Clear): Interrupt clear**

The read value is "0". If "1" is written, the INT bit in the CUCR1 register is cleared.

Value	Description
0	No effect
1	Clear interrupts

**[bit0] Reserved: Reserved bit**





## CHAPTER 29: CRC

This chapter explains the CRC.

---

1. Overview
2. Configuration
3. Operation
4. Registers



## 1. Overview

The CRC (Cyclic Redundancy Check) is one of the error detecting systems. Considering the input data column a high-degree polynomial, the CRC code is the remainder that results when division by the predetermined generator polynomial is implemented. Normally, a CRC code is added to the end of the data column for transmission, and the received data is divided by the generator polynomial in the same way. If there is no remainder, the received data is judged as correct data.

This calculation is available for CCITT CRC16 and IEEE-802.3 CRC32. The generator polynomial is fixed to the values for these two codes. No calculations of the CRC value based on any other generator polynomials are available.

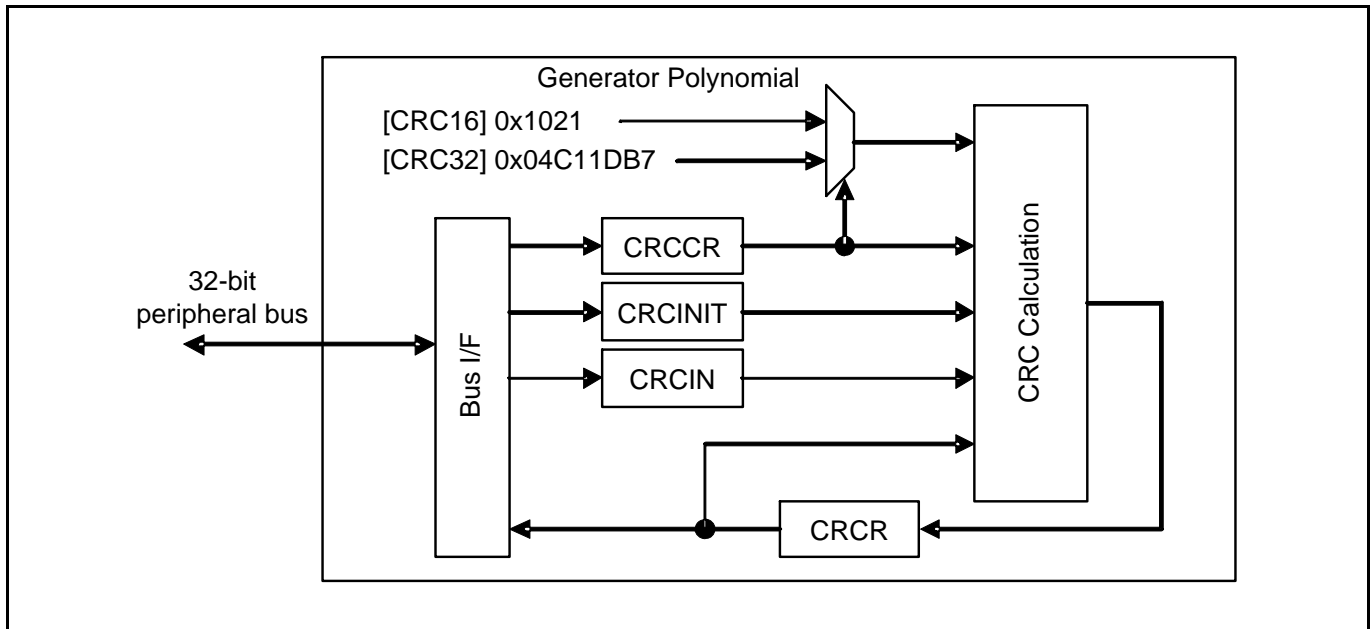
The mounted CRC has the following features.

- CCITT CRC16 generator polynomial : 0x1021
- IEEE-802.3 CRC32 generator polynomial : 0x04C11DB7

## 2. Configuration

This section provides the CRC configuration diagram.

Figure 2-1 CRC Configuration Diagram



- CRCCR (CRC control register)  
This controls CRC calculations.
- CRCINIT (Initial value register)  
This sets the initial value of a CRC calculation.
- CRCIN (Input Data register)  
This sets the input data of a CRC calculation.
- CRCR (CRC register)  
This outputs the result of a CRC calculation.
- CRC Calculation  
This is a circuit that performs CRC calculations.





### 3. Operation

This section explains the operation of CRC.

#### (1) Definition of CRC

[CCITT CRC16 Standard]

Generator polynomial	0x1021	(CRCCR: CRC32="0")
Initial Value	0xFFFF	
Final XOR value	0x0000	(CRCCR: FXOR="0")
Bit order	MSB First	(CRCCR: LSBFST="0")
Output bit order	MSB First	(CRCCR: CRCLSF="0")
(The byte order of I/O can be set arbitrarily.)		

[IEEE-802.3 CRC32 Ethernet Standard]

Generator polynomial	0x04C11DB7	(CRCCR: CRC32="1")
Initial Value	0xFFFF_FFFF	
Final XOR value	0xFFFF_FFFF	(CRCCR: FXOR="1")
Bit order	LSB First	(CRCCR: LSBFST="1")
Output bit order	LSB First	(CRCCR: CRCLSF="1")
(The byte order of I/O can be set arbitrarily.)		

#### (2) Reset Operation

At a reset, set the initial value register (CRCINIT) and the CRC register (CRCR) to 0xFFFF\_FFFF. Clear others to "0".

#### (3) Initialization

To use Initialization bit (CRCCR: INIT) for initialization, load the value of the initial value register to the CRC register (CRCR).

#### (4) Byte Order and Bit Order

The byte order processing and bit order processing are explained with the following examples.  
Enter the following 1 word to the CRC arithmetic unit.

133.82.171.1 = "0b10000101" "0b01010010" "0b10101011" "0b00000001"

When the byte order is set to big endian (CRCCR: LTLEND="0"), the transmission order by bytes is as shown below.

"0b10000101" "0b01010010" "0b10101011" "0b00000001"  
(1st) (2nd) (3rd) (4th)

When the bit order is set to LSB First (CRCCR: LSBFST="1"), the transmission order by bits is as shown below.

"0b10100001" "0b01001010" "0b11010101" "0b10000000"  
(Head) (End)

#### Notes:

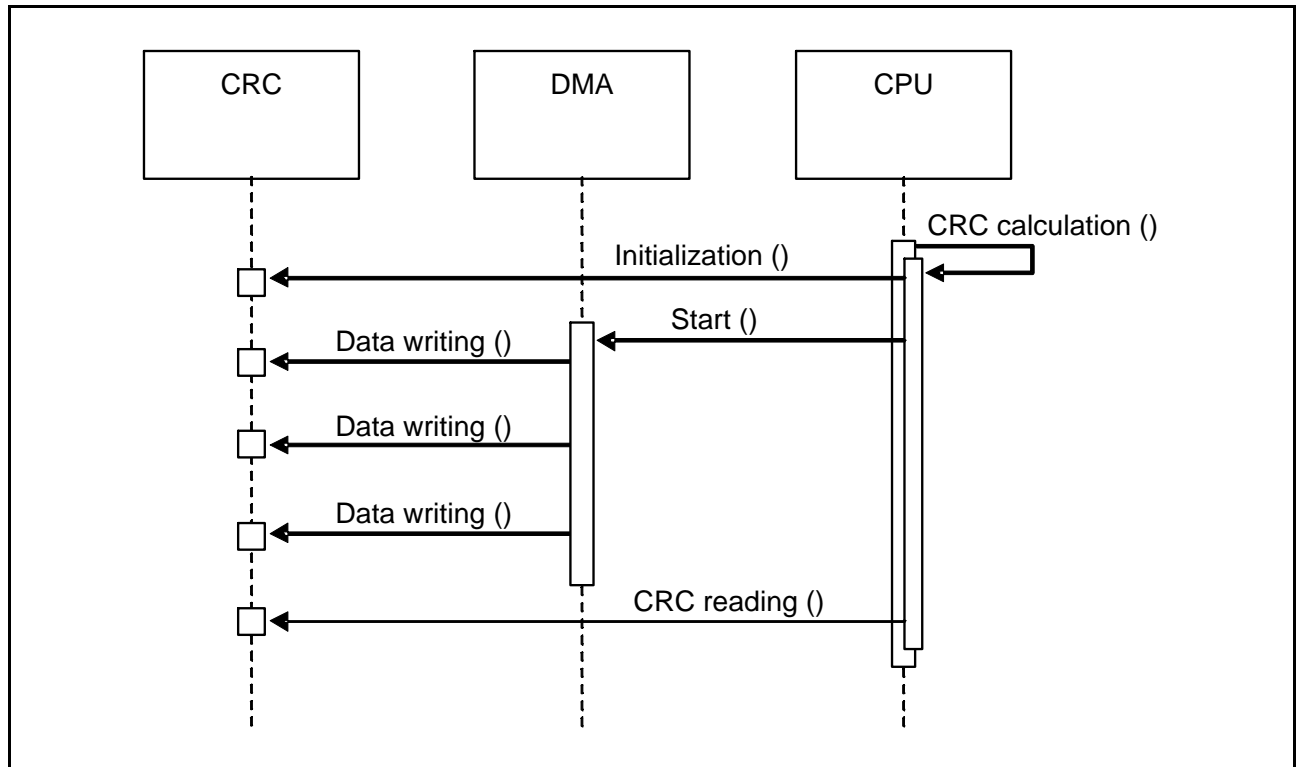
- When CRCCR: CRCLTE="1", the CRC result is rearranged by bytes with 32-bit width in both CRC16 and CRC32 cases.
- Especially for CRC16, it requires special care because the value is output to the positions of bit31 to bit16.

### 3.1. CRC Computing Sequence

Assume the initial value register (CRCINIT) setting, CRC16/32 selection (CRCCR: CRC32), byte order/bit order setting (CRCCR: LTLEND or CRCCR: LSBFST) have been done.

(If the initial value can be 0xFFFFFFFF, the setting operation of the initial value register (CRCINIT) can be omitted.)

Figure 3-1 CRC Computing Sequence



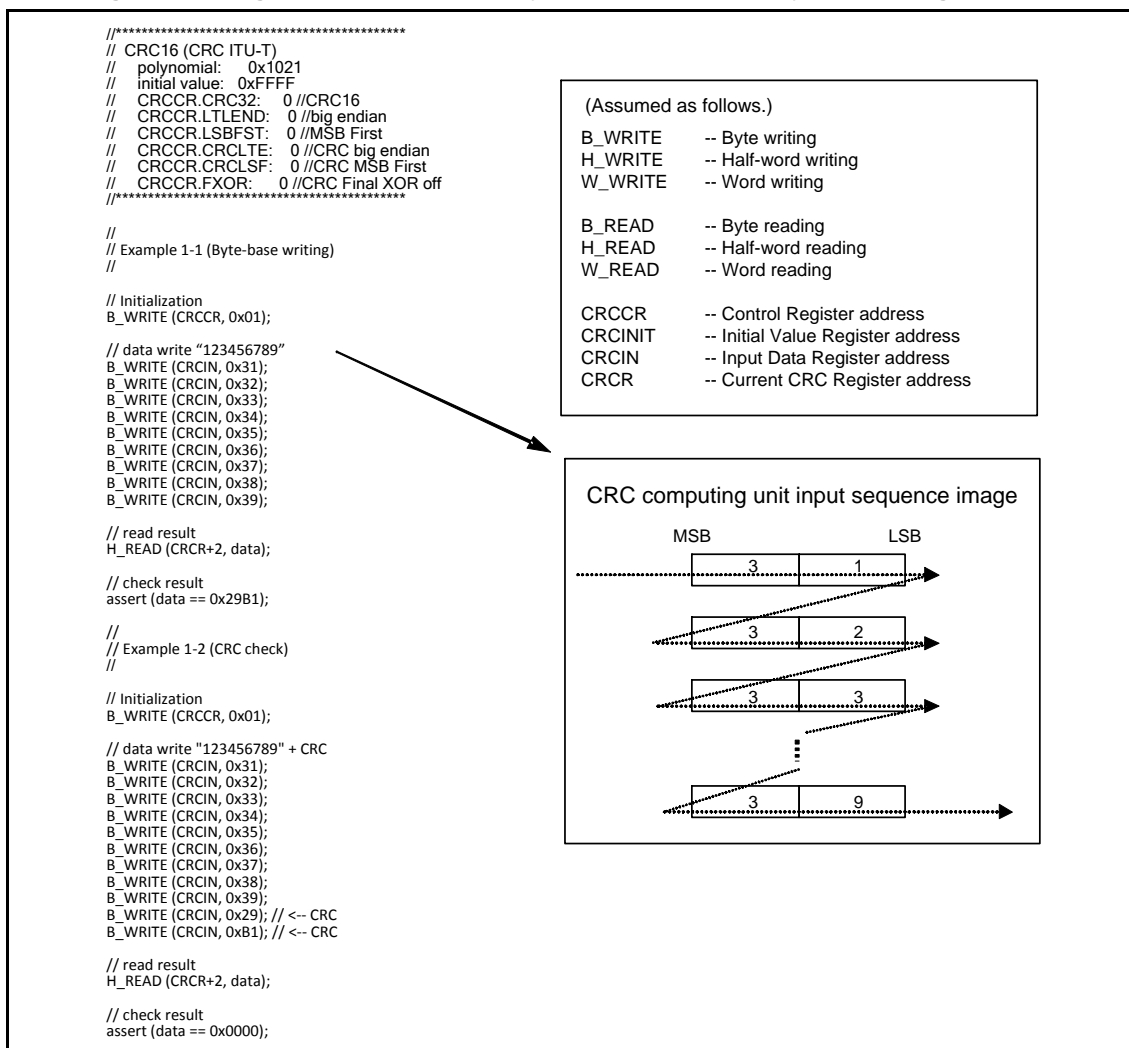
- For initialization, write "1" to the initialization bit (CRCCR:INIT). The value of the initial value register is loaded to the CRC register (CRCR).
- For input data writing, write to the Input Data register (CRCIN). The writing operation starts the CRC calculation. Also, during continuous writing or sequence, writings with different bit width are available.
- For obtaining the CRC code, read the CRC register (CRCR).

## 3.2. CRC Usage Examples

This section provides CRC usage examples.

### (1) Usage Example 1 CRC16, Byte Input Fixed

Figure 3-2 Usage Example 1 (CRC16, Byte Input Fixed, Core Byte Order: Big Endian)



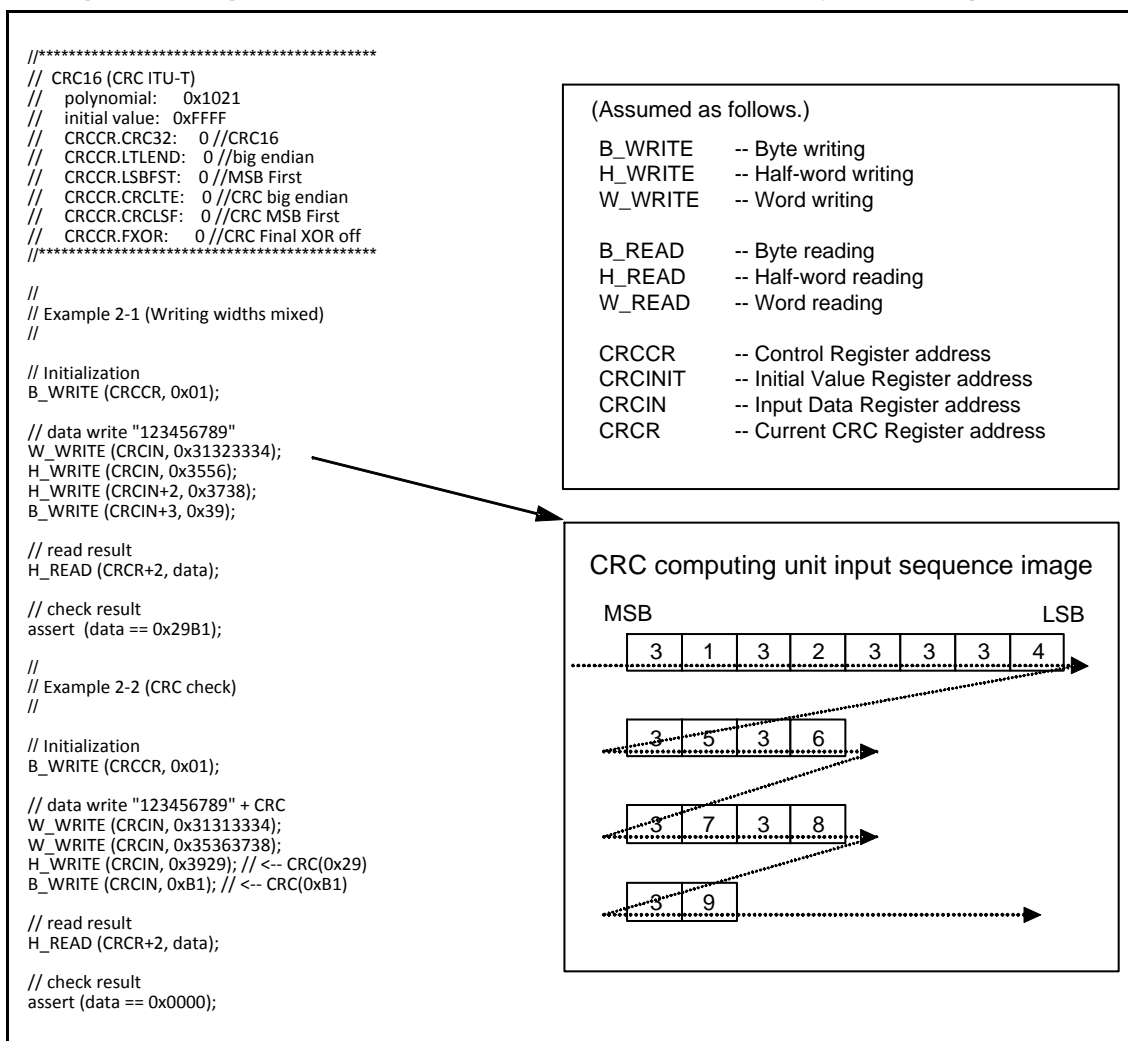
- The writing position of byte/half word is arbitrary. In this usage example, writing is continuous to the +0 position.
- CRC16 indicates the byte orders of the CPU and CRC result, the output position of CRCCR (CRC register), and the reading address in Table 3-1.

Table 3-1 CPU and CRC Result Byte Orders and CRCCR Reading Address

Core Byte Order	CRC Result Byte Order	Output Position in CRCCR	CRCCR H_READ Address
Big endian	Big endian	bit15 to bit0	CRCCR+2
Big endian	Little endian	bit31 to bit16	CRCCR+0
Little endian	Big endian	bit15 to bit0	CRCCR+0
Little endian	Little endian	bit31 to bit16	CRCCR+2

## (2) Usage Example 2 CRC16, Mixed Input Bit-width

Figure 3-3 Usage Example 2 (CRC16, Mixed Input Bit-width, Core Byte Order: Big Endian)



- When the settings of the byte order and bit order are correct and the bit input orders to the CRC arithmetic unit are the same, an arbitrary writing width can be set.  
For example, when the word writing is basically used, if a fraction, such as 1, 2, or 3 bytes is included at the end, it can support the case where byte and half word writings exist.

```

//*****
// CRC32 (IEEE-802.3)
// polynomial:      0x04C11DB7
// initial value:    0xFFFF_FFFF
// CRCCR.CRC32      1 // CRC32
// CRCCR.LTLEND:    0 // big endian
// CRCCR.LSBFST:    1 // LSB First
// CRCCR.CRCLTE:    0 // CRC big endian
// CRCCR.CRCLSF:    1 // CRC LSB First
// CRCCR.FXOR:      1 // CRC Final XOR on
//*****

//
// Example 3-1 (CRC32)
//

// Initialization
B_WRITE (CRCCR, 0x6B);

// data write "123456789"
W_WRITE (CRCIN, 0x31323334);
W_WRITE (CRCIN, 0x35363738);
B_WRITE (CRCIN, 0x39);

// read result
W_READ (CRCR, data);

// check CRC result
assert (data == 0x2639F4CB); // <- big endian & LSB First

```

(Assumed as follows.)

B_WRITE	-- Byte writing
H_WRITE	-- Half-word writing
W_WRITE	-- Word writing
B_READ	-- Byte reading
H_READ	-- Half-word reading
W_READ	-- Word reading
CRCCR	-- Control Register address
CRCIN	-- Initial Value Register address
CRCIN	-- Input Data Register address
CRCR	-- Current CRC Register address

CRC computing unit input sequence image

3	1	3	2	3	3	3	4
---	---	---	---	---	---	---	---

.....

3	5	3	6	3	7	3	8
---	---	---	---	---	---	---	---

.....

3	9
---	---

.....

↓

Head
CRC result (In macro)

9	B	6	3	D	0	2	C
---	---	---	---	---	---	---	---

.....

Head

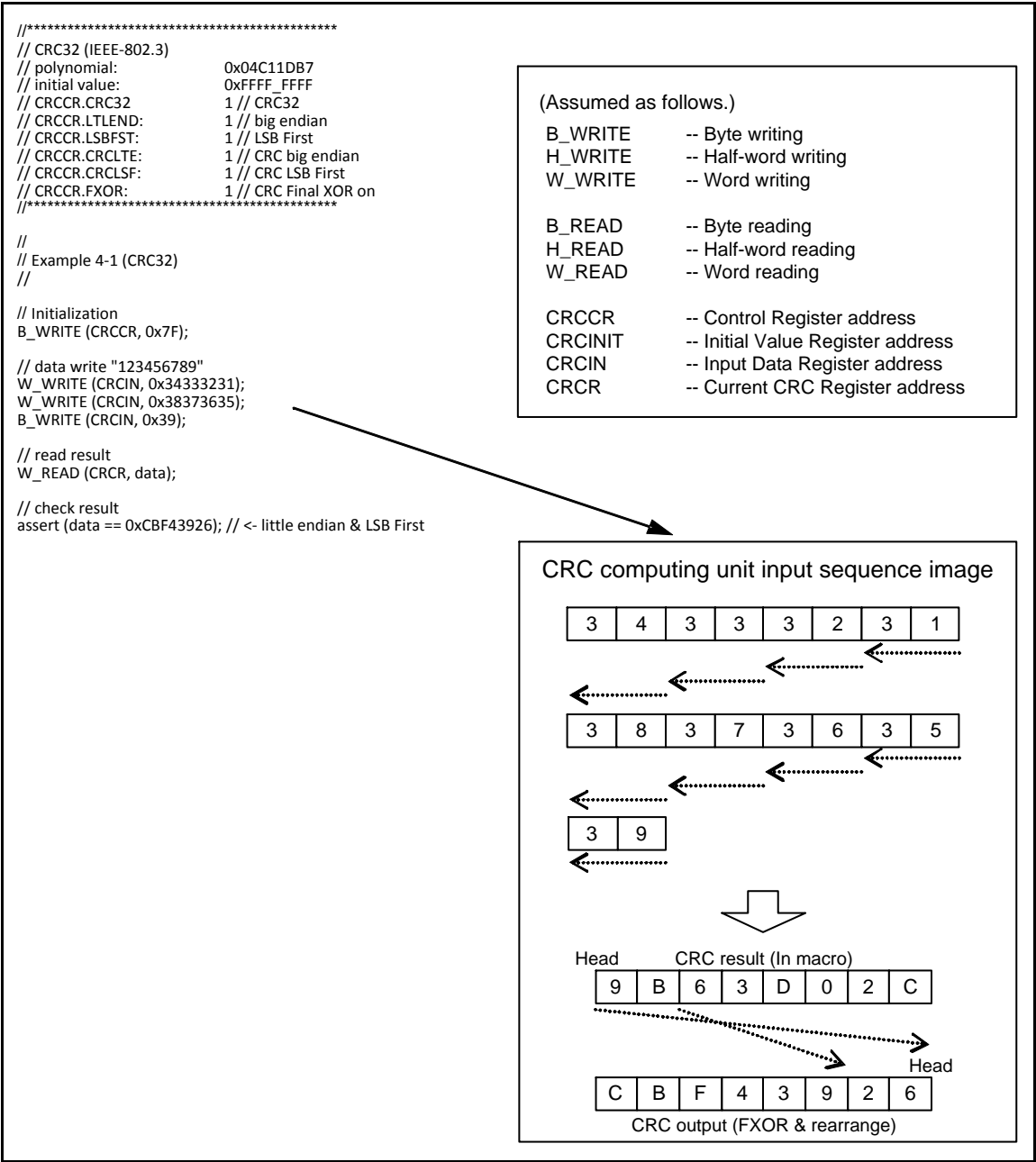
2	6	3	9	F	4	C	B
---	---	---	---	---	---	---	---

CRC output (FXOR & rearrange)

- MB9D560 MN708-00002-4v0-E, March 6, 2015

(4) Usage Example 4 CRC32, Byte Order: Little Endian

Figure 3-5 Usage Example 4 (CRC32, Byte Order: Little Endian)



- For CRC32 (IEEE-802.3), the bit order is set to LSB First, but the byte order can be set to either big endian or little endian. The above figure shows the case of little endian.
- If bit inverting of the CRC result is not required, perform either processing below to cancel the bit inverting.
  - Use 0x3F to perform initialization.
  - After entering data, set the CRCCR:FXOR bit to "0" (for example, set CRCCR = 0x3E).



## 4. Registers

This section describes the CRC registers.

The prefix CRCxx\_ is added to every register name (abbreviation). xx is the channel number (00 or 01).

Table 4-1 provides the register list.

**Table 4-1 List of CRC Registers**

Abbreviated Register Name	Register Name	See
CRCCR	CRC Control Register	4.1
CRCINIT	Initial Value Register	4.2
CRCIN	Input Data Register	4.3
CRCR	CRC Register	4.4

## 4.1. CRC Control Register (CRCCR)

This register controls the CRC calculations.

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved	FXOR	CRCLSF	CRCLTE	LSBFST	LTLEND	CRC32	INIT
ACCESS_TYPE	R0,W0	R/W	R/W	R/W	R/W	R/W	R/W	R0,W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

### [bit7] Reserved: Reserved bit

### [bit6] FXOR: Final XOR control bit

This bit performs XOR operation on the CRC result and the XOR value and outputs the result.

The XOR is set to 0xFFFFFFFF, and at FXOR = 1, the bits are inverted.

This bit is reflected in the CRC result read value soon after the setting is made, because of the processing of the subsequent part of the CRC register.

Value	Description
0	None
1	Available

### [bit5] CRCLSF: CRC result bit order setting bit

This bit rearranges the bits in the byte.

This bit is reflected in the CRC result read value soon after the setting is made, because of the processing of the subsequent part of the CRC register.

Value	Description
0	MSB First
1	LSB First

### [bit4] CRCLTE: CRC result byte order setting bit

This bit rearranges the byte order in the word.

This bit is reflected in the CRC result read value soon after the setting is made, because of the processing of the subsequent part of the CRC register.

For CRC16, if this bit is set to "1," the value is output to bit31 to bit16.

Value	Description
0	Big Endian
1	Little Endian





**[bit3] LSBFST: bit order setting bit**

This bit specifies the first bit of the byte (8 bits).

Combining with the LTLEND setting, 4 processing orders are available.

Value	Description
0	MSB First
1	LSB First

**[bit2] LTLEND: byte order setting bit**

This bit specifies the byte arrangement order in the writing width.

Value	Description
0	Big Endian
1	Little Endian

**[bit1] CRC32: CRC mode selection bit**

Value	Description
0	CRC16
1	CRC32

**[bit0] INIT: Initialization bit**

Writing "1" to this bit starts initialization.

This initialization loads the value of the initial value register to the CRC register.

Be sure to execute the initialization once at the beginning of the CRC calculation.

Value	Description	
	Write	Read
0	Invalid	"0" is always read.
1	Initialization	

## 4.2. Initial Value Register (CRCINIT)

This register stores the initial value of a CRC calculation.

BIT_OFFSET	31-0
BIT_NAME	D
ACCESS_TYPE	R/W
PROT_TYPE	-
INITIAL_VALUE	11111111_11111111_11111111_11111111

### [bit31:0] D[31:0]: CRC initial value bits

Value	Description
	Initial value of CRC calculation

For CRC16, use D[15:0], and ignore D[31:16].



### 4.3. Input Data Register (CRCIN)

Set the input data of a CRC calculation to this register.

BIT_OFFSET	31-0
BIT_NAME	D
ACCESS_TYPE	R/W
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000_00000000

**[bit31:0] D[31:0]: CRC input bits**

The bit widths of 8, 16, and 32 can be used, and different bit widths can be mixed.

Value	Description
	CRC calculation input data

The writing position of byte writing or half word writing is arbitrary. The available address positions are as listed below.

- Byte writing: +0, +1, +2, +3
- Half word writing: +0, +2

## 4.4. CRC Register (CRCR)

This register outputs the result of a CRC calculation. Be sure to perform initialization before starting the calculation.

BIT_OFFSET	31-0
BIT_NAME	D
ACCESS_TYPE	R,WX
PROT_TYPE	-
INITIAL_VALUE	11111111_11111111_11111111_11111111

### [bit31:0] D[31:0]: CRC result bits

These bits can read the results of CRC calculations. Writing "1" to the initialization bit (CRCCR:INIT) starts loading the value of the initial value register (CRCINIT).

When the input data of the CRC calculation is written to the Input Data register (CRCIN), after 1 peripheral clock cycle, the CRC calculation result is set. After all input data programming is completed, the final CRC code is retained.

For CRC16, when the byte order is set to big endian (CRCLTE="0"), the result is output to D[15:0]. When it is set to little endian (CRCLTE="1"), the result is output to D[31:16].

Value	Description
	CRC calculation result





## CHAPTER 30: CAN

This chapter explains the CAN.

---

1. Overview
2. Configuration
3. Operation
4. Registers



## 1. Overview

This section explains the configuration of CAN.

CAN consists of 3 chapters: CAN, CAN Controller, and CAN Message RAM ECC.

"CAN" explains the extended function.

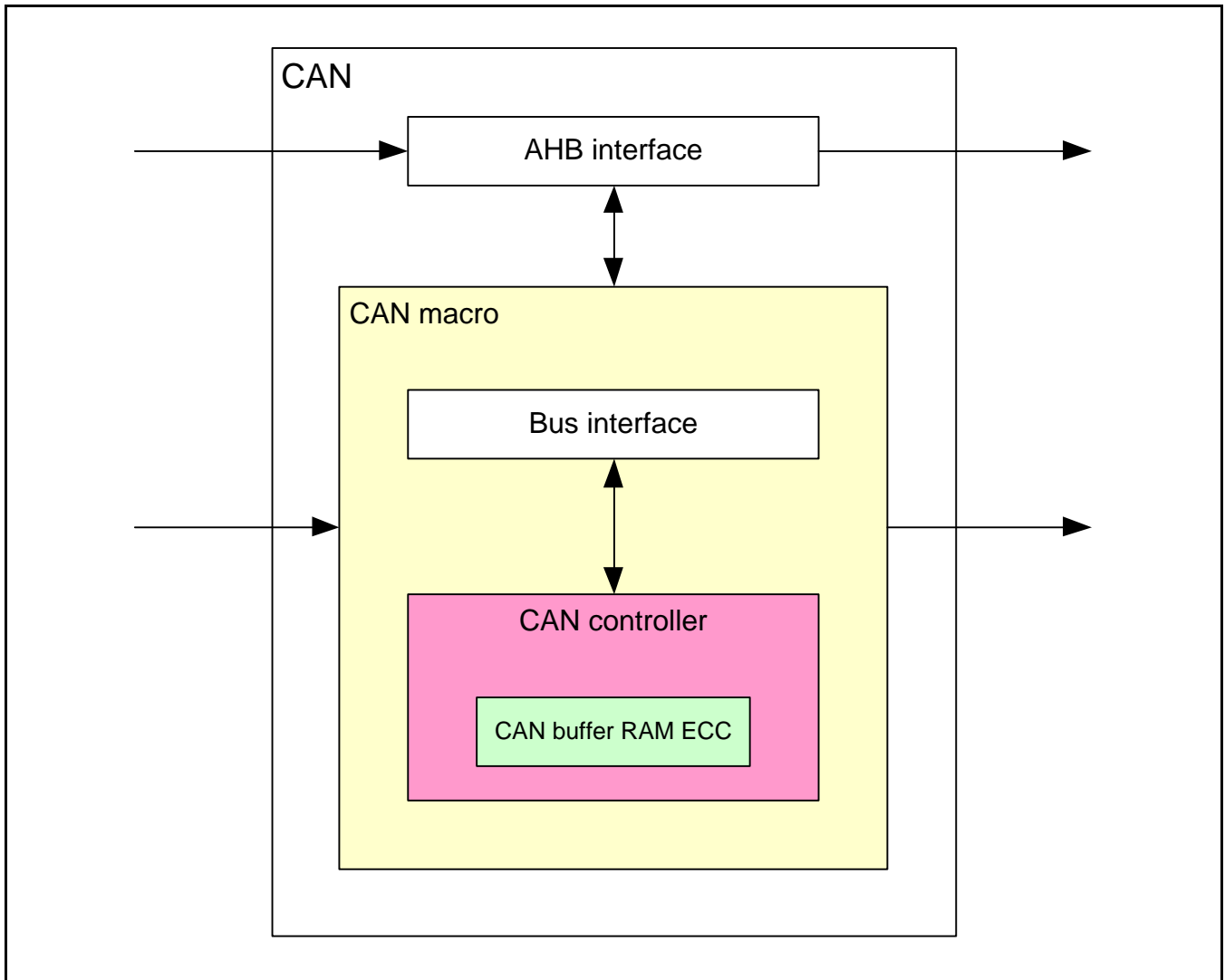
### **Notes:**

- *For details on the CAN controller, see "CHAPTER: CAN Contrpller".*
- *For details on the CAN message RAM ECC, see "CHAPTER: CAN Message RAM ECC".*

## 2. Configuration

This section provides a block diagram of the CAN.

Figure 2-1 CAN Block Diagram



- AHB Bus interface  
Performs signal conversion to connect the CAN macro through AHB.
- Bus interface  
Performs signal conversion to connect the CAN controller through AHB.
- CAN controller  
Controls the CAN.
- CAN buffer RAM ECC
  - Stores CAN message objects.
  - Detects and corrects data errors in the message RAM using the ECC function.





### 3. Operation

This section explains the extended function of the CAN.

This section explains the following functions.

- CAN interrupt request batch read

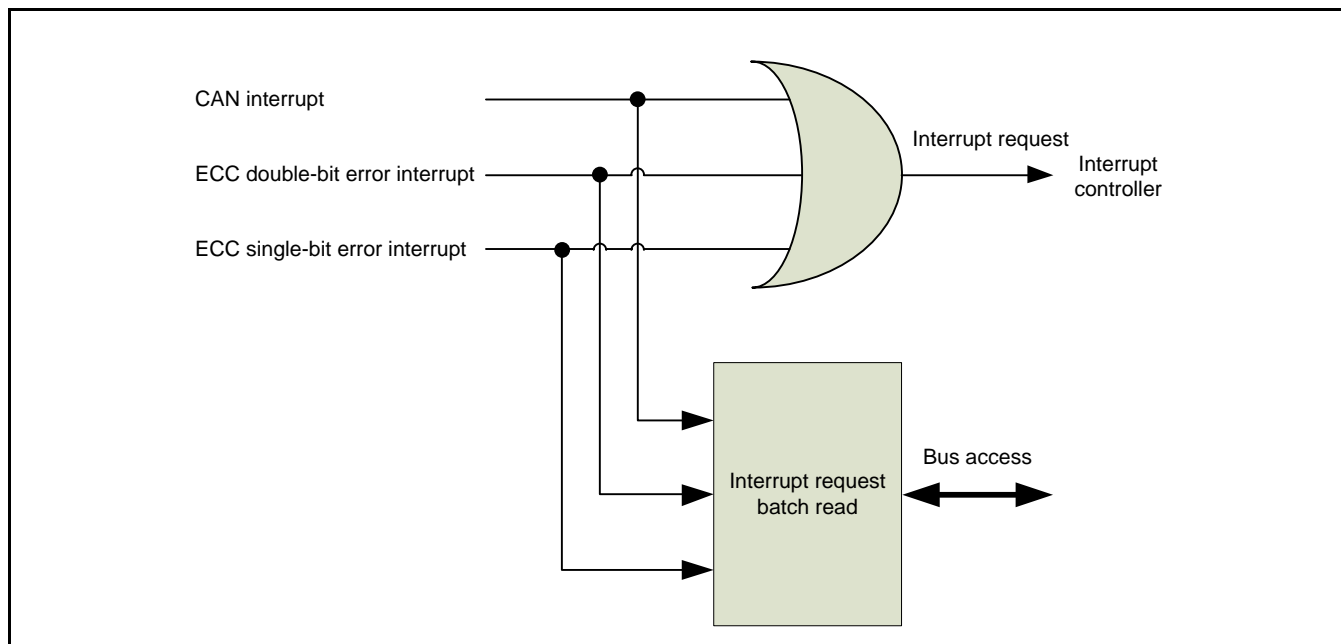
### 3.1. CAN Interrupt Request Batch Read

This section explains the CAN interrupt request batch read function.

The source of an interrupt request can be determined by locating the bit position set to "1" after reading the CAN interrupt request batch read register in the CAN interrupt handler.

Figure 3-1 shows the configuration of an interrupt request.

Figure 3-1 Configuration of Interrupt Request





## 4. Registers

This section explains the registers of the CAN.

The CAN includes the following registers.

- Overall Control Registers
- Message Interface Register
- Message Object Register
- Message Handler Registers
- Message RAM ECC Register
- Extended function register

1024 bytes of address space is assigned to these registers allowing for word, half-word and byte access.

**Notes:**

- *For details on the Overall Control Registers, Message Interface Register, Message Object Register, and Message Handler Registers, see "CHAPTER: CAN Controller".*
- *For details on the Message RAM ECC Register, see "CHAPTER: CAN Message RAM ECC".*

**Table 4-1 Extended Function Registers**

Abbreviated Register Name	Register Name	See
CANxx_CIRRR	CAN Interrupt Request Batch Read Register	4.1

xx=00, 01, 02

## 4.1. CAN Interrupt Request Batch Read Register (CANxx\_CIRRR)

The CAN interrupt and ECC interrupt (double-bit error interrupt and single-bit error interrupt) are assigned to a single interrupt vector. By reading the CAN interrupt request batch read register, you can confirm the interrupt which has occurred.

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved					ECCSEI	ECCDEI	CANINT
ACCESS_TYPE	R0,WX					R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	00000					0	0	0

**[bit7:3] Reserved: Reserved bits**

### **[bit2] ECCSEI: Single-bit error interrupt bit**

This bit indicates the state of a single-bit error interrupt request of the CAN message RAM ECC.

Value	Description
0	Indicate no interrupt request.
1	Indicate that an interrupt request is issued.

### **[bit1] ECCDEI: Double-bit error interrupt bit**

This bit indicates the state of a double-bit error interrupt request of the CAN message RAM ECC.

Value	Description
0	Indicate no interrupt request.
1	Indicate that an interrupt request is issued.

### **[bit0] CANINT: CAN interrupt bit**

This bit indicates the state of a CAN interrupt request.

Value	Description
0	Indicate no interrupt request.
1	Indicate that an interrupt request is issued.





## CHAPTER 31: CAN Controller

This chapter explains CAN controller.

---

1. Overview
2. Configuration
3. Operation
4. Registers



## 1. Overview

The CAN controller complies with CAN protocol version 2.0A/B, a standard protocol for serial communication. CAN is widely used in various industrial fields such as automobile and factory automation.

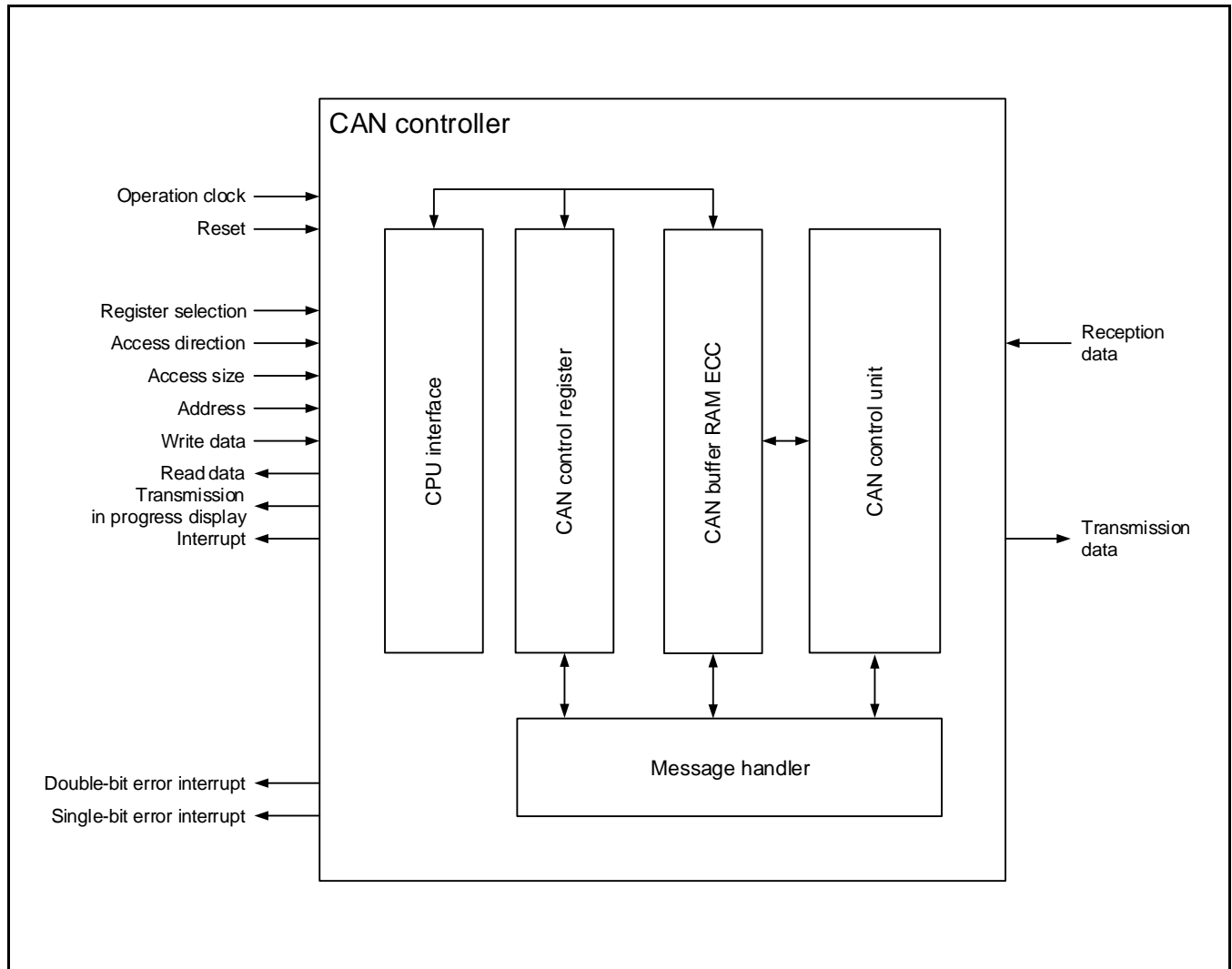
The CAN controller has the following features:

- Supports CAN protocol version 2.0A/B
- Supports a bit rate up to 1 Mbits/s
- Identifier mask for each message object
- Supports programmable FIFO mode (a chain of message objects)
- Maskable interrupt
- Supports 64 message objects
- Supports programmable loop-back mode for self-test operation
- Read and write from/to the message object using message interface registers

## 2. Configuration

Figure 2-1 shows the block diagram of the CAN controller.

Figure 2-1 Block Diagram of CAN Controller



- CAN control unit  
Controls the CAN protocol and the shift registers for serial/parallel conversion to transfer send/received messages.
- Message RAM  
Stores message objects.
- Registers  
All registers used by CAN.
- Message handler  
Controls the message RAM and CAN control unit.
- CPU interface  
Controls the internal bus interface.





### 3. Operation

This section explains the operations and functions of the CAN controller.

Following functions are included:

- Message objects
- Message transmission
- Message reception
- FIFO buffer function
- Interrupt function
- Bit timing
- Test mode
- Software initialization

### 3.1. Message Objects

The following explains message objects and the interface of the message RAM.

#### (1) Message Objects

The configuration of message objects in the message RAM (excluding the MsgVal, NewDat, IntPnd, and TxRqst bits) is not initialized by a hardware reset. Initialize the message objects by one of the following steps after hardware reset.

1. Initialize all message objects by the CPU.
2. Initialize only the used message objects by CPU. Clear MsgVal bit to "0" in the other message objects.

Configure the CAN Bit Timing Register while the Init bit in the CAN Control Register is "1".

A message object must be configured by programming message interface registers (the IFx Command Mask Register, IFx Mask Register, IFx Arbitration Register, IFx Message Control Register, and IFx Data Register), and then writing a message number to the corresponding IFx Command Request Register. By writing the message number, the message interface register data will be transferred to the addressed message object.

When the Init bit in the CAN Control Register is cleared to "0", the CAN controller starts operation. The received data that have passed acceptance filtering are stored into the message RAM. Messages with pending transmission requests are transferred from the message RAM to the shift register in the CAN controller, and then sent to the CAN bus.

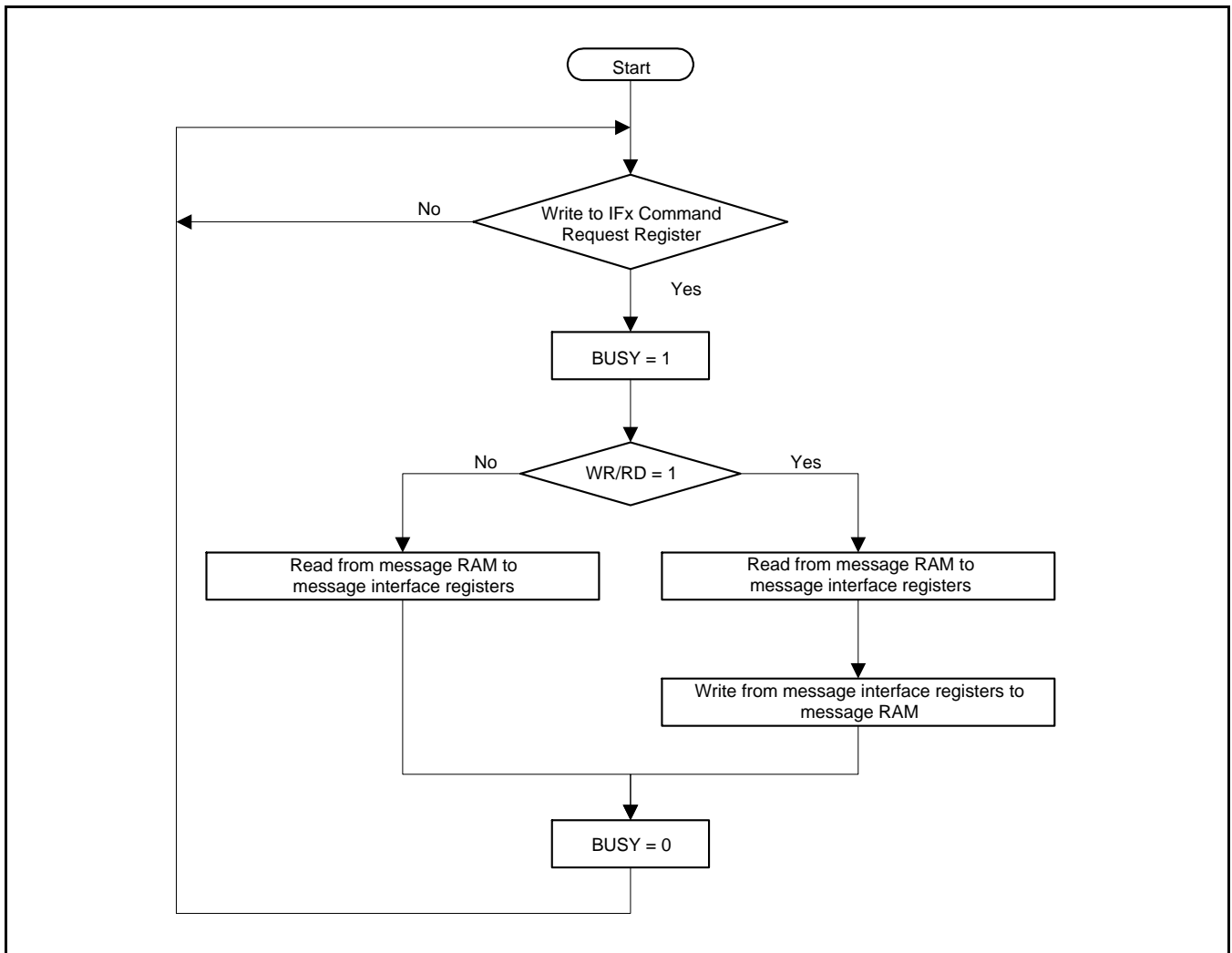
The CPU reads the received messages and updates outgoing messages via message interface registers. The CPU is interrupted according to the configuration of the CAN Control Register and IFx Message Control Register (message object).

#### (2) Data Transfer from/to Message RAM

When data transfer starts between the message interface registers and message RAM, the BUSY bit in the IFx Command Request Register is set to "1". After the transfer has finished, the BUSY bit is cleared to "0". (See Figure 3-1)

The IFx Command Register selects whether to transfer complete data or only partial data of one message object. The structure of the message RAM does not allow the writing of single bits/bytes of one message object. The complete data of one message object is always written to the message RAM. Therefore, the data from the message interface registers to the message RAM is transferred in a read-modify-write cycle.

Figure 3-1 Data Transfer between the Message Interface Registers and Message RAM



## 3.2. Message Transmission

The following explains how to configure the send message objects, and about the transmission.

### (1) Sending Messages

If there is no data transfer between the message interface registers and message RAM, the MsgVal bit in the CAN Message Valid Register and the TxRqst bit in the CAN Transmit Request Register are evaluated. A valid message object with the highest priority of pending transmission requests is transferred to the shift register for transmission. Then the NewDat bit of the message object is reset to "0".

When the transmission has finished successfully, and if there is no new data in the message object (NewDat = "0"), the TxRqst bit is reset to "0". If TxIE is set to "1", then the IntPnd bit is set to "1" after a successful transmission. If the CAN controller lost the arbitration on the CAN bus, or if an error occurred during transmission, the message is resent immediately when the CAN bus becomes idle.

### (2) Transmission Priority

The transmission priority of the message objects is determined by the message number. Message object 1 has the highest priority, while message object 64 (the largest number of the installed message objects) has the lowest priority. If two or more transmission requests are pending, they are transferred in the order of corresponding message number from smallest to largest.

#### Notes:

- *In case of the following procedures, the messages may not be sent until any of the events described below occurs.*
- *Procedures :*
  - 1. *A message object with the lowest priority is used for transmission.*
  - 2. *The TxRqst bit was previously set to "1", but is set to "0" to abort transmission.*
  - 3. *The TxRqst bit is set to "1" again after (2).*
- *Events :*
  - *A valid message flows on the CAN bus.*
  - *A transmission request is issued to another message object.*
  - *CAN is initialized by the Init bit.*
- *If canceling the transmission is required to suit system operations, execute the following steps.*
  - 1. *Execute one of the following steps.*
    - *Do not use a message object with the lowest priority as a send message object.*
    - *After aborting the transmission, generate any of the above events.*
  - 2. *Set the TxRqst bit to "1" again.*
    - *If the ID, DLC, Xtd, and Data7-Data0 in message objects are changed while the TxRqst bit is "1", message objects before and after the change are mixed for transmission, or the message objects after the change may not be transmitted. Therefore, be sure to change them while the TxRqst bit is "0".*



### (3) Configuring a Send Message Object

Table 3-1 shows how a send object should be initialized.

**Table 3-1 Initialization of Send Message Object**

MSgVal	Arb	Data	Mask	EoB	Dir	NewDat	MsgLst	RxIE	TxIE	IntPnd	RmtEn	TxRqst
1	appl.	appl.	appl.	1	1	0	0	0	appl.	0	appl.	0

The IFx Arbitration Register (ID and Xtd bit), given by the application, defines the ID and the type of the outgoing message.

If the standard frame (11-bit ID) is set, then ID[28:18] are used, and ID[17:0] are ignored. If the extended frame (29-bit ID) is set, then ID[28:0] are used.

If TxIE bit is set to "1", then the IntPnd bit is set to "1" after a successful transmission of the message object.

If the RmtEn bit is set to "1", the TxRqst bit is set to "1" after receiving the corresponding remote frame, and a data frame is sent automatically.

The data register (DLC, Data0 to Data7) settings are given by the application.

When UMask is set to "1", the IFx Mask Register (Msk, MXtd, and MDir bits) is used to receive remote frames with the IDs grouped by the mask setting, and then enable the transmission (by setting the TxRqst bit to "1"). For details, see "(4) Remote Frame" in Section "3.3. Message Reception".

**Note:**

- The MDir bit in the IFx Mask Register must be set to "1".

### (4) Updating a Send Message Object

The CPU can update the data of a send message object via the message interface registers.

The send message object data is written by four bytes of the corresponding IFx data register (in the unit of IFx data register A or IFx data register B). Therefore, the send message object cannot be changed by a single byte.

To update 8-byte data, write 0x0087 to the IFx Command Mask Register, and the message number to the IFx Command Request Register. This concurrently updates the send message object data (of 8-byte) and write "1" to the TxRqst bit.

If both the NewDat and TxRqst bits are set to "1", the NewDat bit is reset to "0" once the transmission is started.

**Notes:**

- To update data, update it by four bytes of the IFx Data Register A or IFx Data Register B.
- If the ID, DLC, Xtd, and Data7 to Data0 in message objects are changed while the TxRqst bit is "1", message objects before and after the change are mixed for transmission, or the message objects after the change may not be transmitted. Therefore, be sure to change them while the TxRqst bit is "0".

### 3.3. Message Reception

The following explains how to configure the receive message object and about the reception.

#### (1) Acceptance Filtering for Received Messages

When the arbitration and control field (ID + IDE + RTR + DLC) of a message is completely shifted into the shift register of the CAN controller, scanning of the message RAM is started to compare matching with a valid message object.

Then the arbitration field and mask data (including MsgVal, UMask, NewDat, and EoB) are loaded from a message object in the message RAM, and the message object is compared with the arbitration field of the shift register including mask data.

This operation is repeated "until a matching is detected between a message object and the arbitration field of the shift register", or "until the last word of the message RAM is reached". When a matching is detected, scanning of the message RAM is stopped, and the CAN controller processes data depending of the type of the received frame (data frame or remote frame).

#### (2) Reception Priority

The reception priority of the message objects is determined by the message number. Message object 1 has the highest priority, while message object 64 (the largest number of the installed message objects) has the lowest priority. If two or more objects are matched in the acceptance filtering, therefore, the object with the smallest message number becomes the receive message object.

#### (3) Data Frame Reception

The CAN controller transfers the received message from the shift register into the message RAM of the message object matched in the acceptance filtering. The stored data includes all arbitration fields and the data length code as well as data bytes. This is implemented (to keep the ID and the data bytes) even if the IFx Mask Register is used for masking.

The NewDat bit is set to "1" upon the reception of new data. When the CPU reads the message object, reset the NewDat bit to "0". If the NewDat bit has already been set to "1" upon the reception of a message, the MsgLst is set to "1" indicating that the previous data was lost.

If the RxIE bit has been set to "1", reception of a message object causes the IntPnd bit in the CAN Interrupt Pending Register to be set to "1". Then the TxRqst bit of the message object is reset to "0". This is implemented to prevent transmission of a remote frame when the requested data frame is received during the transmission.

#### (4) Remote Frame

One of the following three operations is selected when a remote frame is received. The selection depends on how the matching message object is configured.

1. Dir = "1" (Direction = Send), RmtEn = "1", UMask = "1" or "0"

Receives the matched remote frame, sets only the TxRqst of this message object to "1", and automatically replies (sends) data frame to the remote frame. (Other than the TxRqst bit, the message object remains unchanged.)

2. Dir = "1" (Direction = Send), RmtEn = "0", UMask = "0"

Does not receive an incoming remote frame, even if it matches the message object, and disables the remote frame. (The TxRqst bit of the message object remains unchanged.)

3. Dir = "1" (Direction = Send), RmtEn = "0", UMask = "1"

If an incoming remote frame matches the message object, the TxRqst bit of the message object is set to "0", and the remote frame is handled as if it were a received data frame. The received arbitration field and control field (ID + IDE + RTR + DLC) are stored into the message object in the message RAM, and the NewDat bit of this message object is set to "1". The data field of the message object remains unchanged.



## (5) Configuring a Receive Message Object

Table 3-2 shows how a receive message object should be initialized.

**Table 3-2 Initialization of a Receive Message Object**

MSgVal	Arb	Data	Mask	EoB	Dir	NewDat	MsgLst	RxIE	TxIE	IntPnd	RmtEn	TxRqst
1	appl.	appl.	appl.	1	0	0	0	appl.	0	0	0	0

The IFx Arbitration Register (ID and Xtd bit) is given by the application. The register defines the ID and the type of a received message, used for the acceptance filtering.

If the standard frame (11-bit ID) is set, then ID[28:18] are used, and ID[17:0] are ignored. When a standard frame is received, ID[17:0] are reset to "0". If the extended frame (29-bit ID) is set, then ID[28:0] are used.

When the RxIE has been set to "1", and when a received data frame is stored into the message object, then the IntPnd bit is set to "1".

The data length code (DLC) is given by the application. When the CAN controller stores the received data frame into the message object, it stores the received data length code and eight bytes data. If the data length code is less than eight, unspecified data is written to the remaining bytes of the message object.

When UMask is set to "1", the IFx Mask Register (Msk, MXtd, and MDir bits) is used to allow the reception of data frames with the IDs grouped by the mask setting. For details, see "(3) Data Frame Reception" in Section "3.3. Message Reception".

### **Note:**

- The MDir bit in the IFx Mask Register must be set to "1".

## (6) Handling a Received Message

The CPU can read a received message any time via the message interface registers.

The following shows an example of handling a received message. Write "0x007F" to the IFx Command Register, and a message number of the message object to the IFx Command Request Register. This procedure transfers a received message of the specified message number from the message RAM to the message interface registers. Then the NewDat bit and IntPnd bit of the message object can be cleared to "0" according to the configuration of the IFx Command Mask Register.

An incoming message is received if it is matched in the acceptance filtering. If the message object uses a mask for acceptance filtering, the masked data is excluded from the acceptance filtering to determine whether or not the message should be received.

The NewDat bit indicates whether a new message has been received since the last time the message object was read.

The MsgLst bit indicates that the previous received data was lost because the next data is received before the previous data is read from the message object. The MsgLst bit is not automatically reset.

During transmission of a remote frame, if a data frame matched in the acceptance filtering is received, the TxRqst bit is automatically reset to "0".

### 3.4. FIFO Buffer Function

The following explains the configuration of a FIFO buffer of the message object and its operations in handling received messages.

#### (1) Configuration of a FIFO Buffer

The configuration of the receive message object belonging to a FIFO buffer is the same as that of a receive message object except the EoB bit. (See "(5) Configuring a Receive Message Object" in Section "3.3. Message Reception".)

A FIFO buffer is used by concatenating two or more receive message objects. To store received messages into this FIFO buffer, the ID and the mask settings of the receive message objects must be matched when they are used.

Receive data is sequentially stored from the message object with lower message number of message object in FIFO buffer. In the last receive message object of the FIFO buffer, set "1" to the EoB bit to indicate that the object is the end of the FIFO buffer. (Except in the last message object, the EoB bit in each message object that uses the FIFO buffer configuration must be set to "0".)

#### Notes:

- Be sure to configure the same settings for the ID and the masks of message objects used in the FIFO buffer.
- When the FIFO buffer is not used, be sure to set the EoB bit to "1".

#### (2) Receiving Messages Using FIFO Buffers

When a received message matches with the FIFO buffer ID, it is stored into the message object with the lowest message number of the FIFO buffer.

When a message is stored into the receive message object in the FIFO buffer, the NewDat bit of this receive message object is set to "1". When the NewDat bit is set in receive message object while the EoB bit is set to "0", the receive message object is protected until the last receive message object (with EoB bit = "1") is reached. Meanwhile, the CAN controller does not write to the FIFO buffer.

When both of the following conditions are met, the next incoming message is written to the last message object and therefore overwrites the previous message.

- Valid data is stored into the last FIFO buffer
- The NewDat bit of the receive message object is not written by "0" (to release the write protect)



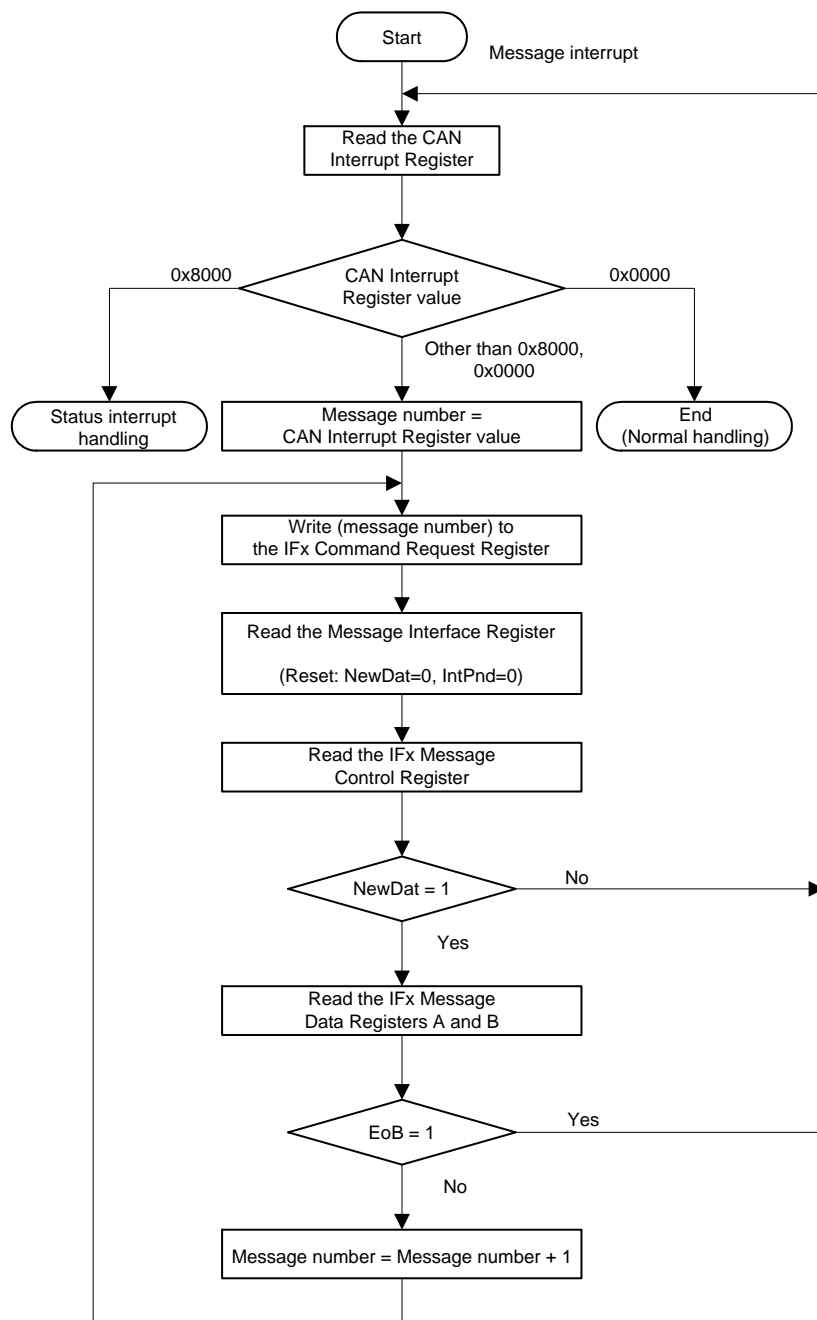
### (3) Reading from a FIFO Buffer

To read the contents of a receive message object, the CPU transfers the object to the Message Interface Register by writing the received message number to the IFx Command Request Register. Then, set WR/RD in the IFx Command Mask Register to "0" (read), set TxRqst/NewDat = 1, CIP = 1, and set the NewDat bit and IntPnd bit to "0".

To assure the correct FIFO buffer function, be sure to first read a receive message object in the FIFO buffer with the lowest message number, and then other objects in ascending order.

Figure 3-2 shows how the CPU handles the message objects the FIFO buffer concatenates.

Figure 3-2 CPU handling of FIFO Buffer



### 3.5. Interrupt Function

The following explains the interrupt handling using the status interrupt (IntId = 0x8000) and message interrupt (IntId = Message number).

If two or more interrupts are pending, the CAN Interrupt Register points to a pending interrupt code with the highest priority. The chronological order of the interrupt codes are neglected, and the interrupt code with the highest priority is always shown. The interrupt code is retained until the CPU clears it.

The status interrupt (0x8000 of the IntId bit) has the highest priority.

Priority of message interrupts is determined by the message number. A smaller number has a higher priority while the larger the lower.

A message interrupt is cleared by clearing the IntPnd bit of the message object. A status interrupt is cleared by reading the CAN Status Register.

The IntPnd bit in the CAN interrupt Pending Register indicates whether an interrupt of message objects has been caused. When no interrupts are pending, the IntPnd bit retains "0".

While the IE bit in the CAN Control Register, and the TxIE bit and RxIE bit in the IFx Message Control Register are set to "1", if the IntPnd bit turns to "1", then the interrupt line to the CPU becomes active. The interrupt line remains active until the CAN Interrupt Pending Register is cleared to "0" (the interrupt cause is reset) or the IE bit in the CAN Control Register is reset to "0".

The 0x8000 value of the CAN Interrupt Register indicates that the CAN Status Register has been updated by the CAN controller. This interrupt has the highest priority. The interrupt by updating the CAN Status Register can enable or disable the setting of the CAN Interrupt Register using the EIE bit and SIE bit in the CAN Control Register. The interrupt line to the CPU can be controlled by the IE bit in the CAN Control Register.

A write access from the CPU can update (reset) the RxOk bit, TxOk bit, and LEC bit in the CAN Status Register. However, the write access cannot generate or reset an interrupt.

Except the 0x8000 and 0x0000 values, the CAN Interrupt Register indicates that a message interrupt is pending, and that the interrupt has the highest priority.

The CAN Interrupt Register is updated even when IE is reset.

The source of a message interrupt to the CPU can be checked from the CAN Interrupt Register or CAN Interrupt Pending Register. (See Section "4.4. Message Handler Registers") When clearing a message interrupt, the message data can be read concurrently. If a message interrupt indicated by the CAN Interrupt Register is cleared, the CAN Interrupt Register sets another interrupt with the next higher priority. This waits for the next interrupt handling. If no interrupts are pending, the CAN Interrupt Register shows the 0x0000 value.

**Note:**

- A write access to the CAN status register will not generate a status interrupt (IntId = 0x8000).

### 3.6. Bit Timing

The following provides the overview of the bit timing and explains about the bit timing in the CAN controller.

Each CAN node in the CAN network has its own clock generator (usually a quartz oscillator). The time parameter of the bit time can be configured individually for each CAN node. Even if each CAN node's oscillator has a different period, a common bit rate can be generated.

The oscillator frequencies vary slightly because of changes in temperature or voltage, or deterioration of components. As long as the frequencies vary only within the tolerance range of the oscillators, the CAN nodes can compensate for the different bit rates by resynchronizing to the bit stream.

The bit time can be divided into four segments according to the CAN specifications (see Figure 3-3), into the synchronization segment (Sync\_Seg), the propagation time segment (Prop\_Seg), the phase buffer segment 1 (Phase\_Seg1), and the phase buffer segment 2 (Phase\_Seg2). Each segment consists of the programmable number of time quanta (See Table 3-3). The basic unit of the time quantum (tq) is defined by CAN controller's system clock "fsys" and the baud rate prescaler (BRP).

$$tq = BRP / f_{sys}$$

CAN's system clock "fsys" is the frequency of its clock input. Synchronization segment Sync\_Seg is a timing in the bit time where edges of the CAN bus level are expected to occur. Propagation time segment Prop\_Seg compensates for the physical delay times within the CAN network. Phase buffer segments Phase\_Seg1 and Phase\_Seg2 define the sampling point. The resynchronization jump width (SJW) defines the width within which resynchronization can move the sampling point to compensate for edge phase errors.

Figure 3-3 Bit Timing

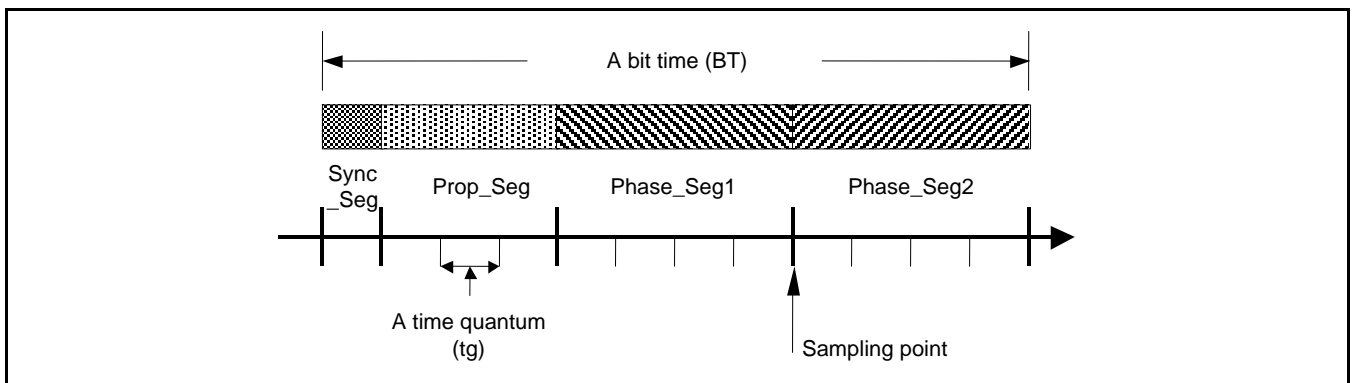
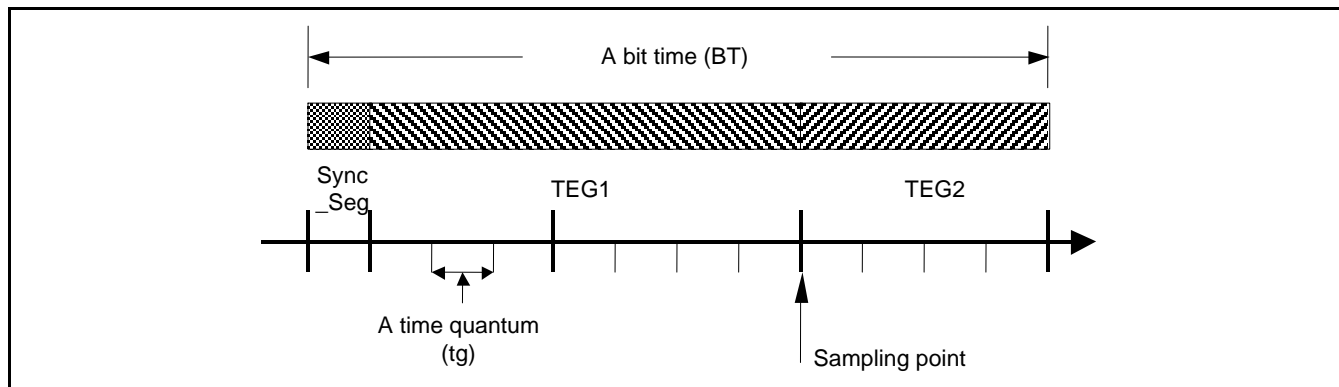


Table 3-3 CAN Bit Time Parameters

Parameter	Range	Function
BRP	[1-32]	Defines the length of time quantum tq.
Sync_Seg	1 tq	Fixed length. Synchronization to system clock.
Prop_Seg	[1-8] tq	Compensates for the physical delay times.
Phase_Seg1	[1-8] tq	Assures edge phase errors before the sampling point. May be prolonged temporarily by synchronization.
Phase_Seg2	[1-8] tq	Assures edge phase errors after the sampling point. May be shortened temporarily by synchronization.
SJW	[1-4] tq	Resynchronization jump width. Will not be longer than either of the phase buffer segments.

The following shows the bit timing in the CAN controller.

**Figure 3-4 The Bit Timing in the CAN Controller**



**Table 3-4 CAN Controller Parameters**

Parameter	Range	Function
BRPE, BRP	[0-1023]	Defines the length of time quantum tq. Can extend the prescaler by up to 1024 by the CAN Bit Timing Register and the CAN Prescaler Extension Register.
Sync_Seg	1 tq	Synchronization to system clock. Fixed length.
TSeg1	[1-15] tq	A time segment before the sampling point. Equivalent to the sum of Prop_Seg and Phase_Seg1. Can be controlled by the CAN Bit Timing Register.
TSeg2	[0-7] tq	A time segment after the sampling point. Equivalent to Phase_Seg2. Can be controlled by the CAN Bit Timing Register.
SJW	[0-3] tq	Resynchronization jump width. Can be controlled by the CAN Bit Timing Register.

The following shows the relations among the parameters:

$$\begin{aligned}
 tq &= ([BRPE, BRP] + 1) / f_{sys} \\
 BT &= Sync\_Seg + TEG1 + TEG2 \\
 &= (1 + (TSeg1 + 1) + (TSeg2 + 1)) \times tq \\
 &= (3 + TSeg1 + TSeg2) \times tq
 \end{aligned}$$

### 3.7. Test Mode

The following explains how to configure test mode, and about its operations.

#### (1) Test Mode Setting

Test mode is entered by setting the Test bit in the CAN Control Register to "1". In test mode, the Tx1, Tx0, LBack, Silent, and Basic bits in the CAN Test Register are enabled.

When the Test bit in the CAN Control Register is set to "0", all CAN Test Register functions (except Rx bit) are disabled.

#### (2) Silent Mode

The CAN controller can be set in Silent mode by programming the Silent bit in the CAN Test Register to "1".

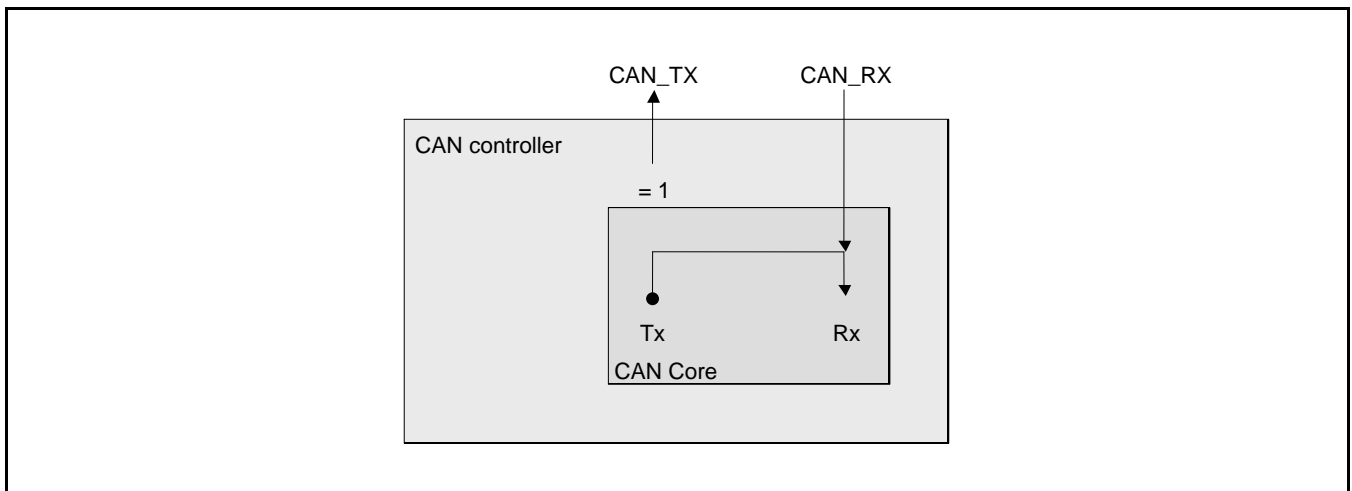
In Silent mode, the CAN controller can receive data frames and remote frames, but only outputs recessive bits onto the CAN bus and does not send messages and ACK.

When the CAN controller is required to send dominant bits (ACK bits, overload flags, active error flags), the CAN controller uses the internal rerouting circuit to send them to the RX side. In this operation, the RX side can receive dominant bits rerouted inside the CAN controller even when the CAN bus remains in a recessive state.

In Silent mode, the analysis of CAN bus traffic is possible without being affected by transmission of the dominant bits (ACK bits, error flags).

Figure 3-5 shows the connection of the CAN\_TX and CAN\_RX signals to the CAN controller in Silent mode.

Figure 3-5 CAN Controller in Silent Mode



### (3) Loop Back Mode

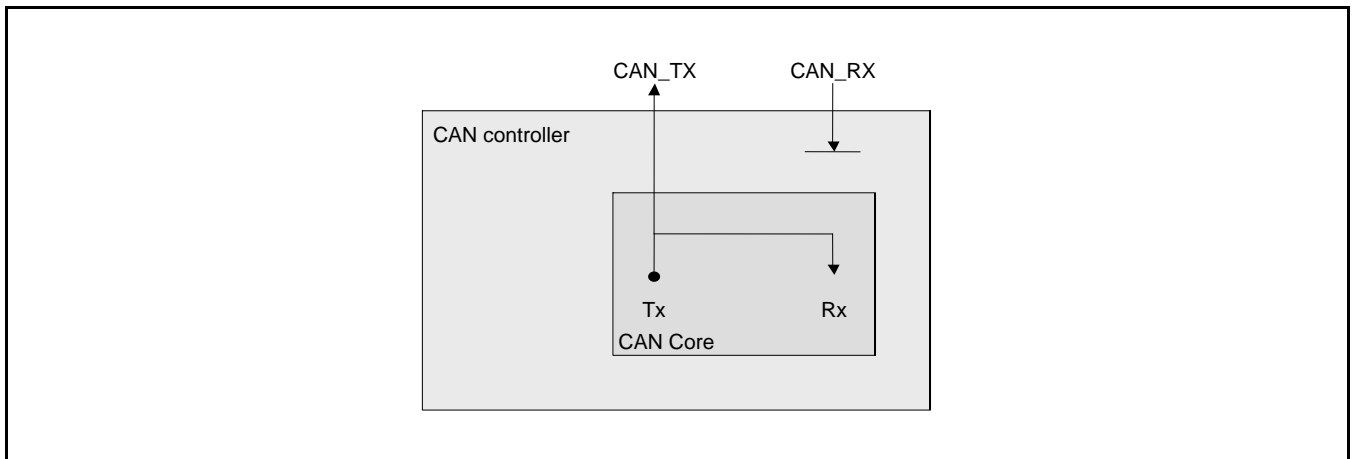
The CAN controller can be set in Loop back mode by programming the LBack bit in the CAN Test Register to "1".

Loop back mode can be used for self-diagnostic functions.

In Loop back mode, TX is connected with RX inside the CAN controller. The CAN controller treats the transmitted messages as messages received by RX, and stores the messages passed acceptance filtering into the receive buffer.

Figure 3-6 shows the connection of the CAN\_TX and CAN\_RX signals to the CAN controller in Loop back mode.

**Figure 3-6 CAN Controller in Loop Back Mode**



**Note:**

- In Loop back mode the CAN controller is independent of external signals, does not sample dominant bits in the acknowledgement slot of a data/remote frame. This usually causes the CAN controller to generate acknowledgement errors. In this test mode, however, the errors are not cased.

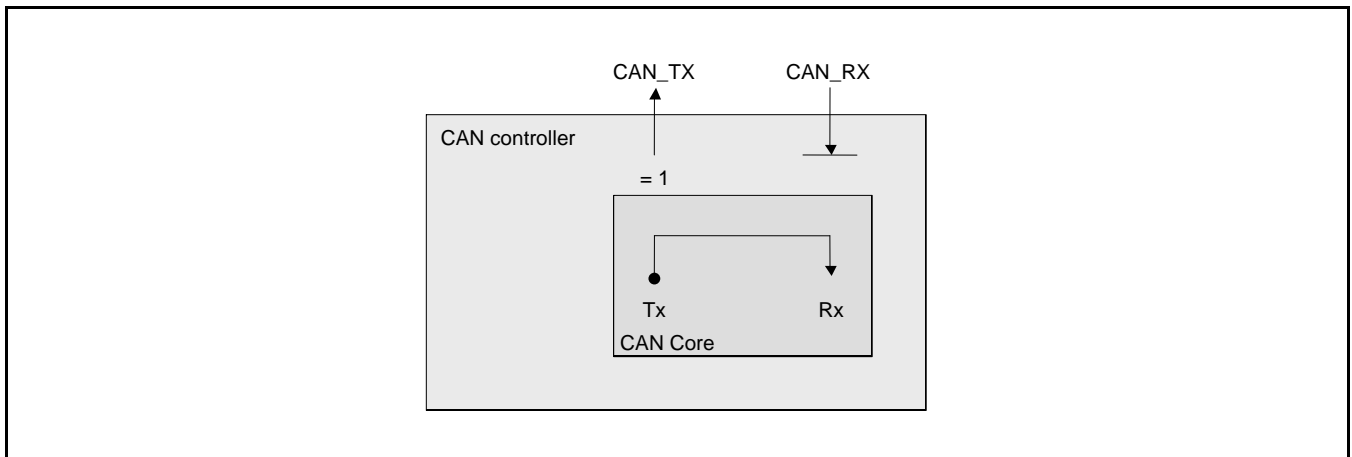
#### (4) Combination of Silent Mode and Loop Back Mode

Loop back mode and Silent mode can be combined by setting the LBack bit and Silent bit in the CAN Test Register to "1" at the same time.

This mode can be used for "Hot self-test". The "Hot self-test" means that the CAN controller can be tested in Loop back mode without affecting operation of the CAN system, because a constant recessive value is output from the CAN\_TX signal and the input to the CAN\_RX signal is ignored.

Figure 3-7 shows the connection of the CAN\_TX and CAN\_RX signals to the CAN controller when Silent mode and Loop back mode are combined.

**Figure 3-7 CAN Controller in Combined Silent and Loop Back Modes**



#### (5) Basic Mode

The CAN controller can be set in Basic mode by programming the Basic bit in the CAN Test Register to "1".

In Basic mode the CAN controller runs without using the message RAM.

The IF1 Message Interface Register is used to control transmission.

First when sending a message, the contents of transmission are configured in the IF1 Message Register. Then the BUSY bit in the IF1 Command Request Register is set to "1" to request transmission. While the BUSY bit is set to "1", the IF1 Message Interface Register is locked or the transmission is pending.

When the BUSY bit is set to "1", the CAN controller performs the following operation:

Immediately when the CAN bus becomes idle, the CAN controller loads the contents of the IF1 Message Interface Register to the send shift register to start transmission. When the transmission has finished successfully, the BUSY bit is reset to "0", and the locked IF1 Message Interface Register is released.

While pending, the transmission can be aborted by resetting the BUSY bit in the IF1 Command Request Register to "0". If the BUSY bit is reset to "0" during the transmission, a possible retransmission in case of lost arbitration or error detection is disabled.

The IF2 Message Interface Register is used to control reception.

When the BUSY bit is set to "1", the CAN controller performs the following operation:

- Stores the received message (the contents of the receive shift register) into the IF2 Message Interface Register without any acceptance filtering.

If a new message is stored into the IF2 Message Interface Register, the CAN controller sets the NewDat bit to "1". When an additional message is received while the NewDat bit is "1", then CAN controller sets MsgLst to "1".

**Notes:**

- *In Basic mode, all the control and status bits related to message objects are ignored as well as the control mode setting of the IFx Command Mask Register.*
- *The message number of the IFx Command Request Register is ignored.*
- *The NewDat bit and MsgLst bit in the IF2 Message Control Register retain their usual function, DLC indicates the received DLC, and other control bits are read as "0".*

**(6) Software Control of the Signal CAN\_TX**

CAN\_TX is a CAN send signal and has four output functions:

- Outputs serial data (Usual output)
- Outputs CAN sampling point signals to monitor the bit timing of the CAN controller
- Outputs a constant dominant value
- Outputs a constant recessive value

The output of constant dominant and recessive values, combined with CAN\_RX monitoring function of the CAN receive signal, can be used to check the CAN bus physical layer.

The output mode of the CAN\_TX signal can be controlled by the Tx1 and Tx0 bits in the CAN Test Register.

**Note:**

- *When using CAN message transmission or any of the Loop back, Silent, or Basic modes, the CAN\_TX signal must be set to the serial data output.*





### 3.8. Software Initialization

The following explains about initialization using software.

The sources of software initialization are as follows:

- Hardware reset
- Setting the Init bit in the CAN Control Register
- Shift to a busoff state

A hardware reset resets all other than the message objects (excluding the MsgVal, NewDat, IntPnd and TxRqst bits). Initialize the message objects by one of the following steps after hardware reset.

1. Initialize all message objects by the CPU.
2. Initialize only the used message objects by CPU. Clear MsgVal bit to "0" in the other message objects.

The CAN Bit Timing Register must be configured before clearing the Init bit in the CAN Control Register to "0".

The Init bit in the CAN Control Register is set to "1" in the following conditions:

- Writing "1" from the CPU
- Hardware reset
- In a busoff state

When the Init bit is set to "1", all message transfer from/to the CAN bus is stopped, and the CAN\_TX signal in the CAN bus output is in a recessive state (excludes software control of the CAN\_TX signal).

Setting the Init bit to "1" does not change the error counter and any register.

When the Init bit and CCE bit in the CAN Control Register are set to "1", the CAN Bit Timing Register for baud rate control and CAN Prescaler Extension Register can be configured.

The software initialization is completed by resetting the Init bit to "0".

By waiting for the occurrence of a consecutive 11 recessive bits (i.e., bus idle) after the Init bit is reset to "0", the message is transferred after synchronization with data transfer on the CAN bus.

Before changing message object masks ID, Xtd, EoB and RmtEn during normal operation, the MsgVal must be disabled.

### 3.9. CAN Wakeup Function

You can wake up with the receive operation of CAN by connecting CAN RX pin to an external interrupt pin.

#### (1) Pins Used for CAN Wakeup Function

Since RX0 and INT0 pins, RX1 and INT1 pins, or RX2 and INT2 pins are shared, the wakeup function is available for these pins.

The relation of CAN wakeup function and RX pin/INT pin is shown in Table 3-5.

Table 3-5 Relation of CAN Wakeup Function and RX Pin/INT Pin

	RX Pin	Interrupt Function
CAN0	RX0	INT0
CAN1	RX1	INT1
CAN2	RX2	INT2

#### (2) CAN Wakeup Function

Return from PSS by CAN reception data is enabled.

**Note:**

- When the wakeup function is used, the external interrupt setting before transition to PSS is required.



## 4. Registers

This section describes the registers of CAN controller.

All registers are prefixed by "CANxx\_". xx is the channel number (00, 01, 02).

An address space of 256 bytes is allocated the CAN register. The CPU gains access to the message RAM via the message interface registers.

## 4.1. Total Control Registers

Total control registers control the CAN protocol and operating modes, and provide status information.

**Table 4-1 List of Total Control Registers**

Abbreviated Register Name	Register Name	See
CTRLR	CAN Control Register	4.1.1
STATR	CAN Status Register	4.1.2
ERRCNT	CAN Error Counter	4.1.3
BTR	CAN Bit Timing Register	4.1.4
INTR	CAN Interrupt Register	4.1.5
TESTR	CAN Test Register	4.1.6
BRPER	CAN Prescaler Extension Register	4.1.7



#### 4.1.1. CAN Control Register (CTRLR)

The CAN Control Register controls the operating modes of the CAN controller.

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved							
ACCESS_TYPE	R0,W0							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Test	CCE	DAR	Reserved	EIE	SIE	IE	Init
ACCESS_TYPE	R/W	R/W	R/W	R0,W0	R/W	R/W	R/W	R,W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	1

**[bit15:8] Reserved: Reserved bits**

**[bit7] Test: Test mode enable bit**

Value	Description
0	Normal operation
1	Test mode

**Note:**

- The Test bit can be set to "1" only while the Init bit is "1".

**[bit6] CCE: Bit Timing Register write enable bit**

Value	Description
0	Disables write access to the CAN Bit Timing Register and CAN Prescaler Extension Register.
1	Enables write access to the CAN Bit Timing Register and CAN Prescaler Extension Register. This setting is valid while the Init bit is "1".

**[bit5] DAR: Automatic retransmission disable bit**

Value	Description
0	Enables automatic retransmission when arbitration is lost or an error is detected.
1	Disables automatic retransmission.

Based on the CAN specification (ISO11898. See 6.3.3 Recovery Sequence), the CAN controller automatically resends frames when arbitration is lost or an error is detected during transfer. To allow the automatic retransmission, set the DAR bit to "0". To operate CAN in Time Triggered CAN (TTCAN. See ISO11898-1) environments, set the DAR bit to "1".

**Notes:**

- In the mode where the DAR bit is set to "1", the TxRqst bit and the NewDat bit of a message object behave as follows. (For message objects, see Section "4.3. Message Object")
- When frame transmission has started, the TxRqst bit of the message object is reset to "0" while NewDat remains set.

- When frame transmission has finished successfully, the NewDat bit is reset to "0".
- If arbitration is lost or an error is detected during transmission, the NewDat bit remains set. To restart the transmission, the CPU must set the TxRqst to "1".
- If the DAR bit in the CAN Control Register (CTRLR) is changed from "0" to "1" during frame transmission (TxRqst = "1"), a frame being transmitted will be transmitted again. Therefore, change the DAR bit only while the Init bit is "1".
- A transmission using two or more message objects while the DAR bit is set to "1" assumes the following operations:
  - If the TxRqst in other message object is set to "1" before or during frame transmission (TxRqst bits in multiple message objects are set to "1"), all the set TxRqst bits are reset to "0" upon the start of frame transmission, and data in the message object with the highest priority will be sent.
- When frame transmission has finished successfully, the NewDat bit of the sent message object is reset to "0" and, if TxIE of the message object is "1" then, IntPnd of the message object is set to "1".
- Data in other message objects will not be sent because their TxRqst bits have been reset to "0" upon the start of frame transmission.  
Check which the message object is sent by NewDat and IntPnd, and then set TxRqst and NewDat to "1" again for another message object to be sent.

**[bit4] Reserved: Reserved bit**

**[bit3] EIE: Error interrupt code enable bit**

Value	Description
0	A change of the BOff or EWarn bit in the CAN Status Register disables the setting of interrupt code in the CAN Interrupt Register.
1	A change of the BOff or EWarn bit in the CAN Status Register enables the setting of status interrupt code in the CAN Interrupt Register.

**[bit2] SIE: Status interrupt code enable bit**

Value	Description
0	A change of the TxOk, RxOk, or LEC bit in the CAN Status Register disables the setting of interrupt code in the CAN Interrupt Register.
1	A change of the TxOk, RxOk, or LEC bit in the CAN Status Register enables the setting of status interrupt code in the CAN Interrupt Register. A change of TxOk, RxOk, or LEC bit caused by write access from the CPU is not set in the CAN Interrupt Register.

**[bit1] IE: Interrupt enable bit**

Value	Description
0	Disables interrupt generation.
1	Enables interrupt generation.

**[bit0] Init: Initialization bit**

Value	Description
0	CAN controller can operate.
1	Initialization

**Notes:**

- The busoff recovery sequence (see CAN Specification Rev. 2.0) cannot be shortened by setting or resetting the Init bit. If the device enters busoff state, the CAN controller itself sets the Init bit to "1", stopping all bus operations. If the Init bit is cleared to "0" from the busoff state, the bus operation remains stopped until 129 bus idle sequences (one bus idle sequence consists of 11 recessive bits) occur consecutively. When the bus recovery sequence has completed, the error counter is reset.
- If the Init bit is set to "1" and then reset to "0" during the busoff recovery sequence, the busoff recovery sequence restarts from the beginning (sends a set of 11 recessive bits 129 times).
- To write to the CAN Bit Timing Register, set the Init and CCE bits to "1".
- Setting the Init bit to "1" during transfer stops data reception immediately.
- The software should write "1" to the Init bit after the CAN transmission is completed.  
If the software wrote "1" to the Init bit during CAN transmission, should execute the following steps.
  1. Clear all transmission request bits (TxRqst) to "0" before it clear the Init bit.
  2. Clear the Init bit to "0".
  3. When CAN transmission is executed, set the transmission request bits (TxRqst) to "1" after more than two bits time from Step2.
- Before making transition to low consumption mode (stop mode or clock mode), and before changing clock supply, the Init bit must be set to "1" to initialize the CAN controller.
- To change the division ratio of clock supplied to the CAN interface by using the following registers, set the Init bit to "1" to stop the CAN controller previously.
  - CAN bit timing register (BTR)
  - CAN prescaler extension register (BRPER)
  - CAN prescaler control register (CANP\_CANPRE) of CAN prescaler.

### 4.1.2. CAN Status Register (STATR)

The CAN Status Register indicates the CAN status and a CAN bus state.

BITS_OFFSET	15	14	13	12	11	10	9	8
BITS_NAME	Reserved							
ACCESS_TYPE	R0,W0							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	BOff	EWarn	EPass	RxOk	TxOk	LEC		
ACCESS_TYPE	R,WX	R,WX	R,WX	R,W	R,W	R,W		
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	000		

[bit15:8] Reserved: Reserved bits

[bit7] BOff: Busoff bit

Value	Description
0	CAN bus is not in busoff state.
1	CAN bus is in busoff state.

[bit6] EWarn: Warning bit

Value	Description
0	Both the Send error counter and Receive error counter are below 96.
1	The Send error counter or Receive error counter has reached or exceeded 96.

[bit5] EPass: Error passive bit

Value	Description
0	Both the Send error counter and Receive error counter are below 128 (error active state).
1	The RP bit of the CAN Error Counter is "1", or the Send error counter is between 128 and 255 (error passive state).

[bit4] RxOk: Successful message reception bit

Value	Description
0	No message has been transferred successfully on the CAN bus, or the bus is in idle state.
1	A messages has been transferred successfully on the CAN bus.

[bit3] TxOk: Successful message transmission bit

Value	Description
0	The bus is in idle state, or no message has been sent successfully.
1	A message has been sent successfully.

**Note:**

- The RxOk and TxOk bits can be reset only by the CPU.



**[bit2:0] LEC[2:0]: Last error code bits**

Value	Description	
	State	Function
0	Normal	Successful transmission or reception.
1	Stuff error	Six or more dominant or recessive bits have been detected consecutively in a message.
2	Form error	A wrong fixed format part of a received frame has been detected.
3	Ack error	A sent message was not acknowledged by another node.
4	Bit 1 error	In the sent message data excluding the arbitration field, bits that have been sent as recessive data is detected as dominant data.
5	Bit 0 error	In the sent message data excluding the arbitration field, bits that have been sent as dominant data is detected as recessive data. This bit is set each time 11 recessive bits are detected during bus recovery. The bus recovery sequence can be monitored by reading this bit.
6	CRC error	The CRC data in a received message did not match with the calculated CRC value.
7	Undetected	If the CPU wrote "7" to the LEC bit, and the LEC value is read as "7" afterward, it indicates that no bus event has been detected since the CPU wrote the value. (The bus is in idle state)

The LEC bit holds a code that indicates the last error occurred on the CAN bus. When a message has been transferred (sent or received) without error, this bit is cleared to "0". The undetected code "7" is written by the CPU to check for code updates.

**Notes:**

- If the BOff and EWarn bits change while the EIE bit is "1", or if the RxOk, TxOk, and LEC bits change while the SIE bit is "1", the status interrupt code (0x8000) is written to the CAN Interrupt Register.
- Writing from the CPU updates the RxOk and TxOk bits, and this erases the RxOk and TxOk bits set by the CAN controller. If the RxOk and TxOk bits are used, clear the RxOk and TxOk bits within the time (45 x BT) after they are set to "1". BT indicates one bit time.
- If a change of the LEC bit causes an interrupt while the SIE bit is "1", do not write to the CAN Status Register.
- No interrupt is caused by a change of the EPass bit, or writing to the RxOk, TxOk, and LEC bits from the CPU.
- When the BOff bit has turned to "1", the EPass bit and EWarn bit are "1". When the EPass bit has turned to "1", the EWarn bit is "1".
- The status interrupt (0x8000) of the CAN Interrupt Register is cleared by reading this register.

### 4.1.3. CAN Error Counter (ERRCNT)

The CAN Error Counter indicates the receive error passive, the receive error counter, and the send error counter.

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	RP	REC						
ACCESS_TYPE	R,WX	R,WX						
PROT_TYPE	-							
INITIAL_VALUE	0	0000000						

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	TEC							
ACCESS_TYPE	R,WX							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

#### [bit15] RP: Receive error passive indication

Value	Description
0	The receive error counter is below the error passive level.
1	The receive error counter has reached the error passive level defined in the CAN specification.

#### [bit14:8] REC[6:0]: Receive error counter

A receive error counter value. The range of the receive error counter value is between 0 to 127.

If the receive error counter reaches or exceeds 128, the RP bit is set to "1", and the counter is not refreshed.

Example: If a receive error adds 8 to REC[6:0] = 127 with RP = 0,  
then REC[6:0] = 127 with RP = 1.  
If a receive error adds 8 to REC[6:0] = 126 with RP = 0,  
then REC[6:0] = 126 with RP = 1.  
If a receive error adds 8 to REC[6:0] = 119 with RP = 0,  
then REC[6:0] = 127 with RP = 0.  
If reception is successful when REC[6:0] = 126 and RP = 1,  
then REC[6:0] = 125 and RP = 0.

#### [bit7:0] TEC[7:0]: Send error counter

A send error counter value. The range of the send error counter value is between 0 and 255.

If the send error counter reaches or exceeds 256, the Init bit of the CAN Control Register is set to "1", and the counter is not refreshed.

Example: If a send error adds 8 to TEC[7:0] = 255 with Init = 0,  
then TEC[7:0] = 255 with Init = 1.  
If a send error adds 8 to TEC[7:0] = 254 with Init = 0,  
then TEC[7:0] = 254 with Init = 1.  
If a send error adds 8 to TEC[7:0] = 247 with Init = 0,  
then TEC[7:0] = 255 with Init = 0.



#### 4.1.4. CAN Bit Timing Register (BTR)

The CAN Bit Timing Register configures the prescaler and the bit timing.

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved	TSeg2			TSeg1			
ACCESS_TYPE	R0,W0	R/W			R/W			
PROT_TYPE	-							
INITIAL_VALUE	0	010			0011			

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	SJW		BRP					
ACCESS_TYPE	R/W		R/W					
PROT_TYPE	-							
INITIAL_VALUE	00		000001					

**[bit15] Reserved: Reserved bit**

##### **[bit14:12] TSeg2[2:0]: Time segment 2 setting bits**

Valid programmed values are 0 to 7. The TSeg2 + 1 value is the time segment 2.

The time segment 2 is equivalent to the Phase Buffer Segment 2 (Phase\_Seg2) in the CAN specification.

##### **[bit11:8] TSeg1[3:0]: Time segment 1 setting bits**

Valid programmed values are 1 to 15. The 0 value must not be used. The TSeg1 + 1 value is the time segment 1.

The time segment 1 is equivalent to the Propagation Segment (Prop\_Seg) + Phase Buffer Segment 1 (Phase\_Seg1) in the CAN specification.

##### **[bit7:6] SJW[1:0]: Resynchronization jump width setting bits**

Valid programmed values are 0 to 3. The SJW + 1 value is the resynchronization jump width.

##### **[bit5:0] BRP[5:0]: Baud rate prescaler setting bits**

Valid programmed values are 0 to 63. The BRP + 1 value is the baud rate prescaler.

It determines the basic unit of time quantum (tq) for the CAN controller by dividing the system clock (fsys).

**Note:**

- The CAN Bit Timing Register and CAN Prescaler Extension Register must be configured while the Init bit and CCE bit in the CAN Control Register are set to "1".

#### 4.1.5. CAN Interrupt Register (INTR)

The CAN Interrupt Register indicates message interrupt code and status interrupt code.

BIT_OFFSET	15-0
BIT_NAME	IntId
ACCESS_TYPE	R,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000

##### [bit15:0] IntId[15:0]: Interrupt code

Value	Description
0x0000	No interrupt
0x0001 to 0x0040	An interrupt cause indicates a message object number. (Message interrupt code)
0x0041 to 0x7FFF	Unused.
0x8000	Indicates an interrupt by a change in the CAN Status Register. (Status interrupt code)
0x8001 to 0xFFFF	Unused.

If two or more interrupts are pending, the CAN Interrupt Register indicates a high-priority interrupt code. If a high-priority interrupt code is generated while an interrupt code is set to the CAN Interrupt Register, the CAN Interrupt Register is updated to the high-priority interrupt code.

High-priority interrupt codes are arranged in the order of status interrupt code (0x8000), message interrupt codes (0x0001, 0x0002, 0x0003, ....., 0x0040).

When the IE bit of the CAN Control Register is set to "1" while the IntId bit is not 0x0000, a CPU interrupt signal becomes active. When the IntId bit is set to 0x0000 (an interrupt cause is reset) or the IE bit of the CAN Control Register is reset to "0", an interrupt signal becomes inactive.

To clear a message interrupt code, reset the IntPnd bit of the target message object (see Section "4.3. Message Object" for the message object) to "0".

A status interrupt code is cleared by reading the CAN Status Register.

**Note:**

- To read the CAN Interrupt Register, access it in half-word or word mode.



#### 4.1.6. CAN Test Register (TESTR)

The CAN test register is used to monitor the setting of test mode and RX pin. For operations, see Section "3.7. Test Mode".

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved							
ACCESS_TYPE	R0,W0							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Rx	Tx		LBack	Silent	Basic	Reserved	
ACCESS_TYPE	R,WX	R/W		R/W	R/W	R/W	R0,W0	
PROT_TYPE	-							
INITIAL_VALUE	X	00		0	0	0	00	

[bit15:8] Reserved: Reserved bits

[bit7] Rx: RX pin monitor bit

Value	Description
0	Indicates that the CAN bus is in the dominant state.
1	Indicates that the CAN bus is in the recessive state.

[bit6:5] Tx[1:0]: TX pin control bits

Value	Description
00	Normal operation.
01	Outputs a sampling point to the TX pin.
10	Outputs a dominant to the TX pin.
11	Outputs a recessive to the TX pin.

[bit4] LBack: Loop back mode

Value	Description
0	Disables loop back mode.
1	Enables loop back mode.

[bit3] Silent: Silent mode

Value	Description
0	Disables silent mode.
1	Enables silent mode.

[bit2] Basic: Basic mode

Value	Description
0	Disables basic mode.
1	Enables basic mode. The IF1 register is used for a sent message, and the IF2 register for a received message.

**[bit1:0] Reserved: Reserved bits**

**Notes:**

- *After setting "1" to the Test bit of the CAN Control Register, write data to this register. When the Test bit of the CAN Control Register is set to "1", test mode becomes valid. If the Test bit of the CAN Control Register is set to "0" during processing, test mode changes to normal mode.*
- *If the Tx bits is set to a value other than "0b00", no message can be sent.*



#### 4.1.7. CAN Prescaler Extended Register (BRPER)

The CAN prescaler extension register is used to extend the prescaler used in the CAN controller by combining it with the prescaler specified at a CAN bit timing.

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved							
ACCESS_TYPE	R0,W0							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved				BRPE			
ACCESS_TYPE	R0,W0				R/W			
PROT_TYPE	-							
INITIAL_VALUE	0000				0000			

**[bit15:4] Reserved: Reserved bits**

##### **[bit3:0] BRPE[3:0]: Baud rate prescaler extension bit**

This bit is used to extend the baud rate prescaler up to 1023 by combining BRP and BRPE in the CAN Bit Timing Register.

The value "{BRPE (MSB: 4 bits), BRP (LSB: 6 bits)} + 1" is set as the prescaler value of the CAN controller.

## 4.2. Message Interface Registers

The CAN controller provides two message interface registers to control an access from the CPU to the message RAM.

The CAN controller provides two message interface registers to control an access from the CPU to the message RAM. These two registers are used to avoid a conflict between an access from the CPU to the message RAM and an access from the CAN controller to the message RAM by buffering the data (message object) transferred or to be transferred. A message object (see Section "4.3. Message Object" for message object) is used to collectively transfer data between the message interface registers and message RAM.

Two message interface registers have the same functions, excluding Basic mode, and can be operated independently. For example, the IF1 Message Interface Register can be used to write data to the message RAM while the IF2 Message Interface Register is being used to read data from the message RAM. Table 4-2 shows two message interface registers.

Each Message Interface Register consists of two components: (1) Command Register (IFx Command Request Register and IFx Command Mask Register) and (2) Message Buffer Register (IFx Mask Register, IFx Arbitration Registers, IFx Message Control Register, and IFx Data Registers) controlled with the Command Register. The IFx Command Mask Register indicates the data transfer direction and also which part in a message object is to be transferred. The IFx Command Request Register is used to select a message number and perform the operation specified in the IFx Command Mask Register.

**Table 4-2 List of Message Interface Registers**

Abbreviated Register Name	Register Name	See
IF1CREQ	IF1 Command Request Register	4.2.1
IF1CMSK	IF1 Command Mask Register	4.2.2
IF1MSK1	IF1 Mask Register 1	4.2.3
IF1MSK2	IF1 Mask Register 2	4.2.3
IF1ARB1	IF1 Arbitration Register 1	4.2.4
IF1ARB2	IF1 Arbitration Register 2	4.2.4
IF1MCTR	IF1 Message Control Register	4.2.5
IF1DTA1	IF1 Data A Register 1	4.2.6
IF1DTA2	IF1 Data A Register 2	4.2.6
IF1DTB1	IF1 Data B Register 1	4.2.6
IF1DTB2	IF1 Data B Register 2	4.2.6
IF2CREQ	IF2 Command Request Register	4.2.1
IF2CMSK	IF2 Command Mask Register	4.2.2
IF2MSK1	IF2 Mask Register 1	4.2.3
IF2MSK2	IF2 Mask Register 2	4.2.3
IF2ARB1	IF2 Arbitration Register 1	4.2.4
IF2ARB2	IF2 Arbitration Register 2	4.2.4
IF2MCTR	IF2 Message Control Register	4.2.5
IF2DTA1	IF2 Data A Register 1	4.2.6
IF2DTA2	IF2 Data A Register 2	4.2.6





Abbreviated Register Name	Register Name	See
IF2DTB1	IF2 Data B Register 1	4.2.6
IF2DTB2	IF2 Data B Register 2	4.2.6

#### 4.2.1. IFx Command Request Register (IFxCREQ)

The IFx Command Request Register is used to select a message number of the message RAM and transfer data between the message RAM and Message Buffer Register. In basic mode, IF1 is used to control sending and IF2 to control receiving.

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	BUSY	Reserved						
ACCESS_TYPE	R/W	R0,W0						
PROT_TYPE	-							
INITIAL_VALUE	0	0000000						

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Message Number							
ACCESS_TYPE	R/W							
PROT_TYPE	-							
INITIAL_VALUE	00000001							

A message transfer starts between the message RAM and Message Buffer Register (IFx Mask Register, IFx Arbitration Registers, IFx Message Control Register, and IFx Data Registers) immediately after a message number has been written to the IFx Command Request Register. This write operation sets the BUSY bit to "1" and continues transfer processing while the BUSY bit is "1". When transfer processing is ended, the BUSY bit is reset to "0".

If the CPU accesses the Message Interface Register while the BUSY bit is "1", the CPU waits until the BUSY bit is set to "0" (for 3 to 6 clock cycles after data has been written to the IFx Command Request Register).

The method for using the BUSY bit is different in Basic mode. The IF1 Command Request Register, which is used as a send message, starts message sending when the BUSY bit is set to "1". When message transfer has finished successfully, the BUSY bit is reset to "0". Resetting the BUSY bit to "0" enables canceling message transfer at any time.

In Basic mode, the IF2 Command Request Register, which is used for receiving message, stores the received message in the IF2 Message Interface Register when the BUSY bit is set to "1".

##### [bit15] BUSY: Busy flag bit

- Other than basic mode

Value	Description
0	Indicates that data transfer is not performed between the Message Interface Register and message RAM. [Initial value]
1	Indicates that data transfer is being performed between the Message Interface Register and message RAM.

- Basic mode



IF1 command request register

Value	Description
0	Disables message sending.
1	Enables message sending.

IF2 command request register

Value	Description
0	Disables message receiving.
1	Enables message receiving.

[bit14:8] Reserved: Reserved bits

[bit7:0] Message Number[7:0]: Message number (64 message objects)

Value	Description
0x00, 0x80, 0xC0	Setting disabled. If specified, it is interpreted as 0x40, causing 0x40 to be read.
0x01 - 0x40	Specifies a message number to perform processing.
0x41 - 0x7F, 0x81 - 0xBF, 0xC1 - 0xFF	Setting disabled. If specified, it is interpreted as one of 0x01 to 0x3F, causing the interpreted value to be read.

**Note:**

- The BUSY bit can be read and written. Therefore, writing any data to this bit does not affect operations, excluding in Basic mode (see Section "3.7. Test Mode" for basic mode).

### 4.2.2. IFx Command Mask Register (IFxCMSK)

The IFx Command Mask Register is used to control the transfer direction between the Message Interface Register and message RAM and specify which data is to be updated. This register is invalid in basic mode.

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved							
ACCESS_TYPE	R0,W0							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	WR/RD	Mask	Arb	Control	CIP	TxRqst/ NewDat	Data A	Data B
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit15:8] Reserved: Reserved bits

[bit7] WR/RD: Writing or reading control bit

Value	Description
0	Indicates that data is read from the message RAM. Reading from the message RAM is performed by writing data to the IFx Command Request Register. What data is to be read from the message RAM depends on the setting of the Mask, Arb, Control, CIP, TxRqst/NewDat, Data A, or Data B bit. [Initial value]
1	Indicates that data is written to the message RAM. Writing to the message RAM is performed by writing data to the IFx Command Request Register. What data is to be written to the message RAM depends on the setting of the Mask, Arb, Control, CIP, TxRqst/NewDat, Data A, or Data B bit.

**Note:**

- After resetting, data of the message RAM is undefined. The message RAM cannot be read while its data is undefined.



[bit6:0] in the IFx Command Mask Register are set to different values depending on the transfer direction specified with the WR or RD bit.

a) When the Transfer Direction is "Writing" (WR/RD="1")

**[bit6] Mask: Mask data update bit**

Value	Description
0	Indicates that mask data (ID mask + MDir + MXtd) of a message object <sup>*1</sup> is not updated. [Initial value]
1	Indicates that mask data (ID mask + MDir + MXtd) of a message object <sup>*1</sup> is updated.

\*1: See Section "4.3. Message Object".

**[bit5] Arb: Arbitration data update bit**

Value	Description
0	Indicates that arbitration data (ID + Dir + Xtd + MsgVal) of a message object <sup>*1</sup> is not updated.
1	Indicates that arbitration data (ID + Dir + Xtd + MsgVal) of a message object <sup>*1</sup> is updated.

\*1: See Section "4.3. Message Object".

**[bit4] Control: Control data update bit**

Value	Description
0	Indicates that control data (IFx Message Control Register) of a message object <sup>*1</sup> is not updated. [Initial value]
1	Indicates that control data (IFx Message Control Register) of a message object <sup>*1</sup> is updated.

\*1: See Section "4.3. Message Object".

**[bit3] CIP: Interrupt clear bit**

If this bit is set to "0" or "1", it does not affect CAN controller operations.

**[bit2] TxRqst/NewDat: Message transmission request bit**

Value	Description
0	Indicates that the TxRqst bits of the message object <sup>*1</sup> and CAN Transmit Request Register are not changed.
1	Indicates that the TxRqst bits of the message object <sup>*1</sup> and CAN Transmit Request Register are set to "1" (transmission requested).

\*1: See Section "4.3. Message Object".

**[bit1] Data A: Data0-Data3 update bit**

Value	Description
0	Indicates that Data0 - Data3 of a message object <sup>*1</sup> is not updated.
1	Indicates that Data0 - Data3 of a message object <sup>*1</sup> is updated.

\*1: See Section "4.3. Message Object".

**[bit0] Data B: Data4-Data7 update bit**

Value	Description
0	Indicates that Data4 - Data7 of a message object <sup>*1</sup> is not updated.
1	Indicates that Data4 - Data7 of a message object <sup>*1</sup> is updated.

\*1: See Section "4.3. Message Object".

**Notes:**

- When the TxRqst or NewDat bit of the IFx Command Mask Register is set to "1", the setting of the TxRqst bit in the IFx Message Control Register becomes invalid.
- This register is invalid in Basic mode.

**b) When the Transfer Direction is "Writing" (WR/RD="0")**

**[bit6] Mask: Mask data update bit**

Value	Description
0	Indicates that data (ID mask + MDir + MXtd) is not transferred from a message object <sup>*1</sup> to IFx Master Register 1 or 2.
1	Indicates that data (ID mask + MDir + MXtd) is transferred from a message object <sup>*1</sup> to IFx Master Register 1 or 2.

\*1: See Section "4.3. Message Object".

**[bit5] Arb: Arbitration data update bit**

Value	Description
0	Indicates that data (ID + Dir + Xtd + MsgVal) is not transferred from a message object <sup>*1</sup> to IFx Arbitration Register 1 or 2.
1	Indicates that data (ID + Dir + Xtd + MsgVal) is transferred from a message object <sup>*1</sup> to IFx Arbitration Register 1 or 2.

\*1: See Section "4.3. Message Object".

**[bit4] Control: Control data update bit**

Value	Description
0	Indicates that control data is not transferred from a message object <sup>*1</sup> to the IFx Message Control Register.
1	Indicates that control data is transferred from a message object <sup>*1</sup> to the IFx Message Control Register.

\*1: See Section "4.3. Message Object".

**[bit3] CIP: Interrupt clear bit**

Value	Description
0	Indicates that the IntPnd bits of the message object <sup>*1</sup> and CAN Interrupt Pending Register are held.
1	Indicates that the IntPnd bits of the message object <sup>*1</sup> and CAN Interrupt Pending Register are cleared to "0".

\*1: See Section "4.3. Message Object".



**[bit2] TxRqst/NewDat: Message transmission request bit**

Value	Description
0	Indicates that the NewDat bits of the message object <sup>*1</sup> and CAN New Data Register are held.
1	Indicates that the NewDat bits of the message object <sup>*1</sup> and CAN New Data Register are cleared to "0".

\*1: See Section "4.3. Message Object".

**[bit1] Data A: Data0-Data3 update bit**

Value	Description
0	Indicates that data of the message object <sup>*1</sup> and IFx Data Register A1 or A2 is held.
1	Indicates that data of the message object <sup>*1</sup> and IFx Data Register A1 or A2 is updated.

\*1: See Section "4.3. Message Object".

**[bit0] Data B: Data4-Data7 update bit**

Value	Description
0	Indicates that data of the message object <sup>*1</sup> and IFx Data Register B1 or B2 is held.
1	Indicates that data of the message object <sup>*1</sup> and IFx Data Register B1 or B2 is updated.

\*1: See Section "4.3. Message Object".

**Notes:**

- The *IntPnd* and *NewDat* bits can be reset to "0" by reading a message object. However, the value before reset by reading is set to the *IntPnd* and *NewDat* bits of the IFx Message Control Register.
- This register is invalid in Basic mode.

### 4.2.3. IFx Mask Registers 1/2 (IFxMSK1, IFxMSK2)

The IFx Mask Registers 1 and 2 are used to write or read message object mask data of the message RAM. The specified mask data is invalid in basic mode.

- IFx mask register 2

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	MXtd	MDir	Reserved	Msk[28:24]				
ACCESS_TYPE	R/W	R/W	R1,W1	R/W				
PROT_TYPE	-							
INITIAL_VALUE	1	1	1	11111				

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Msk[23:16]							
ACCESS_TYPE	R/W							
PROT_TYPE	-							
INITIAL_VALUE	11111111							

- IFx mask register 1

BIT_OFFSET	15-0																												
BIT_NAME	Msk[15:0]																												
ACCESS_TYPE	R/W																												
PROT_TYPE	-																												
INITIAL_VALUE	11111111_11111111																												

For the explanation of each bit in this register, see Section "4.3. Message Object".





#### 4.2.4. IFx Arbitration Registers 1/2 (IFxARB1, IFxARB2)

The IFx Arbitration Registers 1 and 2 are used to write or read arbitration data of message object in the message RAM. This register is invalid in Basic mode.

- IFx arbitration register 2

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	MsgVal	Xtd	Dir	ID[28:24]				
ACCESS_TYPE	R/W	R/W	R/W	R/W				
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	00000				

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	ID[23:16]							
ACCESS_TYPE	R/W							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

- IFx arbitration register 1

BIT_OFFSET	15-0
BIT_NAME	ID[15:0]
ACCESS_TYPE	R/W
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000

For the explanation of each bit in this register, see Section "4.3. Message Object".

**Note:**

- If the MsgVal bit of a message object is cleared to "0" during transmission, the TxOk bit of the CAN Status Register is set to "1" when transmission has been completed. However, the TxRqst bits of the message object and CAN Transmit Request Register are not cleared to "0". Use the Message Interface Register to clear the TxRqst bit to "0".

#### 4.2.5. IFx Message Control Register (IFxMCTR)

The IFx Message Control Register is used to write or read message object control data of the message RAM. This register is invalid in Basic mode. The NewDat and MsgLst bits of the IF2 Message Control Register are used to perform normal operations. The DLC bits indicate the DLC of the received message. The other control bits are invalid ("0").

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	NewDat	MsgLst	IntPnd	UMask	TxE	RxE	RmtEn	TxRqst
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	EoB	Reserved			DLC			
ACCESS_TYPE	R/W	R0,W0			R/W			
PROT_TYPE	-							
INITIAL_VALUE	0	000			0000			

For the explanation of each bit in this register, see Section "4.3. Message Object".

**Notes:**

- The values of the TxRqst, NewDat, and IntPnd bits are set as shown below depending on the setting of the WR or RD bit in the IFx Command Mask Register.
- When the transfer direction is "writing" (IFx Command Mask Register: WR/RD="1")
  - The TxRqst bit of this register is valid only when the TxRqst or NewDat bit of the IFx Command Mask Register is set to "0".
- When the transfer direction is "reading" (IFx Command Mask Register: WR/RD="0")
  - If the IntPnd bits of the message object and CAN Interrupt Pending Register are reset by setting the CIP bit of the IFx Command Mask Register to "1" and writing data to the IFx Command Request Register, the value of the IntPnd bit that is specified before reset is stored in this register.
  - If the NewDat bits of the message object and CAN New Data Register are reset by setting the TxRqst or NewDat bit of the IFx Command Mask Register to "1" and writing data to the IFx Command Request Register, the value of the NewDat bit that is specified before reset is stored in this register.



#### 4.2.6. IFx Data A/B Registers 1/2 (IFxDTA1, IFxDTA2, IFxDTB1, IFxDTB2)

The IFx Data Registers A1, A2, B1, and B2 are used to write or read message object sending or receiving data to or from the message RAM. Those registers are used only to send or receive a data frame, and not to send or receive a remote frame.

BIT_OFFSET	15-0
BIT_NAME	Data
ACCESS_TYPE	R/W
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000

#### [bit15:0] Data: Message object sending or receiving data

Data: Data0 to Data7

##### (1) Relation between Data and Address by Endian

		addr+3	addr+2	addr+1	addr+0
IFx data register	IFx data A register 1 (little endian)			Data1	Data0
	IFx data A register 2 (little endian)	Data3	Data2		
	IFx data B register 1 (little endian)			Data5	Data4
	IFx data B register 2 (little endian)	Data7	Data6		
IFx data mirror	IFx data A register 1 (big endian)			Data2	Data3
	IFx data A register 2 (big endian)	Data0	Data1		
	IFx data B register 1 (big endian)			Data6	Data7
	IFx data B register 2 (big endian)	Data4	Data5		

##### (2) Send Message Data Setting

The set data is sent in the order of Data0, Data1, ..., Data7, beginning with the MSB (bit 7 or bit 15).

##### (3) Received Message Data

The received message data is stored in the order of Data0, Data1, ..., Data7, beginning with the MSB (bit 7 or bit 15).

##### Notes:

- If the received message data is lower than eight bytes in length, undefined data is written to the remaining bytes of the Data Register.
- To transfer data to a message object, it is processed every four bytes in the Data A or Data B Register; therefore, it is impossible to update only a part of 4-byte data.

### 4.3. Message Object

The message RAM provides 64 message objects. To avoid a conflict when simultaneously accessing the message RAM from the CPU and the CAN controller, the CPU cannot directly access message objects. The message RAM is accessed via the Message Interface Register.

This section explains the configuration and functions of a message object.

#### (1) Configuration of Message Object

UMask	Msk	MXtd	MDir	EoB	NewDat		MsgLst	RxIE	TxIE	IntPnd	RmtEn	TxRqst
MsgVal	ID	Xtd	Dir	DLC	Data0	Data1	Data2	Data3	Data4	Data5	Data6	Data7

**Note:**

- A message object is not initialized using the *Init* bit of the CAN Control Register or the hardware reset function. For details, see Section "3.1. Message Objects".

#### (2) Functions of Message Object

The ID, Xtd, and Dir bits are used to indicate the ID and message type when sending a message. They are used in the acceptance filter together with the Msk28-Msk0, MXtd, and MDir bits when receiving a message.

ID, IDE, RTR, DLC, and DATA in a data or remote frame that passed through the acceptance filter are respectively stored in ID, Xtd, Dir, DLC, and Data7 to Data0 of a message object. Xtd indicates whether the received frame is an extension or standard frame. If Xtd is "1", a 29-bit ID (extension frame) is received. If Xtd is "0", a 11-bit ID (standard frame) is received.

When the received data or remote frame matches one or more message objects, it is stored in the message object with the lowest message number. For details, see Acceptance Filter for Received Messages in "3.1. Message Objects".

#### MsgVal: Valid message bit

Value	Description
0	Message objects are invalid. Disables message sending/receiving.
1	Message objects are valid. Enables message sending/receiving.

**Notes:**

- Reset the *MsgVal* bit of an unused message object to "0" before clearing the *Init* bit of the CAN Control Register to "0".
- Be sure to reset the *MsgVal* bit of a message object to "0" before changing the value of ID, Xtd, Dir, or DLC.
- If the *MsgVal* bit of a message object is cleared to "0" during transmission, the *TxOk* bit of the CAN Status Register is set to "1" when transmission has been completed. However, the *TxRqst* bits of the message object and CAN Transmit Request Register are not cleared to "0". Use the Message Interface Register to clear the *TxRqst* bit to "0".

**UMask: Acceptance mask enable bit**

Value	Description
0	Does not use Msk, MXtd, or MDir.
1	Uses Msk, MXtd, or MDir.

**Notes:**

- Change the value of the UMask bit when the Init bit of the CAN Control Register is "1" or the MsgVal bit is "0".
- When the Dir bit is "1" and the RmtEn bit is "0", operations vary depending on the setting of the UMask bit.
  - If the UMask bit is "1", reset the TxRqst bit to "0" when a remote frame has been received through the acceptance filter. The received ID, IDE, RTR, and DLC are stored in a message object, and the NewDat bit is set to "1" while data remains unchanged (data is handled as a data frame).
  - If the UMask bit is "0", the TxRqst bit is held and a remote frame is ignored even if it has been received.

**ID: Message ID**

	Description
ID[28:0]	Specifies a 29-bit ID (extension frame).
ID[28:18]	Specifies a 11-bit ID (standard frame).

**Msk: ID mask**

Value	Description
0	Masks the bit that corresponds to the ID of a message object.
1	Does not mask the bit that corresponds to the ID of a message object.

**Xtd: Extension ID enable bit**

Value	Description
0	Uses the 11-bit ID (standard frame) for message object.
1	Uses the 29-bit ID (extension frame) for message object.

**MXtd: Extension ID mask bit**

Value	Description
0	Does not compare the set value of the Xtd bit in a message object with that of the IDE bit of a received frame. Determine whether to perform the comparison as the ID of a standard frame or extension frame based on the IDE bit of a received frame.
1	Compares the set value of the Xtd bit in a message object with that of the IDE bit of a received frame.

**Note:**

- When a 11-bit ID (standard frame) is set to a message object, the ID of a received data frame is written to ID[28:18]. Msk[28:18] are used to mask the ID.

**Dir: Message direction bit**

Value	Description
0	Indicates the receiving direction. When the TxRqst bit is set to "1", a remote frame is sent. When the TxRqst bit is set to "0", a data frame that passed through the acceptance filter is received.
1	Indicates the transmission direction. When the TxRqst bit is set to "1", a data frame is sent. When the TxRqst is "0" and the RmtEn bit is "1", the CAN controller sets the TxRqst bit to "1" if a data frame that passed through the acceptance filter is received.

**MDir: Message direction mask bit**

Value	Description
0	Masks the message direction bit (Dir) through the acceptance filter.
1	Does not mask the message direction bit (Dir) through the acceptance filter.

**Note:**

- Always set the MDir bit to "1".

**EoB: End of buffer bit (For details, see Section "3.4. FIFO Buffer Function".)**

Value	Description
0	Indicates that a message object is used as a FIFO buffer, not the last message.
1	Indicates a single message object or the last message object in the FIFO buffer.

**Note:**

- The EoB bit is used to configure a FIFO buffer for message objects 2 to 64.
- When processing a single message object without using a FIFO buffer, be sure to set the EoB bit to "1".

**NewDat: Data update bit**

Value	Description
0	Indicates that no valid data resides.
1	Indicates that valid data resides.

**MsgLst: Message lost**

Value	Description
0	Message lost does not occur.
1	Message lost occurs.

**Note:**

- The MsgLst bit is valid only when the Dir bit is "0" (receiving direction).



**RxIE: Receiving interrupt flag enable bit**

Value	Description
0	Does not change the value of the IntPnd bit after frame receiving has succeeded.
1	Changes the IntPnd bit to "1" after frame receiving has succeeded.

**TxIE: Transmission interrupt flag enable bit**

Value	Description
0	Does not change the value of the IntPnd bit after frame transmission has succeeded.
1	Changes the IntPnd bit to "1" after frame transmission has succeeded.

**IntPnd: Interrupt pending bit**

Value	Description
0	No interrupt cause is detected.
1	An interrupt cause is detected. If other high-priority interrupt is not found, the IntId bit of the CAN Interrupt Register indicates this message object.

**RmtEn: Remote enable**

Value	Description
0	Does not change the value of the TxRqst bit when a remote frame has been received.
1	Sets the TxRqst bit to "1" when a remote frame is received while the Dir bit is "1".

**Notes:**

- When the Dir bit is "1" and the RmtEn bit is "0", operations vary depending on the setting of the UMask bit.
- If the UMask bit is "1", reset the TxRqst bit to "0" when a remote frame has been received through the acceptance filter. The received ID, IDE, RTR, and DLC are stored in a message object. The NewDat bit is set to "1" while data remains unchanged (data is handled as a data frame).
- If the UMask bit is "0", the TxRqst bit is held and a remote frame is ignored even if it has been received.

**TxRqst: Transmission request bit**

Value	Description
0	Indicates the sending idle state (neither the sending state nor the sending wait state).
1	Indicates the sending or sending wait state.

**DLC: Data length code**

Value	Description
0 to 8	The data frame length is 0 to 8 bytes.
9 to 15	Setting disabled. 8-byte length if specified.

**Note:**

- The received DLC is stored in the DLC bit if a data frame is received.

**Data0 to Data7: Data0 to Data7**

	Description
Data0	First data byte in CAN data frame
Data1	2nd data byte in CAN data frame
Data2	3rd data byte in CAN data frame
Data3	4th data byte in CAN data frame
Data4	5th data byte in CAN data frame
Data5	6th data byte in CAN data frame
Data6	7th data byte in CAN data frame
Data7	8th data byte in CAN data frame

**Notes:**

- Serial data is output from the MSB (bit 7 or bit 15) to the CAN bus.
- If the received message data is lower than eight bytes in length, undefined data is written to the remaining bytes of the Data Register.
- To transfer data to a message object, it is processed every four bytes in the Data A or Data B Register; therefore, it is impossible to update only a part of 4-byte data.





## 4.4. Message Handler Registers

Message handler registers are all in read only mode. The TxRqst, NewDat, IntPnd, and MsgVal bits of a message object and the IntId bit indicate the status.

**Table 4-1 List of Message Handler Registers**

Abbreviated Register Name	Register Name	See
TREQ1	CAN Transmit Request Register 1	4.4.1
TREQ2	CAN Transmit Request Register 2	4.4.1
TREQ3	CAN Transmit Request Register 3	4.4.1
TREQ4	CAN Transmit Request Register 4	4.4.1
NEWDT1	CAN New Data Register 1	4.4.2
NEWDT2	CAN New Data Register 2	4.4.2
NEWDT3	CAN New Data Register 3	4.4.2
NEWDT4	CAN New Data Register 4	4.4.2
INTPND1	CAN Interrupt Pending Register 1	4.4.3
INTPND2	CAN Interrupt Pending Register 2	4.4.3
INTPND3	CAN Interrupt Pending Register 3	4.4.3
INTPND4	CAN Interrupt Pending Register 4	4.4.3
MSGVAL1	CAN Message Valid Register 1	4.4.4
MSGVAL2	CAN Message Valid Register 2	4.4.4
MSGVAL3	CAN Message Valid Register 3	4.4.4
MSGVAL4	CAN Message Valid Register 4	4.4.4

#### 4.4.1. CAN Transmit Request Registers 1/2/3/4 (TREQ1, TREQ2, TREQ3, TREQ4)

The CAN Transmit Request Register indicates the TxRqst bit of all message objects. This register checks which message object transmission request is pending by reading the TxRqst bit.

- CAN transmit request register 4 (TREQ4)

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	TxRqst64	TxRqst63	TxRqst62	TxRqst61	TxRqst60	TxRqst59	TxRqst58	TxRqst57
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	TxRqst56	TxRqst55	TxRqst54	TxRqst53	TxRqst52	TxRqst51	TxRqst50	TxRqst49
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

##### [bit15] to [bit0] TxRqst64 to TxRqst49: Transmission request bit

The number corresponds to the message object number.

Value	Description
0	Indicates the sending idle state (neither the sending state nor the sending wait state).
1	Indicates the sending or sending wait state.

- CAN transmit request register 3 (TREQ3)

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	TxRqst48	TxRqst47	TxRqst46	TxRqst45	TxRqst44	TxRqst43	TxRqst42	TxRqst41
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	TxRqst40	TxRqst39	TxRqst38	TxRqst37	TxRqst36	TxRqst35	TxRqst34	TxRqst33
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

##### [bit15] to [bit0] TxRqst48 to TxRqst33: Transmission request bit

The number corresponds to the message object number.

Value	Description
0	Indicates the sending idle state (neither the sending state nor the sending wait state).
1	Indicates the sending or sending wait state.



– CAN transmit request register 2 (TREQ2)

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	TxRqst32	TxRqst31	TxRqst30	TxRqst29	TxRqst28	TxRqst27	TxRqst26	TxRqst25
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	TxRqst24	TxRqst23	TxRqst22	TxRqst21	TxRqst20	TxRqst19	TxRqst18	TxRqst17
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

**[bit15] to [bit0] TxRqst32 to TxRqst17: Transmission request bit**

The number corresponds to the message object number.

Value	Description
0	Indicates the sending idle state (neither the sending state nor the sending wait state).
1	Indicates the sending or sending wait state.

– CAN transmit request register 1 (TREQ1)

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	TxRqst16	TxRqst15	TxRqst14	TxRqst13	TxRqst12	TxRqst11	TxRqst10	TxRqst9
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	TxRqst8	TxRqst7	TxRqst6	TxRqst5	TxRqst4	TxRqst3	TxRqst2	TxRqst1
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

**[bit15] to [bit0] TxRqst16 to TxRqst1: Transmission request bit**

The number corresponds to the message object number.

Value	Description
0	Indicates the sending idle state (neither the sending state nor the sending wait state).
1	Indicates the sending or sending wait state.

The following shows conditions to set or reset the TxRqst bit.

- Setting conditions
  - Set "1" to the WR/RD bit of the IFx Command Mask Register and "1" to the TxRqst bit, and write data to the IFx Command Request Register to set the TxRqst bit to "1" in a specific message object.
  - Set "1" to the WR/RD bit of the IFx Command Mask Register, "0" to the TxRqst bit, and "1" to the Control bit, and "1" to the TxRqst bit of the IFx Message Control Register. Then write data to the IFx Command Request Register to set the TxRqst bit to "1" in a specific message object.
  - If the Dir bit is "1" and the RmtEn bit is "1", the TxRqst bit is set to "1" by receiving a remote frame that passed through the acceptance filter.
- Resetting conditions
  - Set "1" to the WR/RD bit of the IFx Command Mask Register, "0" to the TxRqst bit, and "1" to the Control, and "0" to the TxRqst bit of the IFx Message Control Register. Then write data to the IFx Command Request Register to reset the TxRqst bit of a specific message object.
  - The TxRqst bit is reset when frame transmission has finished successfully.
  - If the Dir bit is "1", the RmtEn bit is "0", and the UMask bit is "1", the TxRqst bit is reset by receiving a remote frame that passed through the acceptance filter.

**Notes:**

- *In case of the following procedures, the messages may not be sent until any of the events described below occurs.*
- *Procedures :*
  - 1. A message object with the lowest priority is used for transmission.
  - 2. The TxRqst bit was previously set to "1", but is set to "0" to abort transmission.
  - 3. The TxRqst bit is set to "1" again after step 2.
- *Events :*
  - A valid message flows on the CAN bus.
  - A transmission request is issued to another message object.
  - CAN is initialized by the Init bit.
- *If canceling the transmission is required to suit system operations, execute the following steps.*
- 1. *Execute one of the following steps.*
  - Do not use a message object with the lowest priority as a send message object.
  - After aborting the transmission, generate any of the above events.
- 2. *Set the TxRqst bit to "1" again.*
- *If the ID, DLC, Xtd, and Data7 to Data0 in message objects are changed while the TxRqst bit is "1", message objects before and after the change are mixed for transmission, or the message objects after the change may not be transmitted. Therefore, be sure to change them while the TxRqst bit is "0".*



#### 4.4.2. CAN New Data Register 1/2/3/4 (NEWDT1, NEWDT2, NEWDT3, NEWDT4)

The CAN New Data Register indicates the NewDat bit of all message objects. This register checks which message object data is updated by reading the NewDat bit.

- CAN new data register 4 (NEWDT4)

BITS_OFFSET	15	14	13	12	11	10	9	8
BITS_NAME	NewDat64	NewDat63	NewDat62	NewDat61	NewDat60	NewDat59	NewDat58	NewDat57
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BITS_OFFSET	7	6	5	4	3	2	1	0
BITS_NAME	NewDat56	NewDat55	NewDat54	NewDat53	NewDat52	NewDat51	NewDat50	NewDat49
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

##### [bit15] to [bit0] NewDat64 to NewDat49: Data update bit

The number corresponds to the message object number.

Value	Description
0	Indicates that no valid data resides.
1	Indicates that valid data resides.

- CAN new data register 3 (NEWDT3)

BITS_OFFSET	15	14	13	12	11	10	9	8
BITS_NAME	NewDat48	NewDat47	NewDat46	NewDat45	NewDat44	NewDat43	NewDat42	NewDat41
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BITS_OFFSET	7	6	5	4	3	2	1	0
BITS_NAME	NewDat40	NewDat39	NewDat38	NewDat37	NewDat36	NewDat35	NewDat34	NewDat33
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

##### [bit15] to [bit0] NewDat48 to NewDat33: Data update bit

The number corresponds to the message object number.

Value	Description
0	Indicates that no valid data resides.
1	Indicates that valid data resides.

– CAN new data register 2 (NEWDT2)

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	NewDat32	NewDat31	NewDat30	NewDat29	NewDat28	NewDat27	NewDat26	NewDat25
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	NewDat24	NewDat23	NewDat22	NewDat21	NewDat20	NewDat19	NewDat18	NewDat17
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

**[bit15] to [bit0] NewDat32 to NewDat17: Data update bit**

The number corresponds to the message object number.

Value	Description
0	Indicates that no valid data resides.
1	Indicates that valid data resides.

– CAN new data register 1 (NEWDT1)

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	NewDat16	NewDat15	NewDat14	NewDat13	NewDat12	NewDat11	NewDat10	NewDat9
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	NewDat8	NewDat7	NewDat6	NewDat5	NewDat4	NewDat3	NewDat2	NewDat1
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

**[bit15] to [bit0] NewDat16 to NewDat1: Data update bit**

The number corresponds to the message object number.

Value	Description
0	Indicates that no valid data resides.
1	Indicates that valid data resides.



The following shows conditions to set or reset the NewDat bit.

- Setting conditions
  - Set "1" to the WR/RD bit of the IFx Command Mask Register, and "1" to the Control bit, and "1" to the NewDat bit of the IFx Message Control Register. Then write data to the IFx Command Request Register to set the NewDat bit to "1" in a specific message object.
  - The NewDat bit is set to "1" by receiving a data frame that passed through the acceptance filter.
  - If the Dir bit is "1", the RmtEn bit is "0", and the UMask bit is "1", the NewDat bit is set to "1" by receiving a remote frame that passed through the acceptance filter.
- Resetting conditions
  - Set "0" to the WR/RD bit of the IFx Command Mask Register and "1" to the NewDat bit, and write data to the IFx Command Request Register to reset the NewDat bit to "0" in a specific message object.
  - Set "1" to the WR/RD bit of the IFx Command Mask Register, and "1" to the Control bit, and "0" to the NewDat bit of the IFx Message Control Register. Then write data to the IFx Command Request Register to reset the NewDat bit to "0" in a specific message object.
  - The NewDat bit is reset to "0" after data has been transferred to the transmission shift register (internal register).

### 4.4.3. CAN Interrupt Pending Register 1/2/3/4 (INTPND1, INTPND2, INTPND3, INTPND4)

The CAN Interrupt Pending Register indicates the IntPnd bit of all message objects. This register checks which message object is pending for interrupt by reading the IntPnd bit.

- CAN interrupt pending register 4 (INTPND4)

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	IntPnd64	IntPnd63	IntPnd62	IntPnd61	IntPnd60	IntPnd59	IntPnd58	IntPnd57
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	IntPnd56	IntPnd55	IntPnd54	IntPnd53	IntPnd52	IntPnd51	IntPnd50	IntPnd49
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

#### [bit15] to [bit0] IntPnd64 to IntPnd49: Interrupt pending bit

The number corresponds to the message object number.

Value	Description
0	No interrupt cause is detected.
1	An interrupt cause is detected.

- CAN interrupt pending register 3 (INTPND3)

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	IntPnd48	IntPnd47	IntPnd46	IntPnd45	IntPnd44	IntPnd43	IntPnd42	IntPnd41
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	IntPnd40	IntPnd39	IntPnd38	IntPnd37	IntPnd36	IntPnd35	IntPnd34	IntPnd33
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

#### [bit15] to [bit0] IntPnd48 to IntPnd33: Interrupt pending bit

The number corresponds to the message object number.

Value	Description
0	No interrupt cause is detected.
1	An interrupt cause is detected.





– CAN interrupt pending register 2 (INTPND2)

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	IntPnd32	IntPnd31	IntPnd30	IntPnd29	IntPnd28	IntPnd27	IntPnd26	IntPnd25
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	IntPnd24	IntPnd23	IntPnd22	IntPnd21	IntPnd20	IntPnd19	IntPnd18	IntPnd17
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

**[bit15] to [bit0] IntPnd32 to IntPnd17: Interrupt pending bit**

The number corresponds to the message object number.

Value	Description
0	No interrupt cause is detected.
1	An interrupt cause is detected.

– CAN interrupt pending register 1 (INTPND1)

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	IntPnd16	IntPnd15	IntPnd14	IntPnd13	IntPnd12	IntPnd11	IntPnd10	IntPnd9
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	IntPnd8	IntPnd7	IntPnd6	IntPnd5	IntPnd4	IntPnd3	IntPnd2	IntPnd1
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

**[bit15] to [bit0] IntPnd16 to IntPnd1: Interrupt pending bit**

The number corresponds to the message object number.

Value	Description
0	No interrupt cause is detected.
1	An interrupt cause is detected.

#### 4.4.4. CAN Message Valid Register 1

The CAN Message Valid Register indicates the MsgVal bit of all message objects. This register checks which message object is valid by reading the MsgVal bit.

- CAN message valid register 4 (MSGVAL4)

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	MsgVal64	MsgVal63	MsgVal62	MsgVal61	MsgVal60	MsgVal59	MsgVal58	MsgVal57
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	MsgVal56	MsgVal55	MsgVal54	MsgVal53	MsgVal52	MsgVal51	MsgVal50	MsgVal49
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

##### [bit15] to [bit0] MsgVal64 to MsgVal49: Message valid bit

The number corresponds to the message object number.

Value	Description
0	Message objects are invalid. Disables message sending/receiving.
1	Message objects are valid. Enables message sending/receiving.

- CAN message valid register 3 (MSGVAL3)

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	MsgVal48	MsgVal47	MsgVal46	MsgVal45	MsgVal44	MsgVal43	MsgVal42	MsgVal41
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	MsgVal40	MsgVal39	MsgVal38	MsgVal37	MsgVal36	MsgVal35	MsgVal34	MsgVal33
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

##### [bit15] to [bit0] MsgVal48 to MsgVal33: Message valid bit

The number corresponds to the message object number.

Value	Description
0	Message objects are invalid. Disables message sending/receiving.
1	Message objects are valid. Enables message sending/receiving.



– CAN message valid register 2 (MSGVAL2)

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	MsgVal32	MsgVal31	MsgVal30	MsgVal29	MsgVal28	MsgVal27	MsgVal26	MsgVal25
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	MsgVal24	MsgVal23	MsgVal22	MsgVal21	MsgVal20	MsgVal19	MsgVal18	MsgVal17
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

**[bit15] to [bit0] MsgVal32 to MsgVal17: Message valid bit**

The number corresponds to the message object number.

Value	Description
0	Message objects are invalid. Disables message sending/receiving.
1	Message objects are valid. Enables message sending/receiving.

– CAN message valid register 1 (MSGVAL1)

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	MsgVal16	MsgVal15	MsgVal14	MsgVal13	MsgVal12	MsgVal11	MsgVal10	MsgVal9
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	MsgVal8	MsgVal7	MsgVal6	MsgVal5	MsgVal4	MsgVal3	MsgVal2	MsgVal1
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

**[bit15] to [bit0] MsgVal16 to MsgVal1: Message valid bit**

The number corresponds to the message object number.

Value	Description
0	Message objects are invalid. Disables message sending/receiving.
1	Message objects are valid. Enables message sending/receiving.

The following shows conditions to set or reset the MsgVal bit.

- Setting conditions  
Set "1" to the WR/RD bit of the IFx Command Mask Register, and "1" to the Arb bit, and "1" to the MsgVal bit of the IFx Arbitration Register 2. Then write data to the IFx Command Request Register to set the MsgVal bit to "1" in a specific message object.
- Resetting conditions  
Set "1" to the WR/RD bit of the IFx Command Mask Register, and "1" to the Arb bit, and "0" to the MsgVal bit of the IFx Arbitration Register 2. Then write data to the IFx Command Request Register to reset the MsgVal bit to "0" in a specific message object.





## CHAPTER 32: CAN Message RAM ECC

This section explains the CAN message RAM ECC.

---

1. Overview
2. Configuration
3. Interrupt
4. Operation
5. ECC Error Processing
6. Registers
7. Precautions for Using



## 1. Overview

In the CAN message RAM, CAN message objects are stored. The ECC function of the CAN message RAM enables detection and correction of data errors of the message RAM.

The CAN message RAM ECC function has the following features.

- 2-bit error detection and 1-bit error correction for 136-bit data of the message RAM
- Support for selection of whether to stop or continue CAN controller operation upon error detection and correction
- Support for the error generation function of the message RAM

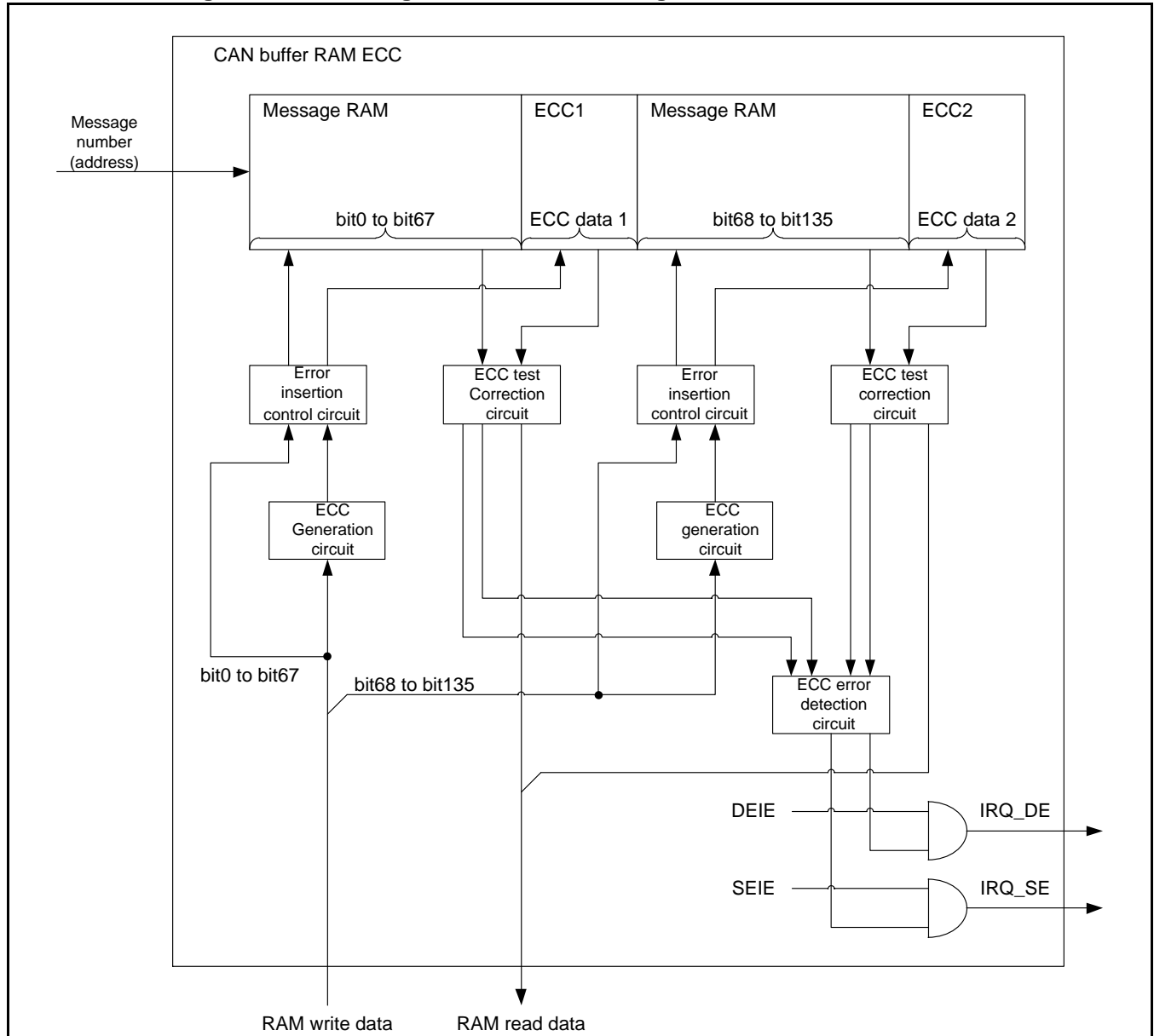
**Note:**

- *For details on the CAN controller, see "CHAPTER: CAN Controller."*

## 2. Configuration

A block diagram of the CAN message RAM ECC is shown below.

**Figure 2-1 Block Diagram of the CAN Message RAM ECC**



- Message RAM  
Consists of 136-bit x 64-word RAM and stores the message objects for each message box.
- ECC1/ECC2  
Made up of 8 bits, and stores ECC data of bit67 to bit0 of the message RAM in ECC1, and ECC data of bit135 to bit68 of the message RAM in ECC2.
- ECC generation circuit  
Generates ECC data from data to be written in the message RAM.
- Error insertion control circuit  
Controls whether to generate ECC data that causes an ECC error or to output normal ECC data.





- ECC test correction circuit  
Combines message RAM data and ECC data together to check for a single- or double-bit error. For a single-bit error, this circuit corrects the data.
- ECC error detection circuit  
Generates CANEESR: DEI and CANEESR: SEI from 2 pairs of a single-bit error and a double-bit error.

### 3. Interrupt

Supported interrupts are shown.

**Table 3-1 Interrupt Controller Bits and Interrupt Factors of the CAN Message RAM ECC**

Interrupt Type	Interrupt Request Flag Bit	Flag Register	Interrupt Factor	Interrupt Factor Enable Bit	Clearing of Interrupt Request Flag
Double-bit error interrupt	DEI	CANEESR	2-bit data error	CANEECR:DEIE	Write "1" to the double-bit error clear bit (CANEESCR: DEIC).
Single-bit error interrupt	SEI	CANEESR	1-bit data error	CANEECR:SEIE	Write "1" to the single-bit error clear bit (CANEESCR: SEIC).



## 4. Operation

This section explains the operation and function of the CAN message RAM ECC.

This section explains the following functions.

- Message RAM ECC generation
- Message RAM ECC test
- ECC error insertion function of the message RAM

## 4.1. Message RAM ECC Generation

The CAN message RAM is made up of the width of 136 bits and an additional 16 bits for ECC data. When data is written to the CAN message RAM, ECC data is generated from the write data based on the syndrome, and is written together with the write data at the same time.



## 4.2. Message RAM ECC Test

When a message object is read from the message RAM, the ECC data is also read. In the ECC test, calculation is performed based on the syndrome to determine whether the data in the message object is correct.

### (1) ECC Test

If it is determined from the ECC test result that the data in the message object is incorrect due to a RAM defect or some other problem, the following operation is performed.

- In the case where only 1 bit in the data is incorrect (single-bit error)  
The single-bit error occurrence bit (CANEEER: SEI) is set to "1" and the message number is set in the CAN ECC single-bit error address register (CANSEEAR). The incorrect 1-bit data is corrected and becomes normal data in the message object.
- In the case where 2 bits in the data are incorrect (double-bit error)  
The double-bit error occurrence bit (CANEEER: DEI) is set to "1" and the message number is set in the CAN ECC double-bit error address register (CANDEEAR). The incorrect data is not corrected and is treated as is in the message object.

If the interrupt enable bit corresponding to each error bit (CANEEER: SEI, CANEEER: DEI) has been set to "1", an interrupt occurs when "1" is set in the error bit.

### (2) Stopping of the CAN Controller upon Error Detection

If an error is generated during message RAM inspection, the CAN controller can be stopped by setting the initialization bit (CTRLR: Init) in the CAN controller to "1". Set the initialization bit (CTRLR: Init) in the CAN controller to "1" under any one of the following conditions.

- If a double-bit error is detected (CANEEER: DEI="1") by reading the object message from the message RAM when transmission starts with the double-bit error CAN stop bit (CANEECR: DEIXS) set to "1"
- If a single-bit error is detected (CANEEER: SEI="1") by reading the object message from the message RAM when transmission starts with the single-bit error CAN stop bit (CANEECR: SEIXS) set to "1"

### (3) Disabling ECC Test

ECC test for each message object is disabled after power-on because the contents of the message RAM are undefined. The ECC test for each message object changes from disabled to enabled under the following condition.

- When the message number is written into the specified message object via the interface register of the CAN controller

For example, if "1" is set in the RD/WR bit in the IFx command mask register (IFxCMSK) of the CAN controller and "0x01" is set in Message Number in the IFx command request register (IFxCREQ), ECC test for message object 1 changes from disabled to enabled.

ECC test changes from enabled to disabled under one of the following conditions.

- Hardware reset (all message objects change from enabled to disabled)
- When the CEIVEN bit in the CAN ECC error control register (CANEECR) is "1", and if each bit of MsgVal64-1 in the CAN message valid registers (MSGVAL1 to MSGVAL4) of the CAN controller is changed from "1" to "0" (the message object corresponding to the changed bit changes from enabled to disabled)

**Notes:**

- The CAN controller reads the message RAM during transmission. However, even if an ECC error occurs, the CAN controller does not stop. Even if an ECC error occurs after the message RAM is read via the interface of the CAN controller, the CAN controller also does not stop.
  - If transmission data overwriting occurs during transmission when the CAN controller `TxRqst=NewDat="1"` is set for the transmission message object, the controller reads the message RAM. In this case, if an ECC error is detected and the function for stopping the CAN controller upon the occurrence of an error is enabled, the CAN controller is stopped even if it is transmitting.
  - When the CAN controller is in basic mode (`TESTR: BASIC="1"`), the message RAM is not used. Therefore, set the double-bit error CAN stop bit (`CANEECR: DEIXS`), single-bit error CAN stop bit (`CANEECR: SEIXS`), and each interrupt enable bit (`CANEECR: DEIE`, `CANEECR: SEIE`) to "0".
  - The initialization bit (`CTRLR: Init`) in the CAN controller is set to "1" 1 operation clock after an error is detected.
  - If the `DEIXS` and `SEIXS` bits need to be changed, do so while the CAN controller is stopped (`CTRLR: Init="1"`).
  - When the CAN controller is stopped upon an ECC error, if the error bit (`CANEESR: DEI` or `CANEESR: SEI`) is "1" and the corresponding CAN stop bit (`CANEECR: DEIXS` or `CANEECR: SEIXS`) is "1", the initialization bit in the CAN controller remains set to "1". If you want to set the initialization bit in the CAN controller to "0", perform one of the following and then set the initialization bit in the CAN controller to "0".
    - Clear the error bit which stopped the CAN controller
    - Write "0" to the CAN stop bit that stopped the CAN controller
- However, if you write "1" again to the CAN stop bit without clearing the error bit, the initialization bit in the CAN controller is set to "1".
- When each bit of `MsgVal64-1` in the CAN message valid registers (`MSGVAL1` to `MSGVAL4`) of the CAN controller changes from "1" to "0", data is read once from the message RAM before the ECC test is disabled. Therefore, an ECC error may be detected.



### 4.3. ECC Error Insertion Function of the Message RAM

This function is used for testing program operation for when an ECC error occurs.

This allows you to write data that causes an ECC error into the message RAM. By reading such data from the message RAM, you can generate a single- or double-bit error. If the error interrupt has been enabled, an interrupt occurs. If ECC error processing has been programmed in the interrupt processing, you can verify whether the ECC error processing operates correctly.

#### Setting Method

The setting methods for message RAM reading and transmitting/receiving are as follows.

- Reading the message RAM
  1. Make setting of the CAN ECC error insertion control register (CANEFECR).  
Set FERR to "1", and then set each bit of EY[9] to EY[0] and EI[15] to EI[0] in which you want an error to occur to "1".
  2. Write to the message RAM from the interface register of the CAN controller<sup>\*1</sup>.  
The bits are inverted according to the CAN ECC error insertion control register (CANEFECR), and then they are written to the message RAM.
  3. Read the message RAM data via the interface register of the CAN controller<sup>\*1</sup>.  
Reading the message RAM generates an ECC error.

<sup>\*1</sup>: For details on reading from/writing to the message RAM, see the description of the CAN controller.
- When transmitting
 

# For step 1, use the same setting procedure as that described in "Reading the message RAM".

  1. Make setting of the CAN ECC error insertion control register (CANEFECR).
  2. Set a transmission request in the interface register of the CAN controller.  
As soon as the bits are inverted according to the CAN ECC error insertion control register (CANEFECR) and then written to the message RAM, transmission starts. The message RAM is first read after starting transmission, thus generating an ECC error.
- When receiving (when scanning the message RAM)
 

Make this setting using the same procedure as that described in "Reading the message RAM" and wait for receiving.  
The message RAM is scanned for data, such as ID, upon receiving, which generates an ECC error.
- When receiving (when finishing receiving)
  1. Write to the message RAM from the interface register of the CAN controller.  
A normal message object is written to the message RAM.
  2. Make setting of the CAN ECC error insertion control register (CANEFECR).  
Set FERR to "1", set each bit of EY[9] to EY[0] and EI[15] to EI[0] in which you want an error to occur to "1", and then wait for transmission to complete.
  3. Read the message RAM via the interface register of the CAN controller, thus generating an ECC error.

## 5. ECC Error Processing

If an ECC error occurs and the error interrupt has been enabled, the error must be handled in the interrupt processing. The following shows an example of the interrupt processing.

- When using an object message other than that which caused a double-bit error (when not stopping the CAN controller)
  1. Read the CAN ECC status register.  
Check the ECC error type.
  2. Write to another message object when a double-bit error occurs.  
Use another message object by writing to a message object that does not use the data such as ID of the message object which caused the error.
  3. Set the CEIVEN bit in the CAN ECC error control register (CANEECR) to "1".  
Allow disabling of the ECC test to exclude the message object which caused the error from the ECC test.
  4. Set the MsgVal bit in IFx arbitration register 2 (IFxARB2) to 0", the RD/WR bit in the IFx command mask register (IFxCMASK) to "1", and Message Number in the IFx command request register (IFxCREQ) to the message number for the message object which caused the error.

The above settings exclude a message object for which MsgVal bit was changed from "1" to "0" from the ECC test.

  5. Set the CEIVEN bit in the CAN ECC error control register (CANEECR) to "0".  
Prohibit disabling of the ECC test.
  6. Set the DEIC bit in the CAN ECC error status clear register (CANEESCR) to "1".  
Clear the double-bit error. Performing step 4 reads the message RAM, thus generating a double-bit error. Therefore, clear the double-bit error after prohibiting disabling of the ECC error test.
- When using an object message other than that which caused a double-bit error (when stopping the CAN controller)

For steps 1 through 6, use the same procedure as that used in "when not stopping the CAN controller."

  7. Set the Init bit in the CAN control register (CTRLR) to "0".  
Join the CAN bus. If the bit representing a double-bit error (CANEESR:DEI) is "1", you cannot write "0" to the Init bit. Therefore, clear the double-bit error before writing "0" to the Init bit.





## 6. Registers

This section explains the registers of the CAN message RAM ECC function.

The prefix "CANxx\_" is added to every register name (abbreviation).

xx is the channel number (00, 01, or 02).

**Table 6-1 List of CAN RAM ECC Registers**

Abbreviated Register Name	Register Name	See
CANEECR	CAN ECC Error Control Register	6.1
CANEESR	CAN ECC Error Status Register	6.2
CANEESCR	CAN ECC Error Status Clear Register	6.3
CANDEEAR	CAN ECC Double-bit Error Address Register	6.4
CANSEEAR	CAN ECC Single-bit Error Address Register	6.5
CANEFECR	CAN ECC Error Insertion Control Register	6.6

## 6.1. CAN ECC Error Control Register (CANEECR)

The CAN ECC error control register (CANEECR) is used to set whether to enable the interrupt when single-bit error correction or double-bit error detection occurs during the ECC test. It also sets whether to stop or continue operation of the CAN controller when an ECC error is detected.

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved			CEIVEN	DEIXS	SEIXS	DEIE	SEIE
ACCESS_TYPE	R0,W0			R/W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	000			0	0	0	0	0

**[bit7:5] Reserved: Reserved bits**

**[bit4] CEIVEN: ECC test disable allow bit**

Value	Description
0	Enable the ECC test.
1	Disable the ECC test.

- When CEIVEN="0", and even if the falling edge of each bit of MsgVal64-1 in the CAN message valid registers (MSGVAL1 to MSGVAL4) of the CAN controller is detected, the ECC test is not disabled.
- When CEIVEN="1", and if the falling edge of each bit of MsgVal64-1 in the CAN message valid registers (MSGVAL1 to MSGVAL4) of the CAN controller is detected, the message object changes from enabled to disabled.

**Note:**

- When each bit of MsgVal64-1 in the CAN message valid registers (MSGVAL1 to MSGVAL4) of the CAN controller changes from "1" to "0", data is read from the message RAM before the ECC test is disabled. Therefore, an ECC error may be detected.

**[bit3] DEIXS: Double-bit error CAN stop bit**

Value	Description
0	Allow the communication of the CAN controller to continue.
1	Stop the communication of the CAN controller.

- When DEIXS="0", and even if a double-bit error is detected, the CAN controller continues to communicate.
- When transmission starts with DEIXS="1", and if a double-bit error is detected, set the Init bit in the CAN control register (CTRLR) to "1" to stop the CAN controller.

**Notes:**

- The CAN controller reads the message RAM during transmission. However, even if an error occurs, the CAN controller does not stop. Even if an error occurs after the message RAM is read via the interface of the CAN controller, the CAN controller also does not stop.
- If transmission data overwriting occurs during transmission when the CAN controller TxRqst=NewDat="1" is set for the transmission message object, the controller reads the message RAM. In this case, if an ECC error is detected and the function for stopping the CAN controller upon the occurrence of an error is enabled, the CAN controller is stopped even if it is transmitting.



- When the CAN controller is in basic mode (TESTR: BASIC="1"), the message RAM is not used. Therefore, set the double-bit error CAN stop bit (CANEECR: DEIXS), single-bit error CAN stop bit (CANEECR: SEIXS), and each interrupt enable bit (CANEECR: DEIE, CANEECR: SEIE) to "0".
- The initialization bit (CTRLR: Init) in the CAN controller is set to "1" 1 operation clock after an error is detected.
- If the DEIXS and SEIXS bits need to be changed, do so while the CAN controller is stopped (CTRLR: Init="1").
- When the CAN controller is stopped upon the occurrence of an ECC error, and if the error bit is "1" and the corresponding CAN stop bit is "1", the initialization bit in the CAN controller remains set to "1". If you want to set the initialization bit in the CAN controller to "0", perform one of the following and then set the initialization bit in the CAN controller to "0".
  - Clear the error bit which stopped the CAN controller
  - Write "0" to the CAN stop bit which stopped the CAN controller

However, if you write "1" again to the CAN stop bit without first clearing the error bit, the initialization bit in the CAN controller is set to "1".

**[bit2] SEIXS: Single-bit error CAN stop bit**

Value	Description
0	Allow the communication of the CAN controller to continue.
1	Stop the communication of the CAN controller.

- When SEIXS="0", and even if a single-bit error is detected, the CAN controller continues communication.
- When transmission starts with SEIXS="1", and if a single-bit error is detected, set the Init bit in the CAN control register (CTRLR) to "1" to stop the CAN controller.

**Notes:**

- The CAN controller reads the message RAM during transmission. However, even if an error occurs, the CAN controller does not stop. Even if an error occurs after the message RAM is read via the interface of the CAN controller, the CAN controller also does not stop.
- If transmission data overwriting occurs during transmission when the CAN controller TxRqst=NewDat="1" is set for the transmission message object, the controller reads the message RAM. In this case, if an ECC error is detected and the function for stopping the CAN controller upon the occurrence of an error is enabled, the CAN controller is stopped even if it is transmitting.
- When the CAN controller is in basic mode (TESTR: BASIC="1"), the message RAM is not used. Therefore, set the double-bit error CAN stop bit (CANEECR: DEIXS), single-bit error CAN stop bit (CANEECR: SEIXS), and each interrupt enable bit (CANEECR: DEIE, CANEECR: SEIE) to "0".
- The initialization bit (CTRLR: Init) in the CAN controller is set to "1" 1 operation clock after an error is detected.
- If the SEIXS bit needs to be changed, do so while the CAN controller is stopped (CTRLR: Init="1").
- When the CAN controller is stopped upon the occurrence of an ECC error, and if the error bit is "1" and the corresponding CAN stop bit is "1", the initialization bit in the CAN controller remains set to "1". If you want to set the initialization bit in the CAN controller to "0", perform one of the following and then set the initialization bit in the CAN controller to "0".
  - Clear the error bit which stopped the CAN controller
  - Write "0" to the CAN stop bit that stopped the CAN controller

However, if you again write "1" to the CAN stop bit without clearing the error bit, the initialization bit in the CAN controller is set to "1".

**[bit1] DEIE: Double-bit error factor interrupt enable bit**

Value	Description
0	Disable the interrupt caused by the double-bit error (CANEESR: DEI).
1	Enable the interrupt caused by the double-bit error (CANEESR: DEI).

- When DEIE is "0", even if a double-bit error is detected (CANEESR:DEI = "1"), the IRQ\_DE signal remains "L".
- When DEIE is "1", if a double-bit error is detected (CANEESR:DEI = "1"), the IRQ\_DE signal becomes "H" to request an interrupt.

**[bit0] SEIE: Single-bit error factor interrupt enable bit**

Value	Description
0	Disable the interrupt caused by the single-bit error (CANEESR: SEI).
1	Enable the interrupt caused by the single-bit error (CANEESR: SEI).

- When SEIE is "0", even if a single-bit error is detected (CANEESR:SEI = "1"), the IRQ\_SE signal remains "L".
- When SEIE is "1", if a single-bit error is detected (CANEESR:SEI = "1"), the IRQ\_SE signal becomes "H" to request an interrupt.



## 6.2. CAN ECC Error Status Register (CANEESR)

The CAN ECC error status register (CANEESR) displays whether a single-bit error has been corrected in the ECC test and whether a double-bit error has been detected. When the enable bit becomes "1", it remains "1" unless it is cleared by using the CAN ECC error status clear register (CANEESCR).

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved						DEI	SEI
ACCESS_TYPE	R0,W0						R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	000000						0	0

[bit7:2] Reserved: Reserved bits

[bit1] DEI: Double-bit error occurrence bit

Value	Description
0	Indicate that a double-bit error has not been detected.
1	Indicate that a double-bit error has been detected.

- When the DEI bit is "1", to clear this bit to "0", write "1" to the DEIC bit in the CAN ECC error status clear register (CANEESCR).

### Notes:

- When the DEI bit is "1", the message RAM is read, and even if a double-bit error occurs for another message number, the CAN ECC double-bit error address register (CANDEEAR) is not updated and retains the previous value.
- Even if the DEI bit is "1", a single-bit error can be detected.

[bit0] SEI: Single-bit error occurrence bit

Value	Description
0	Indicate that a single-bit error has not been detected.
1	Indicate that a single-bit error has been detected.

- When the SEI bit is "1", to clear this bit to "0", write "1" in the SEIC bit in the CAN ECC error status clear register (CANEESCR).

### Notes:

- When the SEI bit is "1", the message RAM is read, and even if a single-bit error occurs for another message number, the CAN ECC single-bit error address register (CANSEEAR) is not updated and the previous value is retained.
- Even if the SEI bit is "1", a double-bit error can be detected.

### 6.3. CAN ECC Error Status Clear Register (CANEESCR)

This clears the bit in the CAN ECC error status register.

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved						DEIC	SEIC
ACCESS_TYPE	R0,W0						R0,W	R0,W
PROT_TYPE	-							
INITIAL_VALUE	000000						0	0

[bit7:2] Reserved: Reserved bits

[bit1] DEIC: Double-bit error clear bit

Value	Description
0	No effect
1	Change the double-bit error occurrence bit (CANEESR: DEI) to "0".

[bit0] SEIC: Single-bit error clear bit

Value	Description
0	No effect
1	Change the single-bit error occurrence bit (CANEESR: SEI) to "0".



## 6.4. CAN ECC Double-bit Error Address Register (CANDEEAR)

The register retains the message number (address in the message RAM) of an error which occurs during double-bit error detection in the ECC test. This register is valid when the DEI bit in the CAN ECC status register (CANEESR) is "1". While the DEI bit in the CAN ECC status register (CANEESR) is "1", the value of this register is retained.

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved							
ACCESS_TYPE	R0,W0							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved	DMN						
ACCESS_TYPE	R0,W0	R,WX						
PROT_TYPE	-							
INITIAL_VALUE	0	0000001						

[bit15:7] Reserved: Reserved bits

[bit6:0] DMN[6:0]: Double error message number bits

Value	Description
0x01	Indicates that a double-bit error has occurred at message number 1.
0x02	Indicates that a double-bit error has occurred at message number 2.
.....	.....
0x3F	Indicates that a double-bit error has occurred at message number 63.
0x40	Indicates that a double-bit error has occurred at message number 64.

- DMN is valid when the DEI bit in the CAN ECC status register (CANEESR) is "1".
- While the DEI bit in the CAN ECC status register (CANEESR) is "1", the value of this register is retained.

## 6.5. CAN ECC Single-bit Error Address Register (CANSEEAR)

The register retains the message number (address in the message RAM) of an error which occurs during single-bit error detection in the ECC test. This register is valid when the SEI bit in the CAN ECC status register (CANEESR) is "1". While the SEI bit in the CAN ECC status register (CANEESR) is "1", the value of this register is retained.

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved							
ACCESS_TYPE	R0,W0							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved	SMN						
ACCESS_TYPE	R0,W0	R,WX						
PROT_TYPE	-							
INITIAL_VALUE	0	0000001						

**[bit15:7] Reserved: Reserved bits**

**[bit6:0] SMN[6:0]: Single error message number bits**

Value	Description
0x01	Indicates that a single-bit error has occurred at message number 1.
0x02	Indicates that a single-bit error has occurred at message number 2.
.....	.....
0x3F	Indicates that a single-bit error has occurred at message number 63.
0x40	Indicates that a single-bit error has occurred at message number 64.

- SMN is valid when the SEI bit in the CAN ECC status register (CANEESR) is "1".
- While the SEI bit in the CAN ECC status register (CANEESR) is "1", the value of this register is retained.





## 6.6. CAN ECC Error Insertion Control Register (CANEFECCR)

This register specifies the bytes and bits to generate an error.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	FERR	Reserved					EY[9:8]	
ACCESS_TYPE	R/W	R0,W0					R/W	
PROT_TYPE	-							
INITIAL_VALUE	0	00000					00	

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	EY[7:0]							
ACCESS_TYPE	R/W							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

BIT_OFFSET	15-0
BIT_NAME	EI
ACCESS_TYPE	R/W
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000

### [bit31] FERR: Error insertion enable bit

Value	Description
0	Disable writing of error data.
1	Enable writing of error data.

When the FERR bit is "1", the error data is written during transmission to the message RAM via the interface register of the CAN controller according to EY and EI.

### [bit30:26] Reserved: Reserved bits

### [bit25:16] EY[9:0]: Error byte specification bits

	Description
EY[0]	Sets bit15 to 0 in the message RAM as error targets.
EY[1]	Sets bit31 to 16 in the message RAM as error targets.
EY[2]	Sets bit47 to 32 in the message RAM as error targets.
EY[3]	Sets bit63 to 48 in the message RAM as error targets.
EY[4]	Sets bit79 to 64 in the message RAM as error targets.
EY[5]	Sets bit95 to 80 in the message RAM as error targets.
EY[6]	Sets bit111 to 96 in the message RAM as error targets.
EY[7]	Sets bit127 to 112 in the message RAM as error targets.
EY[8]	Sets bit135 to 128 in the message RAM as error targets.
EY[9]	Sets bit151 to 136 in the message RAM (bits for ECC) error targets.

- When "1" is set, the specified 2-byte range of the message RAM is the error target.
  - Example 1) In the case of EY[0] = "1", and EY[1:9] = "0", bit15 to bit0 in the message RAM are the error targets.

- Example 2) In the case of EY[0] = EY[3] = "1", and EY[1] = EY[2] = EY[4:9] = "0", bit15 to bit0 and bit63 to bit48 in the message RAM are the error targets.
- When each bit is all "0", error data is not written.
- EY[8] sets the 1-bit range in the message RAM as the error target. Use EI[7:0] to specify bits.

**[bit15:0] EI[15:0]: Error bit specification bits**

	Description
EI[0]	In the 2 bytes specified by EY, bit0 is an error target.
EI[1]	In the 2 bytes specified by EY, bit1 is an error target.
EI[2]	In the 2 bytes specified by EY, bit2 is an error target.
EI[3]	In the 2 bytes specified by EY, bit3 is an error target.
EI[4]	In the 2 bytes specified by EY, bit4 is an error target.
EI[5]	In the 2 bytes specified by EY, bit5 is an error target.
EI[6]	In the 2 bytes specified by EY, bit6 is an error target.
EI[7]	In the 2 bytes specified by EY, bit7 is an error target.
EI[8]	In the 2 bytes specified by EY, bit8 is an error target.
EI[9]	In the 2 bytes specified by EY, bit9 is an error target.
EI[10]	In the 2 bytes specified by EY, bit10 is an error target.
EI[11]	In the 2 bytes specified by EY, bit11 is an error target.
EI[12]	In the 2 bytes specified by EY, bit12 is an error target.
EI[13]	In the 2 bytes specified by EY, bit13 is an error target.
EI[14]	In the 2 bytes specified by EY, bit14 is an error target.
EI[15]	In the 2 bytes specified by EY, bit15 is an error target.

- When data is written via the interface register of the CAN controller, if "1" has been set, the specified bits in the specified 2 bytes are inverted before they are written.
  - Example 1) In the case of EY[3] = "1", EY[0:2] = EY[4:9] = "0", EI[3] = "1", and EI[0:2] = EI[4:15] = "0", the bit data for bit51 in the message RAM is inverted before it is written. The other bits are written as they are.
  - Example 2) In the case of EY[0] = EY[3] = "1", EY[1] = EY[2] = EY[4:9] = "0", EI[3] = "1", and EI[0:2] = EI[4:15] = "0", the bit data for bit3 and bit51 in the message RAM is inverted before it is written. The other bits are written as they are.
- When each bit in EI is all "0", error data is not written.



## 7. Precautions for Using

This section explains precautions on the use of the CAN message RAM ECC.

### ECC Test

- The CAN controller reads the message RAM during transmission. However, even if an error occurs, the CAN controller does not stop. Even if an error occurs after the message RAM is read via the interface of the CAN controller, the CAN controller also does not stop.
- When the CAN controller is in basic mode (TESTR: BASIC="1"), the message RAM is not used. Therefore, set the double-bit error CAN stop bit (CANEECR: DEIXS), single-bit error CAN stop bit (CANEECR: SEIXS), and each interrupt enable bit (CANEECR: DEIE, CANEECR: SEIE) to "0".
- The initialization bit (CTRLR: Init) in the CAN controller is set to "1" 1 operation clock after an error is detected.
- If the CAN stop bits (CANEECR: DEIXS and CANEECR: SEIXS bits) need to be changed, do so while the CAN controller is stopped (CTRLR: Init="1").
- When the CAN controller is stopped upon the occurrence of an ECC error, and if the error bit is "1" and the corresponding CAN stop bit is "1", the initialization bit in the CAN controller remains "1". If you want to set the initialization bit in the CAN controller to "0", perform one of the following and then set the initialization bit in the CAN controller to "0".
  - Clear the error bit which stopped the CAN controller
  - Write "0" to the CAN stop bit that stopped the CAN controllerHowever, if you write "1" again to the CAN stop bit without clearing the error bit, the initialization bit in the CAN controller is set to "1".
- When each bit of MsgVal64-1 in the CAN message valid registers (MSGVAL1 to MSGVAL4) of the CAN controller changes from "1" to "0", data is read once from the message RAM before the ECC test is disabled. Therefore, an ECC error may be detected.



## CHAPTER 33: CAN Prescaler

This chapter explains the CAN prescaler.

---

1. Overview
2. Configuration
3. Operation
4. Registers



## 1. Overview

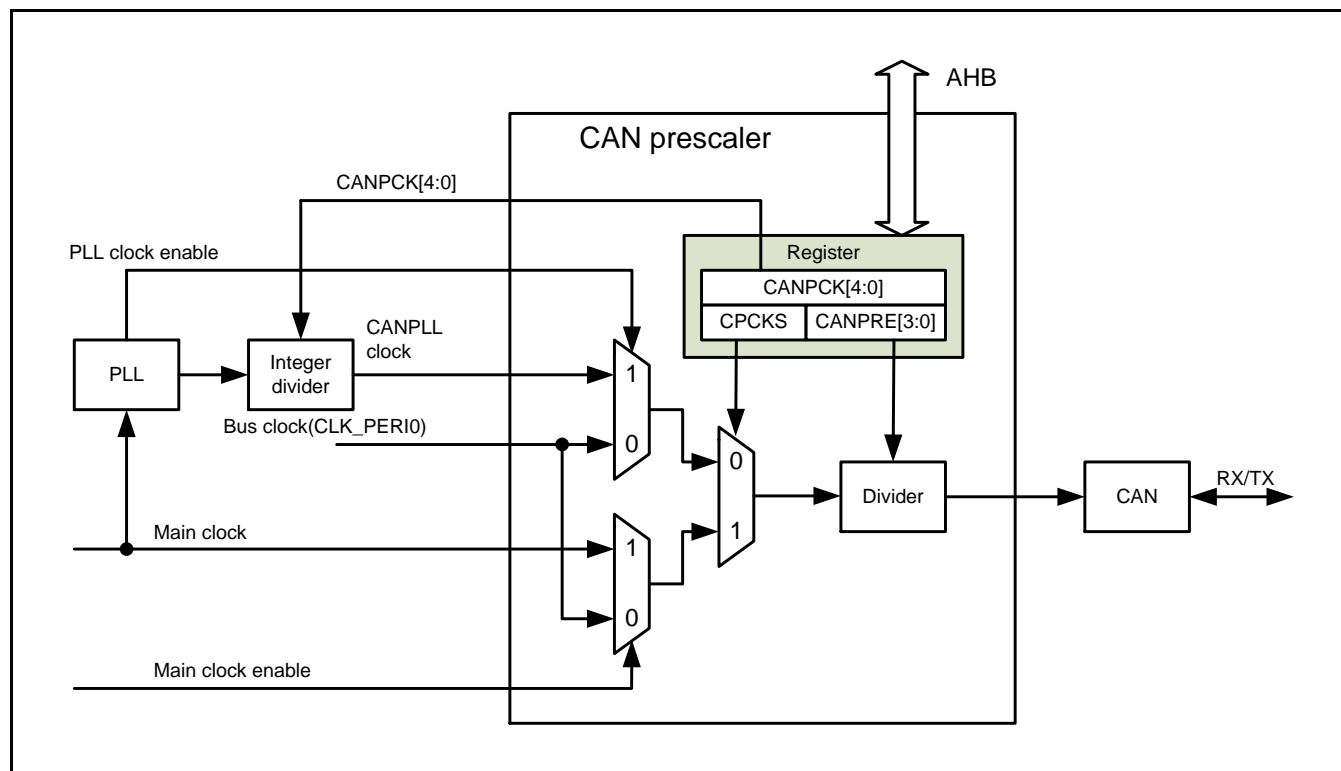
The CAN prescaler generates CAN clocks to be supplied to CAN based on each clock source supplied from the clock system.

- CANPLL clock, bus clock, and main clock are selectable for a source clock of the CAN prescaler.
  - CANPLL clock and bus clock are switched by the PLL clock enable signal indicating oscillation stabilized state of the PLL clock. For details, see "CHAPTER: Clock System" and "CHAPTER: Low-power Consumption".
  - Main clock and bus clock are switched by the main clock enable signal indicating oscillation stabilized state of the main clock. For details, see "CHAPTER: Clock System" and "CHAPTER: Low-power Consumption".
- It is equipped with a divider that can change CAN system clock cycle, so it outputs a clock obtained by dividing an input clock by 1 to 12 depending on its setting.

## 2. Configuration

This section provides a block diagram of CAN prescaler.

Figure 2-1 Block Diagram of CAN Prescaler





### 3. Operation

This section describes the operation of CAN prescaler.

#### CAN System Clock Settings

##### a) CANPLL Clock

CANPLL clock is a clock obtained by integer division of the PLL oscillation clock. With the divider in the CAN prescaler, it can generate the intended CAN system clock.

Set CPCKS to "0".

Set CANPCK[4:0] to determine the CANPLL clock.

For setting the value of the division, set CANPCK[4:0] to "0x01" (divided by 2) or higher.

$$\text{CANPLL clock} = \text{PLL oscillation clock} / (\text{CANPCK} + 1)$$

Set the value used for division of CANPRE[3:0] to the determined CANPLL clock.

When the PLL oscillation clock is 400MHz, we recommend that you use CANPCK to set the CANPLL clock to 80MHz or lower.

Example) PLL oscillation clock: For generating CAN system clock 16MHz from 400MHz (PLL clock 200MHz), make any of the following settings.

1. Divide the PLL oscillation clock 400MHz by 25 to generate 16MHz.

CPCKS	:	"0"	
CANPCK[4:0]	:	"0x18"	(Divided by 25)
CANPRE[3:0]	:	"0x0"	(Divided by 1)

2. Divide the PLL oscillation clock 400MHz by 5, set the CANPLL clock to 80MHz, and use the CAN prescaler to divide it by 5 to generate 16MHz.

CPCKS	:	"0"	
CANPCK[4:0]	:	"0x04"	(Divided by 5)
CANPRE[3:0]	:	"0xC" or "0xD"	(Divided by 5)

##### b) Bus Clock

Use CANPRE[3:0] to set the value to be used for division.

##### c) Main Clock

Set CPCKS to "0x1", and set CANPRE to the value to be used for division.

For details, see Section "4. Registers" and "CHAPTER: Clock Ssystem".

#### Notes:

- Before changing the CAN prescaler setting bit, set the initial bit (CTRLR.Init) of the CAN control register to "0x1" to stop all the bus operations.
- The clock to be supplied by the register setting to the CAN interface should be 16MHz or lower.
- The PLL oscillation clock is a clock to be divided by an even-number to generate the PLL clock.



4. Registers

This section describes the registers of CAN prescaler.

Table 4-1 List of CAN Prescaler Registers

Abbreviated Register Name	Register Name	See
CANP_CANPRE	CAN Prescaler Control Register	4.1
CANP_CANPCK	CAN PLL Clock Control Register	4.2





## 4.1. CAN Prescaler Control Register (CANP\_CANPRE)

This register sets the CAN system clock prescaler.

Before changing the register value, set the initial bit (Init) of the CAN control register (CANxx\_CTRLR) to "1" to stop all the bus operations.

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved			CPCS	CANPRE			
ACCESS_TYPE	R0,WX			R/W	R/W			
PROT_TYPE	-							
INITIAL_VALUE	000			0	0000			

**[bit31:5] Reserved: Reserved bits**

**[bit4] CPCKS: CAN prescaler clock selection**

This bit selects a source clock of the CAN prescaler.

Value	Description
0	CANPLL clock/CLK_PERI0 (bus clock)
1	Main oscillation clock/CLK_PERI0 (bus clock)

When the CANPLL clock and the main clock are stopped, a bus clock is selected.

**[bit3:0] CANPRE[3:0]: CAN prescaler setting bits**

These bits set the CAN system clock frequency.

Value	Description	Input CAN Prescaler Clock (MHz)			
		160	128	80	48
0000	Select a 1/1 cycle.	160.0	128.0	80.0	48.0
0001	Select a 1/2 cycle.	80.0	64.0	40.0	24.0
001X	Select a 1/4 cycle.	40.0	32.0	20.0	12.0
01XX	Select a 1/8 cycle.	20.0	16.0	10.0	6.0
1000	Select a 2/3 cycle. Duty is 67%.	106.7	85.3	53.3	32.0
1001	Select a 1/3 cycle.	53.3	42.7	26.7	16.0
1010	Select a 1/6 cycle.	26.7	21.3	13.3	8.0
1011	Select a 1/12 cycle.	13.3	10.7	6.7	4.0
110X	Select a 1/5 cycle.	32.0	25.6	16.0	9.6
111X	Select a 1/10 cycle.	16.0	12.8	8.0	4.8

X: don't care

## 4.2. CAN PLL Clock Control Register (CANP\_CANPCK)

This register sets the CANPLL clock to be input to the CAN prescaler.

Before changing the register value, set the initial bit (Init) of the CAN control register (CTRLR) to "1" to stop all the bus operations.

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved			CANPCK				
ACCESS_TYPE	R0,WX			R/W				
PROT_TYPE	-							
INITIAL_VALUE	000			00001				

**[bit31:5] Reserved: Reserved bits**

### **[bit4:0] CANPCK[4:0]: Prescaler clock selections**

These bits select the CANPLL clock frequency to be input to the CAN prescaler.

The value obtained by dividing the PLL oscillation clock by the value added 1 to the CANPCK[4:0] value is the CANPLL clock. Divisions by 2 to 32 are available.

Do not set CANPCK[4:0] = "0x00" (divided by 1).

CANPLL clock = PLL oscillation frequency / (CANPCK + 1)

Example) PLL oscillation clock = 400MHz, CANPCK = "0x18"

CAN PLL clock = 400 / (24 + 1) = 16 [MHz]

Value	Description
00000	Disable
00001	PLLout/2
00010	PLLout/3
00011	PLLout/4
-----	-----
11101	PLLout/30
11110	PLLout/31
11111	PLLout/32

PLLout: PLLout oscillation frequency



# CHAPTER 34: Multi-function Serial Interface



This chapter explains the multi-function serial interface.

---

1. Overview
2. Configuration



## 1. Overview

The multi-function serial interface has the following features.

### Interface Mode

Depending on the operation mode setting, any of the following interface modes can be selected for the multi-function serial interface.

- UART0 (Asynchronous normal serial interface)
- UART1 (Asynchronous multi-processor serial interface)
- CSIO (Clock synchronous serial interface) (SPI support available)
- LIN (LIN interface)

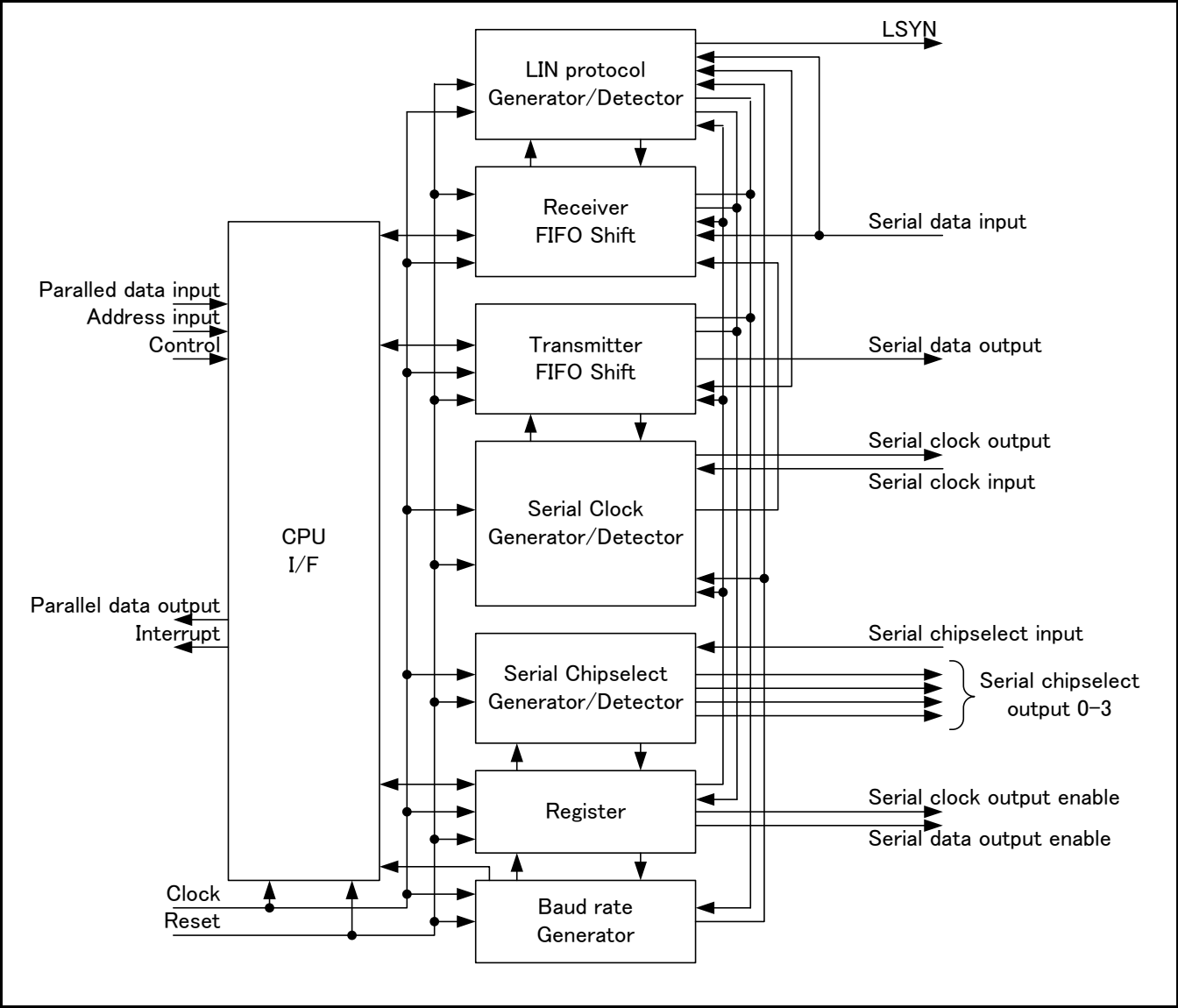
### Note:

- *For details on each interface, see "CHAPTER: UART", "CHAPTER: CSIO" and "CHAPTER: LIN Interface (v2.1)".*

2. Configuration

This section explains the multi-function serial interface configuration.

Figure 2-1 Configuration Diagram of Multi-function Serial Interface





## Explanation of Each Block

### a) Baud Rate Generator

This 15-bit reload counter functions as a dedicated baud rate generator. This block consists of 15-bit registers for reload values, and it generates reception and transmission clocks and LIN Break field detection clocks from external or internal clocks. Also, the count value of the transmission reload counter is read from BGR1 or BGR0.

### b) Receiver/FIFO Shift

This block consists of a reception shift register, reception FIFO, reception bit counter, start bit detection circuit, and reception parity counter. When the reception bit counter counts the reception data bit and completes the reception of 1 piece of data corresponding to the set data length, the reception data full flag bit (SSR:RDRF) is set to "1". At this time, if reception interrupts are enabled, it generates a reception interrupt request. The start bit detection circuit detects the start bit from a serial input signal, synchronizes the falling edge of the start bit, and sends a signal to the reload counter. The reception parity counter calculates the reception data parity.

The reception shift register fetches the received data input from the SIN (serial data input) pin while bits are shifted. After the completion of reception, the register transfers the received data to the RDR register, or to the reception FIFO when using the reception FIFO.

### c) Transmitter/FIFO Shift

This block consists of a transmission shift register, transmission bit counter, transmission start circuit, and transmission parity counter. The transmission bit counter counts the transmission data bits and transmits 1 piece of data corresponding to the set data length. When the transmission bit counter indicates the start of write data transmission, a flag is set in the serial status register. At this time, if transmission interrupts are enabled, it generates a transmission interrupt request. The transmission start circuit starts a transmission operation with data writing of the TDR register. If the transmission parity counter uses parity bits, it generates a parity bit for the transmission data.

The transmission shift register transfers the written data in the TDR register, or in the transmission FIFO when using the transmission FIFO, to the transmission shift register and outputs it to SOUT (serial data output) while bits are shifted.

### d) Serial Clock Generator/Detector

The serial clock generator generates a serial clock during master operations in CSIO mode.

The serial clock detector detects the serial clock during slave operations in CSIO mode. There is synchronization with the detected serial clock to transmit and receive serial data.

### e) LIN Protocol Generator/Detector

The LIN protocol Detector detects the LIN Break field when the LIN master node transmits the message header. Upon detecting the LIN Break field, the detector sets "1" in the LIN Break field detection flag bit (SSR:LBD). It outputs an internal signal (LSYN) to a capture unit to detect the first and the fifth falling edges of the LIN Sync field and to measure the synchronization of the actual serial clock transmitted by the LIN master node.

The LIN protocol Generator generates the LIN break field corresponding to the length selected with the LIN break field length selection bit in the extended status control register.

### f) Serial Chip Select Generator/Detector

The serial chip select generator controls each chip select pin during master operations in CSIO mode.

The serial chip select detector monitors the chip select input status during slave operations in CSIO mode.

**g) Register**

This is the setting part of each register.

**h) CPU I/F**

This is the interface part between the bus and a register.





# CHAPTER 35: UART (Asynchronous Serial Interface)

This chapter describes the UART (Asynchronous Serial Interface) function supported by operation modes 0 and 1 of the multi-function serial interface.



1. Overview
2. Interrupts
3. Operation
4. Serial Timer Operation
5. Test Mode
6. Dedicated Baud Rate Generator
7. Setup Procedure and Program Flow for Operation Mode 0 (Asynchronous Normal Mode)
8. Setup Procedure and Program Flow for Operation Mode 1 (Asynchronous Multi-processor Mode)
9. Registers
10. Precautions for Using



## 1. Overview

UART (Asynchronous Serial Interface) is a general-purpose serial data communication interface for asynchronous communication with external devices. This interface supports bidirectional communication (normal mode) and master/slave-type communication (multi-processor mode: both master and slave roles are supported). In addition, this interface has FIFOs for transmission and reception.

### UART (Asynchronous Serial Interface) Functions

Item		Function
1	Data	<ul style="list-style-type: none"> <li>- Full-duplex double buffer (when FIFO is not used)</li> <li>- Transmission and reception FIFOs (64 bytes for each : when FIFO is used)</li> </ul>
2	Serial input	<ul style="list-style-type: none"> <li>- Oversampling is performed by the bus clock 3 times. The received value is determined by the rule of majority.</li> </ul>
3	Transfer format	<ul style="list-style-type: none"> <li>- Asynchronous</li> </ul>
4	Baud rate	<ul style="list-style-type: none"> <li>- Dedicated baud rate generator (15-bit reload counter configuration)</li> <li>- The external clock input can be adjusted by the reload counter.</li> </ul>
5	Data length	<ul style="list-style-type: none"> <li>- 5 to 9 bits (for normal mode), or 7 to 8 bits (multi-processor mode)</li> </ul>
6	Signaling method	<ul style="list-style-type: none"> <li>- NRZ (Non Return to Zero) and inverted NRZ</li> </ul>
7	Start bit detection	<ul style="list-style-type: none"> <li>- Synchronized to falling edges of the start bit (for NRZ method)</li> <li>- Synchronized to rising edges of the start bit (for inverted NRZ method)</li> </ul>
8	Reception error detection	<ul style="list-style-type: none"> <li>- Framing error</li> <li>- Overrun error</li> <li>- Parity error<sup>*1</sup></li> </ul>
9	Synchronous transmission function	<ul style="list-style-type: none"> <li>- Can automatically transmit data periodically in synchronization with the serial timer.</li> </ul>
10	Timer function	<ul style="list-style-type: none"> <li>- A 16-bit serial timer is available.</li> <li>- A division value can be selected for the operation clock (1 to 256 divisions).</li> </ul>
11	Interrupt request	<ul style="list-style-type: none"> <li>- Reception interrupt (reception completion, framing error, overrun error, parity error<sup>*1</sup>)</li> <li>- Transmission interrupt (transmission data empty, transmission bus idle)</li> <li>- Transmission FIFO interrupt (when the transmission FIFO is not higher than the interrupt trigger level or when the transmission FIFO is empty)</li> <li>- DMA Transfer is supported for both transmission and reception.</li> <li>- Status interrupt (serial timer interrupt)</li> </ul>
12	Master/slave-type communication function (Multi-processor mode)	<ul style="list-style-type: none"> <li>- 1 (master) -to-n (slave) communication is possible. (Both master and slave systems are supported.)</li> </ul>
13	FIFO option	<ul style="list-style-type: none"> <li>- Transmission and reception FIFOs are available (transmission FIFO: 64 bytes, reception FIFO: 64 bytes).</li> <li>- Transmission FIFO and reception FIFO can be selected.</li> <li>- Transmission data can be retransmitted.</li> <li>- The timing of the reception FIFO interrupt can be changed by software.</li> <li>- Independent FIFO reset is supported.</li> </ul>

<sup>\*1</sup>: Parity errors occur only in normal mode.

## 2. Interrupts

UART has transmission/reception interrupts and status interrupts. Interrupt requests can be generated with the factors described below.

- When reception data is set in the reception data register (RDR), or when a reception error occurs
- When transmission data is transferred from the transmission data register (TDR) to the transmission shift register and transmission starts
- Transmission bus idle (no transmission)
- Transmission FIFO data request
- When the serial timer comparison value (STMCR) and the serial timer value (STMR) coincide



### Interrupts from the UART

Table 2-1 lists the interrupt control bits and interrupt factors of UART.

**Table 2-1 UART Interrupt Control Bits and Interrupt Factors**

Interrupt Types	Interrupt Request Flag Bit	Flag Register	Operation Mode		Interrupt Factor	Interrupt Factor Enable Bit	Interrupt Request Flag Clearing
			0	1			
Reception	RDRF	SSR	○	○	1-byte reception	SCR:RIE	Reading of reception data (RDR)
					Reception of as much data as the amount set in FBYTE		Reading of reception data (RDR) until the reception FIFO is empty
					Detection of reception idle for the 8-bit time or longer while the FRIIE bit is "1" and the reception FIFO contains valid data		
	ORE	SSR	○	○	Overrun error	SCR:RIE ECR:REIE	Writing "1" to the reception error flag clear bit (SSR:REC)
	FRE	SSR	○	○	Framing error		
	PE	SSR	○	×	Parity error		
Transmission	TDRE	SSR	○	○	The transmission register is empty.	SCR:TIE	Writing to the transmission data register (TDR), or writing "1" to the transmission FIFO operation enable bit (retransmission) when the bit value is "0" and the transmission FIFO contains valid data <sup>*1</sup>
	TBI	SSR	○	○	No transmission	SCR:TBIE	Writing to the transmission data register (TDR), or writing "1" to the transmission FIFO operation enable bit (retransmission) when the bit value is "0" and the transmission FIFO contains valid data <sup>*1</sup>
	FDRQ	FCR1	○	○	The transmission FIFO stores a data amount equal to or smaller than the FTICR setting value or is empty.	FCR1:FTIE	Writing "0" to the FIFO transmission data request bit (FCR1:FDRQ), or the transmission FIFO is full.
Status	TINT	SACSR	○	○	Coincidence of the serial timer register (STMR) and the serial timer comparison register (STMCR) values	SACSR: TINTE	Writing "0" to the timer interrupt flag bit (SACSR:TINT)

○: Support

×: Not support

<sup>\*1</sup>: Set the TIE bit to "1" after the TDRE bit becomes "0".

## 2.1. Occurrence of Reception Interrupts and Flag Set Timing

There are 2 reception interrupt types: Reception completion (SSR:RDRF) and reception error (SSR:PE, ORE, FRE).

### (1) Occurrence of Reception Interrupts and Flag Set Timing

Upon the detection of the first stop bit, the reception data is stored in the reception data register (RDR). Related flags are set upon the completion of reception (SSR:RDRF = 1) or upon a reception error (SSR:PE, ORE, FRE = 1). At that time, a reception interrupt occurs if the reception interrupt is enabled (SSR:RIE = 1).

**Note:**

- If a reception error occurs, the data in the reception data register (RDR) will be invalid.

Figure 2-1 Set Timing of RDRF (Reception Data Full) Flag Bit

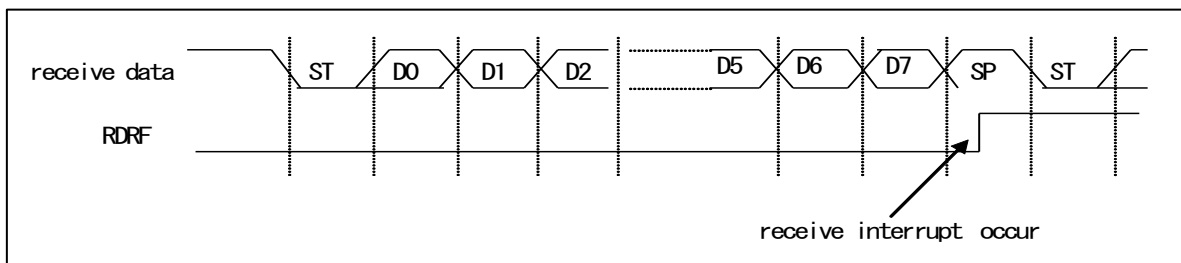
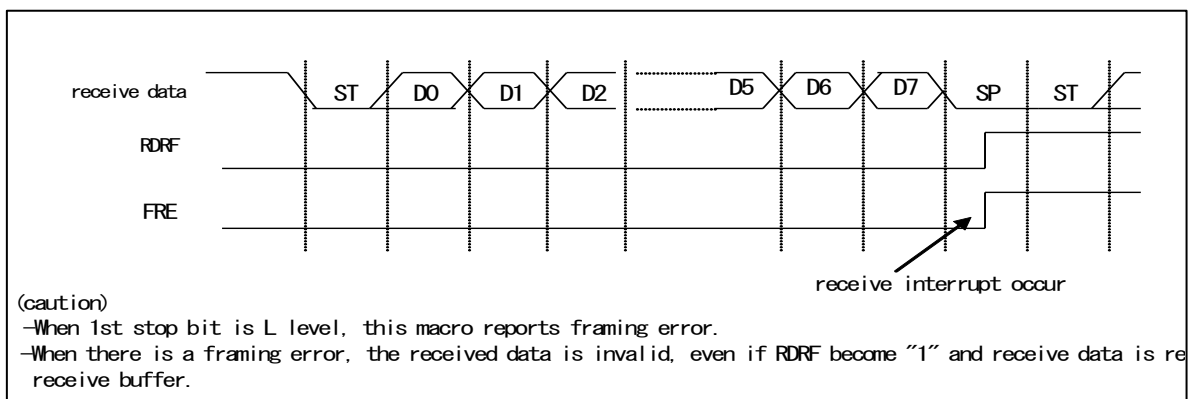


Figure 2-2 Set Timing of FRE (Framing Error) Flag Bit

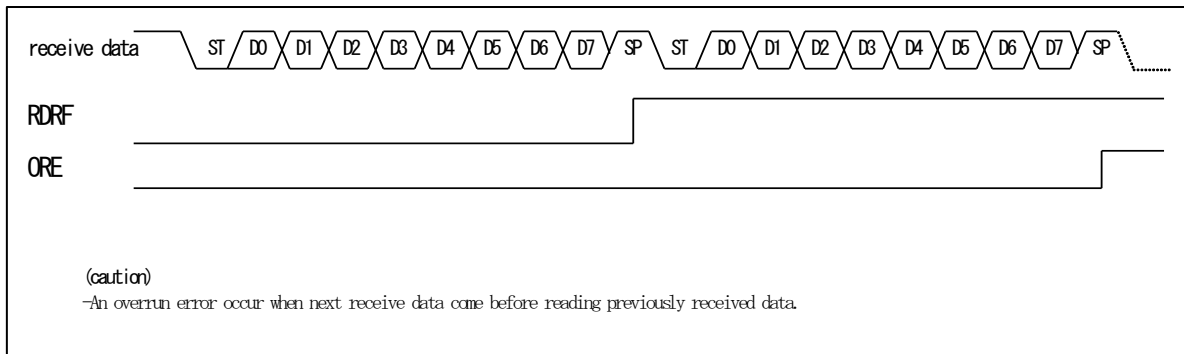


**Notes:**

- During data reception, the following will happen if one of the following edges is detected at the same time as, or 1 or 2 bus clocks earlier than, the stop bit sampling point: The edge becomes invalid and the data that follows may not be received normally. Consecutive frames must have intervals between them.
- Falling edge of serial data (when ESCR:INV="0")
- Rising edge of serial data (when ESCR:INV="1")



**Figure 2-3 Set Timing of ORE (Overrun Error) Flag Bit**



## 2.2. Occurrence of Interrupts and Flag Set Timing When the Reception FIFO is Used

When the reception FIFO is used, an interrupt occurs if an amount of data equal to the amount set in the FBYTE register (FBYTE) is received.

### Occurrence of Reception Interrupts and Flag Set Timing When the Reception FIFO is Used

When the reception FIFO is used, occurrence of interrupts is determined by the value set in the FBYTE register.

- If as much transfer data as the amount set in the FBYTE register is received, the reception data full flag (SSR:RDRF) in the serial status register is set to "1". At this time, a reception interrupt occurs if the reception interrupt is enabled (SCR:RIE).
- When both of the following conditions are satisfied, the continuation of reception idle status for 8 baud rate clocks or longer sets the interrupt flag (SSR:RDRF) to "1".
  - The reception FIFO idle detection enable bit (FCR1:FRIIE) is "1".
  - The number of data items in the reception FIFO does not reach the transfer count.
- During an 8-clock count, the counter is reset to 0 when RDR is read, then the system starts counting the 8 clocks again. The counter is reset to 0 when the reception FIFO is disabled. If the reception FIFO is enabled when there is data remaining in the reception FIFO, counting restarts.
- When the reception FIFO becomes empty as a result of reading the reception data (RDR), the reception data full flag (SSR:RDRF) is cleared.
- If the reception valid data number becomes equal to the value of the FIFO capacity, reception of any subsequent data triggers an overrun error (SSR:ORE = 1).

Figure 2-4 Occurrence Timing of Reception Interrupt When Reception FIFO is Used

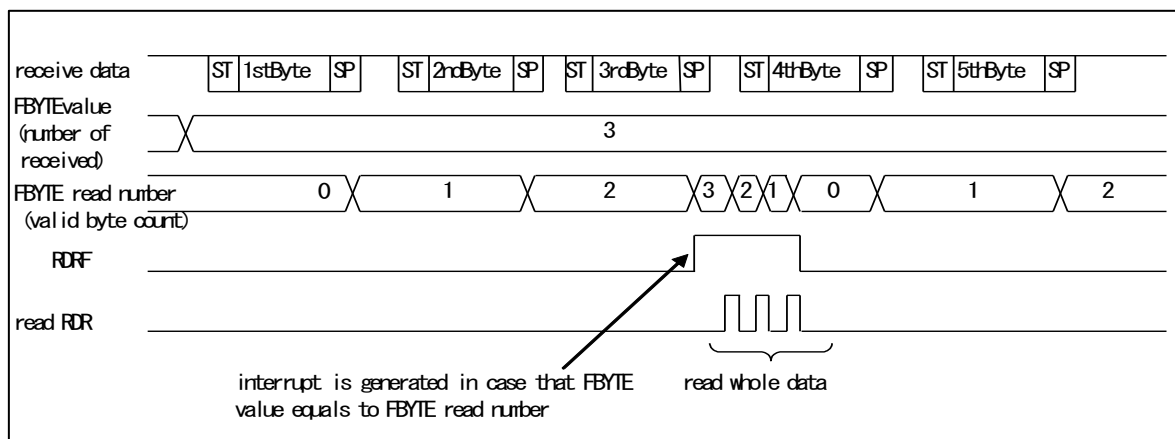
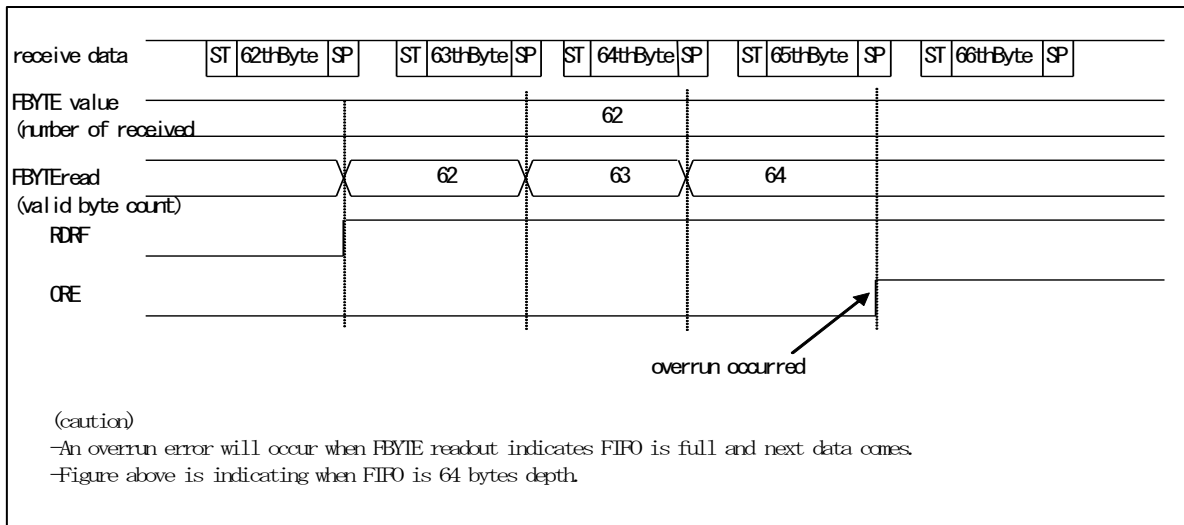






Figure 2-5 Set Timing of ORE (Overrun Error) Flag Bit



## 2.3. Occurrence of Transmission Interrupts and Flag Set Timing

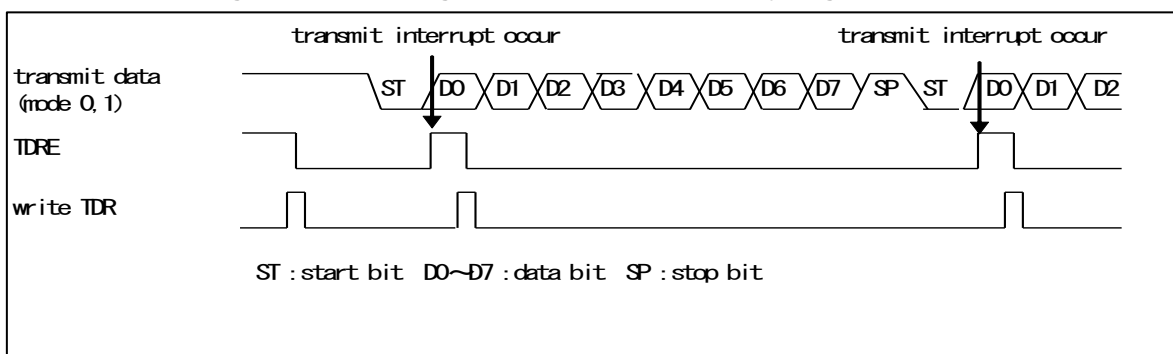
A data transmission interrupt occurs in the following cases: The transmission data is transferred from the transmission data register (TDR) to the transmission shift register (SSR:TDRE = 1) to start data transmission. The transmission operation is not performed (SSR:TBI = 1).

### Occurrence of Transmission Interrupts and Flag Set Timing

#### a) Set Timing of Transmission Data Empty Flag (SSR:TDRE)

Transfer of data from the transmission data register (TDR) to the transmission shift register allows the next data to be written (SSR:TDRE = 1). At that time, a transmission interrupt occurs if the transmission interrupt is enabled (SCR:TIE = 1). The SSR:TDRE bit is a read-only bit, and it is cleared to "0" when data is written to the transmission data register (TDR).

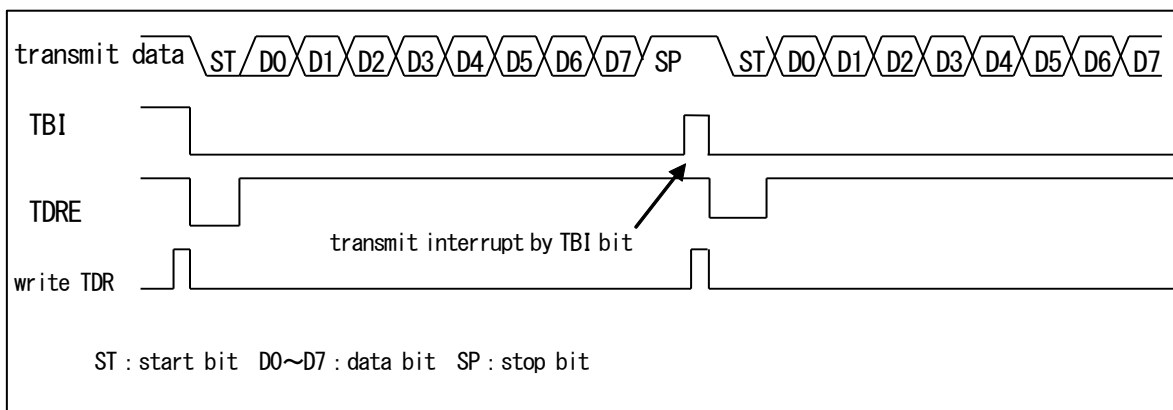
Figure 2-6 Set Timing of Transmission Data Empty Flag (SSR:TDRE)



#### b) Set Timing of Transmission Bus Idle Flag (SSR:TBI)

The SSR:TBI bit is set to "1" when the transmission data register is empty (SSR:TDRE = 1) and transmission is not performed. At that time, a transmission interrupt occurs if the transmission bus idle interrupt is enabled (SCR:TBIE = 1). Writing transmission data to the transmission data register (TDR) clears the SSR:TBI bit and the transmission interrupt request.

Figure 2-7 Set Timing of Transmission Bus Idle Flag (TBI) (Synchronous Transmission Disabled SACSR:TSYNE=0)



**Note:**

- *In the case of synchronous transmission (SACSR:TSYNE=1), the SSR:TBI bit is set to "1" if the transmission data register is empty (SSR:TDRE=0) after transmitting data for the byte count set in the transfer byte count (TBYTE0).*

## 2.4. Occurrence of Interrupts and Flag Set Timing When the Transmission FIFO is Used

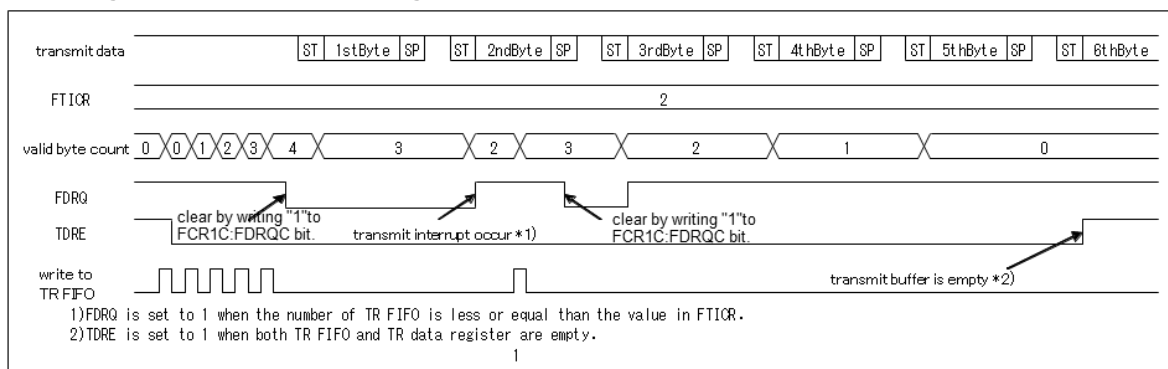
When the transmission FIFO is used, an interrupt occurs if the number of data items stored in the transmission FIFO is equal to or less than the setting of the FTICR register (FTICR).

### Occurrence of Transmission Interrupts and Flag Set Timing When the Transmission FIFO is Used

When the transmission FIFO is used, occurrence of interrupts is determined by the value set in the FTICR register.

- When the amount of data stored in the transmission FIFO is equal to or less than the value set for the FTICR register, the FIFO transmission data request bit (FCR1:FDRQ) is set to "1". At that time, a transmission interrupt occurs if the FIFO transmission interrupt is enabled (FCR1:FTIE=1).
- When you write the necessary data to the transmission FIFO after a transmission interrupt occurs, write "0" to the FIFO transmission data request bit (FCR1:FDRQ) to clear the interrupt request.
- The FIFO transmission data request bit (FCR1:FDRQ) is set to "0" once the transmission FIFO is full.
- The existence of data in the transmission FIFO can be verified by reading the FIFO byte register (FBYTE) or the transmission FIFO interrupt control register (FTICR).  
FBYTE=0x00 and FTICR=0x00 indicate that the transmission FIFO does not contain data.

Figure 2-8 Occurrence Timing of Transmission Interrupt When Transmission FIFO is Used





## 2.5. Timing of Timer Interrupt Generation and Flag Setting

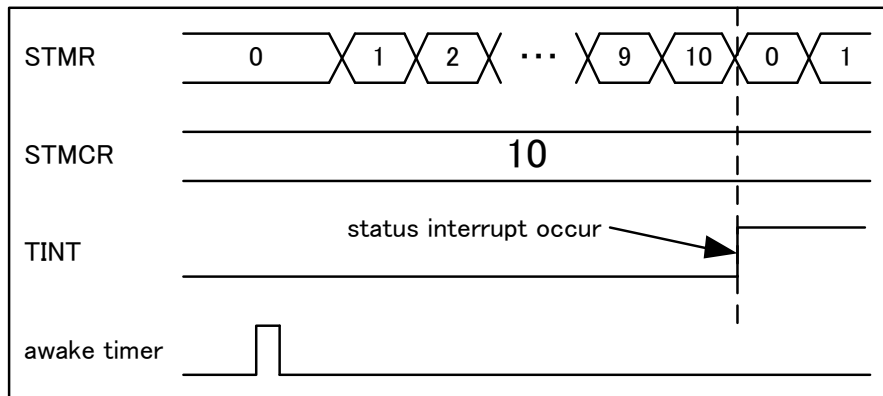
A timer interrupt occurs when the serial timer register (STMR) coincides with the serial timer comparison register (STMCR).

### Timing of Timer Interrupt Generation and Flag Setting

The timer interrupt flag (SACSR:TINT) is set to "1" when the serial timer register (STMR) coincides with the serial timer comparison register.

At that time, a status interrupt occurs if the timer interrupt is enabled (SACSR:TINTE=1).

**Figure 2-9 Occurrence Timing of Timer Interrupt**



### 3. Operation

UART operates in mode 0, which corresponds to bidirectional serial asynchronous communication, and also in mode 1, which corresponds to master/slave multi-processor communication.

#### UART Operation

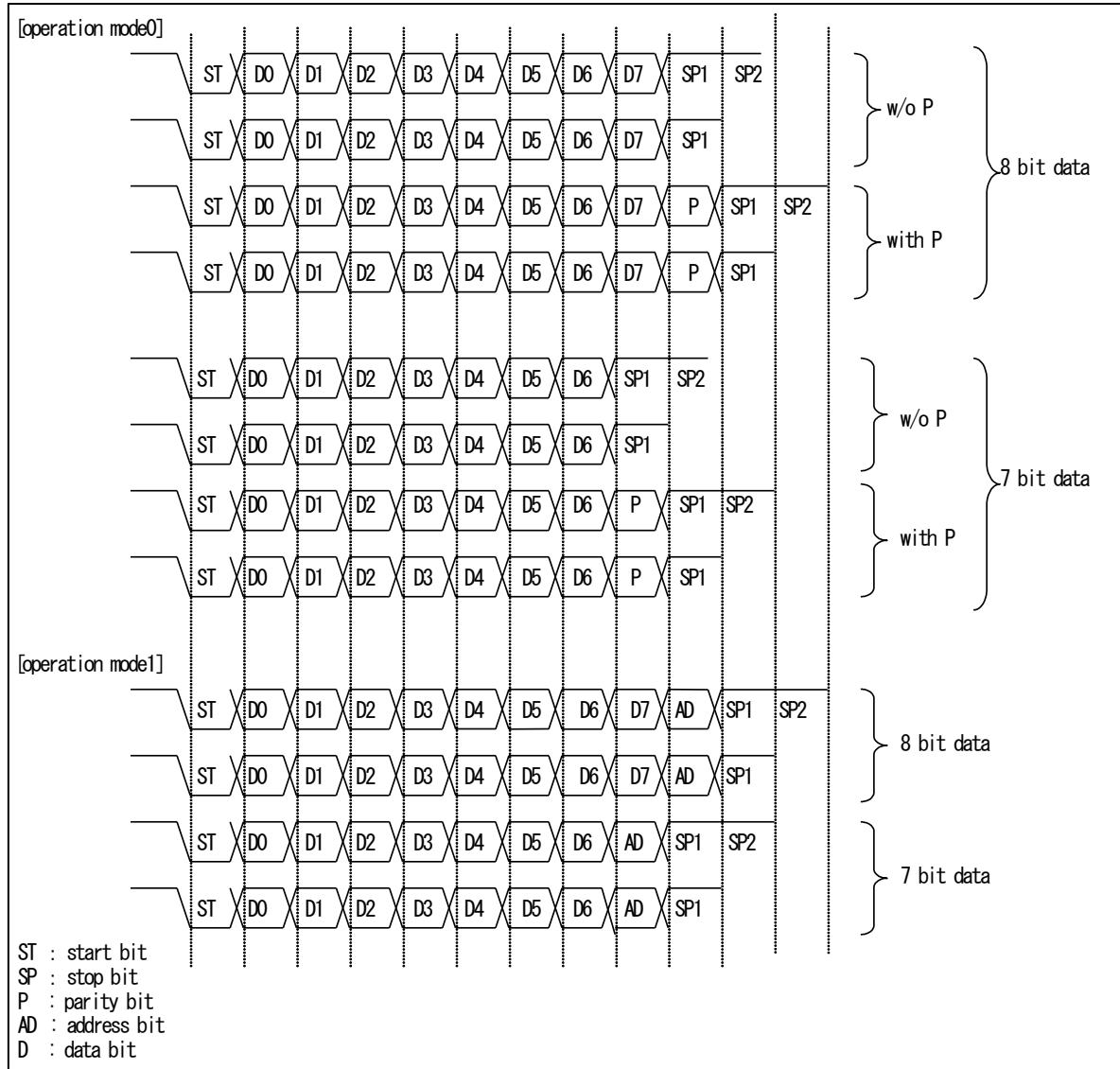
##### a) Transmission and Reception Data Format

- Transmission or reception data always starts with a start bit followed by transmission/reception of a specified data bit length and ending with at least one stop bit.
- The data transfer direction (LSB first or MSB first) is determined by the BDS bit in the serial mode register (SMR). If parity check is enabled, the parity bit is always placed between the last data bit and the first stop bit.
- Parity check can be enabled or disabled for operation mode 0 (normal mode).
- In operation mode 1 (multi-processor mode), AD bits are appended instead of parity bits.

Figure 3-1 shows the transmission and reception data formats of operation modes 0 and 1.



Figure 3-1 Example of Transmission and Reception Data Formats (Operation Modes 0 and 1)



**Notes:**

- The above figure show cases for data lengths of 7 bits and 8 bits. (Data length can be set to 5 bits to 9 bits for operation mode 0.)
- When the BDS bit in the serial mode register (SMR) is set to "1" (MSB first), bits are processed in the order of D7, D6, D5, ..., D1, D0 (P).
- If the data length is set to X bits, the lowest X bits of the transmission and reception data register (RDR/TDR) are valid.

**b) Transmission Operation**

- When the transmission data empty flag bit (TDRE) in the serial status register (SSR) is "1", transmission data can be written to the transmission data register (TDR). (When the transmission FIFO is enabled, transmission data can be written even if TDRE="0".)
- Writing transmission data to the transmission data register (TDR) sets the transmission data empty flag bit (SSR:TDRE) to "0".
- Setting the transmission operation enable bit (SCR:TXE) in the serial control register to "1" loads the transmission data to the transmission shift register and starts transmission from the start bit.
- The transmission data empty flag bit (SSR:TDRE) is set to "1" again when transmission starts. At this time, a transmission interrupt occurs if transmission interrupts are enabled (SCR:TIE=1). The interrupt processing can write the next transmission data to the transmission data register.

**Notes:**

- *Because the initial value of the transmission data empty flag bit (SSR:TDRE) is "1", a transmission interrupt occurs as soon as the transmission interrupt is enabled (SCR:TIE).*
- *Because the initial value of the FIFO transmission data request bit (FCR1:FDRQ) is "1", a transmission interrupt occurs as soon as the FIFO transmission interrupt is enabled (FCR1:FTIE=1).*



**c) Reception Operation**

- When the reception operation is enabled (SCR:RXE = 1), the reception operation starts.
- After a start bit is detected, 1-frame data is received according to the data format set in the extended communication control register (ESCR:PEN, P, L2, L1, and L0) and the serial mode register (SMR:BDS). A start bit is detected when both of the following are satisfied: 1. The noise-filtered result (the result of applying the rule of majority to the 3-time bus clock sampling of the serial data input) shows falling (when ESCR:INV = "0") or rising (when ESCR:INV = "1"). 2. The filtered data shows "L" at the sampling point.
- The reception data full flag bit (SSR:RDRF) is set to "1" when 1-frame data has been received. At that time, a reception interrupt occurs if the reception interrupt is enabled (SCR:RIE = 1).
- When reading the reception data, read the reception data after all of the 1-frame data has been received, and check the error flag in the serial status register (SSR). If a reception error occurs, perform error processing.
- Reading the reception data clears the reception data full flag bit (SSR:RDRF) to "0".
- If the reception FIFO is enabled, the reception data full flag bit (SSR:RDRF) is set to "1" upon the reception of as many frames of data as the value set in the reception FBYTE.
- When both of the following conditions are satisfied, continuation of reception idle status for 8 baud rate clocks or longer sets the interrupt flag (RDRF) to "1".
  - The reception FIFO idle detection enable bit (FRIIE) is "1".
  - The number of data items in the reception FIFO does not reach the transfer count.
- During an 8-clock count, the counter is reset to 0 when RDR is read, and the system starts counting the 8 clocks again. The counter is reset to 0 when the reception FIFO is disabled. If the reception FIFO is enabled while data remains in the reception FIFO, counting restarts.
- When the error flag in the serial status register (SSR) is set to "1" while the reception FIFO is enabled, the data related to the error is not stored in the reception FIFO. At that time, the reception data full flag bit (SSR:RDRF) is not set to "1". (However, an overrun error sets the RDRF flag to "1".) The reception FBYTE indicates the data number that had been received normally before the error occurred. The reception FIFO is not enabled unless the error flag in the serial status register (SSR) is cleared to "0".
- If the reception FIFO is enabled, the reception data full flag bit (SSR:RDRF) is cleared to "0" when the reception FIFO becomes empty.

**Notes:**

- *The data on the receive data register (RDR) is effective when the reception data register full flag bit (SSR:RDRF) is set to "1" and there has been no reception error (SSR:PE, ORE, FRE=0).*
- *There is a built-in noise filter (where the rule of majority is applied to the 3-time bus clock sampling of the serial data input), but erroneous data is received if noise has passed through the filter. We suggest that you design the board in such a way that noise does not pass through the filter. Alternatively, we suggest that you implement a communication method whereby noise passing through the filter does not cause a problem (by, for example, appending a checksum to the data at the end of transmission for retransmission in the event of an error).*
- *During data reception, the following will happen if one of the following edges is detected at the same time as, or 1 or 2 bus clocks earlier than, the stop bit sampling point: The edge becomes invalid and the data that follows may not be received normally. Consecutive frames must have intervals between them.*
  - *Falling edge of serial data (when ESCR:INV="0")*
  - *Rising edge of serial data (when ESCR:INV="1")*

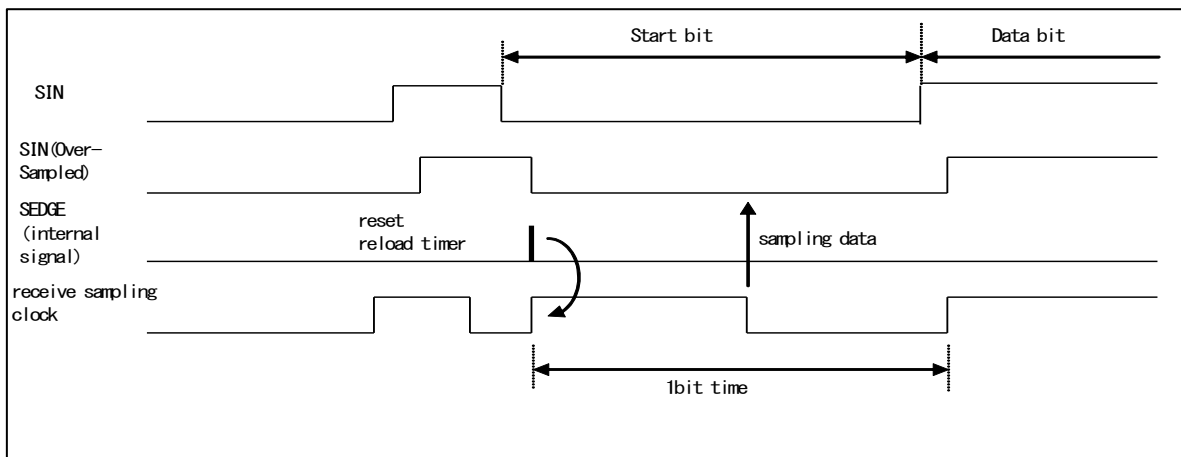
**d) Clock Selection**

- An internal clock or an external clock can be used.
- Set BGR1:EXT=1 when using an external clock. In this case, the external clock is divided by a baud rate generator.

**e) Start Bit Detection**

- In the case of asynchronous mode, a start bit is identified by a falling edge of the SIN signal. Therefore, enabling reception operation (SCR:RXE=1) does not start the reception operation unless there is a falling edge input of the SIN signal.
- When a falling edge of the start bit is detected, the reception reload counter of the baud rate generator is reset, reloaded again, and then starts counting down. This ensures that sampling is always performed in the center of the data.

**Figure 3-2 Start Bit Detection**



**f) Stop Bit**

- A bit length from 1 to 4 can be selected.
- The reception data full flag bit (SSR:RDRF) is set to "1" when the first stop bit is detected.

**g) Error Detection**

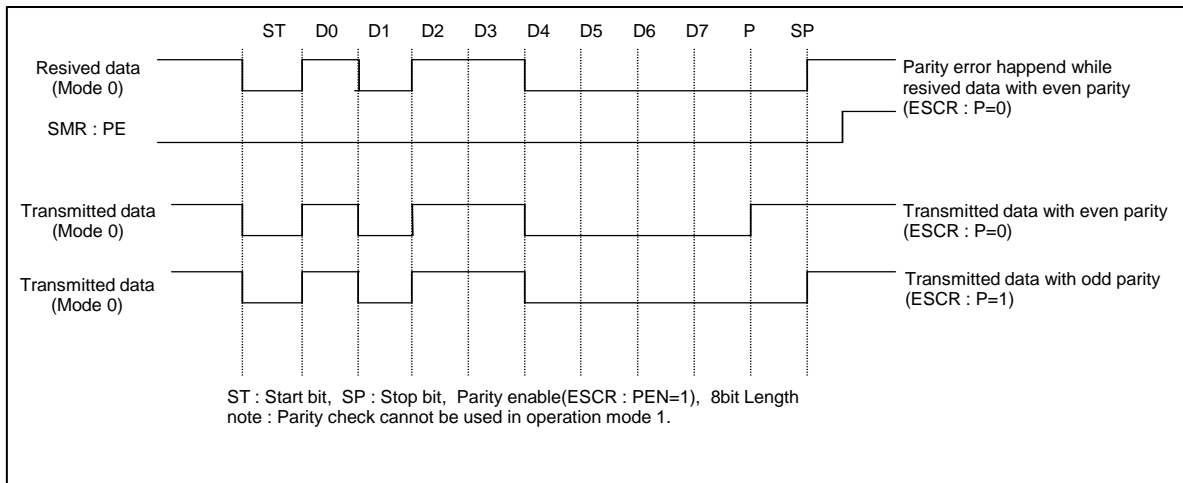
- Operation mode 0 allows the detection of parity errors, overrun errors, and framing errors.
- Operation mode 1 allows the detection of overrun errors and framing errors. The detection of parity errors is not possible.

**h) Parity Bit**

- A parity bit can be appended only in operation mode 0. The parity enable bit (ESCR:PEN) sets the presence or absence of parity check, while the parity selection bit (ESCR:P) sets even or odd parity.
- Parity check cannot be used in operation mode 1.
- Figure 3-3 shows the transmission and reception data when parity check is enabled.



**Figure 3-3 Operation with Parity Check Enabled**

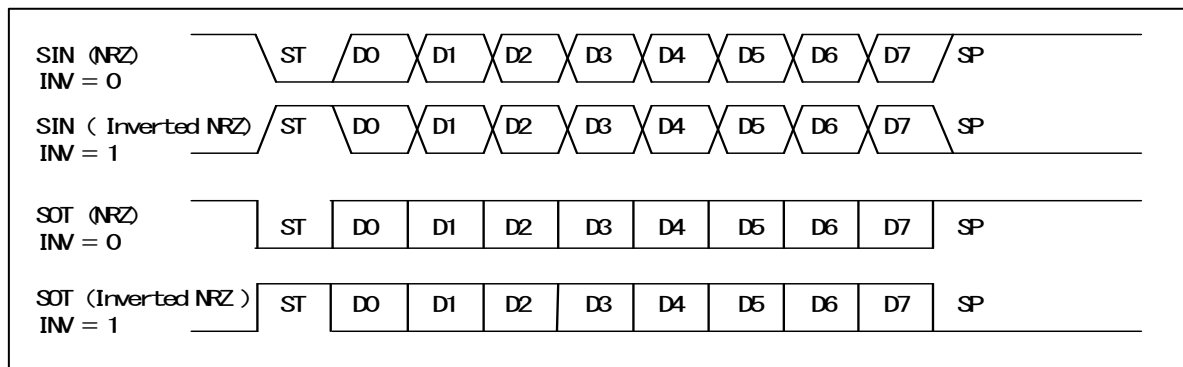


#### i) Data Signaling Method

Setting of the INV bit in the extended communication control register selects the NRZ (Non Return to Zero) signaling method (ESCR:INV=0) or the inverted NRZ signaling method (ESCR:INV=1).

Figure 3-4 shows the NRZ signaling method and the inverted NRZ signaling method.

**Figure 3-4 NRZ (Non Return to Zero) Signaling Method and Inverted NRZ Signaling Method**



#### j) Data Transfer Method

Either LSB first or MSB first can be selected as the data bit transfer method.

## 4. Serial Timer Operation

The serial timer can be used as either a timer function or a synchronous transmission function.

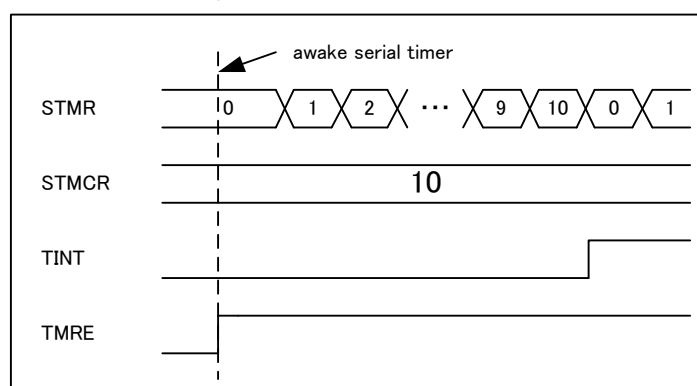
### Serial Timer Operation

#### a) Activating the Serial Timer

The serial timer can be activated by setting the serial timer enable bit (SACSR:TMRE) to "1".

- Activation by the serial timer enable bit (SACSR:TMRE)  
When the serial timer enable bit (SACSR:TMRE) is set to "1", the serial timer starts and causes the serial timer register (STMR) to start counting from 0.

Figure 4-1 Activation by Serial Timer Enable Bit (STMCR=10, TSYNE=0)



#### b) Stopping the Serial Timer

The serial timer stops when the serial timer enable bit (SACSR:TMRE) is set to "0". At that time, the value of the serial timer register (STMR) is retained.

#### c) Timer Operation

When the synchronous transmission enable bit (SACSR:TSYNE) is "0", the serial timer operates as a timer.

The timer interrupt flag (SACSR:TINT) is set to "1" and the serial timer register (STMR) is reset to 0 when the serial timer register (STMR) coincides with the serial timer comparison register (STMCR).



Figure 4-2 Timer Operation (STMCR=10, SACSRC:TSYNE=0)

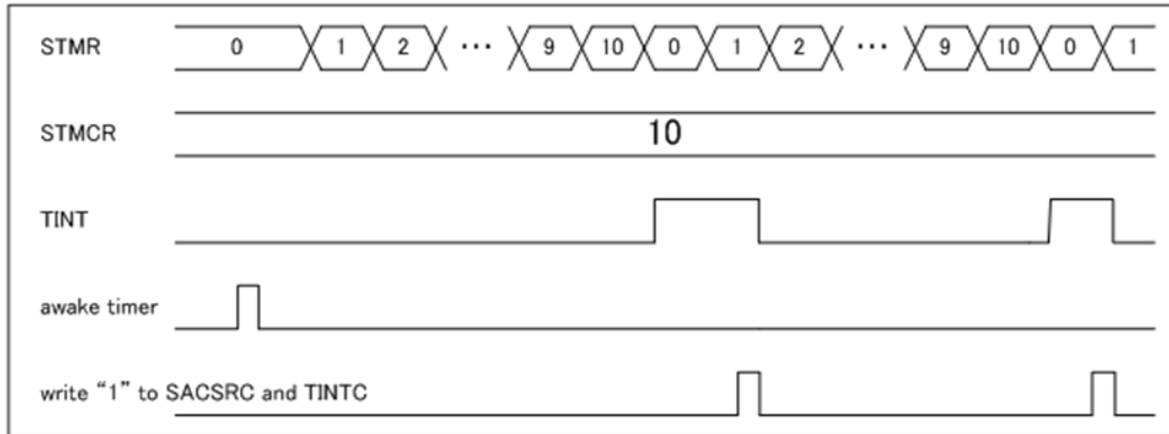


Figure 4-3 Flow Chart of Serial Timer Initialization

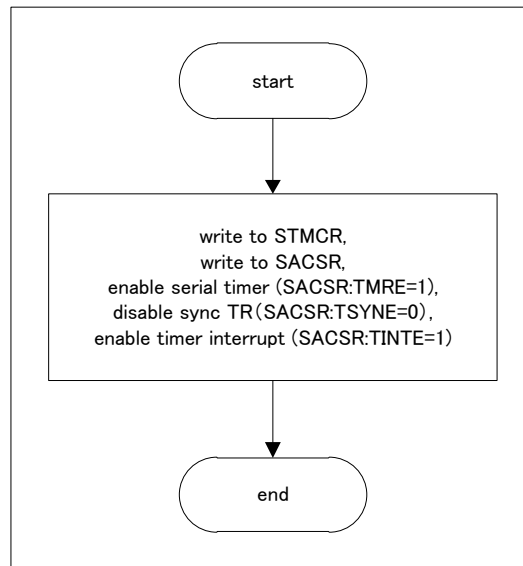
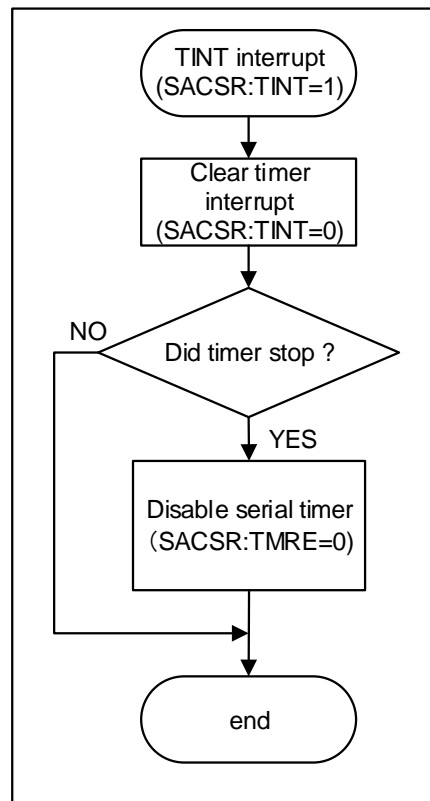


Figure 4-4 Flow Chart of Serial Timer Interrupt Processing



**Note:**

- In the state in which synchronous transmission is disabled (SACSR:TSYNE="0") and "0x0000" is set in the timer comparison register (STMCR), if the timer is operating and the division value (SACSR:TDIV3 to 0) of the timer operation clock is set to "0b0000", the timer interrupt flag (SACSR:TINT) is fixed to "1".

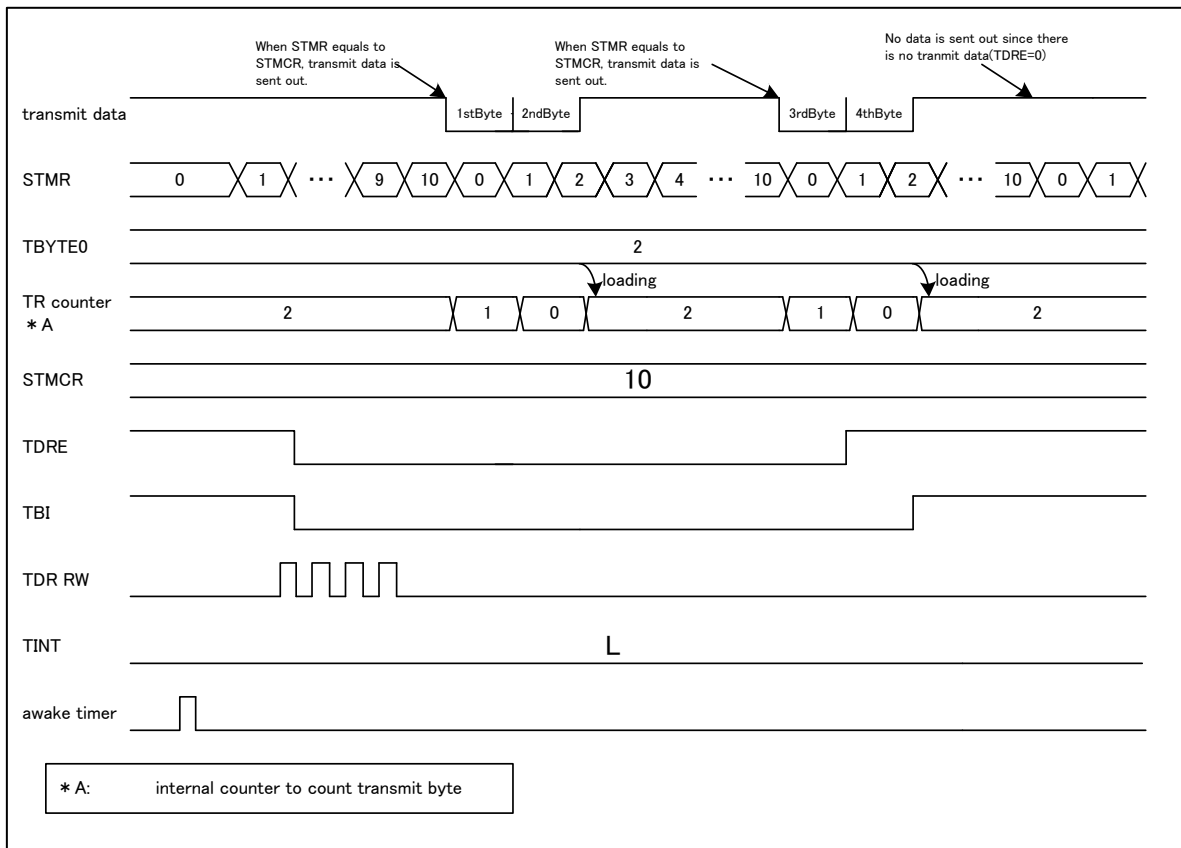
#### d) Transmission Operation Synchronized with the Timer

The serial timer is used for synchronous transmission when the synchronous transmission enable bit (SACSR:TSYNE) is "1".

Timer-synchronized transmission operates in the following manner:

1. If the transmission data register contains data (SSR:TDRE="0") and the serial timer register (STMR) and the serial timer comparison register (STMCR) coincide, transmission operation starts and the serial timer register (STMR) is reset to 0. Data transmission is performed for the data number specified in TBYTE0.
2. After transmitting the data number as specified by TBYTE0, transmission is suspended until the next time the serial timer register (STMR) and serial timer comparison register (STMCR) coincide.

Figure 4-5 Transmission Operation Synchronized with the Timer (SACSR:TSYNE=1, STMR=10, TBYTE0=2)



When synchronous transmission is enabled (SACSR:TSYNE="1") and the serial timer register (STMR) and the serial timer comparison register (STMCR) coincide, transmission is not activated under the following conditions:

- Transmission is disabled (SCR:TXE=0).
- The transmission data register does not contain valid data (SSR:TDRE=1).

However, writing transmission data to the transmission data register instantly starts transmission when all of the following are satisfied: 1. The transmission data register does not contain valid data (SSR:TDRE=1). 2. Synchronous transmission is enabled (SACSR:TSYNE="1"). 3. The serial timer register (STMR) and serial timer comparison register (STMCR) coincide.

If the transmission data register (TDR) contains valid data (SSR:TDRE=0) after transmission of the data number specified by TBYTE0, the data is not transmitted until the next time the serial timer register (STMR) and the serial timer comparison register (STMCR) coincide.

If, however, synchronous transmission is enabled (SACSR:TSYNE="1") and transmission operation is in progress (SSR:TBI=0), and the serial timer register (STMR) and the serial timer comparison register (STMCR) coincide, a transmission reservation is made. If transmission reservation has been made, transmission does not stop after number of transmission specified by TBYTE0, and the subsequent transmission is performed.

Transmission reservation is released under one of the conditions below.

- Programmable reset (SCR:UPCL=1)
- Transmission disabled (SCR:TXE=0)

**Figure 4-6 Flow Chart of Initialization of Transmission Synchronized with the Timer**

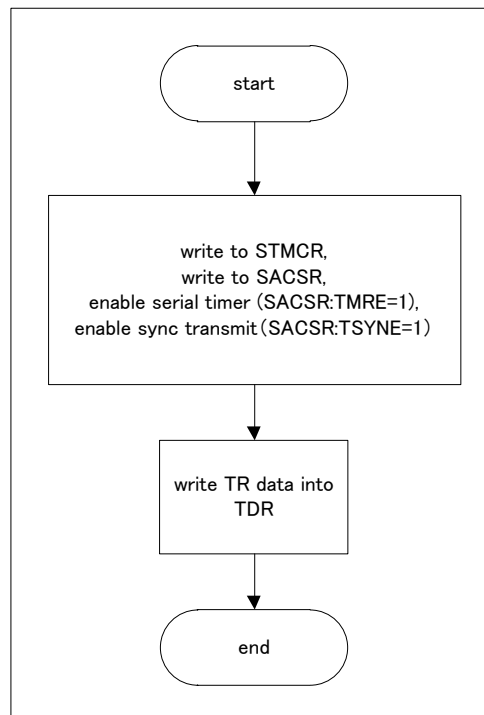
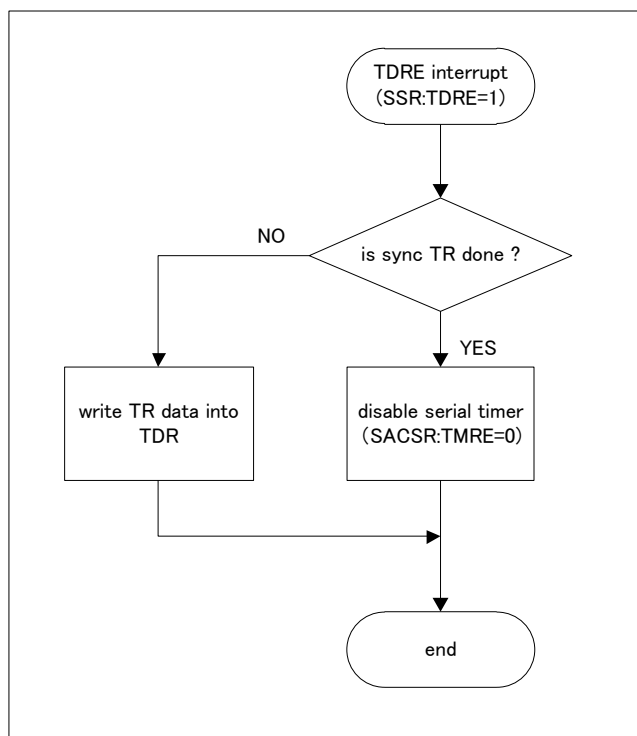




Figure 4-7 Flow Chart of Interrupt Processing in Transmission Synchronized with the Timer



**Note:**

- If the transmission data register (TDR) does not contain valid transmission data (SSR:TDRE=1) before transmitting a data frame of the size specified by TBYTE, transmission is suspended until the transmission data is written to the transmission data register (TDR). Writing transmission data to the transmission data register (TDR) restarts transmission.

## 5. Test Mode

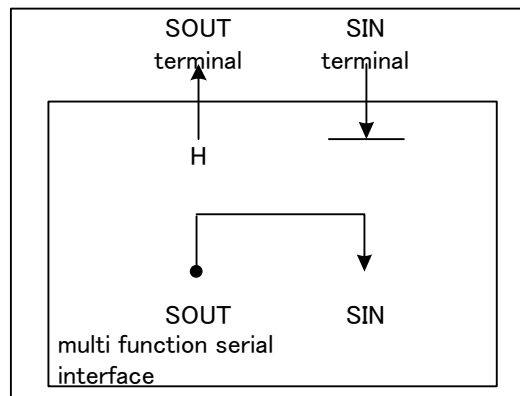
This section describes operation in a test mode.

### Serial Test Mode

When serial test mode is enabled (SACSR:STST=1), SOUT and SIN are connected inside the multifunction serial interface, so that the data transmitted from SOUT can be received from SIN directly.

When serial test mode is enabled (SACSR:STST=1), the SOUT pin is fixed to "H", and the data input to the SIN pin is ignored.

Figure 5-1 Serial Test Mode



**Note:**

- The value of the serial test mode enable bit (SACSR:STST) can be changed only when transmission and reception are prohibited (SCR:TXE=RXE=0).



## 6. Dedicated Baud Rate Generator

A transmission and reception clock source for the UART can be selected from the following.

- Dedicated baud rate generator (reload counter)
- Input of an external clock to the baud rate generator (reload counter)

### UART Baud Rate Selection

A baud rate can be selected from the following 2 types.

#### a) Baud Rate Obtained by Dividing the Internal Clock by the Dedicated Baud Rate Generator (Reload Counter)

There are 2 internal reload counters, each of which corresponds to the transmission/reception serial clock. A baud rate can be selected by setting a 15-bit reload value in baud rate generator register 1 or 0 (BGR1, BGR0).

The reload counters divide the internal clock according to the setting value.

Set the clock source by selecting an internal clock (BGR1:EXT=0).

#### b) Baud Rate Obtained by Dividing an External Clock by the Dedicated Baud Rate Generator (Reload Counter)

An external clock is used as the clock source of the reload counter.

A baud rate can be selected by setting a 15-bit reload value in baud rate generator register 1 or 0 (BGR1, BGR0).

The reload counters divide the external clock according to the setting value.

Set a clock source by selecting an external clock and using a baud rate generator clock (BGR1:EXT=1).

This mode is provided for dividing and using a special-frequency oscillator.

#### Notes:

- Set the external clock (BGR1:EXT=1) under the condition whereby the reload counter is stopped (BGR1/0="0x0000").
- When the external clock setting is specified (BGR1:EXT=1), the "H" width and "L" width of the external clock must be 2 bus clocks or more.

## 6.1. Setting the Baud Rate

This section describes baud rate setting. This section also describes the result of calculating the serial clock frequency.

### (1) Calculation of Baud Rate

The two of 15-bit reload counters are set by baud rate generator register 1, 0 (BGR1, BGR0).  
The baud rate calculation formulas are shown below.

#### a) Reload Value

$$V = \text{Phy} / b - 1$$

V : reload value    b : baud rate    Phy : frequency of bus clock or that of external clock

#### b) Calculation Example

When bus clock is 16MHz, using internal clock, baud rate is 19200 bps, the reload value can be calculated as follows.

reload value :

$$V = (16 \times 1000000) / 19200 - 1 = 832$$

then baud rate will be

$$b = (16 \times 1000000) / (832 + 1) = 19208\text{bps}$$

#### c) Baud Rate Error

The baud rate error is obtained using the following formula.

$$\text{error rate(\%)} = (\text{calculated value} - \text{target value}) / \text{target value} \times 100$$

(ex. When bus clock is 20MHz, target baud rate is 153600 bps,

the result will be as follows.

$$\text{reload value} = (20 \times 1000000) / 153600 - 1 = 129$$

$$\text{calculated baud rate} = 20 \times 1000000 / (129 + 1) = 153846 \text{ (bps)}$$

$$\text{error rate (\%)} = (153846 - 153600) / 153600 \times 100 = 0.16 \text{ (\%)}$$

#### Notes:

- Setting the reload value to "0" stops the reload counter.
- When the reload value is even, the "L" width of the reception serial clock is 1-bus-clock-cycle longer than the "H" width. When the reload value is odd, the "L" width and the "H" width of the serial clock are equal.
- Set the reload value to 4 or larger. However, data may not be received normally depending on baud rate errors and the reload value setting.
- When considering the tolerable baud rate range, also consider the impact of the jitter of the clock input to the macro.



## (2) Setting Example of Reload Values and Baud Rate Values for Individual Bus Clock Frequencies

The following table lists example of reload values and baud rate settings.

Table 6-1 Example of Reload Values and Baud Rate Settings

Baud Rate (bps)	8 MHz		10 MHz		16 MHz		20 MHz		24 MHz		32 MHz	
	Value	ERR	Value	ERR	Value	ERR	Value	ERR	Value	ERR	Value	ERR
4 M	-	-	-	-	-	-	4	0	5	0	7	0
2.5 M	-	-	-	-	-	-	7	0	-	-	-	-
2 M	-	0	4	0	7	0	9	0	11	0	15	0
1 M	7	0	9	0	15	0	19	0	23	0	31	0
500000	15	0	19	0	31	0	39	0	47	0	63	0
460800	-	-	-	-	-	-	-	-	51	0.16	-	-
250000	31	0	39	0	63	0	79	0	95	0	127	0
230400	-	-	-	-	-	-	86	-0.22	103	0.16	138	-0.08
153600	51	0.16	64	0.16	103	0.16	129	0.16	155	0.16	207	0.16
125000	63	0	79	0	127	0	159	0	191	0	255	0
115200	-	-	86	-0.22	138	-0.08	173	-0.22	207	0.16	277	-0.08
76800	103	0.16	129	0.16	207	0.16	259	0.16	312	-0.16	416	-0.08
57600	138	-0.08	173	-0.22	277	-0.08	346	0.06	416	-0.08	555	-0.08
38400	207	0.16	259	0.16	416	-0.08	520	-0.03	624	0	832	0.04
28800	277	-0.08	346	0.06	555	-0.08	693	0.06	832	0.04	1110	0.01
19200	416	-0.08	520	-0.03	832	0.04	1041	-0.03	1249	0	1666	-0.02
10417	767	<0.01	959	<0.01	1535	<0.01	1919	<0.01	2303	<0.01	3071	<0.01
9600	832	0.04	1041	-0.03	1666	-0.02	2082	0.02	2499	0	3332	0.01
7200	1110	0.01	1388	<0.01	2221	0.01	2777	<0.01	3332	0.01	4443	0.01
4800	1666	-0.02	2082	0.02	3332	0.01	4166	<0.01	4999	0	6666	<0.01
2400	3332	0.01	4166	<0.01	6666	<0.01	8332	<0.01	9999	0	13332	<0.01
1200	6666	<0.01	8332	<0.01	13332	<0.01	16666	<0.01	19999	0	26666	<0.01
600	13332	<0.01	16666	<0.01	26666	<0.01	-	-	-	-	-	-
300	26666	<0.01	-	-	-	-	-	-	-	-	-	-

- Value: Setting value of the BGR1/0 registers (decimal)
- ERR: Baud rate error (%)

**Table 6-2 Example of Reload Values and Baud Rate Settings (Continued)**

Baud Rate (bps)	40 MHz		48 MHz		72 MHz		80 MHz	
	Value	ERR	Value	ERR	Value	ERR	Value	ERR
4 M	9	0	11	0	17	0	19	0
2.5 M	15	0	-	-	-	-	31	0
2 M	19	2	23	0	35	0	39	0
1 M	39	0	47	0	71	0	79	0
500000	79	0	95	0	143	0	159	0
460800	86	-0.22	103	0.16	155	0.16	173	-0.22
250000	159	0	191	0	287	0	319	0
230400	173	-0.22	207	0.16	312	-0.16	346	0.06
153600	259	0.16	312	-0.16	468	-0.05	520	-0.03
125000	319	0	383	0	575	0	639	0
115200	346	0.06	416	-0.08	624	0	693	0.06
76800	520	-0.03	624	0	937	-0.05	1041	-0.03
57600	693	0.06	832	0.04	1249	0	1388	<0.01
38400	1041	-0.03	1249	0	1874	0	2082	0.02
28800	1388	<0.01	1666	-0.02	2499	0	2777	<0.01
19200	2082	0.02	2499	0	3749	0	4166	-0.01
10417	3839	<0.01	4607	<0.01	6911	<0.01	7679	0
9600	4166	<0.01	4999	0	7499	0	8332	0
7200	5555	<0.01	6666	<0.01	9999	0	11110	0
4800	8332	<0.01	9999	0	14999	0	16666	0
2400	16666	<0.01	19999	0	29999	0	-	-
1200	-	-	-	-	-	-	-	-
600	-	-	-	-	-	-	-	-
300	-	-	-	-	-	-	-	-

- Value: Setting value of the BGR1/0 registers (decimal)
- ERR: Baud rate error (%)

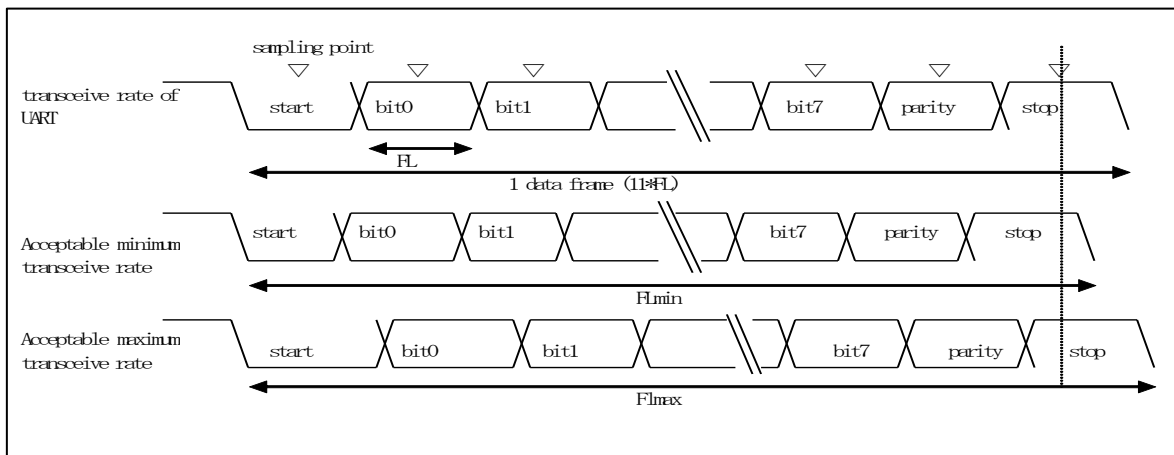


### (3) Reception Baud Rate Tolerance

The following describes the amount of error in the destination baud rate that is tolerable during reception.

Use the following formula to ensure that the baud rate errors during data reception fall within the tolerable range.

**Figure 6-1 Reception Baud Rate Tolerance**



After the detection of a start bit, the sampling timing of the reception data is determined by the counters set in registers BGR1/0, as shown in the figure. The data can be received normally provided all the data up to the last data (stop bit) is included within this sampling timing.

This theoretically gives the following for 11-bit data reception.

For a sampling timing margin of 1 bus clock cycle ( $\Phi$ ), the tolerable minimum transfer rate ( $FL_{min}$ ) will be as follows:

$$FL_{min} = (11 \text{ bits} * (V + 1) - (V + 1) / 2 + 2) / \Phi = (21V + 25) / 2 \Phi \text{ (s)} \quad V: \text{Reload value}, \Phi: \text{Bus clock}$$

Therefore, the maximum destination baud rate ( $BG_{max}$ ) receivable will be as follows:

$$BG_{max} = 11 / FL_{min} = 22 \Phi / (21V + 25) \text{ (bps)} \quad V: \text{Reload value}, \Phi: \text{Bus clock}$$

When receiving data at the tolerable maximum transfer rate ( $FL_{max}$ ), sampling is performed at the first point of the 11th-bit in the reception data.

Therefore, the tolerable maximum transfer rate ( $FL_{max}$ ) will be as follows:

$$10 / 11 * FL_{max} = (11 \text{ bits} * (V + 1) - (V + 1) / 2) / \Phi \quad V: \text{Reload value}, \Phi: \text{Bus clock}$$

$$FL_{max} = (21 / 20 * 11 * (V + 1)) / \Phi$$

For a sampling timing margin ( $\Phi$ ) of 2 clock cycles, the tolerable maximum transfer rate ( $FL_{max}$ ) will be as follows:

$$FL_{max} = (21 / 20 * 11 * (V + 1) - 2) / \Phi = (231V + 191) / 20 \Phi \text{ (s)} \quad V: \text{Reload value}, \Phi: \text{Bus clock}$$

Therefore, the minimum destination baud rate ( $BG_{min}$ ) receivable will be as follows:

$$BG_{min} = 11 / FL_{max} = 220 \Phi / (231V + 191) \text{ (bps)} \quad V: \text{Reload value}, \Phi: \text{Bus clock}$$

The tolerable baud rate errors for the UART and destination will be as follows, given the calculation formulas for the minimum and maximum baud rate values.

**Table 6-3 Tolerable Errors of Reload Values and Baud Rates**

Reload Value (V)	Tolerable Maximum Baud Rate Error	Tolerable Minimum Baud Rate Error
3	0%	0
10	+2.98%	-3.24%
50	+4.37%	-4.44%
100	+4.56%	-4.60%
200	+4.66%	-4.68%
32767	+4.76%	-4.76%

**Note:**

- The receiving accuracy depends on the number of bits in a frame, the bus clock, and the reload value. A higher accuracy is achieved with a higher bus clock and a higher division ratio.

**(4) External Clock**

Writing "1" to the EXT bit in the baud rate generator register (BGR1) causes the baud rate generator to divide the external clock.

**Note:**

- The external clock signal synchronizes with the internal clock of the UART. Therefore, the existence of an external clock that cannot be synchronized makes this operation unstable.

**(5) Reload Counter Function**

There is both a transmission reload counter and a reception reload counter, which function as dedicated baud rate generators. They consist of 15-bit registers for the reload values. They generate a transmission clock and a reception clock from an external clock or an internal clock.

**(6) Start of Counting**

When a reload value is written to the baud rate generator registers (BGR1 and BGR0), the reload counters start counting.

**(7) Restarting**

A reload counter restarts under the following conditions:

**a) For Both the Transmission and Reception Reload Counters**

- Programmable reset (SCR:UPCL bit)

**b) For the Reception Reload Counter**

- Detection of the falling edge of the start bit in the asynchronous mode



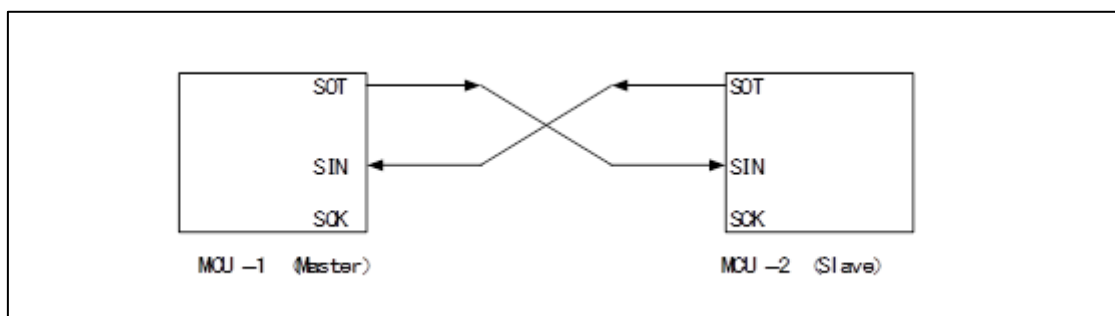
## 7. Setup Procedure and Program Flow for Operation Mode 0 (Asynchronous Normal Mode)

Operation mode 0 allows asynchronous serial bidirectional communication.

### (1) Connection between MCUs

Select bidirectional communication for operation mode 0 (normal mode). As shown in Figure 7-1, interconnect two MCUs.

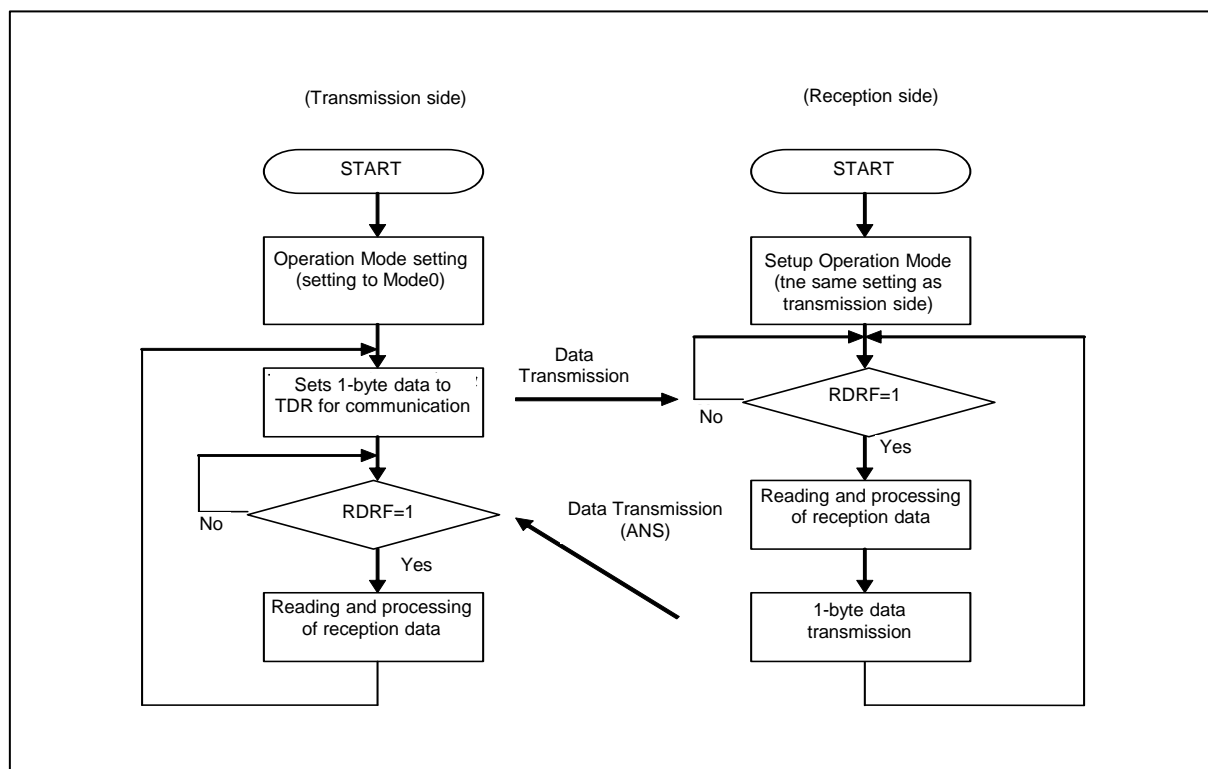
Figure 7-1 Example of Bidirectional Communication Connection for UART Operation Mode 0



### (2) Flow Chart

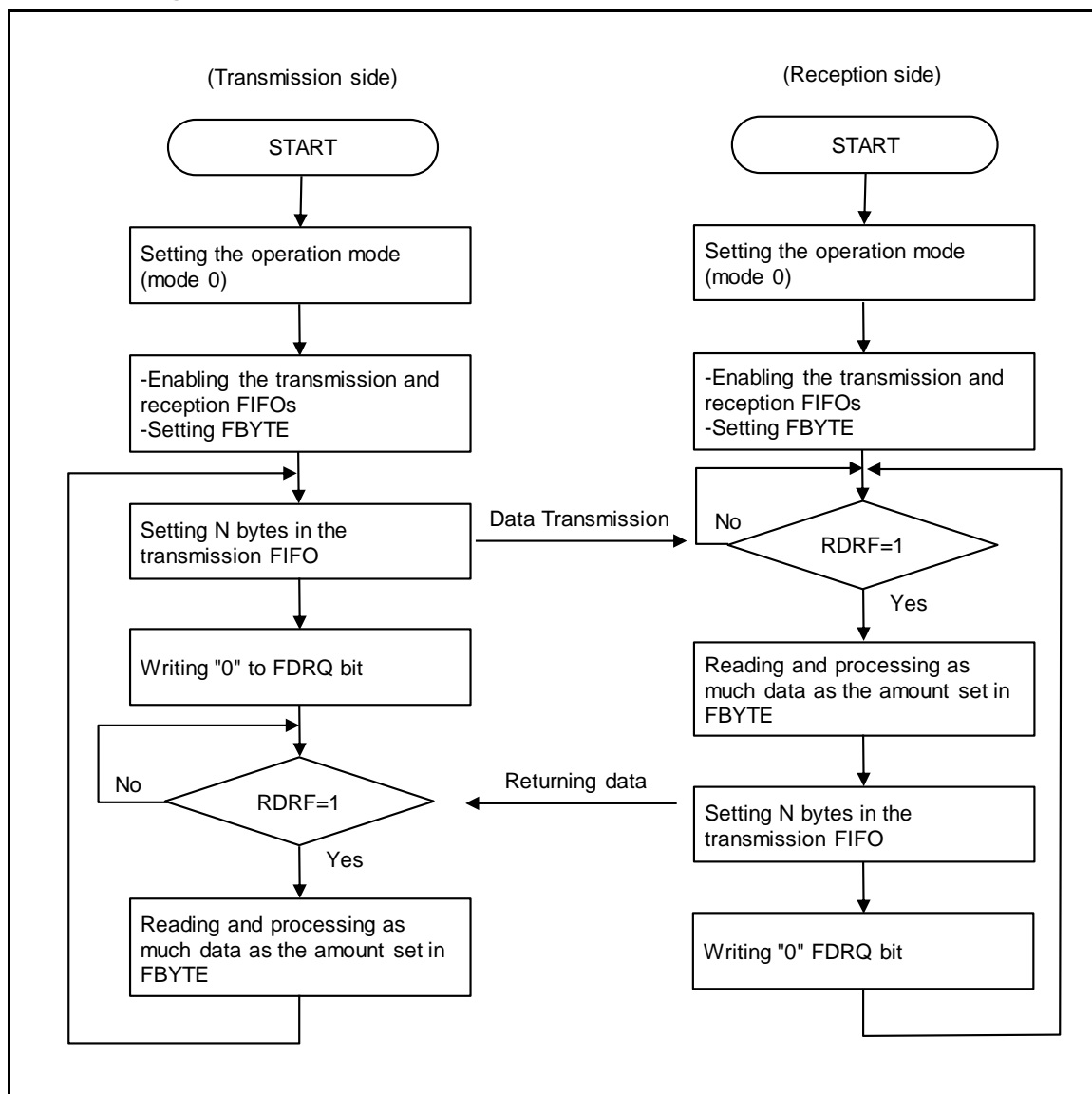
#### a) FIFO Not Used

Figure 7-2 Example of Bidirectional Communication Flow Chart (FIFO Not Used)



b) FIFO Used

Figure 7-3 Example of Bidirectional Communication Flow Chart (FIFO Used)



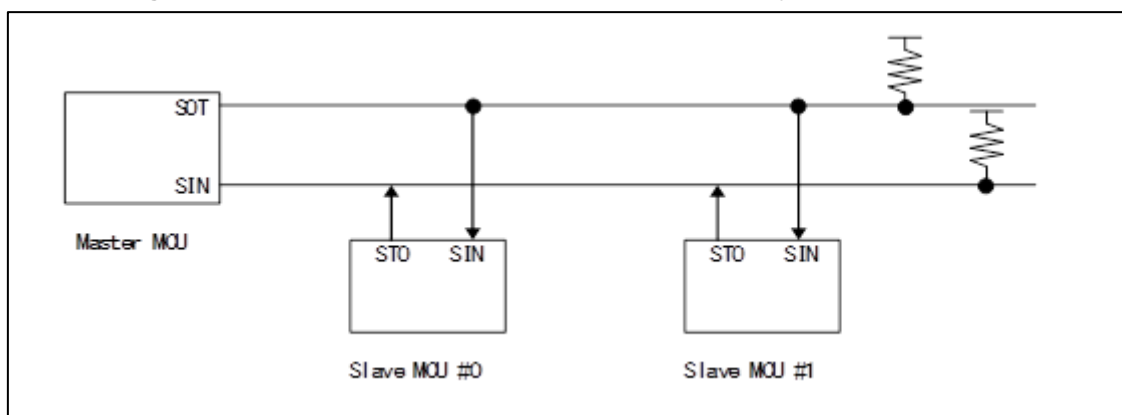
## 8. Setup Procedure and Program Flow for Operation Mode 1 (Asynchronous Multi-processor Mode)

Operation mode 1 (multi-processor mode) allows communication in master/slave connections of multiple MCUs. This mode can be used either as a master or slave.

### (1) Connection between MCUs

Master/slave-type communication enables a communication system that has 2 shared communication lines connecting 1 master MCU and multiple slave MCUs, as shown in the figure. The UART can be used either as a master or a slave.

Figure 8-11 Connection Example for UART Master/Slave-type Communication



### (2) Function Selection

Select an operation mode and a data transfer method for master/slave-type communication as shown in Table 8-1.

Table 8-1 Selection of Master/Slave-type Communication Function

	Operation Mode		Data	Parity	Stop Bit	Bit Direction
	Master MCU	Slave MCU				
Address Transmission/Reception	Mode 1 (A/D bit transmission)	Mode 1 (A/D bit reception)	AD = "1" + 7-bit or 8-bit address	None	1 bit or 2 bits	LSB or MSB first
Data Transmission/Reception			AD = "0" + 7-bit or 8-bit data			

**Note:**

- Access the transmission/reception data (TDR/RDR) in word units for operation mode 1.

### **Communication Procedure**

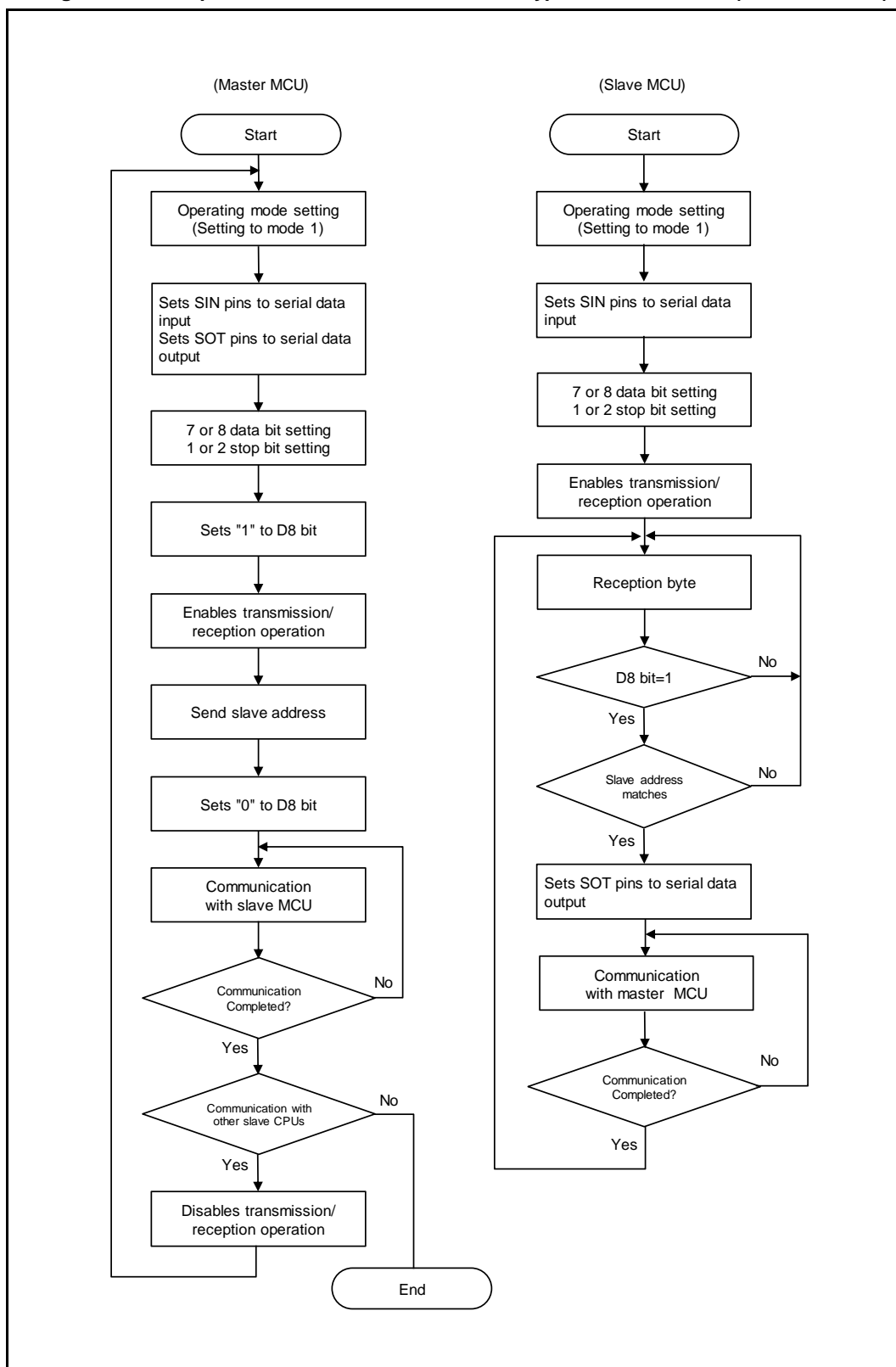
Communication starts when the master MCU transmits address data. Address data, for which the D8 bit is set to "1", selects the destination slave MCU. Each slave MCU checks the address data by its program, and if it coincides with the allocated address, the slave MCU communicates (by normal data) with the master MCU.

Figure 8-2 and Figure 8-3 provide flow charts of master/slave-type communication (multi-processor mode).

### (3) Flow Chart

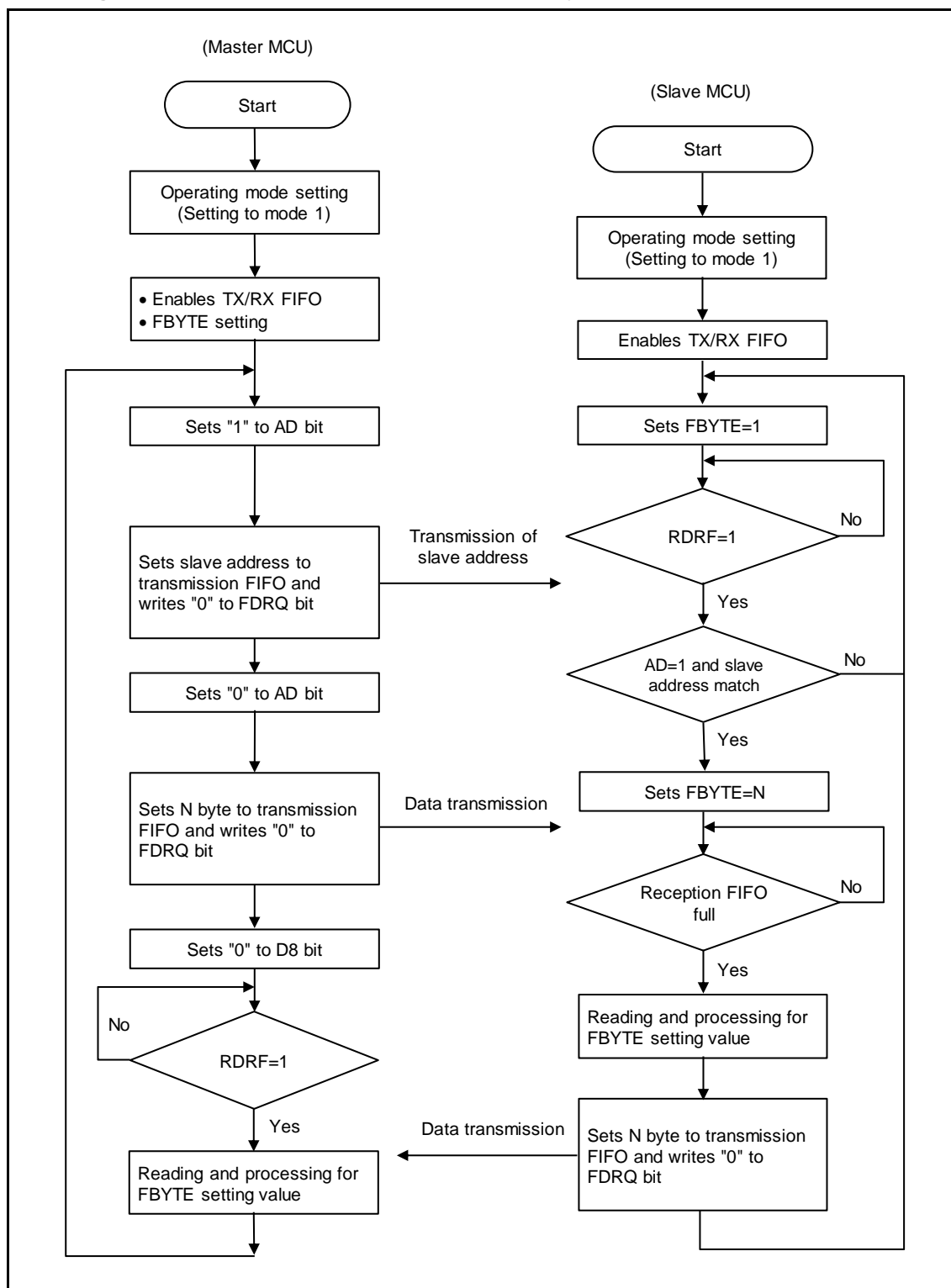
#### a) FIFO Not Used

Figure 8-2 Example of Flow Chart of Master/Slave-type Communication (FIFO Not Used)



b) FIFO Used

Figure 8-3 Example of Flow Chart of Master/Slave-type Communication (FIFO Used)





## 9. Registers

This section provides a list of the registers of the UART (Asynchronous Serial Interface).

All registers are prefixed by "MFSxx\_". xx is the channel number (00, 01, 02, 03 and 04).

**Table 9-1 List of UART (Asynchronous Serial Interface) Registers**

Abbreviated Register Name	Register Name	See
SCR	Serial Control Register	9.1
SMR	Serial Mode Register	9.2
SSR	Serial Status Register	9.3
ESCR	Extended Communication Control Register	9.4
RDR/TDR	Reception Data Register/Transmission Data Register	9.5
SACSR	Serial Auxiliary Control Status Register	9.6
STMR	Serial Timer Register	9.7
STMCR	Serial Timer Comparison Register	9.8
TBYTE0	Transfer Byte Register	9.9
BGR1/0	Baud Rate Generator Register 1/0	9.10
FCR1	FIFO Control Register 1	9.11
FCR0	FIFO Control Register 0	9.12
FBYTE	FIFO Byte Register	9.13
FTICR	Transmission FIFO Interrupt Control Register	9.14
SACSRC	Serial Auxiliary Control Status Clear Register	9.15
FCR1C	FIFO Control Clear Register 1	9.16
FCR0C	FIFO Control Clear Register 0	9.17
SACSRS	Serial Auxiliary Control Status Set Register	9.18
FCR1S	FIFO Control Set Register 1	9.19
FCR0S	FIFO Control Set Register 0	9.20

### Operation Mode

The UART (Asynchronous Serial Interface) can operate in either of 2 modes. The operation mode is determined by MD2, MD1, and MD0 in the serial mode register (SMR).

**Table 9-2 Operation Modes of UART (Asynchronous Serial Interface)**

Operation Mode	MD2	MD1	MD0	Type
0	0	0	0	UART0 (asynchronous normal mode)
1	0	0	1	UART1 (asynchronous multi-processor mode)





## 9.1. Serial Control Register (SCR)

The serial control register (SCR) can be used for permission/prohibition of transmission and reception, permission/prohibition of transmission and reception interrupts, permission/prohibition of transmission bus idle interrupts, and UART reset.

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	UPCL	Reserved		RIE	TIE	TBIE	RXE	TXE
ACCESS_TYPE	R0,W	R0,W0		R/W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	00		0	0	0	0	0

\* Lower byte [bit7:0] of this register is Serial Mode Register (SMR)

### [bit15] UPCL: Programmable clear bit

This bit initializes the UART's internal status.

When "1" is set:

- UART is directly reset (software reset). However, the register settings are retained. At this time, transmission and reception communications are immediately disconnected.
- The baud rate generator reloads the setting values for the BGR1/0 registers and restarts.
- All transmission and reception communications and status interrupt factors (SSR:PE, FRE, ORE, RDRF, TDRE, TBI, SACSR:TINT) are initialized to "0b0000110".

When "0" is set:

There is no effect.

Value	Description	
	Read	Write
0	"0" is always read	No effect
1		Programmable clear

### Notes:

- Execute programmable clear immediately after setting interrupt prohibition.
- When FIFO is used, prohibit FIFO (FCR0:FE2, FE1 = 0) before executing programmable clear.
- Executing programmable clear (SCR:UPCL = 1) does not clear the value of the serial timer register (STMR).

### [bit14:13] Reserved: Reserved bits

**[bit12] RIE: Reception interrupt enable bit**

- This bit enables/disables the output of reception interrupt request to the CPU.
- When the RIE bit and reception data flag bit (SSR:RDRF) are set to "1", or one of the error flag bits (SSR:PE, ORE, FRE) is set to "1", a reception interrupt request is output.

Value	Description
0	Disable reception interrupts
1	Enable reception interrupts

**[bit11] TIE: Transmission interrupt enable bit**

- This bit enables/disables the output of transmission interrupt request to the CPU.
- When the TIE bit and the SSR:TDRE bit are set to "1", a transmission interrupt request is output.

Value	Description
0	Disable transmission interrupts
1	Enable transmission interrupts

**[bit10] TBIE: Transmission bus idle interrupt enable bit**

- This bit enables/disables the output of transmission bus idle interrupt request to the CPU.
- A transmission bus idle interrupt request is issued when the TBIE bit and the TBI bit are "1".

Value	Description
0	Disable transmission bus idle interrupts
1	Enable transmission bus idle interrupts

**[bit9] RXE: Reception operation enable bit**

This bit enables/disables UART reception operation.

- "0": Disable reception operation.
- "1": Enable reception operation.

Value	Description
0	Disable reception
1	Enable reception

**Notes:**

- Even if reception operation is enabled ( $RXE=1$ ), reception does not start unless there is a falling edge of a start bit (for NRZ format ( $ESCR:INV=0$ )). (For inverted NRZ format ( $ESCR:INV=1$ ), reception does not start unless there is a rising edge.)
- If reception operation is disabled ( $RXE=0$ ) during reception, the operation is stopped immediately.

**[bit8] TXE: Transmission operation enable bit**

This bit enables/disables UART transmission operation.

- "0": Disable transmission operation.
- "1": Enable transmission operation.

Value	Description
0	Disable transmission
1	Enable transmission

**Note:**

- *If transmission operation is disabled (TXE = 0) during transmission, the operation is stopped immediately.*

## 9.2. Serial Mode Register (SMR)

The serial mode register (SMR) is responsible for the following: 1. Operation mode setting, 2. Selection of the transfer direction, the data length, and the stop bit length, and 3. Settings of enabling/disabling of output on the serial data and clock pins.

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	MD2	MD1	MD0	Reserved	SBL	BDS	Reserved	SOE
ACCESS_TYPE	R/W	R/W	R/W	R/W0	R/W	R/W	R0,W0	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

### [bit7:5] MD2 to 0: Operation mode setting bits

These bits set the operation mode of the Asynchronous Serial Interface.

"0b000": Set operation mode 0 (asynchronous normal mode).

"0b001": Set operation mode 1 (asynchronous multi-processor mode).

"0b010": Set operation mode 2 (clock synchronous mode).

"0b011": Set operation mode 3 (LIN communication mode).

Value			Description
MD2	MD1	MD0	
0	0	0	Operation mode 0 (asynchronous normal mode)
0	0	1	Operation mode 1 (asynchronous multi-processor mode)
0	1	0	Operation mode 2 (clock synchronous mode)
0	1	1	Operation mode 3 (LIN communication mode)
Other than above			Setting is prohibited

\* For operation mode 2, see "CHAPTER: CSIO". For operation mode 3, see "CHAPTER: LIN Interface".

### Notes:

- To switch the operation mode, execute programmable clear (SCR:UPCL = 1) and then switch the operation mode.
- Set the registers after setting the operation mode.

### [bit4] Reserved: Reserved bit

### [bit3] SBL: Stop bit length selection bit

This bit sets the length of the stop bits (the frame end mark of the transmission data).

Settings for which SBL is "0" and ESCR:ESBL is "0": The stop bit length is set to 1 bit.

Settings for which SBL is "1" and ESCR:ESBL is "0": The stop bit length is set to 2 bits.

Settings for which SBL is "0" and ESCR:ESBL is "1": The stop bit length is set to 3 bits.

Settings for which SBL is "1" and ESCR:ESBL is "1": The stop bit length is set to 4 bits.

Value	Description	
0	ESCR:ESBL = 0	1 bit
	ESCR:ESBL = 1	3 bits
1	ESCR:ESBL = 0	2 bits
	ESCR:ESBL = 1	4 bits

#### Notes:

- Reception always detects only the first stop bit.
- Set this bit when transmission is disabled (SCR:TXE=0).

### [bit2] BDS: Transfer direction selection bit

This bit selects between whether the lowest bit is transferred first (LSB first, BDS = 0) or the highest bit is transferred first (MSB first, BDS = 1), for transfer serial data.

Value	Description
0	LSB first (lowest bit transferred first)
1	MSB first (highest bit transferred first)

#### Note:

- Set this bit when transmission and reception are disabled (SCR:TXE=SCR:RXE=0).

### [bit1] Reserved: Reserved bit

### [bit0] SOE: Serial data output enable bit

This bit enables/disables the output of serial data.

"0": Set the SOUT pin as a general-purpose I/O port.

"1": Set the SOUT pin as the serial data output pin (SOUT).

Value	Description
0	General-purpose I/O port
1	Serial data output pin

#### Note:

- The SOT pin must also be set as the resource output pin. For how it is set, see "CHAPTER: I/O Port".

### 9.3. Serial Status Register (SSR)

The serial status register (SSR) checks the transmission and reception status, checks the reception error flag, and clears the reception error flag.

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	REC	Reserved	PE	FRE	ORE	RDRF	TDRE	TBI
ACCESS_TYPE	R0,W	R0,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	1	1

\* Lower byte [bit7:0] of this register is Extended Communication Control Register (ESCR)

#### [bit15] REC: Reception error flag clear bit

This bit clears the PE, FRE, and ORE flags in the serial status register (SSR).

- Writing "1" clears the error flags.
- Writing "0" does not have any effect.

Value	Description	
	Read	Write
0	"0" is always read	No effect
1		Clear the reception error flags (PE, FRE, and ORE)

#### [bit14] Reserved: Reserved bit

#### [bit13] PE: Parity error flag bit (effective only in operation mode 0)

- This bit is set to "1" when a parity error occurs in data reception with ESCR:PEN = 1, and is cleared when writing "1" to the REC bit in the serial status register (SSR).
- A reception interrupt request is issued when the PE bit and the SCR:RIE bit are both "1".
- If this flag is set, the reception data register (RDR) data will be invalid.
- If this flag is set when the reception FIFO is used, the reception FIFO enable bits will be cleared, and no reception data will be stored in the reception FIFO.

Value	Description
0	No parity error
1	Parity error found

#### [bit12] FRE: Framing error flag bit

- This bit is set to "1" when a framing error occurs in data reception, and is cleared when writing "1" to the REC bit in the serial status register (SSR).
- A reception interrupt request is issued when the FRE bit and the SCR:RIE bit are both "1".
- If this flag is set, the reception data register (RDR) data will be invalid.
- If this flag is set when the reception FIFO is used, the reception FIFO enable bits will be cleared, and no reception data will be stored in the reception FIFO.

Value	Description
0	No framing error
1	Framing error found

**[bit11] ORE: Overrun error flag bit**

- This bit is set to "1" when an overrun error occurs in data reception, and is cleared when writing "1" to the REC bit in the serial status register (SSR).
- A reception interrupt request is issued when the ORE bit and the SCR:RIE bit are both "1".
- If this flag is set, the reception data register (RDR) data will be invalid.
- If this flag is set when the reception FIFO is used, the reception FIFO enable bits will be cleared, and no reception data will be stored in the reception FIFO.

Value	Description
0	No overrun error
1	Overrun error found

**[bit10] RDRF: Reception data full flag bit**

- This flag indicates the status of the reception data register (RDR).
- This flag is set to "1" when reception data is loaded to RDR, and is cleared to "0" when the data in the reception data register (RDR) is read.
- A reception interrupt request is issued when the RDRF bit and the SCR:RIE bit are both "1".
- When the reception FIFO is used, if the reception FIFO receives a prescribed data number, RDRF is set to "1".
- When the reception FIFO is used, if both of the conditions below are satisfied, and the reception idle state continues for at least 8 clock pulses of the baud rate clock, RDRF is set to "1".
  - The reception FIFO idle detection enable bit (FCR1:FRIIE) is "1".
  - The reception FIFO has not received a prescribed data number, and data remains in the reception FIFO.
- During an 8-clock count, the counter is reset to 0 when RDR is read, and the system starts counting the 8 clocks again.
- When the reception FIFO is used, this bit is cleared to "0" when the reception FIFO becomes empty by reading data from it.

Value	Description
0	The reception data register RDR is empty
1	The reception data register RDR contains data

**Note:**

- When the reception FIFO is used and RDRF has become "1", resetting the reception FIFO (FCR0:FCL2, FCL1 = "1") does not result in RDRF being set to "0". Therefore, in order to set RDRF to "0" after resetting the reception FIFO, perform a dummy read of the reception data register during the reception disable status (SCR:RXE = "0").

**[bit9] TDRE: Transmission data empty flag bit**

- This flag indicates the status of the transmission data register (TDR).
- Writing transmission data to TDR sets this bit to "0", indicating that TDR contains valid data. Loading the data to the transmission shift register to start transmission sets this bit to "1", indicating that TDR does not contain valid data.
- A transmission interrupt request is issued when the TDRE bit and the SCR:TIE bit are both "1".
- The TDRE bit is set to "1" when the UPCL bit in the serial control register (SCR) is set to "1".
- For set/reset timing of the TDRE bit when the transmission FIFO is used, see Section "2.4. Occurrence of Interrupts and Flag Set Timing When the Transmission FIFO is Used".

Value	Description
0	The transmission data register TDR contains data
1	The transmission data register is empty

**[bit8] TBI: Transmission bus idle flag bit**

- This bit indicates that the UART is not performing transmission.
- This bit is set to "0" when transmission data is written to the transmission data register (TDR).
- This bit is set to "1" during periods in which the transmission data register is empty (TDRE = 1) and transmission is not being performed.
- The TBI bit is set to "1" when the UPCL bit in the serial control register (SCR) is set to "1".
- A transmission interrupt request is issued if this bit is set to "1" and a transmission bus idle interrupt is enabled (SCR:TBIE = 1).

Value	Description
0	Transmission in progress
1	No transmission operation





## 9.4. Extended Communication Control Register (ESCR)

The extended communication control register (ESCR) sets the transmission and reception data length, enables/disables the parity bits, selects a parity bit, inverts the serial data format, and selects a stop bit length.

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved	ESBL	INV	PEN	P	L2	L1	L0
ACCESS_TYPE	R/W0	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

**[bit7] Reserved: Reserved bit**

### [bit6] ESBL: Extended stop bit length selection bit

This bit sets the length of the stop bits (the frame end mark of the transmission data).

Settings for which SBL is "0" and ESCR:ESBL is "0": The stop bit length is set to 1 bit.

Settings for which SBL is "1" and ESCR:ESBL is "0": The stop bit length is set to 2 bits.

Settings for which SBL is "0" and ESCR:ESBL is "1": The stop bit length is set to 3 bits.

Settings for which SBL is "1" and ESCR:ESBL is "1": The stop bit length is set to 4 bits.

Value	Description	
0	SMR:SBL = 0	1 bit
	SMR:SBL = 1	2 bits
1	SMR:SBL = 0	3 bits
	SMR:SBL = 1	4 bits

#### Notes:

- Reception always detects only the first stop bit.
- Set this bit when transmission is disabled (SCR:TXE = 0).

### [bit5] INV: Inverted serial data format bit

This bit selects the NRZ format or the inverted NRZ format as the serial data format.

Value	Description
0	NRZ format
1	Inverted NRZ format

**[bit4] PEN: Parity enable bit (effective only in operation mode 0)**

This bit specifies whether to append (in transmission) and detect (in reception) parity bits.

- "0": Parity bits are not appended.
- "1": Parity bits are appended.

Value	Description
0	Disable parity
1	Enable parity

**Note:**

- This bit is internally fixed to "0" in operation mode 1.

**[bit3] P: Parity selection bit (effective only in operation mode 0)**

This bit selects the odd parity "1" or the even parity "0" when parity check is enabled (ESCR: PEN = 1).

- "0": Set even parity.
- "1": Set odd parity.

Value	Description
0	Even parity
1	Odd parity

**[bit2:0] L2 to 0: Data length selection bits**

These bits specify the data length of the transmission/reception data.

- "0b000": The data length is 8 bits.
- "0b001": The data length is 5 bits.
- "0b010": The data length is 6 bits.
- "0b011": The data length is 7 bits.
- "0b100": The data length is 9 bits.

Value			Description
L2	L1	L0	
0	0	0	8-bit length
0	0	1	5-bit length
0	1	0	6-bit length
0	1	1	7-bit length
1	0	0	9-bit length
Other than above			Setting is prohibited

**Note:**

- Set the data length to 7 or 8 bits for operation mode 1. Other values are prohibited.



## 9.5. Reception Data Register/Transmission Data Register (RDR/TDR)

The reception data and transmission data registers are placed at the same address. If read, the register functions as a reception data register, and if written to, it functions as a transmission data register.

When FIFO operation is enabled, the RDR/TDR address is the FIFO read/write address.

### (1) Reception Data Register (RDR)

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved							D8
ACCESS_TYPE	R0							R
PROT_TYPE	-							
INITIAL_VALUE	0000000							0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	D7	D6	D5	D4	D3	D2	D1	D0
ACCESS_TYPE	R	R	R	R	R	R	R	R
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

### [bit8:0] D8 to 0: Reception data bits

The reception data register (RDR) is a 9-bit data buffer register for serial data reception.

- Serial data signals transmitted to the serial input pin (SIN pin) are converted by the shift register, and are stored in the reception data register (RDR).
- Higher bits are padded with "0" depending on the data length, as follows.

Data Length	D8	D7	D6	D5	D4	D3	D2	D1	D0
9 bits	X	X	X	X	X	X	X	X	X
8 bits	0	X	X	X	X	X	X	X	X
7 bits	0	0	X	X	X	X	X	X	X
6 bits	0	0	0	X	X	X	X	X	X
5 bits	0	0	0	0	X	X	X	X	X

(X represents a reception data bit.)

- The reception data full flag bit (SSR:RDRF) is set to "1" when reception data is stored in the reception data register (RDR). A reception interrupt request occurs if the reception interrupt is enabled (SCR:RIE = 1).
- Read the reception data register (RDR) when the reception data full flag bit (SSR:RDRF) is "1". The reception data full flag bit (SSR:RDRF) is automatically cleared to "0" when reading data from the reception data register (RDR).
- If a reception error occurs (when SSR:PE, ORE, or FRE becomes "1"), the data in the reception data register (RDR) will be invalid.
- In operation mode 1 (multi-processor mode), the data length is 7 or 8 bits, and the received AD bit is stored in the D8 bit.
- In the case of 9-bit transfer or in operation mode 1, RDR is read in 16-bit access.

**Notes:**

- *When the reception FIFO is used, SSR:RDRF is set to "1" when a predefined amount of data is received by the reception FIFO.*
- *When the reception FIFO is used, SSR:RDRF is cleared to "0" when the reception FIFO becomes empty.*
- *When a reception error occurs (when SSR:PE, ORE, or FRE becomes "1") during use of the reception FIFO, the enable bit in the reception FIFO is cleared and the reception data is not stored in the reception FIFO.*

**(2) Transmission Data Register (TDR)**

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved							D8
ACCESS_TYPE	WX							W
PROT_TYPE	-							
INITIAL_VALUE	1111111							1

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	D7	D6	D5	D4	D3	D2	D1	D0
ACCESS_TYPE	W	W	W	W	W	W	W	W
PROT_TYPE	-							
INITIAL_VALUE	1	1	1	1	1	1	1	1

**[bit8:0] D8 to 0: Transmission data bits**

The transmission data register (TDR) is a 9-bit data buffer register for serial data transmission.

- When data to be transmitted is written to the transmission data register (TDR) while transmission is enabled (SCR:TXE = 1), the transmission data is transferred to the transmission shift register. The transmission data is converted to serial data and sent out from the serial data output pin (SOUT pin).
- Bits are invalidated in descending order depending on the data length, as follows.

Data Length	D8	D7	D6	D5	D4	D3	D2	D1	D0
9 bits	X	X	X	X	X	X	X	X	X
8 bits	Invalid	X	X	X	X	X	X	X	X
7 bits	Invalid	Invalid	X	X	X	X	X	X	X
6 bits	Invalid	Invalid	Invalid	X	X	X	X	X	X
5 bits	Invalid	Invalid	Invalid	Invalid	X	X	X	X	X

- The transmission data empty flag (SSR:TDRE) is cleared to "0" when transmission data is written to the transmission data register (TDR).
- If the transmission FIFO is disabled or empty, the transmission data empty flag (SSR:TDRE) is set to "1" when transmission data is transferred to the transmission shift register to start transmission.
- When the transmission data empty flag (SSR:TDRE) is "1", transmission data can be written. A transmission interrupt occurs if the transmission interrupt is enabled. Write transmission data after the occurrence of a transmission interrupt or when the transmission data empty flag (SSR:TDRE) is "1".
- When the transmission data empty flag (SSR:TDRE) is "0" and the transmission FIFO is disabled or full, transmission data cannot be written.
- In operation mode 1 (multi-processor mode), the data length is 7 or 8 bits, and the AD bit is transmitted by writing it to the D8 bit.
- In the case of 9-bit transfer or in operation mode 1, write data to TDR by 16-bit access.

**Notes:**

- *The transmission data register is a write-only register, and the reception data register is a read-only register. Because the transmission and reception registers are allocated at the same address, the write value and the read value are different.*
- *For the set timing for the transmission data empty flag (SSR:TDRE) when the transmission FIFO is used, see Section "2.4. Occurrence of Interrupts and Flag Set Timing When the Transmission FIFO is Used".*



## 9.6. Serial Auxiliary Control Status Register (SACSR)

The serial auxiliary control status register (SACSR) performs the following: 1. Controlling serial test operation, 2. selecting a serial timer activation method, 3. enabling/disabling timer interrupts, 4. enabling/disabling synchronous transmission, 5. setting division value of the serial timer operation clock, and 6. enabling/disabling the serial timer.

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	STST	Reserved				Reserved		TINT
ACCESS_TYPE	R/W	R0,W0				R/W0		R,W
PROT_TYPE	-							
INITIAL_VALUE	0	0000				00		0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	TINTE	TSYNE	Reserved	TDIV3	TDIV2	TDIV1	TDIV0	TMRE
ACCESS_TYPE	R/W	R/W	R/W0	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

### [bit15] STST: Serial test bit

This bit enables or disables serial test mode.

When serial test mode is enabled, SOUT and SIN are connected inside the multifunction serial interface, so that the data transmitted from SOUT can be received from SIN directly.

When serial test mode is enabled, the SOUT pin is fixed to "H", and any data input to the SIN pin is ignored.

- This bit is set to "0" when the SACSRC:STSTC bit in the clear register is set to "1".
- This bit is set to "1" when the SACSRS:STSTS bit in the set register is set to "1".

Value	Description
0	Disable the serial test mode
1	Enable the serial test mode

### Note:

- This bit can be changed only when transmission/reception are disabled ( $SCR:TXE = 0$ ,  $SCR:RXE = 0$ ).

### [bit14:9] Reserved: Reserved bits

**[bit8] TINT: Timer interrupt flag**

If the serial timer register (STMR) and the serial timer comparison register (STMCR) coincide, the serial timer register (STMR) is set to "0", and this bit is set to "1".

When this bit is "1" and the timer interrupt enable bit (TINTE) is "1", a status interrupt request is issued.

Writing "0" to this bit resets this bit to "0".

Writing "1" to this bit is invalid.

This bit is set to "0" when the SACSRC:TINTC bit in the clear register is set to "1".

Value	Description
0	No timer interrupt request
1	Timer interrupt request issued

**Notes:**

- Software reset (SCR:UPCL="1") resets this bit to "0".
- This bit is not set to "1" when the synchronous transmission enable bit (TSYNE) is "1".

**[bit7] TINTE: Timer interrupt enable bit**

This bit enables/disables timer interrupts to the CPU.

When this bit is "1" and the timer interrupt flag (TINT) is "1", a status interrupt request is issued.

This bit is set to "0" when the SACSRC:TINTEC bit in the clear register is set to "1".

This bit is set to "1" when the SACSRS:TINTES bit in the set register is set to "1".

Value	Description
0	Disable interrupts triggered by the serial timer
1	Enable interrupts triggered by the serial timer

**[bit6] TSYNE: Synchronous transmission enable bit**

This bit enables or disables synchronous transmission.

Transmission is activated if this bit is "1".

- The serial timer register (STMR) and the serial timer comparison register (STMCR) coincide at the time of timer-synchronized transmission.
- This bit is set to "0" when the SACSRC:TSYNEC bit in the clear register is set to "1".
- This bit is set to "1" when the SACSRS:TSYNES bit in the set register is set to "1".

Value	Description
0	Disable synchronous transmission. The serial timer is used as a timer
1	Enable synchronous transmission. The serial timer is not used as a timer

**Notes:**

- This bit can be changed only when the serial timer enable bit (TMRE) is "0".
- When synchronous transmission is enabled (TSYNE=1), if transmission is disabled (SCR:TXE = 0), transmission is not activated in the following case.
- The serial timer register (STMR) and the serial timer comparison register (STMCR) coincide at the time of timer-synchronized transmission.





**[bit5] Reserved: Reserved bit**

**[bit4:1] TDIV3 to 0: Timer operation clock division bits**

These bits set the division ratio of the serial timer.

Value				Description						
TDIV3	TDIV2	TDIV1	TDIV0	Division Ratio	$\Phi =$ 8 MHz	$\Phi =$ 10 MHz	$\Phi =$ 16 MHz	$\Phi =$ 20 MHz	$\Phi =$ 24 MHz	$\Phi =$ 32 MHz
0	0	0	0	$\Phi$	125 ns	100 ns	62.5 ns	50 ns	41.67 ns	31.25 ns
0	0	0	1	$\Phi/2$	250 ns	200 ns	125 ns	100 ns	83.33 ns	62.5 ns
0	0	1	0	$\Phi/4$	500 ns	400 ns	250 ns	200 ns	166.67 ns	125 ns
0	0	1	1	$\Phi/8$	1 $\mu$ s	800 ns	500 ns	400 ns	333.33 ns	250 ns
0	1	0	0	$\Phi/16$	2 $\mu$ s	1.6 $\mu$ s	1 $\mu$ s	800 ns	666.67 ns	500 ns
0	1	0	1	$\Phi/32$	4 $\mu$ s	3.2 $\mu$ s	2 $\mu$ s	1.6 $\mu$ s	1.33 $\mu$ s	1 $\mu$ s
0	1	1	0	$\Phi/64$	8 $\mu$ s	6.4 $\mu$ s	4 $\mu$ s	3.2 $\mu$ s	2.67 $\mu$ s	2 $\mu$ s
0	1	1	1	$\Phi/128$	16 $\mu$ s	12.8 $\mu$ s	8 $\mu$ s	6.4 $\mu$ s	5.33 $\mu$ s	4 $\mu$ s
1	0	0	0	$\Phi/256$	32 $\mu$ s	25.6 $\mu$ s	16 $\mu$ s	12.8 $\mu$ s	10.67 $\mu$ s	8 $\mu$ s

$\Phi$ : Bus clock

**Notes:**

- These bits can be changed only when the serial timer enable bit (TMRE) is "0".
- Settings other than the above are prohibited.

**[bit0] TMRE: Serial timer enable bit**

This bit enables or disables the serial timer operation.

- This bit is set to "0" when the SACSRC:TMREC bit in the clear register is set to "1".
- This bit is set to "1" when the SACSRS:TMRES bit in the set register is set to "1".

Value	Description
0	Stop the serial timer operation. When stopped, the value of the serial timer register (STMR) is retained.
1	Changing this bit from "0" to "1" initializes the serial timer register (STMR) to "0" and starts the operation of the serial timer.

**Note:**

- For serial timer synchronous transmission, change this bit from "0" to "1" if transmission is disabled.

## 9.7. Serial Timer Register (STMR)

The serial timer register (STMR) represents the timer value of the serial timer.

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	TM15	TM14	TM13	TM12	TM11	TM10	TM9	TM8
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	TM7	TM6	TM5	TM4	TM3	TM2	TM1	TM0
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

### [bit15:0] TM15 to 0: Timer data bits

These bits indicate the timer value of the serial timer.

During timer operation, the timer value of the serial timer is incremented by 1 for each timer operation clock (specified by SACSR:TDIV3 to 0).

#### **Note:**

- These bits are initialized to "0" when the timer operation starts.



## 9.8. Serial Timer Comparison Register (STMCR)

The serial timer comparison register (STMCR) sets the timer comparison value for the serial timer.

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	TC15	TC14	TC13	TC12	TC11	TC10	TC9	TC8
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	TC7	TC6	TC5	TC4	TC3	TC2	TC1	TC0
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

### [bit15:0] TC15 to 0: Compare bits

These bits set a comparison value for the serial timer.

These bits are compared with the serial timer register (STMR). If the serial timer register (STMR) coincides with these bits when the serial timer register is updated, the serial timer register is set to "0". At that time, if synchronous transmission is disabled (SACSR:TSYNE="0"), the timer interrupt flag (SACSR:TINT) is set to "1". If synchronous transmission is enabled (SACSR:TSYNE="1"), transmission is activated.

The interval of the operations below is (STMCR:TC + 1) × timer operation clock (specified by SACSR:TDIV3 to 0).

- SACSR:TINT is set to "1".
- Serial-timer-synchronized transmission is activated.

### Notes:

- The timer interrupt flag (SACSR:TINT) is fixed to "1" when all of the following conditions are satisfied.
  - Synchronous transmission is disabled (SACSR:TSYNE="0").
  - "0x0000" is set in this register.
  - Timer operating
  - Set the timer operating clock division value (SACSR:TDIV[3:0]) to "0b0000".
- This register can be changed only when the serial timer is disabled (SACSR:TMRE="0").

## 9.9. Transfer Byte Register (TBYTE0)

The transfer byte (TBYTE0) represents the transfer data amount when each serial chip select pin is active.

BIT_OFFSET	7-0
BIT_NAME	TBYTE
ACCESS_TYPE	R/W
PROT_TYPE	-
INITIAL_VALUE	00000000

### [bit7:0] TBYTE[7:0]: Transfer data number indication bits

The transfer byte register 0 (TBYTE0) is used for synchronous transmission. The data amount set in TBYTE0 is transferred when synchronous transmission starts.

bit7:0	Description
Write	Writing to TBYTE0
Read	Setting of TBYTE0

**Note:**

- If one of these bits is set to "0x00", the transfer count is 8.



### 9.10. Baud Rate Generator Register 1, 0 (BGR1, BGR0)

The baud rate generator register 1, 0 (BGR1, BGR0) set the division ratio of the serial clock. An external clock can also be selected as the clock source of the reload counter.

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	EXT	BGR1						
ACCESS_TYPE	R/W	R/W						
PROT_TYPE	-							
INITIAL_VALUE	0	00000000						

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	BGR0							
ACCESS_TYPE	R/W							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

#### [bit15] EXT: External clock selection bit

This bit selects the internal clock or an external clock as the clock source of the reload counter. EXT = 0 selects the internal clock. EXT = 1 selects an external clock.

Value	Description
0	Use the internal clock
1	Use an external clock

#### [bit14:8] BGR1[6:0]: Baud rate generator register 1

bit14:8	Description
Write	Write to reload counter bits 8 to 14
Read	Read the setting value of BGR1

#### [bit7:0] BGR0[7:0]: Baud rate generator register 0

bit7:0	Description
Write	Write to reload counter bits 0 to 7
Read	Read the setting value of BGR0

**Notes:**

- Use 16-bit access for writing a value to the baud rate generator register (BGR1, BGR0).
- If the setting values of the baud rate generator registers (BGR1, BGR0) are changed, the new values are reloaded when the counter value becomes "0x0000". Therefore, in order to enable the new value immediately, execute programmable clear (UPCL) after changing the values of BGR1/0.
- When the reload value is even, the "L" width of the reception serial clock is 1-bus-clock-cycle longer than the "H" width. When the reload value is odd, the "L" width and the "H" width of the serial clock are equal.
- Set a value of 4 or larger in BGR1/0. However, data may not be received normally depending on baud rate errors and the reload value setting.
- To change the clock setting to that of the external clock (EXT = 1) when the baud rate generator is operating, perform the following: 1. Write "0" to baud rate generators 1, 0 (BGR1, BGR0). 2. Execute programmable clear (UPCL). 3. Set the external clock (EXT = 1).



### 9.11. FIFO Control Register 1 (FCR1)

FIFO control register 1 (FCR1) selects transmission and reception FIFOs, enables transmission FIFO interrupts, and controls the interrupt flag.

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved		Reserved	FLSTE	FRIIE	FDRQ	FTIE	FSEL
ACCESS_TYPE	R/W0		R0,WX	R/W	R/W	R,W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	00		0	0	0	1	0	0

\* Lower byte of this register [bit7:0] is FIFO Control Register 0 (FCR0)

#### [bit15:13] Reserved: Reserved bits

#### [bit12] FLSTE: Retransmission data lost detection enable bit

This bit enables the detection of the FIFO retransmission data lost flag (FLST).

This bit is set to "0" when the FCR1C:FLSTEC bit is set to "1".

This bit is set to "1" when the FCR1S:FLSTES bit is set to "1".

"0": Disable FLST bit detection.

"1": Enable FLST bit detection.

Value	Description
0	Disable data lost detection
1	Enable data lost detection

#### Note:

- To set this bit to "1", set the FSET bit to "1" first and then set this bit to "1".

#### [bit11] FRIIE: Reception FIFO idle detection enable bit

When the reception FIFO contains valid data, this bit enables or disables the detection of reception idle status for the 8-bit time or longer. If the reception interrupt is enabled (SCR:RIE = 1), detection of reception idle status triggers a reception interrupt.

This bit is set to "0" when the FCR1C:FRIIEC bit is set to "1".

This bit is set to "1" when the FCR1S:FRIIES bit is set to "1".

"0": Disable detection of reception idle status.

"1": Enable detection of reception idle status.

Value	Description
0	Disable detection of reception FIFO idle
1	Enable detection of reception FIFO idle

#### Note:

- To use the reception FIFO, set this bit to "1".

**[bit10] FDRQ: Transmission FIFO data request bit**

This bit requests transmission FIFO data.

"1" of this bit indicates that transmission data is requested. At that time, a FIFO transmission interrupt request is issued if the transmission FIFO interrupt is enabled (FTIE = 1).

This bit is set to "0" when the FCR1C:FDRQC bit is set to "1".

FDRQ set conditions

- When transmission FIFO interrupt control is not used
  - FBYTE (for transmission) = 0 (Transmission FIFO is empty)
  - Reset of the transmission FIFO
- When transmission FIFO interrupt control is used
  - FTICR setting value  $\geq$  FTICR read value (The number of the transmission FIFO data is lower than an interrupt trigger level)
  - Reset of the transmission FIFO

FDRQ reset conditions

- "0" is written to this bit.
- The transmission FIFO is full.

Value	Description
0	No transmission FIFO data request
1	Transmission FIFO data request

**Notes:**

- Writing "0" to this bit is valid when the transmission FIFO is enabled.
- Writing "0" to this bit is prohibited when FBYTE (for transmission) = 0.
- Writing "1" to this bit does not affect the operation.
- The change of FSEL bit is prohibited when this bit is "0".
- If write necessary data to the transmission FIFO after a transmission interrupt occurs, write "0" to the FIFO transmission data request bit (FCR1:FDRQ) and clear to the interrupt request.



**[bit9] FTIE: Transmission FIFO interrupt enable bit**

This bit enables a transmission FIFO interrupt. If this bit is set to "1", an interrupt occurs when the FDRQ bit is "1".

This bit is set to "0" when the FCR1C:FTIEC bit is set to "1".

This bit is set to "1" when the FCR1S:FTIES bit is set to "1".

Value	Description
0	Disable transmission FIFO interrupts
1	Enable transmission FIFO interrupts

**[bit8] FSEL: FIFO selection bit**

This bit selects the transmission and reception FIFOs.

This bit is set to "0" when the FCR1C:FSELC bit is set to "1".

This bit is set to "1" when the FCR1S:FSELS bit is set to "1".

"0": Allocation is made as a transmission FIFO: FIFO1 and reception FIFO: FIFO2.

"1": Allocation is made as a transmission FIFO: FIFO2 and reception FIFO: FIFO1.

Value	Description
0	Transmission FIFO: FIFO1, reception FIFO: FIFO2
1	Transmission FIFO: FIFO2, reception FIFO: FIFO1

**Notes:**

- This bit is not cleared by a FIFO reset (FCR0:FCL2, FCL1 = 1).
- To change the value of this bit, disable the FIFO operation (FCR0:FE2, FE1 = 0) first.
- The change of this bit is prohibited when FDRQ = 0.

## 9.12. FIFO Control Register 0 (FCR0)

FIFO control register 0 (FCR0) enables/disables the FIFO operation, resets FIFO, saves the read pointer, and sets retransmission.

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved	FLST	FLD	FSET	FCL2	FCL1	FE2	FE1
ACCESS_TYPE	R0,W0	R,WX	R,W	R0,W	R0,W	R0,W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

**[bit7] Reserved: Reserved bit**

### **[bit6] FLST: FIFO retransmission data lost flag bit**

This bit indicates that the retransmission data of the transmission FIFO has been lost.

FLST set condition

- When data is written in the FIFO (overwriting the contents) when the FLSTE bit in the FIFO control register 1 (FCR1) is "1" and the write pointer of the transmission FIFO and the read pointer saved by the FSET bit coincide.

FLST reset conditions

- FIFO reset (writing "1" to FCL)
- Writing "1" to the FSET bit

If "1" is set in this bit, the data pointed to by the read pointer saved with the FSET bit is overwritten. For this reason, retransmission with the FLD bit cannot be set even if an error occurs. When performing retransmission with this bit set to "1", reset the FIFO and then write the data to the FIFO again.

Value	Description
0	Data has not been lost
1	Data has been lost

**[bit5] FLD: FIFO pointer reload bit**

This bit reloads, to the read pointer, the data saved to the transmission FIFO by the FSET bit. This bit is used for retransmission in cases such as communication errors.

This bit is set to "0" when the retransmission setting has been completed.

This bit is set when the FCR0S:FLDS bit in the set register is set to "1".

Value	Description
0	Does not execute reload
1	Execute reload

**Notes:**

- While this bit is set to "1", reloading to the read pointer is in progress, so writing is prohibited except in the case of a FIFO reset.
- Setting this bit to "1" is prohibited when FIFO is enabled or transmission is in progress.
- First set TIE and TBIE bits to "0" and write "1" to this bit, and then enable the transmission FIFO and set the SCR:TIE and SCR:TBIE bits to "1".

**[bit4] FSET: FIFO pointer saving bit**

This bit stores the read pointer for the transmission FIFO.

If the read pointer is saved before communication starts and then a communication error occurs, retransmission is possible if the FLST bit is "0".

This bit is set when the FCR0S:FSETS bit in the set register is set to "1".

"1": Save the current read pointer value.

"0": No effect.

Value	Description	
	Read	Write
0	"0" is always read	Does not save
1		Save the read pointer value

**Note:**

- Set this bit to "1" when the transmission byte number (FBYTE) is 0.

**[bit3] FCL2: FIFO2 reset bit**

This bit resets FIFO2.

If this bit is set to "1", the internal status of FIFO2 is initialized.

Only the FCR0:FLST bit is initialized and the values of the other bits in the FCR1/0 registers are retained.

This bit is set when the FCR0S:FCL2S bit in the set register is set to "1".

Value	Description	
	Read	Write
0	"0" is always read	No effect
1		Reset FIFO2

**Notes:**

- Disable transmission/reception first and then execute a FIFO2 reset.
- Execute after setting the transmission FIFO interrupt enable bit to "0".
- The valid data number of the FBYTE2 register is set to 0.

**[bit2] FCL1: FIFO1 reset bit**

This bit resets FIFO1.

If this bit is set to "1", the internal status of FIFO1 is initialized.

Only the FCR0:FLST bit is initialized and the values of the other bits in the FCR1/0 registers are retained.

This bit is set when the FCR0S:FCL1S bit in the set register is set to "1".

Value	Description	
	Read	Write
0	"0" is always read	No effect
1		Reset FIFO1

**Notes:**

- Disable transmission/reception first and then execute a FIFO1 reset.
- Execute after setting the transmission FIFO interrupt enable bit to "0".
- The valid data number of the FBYTE1 register is set to 0.

**[bit1] FE2: FIFO2 operation enable bit**

This bit enables or disables the operation of FIFO2.

- Set this bit to "1" when using FIFO2.
- When FIFO2 is specified as the transmission FIFO (FCR1:FSEL=1) and "1" is written to this bit, transmission starts immediately if FIFO2 contains data and UART transmission is enabled (SCR:TXE = 1). At that time, set the SCR:TIE and SCR:TBIE bits to "1", first set them to "0", and then set the SCR:TIE and SCR:TBIE bits to "1".
- A reception error clears this bit to "0" if the FIFO is selected as the reception FIFO by the FSEL bit. After that, this bit cannot be set to "1" unless the reception error is cleared.
- To use FIFO2 as the reception FIFO, first disable reception (SCR:RXE=0) and then change the setting of this bit when the reception buffer is empty (SSR:RDRF = "0") and the reception FIFO does not contain valid data (FBYTE2=0).
- To use FIFO2 as the reception FIFO, first disable reception (SCR:RXE=0) and then set this bit to "1" when the reception buffer is empty (SSR:RDRF = "0").
- The FIFO2 status is retained even if FIFO2 is disabled.
- This bit is reset when the FCR0C:FE2C bit in the clear register is set to "1".
- This bit is set when the FCR0S:FE2S bit in the set register is set to "1".

Value	Description
0	Disable FIFO2 operation
1	Enable FIFO2 operation

**[bit0] FE1: FIFO1 operation enable bit**

This bit enables or disables the operation of FIFO1.

- Set this bit to "1" when using FIFO1.
- When FIFO1 is specified as the transmission FIFO (FCR1:FSEL=0) and "1" is written to this bit, transmission starts immediately if FIFO1 contains data and UART transmission is enabled (SCR:TXE = 1). At that time, set the SCR:TIE and SCR:TBIE bits to "1", first set them to "0", and then set the SCR:TIE and SCR:TBIE bits to "1".
- A reception error clears this bit to "0" if the FIFO is selected as the reception FIFO by the FSEL bit. After that, this bit cannot be set to "1" unless the reception error is cleared.
- To use FIFO1 as the transmission or reception FIFO, set this bit to "1" or "0" when the transmission or reception buffer is empty ((SSR:TDRE=1) or (SSR:RDRF = 0)), respectively.
- To use FIFO1 as the reception FIFO, first disable reception (SCR:RXE=0) and then change the setting of this bit when the reception buffer is empty (SSR:RDRF=0).
- The FIFO1 status is retained even if FIFO1 is disabled.
- This bit is reset when the FCR0C:FE1C bit in the clear register is set to "1".
- This bit is set when the FCR0S:FE1S bit in the set register is set to "1".

Value	Description
0	Disable FIFO1 operation
1	Enable FIFO1 operation

### 9.13. FIFO Byte Register (FBYTE)

The FIFO byte register (FBYTE) indicates the valid data number of the FIFO. This register also specifies whether a reception interrupt is generated when the predefined number of data is received by the reception FIFO.

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	FBYTE2							
ACCESS_TYPE	R,W							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	FBYTE1							
ACCESS_TYPE	R,W							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

**[bit15:8] FBYTE2 [7:0]: FIFO2 data number indication bits**

**[bit7:0] FBYTE1 [7:0]: FIFO1 data number indication bits**

The FBYTE register indicates the valid number of data that has been written or received by the FIFO. Settings with the FCR1:FSEL bit are listed below.

Value	Description	
	FIFO Selection	Data Number Indication
0	FIFO2: Reception FIFO, FIFO1: Transmission FIFO	FIFO2:FBYTE2, FIFO1:FBYTE1
1	FIFO2: Transmission FIFO, FIFO1: Reception FIFO	FIFO2:FBYTE2, FIFO1:FBYTE1

- The initial value of transfer number of the FBYTE register is 0x08.
- Set the number of data items that generate a reception interrupt to FBYTE of the reception FIFO. When the number of the set transfer data and the data number indication in the FBYTE register coincide, the interrupt flag (SSR:RDRF) is set to "1".
- When both of the following conditions are satisfied, the continuation of reception idle status for 8 baud rate clocks or longer sets the interrupt flag (RDRF) to "1".
  - The reception FIFO idle detection enable bit (FRIIE) is "1".
  - The number of data in the reception FIFO does not reach the transfer number.
- During an 8-clock count, the counter is reset to 0 when RDR is read, and then starts counting the 8 clocks again. The counter is reset to 0 when the reception FIFO is disabled. If the reception FIFO is enabled when data remains in the reception FIFO, the counting restarts.

**FBYTE2, FBYTE1: FIFO2 data number indication bit, FIFO1 data number indication bit**

bit	Description
Write	Set the transfer number
Read	Read valid data number



Read (valid data number)

Transmission: The number of data written in the FIFO and not yet transmitted

Reception: The number of data received in the FIFO

Write (transfer number)

Transmission: Set 0x00

Reception: Set the data number that triggers reception interrupts

FIFO Capacity	Operation Mode	Data Length	Max FBYTE Count	Data Number That Can Be Stored
16 bytes	Mode 0	5 to 8 bits	16	16
	Mode 0	9 bits	8	8
	Mode 1	All		
32 bytes	Mode 0	5 to 8 bits	32	32
	Mode 0	9 bits	16	16
	Mode 1	All		
64 bytes	Mode 0	5 to 8 bits	64	64
	Mode 0	9 bits	32	32
	Mode 1	All		
128 bytes	Mode 0	5 to 8 bits	128	128
	Mode 0	9 bits	64	64
	Mode 1	All		

**Notes:**

- Set 0x00 in the FBYTE register for the transmission FIFO.
- Set "1" or a larger value in the FBYTE for the reception FIFO.
- Change the value after disabling the reception operation.
- Any setting that exceeds the FIFO capacity is prohibited.

## 9.14. Transmission FIFO Interrupt Control Register (FTICR)

The transmission FIFO interrupt control register (FTICR) sets the interrupts triggered by the valid data number of the FIFO transmission.

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	FTICR2							
ACCESS_TYPE	R,W							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	FTICR1							
ACCESS_TYPE	R,W							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

[bit15:8] FTICR2[7:0]: FIFO2 data number indication bits

[bit7:0] FTICR1[7:0]: FIFO1 data number indication bits

The FTICR register sets the trigger level for interrupts by the valid transmission data number (remaining amount) in the transmission FIFO. The table below shows the setting by FCR1:FSEL bit.

Value	Description	
	Selection of Transmission FIFO	Transmission FIFO Interrupt Control Register
0	FIFO1	FTICR1
1	FIFO2	FTICR2

- The initial value of the valid data number for triggering interrupts of the FTICR register is 0x00.
- Set the number of data that generate a transmission interrupt to FTICR of the transmission FIFO. The interrupt flag (FDRQ) is set to "1" when the number of the set data and the valid data number of transmission FIFO (FTICR or FBYTE) are equal or smaller.
- Set FTICR that it satisfies:  $FTICR \leq \text{FIFO capacity} - 2$ .
- The read value indicates the valid data number of the FIFO.
- Transmission FIFO: The number of data written in the transmission FIFO and not yet transmitted
- Reception FIFO: The number of data received in the reception FIFO and not yet read

bit	Description
Write	Set the number of valid data that generates an interrupt
Read	Read valid data number



**Notes:**

- Any setting that exceeds the FIFO capacity is prohibited.
- The setting value cannot be read.
- As the FIFO data number for transmission, the transmission data number that has been written, minus 1, is indicated as the valid data number. This is because the data store in transmission FIFO when write the transmission data to TDR register if the TDR register contains data that is yet to be transmitted. When the data in the TDR register is transmitted, the data in the transmission FIFO that has not yet been transmitted is transferred to the TDR register.
- FIFO data number indication at reception indicates the number of data received in the reception FIFO and not yet read. The data receiving in RDR registers does not include.

## 9.15. Serial Auxiliary Control Status Clear Register (SACSRC)

The serial auxiliary control status clear register (SACSRC) can clear the bits in the serial auxiliary control status register (SACSR).

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	STSTC	Reserved						TINTC
ACCESS_TYPE	R0,W	R0,W0						R0,W
PROT_TYPE	-							
INITIAL_VALUE	0	000000						0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	TINTEC	TSYNEC	Reserved					TMREC
ACCESS_TYPE	R0,W	R0,W	R0,W0					R0,W
PROT_TYPE	-							
INITIAL_VALUE	0	0	00000					0

### [bit15] STSTC: Clearing the serial test bit

Writing "1" to this bit resets SACSR:STST to "0".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

### [bit14:9] Reserved: Reserved bits

### [bit8] TINTC: Clearing the timer interrupt flag

Writing "1" to this bit resets SACSR:TINT to "0".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

### [bit7] TINTEC: Clearing the timer interrupt enable bit

Writing "1" to this bit resets SACSR:TINTE to "0".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

### [bit6] TSYNEC: Clearing the synchronous transmission enable bit

Writing "1" to this bit resets SACSR:TSYNE to "0".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

### [bit5:1] Reserved: Reserved bits

**[bit0] TMREC: Clearing the serial timer enable bit**

Writing "1" to this bit resets SACSr:TMRE to "0".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

## 9.16. FIFO Control Clear Register 1 (FCR1C)

FIFO control clear register 1 (FCR1C) can clear the bits in FIFO control register 1 (FCR1).

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved			FLSTEC	FRIIEC	FDRQC	FTIEC	FSELC
ACCESS_TYPE	R0,W0			R0,W	R0,W	R0,W	R0,W	R0,W
PROT_TYPE	-							
INITIAL_VALUE	000			0	0	0	0	0

\* Lower byte [bit7:0] of this register is FIFO Control Clear Register 0 (FCR0C)

### [bit15:13] Reserved: Reserved bits

#### [bit12] FLSTEC: Clearing the retransmission data lost detection enable bit

Writing "1" to this bit resets FCR1:FLSTE to "0".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

#### [bit11] FRIIEC: Clearing the reception FIFO idle detection enable bit

Writing "1" to this bit resets FCR1:FRIIE to "0".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

#### [bit10] FDRQC: Clearing the transmission FIFO data request bit

Writing "1" to this bit resets FCR1:FDRQ to "0".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

#### [bit9] FTIEC: Clearing the transmission FIFO interrupt enable bit

Writing "1" to this bit resets FCR1:FTIE to "0".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

#### [bit8] FSELC: Clearing the FIFO selection bit

Writing "1" to this bit resets FCR1:FSEL to "0".

Writing "0" to this bit is invalid.

"0" is always read from this bit.



### 9.17. FIFO Control Clear Register 0 (FCR0C)

FIFO control clear register 0 (FCR0C) can clear the bits in FIFO control register 0 (FCR0).

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved						FE2C	FE1C
ACCESS_TYPE	R0,W0						R0,W	R0,W
PROT_TYPE	-							
INITIAL_VALUE	00000						0	0

**[bit7:2] Reserved: Reserved bits**

#### **[bit1] FE2C: Clearing the FIFO2 operation enable bit**

Writing "1" to this bit resets FCR0:FE2 to "0".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

#### **[bit0] FE1C: Clearing the FIFO1 operation enable bit**

Writing "1" to this bit resets FCR0:FE1 to "0".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

## 9.18. Serial Auxiliary Control Status Set Register (SACSR)

The serial auxiliary control status set register (SACSR) can set the bits in the serial auxiliary control status register (SACSR).

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	STSTS	Reserved						
ACCESS_TYPE	R0,W	R0,W0						
PROT_TYPE	-							
INITIAL_VALUE	0	0000000						

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	TINTES	TSYNES	Reserved					TMRES
ACCESS_TYPE	R0,W	R0,W	R0,W0					R0,W
PROT_TYPE	-							
INITIAL_VALUE	0	0	00000					0

### [bit15] STSTS: Setting the serial test bit

Writing "1" to this bit sets the SACSR:STST to "1".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

### [bit14:8] Reserved: Reserved bits

### [bit7] TINTES: Setting the timer interrupt enable bit

Writing "1" to this bit sets SACSR:TINTE to "1".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

### [bit6] TSYNES: Setting the synchronous transmission enable bit

Writing "1" to this bit sets SACSR:TSYNE to "1".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

### [bit5:1] Reserved: Reserved bits

### [bit0] TMRES: Setting the serial timer enable bit

Writing "1" to this bit sets SACSR:TMRE to "1".

Writing "0" to this bit is invalid.

"0" is always read from this bit.



## 9.19. FIFO Control Set Register 1 (FCR1S)

FIFO control set register 1 (FCR1S) can set the bits in FIFO control register 1 (FCR1).

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved			FLSTES	FRIIES	Reserved	FTIES	FSELS
ACCESS_TYPE	R0,W0			R0,W	R0,W	R0,W0	R0,W	R0,W
PROT_TYPE	-							
INITIAL_VALUE	000			0	0	0	0	0

\* Lower byte [bit7:0] of this register is FIFO Control Set Register 0 (FCR0S)

**[bit15:13] Reserved: Reserved bits**

**[bit12] FLSTES: Setting the retransmission data lost detection enable bit**

Writing "1" to this bit sets FCR1:FLSTE to "1".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

**[bit11] FRIIES: Setting the reception FIFO idle detection enable bit**

Writing "1" to this bit sets FCR1:FRIIE to "1".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

**[bit10] Reserved: Reserved bit**

**[bit9] FTIES: Setting the transmission FIFO interrupt enable bit**

Writing "1" to this bit sets FCR1:FTIE to "1".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

**[bit8] FSELS: Setting the FIFO selection bit**

Writing "1" to this bit sets FCR1:FSEL to "1".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

## 9.20. FIFO Control Set Register 0 (FCR0S)

FIFO control set register 0 (FCR0S) can set the bits in FIFO control register 0 (FCR0).

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved		FLDS	FSETS	FCL2S	FCL1S	FE2S	FE1S
ACCESS_TYPE	R0,W0		R0,W	R0,W	R0,W	R0,W	R0,W	R0,W
PROT_TYPE	-							
INITIAL_VALUE	00		0	0	0	0	0	0

**[bit7:6] Reserved: Reserved bits**

**[bit5] FLDS: Setting the FIFO pointer reload bit**

Writing "1" to this bit sets FCR0:FLD to "1".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

**[bit4] FSETS: Setting the FIFO pointer saving bit**

Writing "1" to this bit sets FCR0:FSET to "1".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

**[bit3] FCL2S: Setting the FIFO2 reset bit**

Writing "1" to this bit sets FCR0:FCL2 to "1".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

**[bit2] FCL1S: Setting the FIFO1 reset bit**

Writing "1" to this bit sets FCR0:FCL1 to "1".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

**[bit1] FE2S: Setting the FIFO2 operation enable bit**

Writing "1" to this bit sets FCR0:FE2 to "1".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

**[bit0] FE1S: Setting the FIFO1 operation enable bit**

Writing "1" to this bit sets FCR0:FE1 to "1".

Writing "0" to this bit is invalid.

"0" is always read from this bit.





## 10. Precautions for Using

This section describes precautions for using the UART.

### **Note on DMA Transfer**

Using the Interrupt factor occurrence of UART, DMA controller is activated.

Set the DMA controller before activating it by UART. For DMA transfer, only 1 length is supported as for one block length (block count). (DMAi\_An:BC = 0)

For more details about settings of DMA controller, see "CHAPTER: DMA Controller".

# CHAPTER 36: CSIO (Clock Synchronous Serial Interface)



This chapter explains the CSIO functions of the multifunction serial interface, supported by operating mode 2.

---

1. Overview
2. Interrupts of the CSIO (Clock Synchronous Serial Interface)
3. Operation
4. Serial Timer Operation
5. Operation of Serial Chip Select
6. Test Mode
7. Dedicated Baud Rate Generator
8. Registers
9. Precautions for Using



## 1. Overview

The CSIO (Clock Synchronous Serial Interface) is a general-purpose serial data communication interface for performing synchronous communication with an external machine (supports SPI). In addition, this interface incorporates transmission/reception FIFOs (64 bytes each).

**Functions of the CSIO (Clock Synchronous Serial Interface)**

	Item	Function
1	Data buffer	<ul style="list-style-type: none"> <li>- Full-duplex double buffer (when FIFO is not used)</li> <li>- Transmission and reception FIFOs (64 bytes each) (when FIFO is used)</li> </ul>
2	Transfer format	<ul style="list-style-type: none"> <li>- Clock synchronization (no start/stop bits)</li> <li>- Master/slave functions</li> <li>- Supports SPI (supports both master and slave).</li> </ul>
3	Baud rate	<ul style="list-style-type: none"> <li>- A dedicated baud rate generator is provided (configured from a 15-bit reload counter, during master operation).</li> <li>- An external clock can be used.</li> </ul>
4	Data length	Variable to 5 to 16, 20, 24, and 32 bits.
5	Reception error detection	Overrun error
6	Interrupt request	<ul style="list-style-type: none"> <li>- Reception interrupt (reception completion, overrun error)</li> <li>- Transmission interrupt (transmission data empty, transmission bus idle, chip error interrupt)</li> <li>- Transmission FIFO interrupt (when the transmission FIFO is not higher than the interrupt trigger level or when the transmission FIFO is empty)</li> <li>- A DMA Transfer support function is provided for both transmission and reception.</li> <li>- Status interrupt (serial timer interrupt)</li> </ul>
7	Serial chip select	<ul style="list-style-type: none"> <li>- Channel 0: A serial chip select function is not provided.</li> <li>- Channel 1: A serial chip select function is not provided.</li> <li>- Channel 2: A serial chip select function is not provided.</li> <li>- Channel 3: A serial chip select function is not provided.</li> <li>- Channel 4: 4-channel control (independent control, round control)</li> <li>- The setup/hold/deselect times can be made variable.</li> <li>- The active level can be selected for each channel.</li> </ul>
8	Synchronous transmission function	<ul style="list-style-type: none"> <li>- Can automatically transmit data periodically in synchronization with the serial timer.</li> </ul>
9	Timer function	<ul style="list-style-type: none"> <li>- A 16-bit serial timer is provided.</li> <li>- An operating clock division value can be selected (division by 1 to 256).</li> </ul>
10	Synchronous mode	Master or slave function
11	Pin access	The serial data output pin can be set to "1".
12	FIFO option	<ul style="list-style-type: none"> <li>- Incorporates transmission and reception FIFOs (maximum capacity: 64 bytes for the transmission FIFO and 64 bytes for the reception FIFO)</li> <li>- The transmission and reception FIFOs can be selected.</li> <li>- Transmission data can be retransmitted.</li> <li>- The timing of the reception FIFO interrupt can be changed by software.</li> <li>- Independent FIFO reset is supported.</li> </ul>

## 2. Interrupts of the CSIO (Clock Synchronous Serial Interface)

The CSIO (Clock Synchronous Serial Interface) interrupts include reception interrupts, transmission interrupts, and status interrupts. Interrupt requests can be generated according to the factors described below.

- When reception data is set in the reception data register (RDR), or when a reception error occurs
- When transmission data is transferred from the transmission data register (TDR) to the transmission shift register, causing transmission to start
- Transmission bus idle (no transmission)
- Transmission FIFO data request
- When the serial timer comparison value (STMCR) and the serial timer value (STMR) coincide.
- A chip select error occurs.

### CSIO interrupts

Table 2-1 lists the CSIO interrupt control bits and interrupt factors.

**Table 2-1 CSIO Interrupt Control Bits and Interrupt Factors**

Interrupt Type	Interrupt Request Flag Bit	Flag Register	Interrupt Factor	Interrupt Factor Enable Bit	Interrupt Request Flag Clearing
Reception	RDRF	SSR	1-byte reception	SCR:RIE	Reading of reception data (RDR)
			Reception of as much data as the value set in FBYTE		Reading of reception data until the reception FIFO is empty
			Detection of reception idle for the 8-bit time or longer while the FRIIE bit is "1" and the reception FIFO contains valid data		
	ORE	SSR	Overrun error		Writing "1" to the reception error flag clear bit (SSR:REC)
Transmission	TDRE	SSR	The transmission register is empty.	SCR:TIE	Writing to the transmission data register (TDR) or writing "1" to the transmission FIFO operation enable bit (retransmission) when the transmission FIFO operation enable bit is "0" and the transmission FIFO contains valid data <sup>*1</sup>
	TBI	SSR	No transmission operation	SCR:TBIE	Writing to the transmission data register (TDR) or writing "1" to the transmission FIFO operation enable bit (retransmission) when the transmission FIFO operation enable bit is "0" and the transmission FIFO contains valid data <sup>*1</sup>
	FDRQ	FCR1	The amount of data stored in the transmission FIFO is equal to or smaller than the FTICR setting value or it is empty.	FCR1:FTIE	Writing "0" to the FIFO transmission data request clear bit (FCR1:FDRQ), or the transmission FIFO is full condition.
	CSE	SACSR	In slave mode (SCR:MS=1), the serial chip select pin is inactive during transmission. In master mode (SCR:MS = 0), the transmission count is equal to or less than the TBYTE setting and the next transmission data is not written to TDR (SSR:TDRE=1).	SACSR: CSEIE	Writing "0" to the serial chip select flag clear bit (SACSR:CSE)

CHAPTER 36:CSIO (Clock Synchronous Serial Interface)  
2. Interrupts of the CSIO (Clock Synchronous Serial Interface)

H A R D W A R E M A N U A L



Interrupt Type	Interrupt Request Flag Bit	Flag Register	Interrupt Factor	Interrupt Factor Enable Bit	Interrupt Request Flag Clearing
Status	TINT	SACSR	Coincidence of the serial timer register (STMR) and the serial timer comparison register (STMCR) values	SACSR: TINTE	Writing "0" to the timer interrupt flag clear bit (SACRC:TINTC)

\*1: Set the TIE bit to "1" after the TDRE bit becomes "0".



## 2.1. Reception Interrupt Generation and Flag Set Timing

Interrupts at time of reception include reception completion (SSR:RDRF) and reception error occurrence (SSR:ORE).

### Reception Interrupt Generation and Flag Set Timing

Upon the detection of the last data bit, reception data is stored in the reception data register (RDR). Once reception is completed (SSR:RDRF = 1) or when a reception error occurs (SSR:ORE=1), each flag is set. At this time, if reception interrupts are enabled (SSR:RIE=1), a reception interrupt is generated.

**Note:**

- If a reception error occurs, the data in the reception data register (RDR) will be invalid.

Figure 2-1 Reception Operation and Flag Set Timing

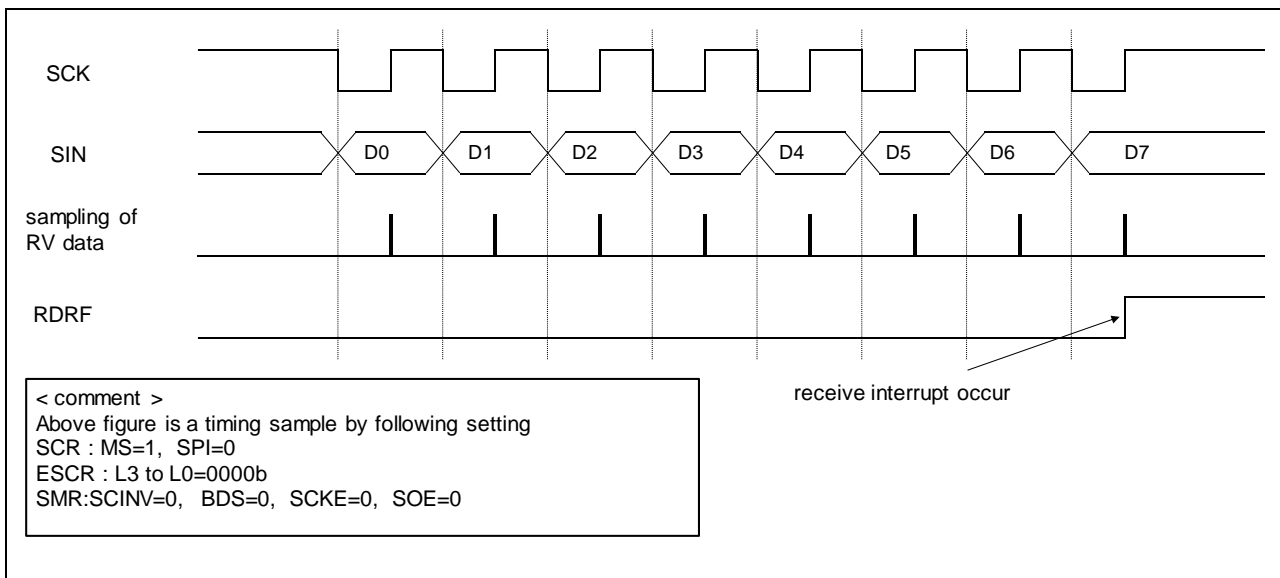
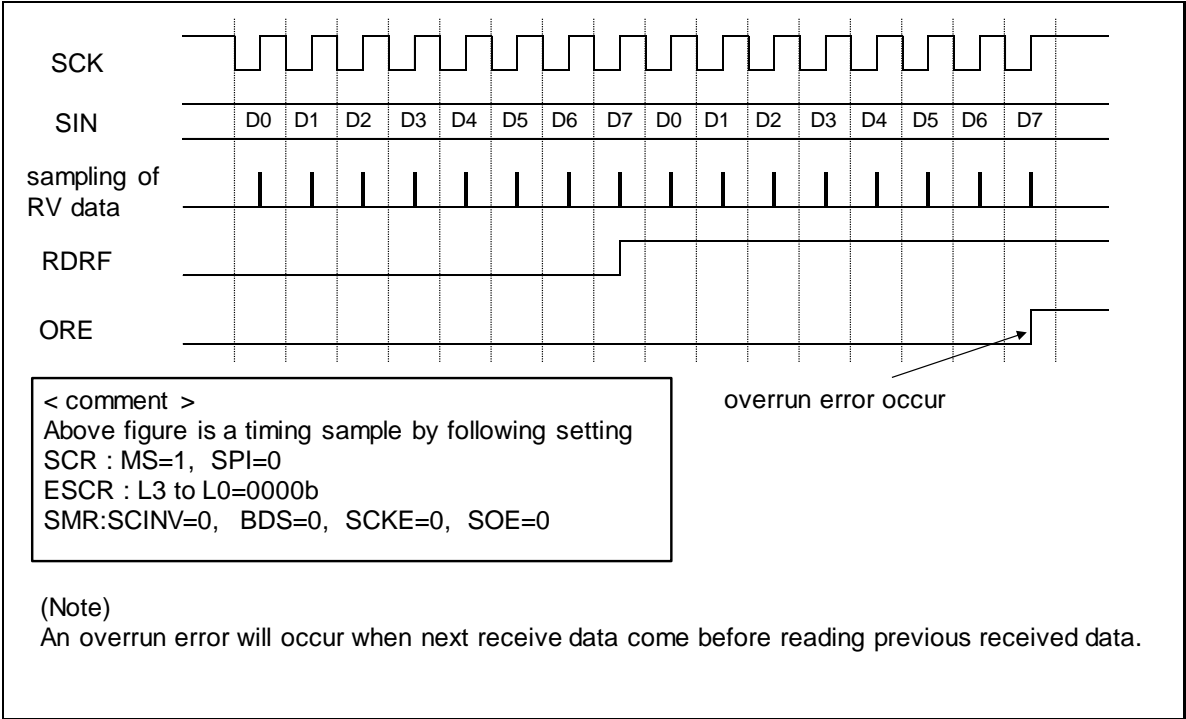




Figure 2-2 ORE (Overrun Error) Flag Set Timing





## 2.2. Interrupt Generation and Flag Set Timing When the Reception FIFO is Used

When the reception FIFO is used, an interrupt is generated if as many data items as the setting in the FBYTE register (FBYTE) are received.

### Reception Interrupt Generation and Flag Set Timing When the Reception FIFO is Used

When the reception FIFO is used, interrupt generation is determined with the setting of the FBYTE register.

- If as much transfer data as the amount set in the FBYTE register is received, the reception data full flag (SSR:RDRF) in the serial status register will be set to "1". At this time, if reception interrupts are enabled (SCR:RIE), a reception interrupt will be generated.
- When both of the following conditions are satisfied, continuation of reception idle status for 8 baud rate clocks or longer will set the interrupt flag (RDRF) to "1".
  - The reception FIFO idle detection enable bit (FRIIE) is "1".
  - The number of data items in the reception FIFO does not reach the transfer count.
- During an 8-clock count, the counter is reset to 0 when RDR is read, so that the system starts counting the 8 clocks again. The counter is reset to 0 when the reception FIFO is disabled. If the reception FIFO is enabled with data remaining in the reception FIFO, the system starts counting again.
- Once the reception FIFO becomes empty through reading of the reception data (RDR), the reception data full flag (SSR:RDRF) is cleared.
- If the reception valid data number becomes equal to the value of the FIFO capacity, reception of the next data triggers an overrun error (SSR:ORE=1).

**Figure 2-3 Reception Interrupt Generation Timing When Reception FIFO is Used**

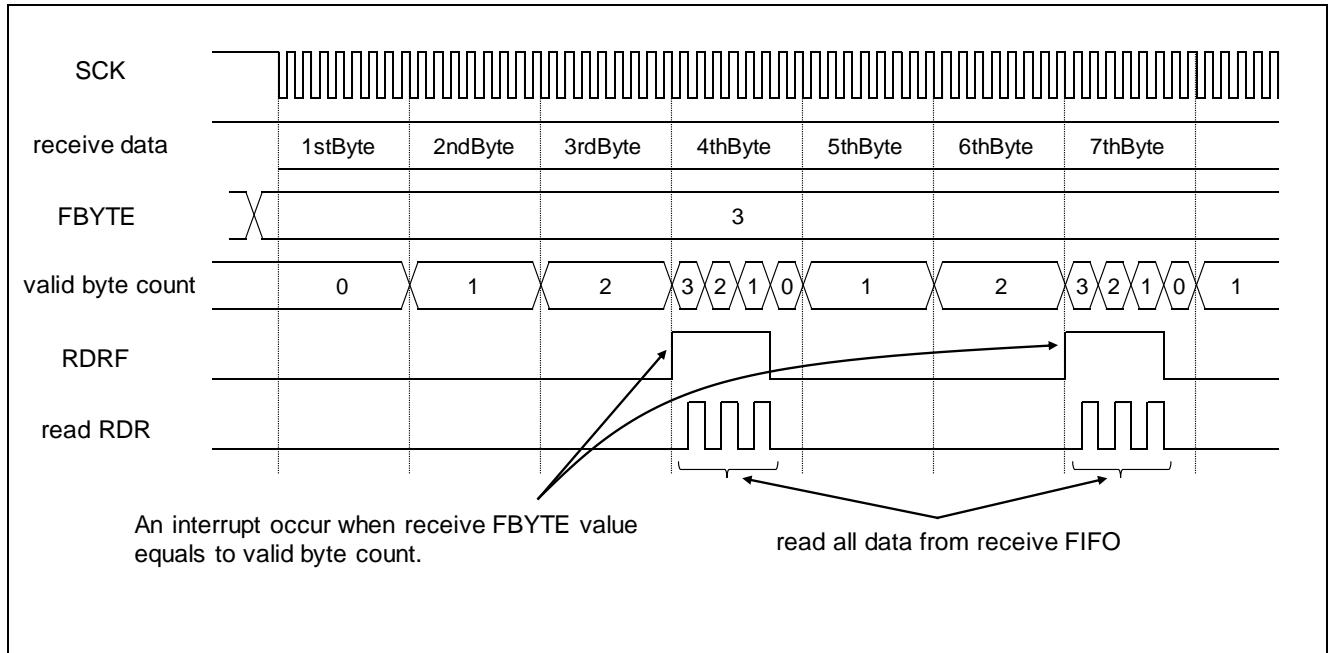
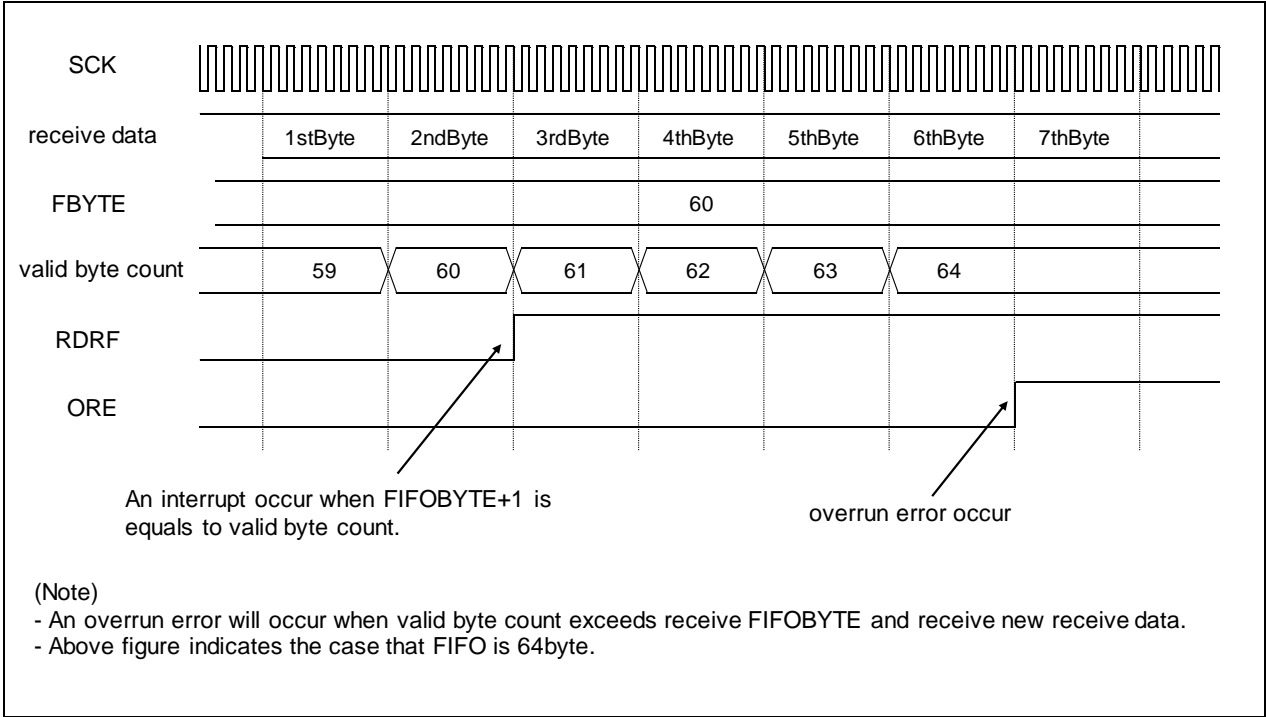




Figure 2-4 Set Timing of ORE (Overrun Error) Flag Bit



## 2.3. Transmission Interrupt Generation and Flag Set Timing

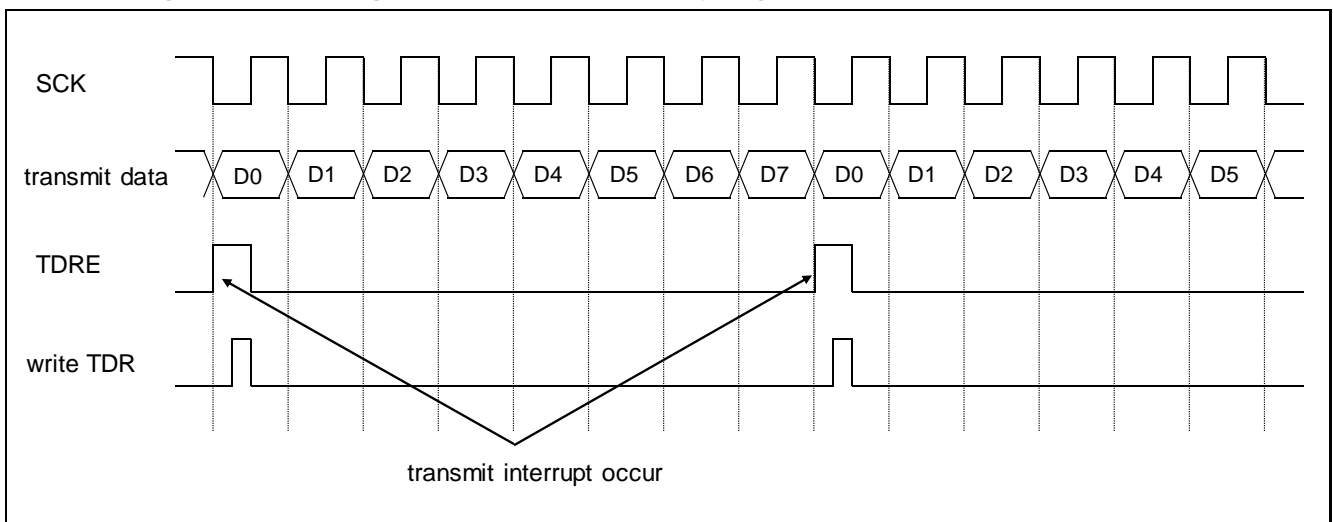
Interrupts at the time of transmission is generated if transmission data is transferred from the transmission data register (TDR) to the transmission shift register (SSR:TDRE=1) and transmission is started, or if transmission is not being performed (SSR:TBI=1).

### Transmission Interrupt Generation and Flag Set Timing

#### a) Set Timing of the Transmission Data Empty Flag (SSR:TDRE)

When the data written to the transmission data register (TDR) is transferred to the transmission shift register, the system enters the state in which the next data can be written (SSR:TDRE=1). At this time, if transmission interrupts are enabled (SCR:TIE=1), a transmission interrupt is generated. The SSR:TDRE bit is a read only bit, so the SSR:TDRE bit is cleared to "0" with the writing of data to the transmission data register (TDR).

Figure 2-5 Set Timing of Transmission Data Empty Flag (SSR:TDRE) (SCR:MS=0, SCR:SPI=0)

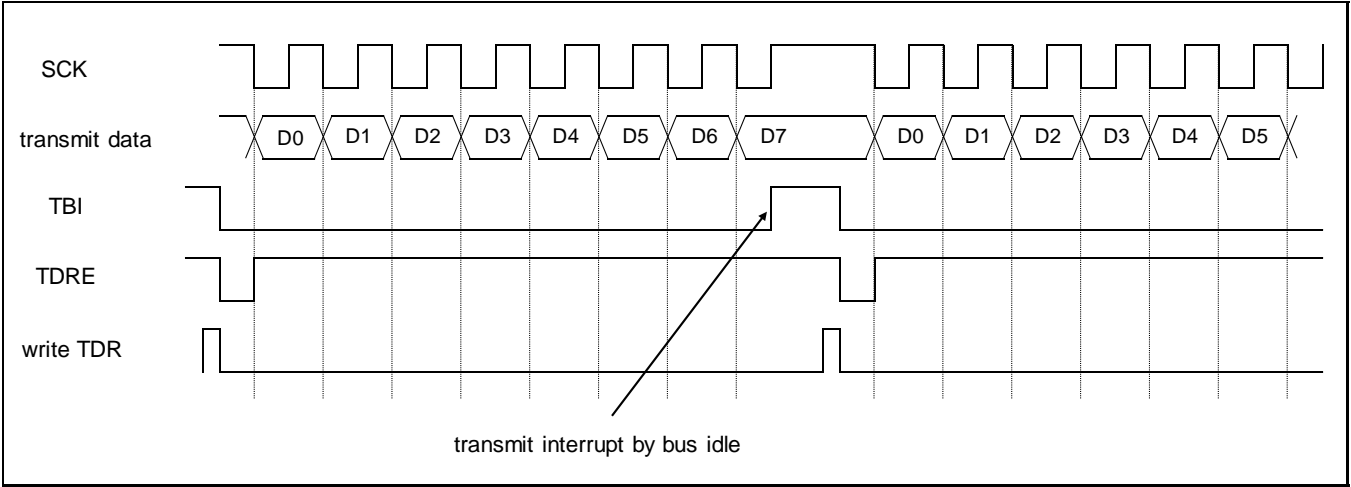


#### b) Set Timing of the Transmission Bus Idle Flag (SSR:TBI)

The SSR:TBI bit is set to "1" during those periods in which the transmission data register is empty (SSR:TDRE=1) and transmission is not being performed. At that time, a transmission interrupt occurs if the transmission bus idle interrupt is enabled (SCR:TBIE=1). If transmission data is set in the transmission data register (TDR), the SSR:TBI bit and the transmission interrupt request are cleared.



Figure 2-6 Set Timing of Transmission Bus Idle Flag (TBI) (SCSCR:CSEN3 to CSEN0 =0b0000, SACSR:TSYNE=0)





## 2.4. Interrupt Generation and Flag Set Timing When the Transmission FIFO is Used

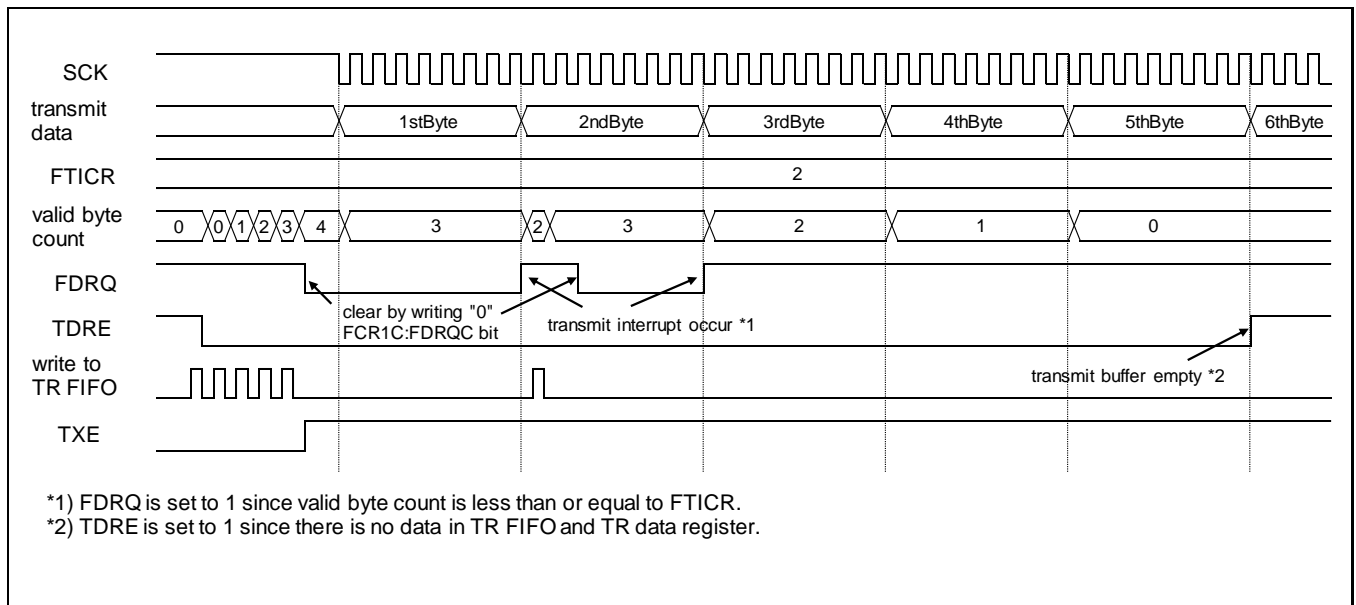
When the transmission FIFO is used, an interrupt is generated if the number of data items stored in the transmission FIFO is equal to or less than the setting made for the FTICR register (FTICR).

### Transmission Interrupt Generation and Flag Set Timing When the Transmission FIFO is Used

When the transmission FIFO is used, interrupt generation is determined with the setting of the FTICR register.

- When the data number in the transmission FIFO is equal to or less than the set value in the FTICR register, the FIFO transmission data request bit (FCR1:FDRQ) is set to "1".  
At this time, a transmission interrupt occurs if the FIFO transmission interrupt is enabled (FCR1:FTIE=1).
- After a transmission interrupt is generated, write the necessary data to the transmission FIFO, and then clear the interrupt request by writing "0" to the FIFO transmission data request bit (FCR1:FDRQ).
- The FIFO transmission data request bit (FCR1:FDRQ) is set to "0" once the transmission FIFO becomes full.
- The existence of data in the transmission FIFO can be verified by reading the FIFO byte register (FBYTE) or the transmission FIFO interrupt control register (FTICR).  
FBYTE=0x00 and FTICR = 0x00 indicate that the transmission FIFO does not contain data.

Figure 2-7 Interrupt Generation Timing When Transmission FIFO is Used



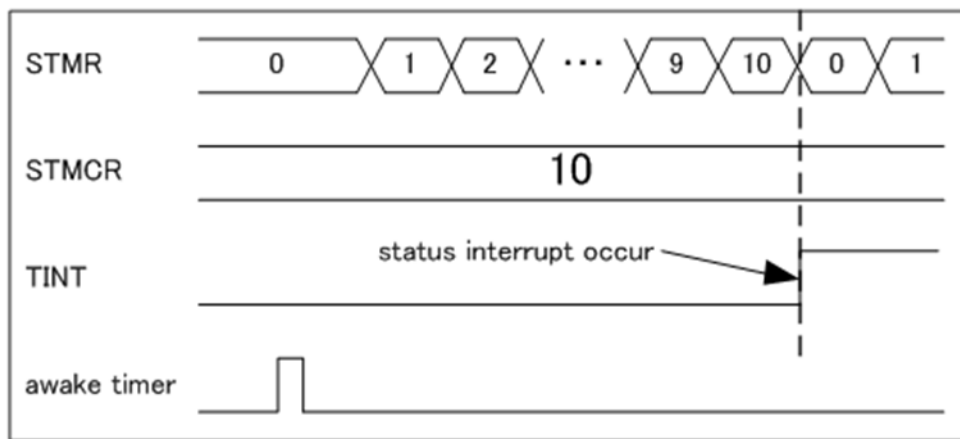
## 2.5. Timer Interrupt Generation and Flag Set Timing

A timer interrupt occurs when the serial timer register (STMR) coincides with the serial timer comparison register (STMCR).

### Timer Interrupt Generation and Flag Set Timing

- The timer interrupt flag (SACSR:TINT) is set to "1" when the serial timer register (STMR) coincides with the serial timer comparison register.
- At this time, a status interrupt occurs if the timer interrupt is enabled (SACSR:TINTE=1).

Figure 2-8 Timer Interrupt Generation Timing



## 2.6. Chip Select Error Occurrence and Flag Set Timing

A chip select error occurs in master mode (SCR:MS=0) if fewer frames than the value set in TBYTE have been transmitted, and the transmission data register (TDR) contains no valid data (SSR:TDRE=1). A chip select error also occurs during transmission in slave mode (SCR:MS=1) if the serial chip select pin becomes inactive.

### Chip Select Error Occurrence and Flag Set Timing

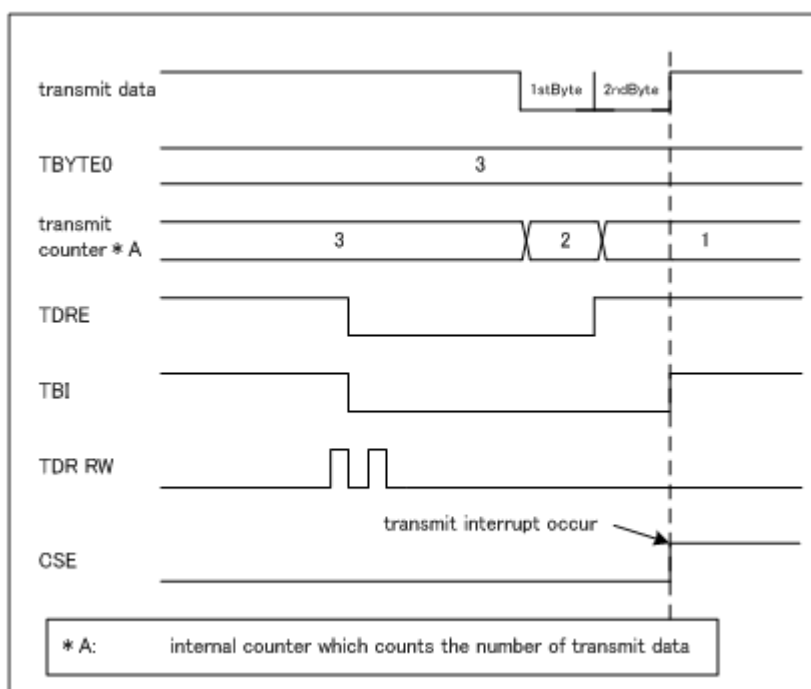
#### a) Master Mode (SCR:MS=0)

A chip select error occurs when transfer byte errors are enabled (TBEEN=1) in any of the following cases, if the transmission data register (TDR) contains no valid transmission data (SSR:TDRE=1) before as many data frames as the setting of TBYTE are transmitted.

- When chip select is used
- When synchronous transmission with the serial timer is used

At this time, if chip select error interrupts are enabled (SACSR:CSEIE=1), a transmission interrupt is generated.

Figure 2-9 Chip Select Error Occurrence Timing (SCSCR:CSEN3 to CSEN0 =0b0000, SACSR:TSYNE=1)



**Notes:**

- When serial chip select is used, the chip select error flag (SACSR:CSE) is set to "1" after the elapse of the deselect time from the occurrence of a chip select error. Even if transmission data is written to the transmission data register (TDR) within the hold delay time, transmission operation does not start, and the chip select error flag (SACSR:CSE) is set to "1" after the elapse of the deselect time.
- If "1" is set in the chip select error flag (SACSR:CSE), transmission does not start even if transmission data is written to the transmission data register (TDR).
- If "1" is set in the chip select error flag (SACSR:CSE) when synchronous transmission is used, transmission operation does not start under the conditions described below.
- The serial timer register (STMR) and the serial timer comparison register match during transmission in synchronization with the serial timer.

**b) Slave Mode (SCR:MS=1)**

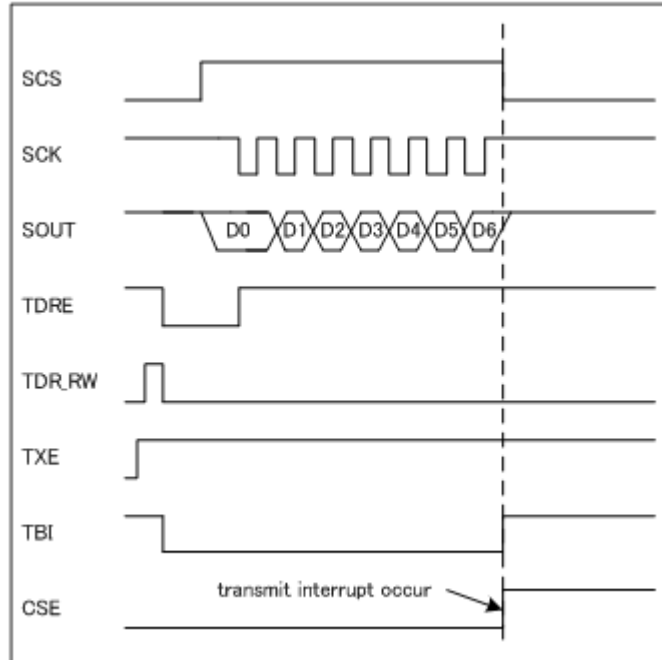
A chip select error occurs if chip select becomes inactive in any of the following cases.

- Serial clock in operation
- The transmission module is not idle and the serial clock has changed.

If the transmission module is not idle, this means that the transmission data has been prepared and that transmission starts when the serial clock is entered.

At this time, if chip select error interrupts are enabled (SACSR:CSEIE=1), a transmission interrupt is generated.

**Figure 2-10 Chip Select Error Occurrence Timing (CSLVL=0, SCR:SPI=0)**



**Note:**

- When the transmission data register (SSR:TDR) is empty (TDRE=1), if a serial chip select error (SACSR:CSE=1) occurs, this bit is set to "1" within the baud rate period.





### 3. Operation

Clock synchronization is used as the transfer method.

#### 3.1. Normal Transfer (I)

##### (1) Features

	Item	Description
1	Serial clock (SCK) mark level	"H"
2	Transmission data output timing	SCK falling edge
3	Reception data sampling	SCK rising edge
4	Data length	5 to 16, 20, 24 and 32 bits

##### (2) Register Setting

The register settings needed for normal transfer (I) are listed below.

SCR:SPI=0<sup>\*1</sup>, SMR:MD2=0, MD1=1, MD0=0, SCINV=0<sup>\*1</sup>

During master operation: SCR:MS=0, SMR:SCKE=1

During slave operation: SCR:MS=1, SMR:SCKE=0

\*1: The bit to be set differs depending on the condition. See Table 5-2.

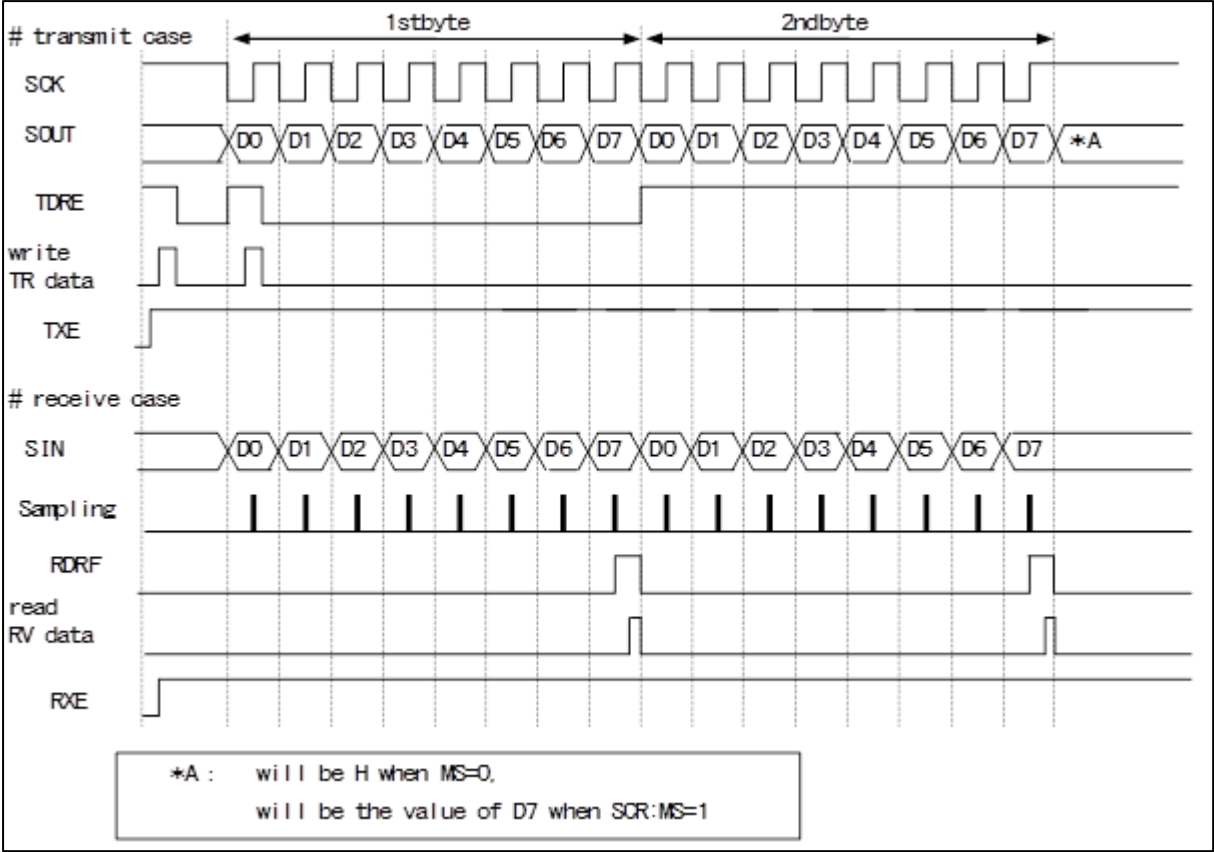
**Note:**

- Except for the bits described above, set the registers according to their use.



(3) Normal Transfer (I) Timing Chart (When the Serial Chip Select Pin is Not Used)

Figure 3-1 Normal Transfer (I) Timing Chart (When the Serial Chip Select Pin is Not Used)





#### (4) Master Operation (SCR:MS=0, SMR:SCKE=1, SCSCR:CSEN3 to CSEN0 ="0000")

##### a) Transmission Operation

1. After serial data output is enabled (SMR:SOE=1), transmission operation is enabled (SCR:TXE=1), and reception operation is disabled (SCR:RXE = 0), the writing of transmission data to TDR causes SSR:TDRE to be set to 0. In this way, transmission data is output in synchronization with a falling edge of the serial clock (SCK) output.
2. When the first 1 bit of the transmission data is output, SSR:TDRE is set to 1. Thus, if the transmission interrupts are enabled (SCR:TIE=1), a transmission interrupt request is output. At this time, the second byte of transmission data can be written.

##### b) Reception Operation

1. If serial data output is disabled (SMR:SOE=0) and transmission operation is enabled (SCR:TXE=1) while reception operation is enabled (SCR:RXE=1), writing dummy data to TDR causes the reception data to be sampled at a rising edge of the serial clock output (SCK).
2. When the last bit is received, SSR:RDRF is set to 1. At this time, if reception interrupts are enabled (SCR:RIE=1), a reception interrupt request is output. At this time, reception data (RDR) can be read.
3. When reception data (RDR) is read, SSR:RDRF is cleared to "0".

##### Notes:

- If performing reception operation only, write dummy data to TDR to output the serial clock (SCK).
- When the transmission/reception FIFOs are enabled, setting the number of frames to transfer in the FBYTE register causes as many serial clock (SCK) pulses as the number of frames to be output.

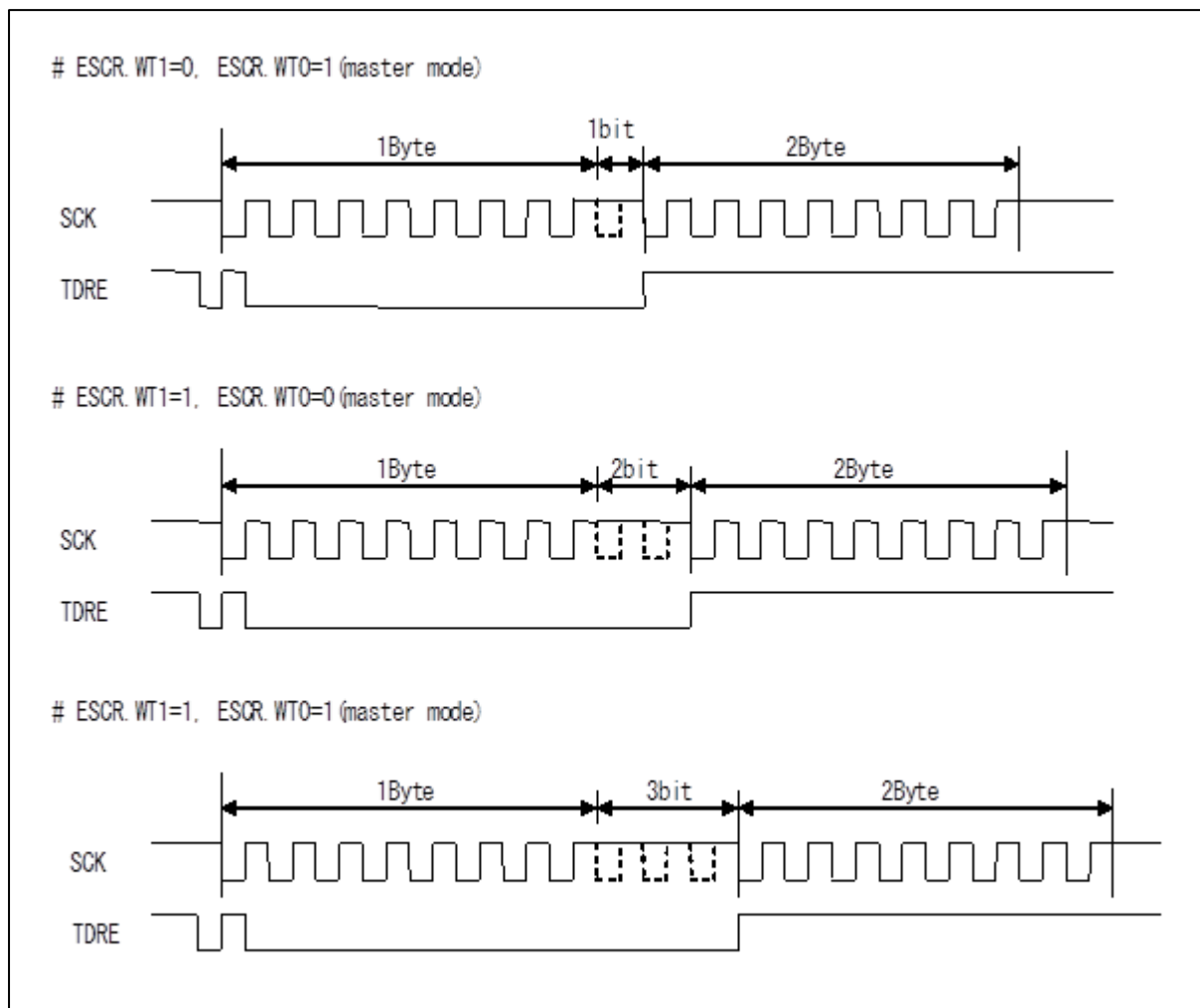
##### c) Transmission/Reception Operations

1. To perform transmission/reception at the same time, enable serial data output (SMR:SOE=1) and enable transmission/reception (SCR:TXE, RXE=1).
2. When transmission data is written to TDR, SSR:TDRE is set to 0, and transmission data is output in synchronization with a falling edge of the serial clock (SCK) output. When the first 1 bit of the transmission data is output, SSR:TDRE is set to 1, and if transmission interrupts are enabled (SCR:TIE=1), a transmission interrupt request is output. At this time, the second byte of the transmission data can be written.
3. Reception data is sampled at a rising edge of the serial clock (SCK) output. When the last bit of the reception data is received, SSR:RDRF is set to 1. If reception interrupts are enabled (SCR:RIE=1), a reception interrupt request is output. At this time, reception data (RDR) can be read. When reception data is read, SSR:RDRF is cleared to "0".

**d) Continuous Data Transmission or Reception Wait Operation**

If settings other than (ESCR:WT1, ESCR:WT0) = (0, 0) are set for continuous data transmission or reception, a wait is inserted between the frames.

**Figure 3-2 Wait Operation**





### (5) Slave Operation (SCR:MS=1, SMR:SCKE=0, SCSCR:CSEN0=0)

#### a) Transmission Operation

1. Enabling serial data output (SMR:SOE=1) as well as transmission (SCR:TXE=1) and then writing transmission data to TDR causes SSR:TDRE to be set to 0. For this reason, transmission data is output in synchronization with a falling edge of the serial clock (SCK) input.
2. When the first 1 bit of the transmission data is output, SSR:TDRE is set to 1. If transmission interrupts are enabled (SCR:TIE=1), a transmission interrupt request is output. At this time, the second byte of the transmission data can be written.

#### Note:

- After transmission operation is enabled (SCR:TXE = 1), if the first writing of transmission data to TDR is performed while the serial clock (SCK) is not at the mark level, the first 1 bit of data is not output, and transmission operation is not performed normally. After transmission is enabled (SCR:TXE = 1), perform the first writing of transmission data to TDR with SSR:TBI = 1 when the serial clock (SCK) is at the mark level.

#### b) Reception Operation

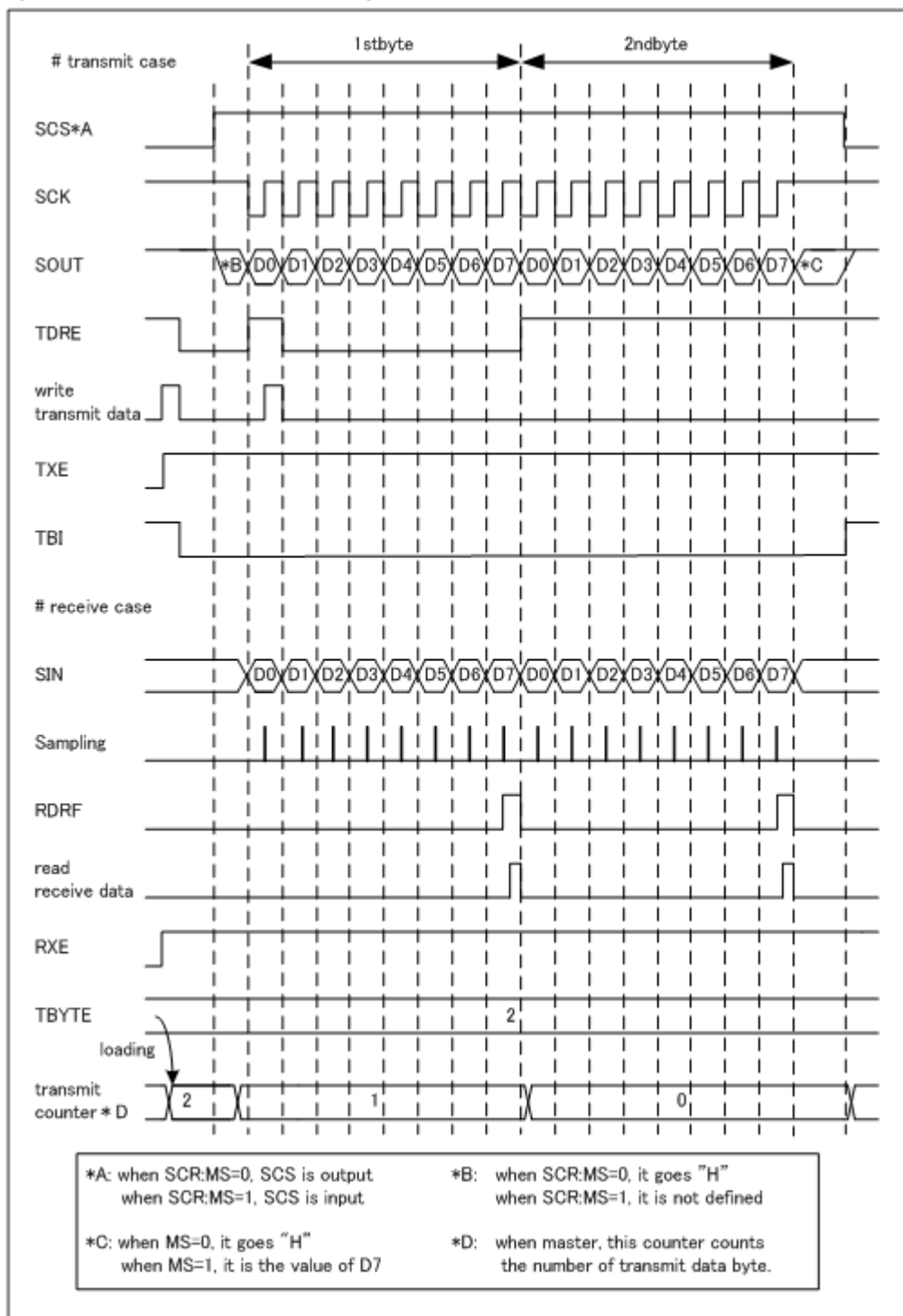
1. If serial data output is disabled (SMR:SOE=0) and reception operation is enabled (SCR:RXE=1), reception data is sampled at a rising edge of the serial clock input (SCK).
2. If the last bit is received, SSR:RDRF is set to 1. If reception interrupts are enabled (SCR:RIE=1), a reception interrupt request is output.  
At this time, reception data (RDR) can be read.
3. When reception data (RDR) is read, SSR:RDRF is cleared to "0".

#### c) Transmission/Reception Operations

1. To perform transmission/reception at the same time, enable serial data output (SMR:SOE=1) and enable transmission/reception (SCR:TXE, RXE=1).
2. When transmission data is written to TDR, SSR:TDRE is set to 0, and transmission data is output in synchronization with a falling edge of the serial clock (SCK) input. When the first 1 bit of transmission data is output, SSR:TDRE is set to 1, and if transmission interrupts are enabled (SCR:TIE=1), a transmission interrupt request is output. At this time, the second byte of transmission data can be written.
3. Reception data is sampled at a rising edge of the serial clock (SCK) input. If the last bit of the reception data is received, SSR:RDRF is set to 1, and if reception interrupts are enabled (SCR:RIE=1), a reception interrupt request is output. At this time, reception data (RDR) can be read. When reception data is read, SSR:RDRF is cleared to "0".

(6) Normal Transfer (I) Timing Chart (When the Serial Chip Select Pin is Used)

Figure 3-3 Normal Transfer (I) Timing Chart (When the Serial Chip Select Pin is Used)



**(7) Master Operation (SCR:MS=0, SMR:SCKE=1, SCSCR:CSCOE=1, SCSCR:CSENn\*=1)**

\*: n is replaced by the serial chip select pin number used.

**a) Transmission Operation**

1. Enabling serial data output (SMR:SOE=1) and transmission operation (SCR:TXE = 1), while disabling reception operation (SCR:RXE = 0), and then writing transmission data to TDR causes SSR:TDRE to be set to 0. Later, the serial chip select pin (SCS) becomes active, and after the elapse of the serial chip select pin setup time, serial clock output is started. After serial clock output is started, transmission data is output in synchronization with a falling edge of the serial clock (SCK) output.
2. When the first 1 bit of transmission data is output, SSR:TDRE is set to 1, and if transmission interrupts are enabled (SCR:TIE=1), a transmission interrupt request is output. At this time, the second byte of transmission data can be written.
3. After the end of data transmission for which the count is set in TBYTE, the serial clock stops.
4. After the serial clock stops, the serial chip select pin (SCS) becomes inactive after the elapse of the serial chip select pin hold time. At this time, however, if the serial chip select active level (SCSCR:SCAM=1) is retained, the serial chip select pin (SCS) retains the active state.

**b) Reception Operation**

1. Disabling serial data output (SMR:SOE=0), enabling transmission operation (SCR:TXE=1), and enabling reception operation (SCR:RXE = 1), and then writing dummy data to TDR, causes the serial chip select pin (SCS) to become active, and after the elapse of the serial chip select pin setup time, serial clock output is started. After serial clock output is started, reception data is sampled at a rising edge of the serial clock output (SCK).
2. When the last bit is received, SSR:RDRF is set to 1, and if reception interrupts are enabled (SCR:RIE = 1), a reception interrupt request is output.
3. At this time, reception data (RDR) can be read.
4. When reception data (RDR) is read, SSR:RDRF is cleared to "0".
5. After the end of data reception for which the count is set in TBYTE, the serial clock stops.
6. After the serial clock stops, the serial chip select pin (SCS) becomes inactive after the elapse of the serial chip select pin hold time. At this time, however, if the serial chip select active level (SCSCR:SCAM = 1) is retained, the serial chip select pin (SCS) retains the active state.

**Notes:**

- When performing reception only, write dummy data to TDR to output the serial clock (SCK).
- When the transmission/reception FIFOs are enabled, setting the number of frames to transfer in the FBYTE register causes as many serial clock (SCK) pulses as the number of frames to be output.

**c) Transmission/Reception Operations**

1. To perform transmission/reception at the same time, enable serial data output (SMR:SOE=1) as well as transmission/reception (SCR:TXE, RXE=1).
2. When transmission data is written to TDR, SSR:TDRE is set to 0. Later, the serial chip select pin (SCS) becomes active, and after the elapse of the serial chip select pin setup time, serial clock output is started. After serial clock output is started, transmission data is output in synchronization with a falling edge of the serial clock (SCK) output. When the first 1 bit of transmission data is output, SSR:TDRE is set to 1, and if transmission interrupts are enabled (SCR:TIE=1), a transmission interrupt request is output. At this time, the second byte of transmission data can be written.

3. During transmission/reception, reception data is sampled at a rising edge of the serial clock (SCK) output. If the last bit of the reception data is received, SSR:RDRF is set to 1, and if reception interrupts are enabled (SCR:RIE = 1), a reception interrupt request is output. At this time, reception data (RDR) can be read. When reception data is read, SSR:RDRF is cleared to "0".
4. After the end of data transmission/reception for which the count is set in TBYTE, the serial clock stops.
5. After the serial clock stops, the serial chip select pin (SCS) becomes inactive after the elapse of the serial chip select pin hold time. At this time, however, if the serial chip select active level (SCSCR:SCAM = 1) is retained, the serial chip select pin (SCS) retains the active state.

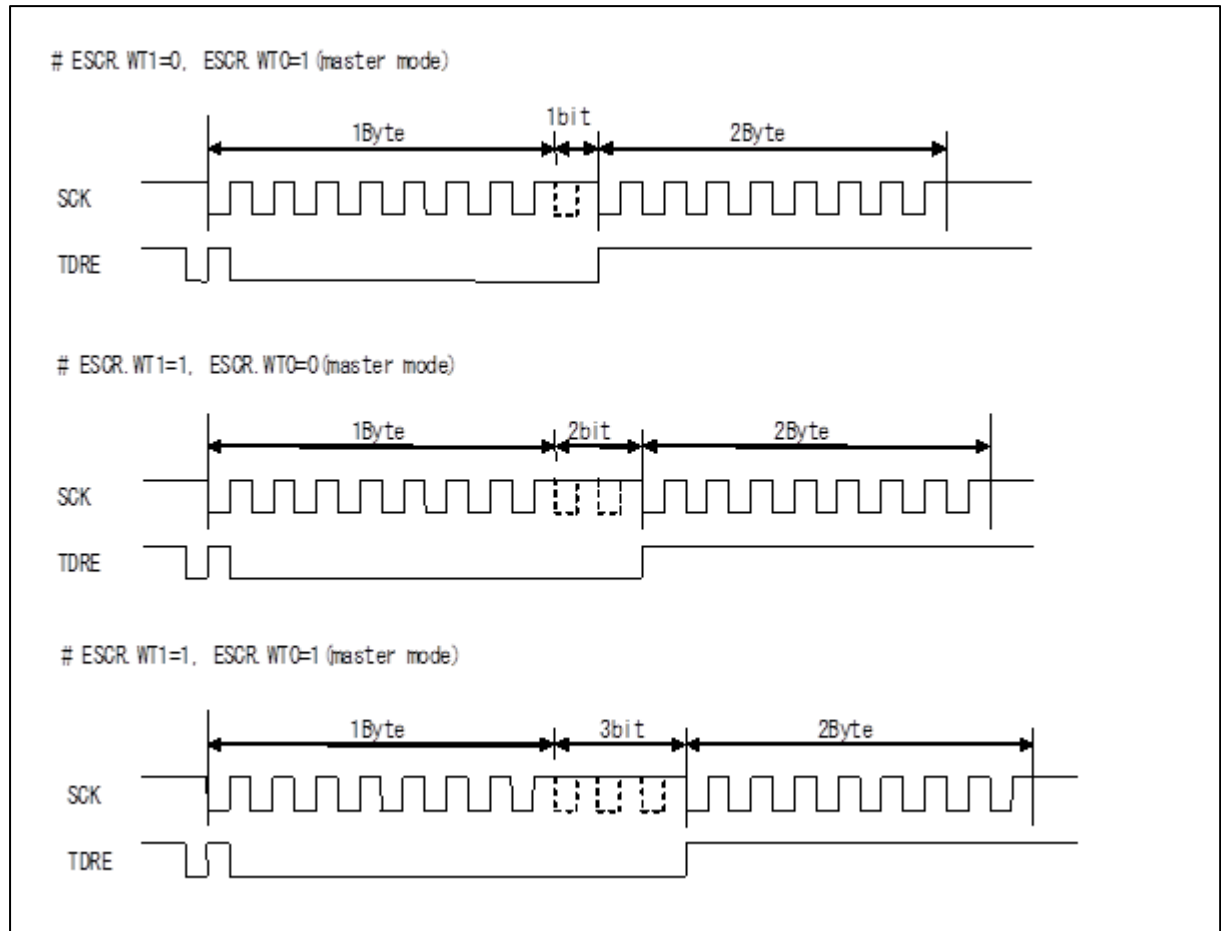




**d) Continuous Data Transmission or Reception Wait Operation**

If settings other than (ESCR:WT1, ESCR:WT0) = (0, 0) are set for continuous data transmission or reception, a wait is inserted between the frames.

**Figure 3-4 Wait Operation**



**(8) Slave Operation (SCR:MS=1, SMR:SCKE=0, SCSCR:CSEN0=1, SCSCR:CSOE=0, SCSCR:SCAM=0)**

**a) Transmission Operation**

1. Enabling serial data output (SMR:SOE=1) and enabling transmission (SCR:TXE=1) and then writing transmission data to TDR causes SSR:TDRE to be set to 0.
2. If the serial chip select pin (SCS) becomes active, transmission is started, and transmission data is output in synchronization with a falling edge of the serial clock (SCK) input.
3. When the first 1 bit of transmission data is output, SSR:TDRE is set to 1, and if transmission interrupts are enabled (SCR:TIE=1), a transmission interrupt request is output. At this time, the second byte of transmission data can be written.
4. If the serial chip select pin (SCS) becomes inactive, transmission operation is ended, and the serial output pin (SOUT) becomes "H".

**Note:**

- After transmission operation is enabled (SCR:TXE=1), if the first writing of transmission data to TDR is performed when the serial clock (SCK) is not at the mark level, the first 1 bit of data is not output, and transmission operation is not performed normally. After transmission is enabled (SCR:TXE = 1), perform the first writing of transmission data to TDR with SSR:TBI = 1 when the serial clock (SCK) is at the mark level.

**b) Reception Operation**

1. If serial data output is disabled (SMR:SOE=0) and reception is enabled (SCR:RXE = 1), and if the serial chip select pin (SCS) becomes active, reception starts, and reception data is sampled at a rising edge of the serial clock input (SCK).
2. When the last bit is received, SSR:RDRF is set to 1, and if reception interrupts are enabled (SCR:RIE=1), a reception interrupt request is output.
3. At this time, reception data (RDR) can be read.
4. When reception data (RDR) is read, SSR:RDRF is cleared to "0".
5. If the serial chip select pin (SCS) becomes inactive, reception operation is ended.

**c) Transmission/Reception Operations**

1. To perform transmission/reception operations at the same time, enable serial data output (SMR:SOE = 1) and enable transmission/reception operations (SCR:TXE, RXE=1).
2. When transmission data is written to TDR, SSR:TDRE is set to 0. Later, if the serial chip select pin (SCS) becomes active, transmission/reception operations are started, and transmission data is output in synchronization with a falling edge of the serial clock (SCK) input. When the first 1 bit of transmission data is output, SSR:TDRE is set to 1, and if transmission interrupts are enabled (SCR:TIE = 1), a transmission interrupt request is output. At this time, the second byte of transmission data can be written.
3. During transmission/reception, reception data is sampled at a rising edge of the serial clock (SCK) input. If the last bit of the reception data is received, SSR:RDRF is set to 1, and if reception interrupts are enabled (SCR:RIE=1), a reception interrupt request is output. At this time, reception data (RDR) can be read. When reception data is read, SSR:RDRF is cleared to "0".
4. If the serial chip select pin (SCS) becomes inactive, transmission/reception operations stop, and the serial output pin (SOUT) becomes "H".



## 3.2. Normal Transfer (II)

### (1) Features

	Item	Description
1	Serial clock (SCK) mark level	"L"
2	Transmission data output timing	SCK rising edge
3	Reception data sampling	SCK falling edge
4	Data length	5 to 16, 20, 24 and 32 bits

### (2) Register Setting

The register settings necessary for normal transfer (II) are listed below.

SCR:SPI<sup>\*1</sup>=0, SMR:MD2 to 0 =0b010, SCINV<sup>\*1</sup>=1

During master operation: SCR:MS=0, SMR:SCKE=1

During slave operation: SCR:MS=1, SMR:SCKE=0

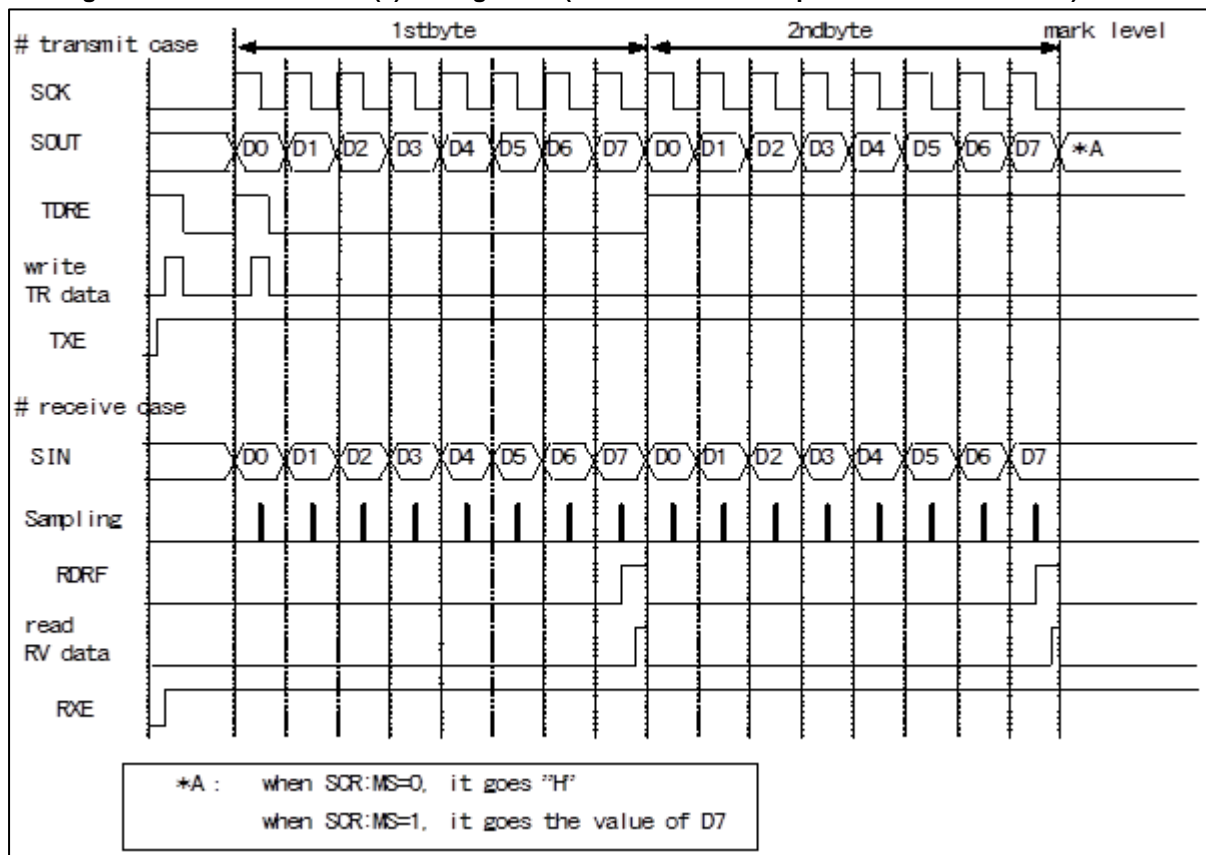
\*1: The bit to be set differs depending on the condition. See Table 5-2.

**Note:**

- Except for the above, set registers according to their use.

(2) Normal Transfer (II) Timing Chart (When the Serial Chip Select Pin is Not Used)

Figure 3-5 Normal Transfer (II) Timing Chart (When the Serial Chip Select Pin is Not Used)





### (3) Master Operation (SCR:MS=0, SMR:SCKE=1, SCSCR:CSEN3 to CSEN3 to 0 =0b000)

#### a) Transmission Operation

1. Enabling serial data output (SMR:SOE=1) and transmission operation (SCR:TXE = 1), while disabling reception operation (SCR:RXE = 0) and then writing transmission data to TDR causes SSR:TDRE to be set to 0. With this, transmission data is output in synchronization with a rising edge of the serial clock (SCK) output.
2. When the first 1 bit of transmission data is output, SSR:TDRE is set to 1. Thus, if transmission interrupts are enabled (SCR:TIE=1), a transmission interrupt request is output. At this time, the second byte of transmission data can be written.

#### b) Reception Operation

1. If serial data output is disabled (SMR:SOE=0), transmission is enabled (SCR:TXE=1), and reception is enabled (SCR:RXE=1), writing dummy data to TDR causes reception data to be sampled at a falling edge of the serial clock output (SCK).
2. If the last bit is received, SSR:RDRF is set to 1. At this time, if reception interrupts are enabled (SCR:RIE=1), a reception interrupt request is output. At this time, reception data (RDR) can be read.
3. When reception data (RDR) is read, SSR:RDRF is cleared to "0".

#### Notes:

- When performing reception only, write dummy data to TDR to output the serial clock (SCK).
- When the transmission/reception FIFOs are enabled, setting the number of frames to transfer in the FBYTE register causes as many serial clock (SCK) pulses as the number of frames to be output.

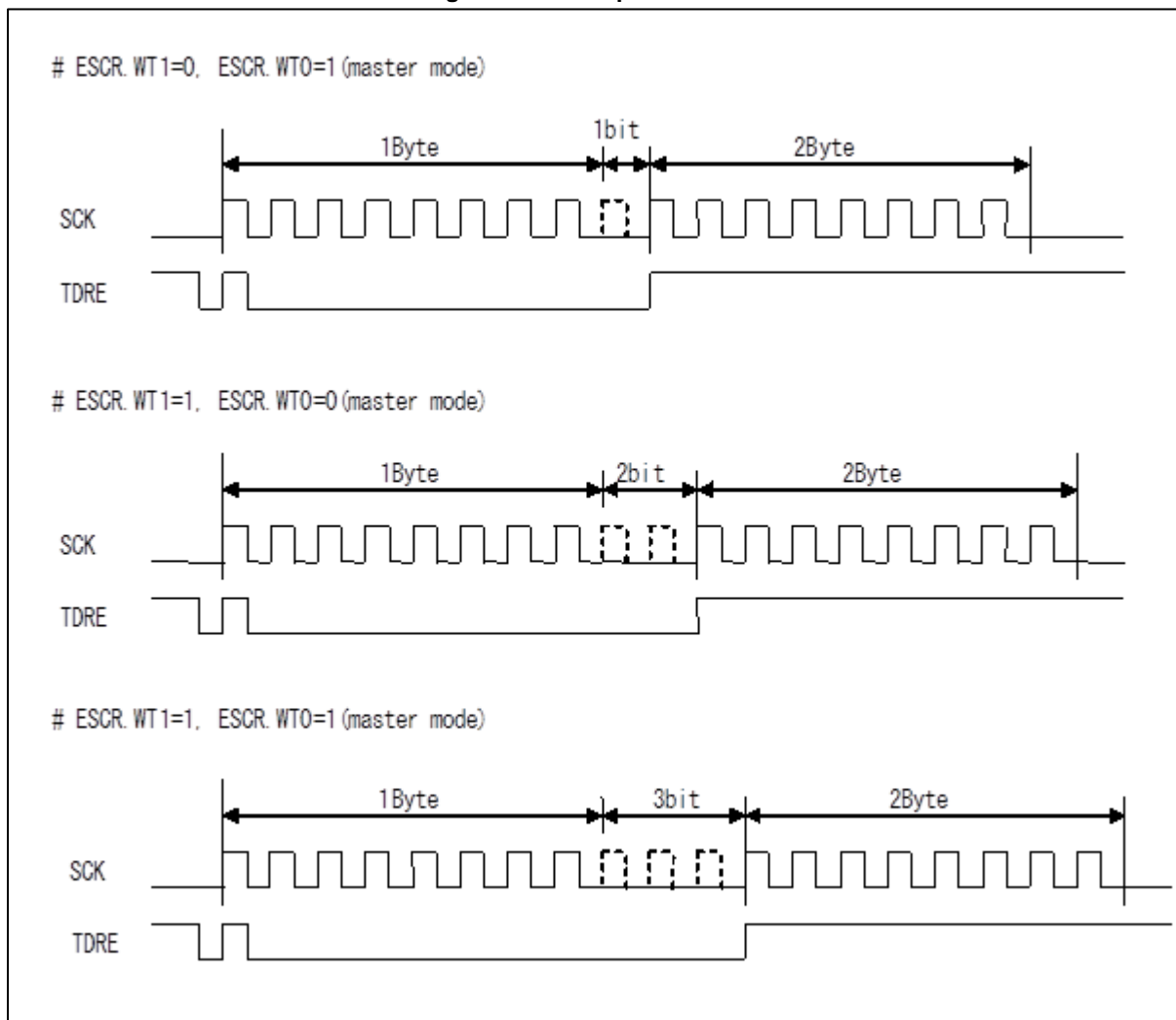
#### c) Transmission/Reception Operations

1. To perform transmission/reception operation at the same time, enable serial data output (SMR:SOE=1) and enable transmission/reception operation (SCR:TXE, RXE=1).
2. When transmission data is written to TDR, SSR:TDRE is set to 0, and transmission data is output in synchronization with a rising edge of the serial clock (SCK) output. When the first 1 bit of transmission data is output, SSR:TDRE is set to 1, and if transmission interrupts are enabled (SCR:TIE=1), a transmission interrupt request is output. At this time, the second byte of transmission data can be written.
3. Reception data is sampled at a falling edge of the serial clock (SCK) output. If the last bit of reception data is received, SSR:RDRF is set to 1. If reception interrupts are enabled (SCR:RIE = 1), a reception interrupt request is output. At this time, reception data (RDR) can be read. When reception data is read, SSR:RDRF is cleared to "0".

**d) Continuous Data Transmission or Reception Wait Operation**

If settings other than (ESCR:WT1, ESCR:WT0) = (0, 0) are set for continuous data transmission or reception, a wait is inserted between the frames.

**Figure 3-6 Wait Operation**





#### (4) Slave Operation (SCR:MS=1, SMR:SCKE=0, SCSCR:CSEN0=0)

##### a) Transmission Operation

1. Enabling serial data output (SMR:SOE = 1) and also transmission operation (SCR:TXE = 1) and then writing transmission data to TDR causes SSR:TDRE to be set to 0. For this reason, transmission data is output in synchronization with a rising edge of the serial clock (SCK) input.
2. When the first 1 bit of transmission data is output, SSR:TDRE is set to 1. If transmission interrupts are enabled (SCR:TIE = 1), a transmission interrupt request is output. At this time, the second byte of transmission data can be written.

##### Note:

- After transmission operation is enabled (SCR:TXE = 1), if the first writing of transmission data to TDR is performed when the serial clock (SCK) is not at the mark level, the first 1 bit of data is not output, and transmission operation is not performed normally. After transmission operation is enabled (SCR:TXE = 1), perform the first writing of transmission data to TDR with SSR:TBI = 1 when the serial clock (SCK) is at the mark level.

##### b) Reception Operation

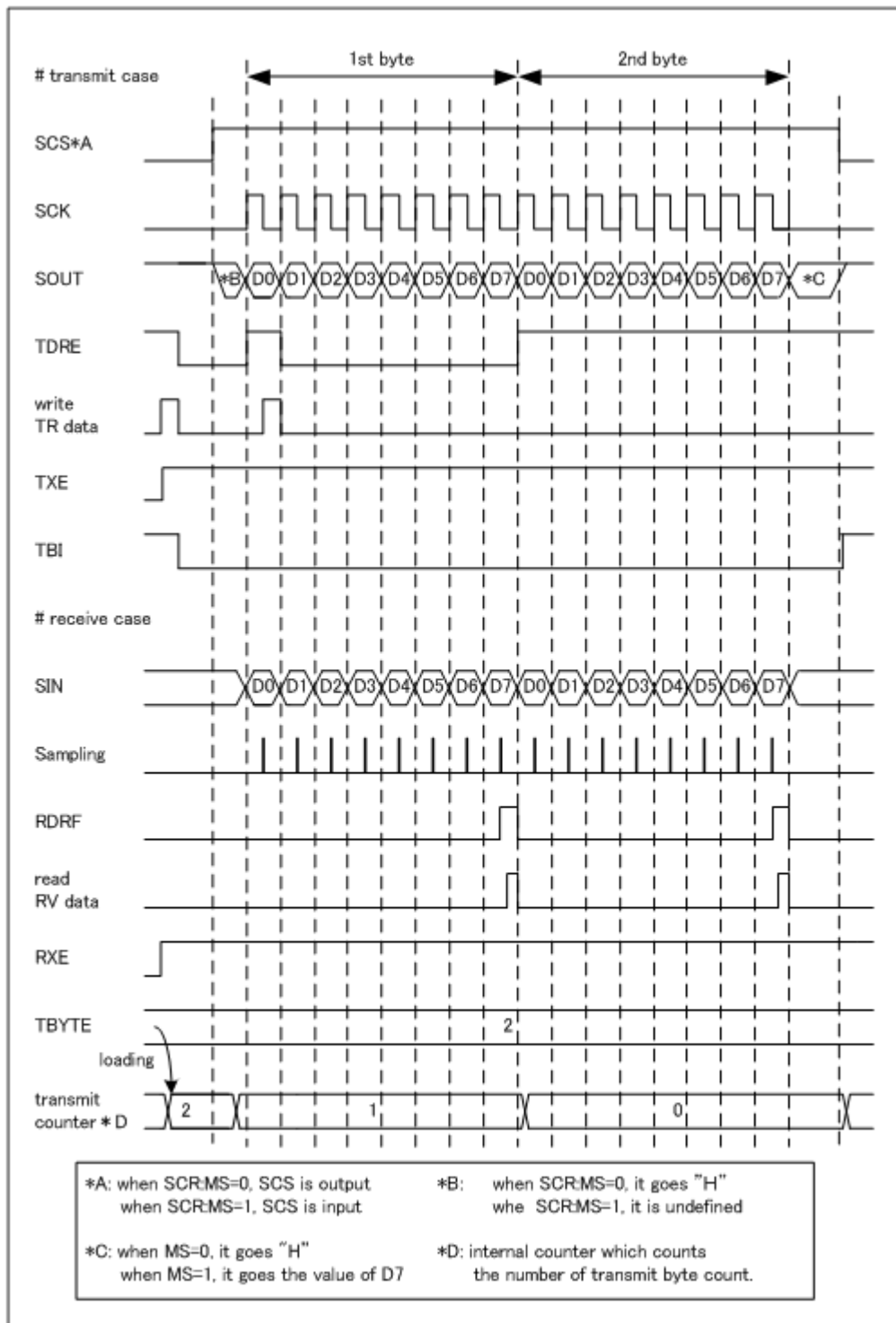
1. If serial data output is disabled (SMR:SOE = 0) and reception is enabled (SCR:RXE = 1), reception data is sampled at a falling edge of the serial clock input (SCK).
2. If the last bit is received, SSR:RDRF is set to 1. If reception interrupts are enabled (SCR:RIE = 1), a reception interrupt request is output.  
At this time, reception data (RDR) can be read.
3. When reception data (RDR) is read, SSR:RDRF is cleared to "0".

##### c) Transmission/Reception Operations

1. To perform transmission/reception operation at the same time, enable serial data output (SMR:SOE = 1) and enable transmission/reception operation (SCR:TXE, RXE = 1).
2. When transmission data is written to TDR, SSR:TDRE is set to 0, and transmission data is output in synchronization with a rising edge of the serial clock (SCK) input. When the first 1 bit of transmission data is output, SSR:TDRE is set to 1, and if transmission interrupts are enabled (SCR:TIE = 1), a transmission interrupt request is output. At this time, the second byte of transmission data can be written.
3. Reception data is sampled at a falling edge of the serial clock (SCK) input. If the last bit of reception data is received, SSR:RDRF is set to 1, and if reception interrupts are enabled (SCR:RIE = 1), a reception interrupt request is output. At this time, reception data (RDR) can be read. When reception data is read, SSR:RDRF is cleared to "0".

### (5) Normal Transfer (II) Timing Chart (When the Serial Chip Select Pin is Used)

Figure 3-7 Normal Transfer (II) Timing Chart (When the Serial Chip Select Pin is Used)





**(6) Master Operation (SCR:MS = 0, SMR:SCKE = 1, SCSCR:CSOE = 1, SCSCR:CSENn = 1)**

\*: n is replaced by the serial chip select pin number used.

**a) Transmission Operation**

1. Enabling serial data output (SMR:SOE = 1) and transmission operation (SCR:TXE = 1), while disabling reception operation (SCR:RXE = 0) and then writing transmission data to TDR causes SSR:TDRE to be set to 0. Later, the serial chip select pin (SCS) becomes active, and after the elapse of the serial chip select pin setup time, serial clock output is started. After serial clock output is started, transmission data is output in synchronization with a rising edge of the serial clock (SCK) output.
2. When the first 1 bit of transmission data is output, SSR:TDRE is set to 1, and if transmission interrupts are enabled (SCR:TIE = 1), a transmission interrupt request is output. At this time, the second byte of transmission data can be written.
3. After the end of data transmission for which the count is set in TBYTE, the serial clock is stopped.
4. After the serial clock stops, the serial chip select pin (SCS) becomes inactive after the elapse of the serial chip select pin hold time. At this time, however, if the serial chip select active level (SCSCR:SCAM = 1) is retained, the serial chip select pin (SCS) retains the active state.

**b) Reception Operation**

1. Disabling serial data output (SMR:SOE = 0) while enabling transmission operation (SCR:TXE = 1) and also reception operation (SCR:RXE = 1), and then writing dummy data to TDR, causes the serial chip select pin (SCS) to become active, and after the elapse of the serial chip select pin setup time, serial clock output is started. After serial clock output starts, reception data is sampled at a falling edge of the serial clock output (SCK).
2. When the last bit is received, SSR:RDRF is set to 1, and if reception interrupts are enabled (SCR:RIE = 1), a reception interrupt request is output.
3. At this time, reception data (RDR) can be read.
4. When reception data (RDR) is read, SSR:RDRF is cleared to "0".
5. After the end of data reception for which the count is set in TBYTE, the serial clock is stopped.
6. After the serial clock stops, the serial chip select pin (SCS) becomes inactive after the elapse of the serial chip select pin hold time. At this time, however, if the serial chip select active level (SCSCR:SCAM = 1) is retained, the serial chip select pin (SCS) retains the active state.

**Notes:**

- When performing reception only, write dummy data to TDR to output the serial clock (SCK).
- When the transmission/reception FIFOs are enabled, setting the number of frames to transfer in the FBYTE register causes as many serial clock (SCK) pulses as the number of frames to be output.

**c) Transmission/Reception Operations**

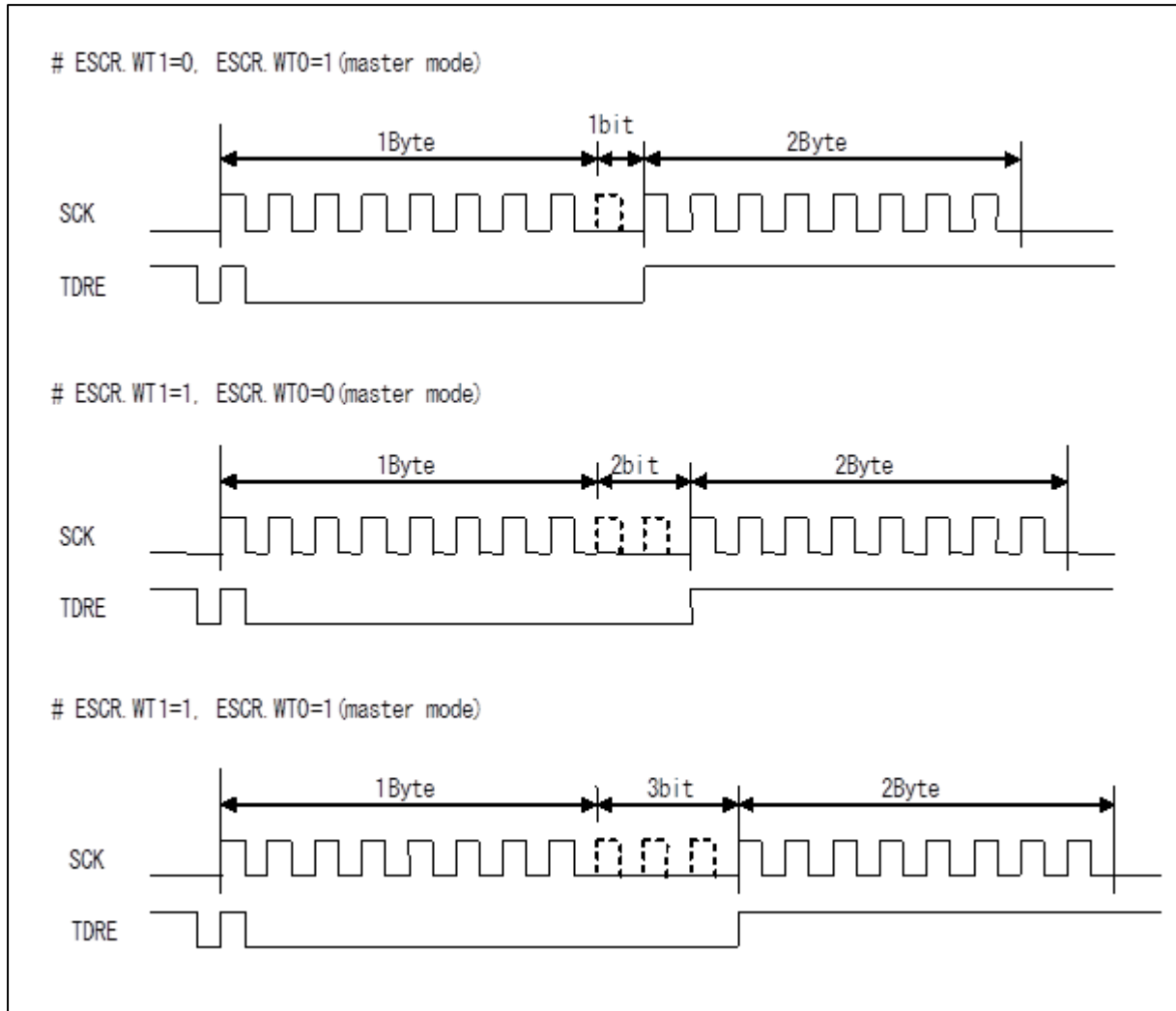
1. To perform transmission/reception operation at the same time, enable serial data output (SMR:SOE = 1) and enable transmission/reception operation (SCR:TXE, RXE = 1).
2. When transmission data is written to TDR, SSR:TDRE is set to 0. Later, the serial chip select pin (SCS) becomes active, and after the elapse of the serial chip select pin setup time, serial clock output is started. After serial clock output is started, transmission data is output in synchronization with a rising edge of the serial clock (SCK) output. When the first 1 bit of transmission data is output, SSR:TDRE is set to 1, and if transmission interrupts are enabled (SCR:TIE = 1), a transmission interrupt request is output. At this time, the second byte of transmission data can be written.

3. During transmission/reception operation, reception data is sampled at a falling edge of the serial clock (SCK) output. If the last bit of the reception data is received, SSR:RDRF is set to 1, and if reception interrupts are enabled (SCR:RIE = 1), a reception interrupt request is output. At this time, reception data (RDR) can be read. When reception data is read, SSR:RDRF is cleared to "0".
4. After the end of data transmission/reception for which the count is set in TBYTE, serial clock output is stopped.
5. After serial clock output is stopped, the serial chip select pin (SCS) becomes inactive after the elapse of the serial chip select pin hold time. At this time, however, if the serial chip select active level (SCSCR:SCAM=1) is retained, the serial chip select pin (SCS) retains the active state.

**d) Continuous Data Transmission or Reception Wait Operation**

If settings other than (ESCR:WT1, ESCR:WT0) = (0,0) are set for continuous data transmission or reception, a wait is inserted between the frames.

**Figure 3-8 Wait Operation**



**(7) Slave Operation (SCR:MS=1, SMR:SCKE=0, SCSCR:CSEN0=1, SCSCR:CCKOE=0, SCSCR:SCAM=0)**

**a) Transmission Operation**

1. Enabling serial data output (SMR:SOE=1) and also transmission operation (SCR:TXE=1) and then writing transmission data to TDR causes SSR:TDRE to be set to 0.
2. If the serial chip select pin (SCS) becomes active, transmission operation is started, and transmission data is output in synchronization with a rising edge of the serial clock (SCK) input.
3. When the first 1 bit of transmission data is output, SSR:TDRE is set to 1, and if transmission interrupts are enabled (SCR:TIE=1), a transmission interrupt request is output. At this time, the second byte of transmission data can be written.
4. If the serial chip select pin (SCS) becomes inactive, transmission operation is stopped, and the serial output pin (SOUT) becomes "H".

**Note:**

- After transmission is enabled (SCR:TXE=1), if the first writing of transmission data to TDR is performed when the serial clock (SCK) is not at the mark level, the first 1 bit of data is not output, and transmission is not performed normally. After transmission is enabled (SCR:TXE=1), perform the first writing of transmission data to TDR with SSR:TBI = 1 when the serial clock (SCK) is at the mark level.

**b) Reception Operation**

1. If serial data output is disabled (SMR:SOE=0) and reception operation is enabled (SCR:RXE = 1), and the serial chip select pin (SCS) becomes active, reception operation starts, and reception data is sampled at a falling edge of the serial clock input (SCK).
2. When the last bit is received, SSR:RDRF is set to 1, and if reception interrupts are enabled (SCR:RIE = 1), a reception interrupt request is output.
3. At this time, reception data (RDR) can be read.
4. When reception data (RDR) is read, SSR:RDRF is cleared to "0".
5. If the serial chip select pin (SCS) becomes inactive, reception operation is stopped.

**c) Transmission/Reception Operations**

1. To perform transmission/reception operation at the same time, enable serial data output (SMR:SOE=1) and enable transmission/reception operation (SCR:TXE, RXE=1).
2. When transmission data is written to TDR, SSR:TDRE is set to 0. Later, if the serial chip select pin (SCS) becomes active, transmission/reception operations are started, and transmission data is output in synchronization with a rising edge of the serial clock (SCK) input. When the first 1 bit of transmission data is output, SSR:TDRE is set to 1, and if transmission interrupts are enabled (SCR:TIE = 1), a transmission interrupt request is output. At this time, the second byte of transmission data can be written.
3. During transmission/reception operation, reception data is sampled at a falling edge of the serial clock (SCK) input. When the last bit of the reception data is received, SSR:RDRF is set to 1, and if reception interrupts are enabled (SCR:RIE=1), a reception interrupt request is output. At this time, reception data (RDR) can be read. When reception data is read, SSR:RDRF is cleared to "0".
4. If the serial chip select pin (SCS) becomes inactive, transmission/reception operations stop, and the serial output pin (SOUT) becomes "H".



### 3.3. SPI Transfer (I)

#### (1) Features

	Item	Description
1	Serial clock (SCK) mark level	"H"
2	Transmission data output timing	SCK rising edge
3	Reception data sampling	SCK falling edge
4	Data length	5 to 16, 20, 24 and 32 bits

#### (2) Register Setting

The register settings necessary for SPI transfer (I) are listed below.

SCR:SPI<sup>\*1</sup>=1, SMR:MD2 to 0 = 0b010, SCINV<sup>\*1</sup>=0

During master operation: SCR:MS=0, SMR:SCKE=1

During slave operation: SCR:MS=1, SMR:SCKE=0

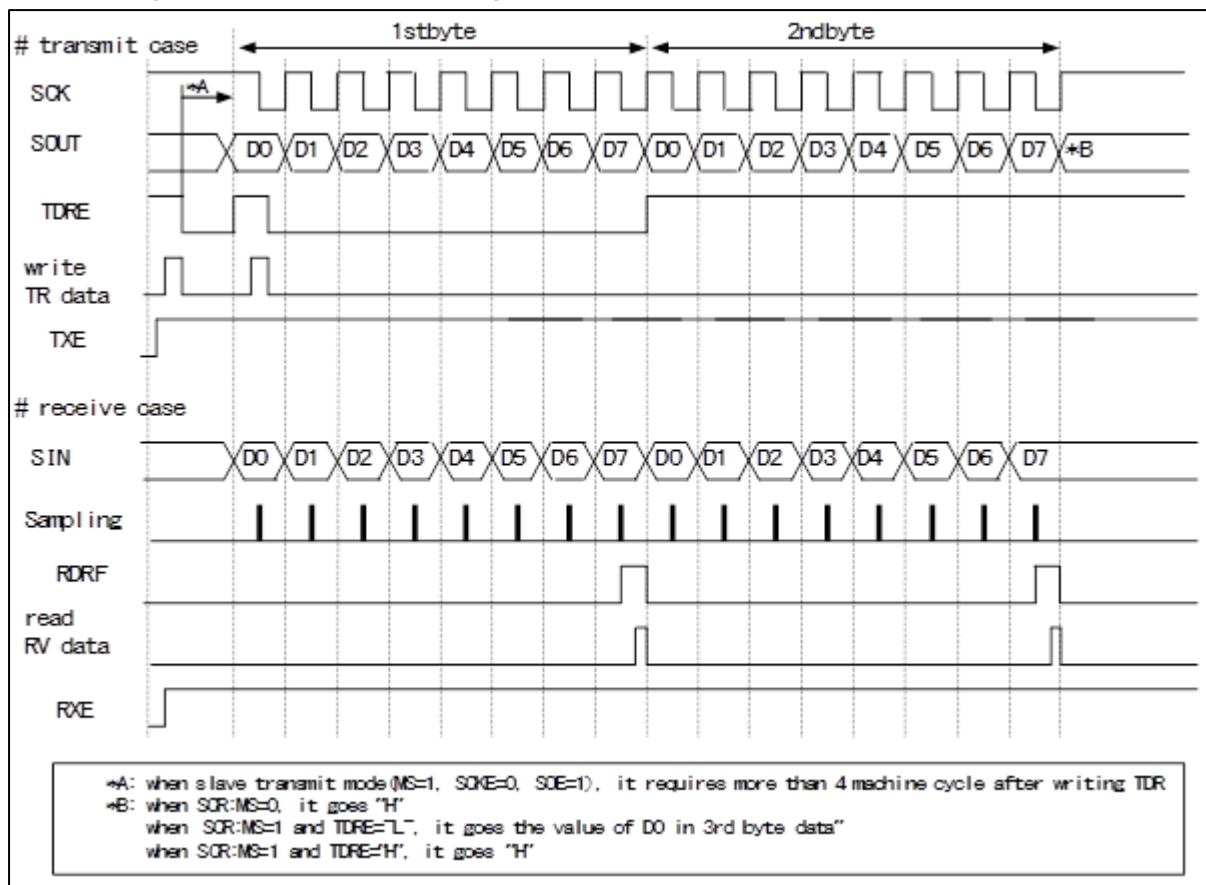
\*1: The bit to be set differs depending on the condition. See Table 5-2.

**Note:**

- Except for those above, set registers according to their use.

(3) SPI Transfer (I) Timing Chart (When the Serial Chip Select Pin is Not Used)

Figure 3-9 SPI Transfer (I) Timing Chart (When the Serial Chip Select Pin is Not Used)



**(4) Master Operation (SCR:MS=0, SMR:SCKE=1, SCSCR:CSEN3 to 0 =0b0000)****a) Transmission Operation**

1. Enabling serial data output (SMR:SOE=1), and also transmission operation (SCR:TXE=1), while disabling reception operation (SCR:RXE=0) and then writing transmission data to TDR causes SSR:TDRE to be set to 0. This causes the first bit to be output. Later, transmission data is output in synchronization with a rising edge of the serial clock (SCK) output.
2. Half a cycle before a falling edge of the first serial clock (SCK) output, SSR:TDRE is set to 1. Thus, if transmission interrupts are enabled (SCR:TIE=1), a transmission interrupt request is output. At this time, the second byte of transmission data can be written.

**b) Reception Operation**

1. If serial data output is disabled (SMR:SOE=0), and transmission operation is enabled (SCR:TXE = 1), and reception operation is enabled (SCR:RXE = 1), writing dummy data to TDR causes reception data to be sampled at a falling edge of the serial clock (SCK) output.
2. When the last bit is received, SSR:RDRF is set to 1. At this time, if reception interrupts are enabled (SCR:RIE=1), a reception interrupt request is output. At this time, reception data (RDR) can be read.
3. When reception data (RDR) is read, SSR:RDRF is cleared to "0".

**Notes:**

- When performing reception only, write dummy data to TDR to output the serial clock (SCK).
- When the transmission/reception FIFOs are enabled, setting the number of frames to transfer in the FBYTE register causes as many serial clock (SCK) pulses as the number of frames to be output.

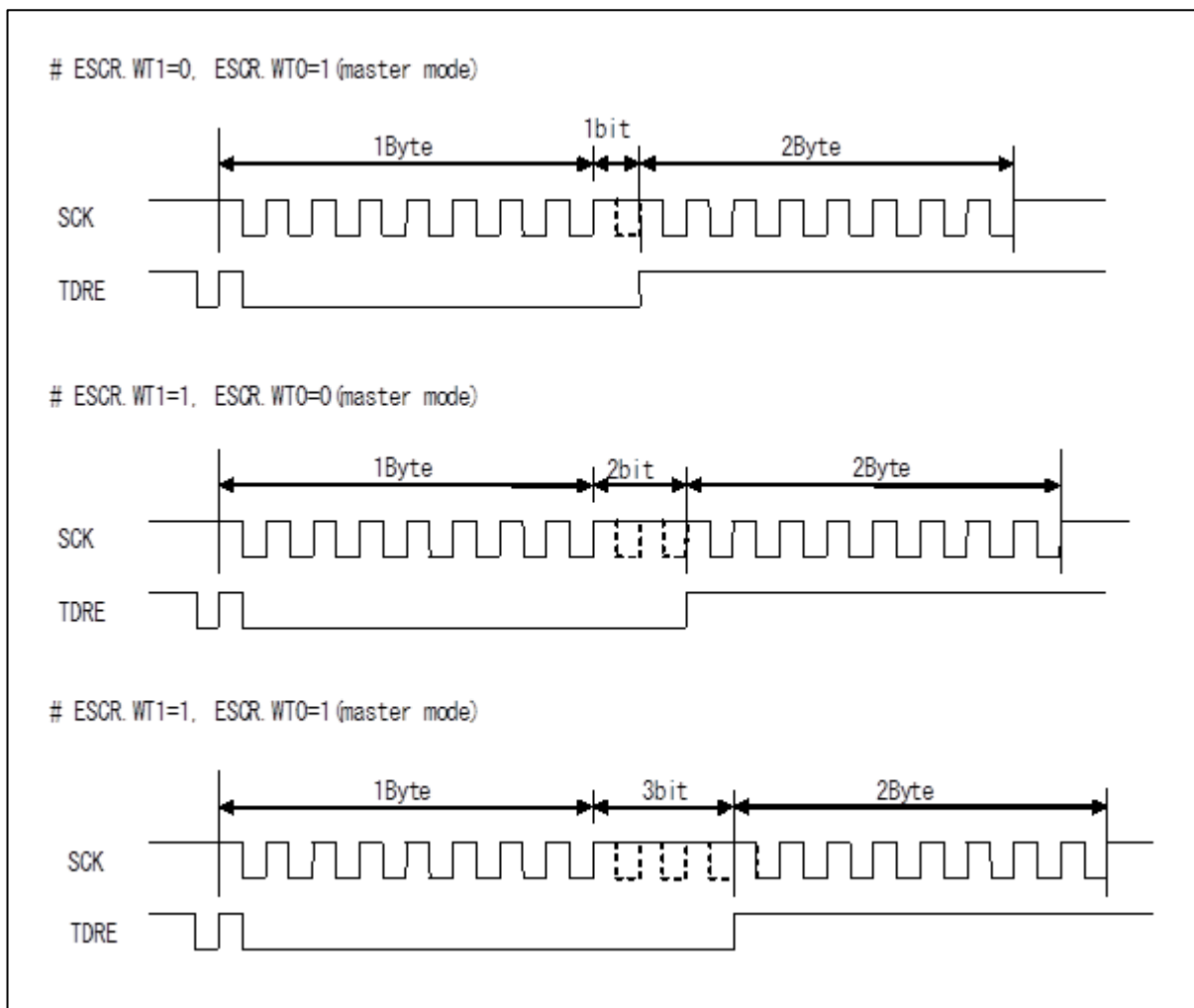
**c) Transmission/Reception Operations**

1. To perform transmission/reception operations at the same time, enable serial data output (SMR:SOE = 1) and enable transmission/reception operations (SCR:TXE, RXE=1).
2. When transmission data is written to TDR, SSR:TDRE is set to 0, and the first bit is output. Later, transmission data is output in synchronization with a rising edge of the serial clock (SCK) output. Half a cycle before a falling edge of the first serial clock, SSR:TDRE is set to 1, and if transmission interrupts are enabled (SCR:TIE=1), a transmission interrupt request is output. At this time, the second byte of transmission data can be written.
3. Reception data is sampled at a falling edge of the serial clock (SCK) output. When the last bit of reception data is received, SSR:RDRF is set to 1. If reception interrupts are enabled (SCR:RIE = 1), a reception interrupt request is output. At this time, reception data (RDR) can be read. When reception data is read, SSR:RDRF is cleared to "0".

**d) Continuous Data Transmission or Reception Wait Operation**

If settings other than (ESCR:WT1, ESCR:WT0) = (0, 0) are set for continuous data transmission or reception, a wait is inserted between the frames.

**Figure 3-10 Wait Operation**







### (5) Slave Operation (SCR:MS=1, SMR:SCKE=0, SCSCR:CSEN0=0)

#### a) Transmission Operation

1. Enabling serial data output (SMR:SOE=1) and also transmission (SCR:TXE=1) and then writing transmission data to TDR causes SSR:TDRE to be set to 0. Thus, the first bit is output. Later, transmission data is output in synchronization with a rising edge of the serial clock (SCK) output.
2. When the first 1 bit of transmission data is output, SSR:TDRE is set to 1. If transmission interrupts are enabled (SCR:TIE = 1), a transmission interrupt request is output. At this time, the second byte of transmission data can be written.

#### Note:

- After transmission is enabled (SCR:TXE=1), if the first writing of transmission data to TDR is performed when the serial clock (SCK) is not at the mark level, the first 1 bit of data is not output, and transmission is not performed normally. After transmission is enabled (SCR:TXE = 1), perform the first writing of transmission data to TDR with SSR:TBI = 1 when the serial clock (SCK) is at the mark level.

#### b) Reception Operation

1. If serial data output is disabled (SMR:SOE = 0) but reception is enabled (SCR:RXE = 1), reception data is sampled at a falling edge of the serial clock input (SCK).
2. When the last bit is received, SSR:RDRF is set to 1. If reception interrupts are enabled (SCR:RIE = 1), a reception interrupt request is output. At this time, reception data (RDR) can be read.
3. When reception data (RDR) is read, SSR:RDRF is cleared to "0".

#### c) Transmission/Reception Operations

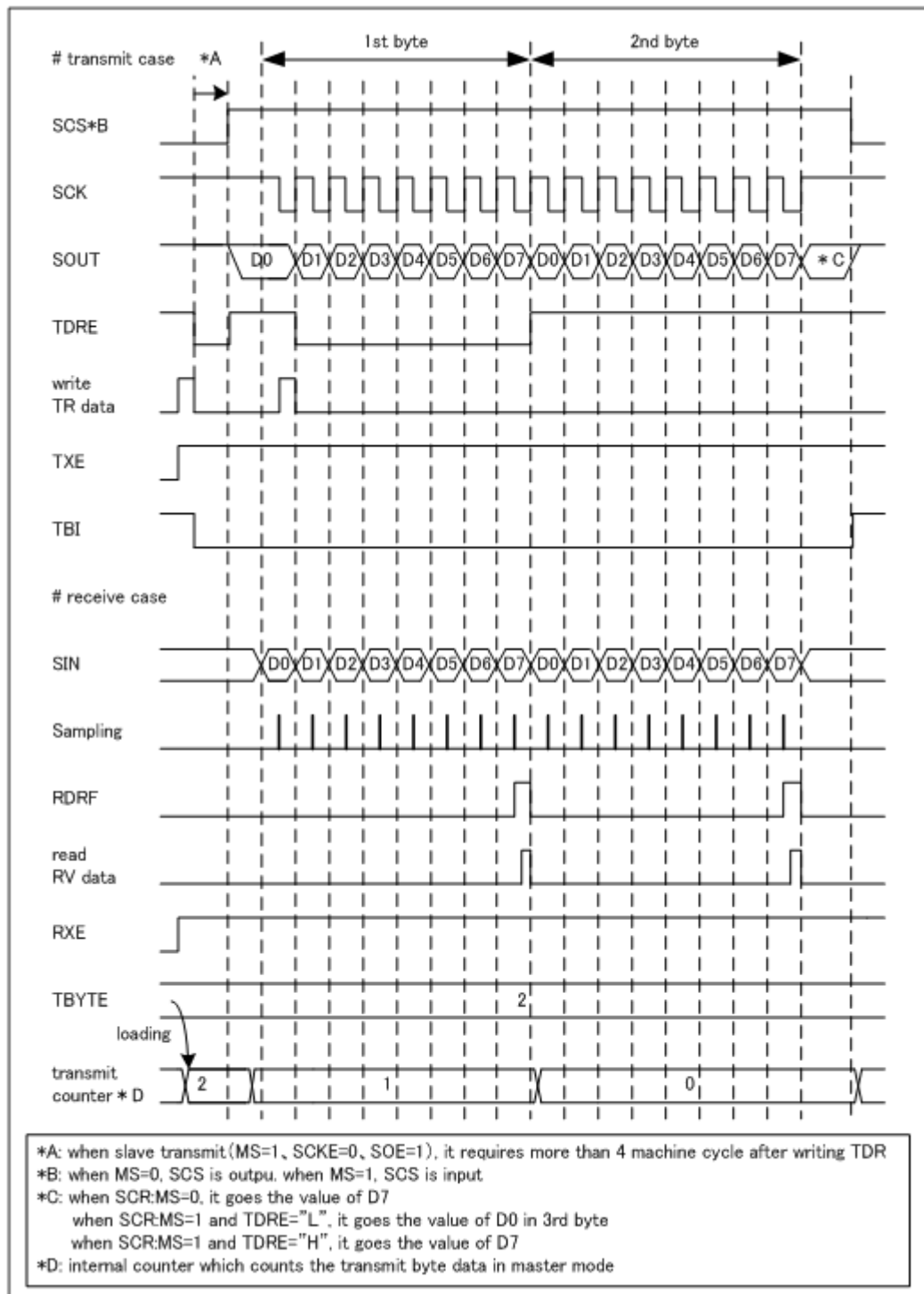
1. To perform transmission/reception at the same time, enable serial data output (SMR:SOE=1) and also transmission/reception (SCR:TXE, RXE=1).
2. When transmission data is written to TDR, SSR:TDRE is set to 0, and the first bit is output. Later, transmission data is output in synchronization with a rising edge of the serial clock (SCK) input. When the first 1 bit of transmission data is output, SSR:TDRE is set to 1, and if transmission interrupts are enabled (SCR:TIE = 1), a transmission interrupt request is output. At this time, the second byte of transmission data can be written.
3. Reception data is sampled at a falling edge of the serial clock (SCK) input. When the last bit of the reception data is received, SSR:RDRF is set to 1, and if reception interrupts are enabled (SCR:RIE = 1), a reception interrupt request is output. At this time, reception data (RDR) can be read. When reception data is read, SSR:RDRF is cleared to "0".

#### d) Continuous Switching from Reception Operation to Transmission Operation

1. Disable serial data output (SMR:SOE=0), enable reception interrupts (SCR:RIE=1) and reception operation (SCR:RXE = 1), and also transmission operation (SCR:TXE=1). If dummy data is written to TDR when the serial clock (SCK) is at the mark level, reception data is sampled at a falling edge of the serial clock input (SCK).
2. To continue reception, write dummy data to TDR between a reception interrupt request and the next rising edge of the serial clock (SCK).
3. To switch from reception to transmission, enable serial data output (SMR:SOE = 1), disable reception interrupts (SCR:RIE = 0), and also reception (SCR:RXE = 0) between a reception interrupt request and the next rising edge of the serial clock (SCK), and write transmission data to TDR; after the end of reception, transmission data is output in synchronization with a rising edge of the serial clock.

(6) SPI Transfer (I) Timing Chart (When the Serial Chip Select Pin is Used)

Figure 3-11 SPI Transfer (I) Timing Chart (When the Serial Chip Select Pin is Used)



**(7) Master Operation (SCR:MS=0, SMR:SCKE=1, SCSCR:CSOE=1, SCSCR:CSENn\*=1)**

\*: n is replaced by the serial chip select pin number used.

**a) Transmission Operation**

1. Enabling serial data output (SMR:SOE=1) and also transmission operation (SCR:TXE = 1), while disabling reception (SCR:RXE = 0) and then writing transmission data to TDR causes SSR:TDRE to be set to 0. Later, at the same time as the first bit is output, the serial chip select pin (SCS) becomes active, and after the elapse of the serial chip select pin setup time, serial clock output is started. After the start of serial clock output, transmission data is output in synchronization with a rising edge of the serial clock (SCK) output.
2. Half a cycle before a falling edge of the first serial clock (SCK) output, SSR:TDRE is set to 1, and if transmission interrupts are enabled (SCR:TIE=1), a transmission interrupt request is output. At this time, the second byte of transmission data can be written.
3. After the end of data transmission for which the count is set in TBYTE, serial clock output is stopped.
4. After the serial clock output is stopped, the serial chip select pin (SCS) becomes inactive after the elapse of the serial chip select pin hold time. At this time, however, if the serial chip select active level (SCSCR:SCAM=1) is retained, the serial chip select pin (SCS) retains the active state.

**b) Reception Operation**

1. Disabling serial data output (SMR:SOE=0), while enabling transmission (SCR:TXE=1) and also reception (SCR:RXE=1), and then writing dummy data to TDR causes the serial chip select pin (SCS) to become active, and after the elapse of the serial chip select pin setup time, serial clock output is started. After serial clock output starts, reception data is sampled at a falling edge of the serial clock (SCK) output.
2. When the last bit is received, SSR:RDRF is set to 1, and if reception interrupts are enabled (SCR:RIE=1), a reception interrupt request is output.
3. At this time, reception data (RDR) can be read.
4. When reception data (RDR) is read, SSR:RDRF is cleared to "0".
5. After the end of data reception for which the count is set in TBYTE, serial clock output is stopped.
6. After serial clock output is stopped, the serial chip select pin (SCS) becomes inactive after the elapse of the serial chip select pin hold time. At this time, however, if the serial chip select active level (SCSCR:SCAM=1) is retained, the serial chip select pin (SCS) retains the active state.

**Notes:**

- When performing reception only, write dummy data to TDR to output the serial clock (SCK).
- When the transmission/reception FIFOs are enabled, setting the number of frames to transfer in the FBYTE register causes as many serial clock (SCK) pulses as the number of frames to be output.

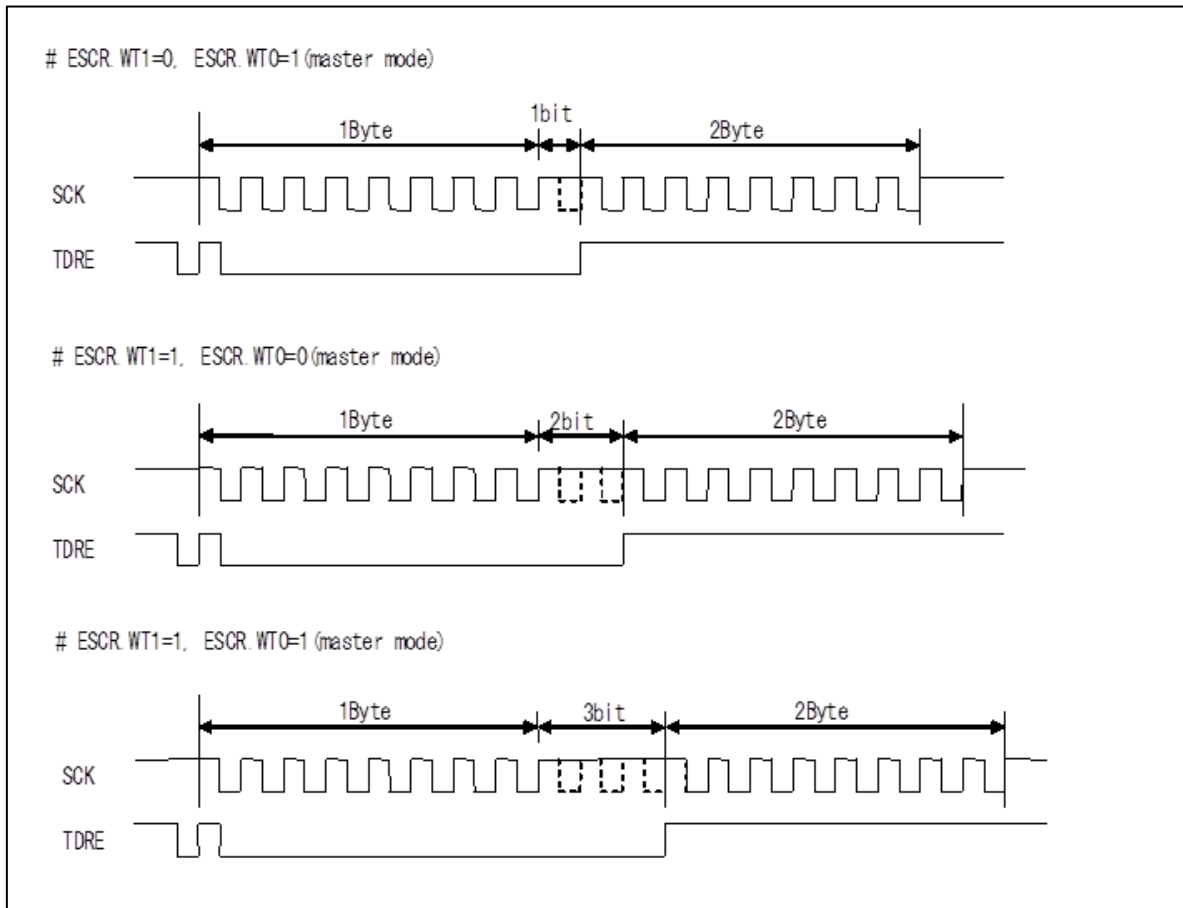
**c) Transmission/Reception Operations**

1. To perform transmission/reception at the same time, enable serial data output (SMR:SOE=1) and also transmission/reception (SCR:TXE, RXE=1).
2. When transmission data is written to TDR, SSR:TDRE is set to 0. Later, at the same time as the first bit is output, the serial chip select pin (SCS) becomes active, and after the elapse of the serial chip select pin setup time, serial clock output is started. After the start of serial clock output, transmission data is output in synchronization with a rising edge of the serial clock (SCK) output. Half a cycle before a falling edge of the first serial clock, SSR:TDRE is set to 1, and if transmission interrupts are enabled (SCR:TIE=1), a transmission interrupt request is output. At this time, the second byte of transmission data can be written.
3. Reception data is sampled at a falling edge of the serial clock (SCK) output. When the last bit of reception data is received, SSR:RDRF is set to 1, and if reception interrupts are enabled (SCR:RIE=1), a reception interrupt request is output. At this time, reception data (RDR) can be read. When reception data is read, SSR:RDRF is cleared to "0".
4. After the end of data transmission/reception for which the count is set in TBYTE, serial clock output is stopped.
5. After serial clock output is stopped, the serial chip select pin (SCS) becomes inactive after the elapse of the serial chip select pin hold time. At this time, however, if the serial chip select active level (SCSCR:SCAM = 1) is retained, the serial chip select pin (SCS) retains the active state.

**d) Continuous Data Transmission or Reception Wait Operation**

If settings other than (ESCR:WT1, ESCR:WT0) = (0, 0) are set for continuous data transmission or reception, a wait is inserted between the frames.

**Figure 3-12 Wait Operation**



**(8) Slave Operation (SCR:MS=1, SMR:SCKE=0, SCSCR:CSEN = 1, SCSCR:SCAM = 0)**

**a) Transmission Operation**

1. Enabling serial data output (SMR:SOE=1) and also transmission (SCR:TXE=1) and then writing transmission data to TDR causes SSR:TDRE to be set to 0.
2. If the serial chip select pin (SCS) becomes active, transmission is started, and the first bit is output. After the start of transmission, transmission data is output in synchronization with a rising edge of the serial clock (SCK) output.
3. When the first 1 bit of transmission data is output, SSR:TDRE is set to 1, and if transmission interrupts are enabled (SCR:TIE=1), a transmission interrupt request is output. At this time, the second byte of transmission data can be written.
4. If the serial chip select pin (SCS) becomes inactive, transmission is stopped, and the serial output pin (SOUT) becomes "H".

**Note:**

- After transmission is enabled (SCR:TXE = 1), if the first writing of transmission data to TDR is performed when the serial clock (SCK) is not at the mark level, the first 1 bit of data is not output, and transmission is not performed normally. After transmission is enabled (SCR:TXE=1), perform the first writing of transmission data to TDR when the serial clock (SCK) is at the mark level.

**b) Reception Operation**

1. If serial data output is disabled (SMR:SOE = 0) and reception is enabled (SCR:RXE = 1), if the serial chip select pin (SCS) becomes active, reception starts, and reception data is sampled at a falling edge of the serial clock input (SCK).
2. If the last bit is received, SSR:RDRF is set to 1, and if reception interrupts are enabled (SCR:RIE = 1), a reception interrupt request is output.
3. At this time, reception data (RDR) can be read.
4. When reception data (RDR) is read, SSR:RDRF is cleared to "0".
5. If the serial chip select pin (SCS) becomes inactive, reception is stopped.

**c) Transmission/Reception Operations**

1. To perform transmission/reception at the same time, enable serial data output (SMR:SOE=1) and also transmission/reception (SCR:TXE, RXE=1).
2. When transmission data is written to TDR, SSR:TDRE is set to 0. If the serial chip select pin (SCS) becomes active, transmission/reception starts, and the first bit is output. After the start of transmission/reception, transmission data is output in synchronization with a rising edge of the serial clock (SCK) input. When the first 1 bit of transmission data is output, SSR:TDRE is set to 1, and if transmission interrupts are enabled (SCR:TIE=1), a transmission interrupt request is output. At this time, the second byte of transmission data can be written.
3. Reception data is sampled at a falling edge of the serial clock (SCK) input. When the last bit of reception data is received, SSR:RDRF is set to 1, and if reception interrupts are enabled (SCR:RIE=1), a reception interrupt request is output. At this time, reception data (RDR) can be read. When reception data is read, SSR:RDRF is cleared to "0".
4. If the serial chip select pin (SCS) becomes inactive, transmission/reception stops, and the serial output pin (SOUT) becomes "H".



### 3.4. SPI Transfer (II)

#### (1) Features

	Item	Description
1	Serial clock (SCK) mark level	"L"
2	Transmission data output timing	SCK falling edge
3	Reception data sampling	SCK rising edge
4	Data length	5 to 16, 20, 24 and 32 bits

#### (2) Register Setting

The register settings necessary for SPI transfer (II) are listed below.

SCR:SPI<sup>\*1</sup>=1, SMR:MD2 to 0=0b010, SCINV<sup>\*1</sup>=1

During master operation: SCR:MS=0, SMR:SCKE=1

During slave operation: SCR:MS=1, SMR:SCKE=0

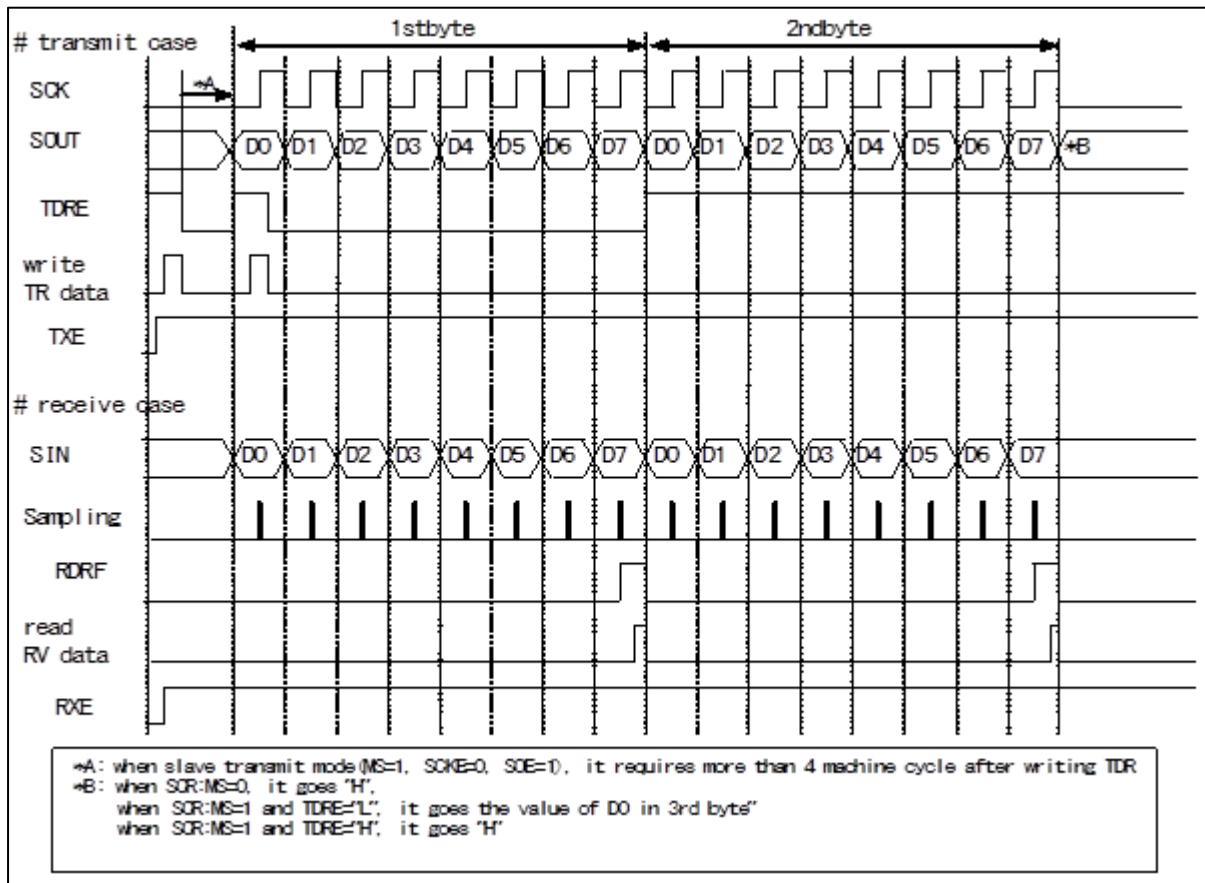
\*1: The bit to be set differs depending on the condition. See Table 5-2.

**Note:**

- Except for those described above, set the registers according to their use.

### (3) SPI Transfer (II) Timing Chart (When the Serial Chip Select Pin is Not Used)

Figure 3-13 SPI Transfer (II) Timing Chart (When the Serial Chip Select Pin is Not Used)







#### (4) Master Operation (Setting to SCR:MS=0, SMR:SCKE=1, SCSCR:CSEN3 to 0=0b0000)

##### a) Transmission Operation

1. Enabling serial data output (SMR:SOE=1) and transmission (SCR:TXE = 1), while disabling reception (SCR:RXE = 0) and then writing transmission data to TDR causes SSR:TDRE to be set to 0. With this, transmission data is output in synchronization with a falling edge of the serial clock (SCK) output.
2. Half a cycle before a rising edge of the first serial clock (SCK) output, SSR:TDRE is set to 1. Thus, if transmission interrupts are enabled (SCR:TIE=1), a transmission interrupt request is output. At this time, the second byte of transmission data can be written.

##### b) Reception Operation

1. If serial data output is disabled (SMR:SOE=0), while transmission (SCR:TXE = 1) and reception (SCR:RXE = 1) are enabled, writing dummy data to TDR causes reception data to be sampled at a rising edge of the serial clock output (SCK).
2. When the last bit is received, SSR:RDRF is set to 1. At this time, if reception interrupts are enabled (SCR:RIE = 1), a reception interrupt request is output. At this time, reception data (RDR) can be read.
3. When reception data (RDR) is read, SSR:RDRF is cleared to "0".

##### Notes:

- When performing reception only, write dummy data to TDR to output the serial clock (SCK).
- When the transmission/reception FIFOs are enabled, setting the number of frames to transfer in the FBYTE register causes as many serial clock (SCK) pulses as the number of frames to be output.

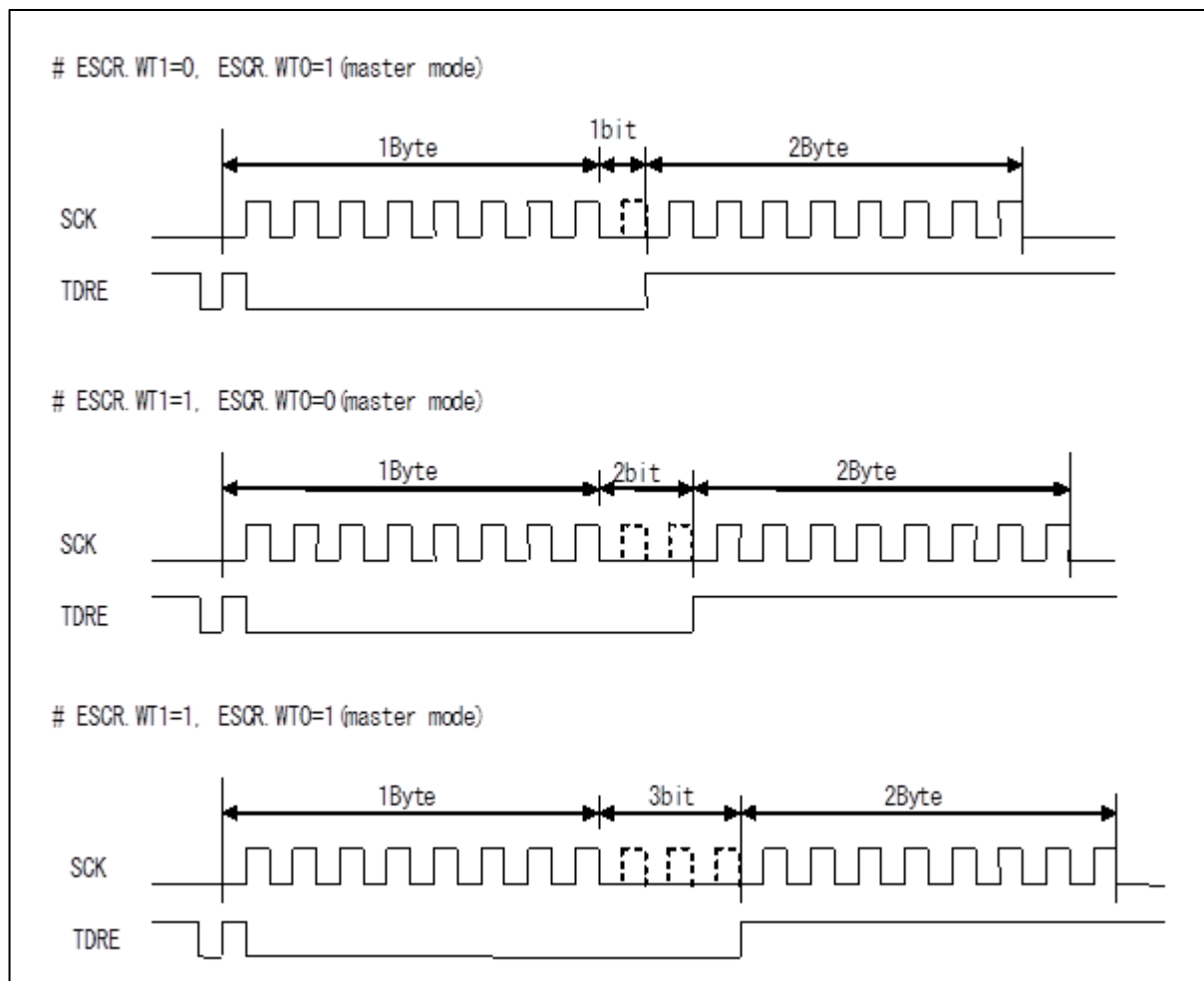
##### c) Transmission/Reception Operations

1. To perform transmission/reception at the same time, enable serial data output (SMR:SOE=1) and also transmission/reception (SCR:TXE, RXE=1).
2. When transmission data is written to TDR, SSR:TDRE is set to 0, and the first bit is output. Later, transmission data is output in synchronization with a falling edge of the serial clock (SCK) output. Half a cycle before a rising edge of the first serial clock, SSR:TDRE is set to 1, and if transmission interrupts are enabled (SCR:TIE=1), a transmission interrupt request is output. At this time, the second byte of transmission data can be written.
3. Reception data is sampled at a rising edge of the serial clock (SCK) output. When the last bit of reception data is received, SSR:RDRF is set to 1. If reception interrupts are enabled (SCR:RIE=1), a reception interrupt request is output. At this time, reception data (RDR) can be read. When reception data is read, SSR:RDRF is cleared to "0".

**d) Continuous Data Transmission or Reception Wait Operation**

If settings other than (ESCR:WT1, ESCR:WT0) = (0, 0) are set for continuous data transmission or reception, a wait is inserted between the frames.

**Figure 3-14 Wait Operation**





### (5) Slave Operation (Setting to SCR:MS=1, SMR:SCKE=0, SCSCR:CSEN0=0)

#### a) Transmission Operation

1. Enabling serial data output (SMR:SOE=1) and transmission (SCR:TXE = 1) and then writing transmission data to TDR causes SSR:TDRE to be set to 0. Thus, the first bit is output. Later, transmission data is output in synchronization with a falling edge of the serial clock (SCK) input.
2. When the first 1 bit of transmission data is output, SSR:TDRE is set to 1. If transmission interrupts are enabled (SCR:TIE=1), a transmission interrupt request is output. At this time, the second byte of transmission data can be written.

#### Note:

- After transmission is enabled (SCR:TXE=1), if the first writing of transmission data to TDR is performed when the serial clock (SCK) is not at the mark level, the first 1 bit of data is not output, and transmission is not performed normally. After transmission is enabled (SCR:TXE = 1), perform the first writing of transmission data to TDR with SSR:TBI=1 when the serial clock (SCK) is at the mark level.

#### b) Reception Operation

1. If serial data output is disabled (SMR:SOE=0) and reception operation is enabled (SCR:RXE=1), reception data is sampled at a rising edge of the serial clock input (SCK).
2. When the last bit is received, SSR:RDRF is set to 1. If reception interrupts are enabled (SCR:RIE=1), a reception interrupt request is output.  
At this time, reception data (RDR) can be read.
3. When reception data (RDR) is read, SSR:RDRF is cleared to "0".

#### c) Transmission/Reception Operations

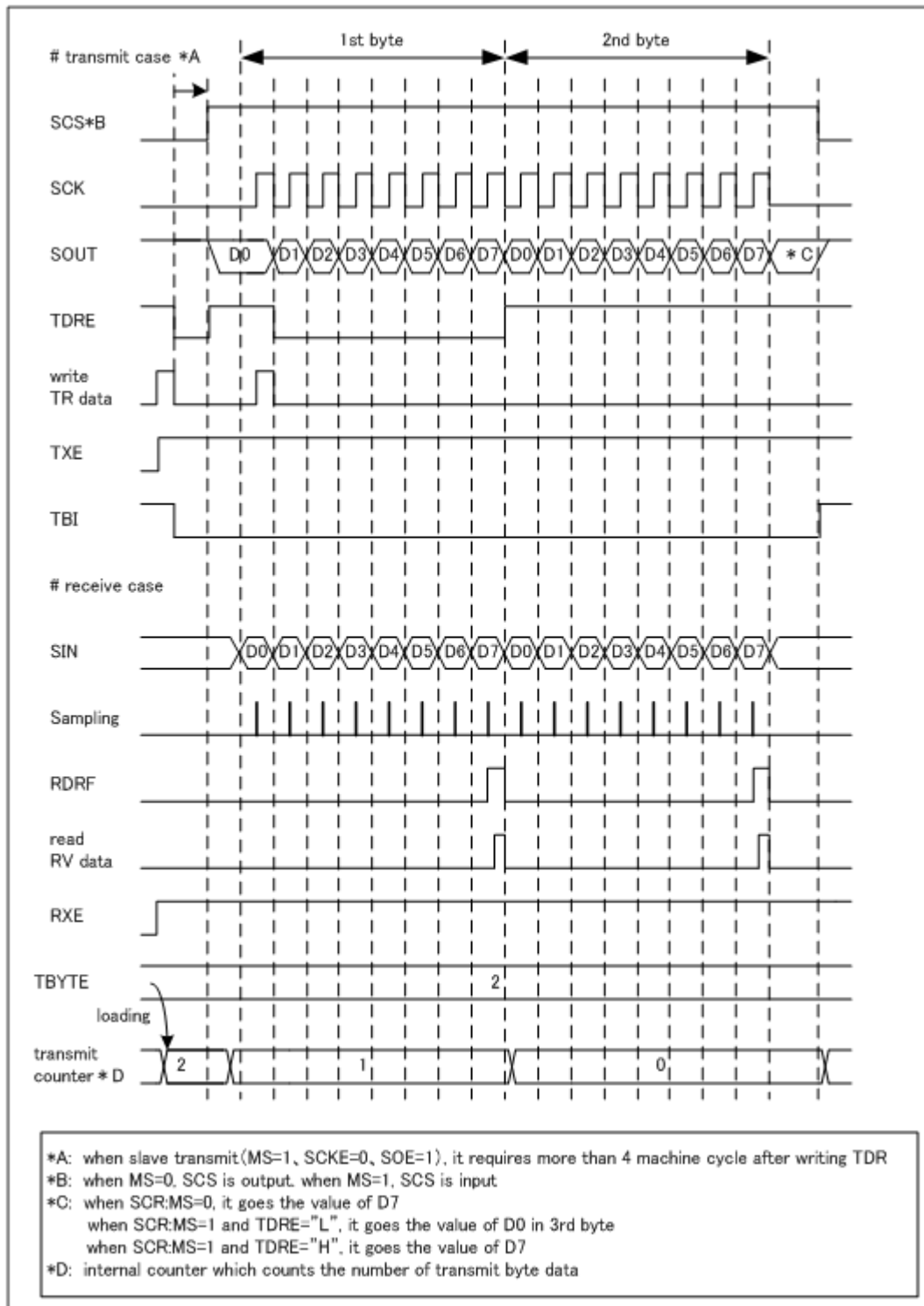
1. To perform transmission/reception at the same time, enable serial data output (SMR:SOE=1) and also transmission/reception (SCR:TXE, RXE=1).
2. When transmission data is written to TDR, SSR:TDRE is set to 0, and the first bit is output. Later, transmission data is output in synchronization with a falling edge of the serial clock (SCK) input. When the first 1 bit of transmission data is output, SSR:TDRE is set to 1, and if transmission interrupts are enabled (SCR:TIE=1), a transmission interrupt request is output. At this time, the second byte of transmission data can be written.
3. Reception data is sampled at a rising edge of the serial clock (SCK) input. If the last bit of the reception data is received, SSR:RDRF is set to 1, and if reception interrupts are enabled (SCR:RIE=1), a reception interrupt request is output. At this time, reception data (RDR) can be read. When reception data is read, SSR:RDRF is cleared to "0".

#### d) Continuous Switching from Reception Operation to Transmission Operation

1. Disable serial data output (SMR:SOE=0), enable reception interrupts (SCR:RIE=1), enable reception (SCR:RXE=1) and also transmission (SCR:TXE=1). If dummy data is written to TDR when the serial clock (SCK) is at the mark level, reception data is sampled at a falling edge of the serial clock input (SCK).
2. To continue reception, write dummy data to TDR between a reception interrupt request and the next rising edge of the serial clock (SCK).
3. To switch from reception to transmission, enable serial data output (SMR:SOE=1), disable reception interrupts (SCR:RIE=0), and disable reception (SCR:RXE=0) between a reception interrupt request and the next rising edge of the serial clock (SCK), and write transmission data to TDR; after the end of reception, transmission data is output in synchronization with a rising edge of the serial clock.

### (6) SPI Transfer (II) Timing Chart (When the Serial Chip Select Pin is Used)

Figure 3-15 SPI Transfer (II) Timing Chart (When the Serial Chip Select Pin is Used)



**(7) Master Operation (SCR:MS=0, SMR:SCKE=1, SCSCR:CSOE=1, SCSCR:CSENn\*=1)**

\*: n is replaced by the serial chip select pin number used.

**a) Transmission Operation**

1. Enabling serial data output (SMR:SOE=1) and transmission (SCR:TXE=1), while disabling reception (SCR:RXE = 0) and then writing transmission data to TDR causes SSR:TDRE to be set to 0. Later, at the same time as the output of the first bit, the serial chip select pin (SCS) becomes active, and after the elapse of the serial chip select pin setup time, serial clock output is started. After the start of serial clock output, transmission data is output in synchronization with a falling edge of the serial clock (SCK) output.
2. Half a cycle before a falling edge of the first serial clock (SCK) output, SSR:TDRE is set to 1, and if transmission interrupts are enabled (SCR:TIE=1), a transmission interrupt request is output. At this time, the second byte of transmission data can be written.
3. After the end of data transmission for which the count is set in TBYTE, serial clock output is stopped.
4. After the serial clock output is stopped, the serial chip select pin (SCS) becomes inactive after the elapse of the serial chip select pin hold time. At this time, however, if the serial chip select active level (SCSCR:SCAM=1) is retained, the serial chip select pin (SCS) retains the active state.

**b) Reception Operation**

1. Disabling serial data output (SMR:SOE=0), while enabling transmission (SCR:TXE=1) and also reception (SCR:RXE=1) and then writing dummy data to TDR causes the serial chip select pin (SCS) to become active, and after the elapse of the serial chip select pin setup time, serial clock output is started. After serial clock output is started, reception data is sampled at a rising edge of the serial clock (SCK) output.
2. When the last bit is received, SSR:RDRF is set to 1, and if reception interrupts are enabled (SCR:RIE=1), a reception interrupt request is output.
3. At this time, reception data (RDR) can be read.
4. When reception data (RDR) is read, SSR:RDRF is cleared to "0".
5. After the end of data reception for which the count is set in TBYTE, the serial clock output is stopped.
6. After the serial clock output is stopped, the serial chip select pin (SCS) becomes inactive after the elapse of the serial chip select pin hold time. At this time, however, if the serial chip select active level (SCSCR:SCAM=1) is retained, the serial chip select pin (SCS) retains the active state.

**Notes:**

- When performing only reception, write dummy data to TDR to cause the serial clock (SCK) to be output.
- When the transmission/reception FIFOs are enabled, setting the number of frames to transfer in the FBYTE register causes as many serial clock (SCK) pulses as the number of frames to be output.

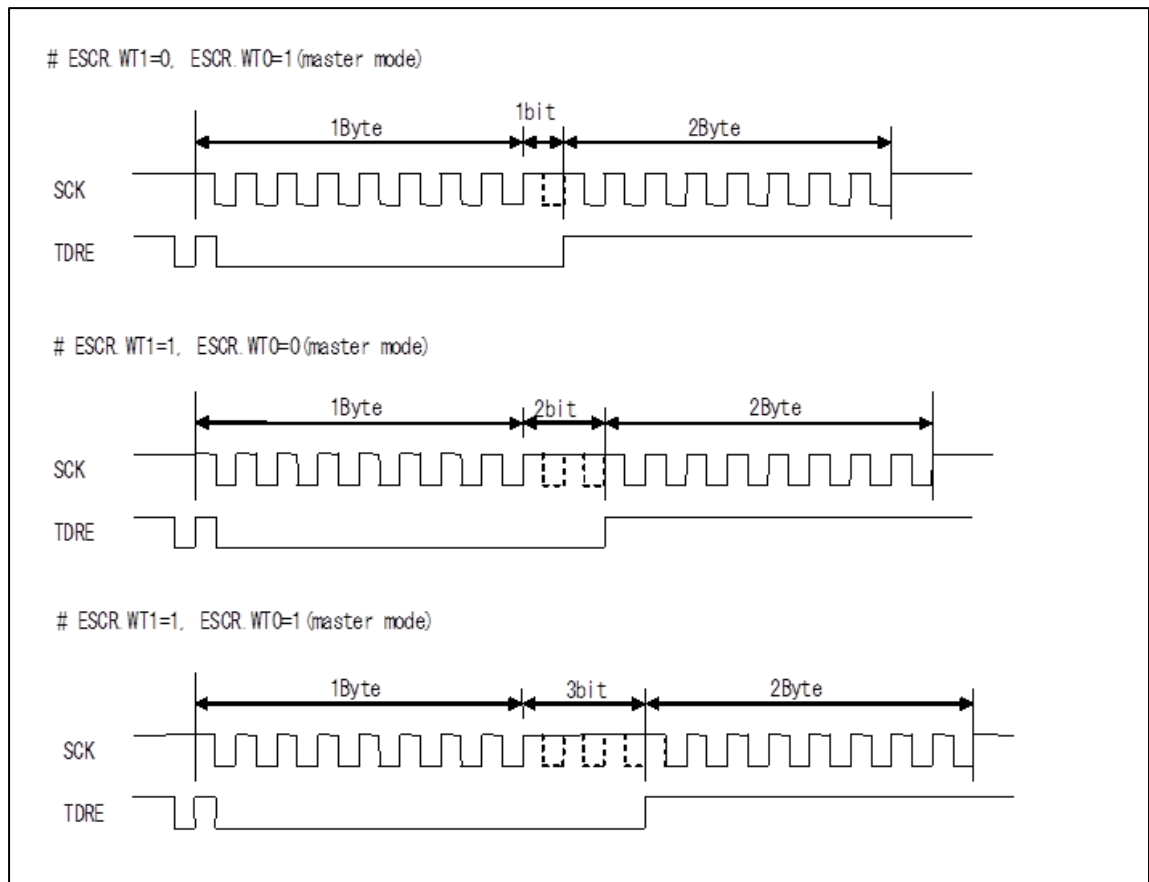
**c) Transmission/Reception Operations**

1. To perform transmission/reception at the same time, enable both serial data output (SMR:SOE = 1) and also transmission/reception (SCR:TXE, RXE=1).
2. When transmission data is written to TDR, SSR:TDRE is set to 0. Later, at the same time as the output of the first bit, the serial chip select pin (SCS) becomes active, and after the elapse of the serial chip select pin setup time, serial clock output is started. After the start of serial clock output, transmission data is output in synchronization with a falling edge of the serial clock (SCK) output. Half a cycle before a rising edge of the first serial clock, SSR:TDRE is set to 1, and if transmission interrupts are enabled (SCR:TIE=1), a transmission interrupt request is output. At this time, the second byte of transmission data can be written.
3. Reception data is sampled at a rising edge of the serial clock (SCK) output. When the last bit of the reception data is received, SSR:RDRF is set to 1, and if reception interrupts are enabled (SCR:RIE=1), a reception interrupt request is output. At this time, reception data (RDR) can be read. When the reception data is read, SSR:RDRF is cleared to "0".
4. After the end of data transmission/reception for which the count is set in TBYTE, the serial clock output is stopped.
5. After serial clock output is stopped, the serial chip select pin (SCS) becomes inactive after the elapse of the serial chip select pin hold time. At this time, however, if the serial chip select active level (SCSCR:SCAM = 1) is retained, the serial chip select pin (SCS) retains the active state.

**d) Continuous Data Transmission or Reception Wait Operation**

If settings other than (ESCR:WT1, ESCR:WT0) = (0, 0) are set for continuous data transmission or reception, a wait is inserted between the frames.

**Figure 3-16 SPI Wait Operation**



**(8) Slave Operation (SCR:MS=1, SMR:SCKE=0, SCSCR:CSEN=1, SCSCR:SCAM=0)**

**a) Transmission Operation**

1. Enabling both serial data output (SMR:SOE=1) and transmission (SCR:TXE=1) and then writing transmission data to TDR causes SSR:TDRE to be set to 0.
2. If the serial chip select pin (SCS) becomes active, transmission operation is started, and the first bit is output. After the start of transmission, transmission data is output in synchronization with a falling edge of the serial clock (SCK) output.
3. When the first 1 bit of transmission data is output, SSR:TDRE is set to 1, and if transmission interrupts are enabled (SCR:TIE=1), a transmission interrupt request is output. At this time, the second byte of transmission data can be written.
4. If the serial chip select pin (SCS) becomes inactive, transmission operation is ended, and the serial output pin (SOUT) becomes "H".

**Note:**

- After transmission is enabled (SCR:TXE=1), if the first writing of transmission data to TDR is performed when the serial clock (SCK) is not at the mark level, the first 1 bit of data is not output, and transmission is not performed normally. After transmission is enabled (SCR:TXE=1), perform the first writing of transmission data to TDR when the serial clock (SCK) is at the mark level.

**b) Reception Operation**

1. If serial data output is disabled (SMR:SOE=0) but reception is enabled (SCR:RXE=1), and if the serial chip select pin (SCS) becomes active, reception operation starts, and reception data is sampled at a rising edge of the serial clock input (SCK).
2. When the last bit is received, SSR:RDRF is set to 1, and if reception interrupts are enabled (SCR:RIE = 1), a reception interrupt request is output.
3. At this time, reception data (RDR) can be read.
4. When reception data (RDR) is read, SSR:RDRF is cleared to "0".
5. If the serial chip select pin (SCS) becomes inactive, reception is stopped.

**c) Transmission/Reception Operations**

1. To perform transmission/reception at the same time, enable both serial data output (SMR:SOE = 1) and also transmission/reception (SCR:TXE, RXE=1).
2. When transmission data is written to TDR, SSR:TDRE is set to 0. If the serial chip select pin (SCS) becomes active, transmission/reception starts, and the first bit is output. After the start of transmission, transmission data is output in synchronization with a falling edge of the serial clock (SCK) input. When the first 1 bit of transmission data is output, SSR:TDRE is set to 1, and if transmission interrupts are enabled (SCR:TIE=1), a transmission interrupt request is output. At this time, the second byte of transmission data can be written.
3. Reception data is sampled at a rising edge of the serial clock (SCK) input. If the last bit of the reception data is received, SSR:RDRF is set to 1, and if reception interrupts are enabled (SCR:RIE=1), a reception interrupt request is output. At this time, reception data (RDR) can be read. When reception data is read, SSR:RDRF is cleared to "0".
4. If the serial chip select pin (SCS) becomes inactive, transmission/reception operations are stopped, and the serial output pin (SOUT) becomes "H".





## 4. Serial Timer Operation

The serial timer can be used as either a timer function or as a synchronous transmission function.

### Serial Timer Operation

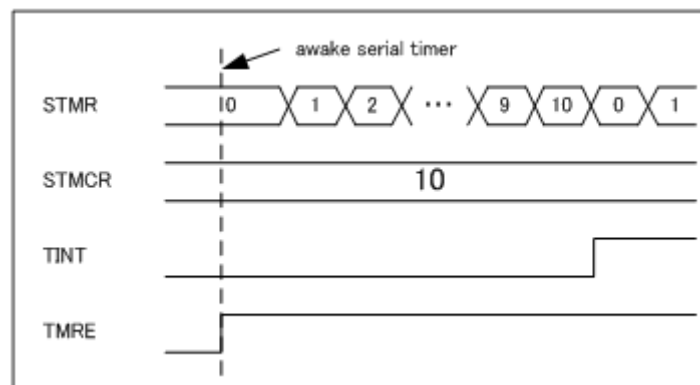
#### a) Activating the Serial Timer

The serial timer can be activated by setting the serial timer enable bit (SACSR:TMRE) to "1".

- Activation by the serial timer enable bit (SACSR:TMRE)

If the serial timer enable bit (SACSR:TMRE) is set to "1", the serial timer is activated, and the serial timer register (STMR) starts counting from 0.

Figure 4-1 Activation by Serial Timer Enable Bit (STMCR=10,TSYNE=0)





**b) Stopping the Serial Timer**

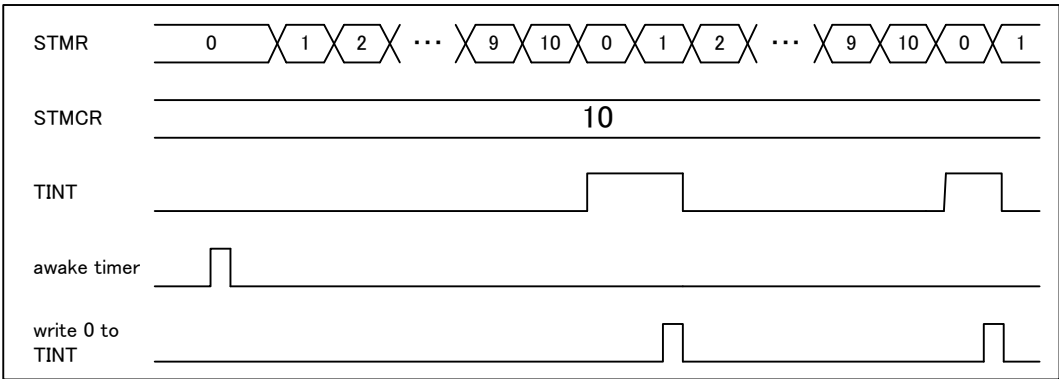
The serial timer stops when the serial timer enable bit (SACSR:TMRE) is set to "0". At this time, the value of the serial timer register (STMR) is retained.

**c) Timer Operation**

When the synchronous transmission enable bit (SACSR:TSYNE) is "0", the serial timer operates as a timer.

The timer interrupt flag (SACSR:TINT) is set to "1" and the serial timer register (STMR) is reset to 0 when the serial timer register (STMR) coincides with the serial timer comparison register (STMCR).

**Figure 4-2 Timer Operation (STMCR=10, SACSR:TSYNE=0)**



**Figure 4-3 Flow Chart of Serial Timer Initialization**

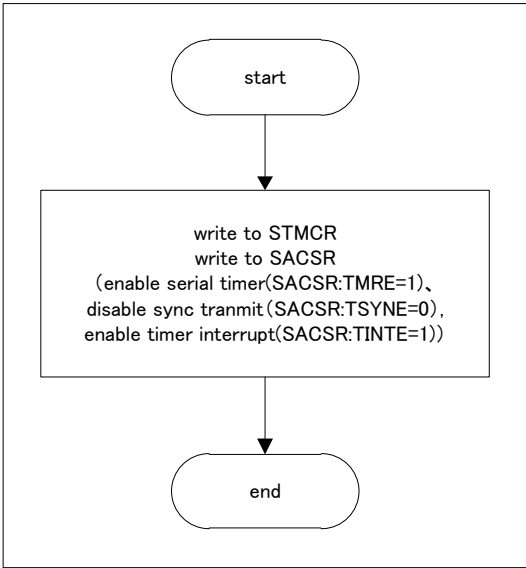
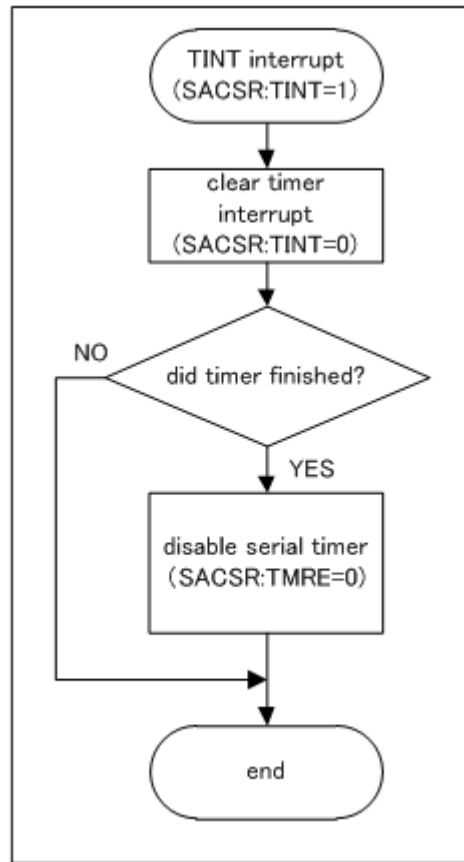


Figure 4-4 Flow Chart of Serial Timer Interrupt Processing



**Note:**

- In the state in which synchronous transmission is disabled (SACSR:TSYNE="0") while (0000) is set in the timer comparison register (STMCR), if the timer is operating and the division value (SACSR:TDIV3-0) of the timer operation clock is set to "0000", the timer interrupt flag (SACSR:TINT) is fixed to "1".

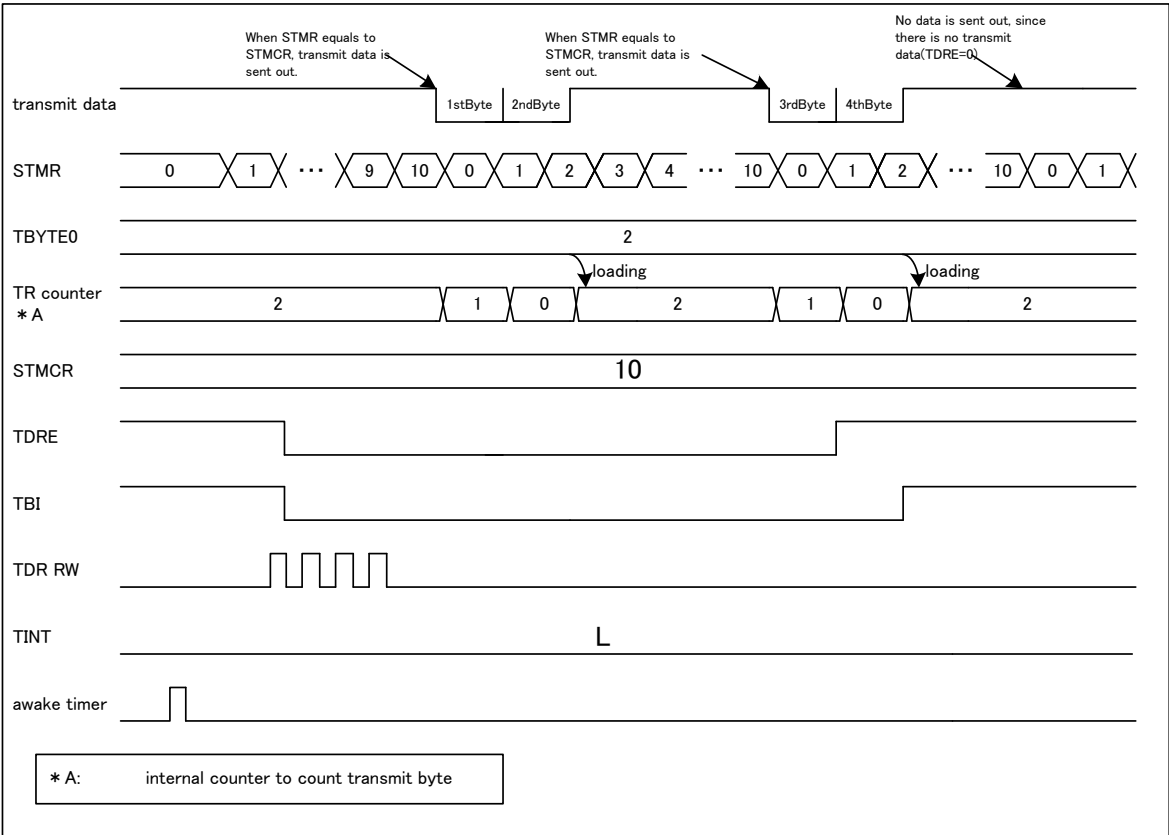
**d) Transmission Operation Synchronized with the Timer**

The serial timer is used for synchronous transmission when the synchronous transmission enable bit (SACSR:TSYNE) is set to "1".

Timer-synchronized transmission operates in the following manner:

1. If the transmission data register contains data (SSR:TDRE="0") and if the contents of the serial timer register (STMR) and the serial timer comparison register (STMCR) match, transmission operation starts, and the serial timer register (STMR) is reset to 0. Data transmission is performed for the data number specified in TBYTE0.
2. After transmitting the data number as specified by TBYTE0, transmission is suspended until the next time the serial timer register (STMR) and serial timer comparison register (STMCR) coincide.

**Figure 4-5 Transmission Operation Synchronized with the Timer (STMR=10, TBYTE0=2, SACSR:TSYNE=1)**





Even if synchronous transmission is enabled (SACSR:TSYNE="1") and the serial timer register (STMR) and the serial timer comparison register (STMCR) coincide, transmission is not activated under the following conditions:

- Transmission is disabled (SCR:TXE=0).
- In slave mode (SCR:MS=1)
- When a chip select error (SACSR:CSE=1) occurs
- The transmission data register does not contain valid data (SSR:TDRE=1).

However, writing transmission data to the transmission data register instantly starts transmission when all of the following are satisfied: 1. The transmission data register does not contain valid data (SSR:TDRE=1). 2. Synchronous transmission is enabled (SACSR:TSYNE="1"). 3. The serial timer register (STMR) and serial timer comparison register (STMCR) coincide.

If the transmission data register (TDR) contains valid data (SSR:TDRE=0) after transmission of the data number specified by TBYTE, the data is not transmitted until the next time the serial timer register (STMR) and the serial timer comparison register (STMCR) coincide.

If, however, synchronous transmission is enabled (SACSR:TSYNE="1"), transmission is in progress (SSR:TBI=0), and the serial timer register (STMR) and the serial timer comparison register (STMCR) match, transmission is reserved. When transmission is reserved, transmission does not stop after number of transmission specified by TBYTE0, and any subsequent transmission is performed.

Transmission reservation is released under one of the conditions below.

- Programmable reset (SCR:UPCL=1)
- Transmission disabled (SCR:TXE=0)
- Chip select error (SACSR:CSE=1)

To perform synchronous reception, disable serial data output (SMR:SOE=0), enable transmission (SCR:TXE=1) and also reception (SCR:RXE=1) and then write dummy data to TDR for the reception count.

**Figure 4-6 Flow Chart of Initialization of Transmission Synchronized with Timer**

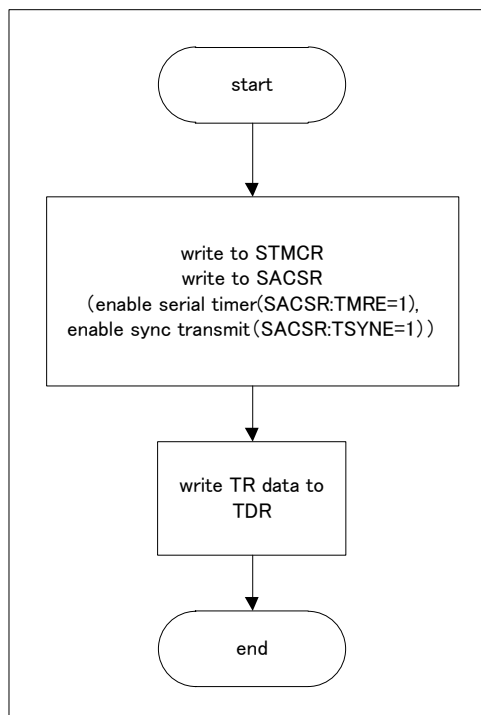
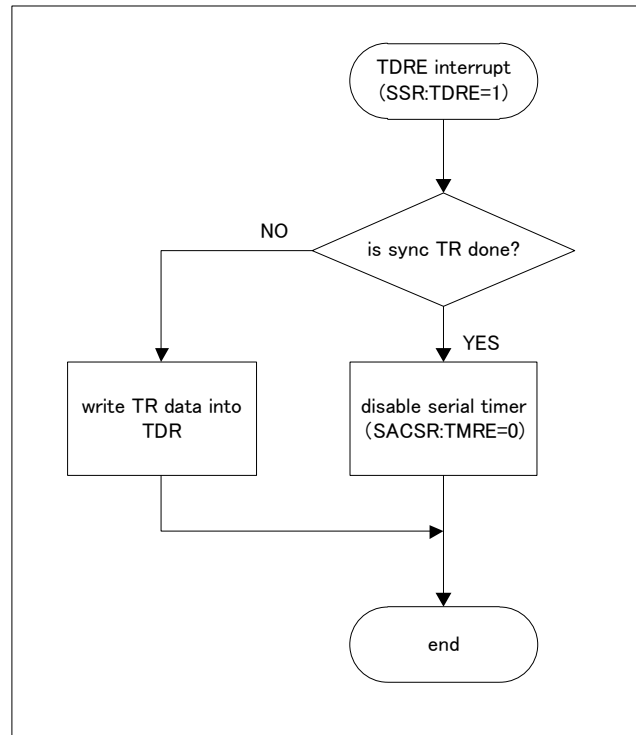


Figure 4-7 Flow Chart of Interrupt Processing in Transmission Synchronized with Timer



**Notes:**

- If the transmission data register (TDR) contains no valid transmission data (SSR:TDRE=1) before the transmission of as many data frames as the setting made for TBYTE, the following operation is performed.
- If transfer byte errors are enabled (TBEEN=1), a chip select error (SACSR:CSE=1) occurs. If "1" is set in the chip select error flag (SACSR:CSE), transmission is not started even if transmission data is written to the transmission data register (TDR).
- If transfer byte errors are disabled (TBEEN=0), transmission is stopped until transmission data is written to the transmission data register (TDR). Writing transmission data to the transmission data register (TDR) restarts the transmission operation.



## 5. Operation of Serial Chip Select

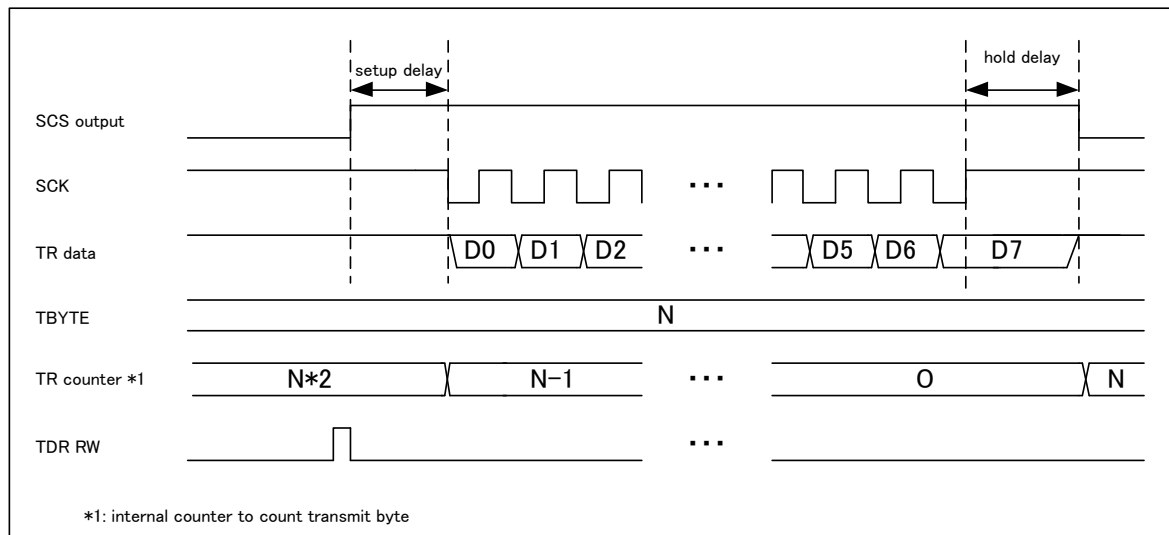
This section explains serial chip select operation.

### a) Operation in Master Mode (SCR:MS=0)

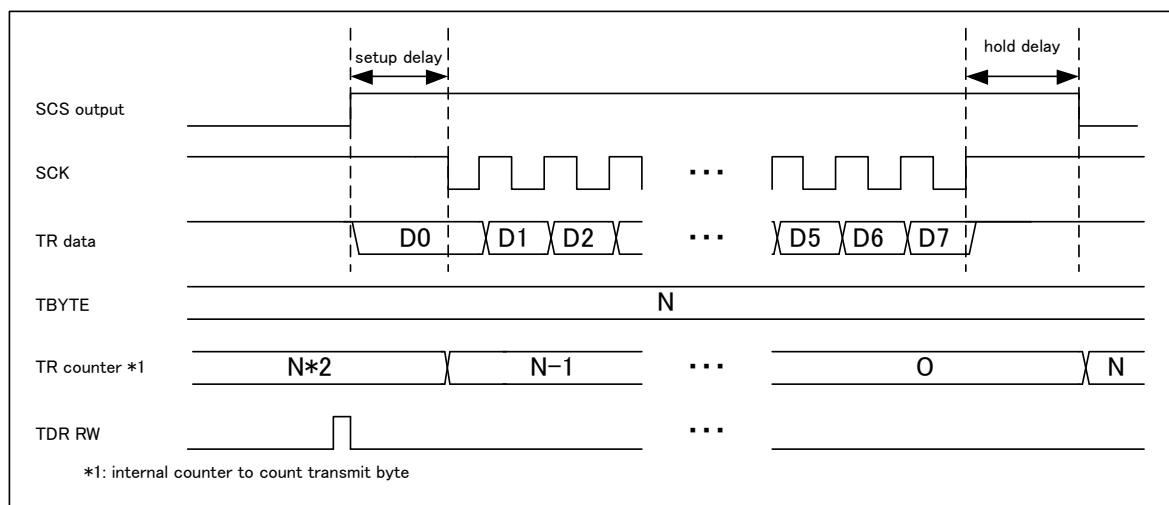
In master mode (SCR:MS=0), the serial chip select pin operates as described below.

1. If you enable both serial chip select (SCSCR:CSENn="1") and transmission (SCR:TXE="1"), the serial chip select pin becomes active when transmission data is written.
2. After the elapse of the serial chip select pin setup time, transmission/reception operations are started.
3. After data transmission/reception for which the count is set in TBYTE, the serial clock stops.
4. After the serial clock is stopped and the serial chip select pin hold time is elapsed, the serial chip select pin becomes inactive.

**Figure 5-1 Serial Chip Select Operation (Master Transmission (MS=0), Normal Transfer (SPI=0), SCINV=0)**



**Figure 5-2 Serial Chip Select Operation (Master Transmission (MS=0), SPI Transfer (SPI=1), SCINV=0)**



**Notes:**

- While the serial chip select pin is active, if transmission is disabled ( $SCR:TXE=0$ ) and software reset is set ( $SCR:UPCL=1$ ), the serial chip select pin becomes inactive.
- If the active state of the serial chip select pin is not retained ( $SCSCR:SCAM=0$ ), the serial chip select pin becomes inactive, and if there is no transmission data ( $SSR:TDRE=1$ ) after the elapse of the deselect time, the transmission bus idle state ( $SSR:TBI=1$ ) is assumed.
- In master mode ( $SCR:MS=0$ ), if  $SCSCR:CSEN3$  to  $CSEN0$  is set to  $0b0000$ , transmission/reception operations are performed regardless of the status of the serial chip select pin.
- When fewer frames than the number set in  $TBYTE$  have been transmitted, if the transmission data register ( $TDR$ ) contains no valid transmission data at the time of completion of 1-frame transmission ( $SSR:TDRE = 1$ ), the operation described below is performed.
  - If transfer byte errors are enabled ( $TBEEN=1$ ), a chip select error ( $SACSR:CSE=1$ ) occurs. Once the hold delay time elapses after the occurrence of a chip select error ( $SACSR:CSE=1$ ), the serial chip select pin becomes inactive. If the chip select error flag is set ( $SACSR:CSE=1$ ), transmission is not started even if transmission data is written to the transmission data register ( $TDR$ ).
  - If transfer byte errors are disabled ( $TBEEN=0$ ), transmission is stopped until transmission data is written to the transmission data register ( $TDR$ ). At this time, the serial chip select pin is active. Writing transmission data to the transmission data register ( $TDR$ ) restarts the transmission operation.





### b) Serial Chip Select Timing Adjustment

If, in master mode (SCR:MS=0), serial chip select operation is enabled (SCSCR:CSENn="1"), by adjusting the serial chip select timing registers (SCSTR3 to SCSTR0), setup delay, hold delay, and deselect time can be adjusted.

#### – Setup delay time

Time from the instant at which the serial chip select pin becomes active until the serial clock is output. For details on the regulations governing the setup delay time, see Figure 5-3 and Figure 5-4. This can be adjusted with the chip select setup delay bits (SCSTR1:CSSU7 to CSSU0).

#### – Hold delay time

Time from the instant at which the output of the serial clock ends until the serial chip select pin becomes inactive. For details on the regulations governing the hold delay time, see Figure 5-3 and Figure 5-4.

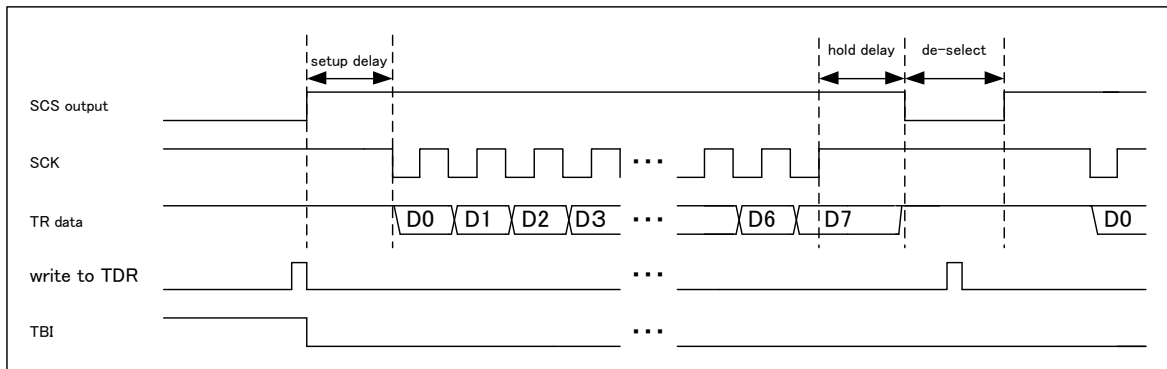
This can be adjusted with the chip select hold delay bits (SCSTR0:CSDH7 to CSDH0).

#### – Deselect time

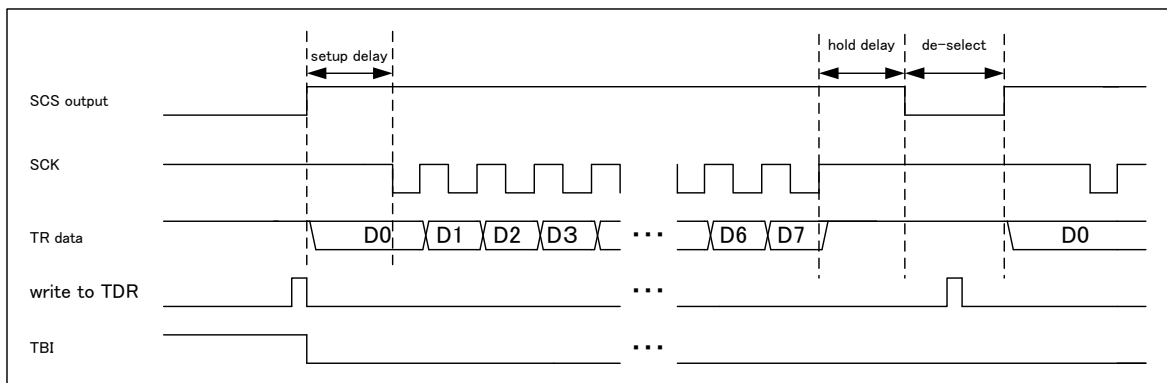
Minimum time from the instant at which the serial chip select pin becomes inactive until the next time that the serial chip select pin becomes active. Even if transmission data is written to the transmission data register (TDR) within the deselect time, the serial chip select pin does not become active until the end of the deselect time. For details on the regulations governing the deselect time, see Figure 5-3 and Figure 5-4.

This can be adjusted with the chip select/deselect bits (SCSTR3 to SCSTR2:CSDS15 to CSDS0).

**Figure 5-3 Timing Adjustment (Normal Transfer (SPI=0), SCINV=0)**



**Figure 5-4 Timing Adjustment (SPI Transfer (SPI=1), SCINV=0)**



**Notes:**

- During normal transfer (SCR:SPI=0) with no hold delay time (SCSTR0:CSHD7 to 0 = 0x00), the chip select pin may become inactive before the sampling of the last bit. In this case, perform adjustments by increasing the values of SCSTR0:CSHD7 to 0.
- During SPI transfer (SCR:SPI=1) with no setup delay time (SCSTR1:CSSU7 to 0 = 0x00), the chip select pin may become active after the sampling of the first bit. In this case, perform adjustments by increasing the values of SCSTR1:CSSU7 to 0.



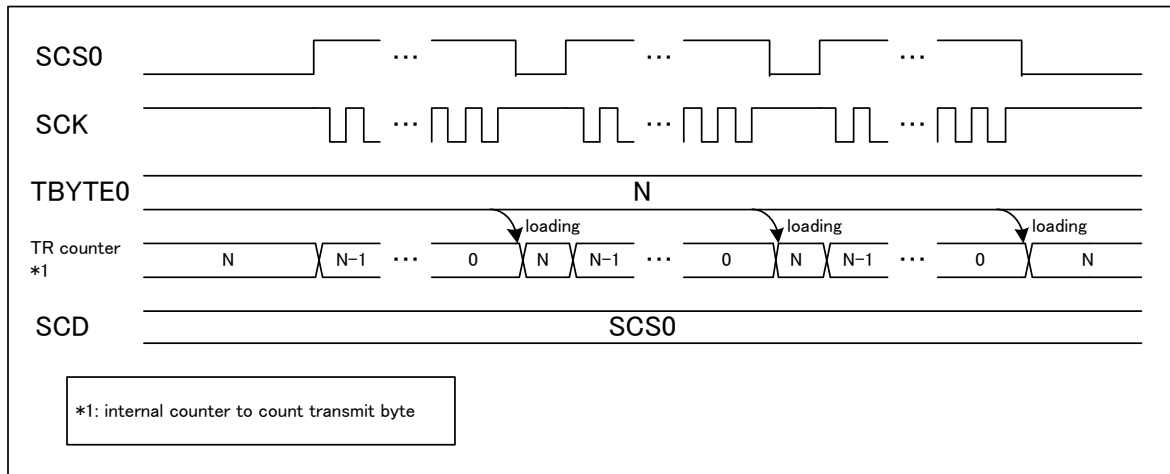
**c) Independent Operation of the Chip Select Pin (Valid only in Master Mode (SCR:MS = 0))**

If the serial chip select start bits (SCSCR:SST1 and SST0) and the serial chip select end bits (SCSCR:SED1 and SED0) are equal, operation is performed based on only the set serial chip select pin.

When the active level of serial chip select is not retained (SCSCR:SCAM=0), the serial chip select pin becomes inactive for each data transmission/reception for which the count is set in TBYTE.

For details on the serial chip select pin operation performed if the active level of serial chip select is retained (SCSCR:SCAM = 1), see "e) Serial chip select active level retention operation".

**Figure 5-5 Independent Operation of Chip Select (SST1 and SST0=0, SED1 and SED0=0, CSEN0=1, SCAM=0)**



**Note:**

- During independent operation, serial chip select pin timing adjustment (setup time, hold time, and deselect time) is valid.

**d) Rounding Operation of the Chip Select Pin (Valid only in Master Mode (SCR:MS = 0))**

If the serial chip select start bits (SCSCR:SST1 and SST0) and the serial chip select end bits (SCSCR:SED1 and SED0) differ, multiple serial chip select pins become active sequentially.

1. If transmission data is written when you enable both serial chip select output (SCSCR:CSOE = "1") and transmission (SCR:TXE = "1"), the serial chip select pins become active, starting with that specified with the serial chip select start bits (SCSCR:SST1 and SST0).
2. When the active level of serial chip select is not retained (SCSCR:SCAM=0), the serial chip select pin becomes inactive after the end of data transmission/reception for which the count is set in TBYTE. Later, the serial chip select pin with the pin number equal to the number of the last active serial chip select pin + 1 becomes active.\*1
3. If the next serial chip select pin is disabled (SCSCR:CSEnN=0), the serial chip select pin does not become active but is instead skipped.
4. If the active serial chip select pin number matches the serial chip select pin specified with the serial chip select end bits (SCSCR:SED1 and SED0), the serial chip select pin specified with the serial chip select start bits (SCSCR:SST1 and SST0) becomes active next.

\*1: If the last active serial chip select pin is pin 0, pin 1 becomes active; if it is pin 1, pin 2 becomes active; if it is pin 2, pin 3 becomes active; and if it is pin 3, pin 0 becomes active.

For details on the serial chip select pin operation performed if the active level of serial chip select is retained (SCSCR:SCAM=1), see "e) Serial Chip Select Active Level Retention Operation".

Figure 5-6 shows the timing chart that is assumed if the serial chip select start pin is SCS0 (SST1 and SST0=0) and the end pin is SCS3 (SED1 and SED0=3).

**Figure 5-6 Rounding Operation of Chip Select (SST1 and SST0=0, SED1 and SED0=3, CSEN3=1, CSEN2=1, CSEN1=1, CSEN0=1, SCAM=0)**

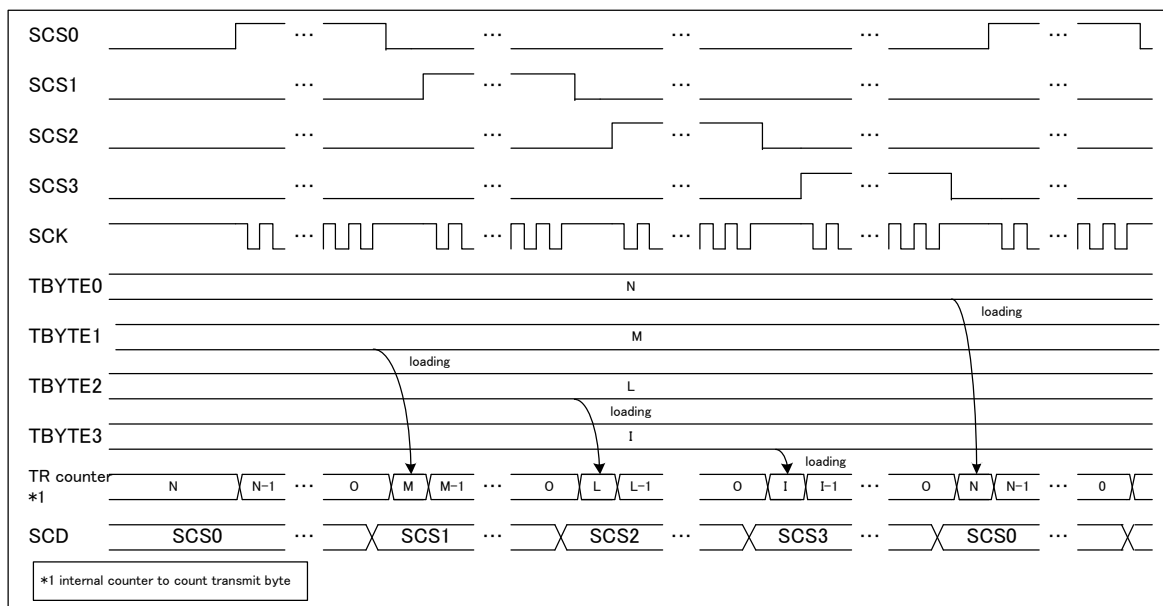




Figure 5-7 shows the timing chart that is assumed if the serial chip select start pin is SCS1 (SST1 and SST0=1) and the end pin is SCS2 (SED1 and SED0=2).

**Figure 5-7 Rounding Operation of Chip Select (SST1 and SST0=1, SED1 to 0=2 CSEN3=0, CSEN2=1, CSEN1=1, CSEN0=0, SCAM=0)**

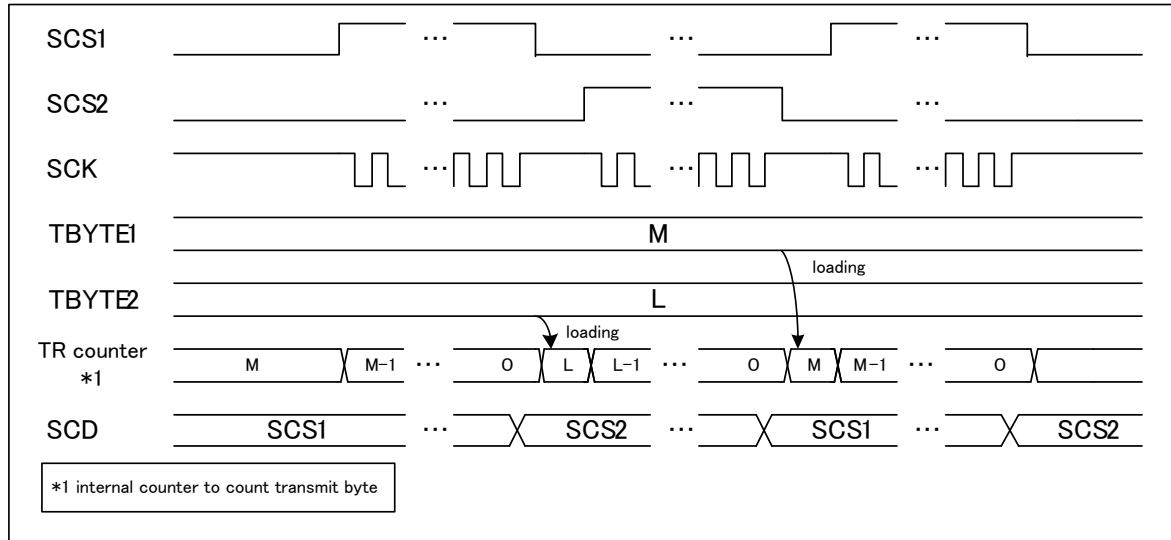
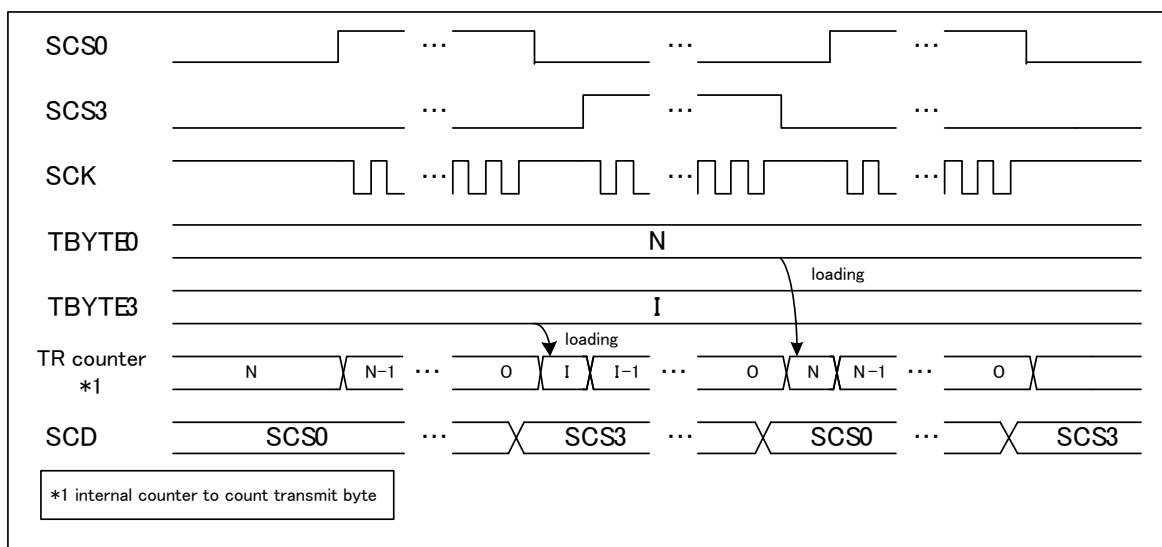


Figure 5-8 shows the timing chart that is assumed if the serial chip select start pin is SCS0 (SST1 and SST0=0), the end pin is SCS3 (SED1 and SED0=3), and chip select pins 1 and 2 are disabled (CSEN1 and CSEN2 =0b00). After serial chip select pin 0 becomes active, pins 1 and 2 are skipped, and pin 3 becomes active.

**Figure 5-8 Rounding Operation of Chip Select (SST1 and SST0=0, SED1 and SED0=3, CSEN3=1, CSEN2=0, CSEN1=0, CSEN0=1, SCAM=0)**



**Notes:**

- Serial chip select pins become active starting with that specified with the serial chip select start bits (SCSCR:SST1 and SST0) in any of the following cases:
  - If a change is made from transmission operation disable (SCR:TXE=0) to transmission operation enable (SCR:TXE=1)
  - If a software reset (SCR:UPCL=1) is performed
- During a rounding operation, serial chip select pin timing adjustment (setup time, hold time, and deselect time) is valid.



**e) Serial Chip Select Active Level Retention Operation (SCSCR:SCAM=1) (Valid only in Master Mode (SCR:MS=0))**

If the serial chip select active level retention bit (SCSCR:SCAM) is set to "1" and transmission is started, the serial chip select pin is retained in the active state.

**Table 5-1 Serial Chip Select Active Level Retention Bit (SCSCR:SCAM)**

Current State	Current SCSCR:SCAM Bit	Current SSR:TDRE Bit	Next State
Transmitting (Transmission count < TBYTE)	0	-	Until as many frames as the value set for TBYTE have been transmitted, the serial chip select pin remains active.
	1		
End of transmission of as many frames as the setting of TBYTE	0	0	After the hold delay time, the serial chip select pin is deactivated. After the elapse of the deselect time, the next transmission is started.
		1	After the hold delay time, the serial chip select pin is deactivated. After the elapse of the deselect time, transmission is stopped until the next transmission is written.
	1	1	The active level of the serial chip select is retained.
		0	With serial chip select being in the active level, transmission operation continues. Until as many frames as the value set for TBYTE have been transmitted, the serial chip select pin remains active.
A chip select error (SACSR:CSE=1) occurs.	-	-	Regardless of the setting of SCAM, the serial chip select pin is deactivated after the hold delay time.
A software reset is executed (SCR:UPCL=1).	-	-	Regardless of the setting of SCAM, the serial chip select pin is deactivated immediately.
Transmission disabled (SCR:TXE=0)			

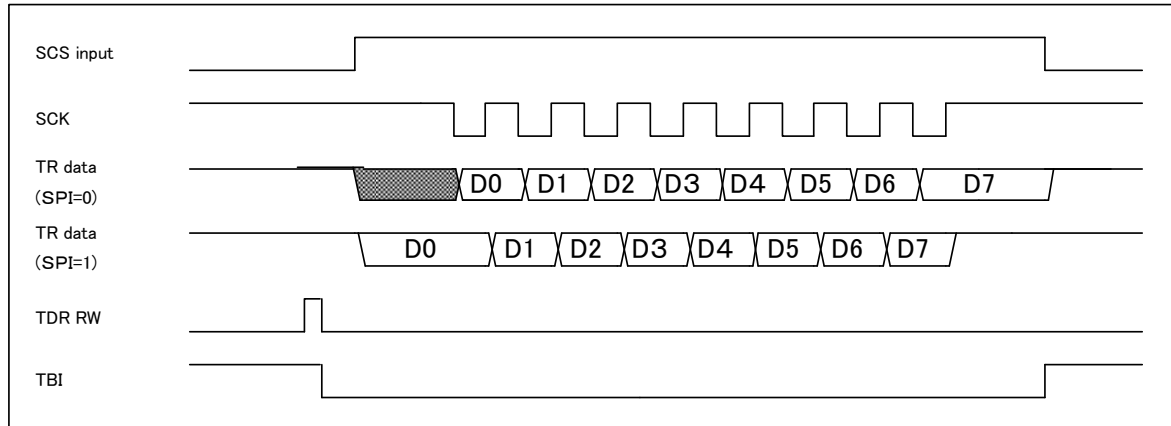
**Notes:**

- If the conditions described below are all satisfied, the serial chip select pin is not retained, and after the elapse of the hold delay time, the serial chip select pin becomes inactive, causing a chip select error (SACSR:CSE=1) to occur.
  - Transfer byte errors are enabled (SACSR:TBEEN=1).
  - If data transmission/reception for which the count is set in TBYTE has not ended.
  - If the transmission data register (TDR) is empty (SSR:TDRE=1).

**f) Operation in Slave Mode (SCR:MS=1)**

If serial chip select pin 0 (SCS0) is enabled (SCSCR:CSEN0="1") and the serial chip select pin input becomes active, transmission or reception is performed in synchronization with the serial clock (SCK). Later, if the serial chip select pin input becomes inactive, transmission or reception is ended.

**Figure 5-9 Serial Chip Select Operation in Slave Mode (Slave Transmission, SCINV=0)**



**Notes:**

- When the serial chip select pin input is inactive, operation is not performed even if the serial clock is input.
- If the serial chip select input becomes inactive before bits are last sampled during reception, the data being received is erased.
- If the serial chip select input becomes inactive during transmission, the data being transmitted is erased and a chip select error occurs (SACSR:CSE).
- If TDR is empty (SSR:TDRE=1) and the serial chip select pin input becomes inactive, transmission bus idle state (SSR:TBI=1) is assumed.
- In slave mode (SCR:MS=1), if SCSCR:CSEN0 is set to "0", transmission/reception operations are performed regardless of the status of the serial chip select pin.





#### g) Serial Chip Select Pin Format Setting

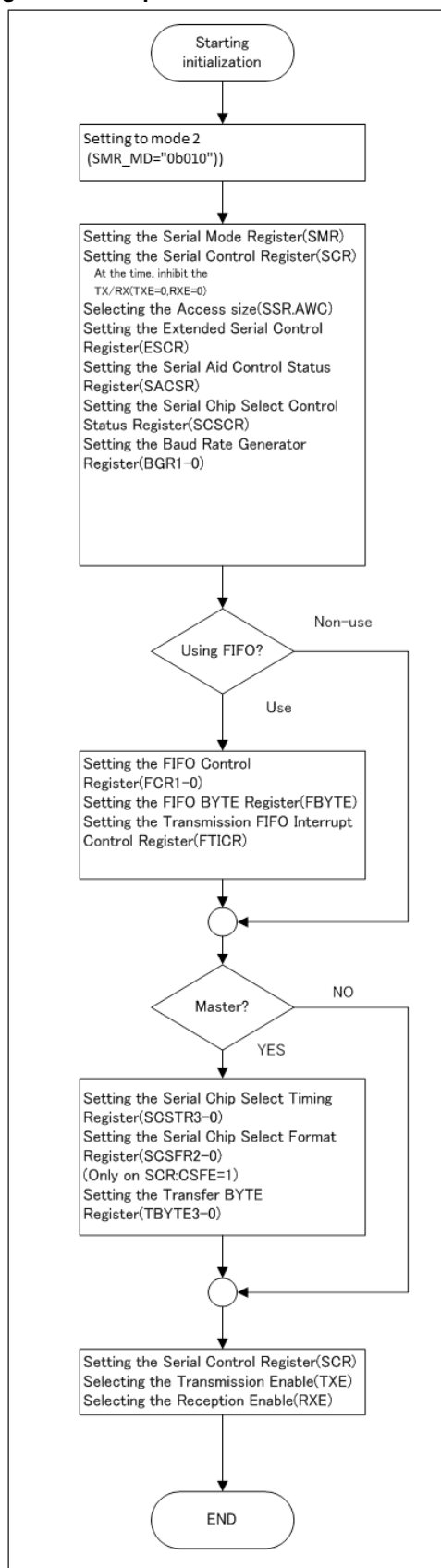
The active level of chip select of each serial chip select pin, mark level of the serial clock, SPI mode enabling and disabling, and data direction and data length of serial data output can be set with the bits listed in Table 5-2.

Table 5-2 Serial Chip Select Pin Format Setting

Conditions		Chip Select Active Level	Serial Clock Inversion	SPI Setting	Data Direction	Data Length
Chip select format is enabled (SCR:CSFE=1) and master mode (SCR:MS=0)	Serial chip select pin 0 output	SCSCR0:SCLVL	SMR:SCINV	SCR:SPI	SMR:BDS	ESCR:L3 to L0
	Serial chip select pin 1 output	SCSFR0: CS1SCLVL	SCSFR0: CS1SCINV	SCSFR0: CS1SPI	SCSFR0: CS1BDS	SCSFR0: CS1L3 to CS1L0
	Serial chip select pin 2 output	SCSFR1: CS2SCLVL	SCSFR1: CS2SCINV	SCSFR1: CS2SPI	SCSFR1: CS2BDS	SCSFR1: CS2L3 to CS2L0
	Serial chip select pin 3 output	SCSFR2: CS3SCLVL	SCSFR2: CS3SCINV	SCSFR2: CS3SPI	SCSFR2: CS3BDS	SCSFR2: CS3L3 to CS3L0
Chip select format disable (ESCR:CSFE=0)		SCSCR0:SCLVL	SMR:SCINV	SCR:SPI	SMR:BDS	ESCR:L3 to L0
Slave mode (MS=1)						
When chip select is not used (CSEN3 to CSEN0=0b0000)						

## h) Initialization Flow

Figure 5-10 Chip Select Initialization Flow





## 6. Test Mode

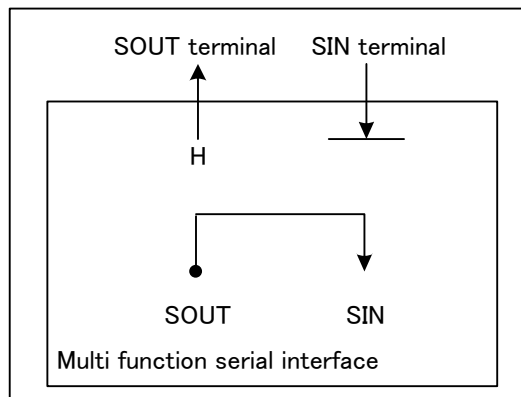
This section describes operation in test mode.

### Serial Test Mode

When serial test mode is enabled (SACSR:STST=1), SOUT and SIN are connected together inside the multifunction serial interface, so that the data transmitted from SOUT can be received from SIN directly.

When serial test mode is enabled (SACSR:STST=1), the SOUT pin is fixed to "H", and the data input to the SIN pin is ignored.

**Figure 6-1 Serial Test Mode**



**Note:**

- The value of the serial test mode enable bit (SACSR:STST) can be changed only when transmission and reception are prohibited (SCR:TXE=RXE=0).

## 7. Dedicated Baud Rate Generator

The dedicated baud rate generator functions during master operation only. When using the reception FIFO, set the dedicated baud rate generator even during slave operation.

### CSIO (Clock Synchronous Serial Interface) Baud Rate Selection

The dedicated baud rate generator must be set differently for master operation and slave operation.

#### [1] During Master Operation

- Use the dedicated baud rate generator to divide the internal clock and select a baud rate.
- There are 2 internal reload counters, each of which corresponds to the transmission or reception serial clock. A baud rate can be selected by setting a 15-bit reload value in baud rate generator register 1 or 0 (BGR1, BGR0).
- The reload counters divide the internal clock by the value that is set.

#### [2] During Slave Operation

- During slave operation (SCR:MS=1), the dedicated baud rate generator does not operate. (Instead, the external clock, input from clock input pin SCK, is input directly.)

**Note:**

- *When using the reception FIFO, set the dedicated baud rate generator even during slave operation.*



## 7.1. Setting the Baud Rate

This section describes the baud rate setting. It also describes the results of calculating serial clock frequencies.

### (1) Baud Rate Calculation

The 2 15-bit reload counters are set by baud rate generator registers 1, 0 (BGR1, BGR0).  
The baud rate is calculated using the formula given below.

#### a) Reload Value:

$$V = \text{Phy} / b - 1$$

V : reload value    b : baud rate    Phy : bus clock frequency

#### b) Calculation Example

When bus clock is 16MHz, using internal clock, baud rate is 19200 bps, the reload value can be calculated as follows.

reload value :

$$V = (16 * 1000000) / 19200 - 1 = 832$$

and then baud rate will be

$$b = (16 * 1000000) / (832 + 1) = 19208 \text{ bps}$$

#### c) Baud Rate Error

The baud rate error is obtained by using the following formula.

$$\text{error rate}(\%) = (\text{calculated value} - \text{target value}) / \text{target value} * 100$$

(ex. When bus clock is 20MHz, target baud rate is 153600 bps,

the result will be as follows.

$$\text{reload value} = (20 * 1000000) / 153600 - 1 = 129$$

$$\text{calculated baud rate} = 20 * 1000000 / (129 + 1) = 153846 \text{ (bps)}$$

$$\text{error rate}(\%) = (153846 - 153600) / 153600 * 100 = 0.16 (\%)$$

#### Notes:

- Setting the reload value to "0" stops the reload counter.
- If the reload value is even-numbered, the "H" width and "L" width of the serial clock will be as described below, depending on the settings made for the SMR:SCINV bit and SCR:SPI bit. If the reload value is odd-numbered, the "H" width and the "L" width of the serial clock will be the same.
  - For normal transfer (SCR:SPI=0) when the mark level of the serial clock is "H" (SMR:SCINV="0") or for SPI transfer (SCR:SPI=1) when the mark level of the serial clock is "L" (SMR:SCINV="1"), the "H" width of the serial clock will be longer by 1 bus clock cycle.
  - For normal transfer (SCR:SPI=0) when the mark level of the serial clock is "L" (SMR:SCINV="1") or for SPI transfer (SCR:SPI=1) when the mark level of the serial clock is "H" (SMR:SCINV="0"), the "L" width of the serial clock will be longer by 1 bus clock cycle.
- Set the reload value to 3 or more.
- When considering the tolerable baud rate range, also consider the impact of the jitter in the clock input on the macro.

## (2) Example Reload Values and Baud Rate Settings for Individual Bus Clock Frequencies

The following table contains example reload values and baud rate settings.

**Table 7-1 Example Reload Values and Baud Rate Settings**

Baud Rate (bps)	8 MHz		10 MHz		16 MHz		20 MHz		24 MHz		32 MHz	
	Value	ERR	Value	ERR	Value	ERR	Value	ERR	Value	ERR	Value	ERR
8 M	-	-	-	-	-	-	-	-	-	-	3	0
6 M	-	-	-	-	-	-	-	-	3	0	-	-
5 M	-	-	-	-	-	-	3	0	-	-	-	-
4 M	-	-	-	-	3	0	4	0	5	0	7	0
2.5 M	-	-	3	0	-	-	7	0	-	-	-	-
2 M	3	0	4	0	7	0	9	0	11	0	15	0
1 M	7	0	9	0	15	0	19	0	23	0	31	0
500000	15	0	19	0	31	0	39	0	47	0	63	0
460800	-	-	-	-	-	-	-	-	51	0.16	-	-
250000	31	0	39	0	63	0	79	0	95	0	127	0
230400	-	-	-	-	-	-	86	-0.22	103	0.16	138	-0.08
153600	51	0.16	64	0.16	103	0.16	129	0.16	155	0.16	207	0.16
125000	63	0	79	0	127	0	159	0	191	0	255	0
115200	-	-	86	-0.22	138	-0.08	173	-0.22	207	0.16	277	-0.08
76800	103	0.16	129	0.16	207	0.16	259	0.16	312	-0.16	416	-0.08
57600	138	-0.08	173	-0.22	277	-0.08	346	0.06	416	-0.08	555	-0.08
38400	207	0.16	259	0.16	416	-0.08	520	-0.03	624	0	832	0.04
28800	277	-0.08	346	0.06	555	-0.08	693	0.06	832	0.04	1110	0.01
19200	416	-0.08	520	-0.03	832	0.04	1041	-0.03	1249	0	1666	-0.02
10417	767	<0.01	959	<0.01	1535	<0.01	1919	<0.01	2303	<0.01	3071	<0.01
9600	832	0.04	1041	-0.03	1666	-0.02	2082	0.01	2499	0	3332	0.01
7200	1110	0.01	1388	<0.01	2221	0.01	2777	<0.01	3332	0.01	4443	0.01
4800	1666	-0.02	2082	-0.02	3332	0.01	4166	<0.01	4999	0	6666	<0.01
2400	3332	0.01	4166	<0.01	6666	<0.01	8332	<0.01	9999	0	13332	0.01
1200	6666	<0.01	8332	<0.01	13332	<0.01	16666	<0.01	19999	0	26666	<0.01
600	13332	<0.01	16666	<0.01	26666	<0.01	-	-	-	-	-	-
300	26666	<0.01	-	-	-	-	-	-	-	-	-	-

- Value: Setting of the BGR1/0 register
- ERR: Baud rate error (%)



Table 7-2 Example Reload Values and Baud Rate Settings (Continued from the Previous Page)

Baud Rate (bps)	40 MHz		48 MHz		72 MHz		80 MHz	
	Value	ERR	Value	ERR	Value	ERR	Value	ERR
8 M	4	0	5	0	8	0	9	0
6 M	-	-	7	0	11	0	-	-
5 M	7	0	-	-	-	-	15	0
4 M	9	0	11	0	17	0	19	0
2.5 M	15	0	-	-	-	-	31	0
2 M	19	0	23	0	35	0	39	0
1 M	39	0	47	0	71	0	79	0
500000	79	0	95	0	143	0	159	0
460800	86	-0.22	103	0.16	155	0.16	173	-0.22
250000	159	0	191	0	287	0	319	0
230400	173	-0.22	207	0.16	312	-0.16	346	0.06
153600	259	0.16	312	-0.16	468	-0.05	520	-0.03
125000	319	0	383	0	575	0	639	0
115200	346	0.06	416	-0.08	624	0	693	0.06
76800	520	-0.03	624	0	937	-0.05	1041	-0.03
57600	693	0.06	832	0.04	1249	0	1388	<0.01
38400	1041	-0.03	1249	0	1874	0	2082	0.01
28800	1388	<0.01	1666	-0.02	2499	0	2777	<0.01
19200	2082	0.01	2499	0	3749	0	4166	-0.01
10417	3839	<0.01	4607	<0.01	6911	<0.01	7679	0
9600	4166	<0.01	4999	0	7499	0	8332	0
7200	5555	<0.01	6666	<0.01	9999	0	11110	0
4800	8332	<0.01	9999	0	14999	0	16666	0
2400	16666	<0.01	19999	0	29999	0	-	-
1200	-	-	-	-	-	-	-	-
600	-	-	-	-	-	-	-	-
300	-	-	-	-	-	-	-	-

- Value: Setting of the BGR1/0 register
- ERR: Baud rate error (%)

**(3) Functions of the Reload Counters**

The reload counters include the transmission reload counter and the reception reload counter. They function as a dedicated baud rate generator. They consist of 15-bit registers for reload values, and generate transmission/reception clocks from their internal clocks.

**(4) Start of Counting**

When a reload value is written to the baud rate generator registers (BGR1 and BGR0), the reload counters start counting.

**(5) Restarting**

The reload counter restarts under the conditions described below.

**For both transmission and reception reload counters**

Programmable reset (SCR:UPCL bit)





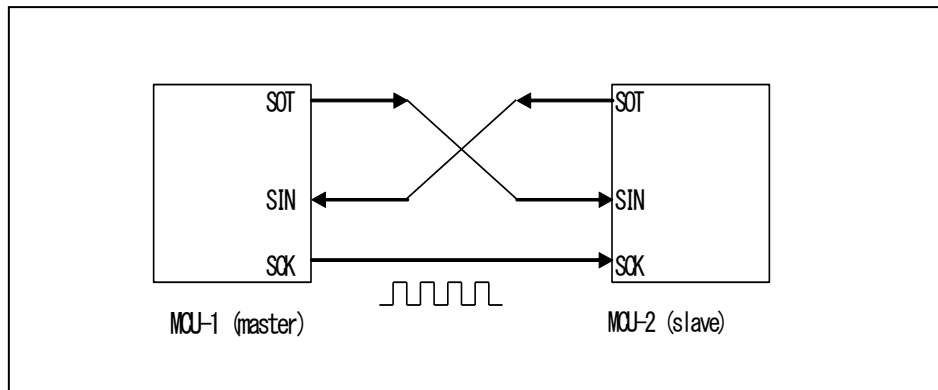
## 7.2. Setup Procedure and Program Flow for the CSIO (Clock Synchronous Serial Interface)

The CSIO (Clock Synchronous Serial Interface) can perform serial bidirectional transmission.

### Connection between MCUs

For the CSIO (Clock Synchronous Serial Interface), select bidirectional communication. As shown in Figure 7-1, 2 MCUs are interconnected.

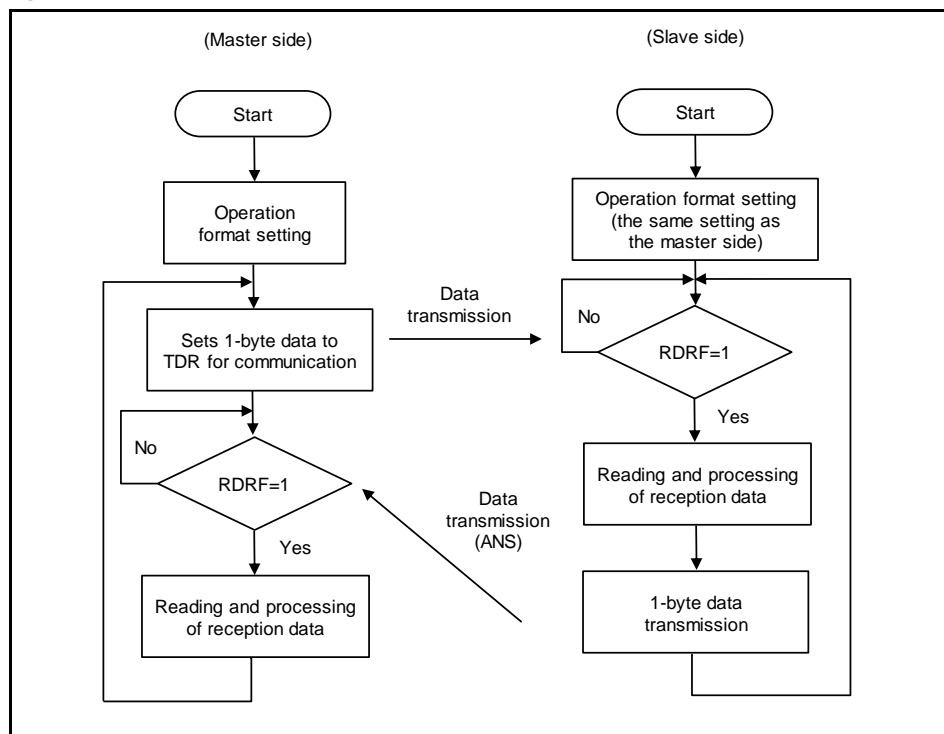
Figure 7-1 Connection Example for CSIO (Clock Synchronous Serial Interface) Bidirectional Communication



### (1) Flow Chart

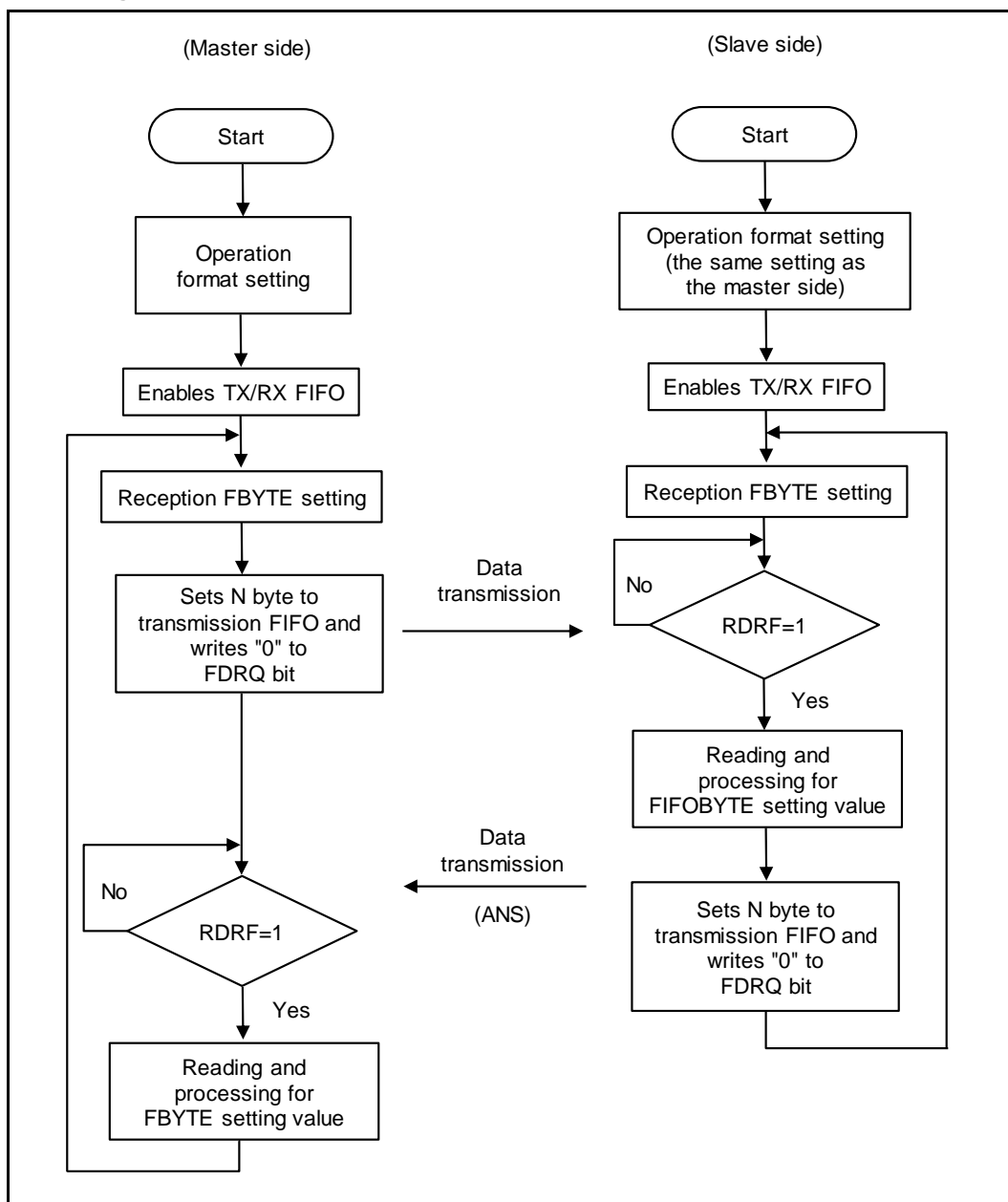
#### a) FIFO Not Used

Figure 7-2 Example of Bidirectional Communication Flow Chart (FIFO Not Used)



b) FIFO Used

Figure 7-3 Example of Bidirectional Communication Flow Chart (FIFO Used)





## 8. Registers

This section lists the registers of the CSIO (Clock Synchronous Serial Interface).

All registers have the prefix "MFSxx\_" attached to their names. xx is a channel number (00, 01, 02, 03 and 04).

**Table 8-1 List of CSIO (Clock Synchronous Serial Interface) Registers**

Abbreviated Register Name	Register Name	See
SCR	Serial Control Register	8.1
SMR	Serial Mode Register	8.2
SSR	Serial Status Register	8.3
ESCR	Extended Communication Control Register	8.4
RDR/TDR	Reception Data Register/Transmission Data Register	8.5
SACSR	Serial Auxiliary Control Status Register	8.6
STMR	Serial Timer Register	8.7
STMCR	Serial Timer Comparison Register	8.8
SCSCR	Serial Chip Select Control Status Register	8.9
SCSTR3 to SCSTR0	Serial Chip Select Timing Registers	8.10
SCSFR2 to SCSFR0	Serial Chip Select Format Registers	8.11
TBYTE3 to TBYTE0	Transfer Byte Registers	8.12
BGR0/1	Baud Rate Generator Registers	8.13
FCR1	FIFO Control Register 1	8.14
FCR0	FIFO Control Register 0	8.15
FBYTE	FIFO Byte Register	8.16
FTICR	Transmission FIFO Interrupt Control Register	8.17
SACSRC	Serial Auxiliary Control Status Clear Register	8.18
FCR1C	FIFO Control Clear Register 1	8.19
FCR0C	FIFO Control Clear Register 0	8.20
SACSRS	Serial Auxiliary Control Status Set Register	8.21
FCR1S	FIFO Control Set Register 1	8.22
FCR0S	FIFO Control Set Register 0	8.23

## 8.1. Serial Control Register (SCR)

The serial control register (SCR) is used to enable/disable transmission/reception interrupts, enable/disable transmission idle interrupts, and enable/disable transmission/reception operations. It can also be used to make settings for connection to SPI and to reset the CSIO.

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	UPCL	MS	SPI	RIE	TIE	TBIE	RXE	TXE
ACCESS_TYPE	R0,W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

The lower byte [bit7:0] of this register is the Serial Mode Register (SMR).

### [bit15] UPCL: Programmable clear bit

This bit initializes the internal state of the CSIO.

When "1" is set:

- Resets the CSIO directly (software reset). However, the register setting is retained. At this time, anything in the transmission/reception state is disconnected immediately.
- The baud rate generator reloads the setting of the BGR1/0 register and then restarts.
- All transmission/reception and status interrupt factors (SSR:TDRE, TBI, RDRF, ORE, SACS:R:TINT, CSE) are initialized.
- All serial chip select pins become inactive.

When "0" is set:

Does not have any effect on the operation.

At reading, "0" is always read.

Value	Description	
	Write	Read
0	No effect	"0" is always read
1	Programmable clear	

### Notes:

- Execute programmable clear (SCR:UPCL = 1) after disabling interrupts.
- When FIFOs are used, execute programmable clear after disabling the FIFOs (FCR0:FE2, FE1 = 0).
- Executing programmable clear (SCR:UPCL = 1) does not clear the value of the serial timer register (STMR).

**[bit14] MS: Master/slave function selection bit**

This bit selects either master or slave mode.

- 0: Set master mode.
- 1: Set slave mode.

Value	Description
0	Master mode
1	Slave mode

**Notes:**

- If slave mode is selected, an external clock is input directly, provided that  $SMR:SCKE = 0$ .
- Set this bit when transmission/reception is disabled ( $TXE = RXE = 0$ ).
- After setting the MS bit, enable reception ( $RXE = 1$ ).
- Set AWC bit of SSR register to "1" when transmission FIFO is used for SPI transfer in slave mode.

**[bit13] SPI: SPI-supporting bit**

This bit is for performing communication that supports SPI. When chip select is used in master mode ( $SCR:MS = 0$ ), the bit is used for communication with serial chip select pin 0.

- 0: Perform normal synchronous communication.
- 1: Support SPI.

Value	Description
0	Normal synchronous transfer
1	SPI support

**Notes:**

- Set this bit when transmission and reception are disabled ( $TXE = RXE = 0$ ).
- This bit is used in any of the following cases:
  - The chip select pins are disabled ( $SCSCR:CSEN3$  to  $CSEN0 = "0000"$ ).
  - In slave mode ( $SCR:MS = 1$ ).
  - When the chip select data format is disabled ( $ESCR:CSFE = 0$ ).
  - When the chip select data format is enabled ( $ESCR:CSFE = 1$ ) and serial chip select pin 0 is active.
- Set AWC bit of SSR register to "1" when transmission FIFO is used for SPI transfer in slave mode.
- When serial chip select is used for normal transfer in master mode, either of the following conditions are satisfied.
  - $Baud\ rate\ period / 2\ [ns] < SCSTR0:CSD7\ to\ 0 \times bus\ clock\ period \times 2^{SCSCR:CDIV2\ to\ 0} + 3 \times bus\ clock\ period\ [ns]$
  - $SCSTR0:CSD7\ to\ 0 \times bus\ clock\ period \times 2^{SCSCR:CDIV2\ to\ 0} + SCSTR1:CSD7\ to\ 0 \times bus\ clock\ period \times 2^{SCSCR:CDIV2\ to\ 0} [ns] < baud\ rate\ period - 2 \times bus\ clock\ period\ [ns]$
- Set data transmission/reception wait select bits ( $ESCR:WT[1:0]$ ) to "0b00" when serial chip select active retention is used ( $SCSCR:SCAM=1$ ) in SPI mode.

**[bit12] RIE: Reception interrupt enable bit**

- This bit enables/disables reception interrupt request output to the CPU.
- If the RIE bit and the reception data flag bit (SSR:RDRF) are "1", or the error flag bit (ORE) is "1", a reception interrupt request will be generated.

Value	Description
0	Disable reception interrupts
1	Enable reception interrupts

**[bit11] TIE: Transmission interrupt enable bit**

- This bit enables/disables transmission interrupt request output to the CPU.
- If the TIE bit and the SSR:TDRE bit are "1", the transmission interrupt request will be generated.

Value	Description
0	Disable transmission interrupts
1	Enable transmission interrupts

**[bit10] TBIE: Transmission bus idle interrupt enable bit**

- This bit enables/disables transmission bus idle interrupt request output to the CPU.
- A transmission bus idle interrupt request is output when the TBIE bit and the SSR:TBI bit are both "1".

Value	Description
0	Disable transmission bus idle interrupts
1	Enable transmission bus idle interrupts

**[bit9] RXE: Reception operation enable bit**

This bit enables/disables CSIO reception.

- 0: Disable data frame reception.
- 1: Enable data frame reception.

Value	Description
0	Disable reception
1	Enable reception

**Notes:**

- If reception is disabled ( $RXE = 0$ ) during reception, reception is stopped immediately.
- After setting the MS bit and the SMR:SCINV bit, enable reception ( $RXE = 1$ ).



**[bit8] TXE: Transmission operation enable bit**

This bit enables/disables CSIO transmission.

- 0: Disable data frame transmission.
- 1: Enable data frame transmission.

Value	Description
0	Disable transmission
1	Enable transmission

**Notes:**

- If transmission is disabled ( $TXE = 0$ ) during transmission, transmission is stopped immediately.
- When serial chip select is used ( $SCSCR:CSEN = 1$ ) in master mode ( $SCR:MS = 0$ ), perform a programmable reset ( $SCR:UPCL = 1$ ) after disabling transmission.

## 8.2. Serial Mode Register (SMR)

The serial mode register (SMR) is used to set the operating mode, transfer direction, and data length, invert the serial clock, and enable/disable the output of serial data and clocks to pins.

BITS_OFFSET	7	6	5	4	3	2	1	0
BITS_NAME	MD2	MD1	MD0	Reserve	SCINV	BDS	SCKE	SOE
ACCESS_TYPE	R/W	R/W	R/W	R/W0	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

### [bit7:5] MD2, MD1, MD0: Operation mode setting bits

These bits set the operation modes.

"0b000": Set operation mode 0 (asynchronous normal mode).

"0b001": Set operation mode 1 (asynchronous multi-processor mode).

"0b010": Set operation mode 2 (clock synchronous mode).

"0b011": Set operation mode 3 (LIN communication mode).

Value			Description
0	0	0	Operation mode 0 (asynchronous normal mode)
0	0	1	Operation mode 1 (asynchronous multi-processor mode)
0	1	0	Operation mode 2 (clock synchronous mode)
0	1	1	Operation mode 3 (LIN communication mode)
Other than the above			Setting is prohibited

\* This section describes the registers and their operations in operation mode 2.

For details on operating modes 0/1, see "CHAPTER: UART". For details on operating mode 3, see "CHAPTER: LIN Interface".

### Notes:

- To switch the operation mode, execute programmable clear (SCR:UPCL = 1) and then switch the operation mode.
- Set the registers after setting an operation mode.

### [bit4] Reserved: Reserved bit



**[bit3] SCINV: Serial clock invert bit**

This bit inverts the serial clock format. When chip select is used in master mode (SCR:MS = 0), this bit is used for communication with serial chip select pin 0.

0:

- The mark level of serial clock output is set to "H".
- During normal transfer, transmission data is output in synchronization with a falling edge of the serial clock. In SPI transfer, it is output in synchronization with a rising edge of the serial clock.
- During normal transfer, reception data is sampled at a rising edge of the serial clock. In SPI transfer, it is sampled at a falling edge of the serial clock.

1:

- The mark level of serial clock output is set to "L".
- During normal transfer, transmission data is output in synchronization with a rising edge of the serial clock. During SPI transfer, it is output in synchronization with a falling edge of the serial clock.
- During normal transfer, reception data is sampled at a falling edge of the serial clock. In SPI transfer, it is sampled at a rising edge of the serial clock.

Value	Description
0	Mark level "H" format
1	Mark level "L" format

**Notes:**

- Set this bit when the serial clock output is disabled (SCKE = 0).
- After setting the SCINV bit, enable reception (SCR:RXE = 1).
- Set this bit when transmission and reception are disabled (TXE = RXE = 0).
- This bit is used in any of the following cases:
  - The chip select pins are disabled (SCSCR:CSEN3 to CSEN0 = "0000").
  - In slave mode (SCR:MS = 1).
  - When the chip select data format is disabled (ESCR:CSFE = 0).
  - When the chip select data format is enabled (ESCR:CSFE = 1) and serial chip select pin 0 is active.
- Set this bit to "0" when serial test mode is enabled (SACSR:STST = 1).

**[bit2] BDS: Transfer direction selection bit**

This bit selects whether the lowest bit is transferred first (LSB first, BDS = 0) or the highest bit is transferred first (MSB first, BDS = 1), when transferring serial data. When chip select is used in master mode (SCR:MS = 0), the bit is used for communication with serial chip select pin 0.

Value	Description
0	LSB first (lowest bit is transferred first)
1	MSB first (highest bit is transferred first)

**Notes:**

- Set this bit when transmission and reception are disabled (SCR:TXE = RXE = 0).
- This bit is used in any of the following cases:
  - The chip select pins are disabled (SCSCR:CSEN3 to CSEN0 = "0000").
  - In slave mode (SCR:MS = 1).
  - When the chip select data format is disabled (ESCR:CSFE = 0).
  - When the chip select data format is enabled (ESCR:CSFE = 1) and serial chip select pin 0 is active.

**[bit1] SCKE: Serial clock output enable bit**

This bit controls the I/O ports of the serial clock.

0: The SCK pin is a general-purpose I/O port or a serial clock input pin.

1: The pin is a serial clock output pin and outputs clocks during transmission.

Value	Description
0	General-purpose I/O port or serial clock input pin
1	Serial clock output pin

**Notes:**

- To use the SCK pin as a serial clock input (SCKE = 0), set a general-purpose I/O port as an input port.
- Enable serial clock output (SCKE = 1) after setting the SCINV bit.
- Also set the SCK pin as a resource output pin. For how to set it, see "CHAPTER: I/O Port".

**[bit0] SOE: Serial data output enable bit**

This bit enables/disables the output of serial data.

0: Indicate that the SOUT pin is a general-purpose I/O port.

1: Indicate that the SOUT pin is a serial data output pin (SOUT).

Value	Description
0	General-purpose I/O port
1	Serial data output pin

**Note:**

- Set the SOT pin as a resource output pin. For how to set it, see "CHAPTER: I/O Port".



### 8.3. Serial Status Register (SSR)

The serial status register (SSR) is used to verify the transmission/reception state, verify the reception error flag, or clear the reception error flag.

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	REC	Reserved		AWC	ORE	RDRF	TDRE	TBI
ACCESS_TYPE	R0,W	R0,W0		R/W	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	00		0	0	0	1	1

\* The lower byte [bit7:0] of this register is the Extended Communication Control Register (ESCR).

#### [bit15] REC: Reception error flag clear bit

This bit clears the ORE flag of the serial status register (SSR).

- Writing "1" clears the error flag.
- Writing "0" does not have any effect.

Upon reading, "0" is always read.

Value	Description	
	Write	Read
0	No effect	"0" is always read
1	Clear the reception error flag (ORE)	

#### [bit14:13] Reserved: Reserved bits

#### [bit12] AWC: Access width control bit

This bit selects between 16- and 32-bit access when accessing the transmission data register (TDR) and the reception data register (RDR).

Value	Description
0	16-bit access
1	32-bit access

#### Notes:

- Change this bit only when transmission and reception are disabled (SCR:TXE = RXE = 0) and TDR and RDR are empty (SSR:TDRE = 1, SSR:RDRF = 0).
- If the data length is 20, 24, or 32 bits, set "1" in this bit.
- Set AWC bit of SSR register to "1" when transmission FIFO is used for SPI transfer in slave mode.

**[bit11] ORE: Overrun error flag bit**

- This bit is set to "1" when an overrun error occurs during data reception, and is cleared when "1" is written to the REC bit of the serial status register (SSR).
- A reception interrupt request is issued when the ORE bit and the SCR:RIE bit are "1".
- If this flag is set, the reception data register (RDR) data will be invalid.
- If this flag is set when the reception FIFO is used, the reception FIFO enable bits are cleared, and no reception data is stored in the reception FIFO.

Value	Description
0	No overrun error
1	Overrun error found

**[bit10] RDRF: Reception data full flag bit**

- This flag indicates the status of the reception data register (RDR).
- This flag is set to "1" when reception data is loaded to the RDR, and is cleared to "0" when the data in the reception data register (RDR) is read.
- A reception interrupt request is issued when the RDRF bit and the SCR:RIE bit are both "1".
- When the reception FIFO is used, if the reception FIFO receives a prescribed number of data items, RDRF is set to "1".
- When the reception FIFO is used, both of the following conditions are satisfied, and the reception idle state continues for at least 8 clock pulses of the baud rate clock, RDRF is set to "1".
  - The reception FIFO idle detection enable bit (FCR1:FRIIE) is set to "1".
  - The reception FIFO has not received the prescribed number of data items, and data remains in the reception FIFO.
- During an 8-clock count, the counter is reset to 0 when RDR is read, and the system starts counting the 8 clocks again.
- When the reception FIFO is used, if the reception FIFO is read, and the reception FIFO becomes empty, this bit is cleared to "0".

Value	Description
0	The reception data register RDR is empty
1	The reception data register RDR contains data

**Note:**

- When the reception FIFO is used and RDRF has become "1", resetting the reception FIFO (FCR0:FCL2, FCL1 = "1") does not cause RDRF to be set to "0". Therefore, in order to set RDRF to "0" after resetting the reception FIFO, perform a dummy reading of the reception data register during the reception disable status (SCR:RXE = "0").

**[bit9] TDRE: Transmission data empty flag bit**

- This bit indicates the status of the transmission data register (TDR).
- Writing transmission data to the TDR sets this bit to "0", indicating that the TDR contains valid data. Loading the data into the transmission shift register to start transmission sets this bit to "1", indicating that the TDR does not contain valid data. A transmission interrupt request is issued when the TDRE bit and the SCR:TIE bit are both "1".
- If the UPCL bit of the serial control register (SCR) is set to "1", the TDRE bit is set to "1".
- For the set/reset timing of the TDRE bit when using the transmission FIFO, see Section "2.4. Interrupt Generation and Flag Set Timing When the Transmission FIFO is Used".

Value	Description
0	The transmission data register (TDR) contains data
1	The transmission data register is empty

**[bit8] TBI: Transmission bus idle flag bit**

- This bit indicates that the CSIO is not performing transmission.
- If data is written to the transmission data register (TDR), this bit is set to "0".
- When the transmission data register (TDR) is empty (TDRE = 1), if the serial chip select pin is deselected and transmission is not being performed, this bit is set to "1".
- If the UPCL bit of the serial control register (SCR) is set to "1", the TDRE bit is set to "1".
- When this bit is "1", if transmission bus idle interrupts are enabled (SCR:TBIE = 1), a transmission interrupt request is output.

Value	Description
0	Transmission in progress
1	No transmission

**Note:**

- When the transmission data register (TDR) is empty (TDRE = 1), if a serial chip select error (CSE = 1) occurs, this bit is set to "1" within the baud rate period.

## 8.4. Extended Communication Control Register (ESCR)

The extended communication control register (ESCR) can be used to set the transmission/reception data length and fix the serial output to "H".

BITS_OFFSET	7	6	5	4	3	2	1	0
BITS_NAME	SOP	L3	CSFE	WT1	WT0	L2	L1	L0
ACCESS_TYPE	R0,W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

### [bit7] SOP: Serial output pin set bit

- This bit sets the serial output pin to "H". When "1" is written to this bit, the SOUT pin is set to "H". Later, "0" need not be written to this bit.
- When this bit is read, "0" is always read.

Value	Description	
	Write	Read
0	No effect	"0" is always read
1	Set the SOUT pin to "H"	

### Note:

- Do not set this bit while transmitting serial data.

**[bit5] CSFE: Serial chip select format enable bit**

This bit enables or disables the setting of a format for each serial chip select pin.

When this bit is set to "1", make the settings described below for each serial chip select pin.

- Inactive level of serial chip select
- Mark level of the serial clock
- Selection between SPI transfer and normal transfer
- Serial data transfer direction
- Data length of serial data

Value	Description
0	Set the same data format and clock format for all of the serial chip select pins
1	Set a data format and clock format for each serial chip select pin

**Notes:**

- The setting of this bit is invalid in any of the cases described below.
  - The chip select pins are disabled (SCSCR0: CSEN3 to CSEN0 = "0000").
  - In slave mode (SCR:MS=1).
- Set this bit when transmission is disabled (SCR:TXE=0).

**[bit4:3] WT1 to 0: Data transmission/reception wait select bits**

These bits specify the number of waits for the transmission or reception of continuous data in master mode. In slave mode, the operation is that for "00".

- "00": Output SCK continuously.
- "01": Output SCK after a 1-bit time wait.
- "10": Output SCK after a 2-bit time wait.
- "11": Output SCK after a 3-bit time wait.

Value		Description
0	0	0 bit
0	1	1 bit
1	0	2 bits
1	1	3 bits

**Notes:**

- When serial chip select active retention is used (SCSCR:SCAM=1) by setting transfer/reception wait selection bits (ESCR:WT[1:0]) for SPI transfer, set transfer data size to 2 or more (TBYTE ≥ 2)
- Set data transmission/reception wait select bits (ESCR:WT[1:0]) to "0b00" when serial chip select active retention is used (SCSCR:SCAM=1) in SPI mode.

**[bit6, bit2:0] L3 to 0: Data length selection bits**

These bits specify the data length of the transmission/reception data. When chip select is used in master mode (SCR:MS=0), this bit is used for communication with serial chip select pin 0.

Value				Description
L3	L2	L1	L0	
0	0	0	0	8-bit length
0	0	0	1	5-bit length
0	0	1	0	6-bit length
0	0	1	1	7-bit length
0	1	0	0	9-bit length
0	1	0	1	10-bit length
0	1	1	0	11-bit length
0	1	1	1	12-bit length
1	0	0	0	13-bit length
1	0	0	1	14-bit length
1	0	1	0	15-bit length
1	0	1	1	16-bit length
1	1	0	0	20-bit length
1	1	0	1	24-bit length
1	1	1	0	32-bit length

**Notes:**

- Settings other than those described above are prohibited.
- These bits are used for any of the following:
  - The chip select pins are disabled (SCSCR:CSEN3 to CSEN0 = "0000").
  - In slave mode (SCR:MS=1).
  - When the chip select data format is disabled (ESCR:CSFE=0).
  - When the chip select data format is enabled (ESCR:CSFE=1) and serial chip select pin 0 is active.





## 8.5. Reception Data Register/Transmission Data Register (RDR/TDR)

The reception data and transmission data registers are placed at the same address. If read, the register functions as a reception data register, while if written to, it functions as a transmission data register.

### (1) Reception Data Register (RDR)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	D31	D30	D29	D28	D27	D26	D25	D24
ACCESS_TYPE	R	R	R	R	R	R	R	R
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	D23	D22	D21	D20	D19	D18	D17	D16
ACCESS_TYPE	R	R	R	R	R	R	R	R
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	D15	D14	D13	D12	D11	D10	D9	D8
ACCESS_TYPE	R	R	R	R	R	R	R	R
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	D7	D6	D5	D4	D3	D2	D1	D0
ACCESS_TYPE	R	R	R	R	R	R	R	R
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

**[bit31:0] D31 to 0: Reception data**

The reception data register (RDR) is a 32-bit data buffer register that is used for serial data reception.

- Serial data signals transmitted to the serial input pin (SIN pin) is converted by the shift register, and the result is stored in the reception data register (RDR).
- According to the data length, the reception data is stored, starting with the lower bits, while the other bits are set to "0". Example: D7 to D0 = "45"h, D31 to D8 = 0 if the data length is 8 bits, and "45"h is received.
- The reception data full flag bit (SSR:RDRF) is set to "1" when reception data is stored to the reception data register (RDR). If reception interrupts are enabled (SCR:RIE = 1), a reception interrupt request is generated.
- Read the reception data register (RDR) when the reception data full flag bit (SSR:RDRF) is "1". The reception data full flag bit (SSR:RDRF) is such that if the serial reception data register (RDR) is read, it is automatically cleared to "0".
- If a reception error occurs (SSR:ORE), the data in the reception data register (RDR) will be invalid.
- To read the RDR, access it as described below.
  - If SSR:AWC = 0, perform 16-bit access to the lower 16 bits of the RDR.
  - If SSR:AWC = 1, perform 32-bit access.

**Notes:**

- *When the reception FIFO is used, RDRF is set to "1" when a predefined amount of data has been received by the reception FIFO.*
- *When the reception FIFO is used, RDRF is cleared to "0" once the reception FIFO is empty.*
- *If, while the reception FIFO is being used, a reception error occurs (SSR:ORE), the reception FIFO enable bit is cleared, and reception data is not stored in the reception FIFO.*
- *When AWC = 0, do not access D31 to D16.*



**(2) Transmission Data Register (TDR)**

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	D31	D30	D29	D28	D27	D26	D25	D24
ACCESS_TYPE	W	W	W	W	W	W	W	W
PROT_TYPE	-							
INITIAL_VALUE	1	1	1	1	1	1	1	1

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	D23	D22	D21	D20	D19	D18	D17	D16
ACCESS_TYPE	W	W	W	W	W	W	W	W
PROT_TYPE	-							
INITIAL_VALUE	1	1	1	1	1	1	1	1

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	D15	D14	D13	D12	D11	D10	D9	D8
ACCESS_TYPE	W	W	W	W	W	W	W	W
PROT_TYPE	-							
INITIAL_VALUE	1	1	1	1	1	1	1	1

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	D7	D6	D5	D4	D3	D2	D1	D0
ACCESS_TYPE	W	W	W	W	W	W	W	W
PROT_TYPE	-							
INITIAL_VALUE	1	1	1	1	1	1	1	1

#### [bit31:0] D31 to 0: Transmission data

The transmission data register (TDR) is a 32-bit data buffer register that is used for serial data transmission.

- If transmission is enabled (SCR:TXE = 1), and the data to be transmitted is written to the transmission data register (TDR), the transmission data is transferred to the transmission shift register, converted into serial data, and is sent out from the serial data output pin (SOUT pin).
- According to the data length, the transmission data is stored, starting with the lower bits, while the other bits are "invalid". Example: If the data length is 8 bits, and "45"h is transmitted, D7 to D0 = "45"h and D31 to D8 will be invalid.
- The transmission data empty flag (SSR:TDRE) is cleared to "0" when transmission data is written to the transmission data register (TDR).
- If the transmission FIFO is disabled or empty, the transmission data empty flag (SSR:TDRE) is set to "1" when transmission data is transferred to the transmission shift register to start transmission.
- When the transmission data empty flag (SSR:TDRE) is set to "1", the next transmission data can be written. A transmission interrupt occurs if the transmission interrupt is enabled. Write the next transmission data after the occurrence of a transmission interrupt or when the transmission data empty flag (SSR:TDRE) is set to "1".
- When the transmission data empty flag (SSR:TDRE) is set to "0", and the transmission FIFO is disabled or the transmission FIFO is full, transmission data cannot be written to the transmission data register (TDR).
- To write to the TDR, perform access as described below.
  - If SSR:AWC = 0, perform 16-bit access to the lower 16 bits of the TDR.
  - If SSR:AWC = 1, perform 32-bit access.

#### Notes:

- *The transmission data register is a write-only register, while the reception data register is a read-only register. The two registers are placed at the same address, so the written value differs from that which is read.*
- *For details on the set timing of the transmission data empty flag (SSR:TDRE) when the transmission FIFO is used, see Section "2.4. Interrupt Generation and Flag Set Timing When the Transmission FIFO is Used".*
- *When AWC = 0, do not access D31 to D16.*



### a) Relationship between the Transmission Data Register (TDR) and the Transmission Data Empty Flag

For 16-bit access (SSR:AWC=0), the TDR register has a 16-bit boundary, so that, with a single write, transmission data is stored in blocks of 16 bits. If the TDR register contains 32-bit transmission data, the transmission data empty flag (SSR:TDRE) is set to "0".

For 32-bit access (SSR:AWC=1), the TDR register has a 32-bit boundary, so that, with a single write, transmission data is stored in blocks of 32 bits.

Data Access Width	Number of Data Items Stored in the TDR Register	TBI Flag	TDRE Flag	Transmission
16-bit access (SSR:AWC = 0)	0 bit	1	1 <sup>*1</sup>	Transmission not possible
	16 bits	0		0
	32 bits			
32-bit access (SSR:AWC = 1)	0 bit	1	1	Transmission not possible
	32 bits	0	0	Transmission possible

\*1: For SSR:AWC = 0, when the number of storage data items in the TDR register is 0, if 16-bit data is written, TDRE is first set to 0 and then to 1.

## 8.6. Serial Auxiliary Control Status Register (SACSR)

The serial auxiliary control status register (SACSR) performs the following: 1. Controls serial test operation, 2. Selects a serial timer activation method, 3. Enables/disables timer interrupts, 4. Enables/disables synchronous transmission, 5. Sets the division value for the serial timer operation clock, and 6. Enables/disables the serial timer.

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	STST	Reserved	TBEEN	CSEIE	CSE	Reserved	TINT	STST
ACCESS_TYPE	R/W	R0,W0	R/W	R/W	R,W	R/W0	R,W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	00	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	TINTE	TSYNE	Reserved	TDIV3	TDIV2	TDIV1	TDIV0	TMRE
ACCESS_TYPE	R/W	R/W	R/W0	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

### [bit15] STST: Serial Test Bit

This bit enables or disables serial test mode.

When serial test mode is enabled, SOUT and SIN are connected together within the multi-function serial interface, so that the data transmitted from SOUT can be received from SIN directly.

When serial test mode is enabled, the SOUT pin is fixed to "H", and any data input to the SIN pin is ignored.

- This bit is reset when the SACSRC:STSTC bit in the clear register is set to "1".
- This bit is set when the SACSRS:STSTS bit in the set register is set to "1".

Value	Description
0	Disable serial test mode
1	Enable serial test mode

### Notes:

- This bit can be changed only when transmission and reception are disabled (SCR:TXE=0, SCR:RXE=0).
- Set this bit to "0" in slave mode (SCR:MS=1).

### [bit14] Reserved: Reserved bit

**[bit13] TBEEN: Transfer byte error enable bit**

In master mode (SCR:MS = 0), this bit enables/disables the occurrence of serial chip select errors.

For details, see Section "2.6. Chip Select Error Occurrence and Flag Set Timing".

- This bit is reset when the SACSRC:TBEENC bit in the clear register is set to "1".
- This bit is set when the SACSRS:TBEENS bit in the set register is set to "1".

Value	Description
0	In master mode (SCR:MS = 0), disable the occurrence of chip select errors
1	In master mode (SCR:MS = 0), enable the occurrence of chip select errors

**Note:**

- Change this bit when transmission and reception are disabled (SCR:TXE = RXE = "0").

**[bit12] CSEIE: Chip select error interrupt enable bit**

- This bit enables/disables the output of chip select error interrupt requests.
- If the CSEIE bit and the chip select error flag bit (CSE) are "1", a transmission interrupt request is output.
- This bit is reset when the SACSRC:CSEIEC bit in the clear register is set to "1".
- This bit is set when the SACSRS:CSEIES bit in the set register is set to "1".

Value	Description
0	Disable chip select error interrupts
1	Enable chip select error interrupts

**[bit11] CSE: Chip select error flag bit**

This bit indicates whether a chip select error has occurred.

For details, see Section "2.6. Chip Select Error Occurrence and Flag Set Timing".

When this bit is "1" and the chip select error interrupt enable bit (CSEIE) is also "1", a transmission interrupt request is output.

Writing to this bit is invalid.

This bit is reset when the SACSRC:CSEC bit in the clear register is set to "1".

Value	Description
0	No chip select error has occurred
1	A chip select error has occurred

**Notes:**

- Software reset (SCR:UPCL = "1") resets this bit to "0".
- When serial chip select is not used (SCSCR:CSEN0 = 0) in slave mode (SCR:MS = 1), this bit is not set to "1".
- When a chip select error occurs (CSE = 1), disable transmission (SCR:TXE = 0), and then write "1" to SACSRC:CSEC. To restart transmission, write "1" to SACSRC:CSEC, enable transmission (SCR:TXE = 1), and then write transmission data to the transmission data buffer (TDR).
- If noise of 1 bus clock or greater is generated in the serial chip select input during slave transmission, this bit may be set to "1". Should this occur, restart transmission after the end of master transfer.

**[bit10:9] Reserved: Reserved bits**

**[bit8] TINT: Timer interrupt flag**

If the serial timer register (STMR) and the serial timer comparison register (STMCR) match, the serial timer register (STMR) is set to "0", and this bit is set to "1".

When this bit is "1" and the timer interrupt enable bit (TINTE) is "1", a status interrupt request is issued.

If "0" is written to this bit, it is reset to "0".

Writing "1" to this bit is invalid.

- This bit is reset when the SACSRC:TINTC bit in the clear register is set to "1".

Value	Description
0	No timer interrupt request
1	Timer interrupt request issued

**Notes:**

- Software reset (SCR:UPCL = "1") resets this bit to "0".
- This bit is not set to "1" when the synchronous transmission enable bit (TSYNE) is "1".

**[bit7] TINTE: Timer interrupt enable bit**

This bit enables/disables timer interrupts to the CPU.

When this bit is "1" and the timer interrupt flag (TINT) is "1", a status interrupt request is issued.

- This bit is reset when the SACSRC:TINTEC bit in the clear register is set to "1".
- This bit is set when the SACSRS:TINTES bit in the set register is set to "1".

Value	Description
0	Disable interrupts triggered by the serial timer
1	Enable interrupts triggered by the serial timer

**[bit6] TSYNE: Synchronous transmission enable bit**

This bit enables or disables synchronous transmission.

Transmission is activated if this bit is set to "1", as well as in any of the cases described below.

- The serial timer register (STMR) and the serial timer comparison register (STMCR) values coincide in the timer-synchronized transmission.
- This bit is reset when the SACSRC:TSYNEC bit in the clear register is set to "1".
- This bit is set when the SACSRS:TSYNES bit in the set register is set to "1".

Value	Description
0	Disable synchronous transmission The serial timer is used as a timer
1	Enable synchronous transmission The serial timer is not used as a timer

**Notes:**

- This bit can be changed only when the serial timer enable bit (TMRE) is set to "0".
- When synchronous transmission is enabled (TSYNE = 1) and transmission is disabled (SCR:TXE = 0), transmission is not activated, even in any of the cases below.
  - The serial timer register (STMR) and the serial timer comparison register (STMCR) values coincide in the timer-synchronized transmission.
- In slave mode (SCR:MS = "1"), this bit is internally fixed to "0".





[bit5] Reserved: Reserved bit

**[bit4:1] TDIV3-0: Timer operation clock division bits**

These bits set the division ratio for the serial timer.

Value				Description						
				Division Ratio	$\phi=$ 8 MHz	$\phi=$ 10 MHz	$\phi=$ 16MHz	$\phi=$ 20MHz	$\phi=$ 24MHz	$\phi=$ 32MHz
0	0	0	0	$\phi$	125ns	100ns	62.5ns	50ns	41.67ns	31.25ns
0	0	0	1	$\phi/2$	250ns	200ns	125ns	100ns	83.33ns	62.5ns
0	0	1	0	$\phi/4$	500ns	400ns	250ns	200ns	166.67ns	125ns
0	0	1	1	$\phi/8$	1 $\mu$ s	800ns	500ns	400ns	333.33ns	250ns
0	1	0	0	$\phi/16$	2 $\mu$ s	1.6 $\mu$ s	1 $\mu$ s	800ns	666.67ns	500ns
0	1	0	1	$\phi/32$	4 $\mu$ s	3.2 $\mu$ s	2 $\mu$ s	1.6 $\mu$ s	1.33 $\mu$ s	1 $\mu$ s
0	1	1	0	$\phi/64$	8 $\mu$ s	6.4 $\mu$ s	4 $\mu$ s	3.2 $\mu$ s	2.67 $\mu$ s	2 $\mu$ s
0	1	1	1	$\phi/128$	16 $\mu$ s	12.8 $\mu$ s	8 $\mu$ s	6.4 $\mu$ s	5.33 $\mu$ s	4 $\mu$ s
1	0	0	0	$\phi/256$	32 $\mu$ s	25.6 $\mu$ s	16 $\mu$ s	12.8 $\mu$ s	10.67 $\mu$ s	8 $\mu$ s

$\phi$ : Bus clock

**Notes:**

- These bits can be changed only when the serial timer enable bit (TMRE) is set to "0".
- Settings other than the above are prohibited.

**[bit0] TMRE: Serial timer enable bit**

This bit enables or disables serial timer operation.

- This bit is reset when the SACSRC:TMREC bit in the clear register is set to "1".
- This bit is set when the SACSRS:TMRES bit in the set register is set to "1".

Value	Description
0	Stop the serial timer operation. When stopped, the value of the serial timer register (STMR) is retained.
1	Changing the setting of this bit from "0" to "1" initializes the serial timer register (STMR) to "0" and starts the serial timer operation.

**Note:**

- For serial timer synchronous transmission or external trigger transmission, change the setting of this bit from "0" to "1" if transmission is disabled.

## 8.7. Serial Timer Register (STMR)

The serial timer register (STMR) represents the timer value of the serial timer.

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	TM15	TM14	TM13	TM12	TM11	TM10	TM9	TM8
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	TM7	TM6	TM5	TM4	TM3	TM2	TM1	TM0
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

### [bit15:0]TM15 to 0: Timer data bits

These bits indicate the timer value of the serial timer.

During timer operation, the timer value of the serial timer is incremented by 1 for each timer operation clock (specified by SACSR:TDIV3 to 0).

**Note:**

- These bits are initialized to "0" when the timer operation starts.



## 8.8. Serial Timer Comparison Register (STMCR)

The serial timer comparison register (STMCR) sets the timer comparison value for the serial timer.

BITS_OFFSET	15	14	13	12	11	10	9	8
BITS_NAME	TC15	TC14	TC13	TC12	TC11	TC10	TC9	TC8
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BITS_OFFSET	7	6	5	4	3	2	1	0
BITS_NAME	TC7	TC6	TC5	TC4	TC3	TC2	TC1	TC0
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

### [bit15:0] TC15 to 0: Compare bits

These bits set a comparison value for the serial timer.

These bits are compared with the serial timer register (STMR). If the serial timer register (STMR) coincides with these bits when the serial timer register is updated, the serial timer register is set to "0". At this time, if synchronous transmission is disabled (SACSR:TSYNE="0"), the timer interrupt flag (SACSR:TINT) is set to "1". If synchronous transmission is enabled (SACSR:TSYNE="1"), the transmission is activated.

The interval for the operations described below is  $(\text{STMCR:TC} + 1) \times \text{timer operation clock}$  (specified by SACSR:TDIV3 to TDIV0).

- SACSR:TINT is set to "1".
- Serial-timer-synchronized transmission is activated.

### Notes:

- The timer interrupt flag (SACSR:TINT) is fixed to "1" when all of the following conditions are satisfied.
  - Synchronous transmission is disabled (SACSR:TSYNE="0").
  - 0x0000 is set in this register.
  - The timer is operating.
  - 0b0000 is set as the timer operating clock division value (SACSR:TDIV3-0).
  - Transmission is enabled (SCR:TXE=1), and transmission data exists (SSR:TDRE=0).
- This register can be changed only when the serial timer is disabled (SACSR:TMRE="0").

## 8.9. Serial Chip Select Control Status Register (SCSCR)

The serial chip select control register (SCSCR) is used to select the serial chip select start and end pins, display a serial chip select output pin, retain the active level of serial chip select, invert serial chip select, and enable/disable the output of the serial chip select pins.

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	SST1	SST0	SED1	SED0	SCD1	SCD0	SCAM	CDIV2
ACCESS_TYPE	R/W	R/W	R/W	R/W	R,WX	R,WX	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	CDIV1	CDIV0	CSLVL	CSEN3	CSEN2	CSEN1	CSEN0	CSOE
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	0	1	0	0	0	0	0

### [bit15:14] SST1-0: Serial chip select start bits

These bits select the pin at which serial chip select is to start.

If transmission is changed from the disabled state (SCR:TXE = "0") to the enabled state (SCR:TXE = "1"), and transmission data is written to TDR, the serial chip select pins become active sequentially, starting with the one set with these bits.

Value		Description
0	0	SCS0
0	1	SCS1
1	0	SCS2
1	1	SCS3

#### Notes:

- These bits can be changed only when transmission and reception are disabled (SCR:TXE = RXE = "0").
- If the serial chip select start bits (SST1, SST0) and the serial chip select end bits (SED1, SED0) are set to the same values, only the set serial chip select pin becomes active.
- In slave mode (SCR:MS = 1), the settings of these bits are invalid.
- Only those serial chip select pins for which serial chip select is enabled (CSEN = 1) become active.
- When using serial chip select in master mode (SCR:MS = "0"), enable serial chip select (CSEN = 1) for the serial chip select pin that is set with these bits.
- When rounding operation is used for normal transfer, either of the following conditions are satisfied.
  - The chip select data format is disabled (ESCR:CSFE=0).
  - When the chip select data format is enabled (ESCR:CSFE=1), use it on the following conditions.
    - Reception FIFO is enabled.
    - Set the hold delay of serial chip selection to 2 or more (SCSTR0:CSHD7 to 0  $\geq$  2).
    - Set data length of each serial chip selection to 9 or less or 10 or more.

**[bit13:12] SED1-0: Serial chip select end bits**

These bits select the pin at which serial chip select is to end.

If the serial chip select pins become active, up to the serial chip select pin that is set with these bits, the next serial chip select pin to become active is that specified with the serial chip select start bits (SST1, SST0).

Value		Description
0	0	SCS0
0	1	SCS1
1	0	SCS2
1	1	SCS3

**Notes:**

- These bits can be changed only when transmission and reception are disabled (SCR:TXE = RXE = "0").
- If the serial chip select start bits (SST1, SST0) and the serial chip select end bits (SED1, SED0) are set to the same values, only the set serial chip select pin becomes active.
- Only those serial chip select pins for which serial chip select is enabled (CSEN = 1) become active.
- In slave mode (SCR:MS = 1), the settings of these bits are invalid.
- When serial chip select is used in master mode (SCR:MS = "0"), enable serial chip select (CSEN = 1) for the serial chip select pin that is set with these bits.
- When rounding operation is used for normal transfer, either of the following conditions are satisfied.
  - The chip select data format is disabled (ESCR:CSFE=0).
  - When the chip select data format is enabled (ESCR:CSFE=1), use it on the following conditions.
    - Reception FIFO is enabled.
    - Set the hold delay of serial chip selection to 2 or more (SCSTR0:CSHD7 to 0  $\geq$  2).
    - Set data length of each serial chip selection to 9 or less or 10 or more.

**[bit11:10] SCD1-0: Serial chip select display bits**

These bits display the active serial chip select pin.

Value		Description
0	0	SCS0
0	1	SCS1
1	0	SCS2
1	1	SCS3

**Notes:**

- If the serial chip select pins are inactive, these bits display the next serial chip select pin to become active.
- These bits are set to "00"b in slave mode (SCR:MS = "1"), at a software reset (SCR:UPCL = 1), or when transmission is disabled (SCR:TXE = "0").

#### [bit9] SCAM: Serial chip select active retention bit

This bit selects whether to retain the active state of the serial chip select pin.

For details, see "Serial chip select active level retention operation (SCSCR:SCAM=1) (valid only in master mode (SCR:MS=0))" in Section "5. Operation of Serial Chip Select".

Value	Description
0	Do not retain the active state of the serial chip select pin
1	Retain the active state of the serial chip select pin

#### Notes:

- If transmission is disabled (SCR:TXE = "0") and a software reset is performed (SCR:UPCL = "1"), the serial chip select pin will be inactive regardless of the value of this bit.
- When a serial chip select error occurs (SACSR:CSE = 1), the serial chip select pin will be inactive regardless of the value of this bit.
- Set data transmission/reception wait select bits (ESCR:WT[1:0]) to "0b00" when serial chip select active retention is used (SCSCR:SCAM=1) in SPI mode.

#### [bit8:6] CDIV2 to 0: Serial chip select timing operating clock division bits

These bits set the division ratio of the serial chip select timing operating clock.

Value			Description						
			Division Ratio	$\phi$ = 8 MHz	$\phi$ = 10 MHz	$\phi$ = 16 MHz	$\phi$ = 20 MHz	$\phi$ = 24 MHz	$\phi$ = 32 MHz
0	0	0	$\phi$	125 ns	100 ns	62.5 ns	50 ns	41.67 ns	31.25 ns
0	0	1	$\phi/2$	250 ns	200 ns	125 ns	100 ns	83.33 ns	62.5 ns
0	1	0	$\phi/4$	500 ns	400 ns	250 ns	200 ns	166.67 ns	125 ns
0	1	1	$\phi/8$	1 $\mu$ s	800 ns	500 ns	400 ns	333.33 ns	250 ns
1	0	0	$\phi/16$	2 $\mu$ s	1.6 $\mu$ s	1 $\mu$ s	800 ns	666.67 ns	500 ns
1	0	1	$\phi/32$	4 $\mu$ s	3.2 $\mu$ s	2 $\mu$ s	1.6 $\mu$ s	1.33 $\mu$ s	1 $\mu$ s
1	1	0	$\phi/64$	8 $\mu$ s	6.4 $\mu$ s	4 $\mu$ s	3.2 $\mu$ s	2.67 $\mu$ s	2 $\mu$ s

$\phi$ : Bus clock

#### Notes:

- These bits can be changed only when transmission and reception are disabled (SCR:TXE = RXE = "0").
- In slave mode (SCR:MS = 1), the settings of these bits are invalid.
- Settings other than those above are prohibited.

**[bit5] CSLVL: Serial chip select level set bit**

This bit switches the inactive-time level of the serial chip select pin between "H" and "L".

This bit is used for chip select pin 0.

Value	Description
0	Set the inactive level to "L"
1	Set the inactive level to "H"

**Notes:**

- This bit can be changed only when transmission and reception are disabled (SCR:TXE = RXE = "0").
- This bit setting is used in any of the following:
  - In slave mode (SCR:MS = 1).
  - The chip select data format is disabled (ESCR:CSFE=0).
  - The chip select data format is enabled (ESCR:CSFE=1) and serial chip select pin 0 is active.

**[bit4:1] CSEN3 to 0: Serial chip select enable bits**

These bits enable or disable each serial chip select pin.

The CSEN3 bit corresponds to the SCS3 pin, the CSEN2 bit to the SCS2 pin, the CSEN1 bit to the SCS1 pin, and the CSEN0 bit to the SCS0 pin.

In slave mode (SCR:MS=1), only the CSEN0 bit is used to enable or disable serial chip pins.

Value	Description
0	Disable the operation of the serial chip select pin
1	Enable the operation of the serial chip select pin

**Notes:**

- These bits can be changed only when transmission and reception are disabled (SCR:TXE = RXE = 0).
- In master mode (SCR:MS = 0), if CSEN3 to CSEN0 are set to 0b0000, transmission/reception operations are performed regardless of the status of the serial chip select pin.
- In slave mode (SCR:MS = 1), if CSEN0 is set to "0", transmission/reception operations are performed regardless of the status of the serial chip select pin.
- Disable any serial chip select pins that are not used.
- When serial chip select is used for normal transfer in master mode, either of the following conditions are satisfied.
  - $\text{Baud rate period} / 2 \text{ [ns]} < \text{SCSTR0:CSHD7 to 0} \times \text{bus clock period} \times 2^{\text{SCSCR:CDIV2 to 0}} + 3 \times \text{bus clock period [ns]}$
  - $\text{SCSTR0:CSHD7 to 0} \times \text{bus clock period} \times 2^{\text{SCSCR:CDIV2 to 0}} + \text{SCSTR1:CSSU7 to 0} \times \text{bus clock period} \times 2^{\text{SCSCR:CDIV2 to 0}} \text{ [ns]} < \text{baud rate period} - 2 \times \text{bus clock period [ns]}$

**[bit0] CSOE: Serial chip select output enable bit**

This bit enables or disables the output of serial chip select pins.

Value	Description
0	Disable the output of all serial chip select pins
1	Enable the output of all serial chip select pins

**Notes:**

- This bit can be changed only when transmission and reception are disabled ( $SCR:TXE = RXE = "0"$ ).
- In slave mode ( $SCR:MS = "1"$ ), set this bit to "0".





## 8.10. Serial Chip Select Timing Registers (SCSTR3 to SCSTR0)

The serial chip select timing registers (SCSTR3 to SCSTR0) are used to set the serial chip select setup delay time, serial chip select hold delay time, and serial chip select deselect time.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	CSDS15	CSDS14	CSDS13	CSDS12	CSDS11	CSDS10	CSDS9	CSDS8
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	CSDS7	CSDS6	CSDS5	CSDS4	CSDS3	CSDS2	CSDS1	CSDS0
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	CSSU7	CSSU6	CSSU5	CSSU4	CSSU3	CSSU2	CSSU1	CSSU0
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	CSHD7	CSHD6	CSHD5	CSHD4	CSHD3	CSHD2	CSHD1	CSHD0
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

### [bit31:16] CSDS15 to 0: Serial chip deselect bits

These bits set the minimum time from the instant the serial chip select pin becomes inactive until the next time that the serial chip select pin becomes active.

Value							Description
CSDS15	CSDS14	CSDS13	...	CSDS2	CSDS1	CSDS0	
0	0	0	...	0	0	0	No minimum deselect time specified (5 bus clock time)
0	0	0	...	0	0	1	1x Serial chip select timing operating clock
0	0	0	...	0	1	0	2x Serial chip select timing operating clock
-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-
1	1	1	...	1	1	0	65534x Serial chip select timing operating clock
1	1	1	...	1	1	1	65535x Serial chip select timing operating clock

**Notes:**

- These bits can be changed only when transmission and reception are disabled (SCR:TXE = RXE = "0").
- In slave mode (SCR:MS = 1), the settings of these bits are invalid.
- Regardless of the deselect time setting, a minimum time of at least 5 bus clocks is required from the time the serial chip select pin becomes inactive until the next time it becomes active.
- Do not make the settings: SCSTR3-2:CSDS = 0x0001 and SCSCR:CDIV = 0b000.

**[bit15:8] CSSU7 to 0: Serial chip select setup delay bits**

These bits set the time from the instant that the serial chip select pin becomes active until the serial clock is output. If "00"h is set in these bits, the serial clock is output at the same time as the serial chip select pin becomes active.

Value								Description
CSSU7	CSSU6	CSSU5	CSSU4	CSSU3	CSSU2	CSSU1	CSSU0	
0	0	0	0	0	0	0	0	The output of the serial clock is started at the same time as the serial chip select pin becomes active.
0	0	0	0	0	0	0	1	1x Serial chip select timing operating clock
0	0	0	0	0	0	1	0	2x Serial chip select timing operating clock
-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-
1	1	1	1	1	1	1	0	254x Serial chip select timing operating clock
1	1	1	1	1	1	1	1	255x Serial chip select timing operating clock

**Notes:**

- These bits can be changed only when transmission and reception are disabled (SCR:TXE = RXE = "0").
- In slave mode (SCR:MS = 1), the settings of these bits are invalid.

**[bit7:0] CSHD7 to 0: Serial chip select hold delay bits**

These bits set the time from the instant that the output of the serial clock ends until the serial chip select pin becomes inactive.

If these bits are set to "00"h, the output of the serial clock ends at the same time as the serial chip select pin becomes inactive.

Value								Description
CSHD7	CSHD6	CSHD5	CSHD4	CSHD3	CSHD2	CSHD1	CSHD0	
0	0	0	0	0	0	0	0	The output of the serial clock ends at the same time as the serial chip select pin becomes inactive.
0	0	0	0	0	0	0	1	1x Serial chip select timing operating clock
0	0	0	0	0	0	1	0	2x Serial chip select timing operating clock
-	-	-	-	-	-	-	-	-
1	1	1	1	1	1	1	0	254x Serial chip select timing operating clock
1	1	1	1	1	1	1	1	255x Serial chip select timing operating clock

**Notes:**

- These bits can be changed only when transmission and reception are disabled (SCR:TXE = RXE = "0").
- In slave mode (SCR:MS = 1), the settings of these bits are invalid.

## 8.11. Serial Chip Select Format Registers (SCSFR2 to SCSFR0)

The serial chip select format registers (SCSFR2 to SCSFR0) are used to select the chip select active level of each serial chip select, invert the serial clock, make the settings for connecting to SPI, and set the data direction and data length of serial data output.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R1,WX							
PROT_TYPE	-							
INITIAL_VALUE	11111111							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	CS3CSLVL	CS3SCINV	CS3SPI	CS3BDS	CS3L3	CS3L2	CS3L1	CS3L0
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	1	0	0	0	0	0	0	0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	CS2CSLVL	CS2SCINV	CS2SPI	CS2BDS	CS2L3	CS2L2	CS2L1	CS2L0
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	1	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	CS1CSLVL	CS1SCINV	CS1SPI	CS1BDS	CS1L3	CS1L2	CS1L1	CS1L0
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	1	0	0	0	0	0	0	0

**[bit31:24] Reserved: Reserved bits**

**[bit23] CS3CSLVL: Bit for setting the serial chip select level of chip select 3**

When the chip select data format is enabled (ESCR:CSFE = 1), this bit selects the inactive-time level of serial chip select pin 3.

Value	Description
0	Set the inactive level to "L"
1	Set the inactive level to "H"

**Notes:**

- This bit can be changed only when transmission and reception are disabled (SCR:TXE = RXE = "0").
- In slave mode (SCR:MS = 1), the setting of this bit is invalid.
- When the chip select data format is disabled (ESCR:CSFE = 0), the setting of this bit is invalid.

**[bit22] CS3SCINV: Bit for inverting the serial clock of chip select 3**

When the chip select data format is enabled (ESCR:CSFE = 1), this bit sets the serial clock format to be assumed when serial chip select pin 3 is active.

0:

- Set the mark level of the serial clock output to "H".
- In normal transfer, transmission data is output in synchronization with a falling edge of the serial clock. In SPI transfer, it is output in synchronization with a rising edge of the serial clock.
- In normal transfer, reception data is sampled at a rising edge of the serial clock. In SPI transfer, it is sampled at a falling edge of the serial clock.

1:

- Set the mark level of the serial clock output to "L".
- In normal transfer, transmission data is output in synchronization with a rising edge of the serial clock. In SPI transfer, it is output in synchronization with a falling edge of the serial clock.
- In normal transfer, reception data is sampled at a falling edge of the serial clock. In SPI transfer, it is sampled at a rising edge of the serial clock.

Value	Description
0	Mark level "H" format
1	Mark level "L" format

**Notes:**

- This bit can be changed only when transmission and reception are disabled (SCR:TXE = RXE = "0").
- In slave mode (SCR:MS = 1), the setting of this bit is invalid.
- When the chip select data format is disabled (ESCR:CSFE = 0), the setting of this bit is invalid.

**[bit21] CS3SPI: Bit for making serial chip select pin 3 support SPI**

When the chip select data format is enabled (ESCR:CSFE = 1), this bit is used for performing communication supporting SPI when serial chip select pin 3 is active.

- 0: Perform normal synchronous communication.
- 1: Support SPI.

Value	Description
0	Normal synchronous transfer
1	SPI support

**Notes:**

- This bit can be changed only when transmission and reception are disabled (SCR:TXE = RXE = "0").
- In slave mode (SCR:MS = 1), the setting of this bit is invalid.
- When the chip select data format is disabled (ESCR:CSFE = 0), the setting of this bit is invalid.

**[bit20] CS3BDS: Bit for selecting the transfer direction of chip select pin 3**

When the chip select data format is enabled (ESCR:CSFE = 1), this bit selects the transfer of transfer serial data when serial chip select pin 3 is active, setting either transfer of the lowest bit first (LSB first, BDS = 0) or transfer of the highest bit first (MSB first, BDS = 1).

Value	Description
0	LSB first (lowest bit transferred first)
1	MSB first (highest bit transferred first)

**Notes:**

- This bit can be changed only when transmission and reception are disabled (SCR:TXE = RXE = "0").
- In slave mode (SCR:MS = 1), the setting of this bit is invalid.
- When the chip select data format is disabled (ESCR:CSFE = 0), the setting of this bit is invalid.

**[bit19:16] CS3L3 to 0: Bits for selecting the data length of serial chip select pin 3**

When the chip select data format is enabled (ESCR:CSFE = 1), these bits specify the data length of the transmission/reception data to be assumed when serial chip select pin 3 is active.

Value				Description
CS3L3	CS3L2	CS3L1	CS3L0	
0	0	0	0	8-bit length
0	0	0	1	5-bit length
0	0	1	0	6-bit length
0	0	1	1	7-bit length
0	1	0	0	9-bit length
0	1	0	1	10-bit length
0	1	1	0	11-bit length
0	1	1	1	12-bit length
1	0	0	0	13-bit length
1	0	0	1	14-bit length
1	0	1	0	15-bit length
1	0	1	1	16-bit length
1	1	0	0	20-bit length
1	1	0	1	24-bit length
1	1	1	0	32-bit length

**Notes:**

- Settings other than above are prohibited.
- These bits can be changed only when transmission and reception are disabled (SCR:TXE=RXE = "0").
- In slave mode (SCR:MS = 1), the settings of these bits are invalid.
- When the chip select data format is disabled (ESCR:CSFE = 0), the settings of these bits are invalid.

**[bit15] CS2CSLVL: Bit for setting the serial chip select level of chip select 2**

When the chip select data format is enabled (ESCR:CSFE = 1), this bit selects the inactive-time level of serial chip select pin 2.

Value	Description
0	Set the inactive level to "L"
1	Set the inactive level to "H"

**Notes:**

- This bit can be changed only when transmission and reception are disabled (SCR:TXE = RXE = "0").
- In slave mode (SCR:MS = 1), the setting of this bit is invalid.
- When the chip select data format is disabled (ESCR:CSFE = 0), the setting of this bit is invalid.

**[bit14] CS2SCINV: Bit for inverting the serial clock of chip select 2**

When the chip select data format is enabled (ESCR:CSFE = 1), this bit sets the serial clock format to be assumed when serial chip select pin 2 is active.

0:

- Set the mark level of serial clock output to "H".
- In normal transfer, transmission data is output in synchronization with a falling edge of the serial clock. In SPI transfer, it is output in synchronization with a rising edge of the serial clock.
- In normal transfer, reception data is sampled at a rising edge of the serial clock. In SPI transfer, it is sampled at a falling edge of the serial clock.

1:

- Set the mark level of serial clock output to "L".
- In normal transfer, transmission data is output in synchronization with a rising edge of the serial clock. In SPI transfer, it is output in synchronization with a falling edge of the serial clock.
- In normal transfer, reception data is sampled at a falling edge of the serial clock. In SPI transfer, it is sampled at a rising edge of the serial clock.

Value	Description
0	Mark level "H" format
1	Mark level "L" format

**Notes:**

- *This bit can be changed only when transmission and reception are disabled (SCR:TXE = RXE = "0").*
- *In slave mode (SCR:MS = 1), the setting of this bit is invalid.*
- *When the chip select data format is disabled (ESCR:CSFE = 0), the setting of this bit is invalid.*



**[bit13] CS2SPI: Bit for making serial chip select pin 2 support SPI**

When the chip select data format is enabled (ESCR:CSFE = 1), this bit is used for performing communication supporting SPI when serial chip select pin 2 is active.

- 0: Perform normal synchronous communication.
- 1: Support SPI.

Value	Description
0	Normal synchronous transfer
1	SPI support

**Notes:**

- This bit can be changed only when transmission and reception are disabled (SCR:TXE = RXE = "0").
- In slave mode (SCR:MS = 1), the setting of this bit is invalid.
- When the chip select data format is disabled (ESCR:CSFE = 0), the setting of this bit is invalid.

**[bit12] CS2BDS: Bit for selecting the transfer direction of chip select pin 2**

When the chip select data format is enabled (ESCR:CSFE = 1), this bit selects the transfer of transfer serial data when serial chip select pin 2 is active, setting either transfer of the lowest bit first (LSB first, BDS = 0) or transfer of the highest bit first (MSB first, BDS = 1).

Value	Description
0	LSB first (lowest bit transferred first)
1	MSB first (highest bit transferred first)

**Notes:**

- This bit can be changed only when transmission and reception are disabled (SCR:TXE = RXE = "0").
- In slave mode (SCR:MS = 1), the setting of this bit is invalid.
- When the chip select data format is disabled (ESCR:CSFE = 0), the setting of this bit is invalid.

**[bit11:8] CS2L3 to 0: Bits for selecting the data length of serial chip select pin 2**

When the chip select data format is enabled (ESCR:CSFE = 1), these bits specify the data length of the transmission/reception data to be assumed when serial chip select pin 2 is active.

Value				Description
CS2L3	CS2L2	CS2L1	CS2L0	
0	0	0	0	8-bit length
0	0	0	1	5-bit length
0	0	1	0	6-bit length
0	0	1	1	7-bit length
0	1	0	0	9-bit length
0	1	0	1	10-bit length
0	1	1	0	11-bit length
0	1	1	1	12-bit length
1	0	0	0	13-bit length
1	0	0	1	14-bit length
1	0	1	0	15-bit length
1	0	1	1	16-bit length
1	1	0	0	20-bit length
1	1	0	1	24-bit length
1	1	1	0	32-bit length

**Notes:**

- Settings other than those above are prohibited.
- These bits can be changed only when transmission and reception are disabled (SCR:TXE = RXE = "0").
- In slave mode (SCR:MS = 1), the settings of these bits are invalid.
- When the chip select data format is disabled (ESCR:CSFE = 0), the settings of these bits are invalid.

**[bit7] CS1CSLVL: Bit for setting the serial chip select level of chip select 1**

When the chip select data format is enabled (ESCR:CSFE = 1), this bit selects the inactive-time level of serial chip select pin 1.

Value	Description
0	Set the inactive level to "L"
1	Set the inactive level to "H"

**Notes:**

- This bit can be changed only when transmission and reception are disabled (SCR:TXE = RXE = "0").
- In slave mode (SCR:MS = 1), the setting of this bit is invalid.
- When the chip select data format is disabled (ESCR:CSFE = 0), the setting of this bit is invalid.

**[bit6] CS1SCINV: Bit for inverting the serial clock of chip select 1**

When the chip select data format is enabled (ESCR:CSFE = 1), this bit sets the serial clock format to be assumed when serial chip select pin 1 is active.

0:

- Set the mark level of serial clock output to "H".
- In normal transfer, transmission data is output in synchronization with a falling edge of the serial clock. In SPI transfer, it is output in synchronization with a rising edge of the serial clock.
- In normal transfer, reception data is sampled at a rising edge of the serial clock. In SPI transfer, it is sampled at a falling edge of the serial clock.

1:

- Set the mark level of serial clock output to "L".
- In normal transfer, transmission data is output in synchronization with a rising edge of the serial clock. In SPI transfer, it is output in synchronization with a falling edge of the serial clock.
- In normal transfer, reception data is sampled at a falling edge of the serial clock. In SPI transfer, it is sampled at a rising edge of the serial clock.

Value	Description
0	Mark level "H" format
1	Mark level "L" format

**Notes:**

- This bit can be changed only when transmission and reception are disabled (SCR:TXE = RXE = "0").
- In slave mode (SCR:MS = 1), the setting of this bit is invalid.
- When the chip select data format is disabled (ESCR:CSFE = 0), the setting of this bit is invalid.

**[bit5] CS1SPI: Bit for making serial chip select pin 1 support SPI**

When the chip select data format is enabled (ESCR:CSFE = 1), this bit is used for performing communication supporting SPI when serial chip select pin 1 is active.

- 0: Perform normal synchronous communication.
- 1: Support SPI.

Value	Description
0	Normal synchronous transfer
1	SPI support

**Notes:**

- *This bit can be changed only when transmission and reception are disabled (SCR:TXE = RXE = "0").*
- *In slave mode (SCR:MS = 1), the setting of this bit is invalid.*
- *When the chip select data format is disabled (ESCR:CSFE = 0), the setting of this bit is invalid.*

**[bit4] CS1BDS: Bit for selecting the transfer direction of chip select pin 1**

When the chip select data format is enabled (ESCR:CSFE = 1), this bit selects the transfer of transfer serial data when serial chip select pin 1 is active, setting either transfer of the lowest bit first (LSB first, BDS = 0) or transfer of the highest bit first (MSB first, BDS = 1).

Value	Description
0	LSB first (lowest bit transferred first)
1	MSB first (highest bit transferred first)

**Notes:**

- *This bit can be changed only when transmission and reception are disabled (SCR:TXE = RXE = "0").*
- *In slave mode (SCR:MS = 1), the setting of this bit is invalid.*
- *When the chip select data format is disabled (ESCR:CSFE = 0), the setting of this bit is invalid.*

**[bit3:0] CS1L3 to 0: Bits for selecting the data length of serial chip select pin 1**

When the chip select data format is enabled (ESCR:CSFE = 1), these bits specify the data length of the transmission/reception data to be assumed when serial chip select pin 1 is active.

Value				Description
CS1L3	CS1L2	CS1L1	CS1L0	
0	0	0	0	8-bit length
0	0	0	1	5-bit length
0	0	1	0	6-bit length
0	0	1	1	7-bit length
0	1	0	0	9-bit length
0	1	0	1	10-bit length
0	1	1	0	11-bit length
0	1	1	1	12-bit length
1	0	0	0	13-bit length
1	0	0	1	14-bit length
1	0	1	0	15-bit length
1	0	1	1	16-bit length
1	1	0	0	20-bit length
1	1	0	1	24-bit length
1	1	1	0	32-bit length

**Notes:**

- Settings other than above are prohibited.
- These bits can be changed only when transmission and reception are disabled (SCR:TXE = RXE = "0").
- In slave mode (SCR:MS = 1), the settings of these bits are invalid.
- When the chip select data format is disabled (ESCR:CSFE = 0), the settings of these bits are invalid.

## 8.12. Transfer Byte Registers (TBYTE3 to TBYTE0)

The transfer byte registers (TBYTE3 to TBYTE0) are used to set the number of transfer data items when the respective serial chip select pins are active.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	TBYTE3							
ACCESS_TYPE	R/W							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	TBYTE2							
ACCESS_TYPE	R/W							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	TBYTE1							
ACCESS_TYPE	R/W							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	TBYTE0							
ACCESS_TYPE	R/W							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

[bit31:24] TBYTE3[7:0]: Bits 3 for displaying the number of transfer data items

[bit23:16] TBYTE2[7:0]: Bits 2 for displaying the number of transfer data items

[bit15:8] TBYTE1[7:0]: Bits 1 for displaying the number of transfer data items

**[bit7:0] TBYTE0[7:0]: Bits 0 for displaying the number of transfer data items**

The transfer byte registers can set the number of transfer data items when the respective serial chip select pins are active. After a serial chip select pin becomes active, if the transfer of data items for which a number is set in the corresponding one of these bits is completed, the serial chip select pin becomes inactive.

Serial chip select pin 0 (SCS0) corresponds to TBYTE0, serial chip select pin 1 (SCS1) to TBYTE1, serial chip select pin 2 (SCS2) to TBYTE2, and serial chip select pin 3 (SCS3) to TBYTE3.

When serial chip select is disabled (SCSCR:CSEN3-0 = "0000"b), transfer byte register 0 (TBYTE0) is used for transmission synchronized with the timer or transmission synchronized with an external trigger. After the start of transmission due to transmission being synchronized with the timer, or transmission synchronized with an external trigger, data items for which a number is set in TBYTE0 are transferred.

If the value of one of these bits is changed during transmission (SSR:TBI = 0), the new number of transfer data items becomes valid after the end of transmission with the old number of transfer data items.

bit7:0	Description
Write	Writing to TBYTE
Read	Setting of TBYTE

**Notes:**

- If one of these bits is set to (00)h, the transfer count will be 8.
- In slave mode (SCR:MS = 1), the setting of this bit is invalid.

### 8.13. Baud Rate Generator Registers 1, 0 (BGR1, BGR0)

The baud rate generator registers 1, 0 (BGR1 and BGR0) are used to set the division ratio of the serial clock.

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved	BGR1						
ACCESS_TYPE	R0,W0	R/W						
PROT_TYPE	-							
INITIAL_VALUE	0	0000000						

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	BGR0							
ACCESS_TYPE	R/W							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

[bit15] Reserved: Reserved bit

#### [bit14:8] BGR1: Baud rate generator register 1

bit14:8	Description
Write	Write to reload counter bits 8 to 14
Read	Read the setting value of BGR1

#### [bit7:0] BGR0: Baud rate generator register 0

bit7:0	Description
Write	Write to reload counter bits 0 to 7
Read	Read the setting value of BGR0

#### Notes:

- Use 16-bit access for writing a value to baud rate generator register (BGR1, BGR0).
- If the reload value is even-numbered, the "H" width and the "L" width of the serial clock will be as described below, depending on the setting of the SCINV bit. If it is odd-numbered, the "H" width and the "L" width of the serial clock will be the same.  
When SMR:SCINV = "0", the "H" width of the serial clock will be longer by 1 cycle of the bus clock.  
When SMR:SCINV = "1", the "L" width of the serial clock will be longer by 1 cycle of the bus clock.
- Set the reload value to 2 or more in master mode.
- If the set values of the baud rate generator registers (BGR1, BGR0) are changed, the new values are reloaded when the counter value becomes "00000'h. Thus, to have the new settings take effect immediately, execute a CSIO reset (SCR:UPCL) after changing the settings of BGR1/0.
- When the reception FIFO is used, set the reception FIFO idle detection enable bit (FCR1:FRIIE) to "1", and if performing operation in slave mode, set baud rates in BGR1/0.





### 8.14. FIFO Control Register 1 (FCR1)

FIFO control register 1 (FCR1) sets the FIFO test, selects the transmission and reception FIFOs, enables the transmission FIFO interrupts, and controls the interrupt flag.

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved		Reserved	FLSTE	FRIIE	FDRQ	FTIE	FSEL
ACCESS_TYPE	R0,W0		R0,W0	R/W	R/W	R,WX	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	00		0	0	0	1	0	0

\* The lower byte [bit7:0] of this register is FIFO control register 0 (FCR0).

#### [bit15:13] Reserved: Reserved bits

#### [bit12] FLSTE: Retransmission data lost detection enable bit

This bit enables FLST bit detection.

- This bit is reset when the FCR1C:FLSTEC bit in the clear register is set to "1".
- This bit is set when the FCR1S:FLSTES bit in the set register is set to "1".

"0": Disable FLST bit detection.

"1": Enable FLST bit detection.

Value	Description
0	Disable data lost detection
1	Enable data lost detection

#### Note:

- To set this bit to "1", first set the FSET bit to "1" and then set this bit to "1".

**[bit11] FRIIE: Reception FIFO idle detection enable bit**

When the reception FIFO contains valid data, this bit enables or disables the detection of the continuation of reception idle status for the 8-bit time or longer. If the reception interrupt is enabled (SCR:RIE = 1), detection of the reception idle status triggers a reception interrupt.

- This bit is reset when the FCR1C:FRIIEC bit in the clear register is set to "1".
- This bit is set when the FCR1S:FRIIES bit in the set register is set to "1".

"0": Disable the detection of reception idle status.

"1": Enable the detection of reception idle status.

Value	Description
0	Disable the detection of reception FIFO idle
1	Enable the detection of reception FIFO idle

**Note:**

- To use the reception FIFO, set this bit to "1".

**[bit10] FDRQ: Transmission FIFO data request bit**

This bit requests transmission FIFO data.

Value "1" of this bit indicates that transmission data is requested. At this time, if transmission FIFO interrupts are enabled (FTIE = 1), a transmission FIFO interrupt request is output.

This bit is reset if the bit FCR1C:FDRQC in the clear register is set to "1".

FDRQ set conditions

- When transmission FIFO interrupt control is not used
  - FBYTE (for transmission) = 0 (Transmission FIFO is empty.)
  - Reset of the transmission FIFO
- When transmission FIFO interrupt control is used
  - FTICR setting value  $\geq$  FTICR read value (The amount of data in the transmission FIFO is at the interrupt trigger level or lower.)
  - Reset of the transmission FIFO

FDRQ reset conditions

- Writing "0" to this bit.
- If the transmission FIFO becomes full.

Value	Description
0	No transmission FIFO data request
1	Transmission FIFO data request issued

**Notes:**

- When FBYTE (for transmission)=0, the writing of "0" to this bit is prohibited.
- When the transmission FIFO is enabled, the writing of "0" is valid.
- When this bit is "0", the changing of the FSEL bit is prohibited.
- Setting "1" in this bit does not have an effect on operation.
- If a transmission interrupt is generated, and the necessary data is written to the transmission FIFO, write "0" to the FIFO transmission data request bit (FCR1:FDRQ) to clear the interrupt request.

**[bit9] FTIE: Transmission FIFO interrupt enable bit**

This bit enables a transmission FIFO interrupt. If this bit is set to "1", an interrupt occurs when the FDRQ bit is "1".

- This bit is reset when the FCR1C:FTIEC bit in the clear register is set to "1".
- This bit is set when the FCR1S:FTIES bit in the set register is set to "1".

Value	Description
0	Disable transmission FIFO interrupts
1	Enable transmission FIFO interrupts

**[bit8] FSEL: FIFO selection bit**

This bit selects the transmission and reception FIFO.

- This bit is reset when the FCR1C:FSELC bit in the clear register is set to "1".
- This bit is set when the FCR1S:FSELS bit in the set register is set to "1".

"0": Allocate as the transmission FIFO: FIFO1, and as the reception FIFO: FIFO2.

"1": Allocate as the transmission FIFO: FIFO2, and as the reception FIFO: FIFO1.

Value	Description
0	Transmission FIFO: FIFO1, reception FIFO: FIFO2
1	Transmission FIFO: FIFO2, reception FIFO: FIFO1

**Notes:**

- This bit is not cleared by a FIFO reset ( $FCL2, FCL1 = 1$ ).
- To change this bit, first disable FIFO operation ( $FCR0:0FE2, FE1 = 0$ ).
- When  $FDRQ = 0$ , the changing of this bit is prohibited.

## 8.15. FIFO Control Register 0 (FCR0)

FIFO control register 0 (FCR0) enables/disables FIFO operation, resets the FIFO, saves the read pointer, and makes the retransmission setting.

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved	FLST	FLD	FSET	FCL2	FCL1	FE2	FE1
ACCESS_TYPE	R0,W0	R,WX	R,W	R0,W	R0,W	R0,W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

**[bit7] Reserved: Reserved bit**

### **[bit6] FLST: FIFO retransmission data lost flag bit**

This bit indicates that the retransmission data of the transmission FIFO has been lost.

FLST set condition

- Data is written in the FIFO when the FLSTE bit in FIFO control register 1 (FCR1) is "1" and the write pointer of the transmission FIFO and the read pointer saved by the FSET bit coincide.

FLST reset conditions

- FIFO reset (writing "1" to FCL)
- Writing "1" to the FSET bit

If "1" is set in this bit, the data pointed to by the read pointer saved with the FSET bit is overwritten. For this reason, re-transmission with the FLD bit cannot be set even if an error occurs. When performing retransmission with this bit set to "1", reset the FIFO and then write the data to the FIFO again.

Value	Description
0	Data has not been lost
1	Data has been lost

**[bit5] FLD: FIFO pointer reload bit**

This bit reloads, to the read pointer, the data saved in the transmission FIFO by the FSET bit. This bit is used for retransmission in cases such as communication errors.

This bit is set to "0" when the retransmission setting has been completed.

- This bit is set when the FCR0S:FLDS bit in the set register is set to "1".

Value	Description
0	Do not execute reload
1	Execute reload

**Notes:**

- While this bit is set to "1", reloading to the read pointer is in progress, so writing of other than a FIFO reset is prohibited.
- Setting this bit to "1" is prohibited when a FIFO is enabled or transmission is in progress.
- First set the SCR:TIE bit and the SCR:TBIE bit to "0" and then write "1" to this bit, and after the transmission FIFO is enabled, set the SCR:TIE and SCR:TBIE bits to "1".

**[bit4] FSET: FIFO pointer saving bit**

This bit stores the read pointer of the transmission FIFO.

If the read pointer is saved before transmission starts and then a communication error occurs, retransmission is possible if the FLST bit is "0".

- This bit is set when the FCR0S:FSETS bit in the set register is set to "1".

"1": Save the current read pointer value.

"0": No effect.

Value	Description	
	Write	Read
0	Do not save	Always read "0"
1	Save the read pointer value	

**Note:**

- Set this bit to "1" when the transmission byte count (FBYTE) is 0.

**[bit3] FCL2: FIFO2 reset bit**

This bit resets FIFO2.

If this bit is set to "1", the internal status of FIFO2 is initialized.

Only the FCR0:FLST bit is initialized and the values of other bits in the FCR1/0 registers are retained.

- This bit is set when the FCR0S:FCL2S bit in the set register is set to "1".

Value	Description	
	Write	Read
0	No effect	"0" is always read
1	Reset FIFO2	

**Notes:**

- Disable transmission/reception first and then execute a FIFO2 reset.
- Execute the reset after setting the transmission FIFO interrupt enable bit to "0".
- The valid data number of the FBYTE2 register is set to 0.

**[bit2] FCL1: FIFO1 reset bit**

This bit resets FIFO1.

If this bit is set to "1", the internal status of FIFO1 is initialized.

Only the FCR0:FLST bit is initialized and the values of other bits in the FCR1/0 registers are retained.

- This bit is set when the FCR0S:FCL1S bit in the set register is set to "1".

Value	Description	
	Write	Read
0	No effect	"0" is always read
1	Reset FIFO1	

**Notes:**

- Disable transmission/reception first and then execute a FIFO1 reset.
- Execute this reset after setting the transmission FIFO interrupt enable bit to "0".
- The valid data number of the FBYTE1 register is set to 0.

**[bit1] FE2: FIFO2 operation enable bit**

This bit enables or disables the operation of FIFO2.

- Set this bit to "1" when using FIFO2.
- When FIFO2 is set as the transmission FIFO (FCR1:FSEL = 1), FIFO2 contains data when "1" is written to this bit, and CSIO is transmission-enabled (SCR:TXE = 1), transmission is started immediately. At this time, to set the SCR:TIE and SCR:TBIE bits to "1", first set them to "0" and write "1" to this bit, and then set the SCR:TIE and SCR:TBIE bits to "1".
- A reception error clears this bit to "0" if the FIFO is selected as the reception FIFO by the FSEL bit. After this, this bit cannot be set to "1" unless the reception error is cleared.
- To use FIFO1 as the transmission or reception FIFO, set this bit to "1" or "0" when the transmission or reception buffer is empty ((SSR:TDRE = "1") or (SSR:RDRF = "0")), respectively.
- To use FIFO1 as a reception FIFO, first disable reception (SCR:RXE = 0) and then change the setting of this bit when the reception buffer is empty (SSR:RDRF = "0").
- Disabling FIFO2 does not change the state of FIFO2.
- This bit is reset when the FCR0C:FE2C bit in the clear register is set to "1".
- This bit is set when the FCR0S:FE2S bit in the set register is set to "1".

Value	Description
0	Disable FIFO2 operation
1	Enable FIFO2 operation

**[bit0] FE1: FIFO1 operation enable bit**

This bit enables or disables the operation of FIFO1.

- Set this bit to "1" when using FIFO1.
- When FIFO1 is set as the transmission FIFO (FCR1:FSEL = 0), FIFO1 contains data when "1" is written to this bit, and CSIO is transmission-enabled (SCR:TXE = 1), transmission is started immediately. At this time, to set the SCR:TIE and SCR:TBIE bits to "1", first set them to "0" and write "1" to this bit, and then set the TIE and TBIE bits to "1".
- A reception error clears this bit to "0" if the FIFO is selected as the reception FIFO by the FSEL bit. Subsequently, this bit cannot be set to "1" unless the reception error is cleared.
- To use FIFO1 as the transmission or reception FIFO, set this bit to "1" or "0" when the transmission or reception buffer is empty ((SSR:TDRE = "1") or (SSR:RDRF = "0")), respectively.
- To use FIFO1 as a reception FIFO, first disable reception (SCR:RXE = 0) and then change the setting of this bit when the reception buffer is empty (SSR:RDRF = "0").
- Disabling FIFO1 does not change the state of FIFO1.
- This bit is reset when the FCR0C:FE1C bit in the clear register is set to "1".
- This bit is set when bit FCR0S:FE1S in the set register is set to "1".

Value	Description
0	Disable FIFO1 operation
1	Enable FIFO1 operation

## 8.16. FIFO Byte Register (FBYTE)

The FIFO byte register (FBYTE) indicates the valid data number of the FIFO. This register also specifies whether a reception interrupt is generated when the predefined amount of data is received by the reception FIFO.

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	FBYTE2							
ACCESS_TYPE	R,W							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	FBYTE1							
ACCESS_TYPE	R,W							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

**[bit15:8] FBYTE2[7:0]: Bits for displaying the number of data items in FIFO2**

**[bit7:0] FBYTE1[7:0]: Bits for displaying the number of data items in FIFO1**

The FBYTE register indicates the valid data number for the FIFO. The settings made with the FCR1:FSEL bit are listed below.

Value	Description	
	FIFO Selection	Data Number Indication
0	FIFO2: Reception FIFO, FIFO1: Transmission FIFO	FIFO2:FBYTE2, FIFO1:FBYTE1
1	FIFO2: Transmission FIFO, FIFO1: Reception FIFO	FIFO2:FBYTE2, FIFO1:FBYTE1

- The initial transfer count value of the FBYTE register is 0x08.
- The FBYTE for the reception FIFO contains the data number for setting the reception interrupt flag. If the transfer count that is set matches the data display of the FBYTE register, the interrupt flag (SSR:RDRF) is set to "1".
- When both of the following conditions are satisfied, the continuation of reception idle status for 8 baud rate clocks or longer sets the interrupt flag (RDRF) to "1".
  - The reception FIFO idle detection enable bit (FRIIE) is "1".
  - The number of data items in the reception FIFO does not reach the transfer count.
- During an 8-clock count, the counter is reset to 0 when RDR is read, and the system starts counting the 8 clocks again. The counter is reset to 0 when the reception FIFO is disabled. If the reception FIFO is enabled while there is data remaining in the reception FIFO, counting restarts.
- To receive data with the master operation (master reception), set the SCR:TIE bit and the SCR:TBIE bit to "0", set the number of reception data items in the FBYTE register of the transmission FIFO, and write "0" to the FCR1:FDRQ bit. Later, when the SCR:TXE bit is "1", the serial clock pulses needed for the data items that have been set are output, and as many data items as set can be received. To set the SCR:TIE bit and the SCR:TBIE bit to "1", set them to "1" after FCR1:FDRQ is set to "1".
- Whenever transmission data is written to TDR, the FBYTE of the transmission FIFO is incremented by 1.
- Whenever reception data is read from RDR, the FBYTE of the reception FIFO is decremented by 1.



**FBYTE2, FBYTE1: FIFO2 data number indication bit, FIFO1 data number indication bit**

Write	Set the transfer count
Read	Read valid data number

Read (valid data number)

Transmission: The data number written to the FIFO but not yet transmitted

Reception: The data number received by the FIFO

Write (transfer count)

Transmission: Set 0x00.

Reception: Set the data number that triggers reception interrupts.

FIFO Capacity	Data Length	SSR:AWC	Maximum FBYTE Value	Number of Data Items That Can Be Stored
16 bytes	5 to 16 bits	0	8	8
		1	4	4
	20, 24, and 32 bits	1	4	
32 bytes	5 to 16 bits	0	16	16
		1	8	8
	20, 24, and 32 bits	1	8	
64 bytes	5 to 16 bits	0	32	32
		1	16	16
	20, 24, and 32 bits	1	16	
128 bytes	5 to 16 bits	0	64	64
		1	32	32
	20, 24, and 32 bits	1	32	

**Notes:**

- Set the FBYTE of the transmission FIFO to (00)h, except when receiving data during master operation.
- When receiving data during master operation, set the number of transmission data items when the transmission FIFO is empty and the SCR:TIE and SSR:TBIE bits are both "0".
- To disable reception (SCR:RXE = 0) while receiving data during master operation, disable the transmission FIFO first and then disable transmission/reception.
- Set reception FIFO FBYTE to "2" or a larger value when block transfer. At other conditions, set "1" or a larger value in the FBYTE for the reception FIFO.
- To change the FBYTE of the reception FIFO, first disable reception and then change the setting.
- Any setting that exceeds the FIFO capacity is prohibited.

## 8.17. Transmission FIFO Interrupt Control Register (FTICR)

The transmission FIFO interrupt control register (FTICR) sets the condition for interrupts triggered by the valid data number of FIFO transmission.

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	FTICR2							
ACCESS_TYPE	R,W							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	FTICR1							
ACCESS_TYPE	R,W							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

[bit15:8] FTICR2[7:0]: Bits for displaying the number of data items in FIFO2

[bit7:0] FTICR1[7:0]: Bits for displaying the number of data items in FIFO1

The FTICR register sets an interrupt trigger level with the number of valid transmission data items (remaining amount) of the transmission FIFO. Settings with the FCR1:FSEL bit are listed below.

Value	Description	
	Selection of Transmission FIFO	Transmission FIFO Interrupt Control Register
0	FIFO1	FTICR1
1	FIFO2	FTICR2

- The initial valid data number for triggering interrupts of the FTICR register is 0x00.
- Set the number of data items that generates a transmission interrupt to the FTICR of the transmission FIFO. If the number of data items that is set matches, or becomes smaller than, the display of the number of valid data items in the transmission FIFO (FTICR or FBYTE), the interrupt flag (FDRQ) is set to "1".
- Set FTICR so that it satisfies:  $FTICR \leq \text{FIFO capacity} - 2$ .
- The read value indicates the valid data number of the FIFO.
- Transmission FIFO: Data number written to the transmission FIFO that has not yet been transmitted.
- Reception FIFO: Data number received by in the reception FIFO that has not yet been read.

**FTICR2, FTICR1: FIFO2 data number indication bit, FIFO1 data number indication bit**

Write	Set the number of valid data items that generate an interrupt
Read	Read valid data number

**Notes:**

- Any setting that would exceed the FIFO capacity is prohibited.
- The setting value cannot be read.
- As the FIFO data number for transmission, the transmission data number that has been written, minus 1, is displayed as the valid data number. This is because any attempt to write data to the TDR register stores the data in the transmission FIFO if the TDR register contains data that has yet to be transmitted. When the data in the TDR register is transmitted, data in the transmission FIFO that has yet to be transmitted is transferred to the TDR register.
- The FIFO data number for reception represents the data number received by the reception FIFO and which has not yet been read. The data number does not include the data being received by the RDR register.
- If block transfer is performed during DMA transfer, block size 1 can only be set.

## 8.18. Serial Auxiliary Control Status Clear Register (SACSRC)

The serial auxiliary control status clear register (SACSRC) can clear the bits in the serial auxiliary control status register (SACSR).

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	STSTC	Reserved	TBEENC	CSEIEC	CSEC	Reserved		TINTC
ACCESS_TYPE	R0,W	R0,W0	R0,W	R0,W	R0,W	R0,W0		R0,W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	00		0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	TINTEC	TSYNEC	Reserved					TMREC
ACCESS_TYPE	R0,W	R0,W	R0,W0					R0,W
PROT_TYPE	-							
INITIAL_VALUE	0	0	00000					0

### [bit15] STSTC: Clearing the serial test bit

Writing "1" to this bit resets the SACSR:STST to "0".

Writing "0" to this bit is invalid.

### [bit14] Reserved: Reserved bit

### [bit13] TBEENC: Clearing the transfer byte error enable bit

If "1" is written to this bit, SACSR:TBEEN is reset to "0".

Writing "0" to this bit is invalid.

### [bit12] CSEIEC: Clearing the chip select error interrupt enable bit

If "1" is written to this bit, SACSR:CSEIE is reset to "0".

Writing "0" to this bit is invalid.

### [bit11] CSEC: Clearing the chip select error flag

If "1" is written to this bit, SACSR:CSE is reset to "0".

Writing "0" to this bit is invalid.

### [bit10:9] Reserved: Reserved bits

### [bit8] TINTC: Clearing the timer interrupt flag

Writing "1" to this bit resets SACSR:TINT to "0".

Writing "0" to this bit is invalid.

**[bit7] TINTEC: Clearing the timer interrupt enable bit**

Writing "1" to this bit resets SACSR:TINTE to "0".

Writing "0" to this bit is invalid.

**[bit6] TSYNEC: Clearing the synchronous transmission enable bit**

Writing "1" to this bit resets SACSR:TSYNE to "0".

Writing "0" to this bit is invalid.

**[bit5:1] Reserved: Reserved bits****[bit0] TMREC: Clearing the serial timer enable bit**

Writing "1" to this bit resets SACSR:TMRE to "0".

Writing "0" to this bit is invalid.

## 8.19. FIFO Control Clear Register 1 (FCR1C)

FIFO control clear register 1 (FCR1C) can clear a bit in FIFO control register 1 (FCR1).

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved			FLSTEC	FRIIEC	FDRQC	FTIEC	FSELC
ACCESS_TYPE	R0,W0			R0,W	R0,W	R0,W	R0,W	R0,W
PROT_TYPE	-							
INITIAL_VALUE	000			0	0	0	0	0

\* The lower byte [bit7:0] of this register is FIFO control register 0 (FCR0C).

**[bit15:13] Reserved: Reserved bits**

**[bit12] FLSTEC: Clearing the retransmission data lost detection enable bit**

Writing "1" to this bit resets FCR1:FLSTE to "0".

Writing "0" to this bit is invalid.

**[bit11] FRIIEC: Clearing the reception FIFO idle detection enable bit**

Writing "1" to this bit resets FCR1:FRIIE to "0".

Writing "0" to this bit is invalid.

**[bit10] FDRQC: Clearing the transmission FIFO data request bit**

Writing "1" to this bit resets FCR1:FDRQ to "0".

Writing "0" to this bit is invalid.

**[bit9] FTIEC: Clearing the transmission FIFO interrupt enable bit**

Writing "1" to this bit resets FCR1:FTIE to "0".

Writing "0" to this bit is invalid.

**[bit8] FSELC: Clearing the FIFO selection bit**

Writing "1" to this bit resets FCR1:FSEL to "0".

Writing "0" to this bit is invalid.



## 8.20. FIFO Control Clear Register 0 (FCR0C)

FIFO control clear register 0 (FCR0C) can clear a bit in FIFO control register 0 (FCR0).

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved						FE2C	FE1C
ACCESS_TYPE	R0,W0						R0,W	R0,W
PROT_TYPE	-							
INITIAL_VALUE	000000						0	0

**[bit7:2] Reserved: Reserved bits**

**[bit1] FE2C: Clearing the FIFO2 operation enable bit**

Writing "1" to this bit resets FCR0:FE2 to "0".

Writing "0" to this bit is invalid.

**[bit0] FE1C: Clearing the FIFO1 operation enable bit**

Writing "1" to this bit resets FCR0:FE1 to "0".

Writing "0" to this bit is invalid.

## 8.21. Serial Auxiliary Control Status Set Register (SACSR)

The serial auxiliary control status set register (SACSR) can clear a bit in the serial auxiliary control status register (SACSR).

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	STSTS	Reserved	TBEENS	CSEIES	Reserved			
ACCESS_TYPE	R0,W	R0,W0	R0,W	R0,W	R0,W0			
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0000			

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	TINTES	TSYNES	Reserved					TMRES
ACCESS_TYPE	R0,W	R0,W	R0,W0					R0,W
PROT_TYPE	-							
INITIAL_VALUE	0	0	00000					0

### [bit15] STSTS: Setting the serial test bit

Writing "1" to this bit sets the SACSR:STST to "1".

Writing "0" to this bit is invalid.

### [bit14] Reserved: Reserved bit

### [bit13] TBEENS: Transfer byte error enable set bit

Writing "1" to this bit sets SACSR:TBEEN to "1".

Writing "0" to this bit is invalid.

### [bit12] CSEIES: Chip select error interrupt enable set bit

Writing "1" to this bit sets SACSR:CSEIE to "1".

Writing "0" to this bit is invalid.

### [bit11:8] Reserved: Reserved bits

### [bit7] TINTES: Setting the timer interrupt enable bit

Writing "1" to this bit sets SACSR:TINTE to "1".

Writing "0" to this bit is invalid.

### [bit6] TSYNES: Setting the synchronous transmission enable bit

Writing "1" to this bit sets SACSR:TSYNE to "1".

Writing "0" to this bit is invalid.

### [bit5:1] Reserved: Reserved bits



**[bit0] TMRES: Setting the serial timer enable bit**

Writing "1" to this bit sets SACSR:TMRE to "1".

Writing "0" to this bit is invalid.

## 8.22. FIFO Control Set Register 1 (FCR1S)

FIFO control set register 1 (FCR1S) can set a bit in FIFO control register 1 (FCR1).

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved			FLSTES	FRIIES	Reserved	FTIES	FSELS
ACCESS_TYPE	R0,W0			R0,W	R0,W	R0,W0	R0,W	R0,W
PROT_TYPE	-							
INITIAL_VALUE	000			0	0	0	0	0

\* The lower byte [bit7:0] of this register is FIFO control set register 0 (FCR0S)

**[bit15:13] Reserved: Reserved bits**

**[bit12] FLSTES: Setting the retransmission data lost detection enable bit**

Writing "1" to this bit sets FCR1:FLSTE to "1".

Writing "0" to this bit is invalid.

**[bit11] FRIIES: Setting the reception FIFO idle detection enable bit**

Writing "1" to this bit sets FCR1:FRIIE to "1".

Writing "0" to this bit is invalid.

**[bit10] Reserved: Reserved bit**

**[bit9] FTIES: Setting the transmission FIFO interrupt enable bit**

Writing "1" to this bit sets FCR1:FTIE to "1".

Writing "0" to this bit is invalid.

**[bit8] FSELS: Setting the FIFO selection bit**

Writing "1" to this bit sets FCR1:FSEL to "1".

Writing "0" to this bit is invalid.



### 8.23. FIFO Control Set Register 0 (FCR0S)

FIFO control set register 0 (FCR0S) can set a bit in FIFO control register 0 (FCR0).

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved		FLDS	FSETS	FCL2S	FCL1S	FE2S	FE1S
ACCESS_TYPE	R0,W0		R0,W	R0,W	R0,W	R0,W	R0,W	R0,W
PROT_TYPE	-							
INITIAL_VALUE	00		0	0	0	0	0	0

**[bit7:6] Reserved: Reserved bits**

**[bit5] FLDS: Setting the FIFO pointer reload bit**

Writing "1" to this bit sets FCR0:FLD to "1".

Writing "0" to this bit is invalid.

**[bit4] FSETS: Setting the FIFO pointer saving bit**

Writing "1" to this bit sets FCR0:FSET to "1".

Writing "0" to this bit is invalid.

**[bit3] FCL2S: Setting the FIFO2 reset bit**

Writing "1" to this bit sets FCR0:FCL2 to "1".

Writing "0" to this bit is invalid.

**[bit2] FCL1S: Setting the FIFO1 reset bit**

Writing "1" to this bit sets FCR0:FCL1 to "1".

Writing "0" to this bit is invalid.

**[bit1] FE2S: Setting the FIFO2 operation enable bit**

Writing "1" to this bit sets FCR0:FE2 to "1".

Writing "0" to this bit is invalid.

**[bit0] FE1S: Setting the FIFO1 operation enable bit**

Writing "1" to this bit sets FCR0:FE1 to "1".

Writing "0" to this bit is invalid.

## 9. Precautions for Using

Precautions for using CSIO are provided below.

### Notes on DMA Transfer

By using the interrupt factor generation of CSIO, the DMA controller can be activated.

Before activating the DMA controller from CSIO, make settings for the DMA controller. If DMA transfer is performed, only 1 is supported as the entire length (block count) of 1 block. (DMAi\_An:BC=0)

For the settings of the DMA controller and for the details, see "CHAPTER: DMA Controller".





## CHAPTER 37: LIN Interface (v2.1) (LIN Communication Control Interface (v2.1))

This chapter describes the LIN communication function supported by operation mode 3 of the multi-function serial interface functions. The LIN communication function has the following 2 modes: Manual mode, which allows the transmission/reception of the LIN header section using interrupt functions, and the assist mode, which allows automatic transmission/reception of the LIN header.

---

1. Overview
2. Interrupts
3. Serial Timer Operation
4. Test Mode
5. Dedicated Baud Rate Generator
6. Operation
7. Setup Procedure and Program Flow for Operation Mode 3 (LIN Communication Mode)
8. Registers
9. Precautions for Using



## 1. Overview

### 1.1. Manual Mode

The LIN interface (v2.1) (LIN communication control interface (v2.1)) provides the functions to support the LIN bus. In addition, this interface incorporates transmission/reception FIFOs (maximum 64 bytes each).

**LIN Interface (v2.1) (LIN Communication Control Interface (v2.1)) Functions (Manual Mode)**

	Item	Function
1	Data Buffer	<ul style="list-style-type: none"> <li>- Full-duplex double buffering (when FIFO is not used)</li> <li>- Transmission and reception FIFOs (maximum 64 bytes each) (when FIFO is used)</li> </ul>
2	Serial Input	Over sampling is performed by the bus clock 3 times. The reception value is determined by the rule of majority.
3	Transfer Mode	Asynchronous
4	Baud Rate	<ul style="list-style-type: none"> <li>- Dedicated baud rate generator is provided (consisting of a 15-bit reload counter).</li> <li>- The external clock can be adjusted by the reload counter.</li> <li>- Auto baud rate adjustment with Sync Field reception</li> </ul>
5	Data Length	8 bits
6	Signaling Method	NRZ (Non Return to Zero)
7	Start Bit Detection	Synchronized with falling edges of the start bit
8	Reception Error Detection	<ul style="list-style-type: none"> <li>- Framing error</li> <li>- Overrun error</li> </ul>
9	Interrupt Request	<ul style="list-style-type: none"> <li>- Reception interrupt (reception completion, framing error, overrun error)</li> <li>- Transmission interrupt (transmission data empty, transmission bus idle)</li> <li>- Status interrupts (LIN Break Field detection and serial timer interrupt)</li> <li>- Interrupt request to ICU (LIN Sync Field detection: LSYN)</li> <li>- Transmission FIFO interrupt (when the transmission FIFO is not higher than the interrupt threshold or when the transmission FIFO is empty)</li> <li>- DMA transfer is supported for both transmission and reception.</li> </ul>
10	Timer Function	<ul style="list-style-type: none"> <li>- A 16-bit serial timer is available.</li> <li>- A division value can be selected for the operation clock (1 to 256 divisions).</li> <li>- Activation by an external trigger is available.</li> </ul>
11	LIN Bus Option	<ul style="list-style-type: none"> <li>- Support for LIN Protocol Revision 2.1</li> <li>- Master device operation</li> <li>- Slave device operation</li> <li>- LIN Break Field generation (changeable from 13 to 16 bits)</li> <li>- LIN Break delimiter generation (changeable from 1 to 4 bits)</li> <li>- LIN Break Field detection</li> <li>- Detection of the start/stop edge of the LIN Sync Field connected to an input capture</li> </ul>
12	FIFO Option	<ul style="list-style-type: none"> <li>- Transmission and reception FIFOs are provided (maximum capacity 64 bytes for transmission FIFO: 64 bytes for reception FIFO).</li> <li>- Transmission FIFO and reception FIFO can be selected.</li> <li>- Transmission data can be retransmitted.</li> <li>- The timing of the reception FIFO interrupt can be changed by software.</li> <li>- Independent FIFO reset is supported.</li> </ul>

**Note:**

- The LIN Wake Up function is not supported.

## 1.2. Assist Mode

The LIN interface (v2.1) (LIN communication control interface (v2.1)) provides the functions to support the LIN bus. The header section can be automatically transmitted/detected in LIN communication.

In addition, this interface incorporates transmission/reception FIFOs (maximum 64 bytes each).

**LIN Interface (v2.1) (LIN Communication Control Interface (v2.1)) Functions (Assist Mode)**

	Item	Function
1	Data Buffer	<ul style="list-style-type: none"> <li>Full-duplex double buffering (when FIFO is not used)</li> <li>Transmission and reception FIFOs (maximum 64 bytes each)(when a FIFO is used)</li> </ul>
2	Serial Input	Over sampling is performed by the bus clock 3 times. The reception value is determined by the rule of majority.
3	Transfer Mode	Asynchronous
4	Baud Rate	<ul style="list-style-type: none"> <li>Dedicated baud rate generator is provided (consisting of 15-bit reload counter).</li> <li>The external clock can be adjusted by the reload counter.</li> <li>Auto baud rate adjustment with Sync Field reception</li> </ul>
5	Data Length	8 bits
6	Signaling Method	NRZ (Non Return to Zero)
7	Start Bit Detection	Synchronized with falling edges of the start bit
8	Reception Error Detection	<ul style="list-style-type: none"> <li>An error is detected by a self-check on the transmission side <ul style="list-style-type: none"> <li>LIN bus error</li> </ul> </li> <li>An error is detected by a self-check on the transmission side or is detected at the reception side - <ul style="list-style-type: none"> <li>Framing error</li> <li>Overrun error</li> <li>LIN ID parity error</li> <li>LIN checksum error</li> </ul> </li> <li>An error is detected on the reception side when auto baud rate adjustment is disabled - <ul style="list-style-type: none"> <li>LIN Sync Data error</li> </ul> </li> </ul>
9	Interrupt Request	<ul style="list-style-type: none"> <li>Transmission interrupts <ul style="list-style-type: none"> <li>Data transmission interrupt (transmission data empty, transmission bus idle)</li> <li>Transmission FIFO interrupt (when the transmission FIFO is not higher than the interrupt threshold or when the transmission FIFO is empty)</li> </ul> </li> <li>Reception interrupt <ul style="list-style-type: none"> <li>Data reception interrupt (reception completion)</li> <li>Reception FIFO interrupt (when reception FIFO is equal to or higher than the interrupt threshold)</li> <li>Various types of error interrupts (LIN bus error, LIN Sync Data error, LIN ID parity error, framing error, overrun error, and LIN checksum error)</li> </ul> </li> <li>Status interrupt <ul style="list-style-type: none"> <li>Auto header completion interrupt</li> <li>Sync Field detection interrupt</li> <li>Checksum calculation completion interrupt</li> </ul> </li> <li>DMA Transfer is supported for both transmission and reception.</li> </ul>
10	Timer Function	<ul style="list-style-type: none"> <li>A 16-bit serial timer is provided.</li> <li>A division value can be selected for the operation clock (1 to 256 divisions).</li> </ul>
11	LIN Bus Option	<ul style="list-style-type: none"> <li>Support for LIN Protocol Revision 2.1</li> <li>Automatic transmission/reception of the header of the master/slave device <ul style="list-style-type: none"> <li>LIN Break Field generation (changeable from 13 to 20 bits)</li> <li>LIN Break delimiter generation (changeable from 1 to 4 bits)</li> <li>Automatic generation of Sync Field and automatic check of data value (0x55)</li> <li>Automatic parity generation/check in the ID Field</li> <li>Automatic generation/check of a checksum (support for standard/extended checksum)</li> </ul> </li> </ul>





	Item	Function
12	FIFO Option	<ul style="list-style-type: none"> <li>- Transmission and reception FIFOs are provided (maximum capacity 64 bytes for transmission FIFO: 64 bytes for reception FIFO).</li> <li>- Transmission FIFO and reception FIFO can be selected.</li> <li>- Transmission data can be retransmitted.</li> <li>- The timing of the transmission/reception FIFO interrupt can be changed by software.</li> <li>- Independent FIFO reset is supported.</li> </ul>
13	LIN Communication Test Function	<ul style="list-style-type: none"> <li>- Serial communication test function</li> <li>- Pseudo error generation functions (LIN bus error, LIN ID parity error, LIN checksum error, LIN Sync Data error, and framing error)</li> </ul>

**Note:**

- *The LIN Wake Up function is not supported.*

## 2. Interrupts

### 2.1. Manual Mode

The LIN interface (v2.1) can generate reception interrupts, transmission interrupts, and status interrupts. In manual mode, an interrupt request can be generated by any of the following factors.

- When reception data is set in the reception data register (RDR) or when a reception error occurs.
- When transmission is started after transmission data is transferred from the transmission data register (TDR) to the transmission shift register.
- Transmission bus idle (no transmission operation)
- Transmission FIFO data request
- LIN Break Field detection
- LIN Sync Field detection
- The serial timer comparison value (STMCR) matches the serial timer value (STMR).

#### LIN Interface (v2.1) Interrupts (Manual Mode)

Table 2-1 shows the interrupt control bits of the LIN interface (v2.1), together with the interrupt factors in manual mode.

**Table 2-1 Interrupt Control Bits and Interrupt Factors of LIN Interface (v2.1) (Manual Mode)**

Interrupt Type	Interrupt Request Flag Bit	Flag Register	Interrupt Factor	Interrupt Factor Enable Bit	Clearing of Interrupt Request Flag
Reception	RDRF	SSR	1-byte reception	SCR:RIE	Reading of reception data (RDR)
			Reception of as much data as the amount set in FBYTE		Reading of reception data until the reception FIFO becomes empty
			Detection of reception idle for 8-bit time or longer while the FRIIE bit is "1" and the reception FIFO contains valid data		
	ORE	SSR	Overrun error		Writing "1" to the reception error flag clear bit (SSR:REC)
	FRE	SSR	Framing error		
Transmission	TDRE	SSR	The transmission register is empty.	SCR:TIE	Writing to the transmission data (TDR), or writing "1" to the transmission FIFO operation enable bit (FCR0:FE1 or FCR0:FE2) (retransmission) while the bit value is "0" and the transmission FIFO contains valid data* <sup>1</sup>
	TBI	SSR	No transmission operation	SCR:TBIE	Writing to the transmission data (TDR), writing "1" to the LIN Break Field setting bit (LBR), or writing "1" to the transmission FIFO operation enable bit (FCR0:FE1 or FCR0:FE2) (retransmission) while the bit value is "0" and the transmission FIFO contains valid data* <sup>1</sup>



Interrupt Type	Interrupt Request Flag Bit	Flag Register	Interrupt Factor	Interrupt Factor Enable Bit	Clearing of Interrupt Request Flag
	FDRQ	FCR1	The amount of data in the transmission FIFO is equal to or smaller than the FTICR setting value or it is empty.	FCR1:FTIE	Writing "0" to the FIFO transmission data request bit (FCR1:FDRQ) , or transmission FIFO full
Status (Manual Mode)	LBD	SSR	LIN Break Field detection	ESCR:LBIE	Writing "0" to the SSR:LBD bit
	SFD	SACSR	Sync Field detection	SACSR:SFDE	Writing "0" to the Sync Field detection flag bit (SACSR:SFD)
	TINT	SACSR	Coincidence of the serial timer register (STMR) and the serial timer comparison register (STMCR) values	SACSR:TINTE	Writing "0" to the timer interrupt flag bit (SACSR:TINT)
Input Capture* <sup>2</sup>	ICP0	ICS0	1st falling edge of the LIN Sync Field	ICS0:ICE0	Disabling the ICP0
	ICP0	ICS0	5th falling edge of the LIN Sync Field		

\*1: Set the TIE bit to "1" after the TDRE bit becomes "0".

\*2: When auto baud rate adjustment is enabled (SACSR:AUTE=1), this interrupt is not generated.

### 2.1.1. Occurrence of Reception Interrupts and Flag Set Timing

There are 2 reception interrupt types: reception completion (SSR:RDRF) and reception error (SSR:ORE, FRE).

#### Occurrence of Reception Interrupts and Flag Set Timing

Detection of the first stop bit stores the reception data to the reception data register (RDR). Related flags are set upon the completion of reception (SSR:RDRF=1) or upon a reception error (SSR:ORE, FRE=1). At this time, a reception interrupt occurs if reception interrupt is enabled (SCR:RIE=1).

**Note:**

- If a reception error occurs, the data of the reception data register (RDR) is invalid.

Figure 2-1 Set Timing of RDRF (Reception Data Full) Flag Bit

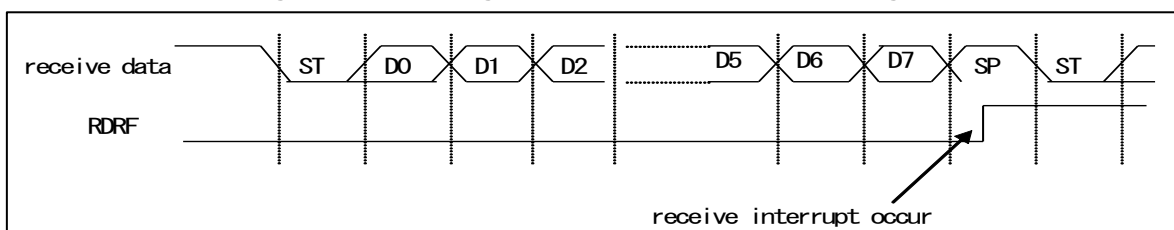
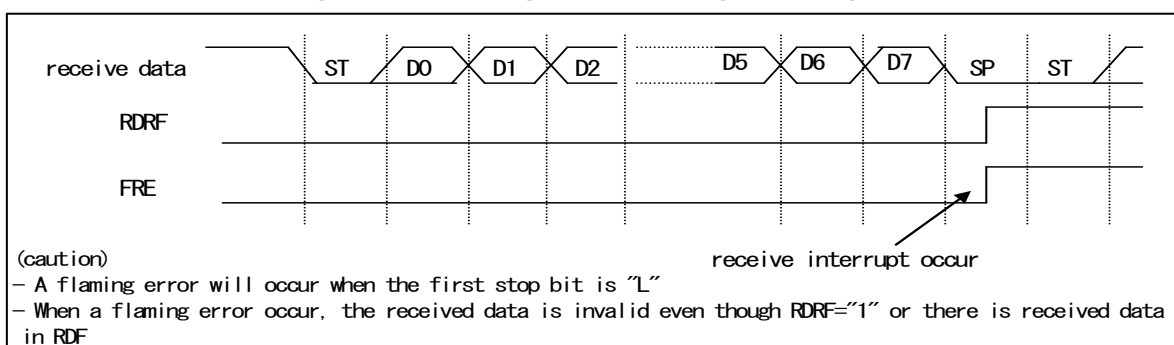


Figure 2-2 Set Timing of FRE (Framing Error) Flag Bit

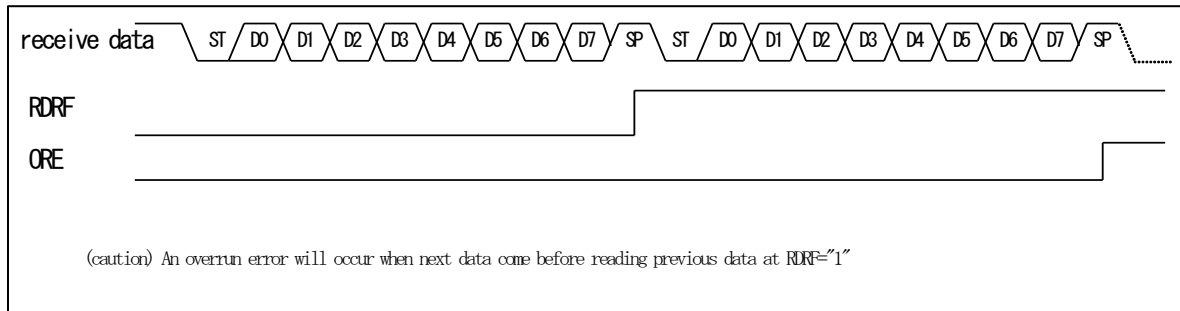


**Note:**

- During data reception, the following occurs if a falling edge of serial data is detected at the same time as, or 1 or 2 bus clocks earlier than, the sampling point of stop bit: The edge becomes invalid and the data that follows may not be received normally. Consecutive frames must have intervals between them.



**Figure 2-3 Set Timing of ORE (Overrun Error) Flag Bit**



## 2.1.2. Occurrence of Interrupts and Flag Set Timing when Using Reception FIFO

An interrupt occurs during the use of the reception FIFO when as much data as the amount set in the FIFO byte register (FBYTE) is received.

### Occurrence of Reception Interrupts and Flag Set Timing when Using Reception FIFO

The occurrence of interrupts during the use of the reception FIFO is determined by the value set in the FBYTE register.

- If as much transfer data as the amount set in the FBYTE register is received, the reception data full flag (SSR:RDRF) in the serial status register is set to "1". At this time, if reception interrupts are enabled (SCR:RIE), a reception interrupt is generated.
- When the amount of data in the reception FIFO is "1" or greater and both of the following conditions are satisfied, the continuation of the reception idle status for 8 baud rate clocks or longer sets the interrupt flag (SSR:RDRF) to "1".
  - The reception FIFO idle detection enable bit (FCR1:FRIDE) is "1".
  - The number of data items in the reception FIFO does not reach the transfer count.
- During an 8-clock count, the counter is reset to 0 when RDR is read, and the system starts counting the 8 clocks again. The counter is reset to 0 when the reception FIFO is disabled. After the reception FIFO is enabled while there is still data in the reception FIFO, counting restarts.
- When the reception FIFO becomes empty as a result of reading the reception data (RDR), the reception data full flag (SSR:RDRF) is cleared.
- If the reception valid data count becomes equal to the value of the FIFO capacity, the reception of the next data triggers an overrun error (SSR:ORE=1).

Figure 2-4 Occurrence Timing of Reception Interrupt When Reception FIFO is Used

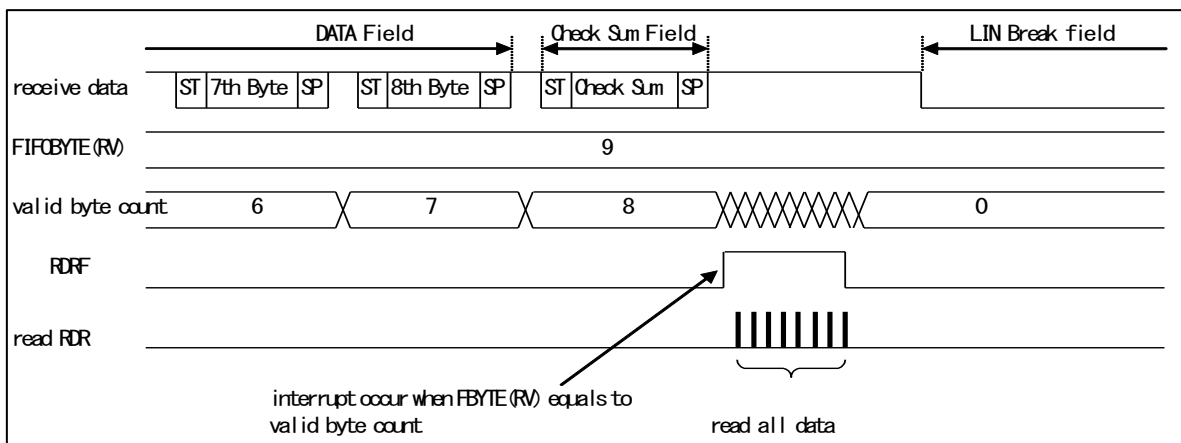
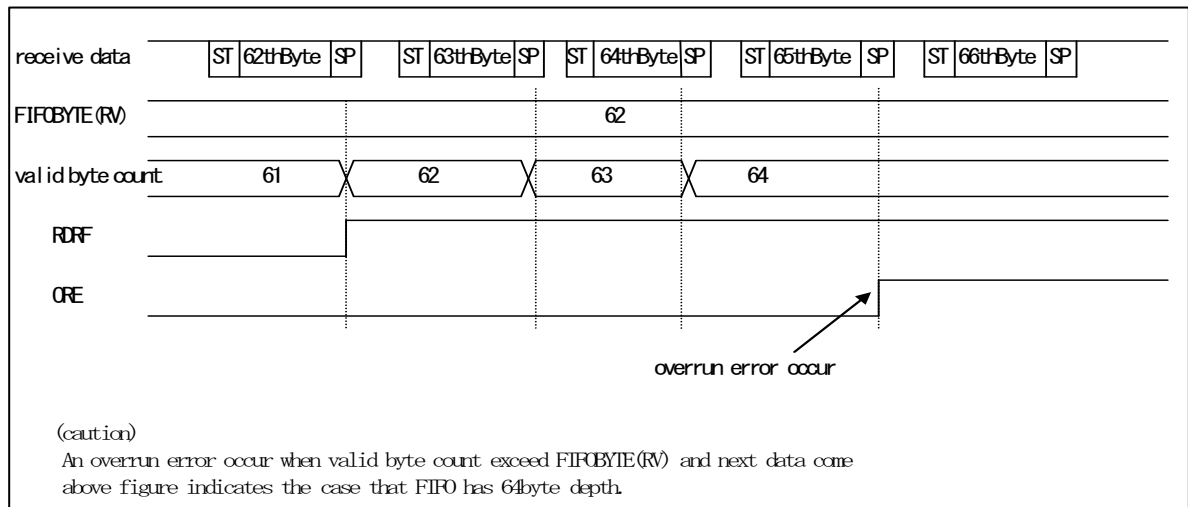




Figure 2-5 Set Timing of ORE (Overrun Error) Flag Bit



2.1.3. Occurrence of Transmission Interrupts and Flag Set Timing

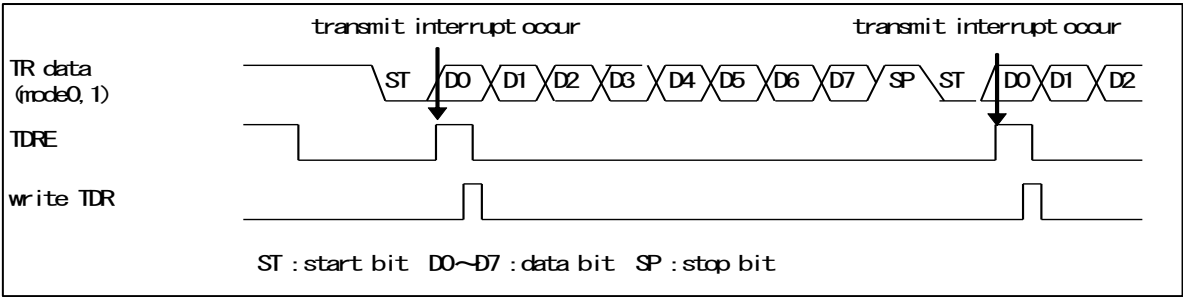
A data transmission interrupt occurs in the following cases: 1. The transmission data is transferred from the transmission data register (TDR) to the transmission shift register (SSR:TDRE=1) for start of the data transmission. 2. Transmission is not performed (SSR:TBI=1).

Occurrence of Transmission Interrupts and Flag Set Timing

a) Set Timing of Transmission Data Empty Flag (TDRE)

The transfer of data from the transmission data register (TDR) to the transmission shift register allows the next data to be written (SSR:TDRE=1). At this time, a transmission interrupt occurs if transmission interrupt is enabled (SCR:TIE=1). The TDRE bit is a read-only bit, and the SSR:TDRE bit is cleared to "0" when data is written to the transmission data register (TDR).

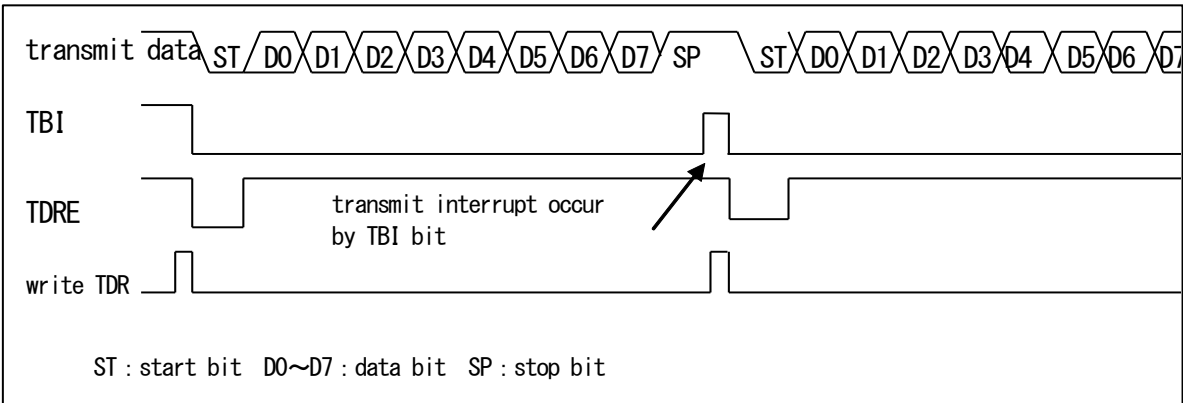
Figure 2-6 Set Timing of Transmission Data Empty Flag (SSR:TDRE)



b) Set Timing of Transmission Bus Idle Flag (TBI)

The SSR:TBI bit is set to "1" during those periods in which the transmission data register is empty (SSR:TDRE=1) and transmission is not performed. At this time, a transmission interrupt occurs if transmission bus idle interrupt is enabled (SCR:TBIE=1). Setting transmission data to the transmission data register (TDR) clears the TBI bit and the transmission interrupt request.

Figure 2-7 Set Timing of Transmission Bus Idle Flag (TBI)







### 2.1.4. Occurrence of Interrupts and Flag Set Timing When Using Transmission FIFO

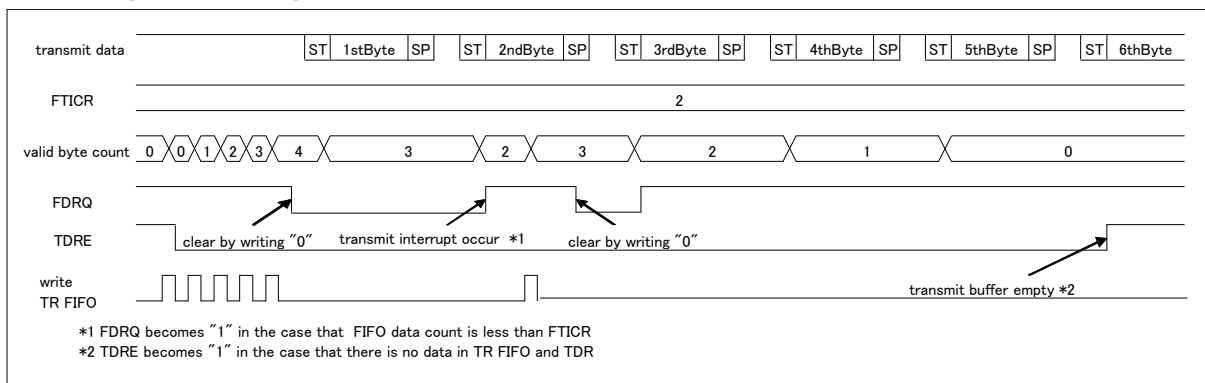
An interrupt occurs during the use of the transmission FIFO when the data count in the transmission FIFO is equal to or less than the value set in the transmission FIFO interrupt control register (FTICR).

#### Occurrence of Transmission Interrupts and Flag Set Timing When Using Transmission FIFO

The occurrence of interrupts during the use of the transmission FIFO is determined by the value set in the FTICR register.

- When the data count in the transmission FIFO is equal to or less than the value set in the FTICR register, the FIFO transmission data request bit (FCR1:FDRQ) is set to "1".  
At this time, a transmission interrupt occurs if the FIFO transmission interrupt is enabled (FCR1:FTIE=1).
- When you write the necessary data to the transmission FIFO after a transmission interrupt occurs, write "0" to the FIFO transmission data request bit (FCR1:FDRQ) to clear the interrupt request.
- The FIFO transmission data request bit (FCR1:FDRQ) is set to "0" when the transmission FIFO becomes full.
- Whether data is present in the transmission FIFO can be checked by reading the FIFO byte register (FBYTE). This can also be checked by reading the transmission FIFO interrupt control register (FTICR).  
FBYTE = 0x00 indicates that the transmission FIFO does not contain data.

**Figure 2-8 Timing of Transmission Interrupt Occurrence When Transmission FIFO is Used**





2.1.5.      **Occurrence of Timer Interrupts and Flag Set Timing**

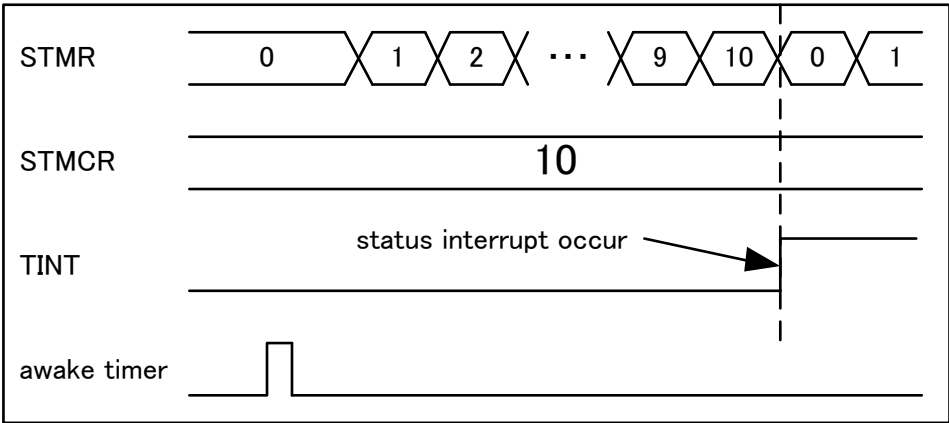
A timer interrupt occurs when the value of the serial timer register (STMR) coincides with that of the serial timer comparison register (STMCR).

**Timer Interrupt Generation and Flag Set Timing**

The timer interrupt flag (SACSR:TINT) is set to "1" when the value of the serial timer register (STMR) coincides with that of the serial timer comparison register.

At this time, a status interrupt occurs if the timer interrupt is enabled (SACSR:TINTE=1).

**Figure 2-9 Occurrence Timing of Timer Interrupt**

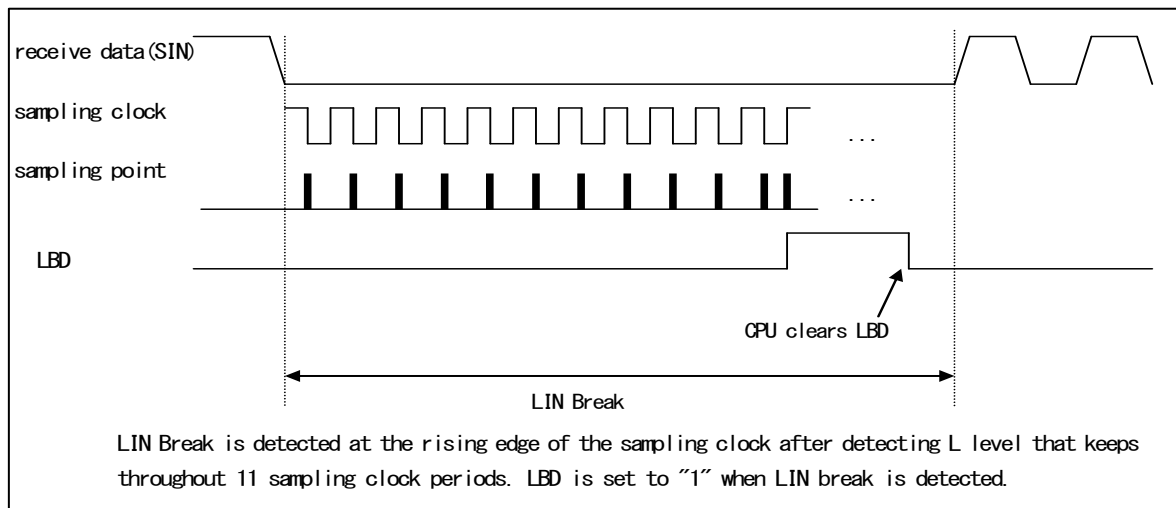




### 2.1.6. Set Timing of LIN Break Field Detection Flag (LBD)

The LBD bit is set to "1" when the serial input (SIN) has "0" input for an interval of 11 bits or more. At this time, a status interrupt occurs if the LIN Break Field interrupt is enabled (ESCR:LBIE=1).

Figure 2-10 Set Timing of LBD (LIN Break Field Detection) Flag



**Note:**

- During the reception of a LIN Break Field, a framing error (SSR:FRE=1) is detected before the detection of a LIN Break Field if reception is enabled (SCR:RXE=1).

### 2.1.7. Occurrence of Sync Field Detection Interrupt and Flag Set Timing

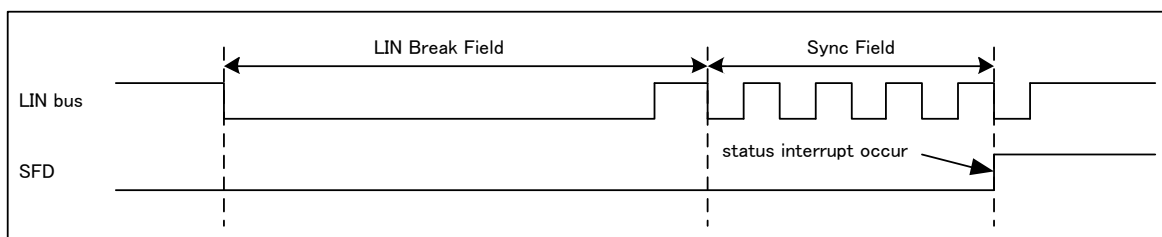
A Sync Field detection interrupt occurs if the detection of the Sync Field is complete.

#### Occurrence of Sync Field Detection Interrupt and Flag Set Timing

The Sync Field detection flag (SACSR:SFD) is set to "1" once the 5th falling edge of the LIN bus is detected in the Sync Field when auto baud rate adjustment is enabled (SACSR:AUTE=1).

At this time, a status interrupt occurs if the Sync Field interrupt is enabled (SACSR:SFDE=1).

Figure 2-11 Timing of Sync Field Detection Interrupt





## 2.2. Assist Mode

The LIN interface (v2.1) can generate reception interrupts, transmission interrupts, and status interrupts. In assist mode, an interrupt request can be generated due to the following factors:

- When reception data is set in the reception data register (RDR) or when a reception error occurs.
- When transmission is started after transmission data is transferred from the transmission data register (TDR) to the transmission shift register.
- Transmission bus idle (no transmission operation)
- Transmission FIFO data request
- LIN Break Field detection
- LIN Sync Field detection
- The serial timer comparison value (STMCR) matches the serial timer value (STMR)
- Detection of the completion of the LIN auto header or the checksum calculation

### LIN Interface (v2.1) Interrupts (Assist Mode)

Table 2-2 shows the interrupt control bits of the LIN interface (v2.1) and the interrupt factors in assist mode.

**Table 2-2 LIN Interface (v2.1) Interrupt Control Bits and Interrupt Factors (Assist Mode)**

Interrupt Type	Interrupt Request Flag Bit	Flag Register	Interrupt Factor	Interrupt Factor Enable Bit	Clearing of Interrupt Request Flag
Reception	RDRF	SSR	1-byte reception	SCR:RIE	Reading of reception data (RDR)
			Reception of as much data as the amount set in FBYTE		Reading of reception data (RDR) until the reception FIFO becomes empty
			Detection of reception idle for the 8-bit time or longer while the FRIIE bit is "1" and the reception FIFO contains valid data		
	ORE	SSR	Overrun error		Writing "1" to the reception error flag clear bit (SSR:REC)
	FRE	SSR	Framing error		
	LBSE	LAMESR	LIN bus error detection	LAMIER: LBSEIE	Writing "0" to LAMESR:LBSE
	LSFER	LAMESR	LIN Sync Data error detection	LAMIER: LSFERIE	Writing "0" to LAMESR:LSFER
	LPTE	LAMESR	LIN ID parity error detection	LAMIER: LPTEIE	Writing "0" to LAMESR:LPTE
	LCSE	LAMESR	LIN checksum error detection	LAMIER: LCSEIE	Writing "0" to LAMESR:LCSE
Transmission	TDRE	SSR	The transmission register is empty.	SCR:TIE	Writing to the transmission data (TDR), or writing "1" to the transmission FIFO operation enable bit (FCR0:FE1 or FCR0:FE2) (retransmission) when the bit value is "0" and the transmission FIFO contains valid data <sup>*1</sup>
Transmission	TBI	SSR	No transmission operation	SCR:TBIE	Writing to the transmission data (TDR), writing "1" to the LIN Break Field setting bit (LBR), or writing "1" to the transmission FIFO operation enable bit (retransmission) when the bit value is "0" and the transmission FIFO contains valid data <sup>*1</sup>

Interrupt Type	Interrupt Request Flag Bit	Flag Register	Interrupt Factor	Interrupt Factor Enable Bit	Clearing of Interrupt Request Flag
	FDRQ	FCR1	The amount of data in the transmission FIFO is equal to or less than the FTICR setting value or it is empty.	FCR1:FTIE	Writing "0" to the FIFO transmission data request bit (FCR1:FDRQ), or the transmission FIFO is full
Status (Assist Mode)	LBD	SSR	LIN Break Field detection	ESCR:LBIE	Writing "0" to the SSR:LBD bit
	SFD	SACSR	Sync Field detection	SACSR:SFDE	Writing "0" to the Sync Field detection flag bit (SACSR:SFD)
	TINT	SACSR	Coincidence of the value in the serial timer register (STMR) and that in the serial timer comparison register (STMCR)	SACSR:TINTE	Writing "0" to the timer interrupt flag bit (SACSR:TINT)
	LAHC	LAMSR	Completion of auto header	LAMIER:LAHCIE	Writing "0" to LAMSR:LAHC or reading from the ID register (LAMRID)
	LCSC	LAMSR	Completion of checksum calculation	LAMIER:LCSCIE	Writing "0" to LAMSR:LCSC

\*1: Set the TIE bit to "1" after the TDRE bit becomes "0".



### 2.2.1. Occurrence of Reception Interrupts and Flag Set Timing in Assist Mode

There are 2 reception interrupt types: Reception completion (SSR:RDRF) and reception error (SSR:ORE, FRE, LAMESR:LBSE, LSFE, LPTER, LCSE).

#### (1) Occurrence of Reception Completion Interrupts and Flag Set Timing

In assist mode (LAMCR:LAMEN=1), reception data is stored in the reception data register (RDR) every time the stop bit of each of the following fields is detected. A flag is set (SSR:RDRF=1) once reception is complete. At this time, a reception interrupt occurs if reception interrupt is enabled (SCR:RIE=1).

- Data Field for Response
- ID Field when the ID register is not used (LAMCR:LIDEN=0)

The timing at which the reception data full flag bit (SSR:RDRF) is set is the same as that in manual mode, as described in Section "2.1.1. Occurrence of Reception Interrupts and Flag Set Timing". See Figure 2-1.

#### Notes:

- If a reception error occurs, the data in the reception data register (RDR) will be invalid.
- Suppose that the setting is such that the LIN assist mode reception ID register is used for the reception of ID Fields (LAMCR:LIDEN=1). Then, when an ID Field is received, the received ID value is not stored into the reception data register (RDR) and the reception data full flag bit (SSR:RDRF) is not set.
- In slave operation, suppose that the setting is such that the reception data register (RDR) is used for the reception of ID Fields (LAMCR:LIDEN=0). Then, when an ID Field is received, the received ID value is stored to the reception data register (RDR) and the reception data full flag bit is set (SSR:RDRF=1). Check the ID value after the LIN auto header completion flag is set (LAMESR:LAHC=1).
- Neither the Sync Field nor the checksum are stored to the reception data register (RDR) and the reception data full flag bit (SSR:RDRF) is not set.

#### (2) Occurrence of Framing Error Interrupt and Flag Set Timing

In assist mode (LAMCR:LAMEN=1), a framing error is detected and the framing error flag is set (SSR:FRE=1) if the "L" level is detected on the stop bit when the Sync Field, ID Field, data, and checksum are received. A reception interrupt occurs if reception interrupt is enabled (SCR:RIE=1). The master node that transmits the Sync Field and ID Field or the master/slave node that transmits the checksum executes self-check on the transmitted data. If the "L" level is detected on the stop node, then a framing error is detected and the framing error flag is set (SSR:FRE=1). A reception interrupt occurs if reception interrupt is enabled (SCR:RIE=1).

Transmission/reception processing of a header and response in assist mode stops once a framing error is detected.

The operation enable bit for the reception FIFO is cleared (FCR0:FE1=0 or FCR0:FE2=0) while the framing error flag is set (SSR:FRE=1).

The timing at which the framing error flag bit (SSR:FRE) is set is the same as that in manual mode, as described in Section "2.1.1. Occurrence of Reception Interrupts and Flag Set Timing". See Figure 2-2.

### (3) Occurrence of Overrun Error Interrupt and Flag Set Timing

An overrun error is detected if the following data reception has been detected before reading of the reception data (SSR:RDRF=1). The overrun error flag is set (SSR:ORE=1) when reception is complete (SSR:RDRF=1). A reception interrupt occurs if reception interrupt is enabled (SCR:RIE=1).

The reception processing of a header section and response section in assist mode stops once an overrun error is detected.

The operation enable bit for the reception FIFO is cleared (FCR0:FE1=0 or FCR0:FE2=0) while the overrun error flag is set (SSR:ORE=1).

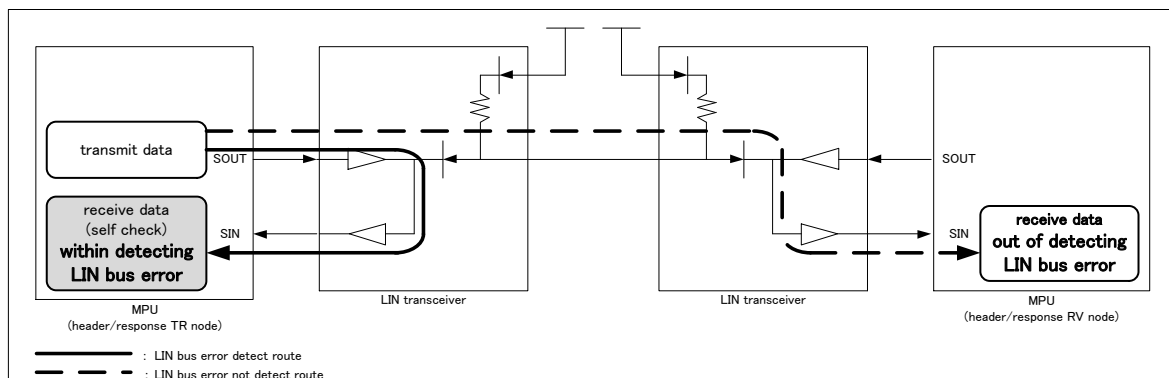
The timing at which the overrun error flag bit (SSR:ORE) is set is the same as that in manual mode, as described in Section "2.1.1. Occurrence of Reception Interrupts and Flag Set Timing". See Figure 2-3.

### (4) Occurrence of LIN Bus Error Detection Interrupt and Flag Set Timing

In assist mode (LAMCR:LAMEN=1), LIN bus error detection is executed by the self-check on the side that transmits the header/response. The side that receives the header/response cannot execute LIN bus error detection.

Figure 2-12 shows the LIN bus error detection target.

Figure 2-12 LIN BUS Error Detection Target



The target range of the LIN bus error detection is the start bit and byte data of the LIN Break and Sync Field/ID Field/Data Field/Check Sum Field. The stop bit is not included in the LIN bus error detection target. A framing error (SSR:FRE=1) is detected when the "L" level is detected on the stop bit.

The transmission processing of a header section and response section in assist mode stops once a LIN bus error is detected.

The LIN auto header completion flag (LAMSR:LAHC=1) is set even if a LIN bus error occurs when the ID Field transmission is complete.

#### a) Occurrence of LIN Bus Error Detection Interrupt and Flag Set Timing on the Master Side

The master side (SCR:MS=0) performs LIN bus error detection when transmitting a header/response.

If an abnormality is detected as a result of comparing the transmission LIN Break length and the reception LIN Break length or the transmission data and reception data, then a LIN bus error is detected and a flag is set (LAMESR:LBSE=1). A reception interrupt occurs if the interrupt is enabled (LAMIER:LBSEIE=1).



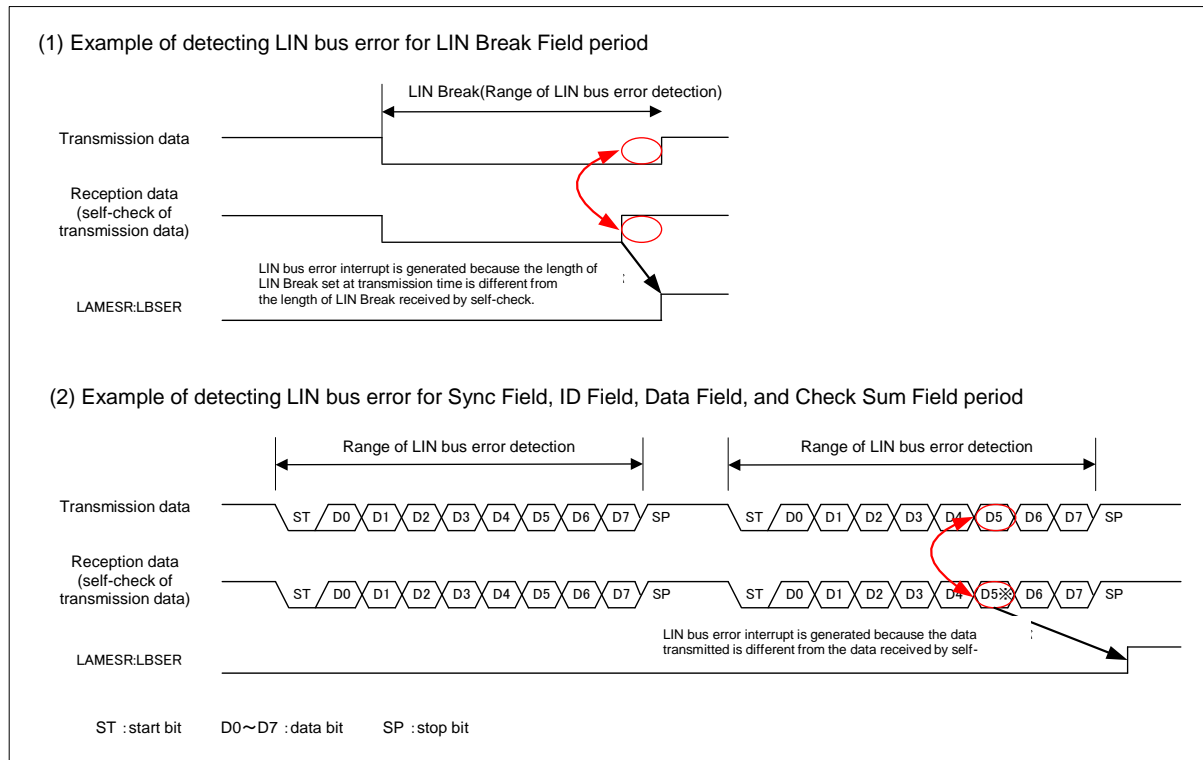


### b) Occurrence of LIN Bus Error Detection Interrupt and Flag Set Timing on the Slave Side

The slave side (SCR:MS=1) performs LIN bus error detection when transmitting a response.

If an abnormality is detected as a result of comparing transmission data and reception data, then a LIN bus error is detected and a flag is set (LAMESR:LBSE=1). A reception interrupt occurs if the interrupt is enabled (LAMIER:LBSEIE=1).

**Figure 2-13 Set Timing of LIN Bus Error Detection Flag (LAMESR:LBSE)**

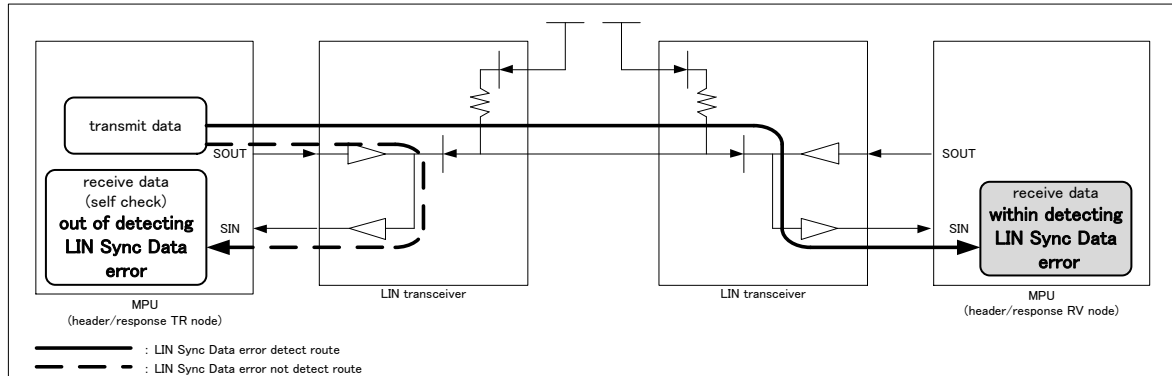


### (5) Occurrence of LIN Sync Data Error Detection Interrupt and Flag Set Timing

The LIN Sync Data error detection is executed if the auto baud rate adjustment (SACSR:AUTE=0) is disabled when slave mode (SCR:MS=1) is set in assist mode (LAMCR:LAMEN=1).

Figure 2-14 shows the LIN Sync Data error detection target.

Figure 2-14 LIN Sync Data Error Detection Target



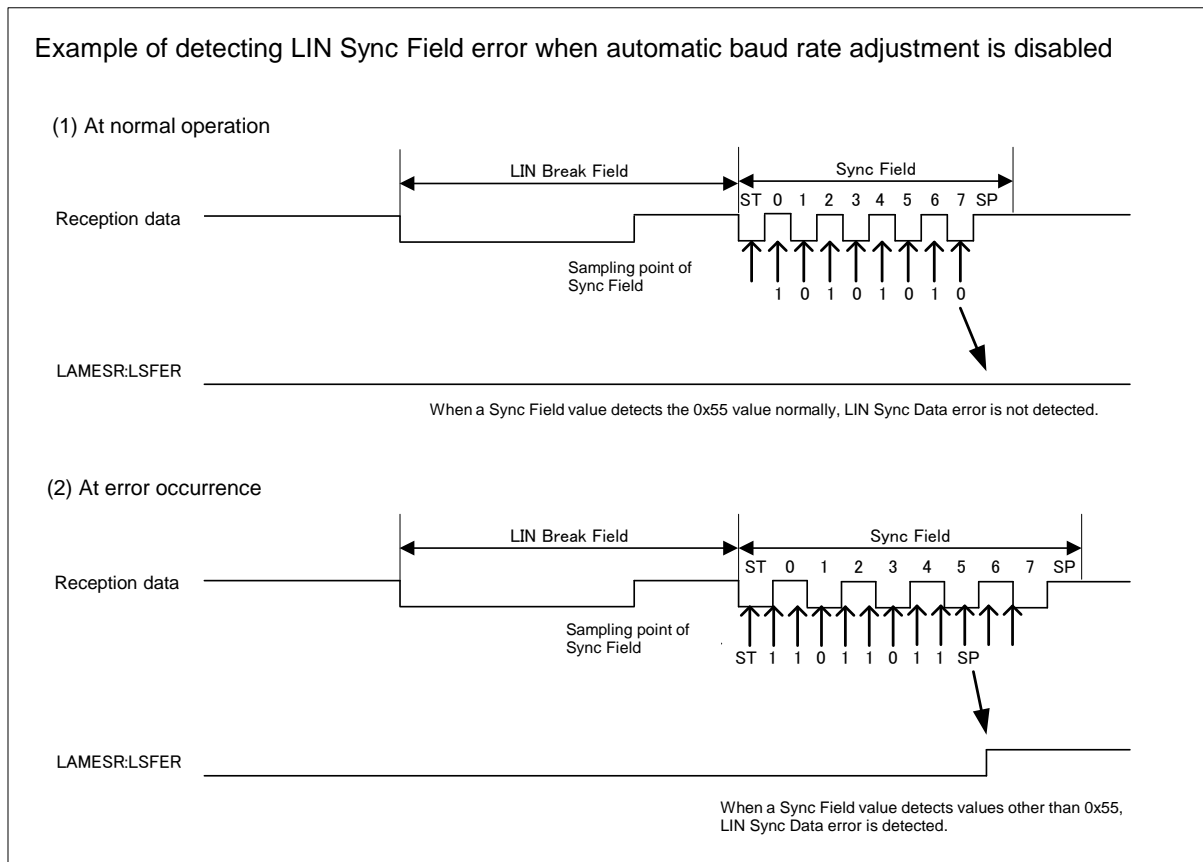
The target range of LIN Sync Data error detection is the start bit and byte data of the Sync Field. The stop bit is not included in the LIN Sync Data error detection target. A framing error (SSR:FRE=1) is detected when the "L" level is detected on the stop bit.

#### a) Occurrence of LIN Sync Data Error Detection Interrupt and Flag Set Timing When the Auto Baud Rate Adjustment is Disabled

In slave mode (SCR:MS=1) when auto baud rate adjustment is disabled (SACSR:AUTE=0), the data value of the Sync Field is checked. If any value other than 0x55 is detected, then a LIN Sync Data error is detected and a flag is set (LAMESR:LSFER=1). At this time, a reception interrupt occurs if the interrupt is enabled (LAMIER:LSFERIE=1).



**Figure 2-15 Set Timing of LIN Sync Data Error Detection Flag (LAMESR:LSFER) (When Auto Baud Rate Adjustment is Disabled)**

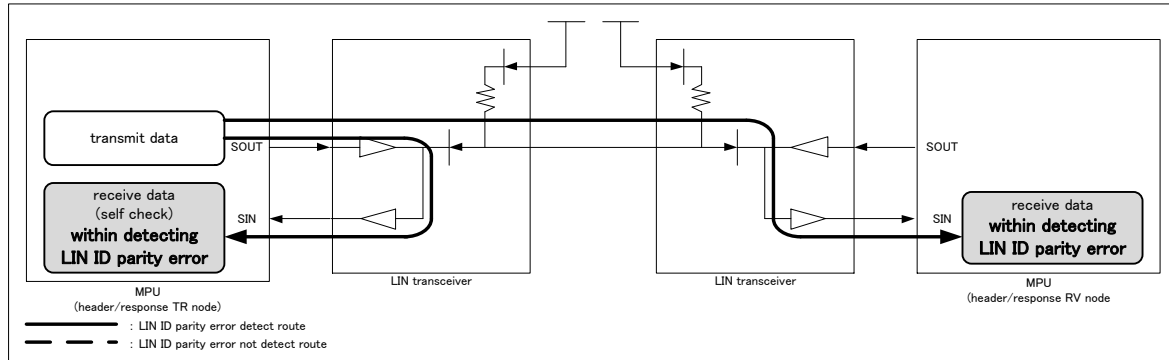


### (6) Occurrence of LIN ID Parity Error Detection Flag Interrupt and Flag Set Timing

In assist mode (LAMCR:LAMEN=1), the LIN ID parity error detection is executed on the slave that receives the ID Field and the self-check is performed on the master that transmits the ID Field.

Figure 2-16 shows the LIN ID parity error detection target.

Figure 2-16 LIN ID Parity Error Detection Target



The target range of LIN ID parity error detection is the ID data and the byte data of the parity. The start and stop bits are not included in the LIN ID parity error detection target. A framing error occurs (SSR:FRE=1) if the "L" level is detected on the stop bit.

In LIN assist mode (LAMCR:LAMEN=1), the auto header completion flag is set (LAMSR:LAHC=1) if a LIN ID parity error occurs during auto header transmission/reception.

Transmission/reception processing for a response in assist mode stops once a LIN ID parity error is detected.

#### a) Occurrence of LIN ID Parity Error Detection Interrupt and Flag Set Timing on the Master Side

The master (SCR:MS=0) set in assist mode (LAMCR:LAMEN=1) performs LIN ID parity error detection when transmitting the ID Field. The master executes a parity operation for the 6-bit Frame ID set in the transmission data register (TDR) or the LIN assist mode transmission ID register (LAMTID). Then, the master automatically creates and transmits an ID Field.

If an ID Field is received for self-check and if the parity operation result of the Frame ID value differs from the received parity value, then a LIN ID parity error is detected and a flag is set (LAMESR:LPTER=1).

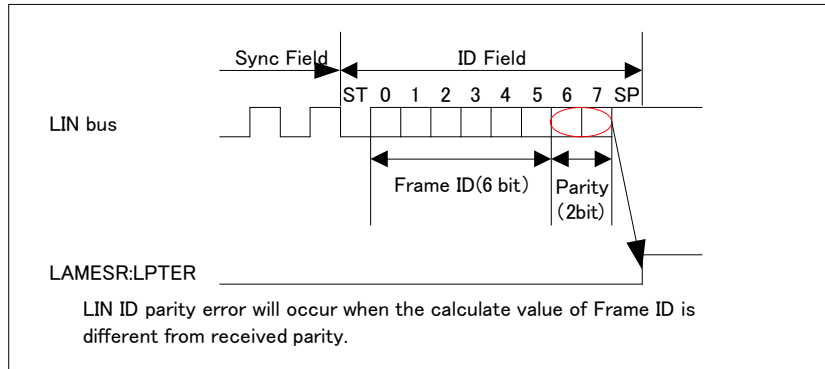
At this time, a reception interrupt occurs if the interrupt is enabled (LAMIER:LPTERIE=1).

#### b) Occurrence of LIN ID Parity Error Detection Interrupt and Flag Set Timing on the Slave Side

The slave (SCR:MS=1) set in assist mode (LAMCR:LAMEN = 1) performs LIN ID parity error detection when it receives the ID Field. If the ID Field is received and if the parity operation result of the Frame ID value differs from the received parity value, then a LIN ID parity error is detected and a flag is set (LAMESR:LPTER=1).

At this time, a reception interrupt occurs if the interrupt is enabled (LAMIER:LPTERIE=1).

Figure 2-17 Set Timing of LIN ID Parity Error Detection Flag (LAMESR:LPTEr)

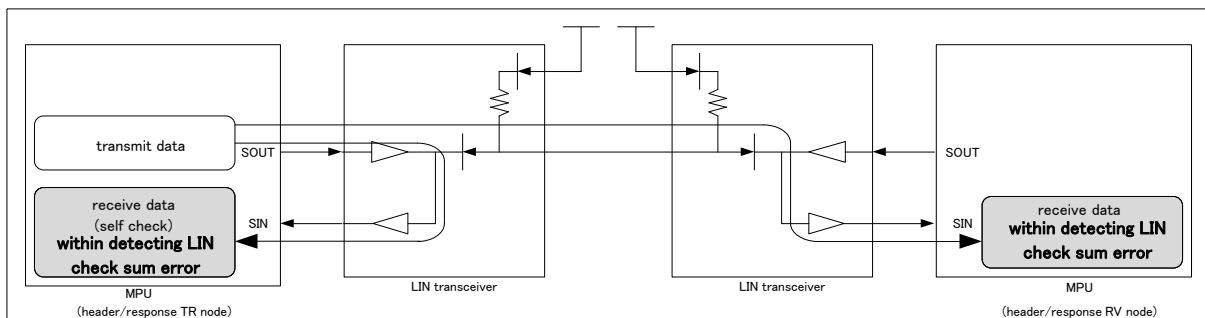


### (7) Occurrence of LIN Checksum Error Detection Flag Interrupt and Flag Set Timing

In assist mode (LAMCR:LAMEN=1), LIN checksum error detection is executed by the self-check on the side that transmits the checksum and is also executed by the side that receives the checksum.

Figure 2-18 shows the LIN checksum error detection target.

Figure 2-18 LIN Checksum Error Detection Target



As the calculation method for the checksum that is automatically transmitted, select standard (target: data) or extended (target: ID Field + data) checksum with the LAMCR:LCSTYP bit.

#### a) Occurrence of LIN Checksum Error Detection Interrupt and Flag Set Timing when Setting Standard Checksum Calculation

When the standard checksum calculation is set (LAMCR:LCSTYP=0), the side that transmits a response (data and checksum) performs the checksum calculation based on the transmission data for the specified LIN data length (LAMCR:LDL3 to 0). Then, after transmitting the last data, the transmission side automatically transmits the checksum.

The side that receives a response (data and checksum) executes the checksum calculation based on the reception data for the specified LIN data length (LAMCR:LDL3 to 0). If the received checksum differs from the calculation result value, then a LIN checksum error is detected and a flag is set (LAMESR:LCSEr=1).

At this time, a reception interrupt occurs if the interrupt is enabled (LAMIER:LCSErIE=1).

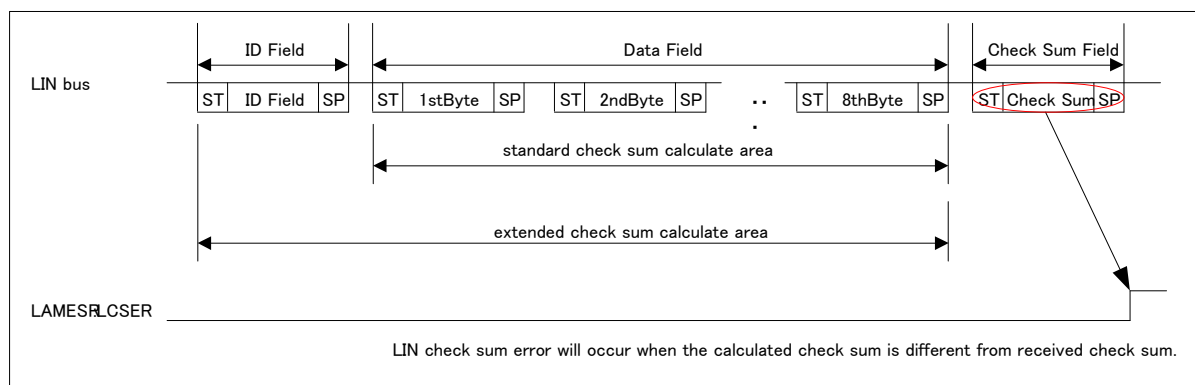
### b) Occurrence of LIN Checksum Error Detection Interrupt and Flag Set Timing when Setting Extended Checksum Calculation

When extended checksum calculation is set (LAMCR:LCSTYP=1), the side that transmits a response (data and checksum) performs the checksum calculation based on the ID Field value and the transmission data for the specified LIN data length (LAMCR:LDL3 to 0). Then, after transmitting the last data, the transmission side automatically transmits the checksum.

The side that receives a response (data and checksum) executes the checksum calculation based on the ID Field value and the reception data for the specified LIN data length (LAMCR:LDL3 to 0). If the received checksum differs from the calculation result value, then a LIN checksum error is detected and a flag is set (LAMESR:LCSER=1).

At this time, a reception interrupt occurs if the interrupt is enabled (LAMIER:LCSERIE=1).

Figure 2-19 Set Timing of LIN Checksum Error Detection Flag (LAMESR:LCSER)



**Note:**

- Regardless of the standard/extended checksum calculation setting, the processing performed in assist mode stops and checksum calculation is not executed if an error (LIN bus error, LIN ID parity error, LIN Sync Data error, or framing error) is detected in the data of the header section and the response section.



### 2.2.2. Occurrence of Interrupts and Flag Set Timing When Using Reception FIFO

The timing is the same as that in manual mode, as described in "2.1.2 Occurrence of Interrupts and Flag Set Timing when Using Reception FIFO"

### 2.2.3. Occurrence of Transmission Interrupts and Flag Set Timing

An interrupt occurs by the SSR:TDRE flag and the SSR:TBI flag at transmission.

#### Occurrence of Transmission Interrupts and Flag Set Timing

##### a) Set Timing of Transmission Data Empty Flag (TDRE)

The timing is the same as that in manual mode, as described in "2.1.3 Occurrence of Transmission Interrupts and Flag Set Timing."

##### b) Set Timing of Transmission Bus Idle Flag (TBI)

The transmission bus idle flag bit (SSR:TBI) is set to "1" when all the following conditions are satisfied. At this time, a transmission interrupt occurs if the transmission bus idle interrupt is enabled (SCR:TBIE=1).

- The transmission data register is empty (SSR:TDRE=1) and transmission is not performed.
- A header (LIN Break Field, Sync Field, or ID Field) has not been transmitted when the master operation (SCR:MS=0) is used in assist mode (LAMCR:LAMEN=1).
- A response (data and checksum) has not been transmitted in assist mode (LAMCR:LAMEN=1).

The transmission bus idle flag bit (SSR:TBI) and a transmission interrupt request are cleared in any of the following cases.

- When writing transmission data to the transmission data register (TDR)  
(The transmission data register is not empty (SSR:TDRE=0))
- When a header (LIN Break Field, Sync Field, or ID Field) is being transmitted while the master is being used (SCR:MS=0) in assist mode (LAMCR:LAMEN=1).
- A response (data and checksum) is being transmitted in assist mode (LAMCR:LAMEN=1)



Figure 2-20 Set Timing of Transmission Bus Idle Flag (TBI) (When Master Response is Transmitted and ID Register is Used)

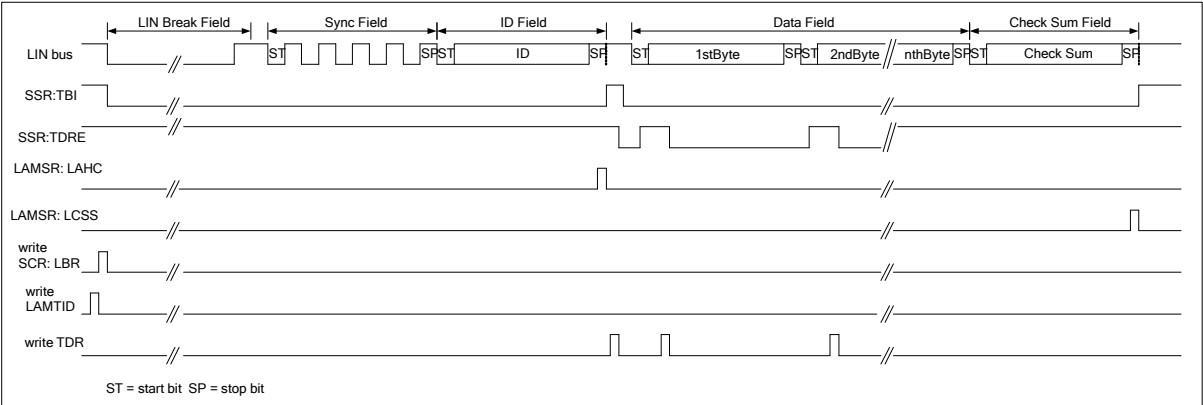


Figure 2-21 Set Timing of Transmission Bus Idle Flag (TBI) (When Master Response is Transmitted and ID Register is Not Used)

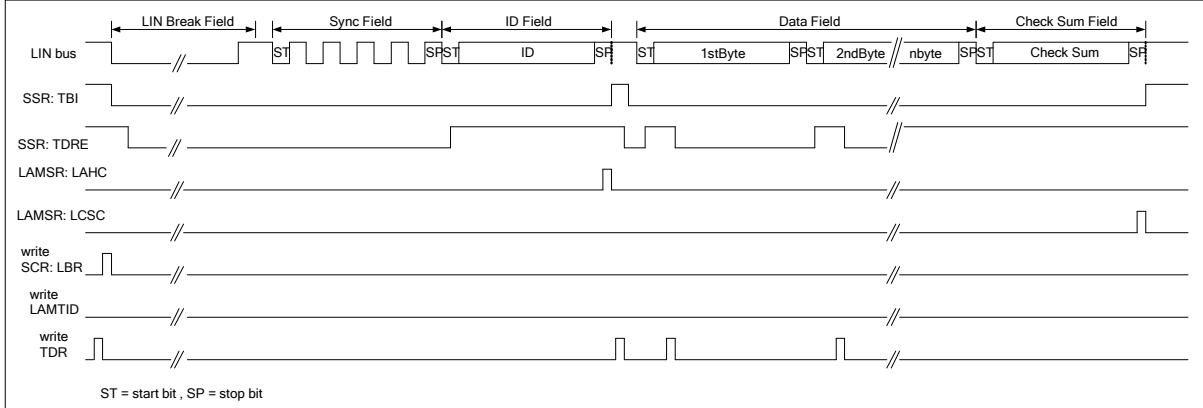
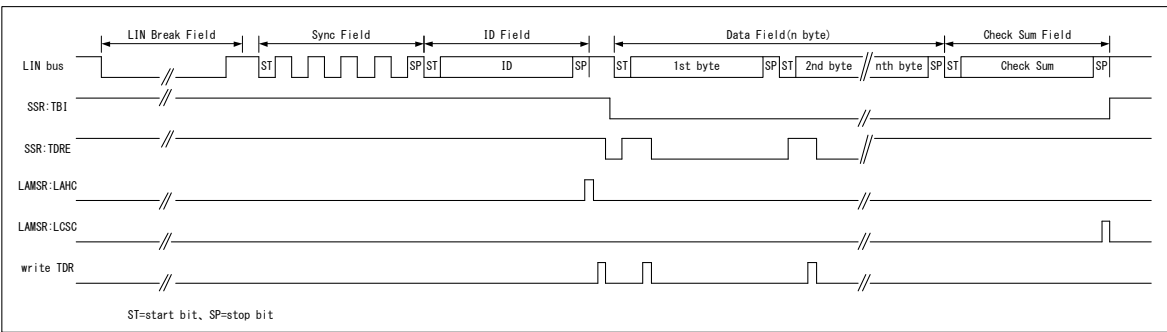


Figure 2-22 Set Timing of Transmission Bus Idle Flag (TBI) (When Slave Response is Transmitted)







### 2.2.4. Occurrence of Interrupts and Flag Set Timing When Using Transmission FIFO

The timing is the same as that in manual mode, as described in "2.1.4 Occurrence of Interrupts and Flag Set Timing When Using Transmission FIFO".

### 2.2.5. Occurrence of Timer Interrupts and Flag Set Timing

The timing is the same as that in manual mode, as described in "2.1.5 Occurrence of Timer Interrupts and Flag Set Timing".

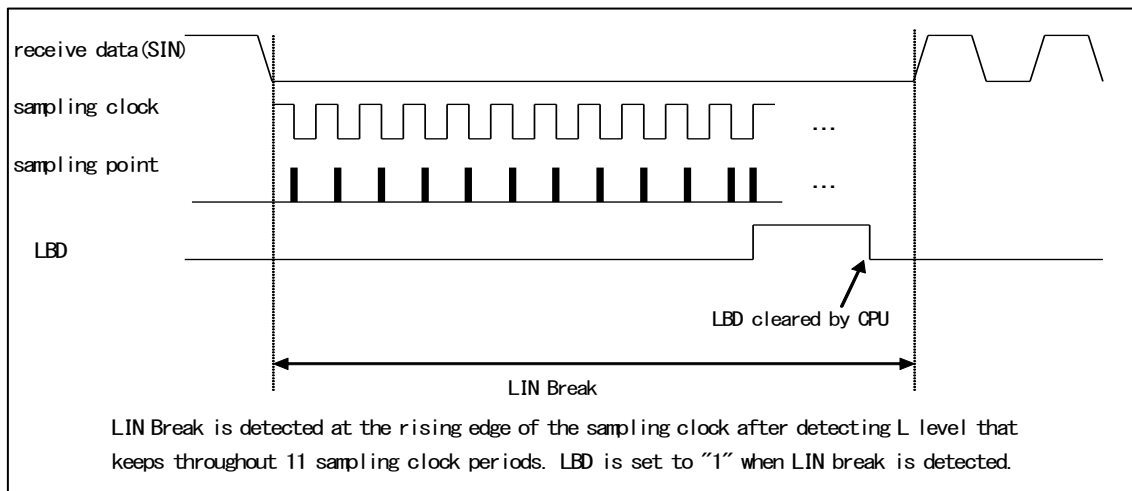
### 2.2.6. Occurrence of Status Interrupt and Flag Set Timing in Assist Mode

In assist mode, a status interrupt occurs when the LIN Break Field (SSR:LBD) is detected, the Sync Field is detected (SACSR:SFD), the auto header is complete (LAMSR:LAHC), or checksum calculation is complete (LAMSR:LCSC).

#### (1) Set Timing of LIN Break Field Detection Flag (LBD)

The LBD bit is set to "1" when the serial input (SIN) has "0" input for an interval of 11 bits or more. At this time, a status interrupt occurs if the LIN Break Field interrupt is enabled (ESCR:LBIE=1).

Figure 2-23 Set Timing of LBD (LIN Break Field Detection) Flag



**Note:**

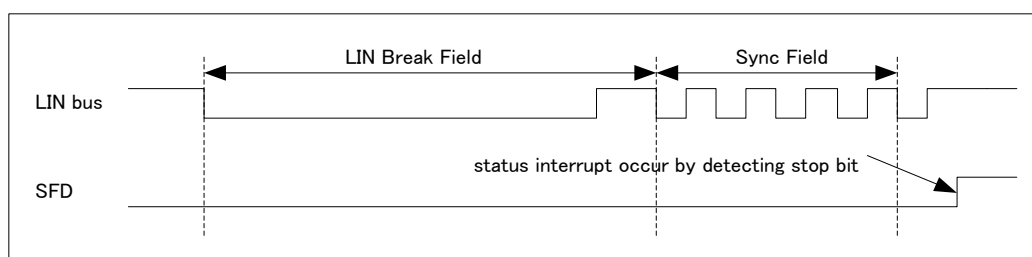
- During the reception of a LIN Break Field, a framing error is detected before the detection of a LIN Break Field if reception is enabled (SCR:RXE=1).

### (2) Sync Field Detection Interrupt and Flag Set Timing

The Sync Field detection interrupt and flag set timing varies as follows based on the setting value of the auto baud rate adjustment bit (SACSR:AUTE).

- When auto baud rate adjustment is enabled (SACSR:AUTE=1)  
The timing is the same as that in manual mode, as described in "2.1.7 Occurrence of Sync Field Detection Interrupt and Flag Set Timing".
- When auto baud rate adjustment is disabled (SACSR:AUTE=0)  
The Sync Field detection flag is set (SACSR:SFD=1) when the stop bit of the Sync Field is detected. A status interrupt occurs if the interrupt is enabled (SACSR:SFDE=1).

**Figure 2-24 Set Timing of Sync Field Detection Flag (SACSR:SFD) When Auto Baud Rate Adjustment is Disabled (SACSR:AUTE=0)**

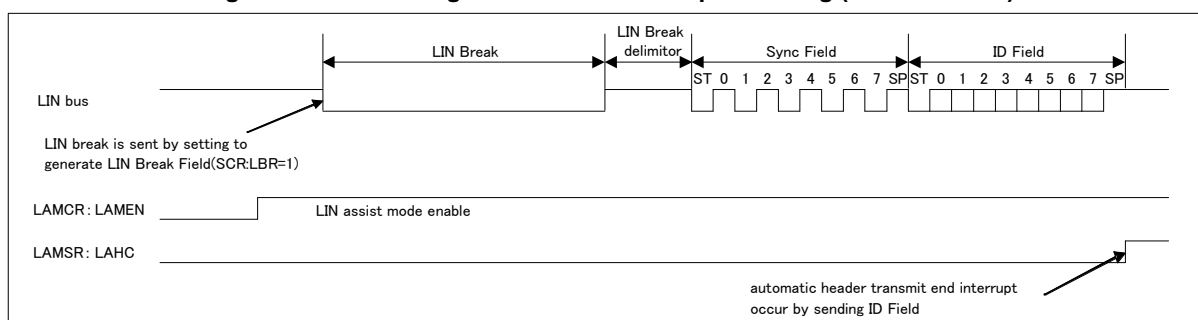


### (3) Auto Header Completion Interrupt and Flag Set Timing at Transmission

When the master is set in LIN assist mode (LAMCR:LAMEN=1), a flag is set (LAMSR:LAHC=1) once transmission of the headers from the LIN Break to the ID Field is complete. A status interrupt occurs if the interrupt is enabled (LAMIER:LAHCIE=1).

In LIN assist mode, the auto header completion flag is set (LAMSR:LAHC=1) even when a LIN bus error/LIN ID parity error/framing error occurs in the ID Field. However, the transmission/reception processing of the response section stops in LIN assist mode.

**Figure 2-25 Set Timing of Auto Header Completion Flag (LAMSR:LAHC)**



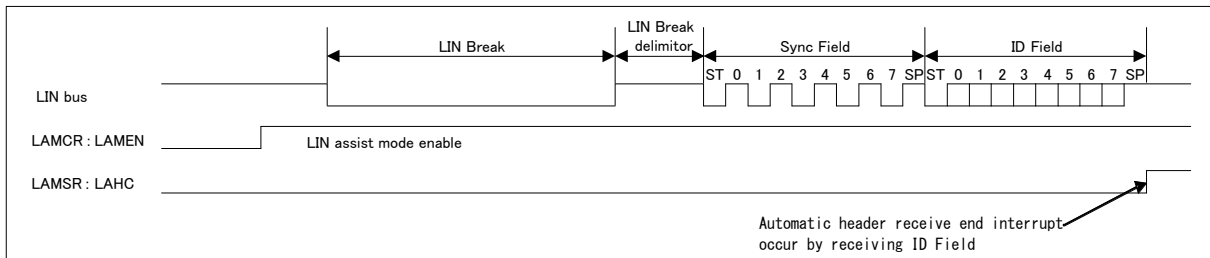
### (4) Auto Header Reception Completion Interrupt and Flag Set Timing at Reception

When the slave is set in LIN assist mode (LAMCR:LAMEN=1), a flag is set (LAMSR:LAHC=1) once reception of the headers from the LIN Break to the ID Field is complete. A status interrupt occurs if the interrupt is enabled (LAMIER:LAHCIE=1).

In LIN assist mode, the auto header completion flag is set (LAMSR:LAHC=1) even if a LIN bus error/LIN ID parity error/framing error occurs in the ID Field. However, transmission/reception processing for the response section stops in LIN assist mode.



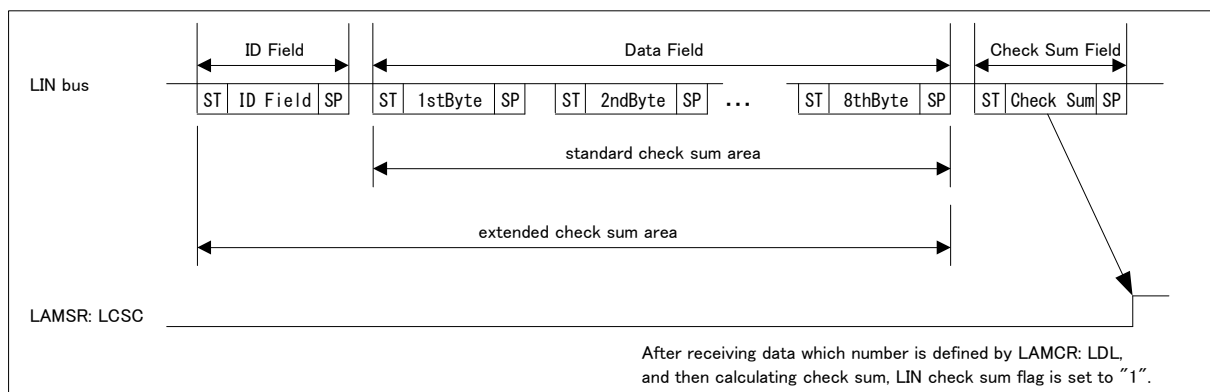
Figure 2-26 Set Timing of Auto Header Completion Flag (LAMSR:LAHC)

**(5) Occurrence of LIN Checksum Detection Completion Flag Interrupt and Flag Set Timing**

When assist mode is set (LAMCR:LAMEN=1), checksum detection is executed by the self-check performed on the side that transmits the checksum, and is also executed by the side that receives the checksum. When the data for the specified data length (LAMCR:LDL3 to 0) and checksum are received, the checksum calculation is complete so a flag is set (LAMSR:LCSC=1). A status interrupt occurs if the interrupt is enabled (LAMIER:LCSCIE=1).

When checksum reception is complete, the received checksum value is not stored in the RDR register and the SSR:RDRF is not set to "1". When a FIFO is used, the value is not stored in the reception FIFO.

Figure 2-27 Set Timing of LIN Checksum Detection Completion Flag (LAMSR:LCSC)

**Notes:**

- The checksum calculation stops as soon as a framing error is detected in the last data of the specified data length (LAMCR:LDL3 to 0).
- If a framing error is detected in the checksum calculation result is displayed. However, the calculation result is not guaranteed.



3. Serial Timer Operation

The serial timer provides timer functions.

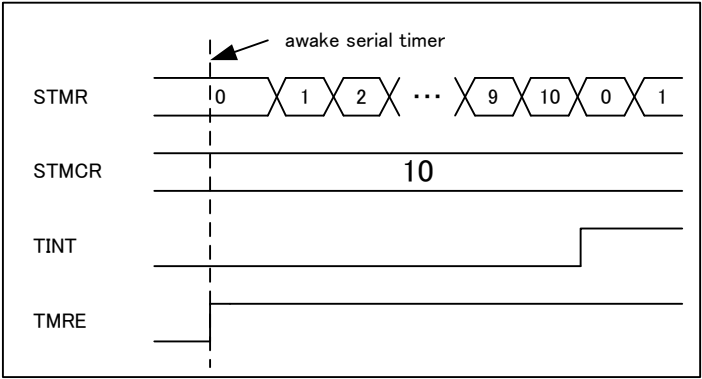
Serial Timer Operation

a) Activating Serial Timer

The serial timer can be activated by setting the serial timer enable bit (SACSR:TMRE) to "1" by the Sync Field.

- Activation by the serial timer enable bit (SACSR:TMRE)  
Setting the serial timer enable bit (SACSR:TMRE) to "1" activates the serial timer and starts counting of the serial timer register (STMR) from 0.

Figure 3-1 Activation by Serial Timer Enable Bit (STMCR=10, SACSR:TRGE=0)

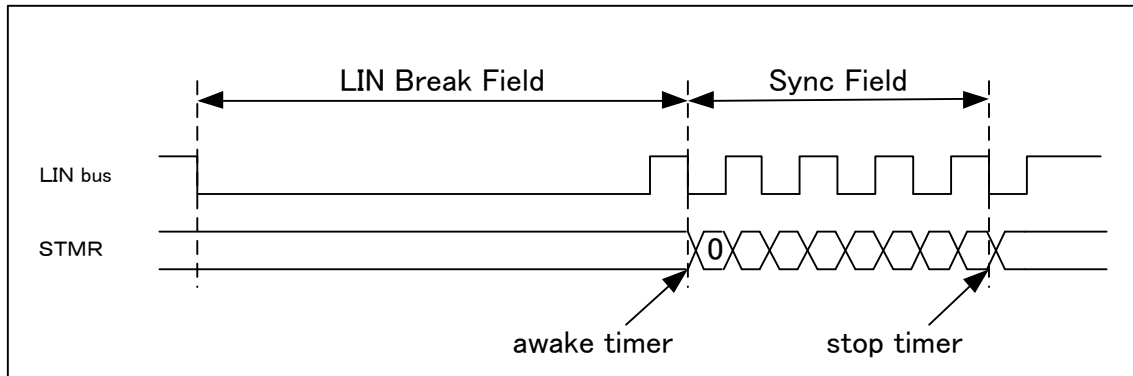




- Activation by Sync Field reception

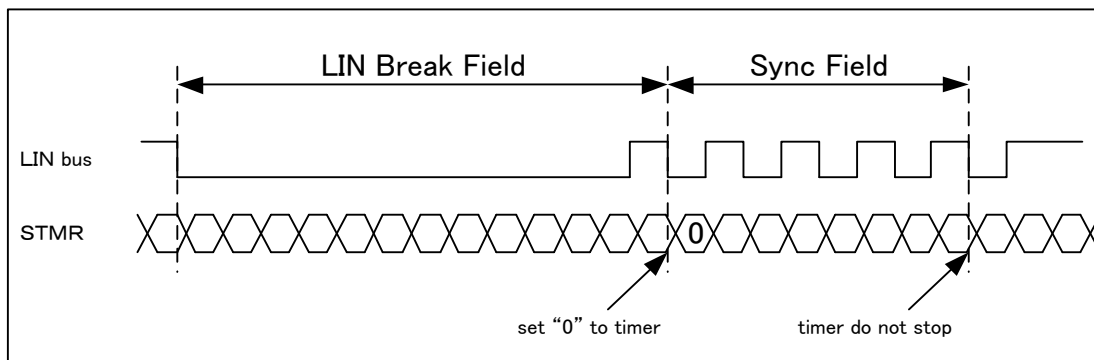
The serial timer is activated and counting of the serial timer register (STMR) starts from 0 when all of the following are satisfied: 1. The serial timer is stopped. 2. The auto baud rate adjustment bit (SACSR:AUTE) is set to "1". 3. The LIN interface (v2.1) detects the first falling edge of the Sync Field.

**Figure 3-2 Reception of Sync Field while Serial Timer is Stopped (SACSR:AUTE=1, TMRE=0)**



Counting of the serial timer register (STMR) starts from 0 when all of the following are satisfied: 1. The serial timer is operating. 2. The auto baud rate adjustment bit (SACSR:AUTE) is set to "1". 3. The LIN interface (v2.1) detects the first falling edge of the Sync Field.

**Figure 3-3 Reception of Sync Field while Serial Timer is Operating (SACSR:AUTE=1, TMRE=1)**



**Note:**

- Detection of an external trigger edge specified by the trigger selection bit (SACSR:TRG1, 0) does not start the serial timer when both of the following are satisfied: 1. The external trigger enable bit (SACSR:TRGE) is set to "1". 2. The serial timer enable bit (SACSR:TMRE) is set to "0".

#### **b) Stopping the Serial Timer**

The serial timer stops under the following conditions.

- The serial timer stops when both of the following are satisfied: 1. The auto baud rate adjustment bit (AUTE) is set to "0". 2. The serial timer enable bit (SACSR:TMRE) is reset to "0". At this time, the value of the serial timer register (STMR) is retained.
- The serial timer stops when all of the following are satisfied: 1. The auto baud rate adjustment bit (AUTE) is set to "1". 2. The serial timer enable bit (SACSR:TMRE) is set to "1". 3. The serial timer enable bit (SACSR:TMRE) is reset to "0" in any status other than that indicating that the Sync Field is being received. At this time, the value of the serial timer register (STMR) is retained.
- The serial timer stops and the value of the serial timer register (STMR) is retained if the LIN interface (v2.1) detects the 5th falling edge of the Sync Field, when both of the following are satisfied: 1. The auto baud rate adjustment bit (AUTE) is set to "1". 2. The serial timer enable bit (SACSR:TMRE) is set to "0".

**Note:**

- *The serial timer does not stop and continues operating even if the LIN interface (v2.1) detects the 5th falling edge of the Sync Field, when both of the following are satisfied: 1. The auto baud rate adjustment bit (AUTE) is set to "1". 2. The serial timer enable bit (SACSR:TMRE) is set to "1".*

**c) Timer Operation**

If the serial timer register (STMR) and the serial timer comparison register (STMCR) match, the timer interrupt flag (SACSR:TINT) is set to "1", and the serial timer register (STMR) is reset to "0".

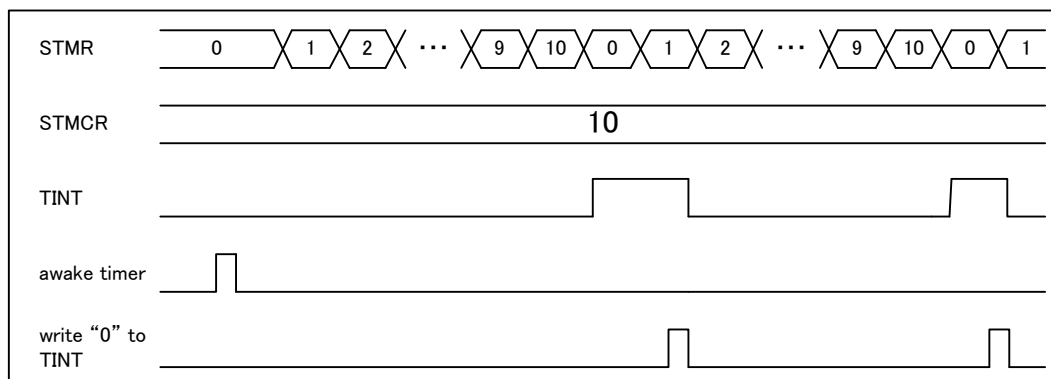
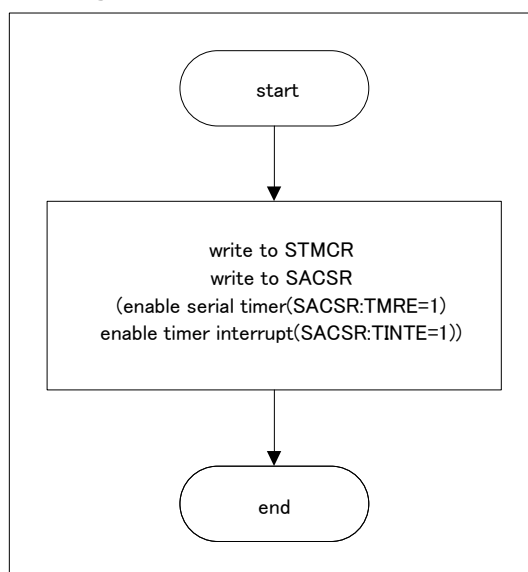
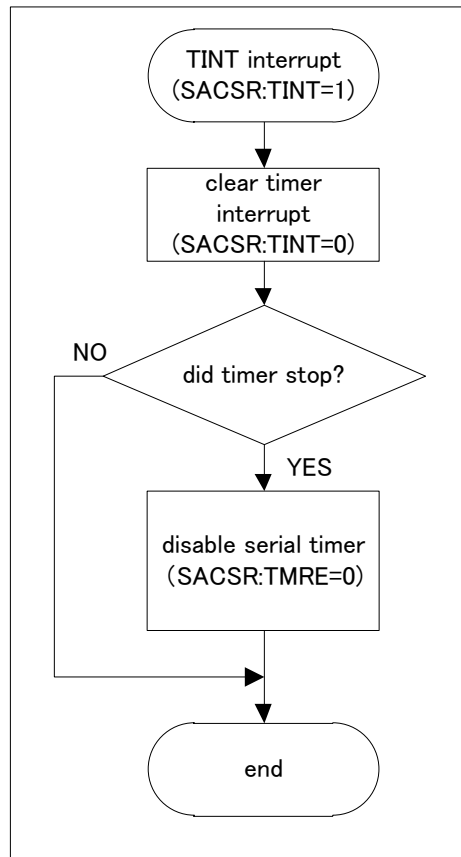
**Figure 3-4 Timer Operation (STMCR=10)****Figure 3-5 Serial Timer Initialization**

Figure 3-6 Serial Timer Interrupt



**Notes:**

- The timer interrupt flag (SACSR:TINT) is fixed to "1" when the timer comparison register (STMCR) is set to 0x0000, the timer is operating, and the division value (SACSR:TDIV) of the timer operation clock is set to "0b0000".
- The serial timer register (STMR) is reset to "0" after receiving the Sync Field when the auto baud rate adjustment bit (SACSR:AUTE) is set to "1".





## 4. Test Mode

This chapter describes the operation in test mode.

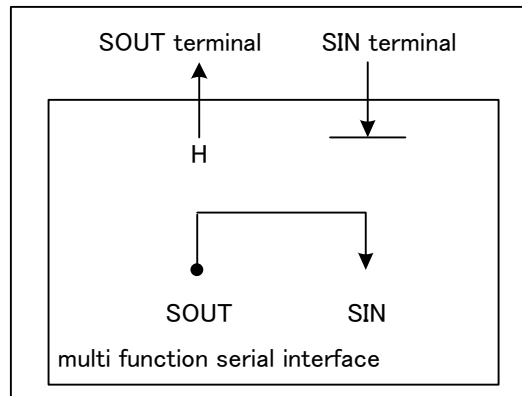
### 4.1. Manual Mode

#### Serial Test Mode

When serial test mode is enabled (SACSR:STST=1), SOUT and SIN are connected together within the multifunction serial interface, so that the data transmitted from SOUT can be received through SIN directly.

When serial test mode is enabled (SACSR:STST=1), the SOUT pin is fixed to "H", and the data input to the SIN pin is ignored.

Figure 4-1 Serial Test Mode



#### Notes:

- The value of the serial test mode enable bit (SACSR:STST) can be changed only when transmission and reception are prohibited (SCR:TXE=RXE=0).
- In manual mode (LAMCR:LAMEN=0), do not set the pseudo error test mode.

## 4.2. Assist Mode

### (1) Serial Test Mode

This mode is the same as Serial test mode in manual mode.

However, in LIN assist mode (LAMCR:LAMEN=1), the serial test can be executed only on the master node (SCR:MS=0). The serial test result is checked by the transmission/reception flag and the status flag. For the transmission/reception flag and status flag, see Table 2-2.

#### Notes:

- In LIN assist mode (LAMCR:LAMEN=1), the serial test cannot be executed on the slave node (SCR:MS=1).
- When the serial test is executed on the master node (SCR:MS=0) in LIN assist mode (LAMCR:LAMEN=1), transmission data (the Sync Field, ID Field, and response data, as well as the checksum) cannot be read by the reception data register (RDR).

### (2) Pseudo Error Test Mode

In assist mode (LAMCR:LAMEN=1), a LIN bus error, LIN Sync Data error, LIN ID parity error, LIN checksum error, and framing error can be generated as a pseudo error. More than one of these errors can be generated at the same time.

Any of the following self-diagnostics can be executed by implementing serial test mode together.

- Pseudo LIN bus error test mode
- Pseudo LIN ID parity error test mode
- Pseudo LIN checksum error test mode
- Pseudo framing error test mode

#### a) Activating Pseudo Error Test Mode

To activate pseudo error test mode, it is necessary to enable the pseudo error setting by writing values to the key code control bit (LAMERT:KEY1, KEY0) according to the following procedure.

- KEY1, KEY0 = 0b00 + write a pseudo error setting value
- KEY1, KEY0 = 0b01 + write the pseudo error setting value (same as the previous value)
- KEY1, KEY0 = 0b10 + write the pseudo error setting value (same as the previous value)
- KEY1, KEY0 = 0b11 + write the pseudo error setting value (same as the previous value)
- The pseudo error setting value is enabled when the value is written for the 4th time.

The value written to this register is invalid if you do not follow this setting procedure (if a value is written or read to/from another register during the writing process, if the value written is incorrect, or if a value is read from this register during the writing process).

To clear a pseudo error setting, follow the same procedure for making a pseudo error setting.

#### Notes:

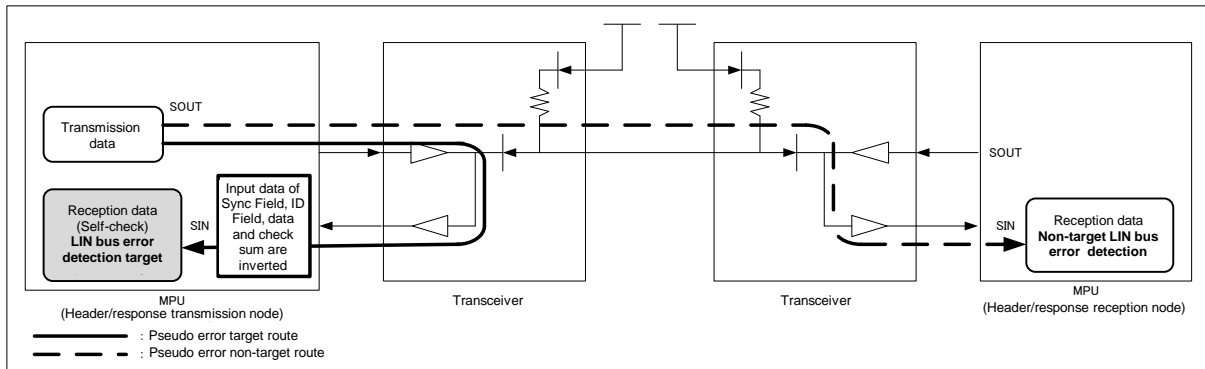
- LIN communication stops when any of the following pseudo errors occurs in assist mode.
  - LIN bus error
  - LIN ID parity error
  - Framing error
- In manual mode (LAMCR:LAMEN=0), do not set pseudo error test mode.



### b) Overview of Pseudo LIN Bus Error Test Mode

During a pseudo LIN bus error test, a node that transmits data receives the transmitted data by inverting it.

Figure 4-2 Overview of Pseudo LIN Bus Error Test Mode



To activate pseudo LIN bus error test mode, set the LIN bus error pseudo error setting bit (LAMERT:LBSERT=1) according to the method used to activate pseudo error test mode.

Activating pseudo LIN bus error test mode causes the following operation to be performed.

- Master
  - The Sync Field, ID Field, data, and checksum are transmitted.
  - Once the LIN bus error pseudo error setting is set (LAMERT:LBSERT=1), the reception data is inverted at the stop bit timing. Then, a LIN bus error occurs during the self-check and the flag bit (LAMESR:LBSER) is set to "1".
- Slave
  - Data and the checksum are transmitted.
  - Once the LIN bus error pseudo error setting is set (LAMERT:LBSERT=1), the reception data is inverted at the stop bit timing. Then, a LIN bus error occurs during the self-check and the flag bit (LAMESR:LBSER) is set to "1".

#### Notes:

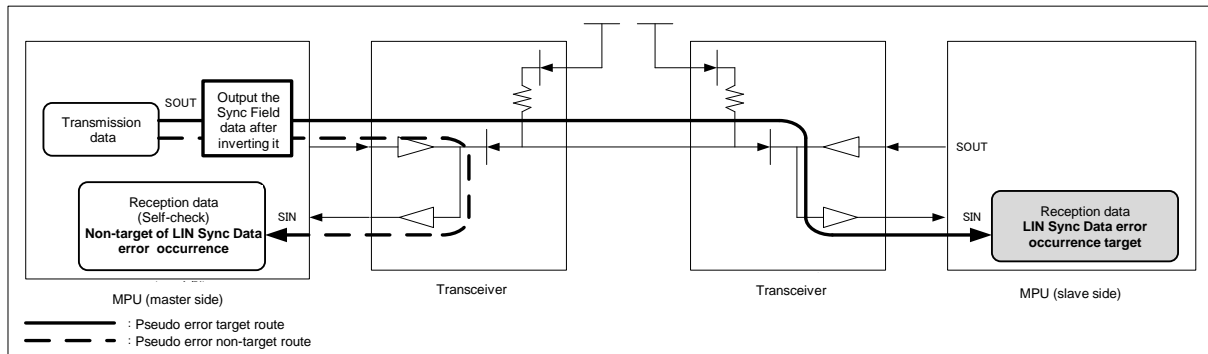
- When a LIN bus error is detected (LAMESR:LBSER=1), the transmission/reception processing of the header and response sections in assist mode stops.
- Enable this test mode only when transmitting a header or response.

### c) Overview of Pseudo LIN Sync Data Error Test Mode

In a pseudo LIN Sync Data error test, the Sync Field value (0x55) is transmitted after being inverted.

The master that transmits the Sync Field cannot detect a pseudo LIN Sync Data error.

Figure 4-3 Overview of Pseudo LIN Sync Data Error Test Mode



To activate pseudo LIN Sync Data error test mode, it is necessary to enable the LIN Sync Data error pseudo error setting bit (LAMERT:LSFERT=1) according to the method used to activate pseudo error test mode.

When the pseudo LIN Sync Data error pseudo error setting (LAMERT:LSFERT=1) is set in the master before the start bit of the Sync Field, the master outputs the value (0x55) after inverting it when transmitting the Sync Field.

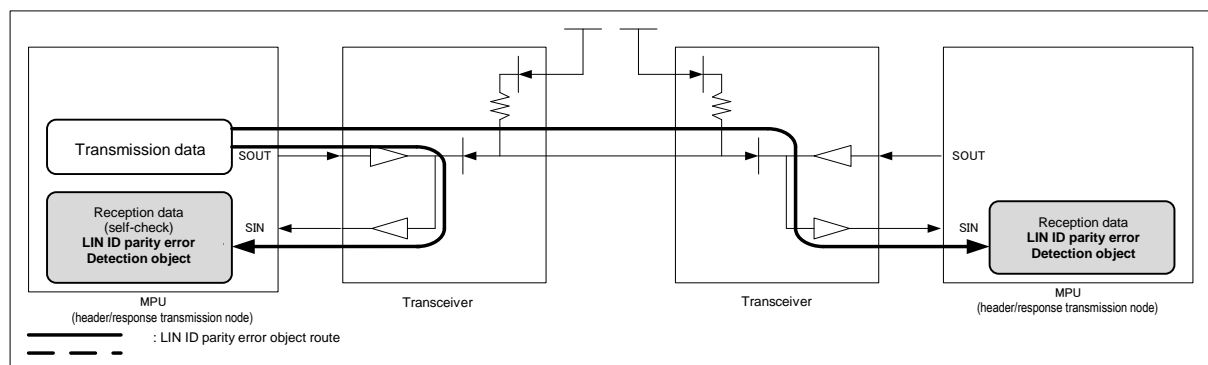
**Notes:**

- A LIN Sync Data error is detected in the slave (SCR:MS=1) in assist mode (LAMCR:LAMEN=1).
- When a LIN Sync Data error is detected (LAMESR:LSFER=1), the reception processing of the header and transmission/reception processing for the response in assist mode stops.
- Both master and slave modes can be set; however, a pseudo LIN Sync Data error can be generated only in master mode.
- Set (SCR:LBR=1) before LIN communication starts.
- Set the LIN bus error pseudo error setting bit (LAMERT:LBSERT=1) at the same time as this bit (LAMERT:LSFERT=1) is set.

**d) Overview of Pseudo LIN ID Parity Error Test Mode**

During the pseudo LIN ID parity error test, a parity bit is transmitted after being inverted.

Figure 4-4 Overview of Pseudo LIN ID Parity Error Test Mode



To activate pseudo LIN ID parity error test mode, it is necessary to enable the LIN ID parity error pseudo error setting bit (LAMERT:LPTERT=1) according to the method used to activate the pseudo error test mode.

When the pseudo LIN ID parity error pseudo error setting (LAMERT:LPTERT=1) is made on the master before the start bit of the ID Field, the master outputs the parity values (2 bits) to the ID Field after inverting them all when transmitting the ID Field.

When receiving the ID Field, a LIN ID parity error occurs and the flag bit (LAMESR:LPTER) is set to "1".

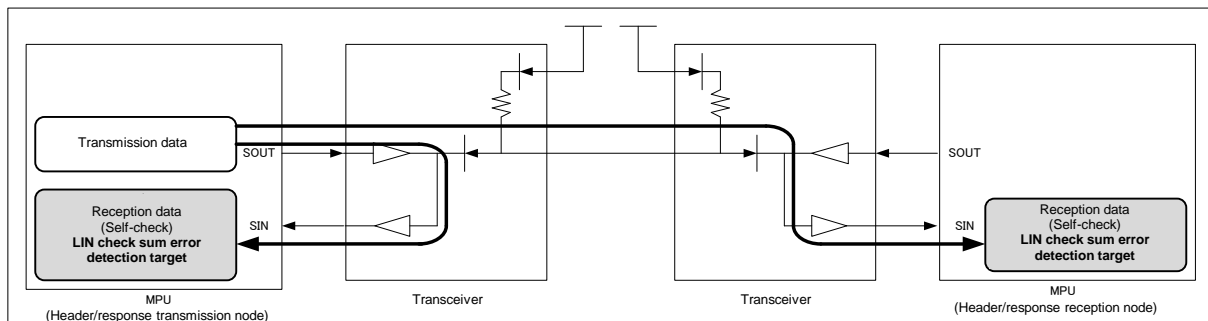
**Notes:**

- The transmission/reception processing for a response in assist mode stops once a LIN ID parity error is detected (LAMESR:LPTER=1).
- Both master and slave mode can be set; however, a pseudo LIN parity error can be generated only in master mode.
- Set (SCR:LBR=1) before the LIN communication starts.

**e) Overview of Pseudo LIN Checksum Error Test Mode**

During the pseudo LIN checksum error test, the checksum data is transmitted after being inverted.

**Figure 4-5 Overview of Pseudo LIN Checksum Error Test Mode**



To activate pseudo LIN checksum error test mode, it is necessary to enable the LIN checksum error pseudo error setting bit (LAMERT:LCSERT=1) according to the method used to activate pseudo error test mode.

When transmitting the checksum, the checksum data is transmitted after being inverted. At this time, the flag bit is set to "1" (LAMESR:LCSER) because the LIN bus is monitored.

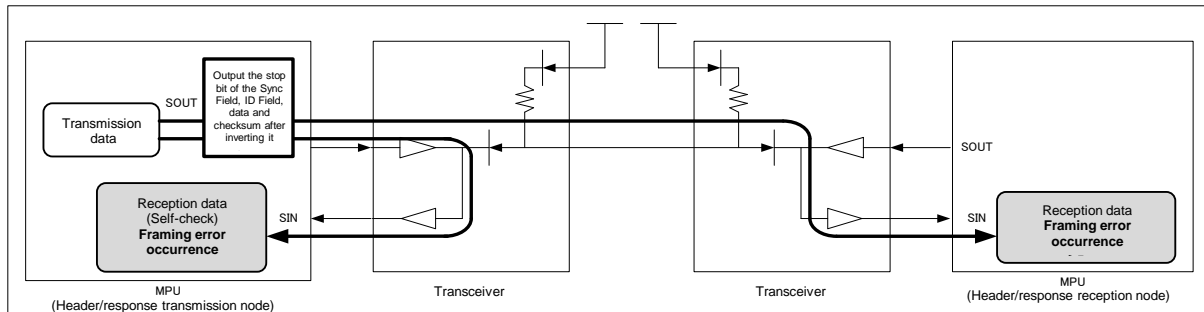
**Notes:**

- In master mode, set (SCR:LBR=1) before LIN communication starts.
- In slave mode, make this setting before transmitting a response.

#### f) Overview of Pseudo Framing Error Test Mode

During the pseudo framing error test, the stop bit is transmitted after being inverted.

Figure 4-6 Overview of Pseudo Framing Error Test Mode



To activate pseudo framing error test mode, it is necessary to enable the framing error pseudo error setting bit (LAMERT:FRET=1) according to the method used to activate pseudo error test mode.

Activating pseudo framing error test mode causes the following operation to be performed.

- Master  
When the framing error pseudo error setting is made (LAMERT:FRET=1) before the stop bit of each Field, the stop bit value ("H" level) is output after being inverted when transmitting the Sync Field, ID Field, data, and checksum.  
During reception, a framing error occurs and the flag bit (SSR:FRE) is set to "1".
- Slave  
When the framing error pseudo error setting is made (LAMERT:FRET=1) before the stop bit of each Field, the stop bit value ("H" level) is output after being inverted when transmitting the data and checksum.  
During reception, a framing error occurs and the flag bit (SSR:FRE) is set to "1".

**Note:**

- When a framing error is detected (SSR:FRE=1), transmission/reception processing of the header and response sections in assist mode stops.



## 5. Dedicated Baud Rate Generator

The transmission and reception clock source for the LIN interface (v2.1) can be selected from the following.

- Dedicated baud rate generator (reload counter)
- Input of an external clock to the baud rate generator (reload counter)

### LIN Interface (v2.1) Baud Rate

A baud rate can be selected from the following 2 types:

#### a) Baud Rate Obtained by Dividing the Internal Clock by the Dedicated Baud Rate Generator (Reload Counter)

There are 2 internal reload counters, each of which corresponds to the transmission or reception serial clock. A baud rate can be selected by setting a 15-bit reload value in the baud rate generator register 1 or 0 (BGR1, BGR0).

The reload counters divide the internal clock according to the set value.

Set the clock source by selecting an internal clock (BGR1:EXT=0).

#### b) Baud Rate Obtained by Dividing an External Clock by the Dedicated Baud Rate Generator (Reload Counter)

An external clock is used as the clock source for the reload counter.

A baud rate can be selected by setting a 15-bit reload value in the baud rate generator register 1 or 0 (BGR1, BGR0).

The reload counters divide the external clock according to the set value.

Set a clock source by selecting the use of an external clock and the baud rate generator clock (BGR1:EXT=1).

This mode is provided for dividing and using an oscillator with a special frequency.

#### Notes:

- *Make the setting of the external clock (BGR1:EXT=1) under the condition whereby the reload counter is stopped (BGR1/0 = 0x0000).*
- *When the external clock setting is specified (BGR1:EXT=1), the "H" width and "L" width of the external clock must be 2 bus clocks or more.*

## 5.1. Setting Baud Rate

This section describes baud rate setting. This section also describes the result of calculating the serial clock frequency.

### (1) Calculation of Baud Rate

The 2 15-bit reload counters are set by baud rate generator registers 1, 0 (BGR1, BGR0).  
The baud rate calculation formulas are shown below.

#### a) Reload value

$$V = \text{Phy} / b - 1$$

V : reload value    b : baud rate    Phy : frequency of bus clock or that of external clock

#### b) Calculation example

When bus clock is 16MHz, using internal clock, baud rate is 19200 bps, the reload value can be calculated as follows.

reload value :

$$V = (16 \times 1000000) / 19200 - 1 = 832$$

then baud rate will be

$$b = (16 \times 1000000) / (832 + 1) = 19208\text{bps}$$

#### c) Baud rate error

The baud rate error is obtained by the following formula.

$$\text{error rate}(\%) = (\text{calculated value} - \text{target value}) / \text{target value} \times 100$$

(ex. When bus clock is 20MHz, target baud rate is 153600 bps,

the result will be as follows.

$$\text{reload value} = (20 \times 1000000) / 153600 - 1 = 129$$

$$\text{calculated baud rate} = (20 \times 1000000) / (129 + 1) = 153846 \text{ (bps)}$$

$$\text{error rate}(\%) = (153846 - 153600) / 153600 \times 100 = 0.16 \text{ (\%)}$$

#### Notes:

- Setting the reload value to "0" stops the reload counter.
- When the reload value is even, the "L" width of the serial clock is 1-bus-clock-cycle longer than the "H" width. When the reload value is odd, the "L" width and the "H" width of the serial clock are equal.
- Set the reload value to 3 or larger. However, data may not be received normally depending on the baud rate errors and the reload value setting.
- When considering the tolerable baud rate range, also consider the impact of any jitter in the clock input on the macro.





## (2) Example of Reload Values and Baud Rate Settings for Individual Bus Clock Frequencies

The following table shows example reload values and baud rate settings.

Table 5-1 Example Reload Values and Baud Rate Settings

Baud Rate (bps)	8MHz		10MHz		16MHz		20MHz		24MHz		32MHz	
	Value	ERR	Value	ERR	Value	ERR	Value	ERR	Value	ERR	Value	ERR
8M	-	-	-	-	-	-	-	-	-	-	3	0
6M	-	-	-	-	-	-	-	-	3	0	-	-
5M	-	-	-	-	-	-	3	0	-	-	-	-
4M	-	-	-	-	3	0	4	0	5	0	7	0
2.5M	-	-	3	0	-	-	7	0	-	-	-	-
2M	3	0	4	0	7	0	9	0	11	0	15	0
1M	7	0	9	0	15	0	19	0	23	0	31	0
500k	15	0	19	0	31	0	39	0	47	0	63	0
460.8k	-	-	-	-	-	-	-	-	51	0.16	-	-
250k	31	0	39	0	63	0	79	0	95	0	127	0
230.4k	-	-	-	-	-	-	86	-0.22	103	0.16	138	-0.08
153.6k	51	0.16	64	0.16	103	0.16	129	0.16	155	0.16	207	0.16
125k	63	0	79	0	127	0	159	0	191	0	255	0
115.2k	-	-	86	-0.22	138	-0.08	173	-0.22	207	0.16	277	-0.08
76.8k	103	0.16	129	0.16	207	0.16	259	0.16	312	-0.16	416	-0.08
57.6k	138	-0.08	173	-0.22	277	-0.08	346	0.06	416	-0.08	555	-0.08
38.4k	207	0.16	259	0.16	416	-0.08	520	-0.03	624	0	832	0.04
28.8k	277	-0.08	346	0.06	555	-0.08	693	0.06	832	0.04	1110	0.01
19.2k	416	-0.08	520	-0.03	832	0.04	1041	-0.03	1249	0	1666	-0.02
10417	767	<0.01	959	<0.01	1535	<0.01	1919	<0.01	2303	<0.01	3071	<0.01
9600	832	0.04	1041	-0.03	1666	-0.02	2082	0.02	2499	0	3332	0.01
7200	1110	0.01	1388	<0.01	2221	0.01	2777	<0.01	3332	0.01	4443	0.01
4800	1666	-0.02	2082	0.02	3332	0.01	4166	<0.01	4999	0	6666	<0.01
2400	3332	0.01	4166	<0.01	6666	<0.01	8332	<0.01	9999	0	13332	<0.01
1200	6666	<0.01	8332	<0.01	13332	<0.01	16666	<0.01	19999	0	26666	<0.01
600	13332	<0.01	16666	<0.01	26666	<0.01	-	-	-	-	-	-
300	26666	<0.01	-	-	-	-	-	-	-	-	-	-

- Value: Setting of the BGR1/0 register
- ERR: Baud rate error (%)

**Table 5-2 Example Reload Values and Baud Rate Settings (Continued from Previous Page)**

Baud Rate (bps)	40MHz		48MHz		72MHz		80MHz	
	Value	ERR	Value	ERR	Value	ERR	Value	ERR
8M	4	0	5	0	8	0	9	0
6M	-	-	7	0	11	0	-	-
5M	7	0	-	-	-	-	15	0
4M	9	0	11	0	17	0	19	0
2.5M	15	0	-	-	-	-	31	0
2M	19	0	23	0	35	0	39	0
1M	39	0	47	0	71	0	79	0
500000	79	0	95	0	143	0	159	0
460800	86	-0.22	103	0.16	155	0.16	173	-0.22
250000	159	0	191	0	287	0	319	0
230400	173	-0.22	207	0.16	312	-0.16	346	0.06
153600	259	0.16	312	-0.16	468	-0.05	520	-0.03
125000	319	0	383	0	575	0	639	0
115200	346	0.06	416	-0.08	624	0	693	0.06
76800	520	-0.03	624	0	937	-0.05	1041	-0.03
57600	693	0.06	832	0.04	1249	0	1388	<0.01
38400	1041	-0.03	1249	0	1874	0	2082	0.01
28800	1388	<0.01	1666	-0.02	2499	0	2777	<0.01
19200	2082	0.02	2499	0	3749	0	4166	-0.01
10417	3839	<0.01	4607	<0.01	6911	<0.01	7679	0
9600	4166	<0.01	4999	0	7499	0	8332	0
7200	5555	<0.01	6666	<0.01	9999	0	11110	0
4800	8332	<0.01	9999	0	14999	0	16666	0
2400	16666	<0.01	19999	0	29999	0	-	-
1200	-	-	-	-	-	-	-	-
600	-	-	-	-	-	-	-	-
300	-	-	-	-	-	-	-	-

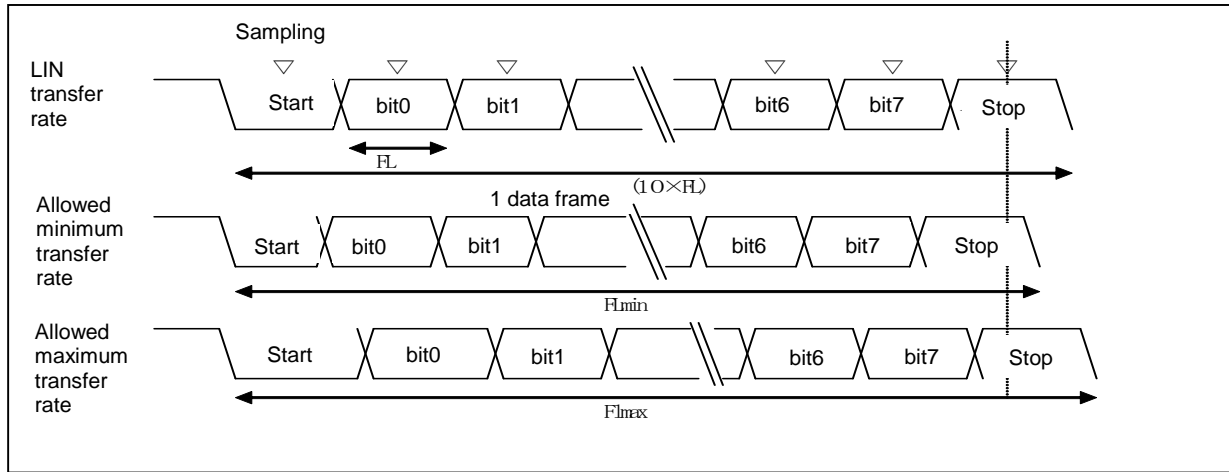
- Value: Setting of the BGR1/0 register
- ERR: Baud rate error (%)

### (3) Reception Baud Rate Tolerance

Described here is the number of errors in the receiver baud rate that can be tolerated during reception.

Use the following formula to ensure that the baud rate errors during data reception fall within the tolerable range.

Figure 5-1 Reception Baud Rate Tolerance



After the detection of a start bit, the sampling timing for the reception data is determined by the counters set in registers BGR1/0 as shown in the figure. The data can be received normally if all the data up to the last data (stop bit) is included in this sampling timing.

Theoretically, this derives the following for 10-bit data reception.

When the sampling timing margin is 1 bus clock cycle ( $\phi$ ), the tolerable minimum transfer rate ( $FL_{min}$ ) will be as follows:

$$FL_{min} = (10 \text{ bits} * (V + 1) - (V + 1) / 2 + 2) / \phi = (19V + 23) / 2 \phi \text{ (s)} \quad V: \text{Reload value}, \phi: \text{Bus clock}$$

Therefore, the maximum transmitted baud rate ( $BG_{max}$ ) that can be received will be as follows:

$$BG_{max} = 10 / FL_{min} = 20 \phi / (19V + 23) \text{ (bps)} \quad V: \text{Reload value}, \phi: \text{Bus clock}$$

When receiving data at the tolerable maximum transfer rate ( $FL_{max}$ ), sampling is done at the first point of the received 10th-bit.

Therefore, the tolerable maximum transfer rate ( $FL_{max}$ ) will be as follows:

$$9 / 10 * FL_{max} = (10 \text{ bits} * (V + 1) - (V + 1) / 2) / \phi \quad V: \text{Reload value}, \phi: \text{Bus clock}$$

$$FL_{max} = (19 / 18 * 10 * (V + 1)) / \phi$$

When the sampling timing margin ( $\phi$ ) is 2 clock cycles, the tolerable maximum transfer rate ( $FL_{max}$ ) will be as follows:

$$9 / 10 * FL_{max} = (10 \text{ bits} * (V + 1) - (V + 1) / 2 - 2) / \phi \quad V: \text{Reload value}, \phi: \text{Bus clock}$$

$$FL_{max} = (19 / 18 * 10 * (V + 1) - 40 / 18) / \phi = (190V + 150) / 20 \phi \text{ (s)} \quad V: \text{Reload value}, \phi: \text{Bus clock}$$

Therefore, the minimum transmitted baud rate ( $BG_{min}$ ) that can be received will be as follows:

$$BG_{min} = 10 / FL_{max} = 18 \phi / (19V + 15) \text{ (bps)} \quad V: \text{Reload value}, \phi: \text{Bus clock}$$

The baud rate tolerable errors for the LIN interface (v2.1) and the transmission destination will be as follows, when calculated using the above formulas for the minimum and maximum baud rate values.

Reload Value (V)	Maximum Baud Rate Error Tolerable	Minimum Baud Rate Error Tolerable
3	0%	0
10	+3.28%	-3.41%
50	+4.83%	-4.87%
100	+5.04%	-5.07%
200	+5.15%	-5.16%
32767	+5.26%	-5.26%

**Note:**

- The receiving accuracy depends on the number of bits in a frame, the bus clock, and the reload value. A higher accuracy is achieved through the use of a higher bus clock and a higher division ratio.

**(4) External Clock**

Writing "1" to the EXT bit in the baud rate generator register (BGR1) causes the baud rate generator to divide the external clock.

**Note:**

- The external clock synchronizes with the internal clock of the LIN interface (v2.1). Therefore, an external clock that cannot be synchronized causes the operation to become unstable.

**(5) Reload Counter Function**

There are the transmission reload counter and the reception reload counter, which function as dedicated baud rate generators. They consist of 15-bit registers for the reload values. They generate a transmission clock and a reception clock from an external clock or the internal clock.

**(6) Start of Counting**

When a reload value is written to the baud rate generator registers (BGR1 and BGR0), the reload counters start counting.

**(7) Restarting**

A reload counter restarts under the following conditions:

**a) For both the Transmission and Reception Reload Counters**

- Programmable reset (SCR:UPCL bit)

**b) For Reception Reload Counter**

- Detection of a falling edge of the start bit in asynchronous mode



## 6. Operation

The LIN interface (v2.1) operates using master/slave bidirectional LIN communication.

### 6.1. Manual Mode

This section describes the operations in manual mode.

#### (1) Master Operation

##### a) Selecting Master Operation

To enable operation as the master, set the SCR:MS bit to "0".

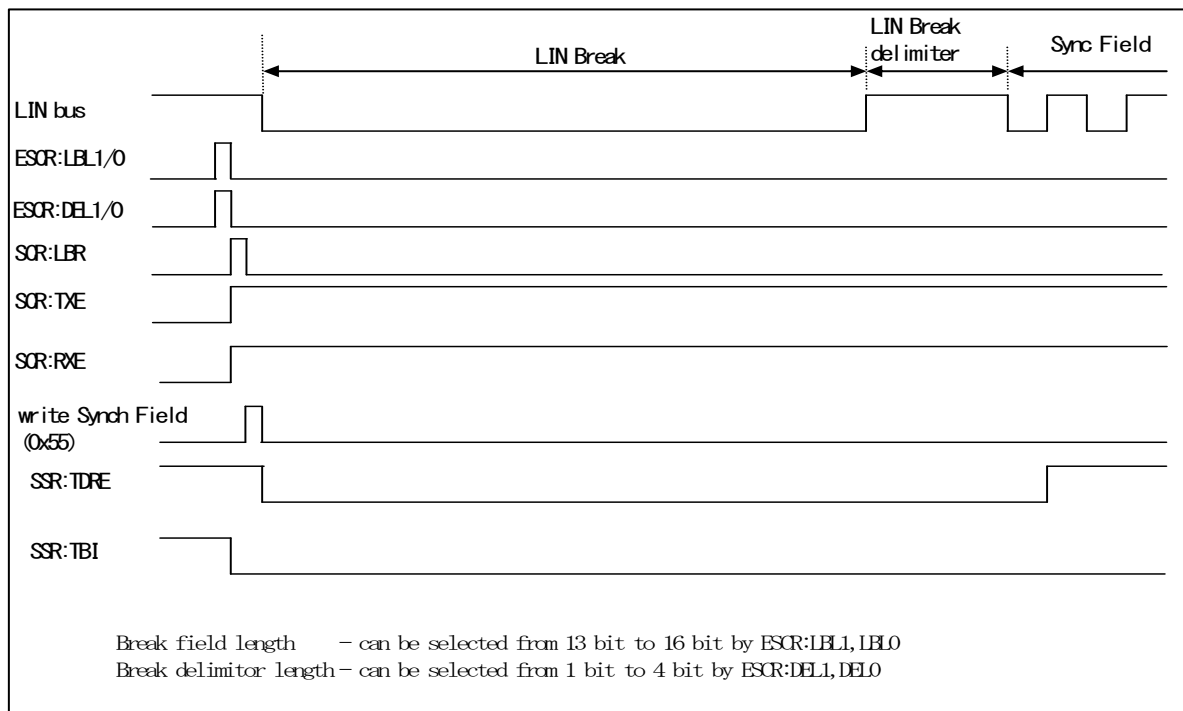
##### b) LIN Break Field Transmission to Sync Field Transmission

- The LIN Break Field length (ESCR:LBL1, LBL0) and LIN Break Field delimiter length (ESCR:DEL1, DEL0) can be selected.
- When transmission is enabled (SCR:TXE=1) and the SCR:LBR bit (LIN Break Field setting bit) is set to "1", LIN Break Field is transmitted.
- Sync Field will be transmitted by writing 0x55 to the transmission data register (TDR).

##### Notes:

- After setting the SCR:LBR bit (LIN Break Field setting bit) to "1", set the transmission data register (TDR) to 0x55.
- Even when the SCR:RXE bit (reception operation enable bit) is set to "1", the LIN Break Field part does not perform reception.

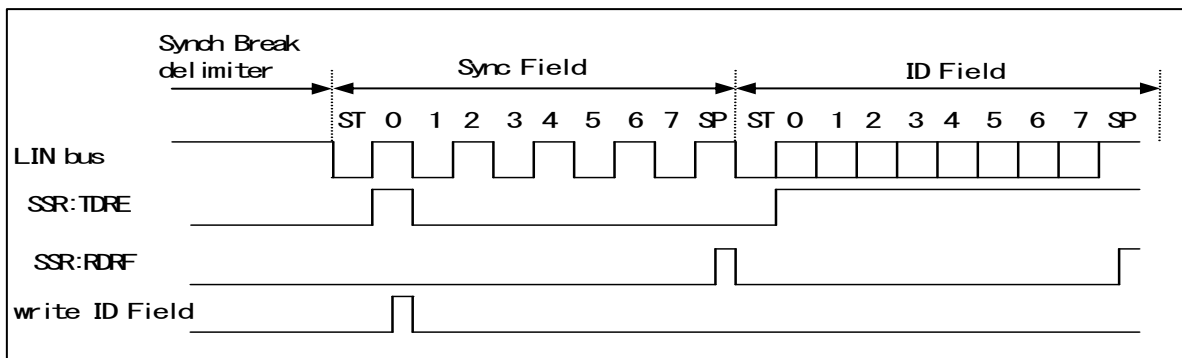
Figure 6-1 LIN Break Field to Sync Field Transmission



### c) Sync Field Transmission to ID Field Transmission

- When the 1st bit of Sync Field (0x55) is transmitted, the SSR:TDRE (transmission data empty) bit is set to "1".  
At this time, a transmission interrupt occurs if transmission interrupt is enabled (SCR:TIE=1).
- When a transmission interrupt occurs, the ID Field can be written in the transmission data register (TDR).
- When a reception interrupt occurs, the transmission data and reception data are compared to verify that there is no error.
- The data length of the ID Field is 8 bits, and the output is LSB first.

Figure 6-2 Sync Field to ID Field Transmission



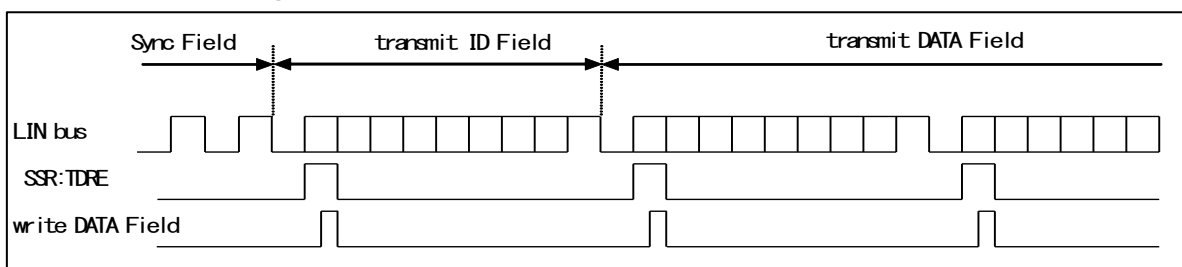
### d) ID Field Transmission to DATA Field Transmission and Reception

This step selects whether the DATA Field is to be transmitted to or received by a slave device.

(When the DATA Field is transmitted)

When the 1st bit of the ID Field is transmitted, it is set as SSR:TDRE = 1. At this time, the DATA Field can be written.

Figure 6-3 ID Field Transmission to DATA Field Transmission

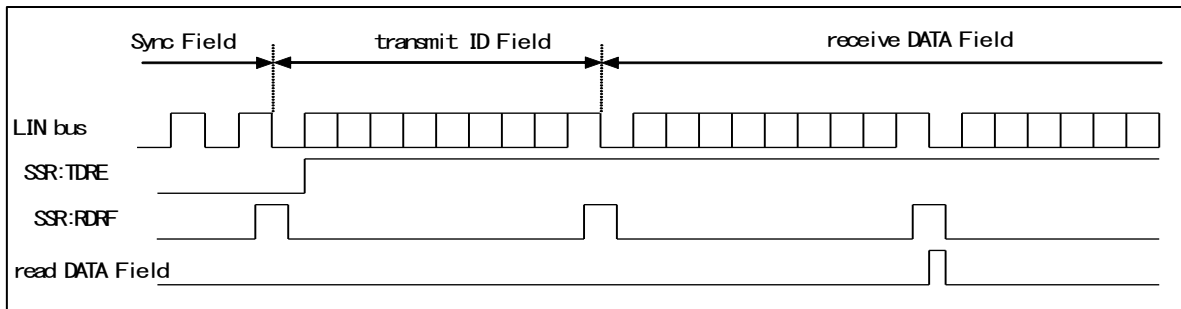


(When the DATA Field is received)

- When the 1st bit of the ID Field is transmitted, it is set as SSR:TDRE=1. However, do not write any transmission data.  
In addition, disable transmission interrupt (SCR:TIE=0).
- When the DATA Field is received, SSR:RDRF is set to 1. At this time, a reception interrupt occurs if the reception interrupt is enabled (SSR:RIE=1).
- A start bit is detected when both of the following are satisfied: 1. The noise-filtered result (the result of the rule of majority on the 3-time bus clock sampling of the serial data input) shows falling. 2. The filtered data shows "L" at the sampling point.



Figure 6-4 ID Field Transmission to DATA Field Reception

**Notes:**

- There is a built-in noise filter (the rule of majority being applied to 3-time bus clock sampling of the serial data input). However, we suggest that you design the board in such a way that noise does not pass through the filter. Alternatively, we suggest that you implement a communication method whereby noise passing through the filter does not cause a problem (by, for example, appending a checksum to the data at the end of transmission for retransmission in case of an error).
- During data reception, the following occurs if a falling edge of serial data is detected at the same time as, or 1 or 2 bus clocks earlier than, the sampling point: The edge becomes invalid and the frame that follows cannot be received normally. When frames are successively output, it is recommended that intervals be provided between the frames.

**e) Master Operation Timing Chart (When a FIFO is Not Used)**

Figure 6-5 LIN Bus Timing (When a DATA Field is Transmitted: When a FIFO is Not Used)

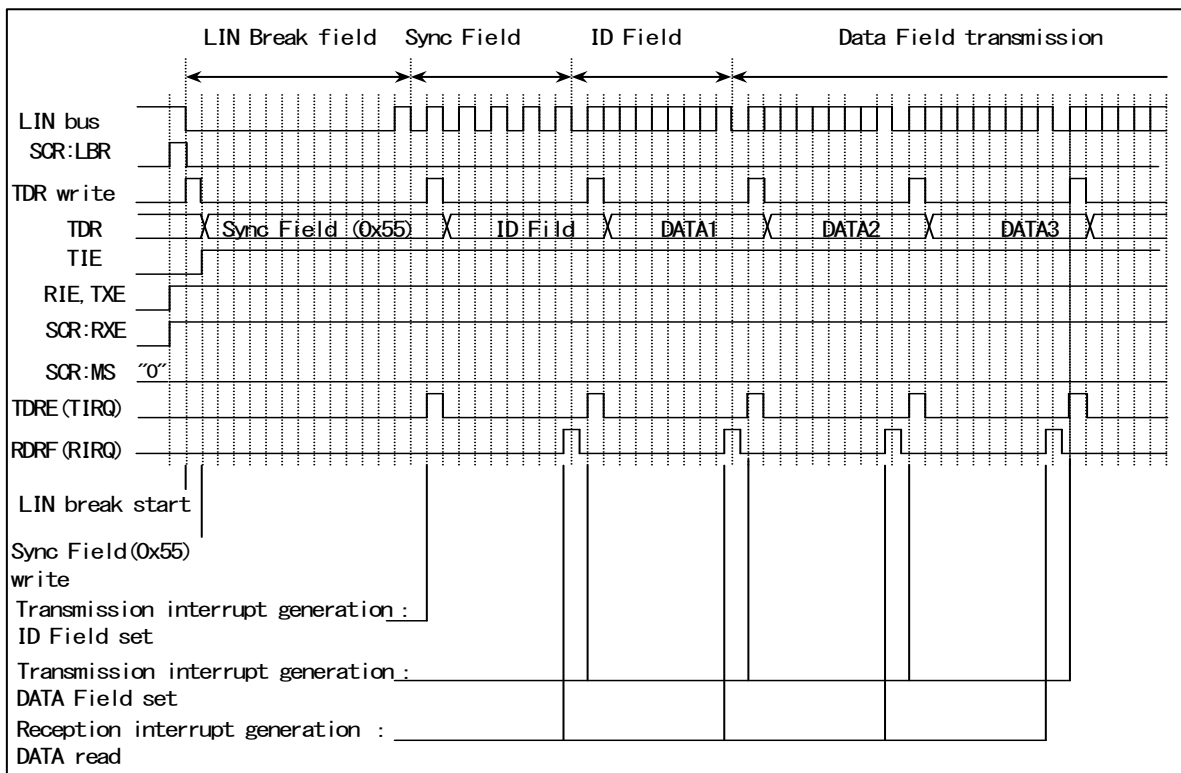
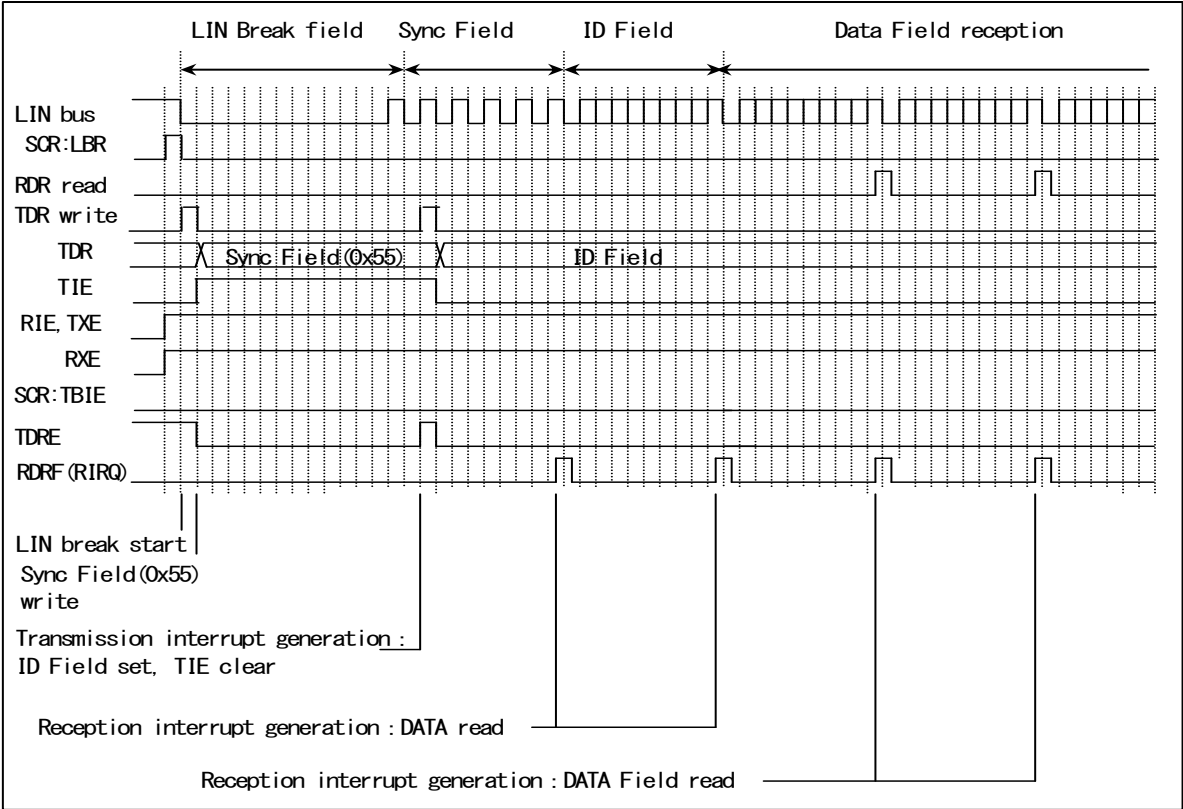




Figure 6-6 LIN Bus Timing (When a DATA Field is Received: When a FIFO is Not Used)







f) Master Operation Timing Chart (When a FIFO is Used)

Figure 6-7 LIN Bus Timing (When a DATA Field is Transmitted: When a FIFO is Used)

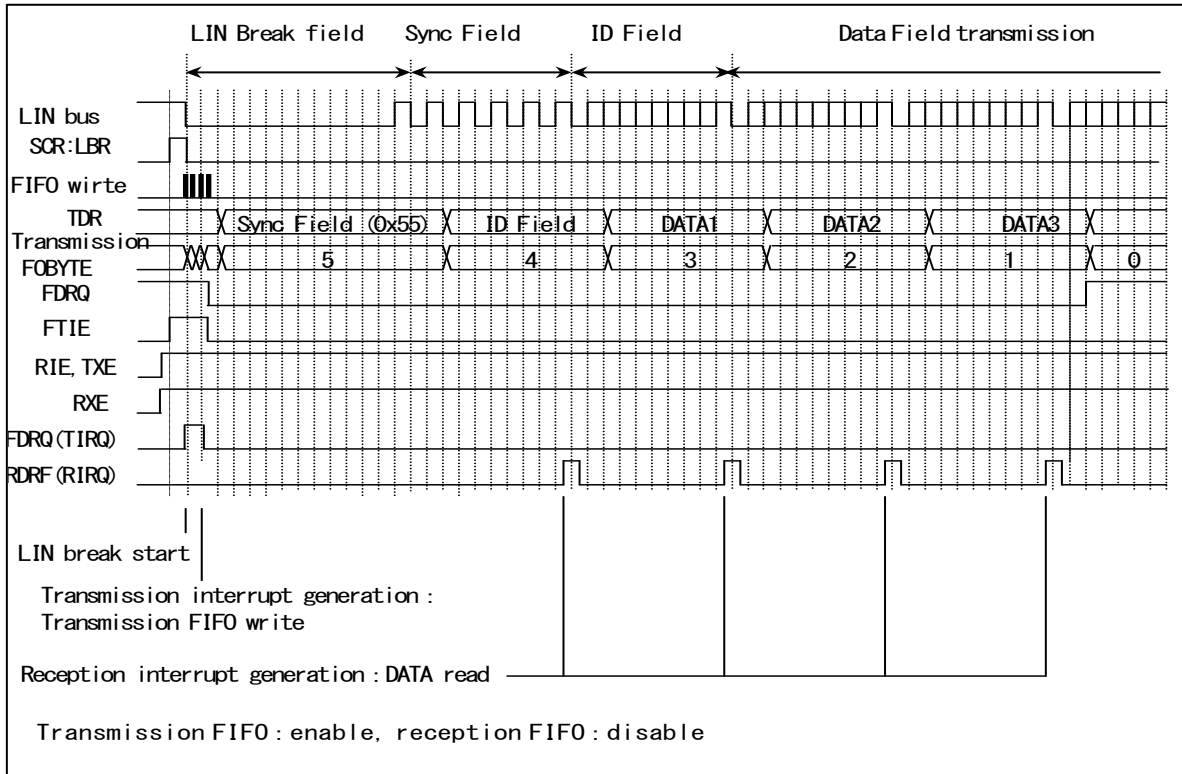
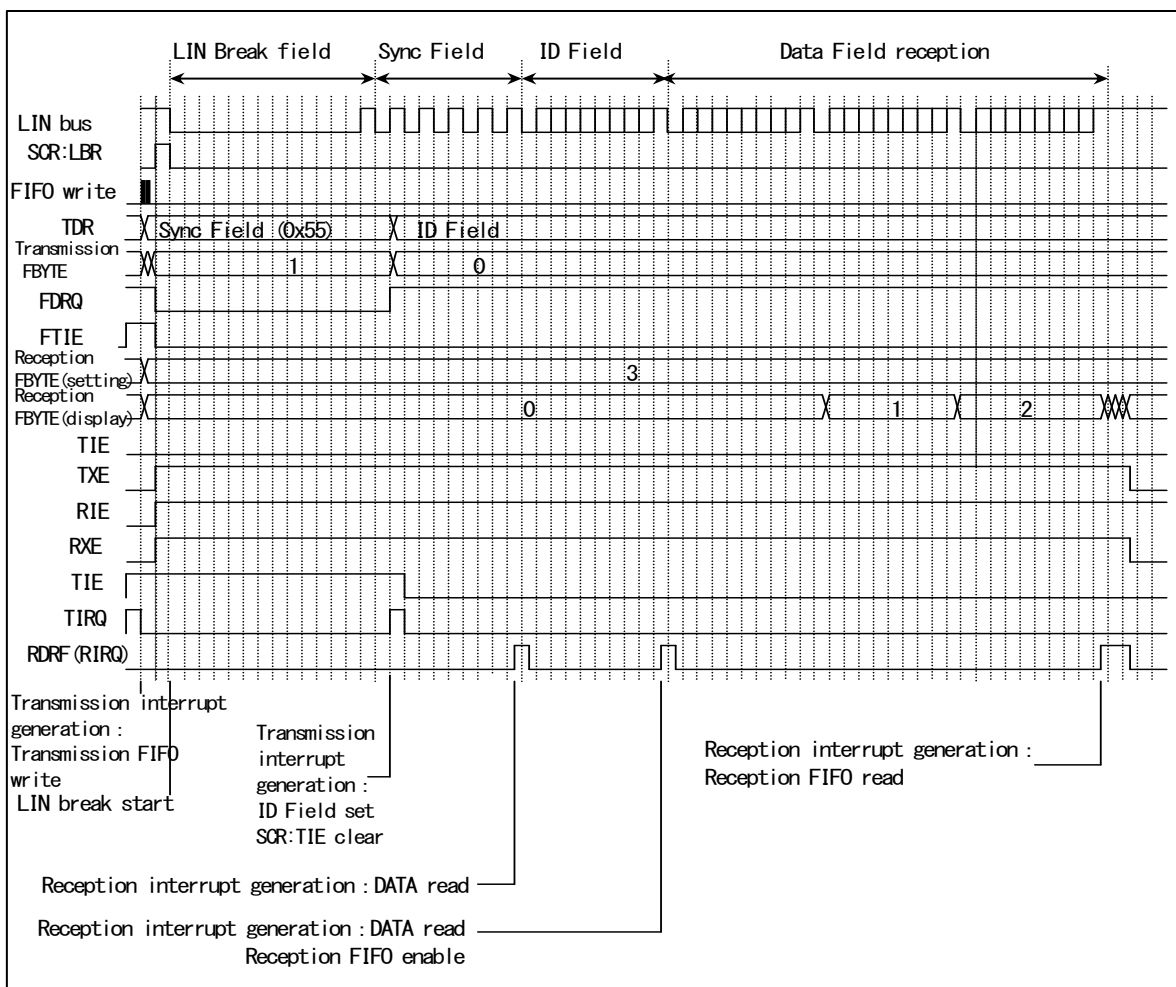


Figure 6-8 LIN Bus Timing (When a DATA Field is Received: When a FIFO is Used)





## (2) Slave Operation

### a) Selecting Slave Operation

To enable operation as a slave, set SCR:MS bit to "1".

### b) LIN Break Field Reception to Sync Field Reception

There are 2 methods to verify whether auto baud rate adjustment was applied between LIN Break Field reception and Sync Field reception, as follows:

- Method for comparing BGR and STMR
- Method for confirming the SACS:BST bit

The process using each method is as follows:

#### 1. Method for comparing BGR and STMR

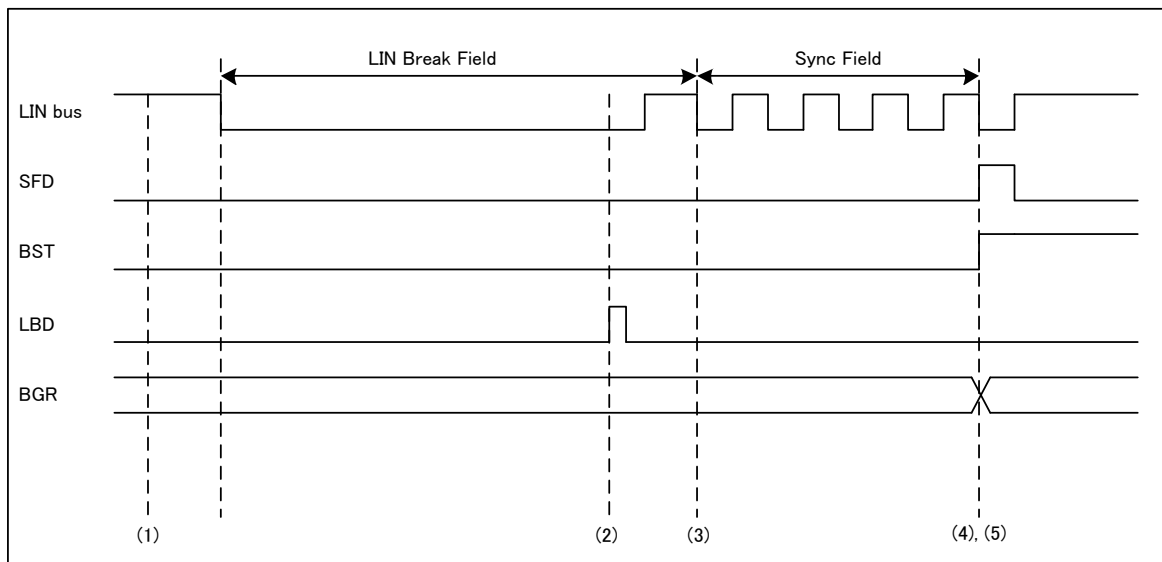
- (1) Enable auto baud rate adjustment (SACS:AUTE=1)
- (2) When the LIN Break Field is entered, the LIN Break Field is detected (SSR:LBD=1) at the 11th bit. Then a status interrupt occurs if the ESCR:LBIE bit is set to "1". After the LIN Break Field is detected (SSR:LBD=1), the serial timer is disabled (SACS:TMRE=0).
- (3) When the LIN Interface (v2.1) detects the 1st falling edge of the Sync Field, it initializes the serial timer register (STMR) to 0.
- (4) When the 5th falling edge of the Sync Field is detected, the Sync Field detection flag (SACS:SFD) is set to "1".
- (5) When the 5th falling edge of the Sync Field is detected, the Sync Field detection flag (SACS:SFD) is set to "1". Then, the following is verified to check whether auto baud rate adjustment was performed.
  - When auto baud rate adjustment was performed, the data read from the serial timer register (STMR) and baud rate generator register (BGR) are the same upon the detection of the Sync Field (SACS:SFD=1).
  - When auto baud rate adjustment was not performed, the data read from the serial timer register (STMR) and baud rate generator register (BGR) differs upon the detection of the Sync Field (SACS:SFD=1).

#### 2. Method of using the SACS:BST bit

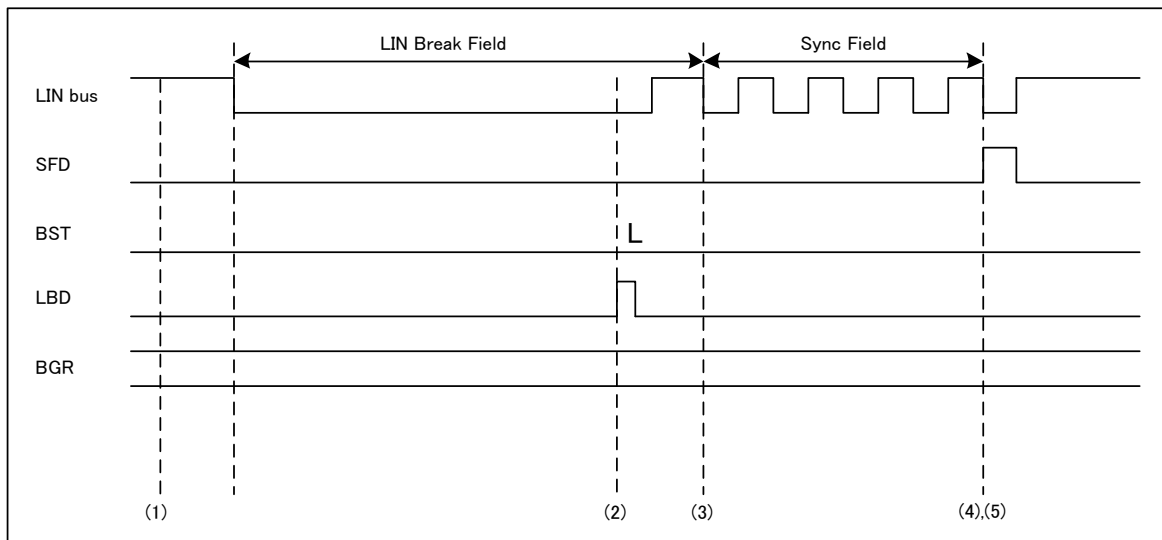
- (1) Enable auto baud rate adjustment (SACS:AUTE=1)
- (2) When the LIN Break Field is entered, the LIN Break Field will be detected (SSR:LBD=1) at the 11th bit. Then a status interrupt occurs if ESCR:LBIE bit is set to "1".
- (3) When LIN Interface (v2.1) detects the 1st falling edge of the Sync Field, it initializes serial timer register (STMR) to 0.
- (4) When the 5th falling edge of the Sync Field is detected, the Sync Field detection flag (SACS:SFD) is set to "1".
- (5) When the falling edge of the 5th Sync Field is detected, the following operation is performed depending on the value of the serial timer register (STMR).
  - When the value of serial timer register (STMR) is between that of the Sync Field lower limit register (SFLR) and Sync Field upper limit register (SFUR), the value of the serial timer register (STMR) is set in the baud rate generator register (BGR), and the baud rate setting flag (SACS:BST) is set to "1".

- When the value of serial timer register (STMR) is below that of the Sync Field lower limit register (SFLR) or exceeds that of the Sync Field upper limit register (SFUR), the baud rate generator register (BGR) will not be changed and the baud rate setting flag (SACSR:BST) is reset to "0".

**Figure 6-9 LIN Break Field Reception to Sync Field Reception (When STMR is between SFUR and SFLR)**



**Figure 6-10 LIN Break Field Reception to Sync Field Reception (When STMR is Not between SFUR and SFLR)**



**Note:**

- Disable reception (SCR:RXE=0) for the LIN Break Field and Sync Field.



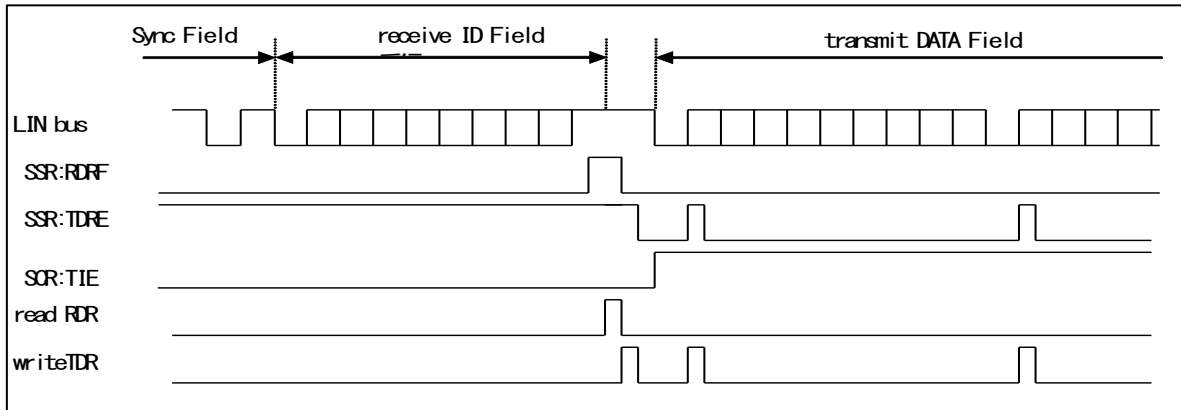
### c) ID Field Reception to DATA Field Transmission and Reception

After the reception of the ID Field, it is possible to select whether the DATA Field is to be transmitted to or received by the master device.

(When the DATA Field is transmitted)

After receiving the ID Field, write the data into the transmission data register (TDR). In this case, enable the transmission interrupt (SCR:TIE=1).

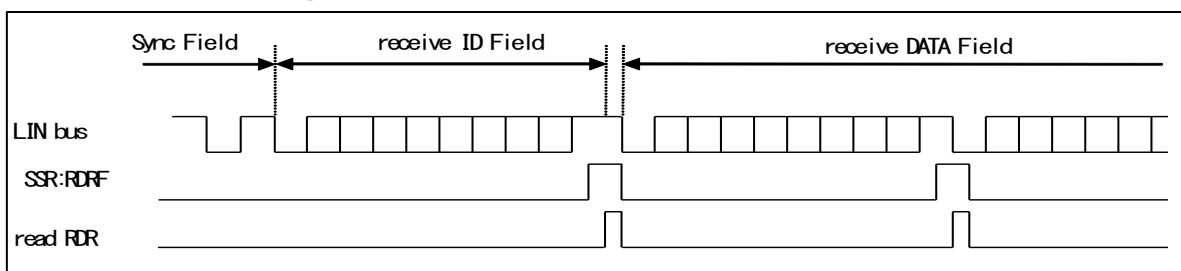
**Figure 6-11 ID Field Reception to DATA Field Transmission**



(When the DATA Field is received)

- Every time the DATA Field is received, SSR:RDRF is set to "1". At this time, a reception interrupt occurs if reception interrupt has been enabled (SCR:RIE=1).
- A start bit is detected when both of the following are satisfied: 1. The noise-filtered result (the result of the rule of majority on the 3-time bus clock sampling of the serial data input) shows falling. 2. The filtered data shows "L" at the sampling point.

**Figure 6-12 ID Field Reception to DATA Field Reception**



#### Notes:

- There is a built-in noise filter (the rule of majority being applied to 3-time bus clock sampling of the serial data input). However, we suggest that you design the board in such a way that noise does not pass through the filter. Alternatively, we suggest that you implement a communication method whereby noise passing through the filter does not cause a problem (by, for example, appending a checksum to the data at the end of transmission for retransmission in case of an error).
- During data reception, the following occurs if a falling edge of serial data is detected at the same time as, or 1 or 2 bus clocks earlier than, the sampling point: The edge becomes invalid and the frame that follows cannot be received normally. When frames are successively output, it is recommended that intervals be provided between the frames.



d) Slave Operation Timing Chart

Figure 6-13 LIN Bus Timing (When the DATA Field is Transmitted: When a FIFO is Not Used and SACSR:AUTE=1)

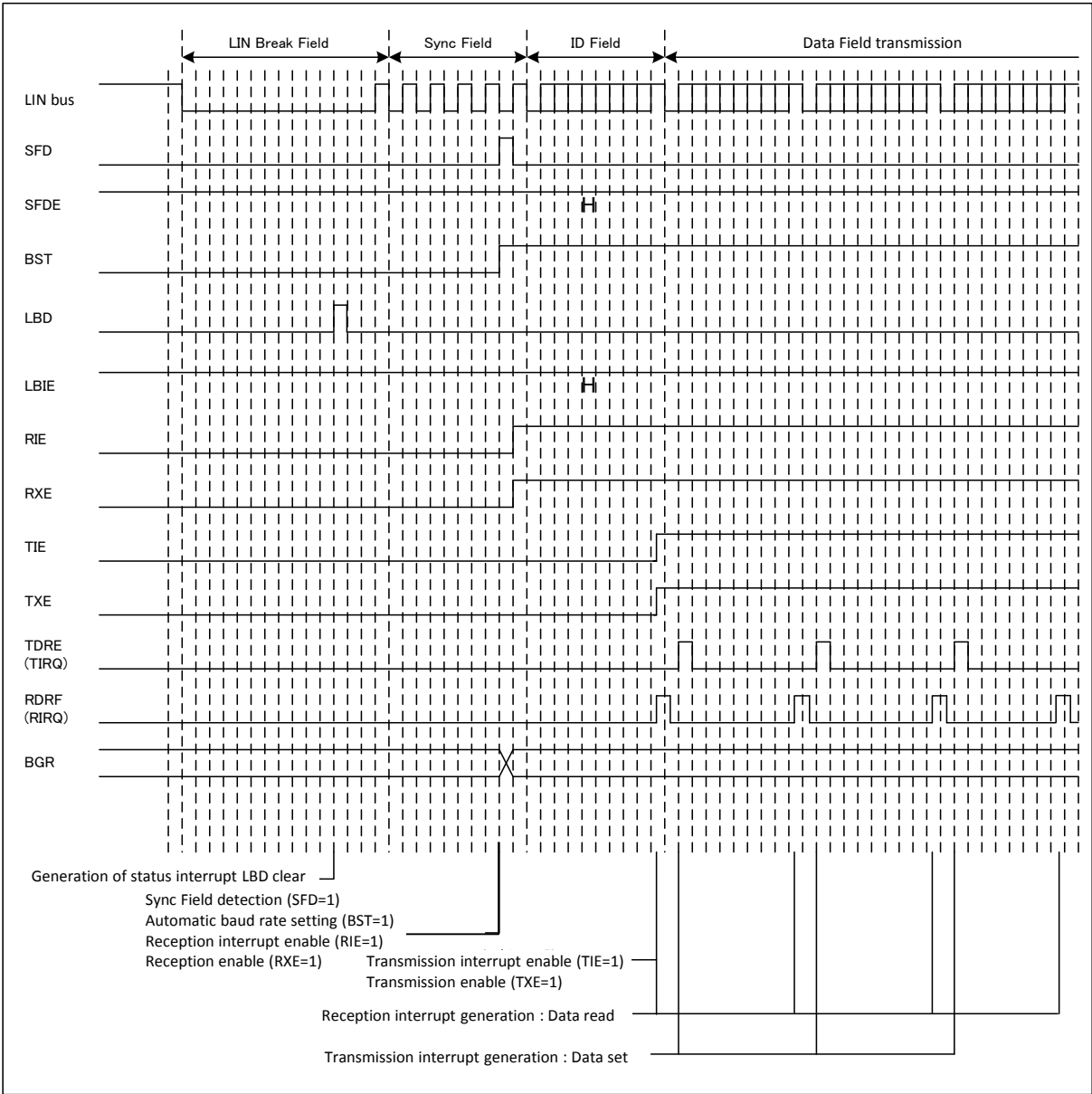
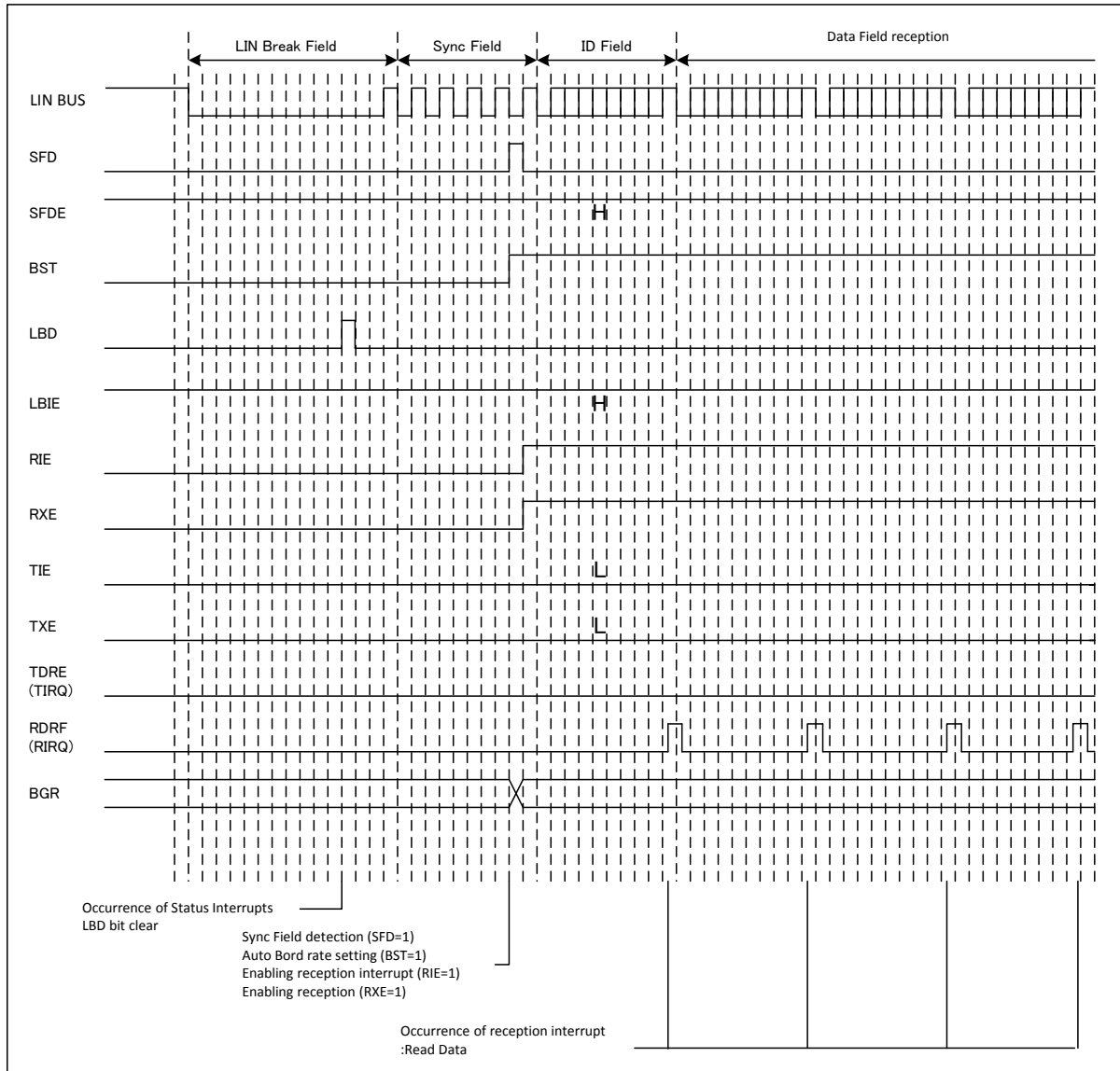




Figure 6-14 LIN Bus Timing (When the DATA Field is Received: When a FIFO is Not Used and SACSr:AUTE = 1)





When a FIFO is used

Figure 6-15 LIN Bus Timing (When DATA Field is Transmitted: When a FIFO is Used and SACSr:AUTE = 1)

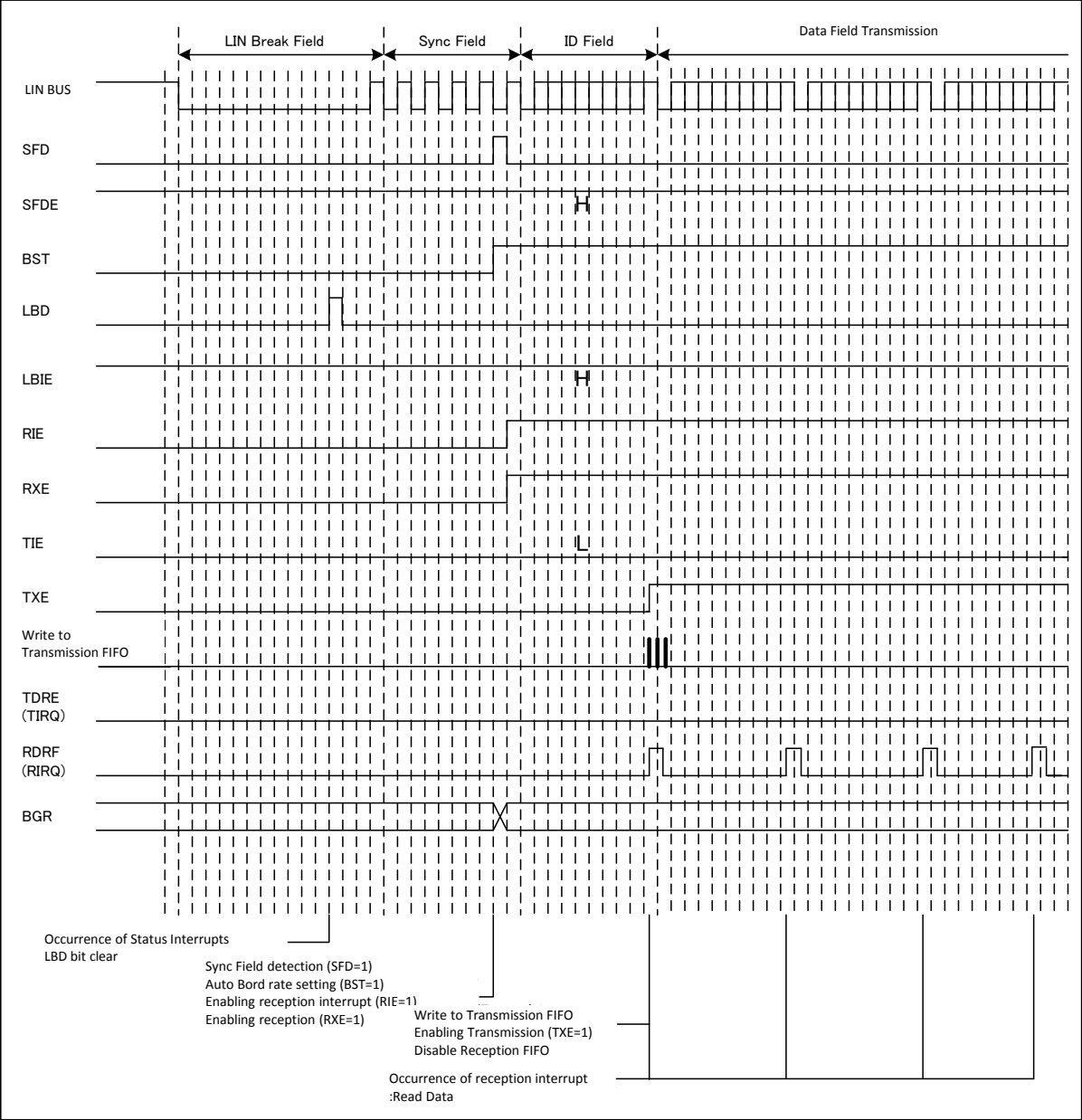
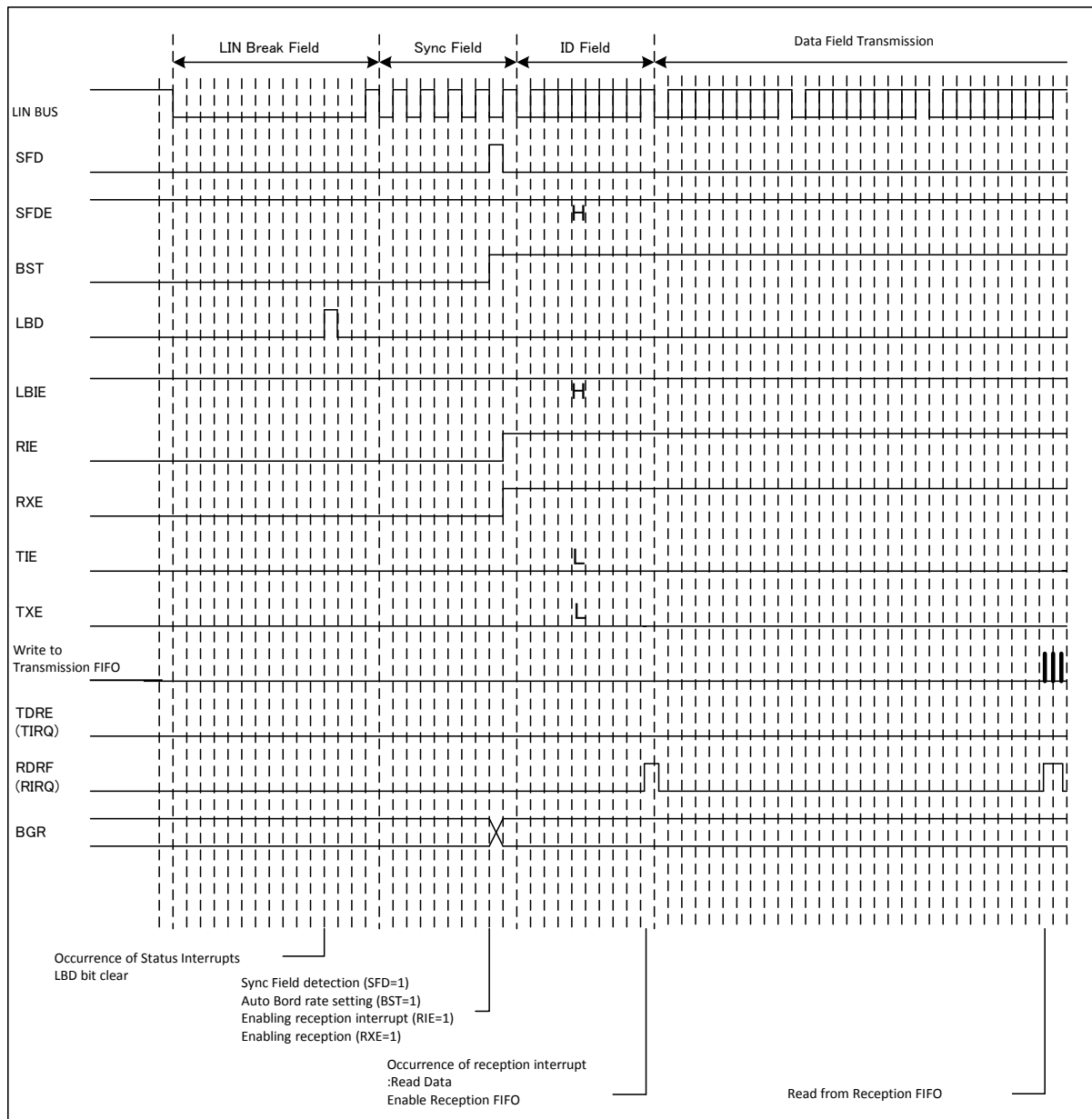






Figure 6-16 LIN Bus Timing (When DATA Field is Received: When a FIFO is Used and SACS: AUTE = 1)



## 6.2. Assist Mode

Assist mode processes the auto transmission/reception of the LIN header and the following generation and detection.

- Generation and detection of parity for ID Field
- Generation and detection of checksum
- Detection of LIN bus error

### (1) Master Operation

#### a) Auto Header Transmission Setting

To process auto header transmission using assist mode, after initial setting, set the SCR:LBR bit (LIN Break Field setting bit) to "1". Setting to "1" initiates the auto transmission of the LIN Break Field, Sync Field and ID Field. How to set transmission is described below:

- To enable operation as the master, set the SCR:MS bit (master/slave function selection bit) to "0".
- Set the LAMCR:LAMEN bit (LIN assist mode processing enable bit) to "1".
- Select and set the LIN Break Field length (ESCR:LBL2, LBL1, LBL0) and LIN Break Field delimiter length (ESCR:DEL1, DEL0).
- Select and set the stop bit length (SMR:SBL and ESCR:ESBL).
- When the assist mode transmission ID register (LAMTID) is used for ID transmission, set the LAMCR:LIDEN bit (LIN ID register enable bit) to "1".  
When the data transmission register (TDR) is used for ID transmission, set the LAMCR:LIDEN bit (LIN ID register use enable bit) to "0".
- Set the ID value in the register selected for ID transmission.
- Set the LIN data length (LAMCR:LDL3 to LDL0) which corresponds to the ID.
- Select and set the checksum to either standard or extended. (LAMCR:LCSTYP).
- Disable the LIN Break Field interrupt enable bit (ESCR:LBIE=0). When LIN Break Field interrupt enable bit is enabled (ESCR:LBIE=1), since LIN Break Field is also detected at the master, a status interrupt (SSR:LBD bit (LIN Break Field detection flag)) occurs.
- Disable Sync Field detection interrupt enable bit (SACSR:SFDE=0). When the Sync Field detection interrupt enable bit is enabled (SACSR:SFDE=1), since the Sync Field is also detected at the master, a status interrupt (SACSR:SFD bit (Sync Field detection flag)) occurs.
- Set the reception operation enable bit (SCR:RXE) to "0" (reception disabled).

#### b) LIN Break Field to ID Field Transmission

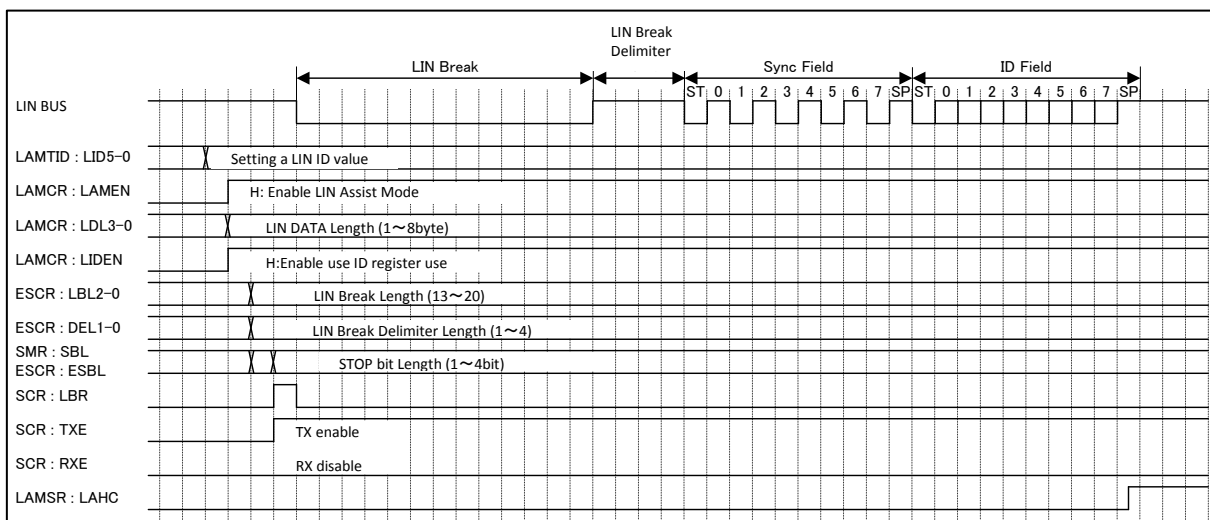
- Set the transmission operation enable bit (SCR:TXE) to "1" (transmission enabled).
- Set the LIN Break Field setting bit (SCR:LBR) to "1" (LIN Break Field generation). The LIN Break Field set at ESCR:LBL2 to LBL0 is transmitted.
- The master receives the LIN Break Field transmitted by the master, and checks for bus errors.
- After the transmission of the LIN Break Field, the LIN Break Field delimiter set at ESCR:DEL1, DEL0 will be transmitted.
- After the transmission of the LIN Break Field delimiter, the Sync Field (0x55 fixed value) is transmitted.
- The master receives the Sync Field transmitted by the master, and checks for bus error/framing errors.
- After the transmission of the Sync Field, the set ID Field value is transmitted. When the LAMCR:LIDEN bit (LIN ID register use enable bit) is set to "0", the ID Field value is transmitted as set in the transmission data register (TDR). When the LAMCR:LIDEN bit (LIN ID register use enable bit) is set to "1", the ID field value is transmitted as set in the LIN assist mode transmission ID register (LAMTID).
- When the transmission data register (TDR) is used for ID Field transmission (LAMCR:LIDEN=0), and the 1st bit of the ID Field is transmitted, the SSR:TDRE (transmission data empty) bit is set to "1". At this time, a transmission interrupt occurs if the transmission interrupt is enabled (SCR:TIE=1).  
When a transmission interrupt (TDRE) occurs, the transmission data can be written in the transmission data register (TDR).



- The data length of the ID Field is 8 bits, and the output is LSB first. The LIN parity in the ID Field is calculated automatically.
- The master receives the ID Field transmitted by the master, and checks for a bus error/framing error.
- When the ID Field transmission is completed, the LIN auto header completion flag is set (LAMSR:LAHC=1). In this case, when the LIN auto header transmission completion interrupt enable bit is enabled (LAMIER:LAHCIE=1), a status interrupt occurs.
- When the following errors occur, transmission is halted.
  - LIN bus error
  - LIN ID parity error
  - Framing error

**Notes:**

- In the header transmission setting (from LBR activation (SCR:LBR=1) to LIN auto header completion (LAMSR:LAHC=1)), the auto header transmission setting shall not be changed.
- Disable reception (SCR:RXE=0) while the master header is being transmitted in assist mode.
- While the master is operating in assist mode, the transmission data values of the Sync Field and ID Field are not stored to the RDR register.
- When response data is received without using the LIN assist mode transmission ID register (LAMTID), and the 1st bit of the ID Field is transmitted, it is set as SSR:TDRE=1. However, do not write data. In addition, disable the transmission interrupt (SCR:TIE=0).
- When the response data is received using the LIN assist mode transmission ID register (LAMTID), the transmission data register (TDR) can be written after the LIN Break Field setting bit (SCR:LBR) is set to "1". However, do not write data.

**Figure 6-17 LIN Break Field to ID Field Transmission****c) LIN Break Field Retransmission During Assist Mode Processing**

The LIN Break Field (SCR:LBR=1) shall be set at the following timing.

- After the completion of header transmission (LAMSR:LAHC=1)
- After the completion of response transmission and reception (LAMSR:LCSC=1)

#### d) Data Field Transmission and Reception

This step selects whether the DATA Field shall be transmitted to or received by a slave device.

(When the DATA Field is transmitted)

- When the LIN assist mode transmission ID register (LAMTID) is not used, after the 1st bit of the ID Field is transmitted, it is set as SSR:TDRE=1. At this time, the DATA Field can be written.
- When the LIN assist mode transmission ID register (LAMTID) is used, and the LIN Break Field setting bit (SCR:LBR) is set to "1", the DATA Field becomes writable.
- After the LIN Break Field setting bit (SCR:LBR) is set to "1" but before the start of response transmission, enable transmission (SCR:TXE=1).
- In LIN assist mode, the checksum calculation is performed automatically. For the checksum calculation, the calculation method can be selected with the LIN checksum type selection bit (LAMCR:LCSTYP).
- After the completion of the checksum calculation, the checksum calculation completion flag (LAMSR:LCSC) is set. In this case, when the checksum calculation completion interrupt enable bit is set (LAMIER:LCSCIE=1), a status interrupt occurs.
- After the completion of response transmission (LAMSR:LCSC=1), disable transmission (SCR:TXE=0).

#### Notes:

- *While a response is being transmitted during an assist mode operation, do not enable reception (SCR:RXE=1).*
- *During assist mode operation, response transmission data (Data Field, checksum) cannot be stored into the RDR register.*
- *When a FIFO is being used, after setting the LIN Break Field setting bit (SCR:LBR) to "1" (LIN Break Field generation bit), write data to the FIFO.*
- *When the LIN data length is set to 0 bytes (LAMCR:LDL3 to LDL0 = 0000) for response transmission, write dummy data into the TDR register to automatically perform the checksum calculation and then transmit the data. (Any value can be written.) The TDR setting value does not affect the checksum calculation.*
- *When the LIN data length is set to "0" (LAMCR:LDL3 to LDL0 = 0b0000), the checksum values will be as follows:*
  - *With the standard checksum setting (LAMCR:LCSTYP=0), the checksum value is 0xFF.*
  - *With the extended checksum setting (LAMCR:LCSTYP=1), the checksum value is the inverse value of the ID Field.*
- *During the transmission of response data, do not change the transmission data length (LAMCR:LDL[3:0]).*



Figure 6-18 ID Field Transmission to DATA Field Transmission (When ID Register is Used and a FIFO is Not Used)

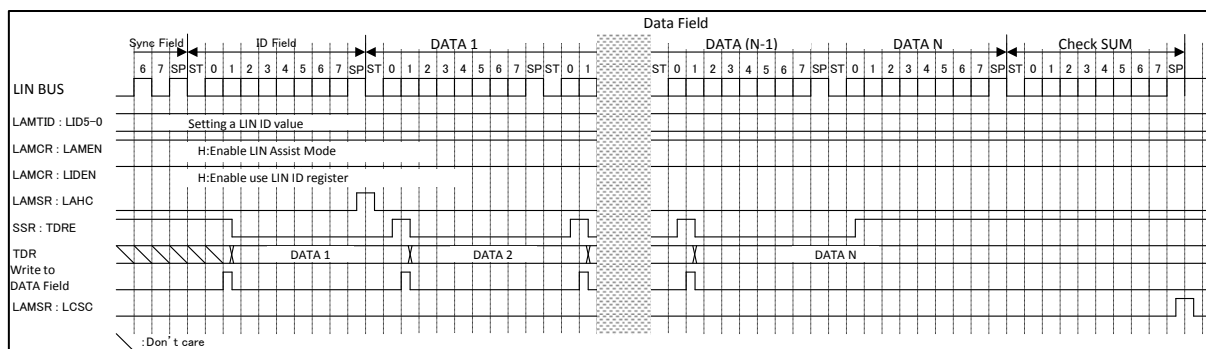
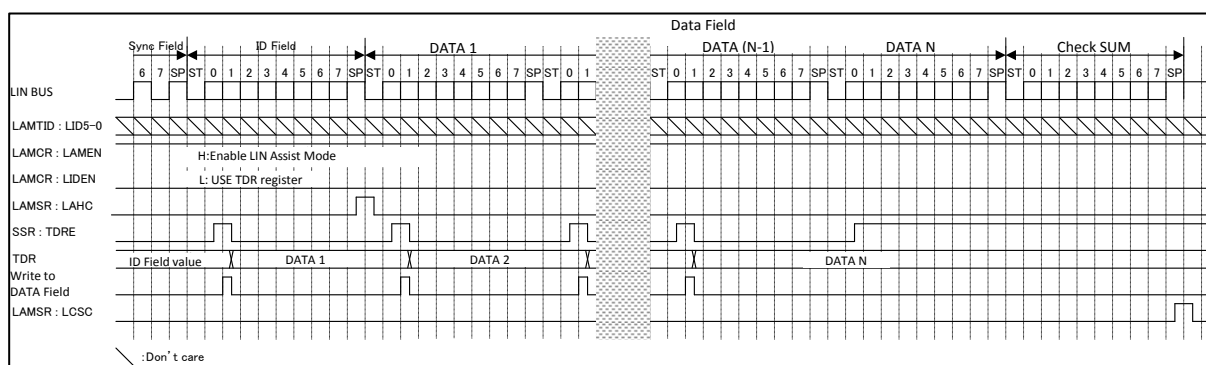


Figure 6-19 ID Field Transmission to DATA Field Transmission (When ID Register is Not Used and a FIFO is Not Used)



(When the DATA Field is received)

- Between LIN auto header completion (LAMSR:LAHC=1) and the start of response reception, enable reception (SCR:RXE = 1) and disable transmission (SCR:TXE=0).
- When the DATA Field is received, SSR:RDRF is set to 1. At this time, a reception interrupt occurs if a reception interrupt is enabled (SSR:RIE=1).
- After the completion of checksum reception, the LIN checksum calculation completion flag is set (LAMSR:LCSC=1). In this case, when the checksum calculation completion interrupt enable bit is set (LAMIER:LCSCIE=1), a status interrupt occurs.
- After the completion of checksum reception (LAMSR:LCSC=1), disable reception (SCR:RXE=0).

#### Notes:

- During data reception, the following occurs if a falling edge of serial data is detected at the same time as, or 1 or 2 bus clocks earlier than, the sampling point: The edge becomes invalid and the frame that follows cannot be received normally. When frames are successively output, it is recommended that intervals be provided between the frames.
- The checksum value for response reception during assist mode operation is not stored into the RDR register.
- When the LIN data length is set to "0" (LAMCR:LDL3 to LDL0 = 0b0000), the checksum values will be as follows:
  - With the standard checksum setting (LAMCR:LCSTYP=0), the checksum value is 0xFF.
  - With the extended checksum setting (LAMCR:LCSTYP=1), the checksum value is the inverse value of the ID Field.
- During the reception of response data, do not change the reception data length (LAMCR:LDL[3:0]).



Figure 6-20 ID Field Transmission to DATA Field Reception (When the ID Register is Used and a FIFO is Not Used)

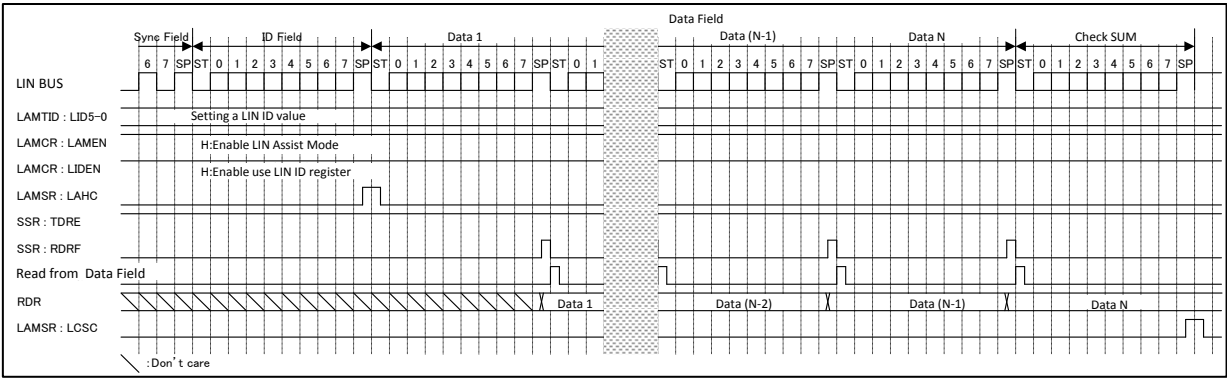
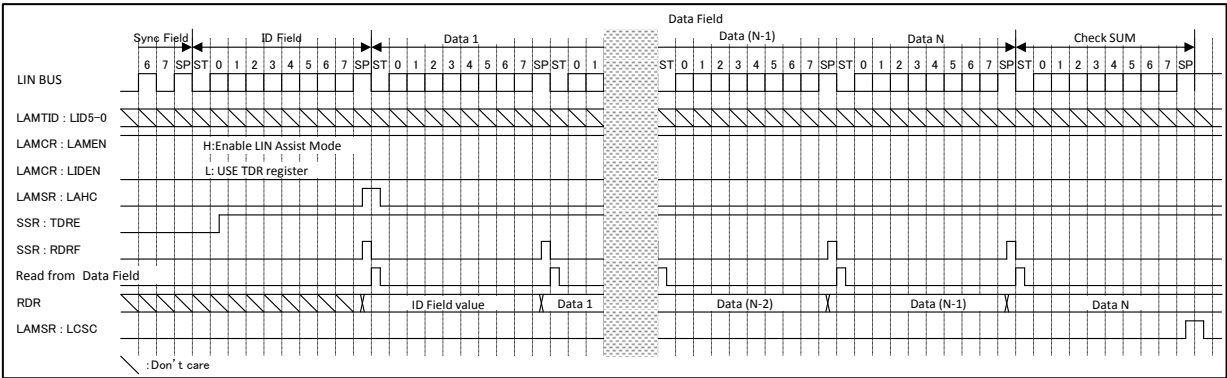


Figure 6-21 ID Field Transmission to DATA Field Reception (When the ID Register is Not Used and a FIFO is Not Used)





e) Master Operation Time Chart (When a FIFO is Not Used)

Figure 6-22 LIN Bus Timing (The ID Register is Used, the DATA Field is Transmitted, and a FIFO is Not Used)

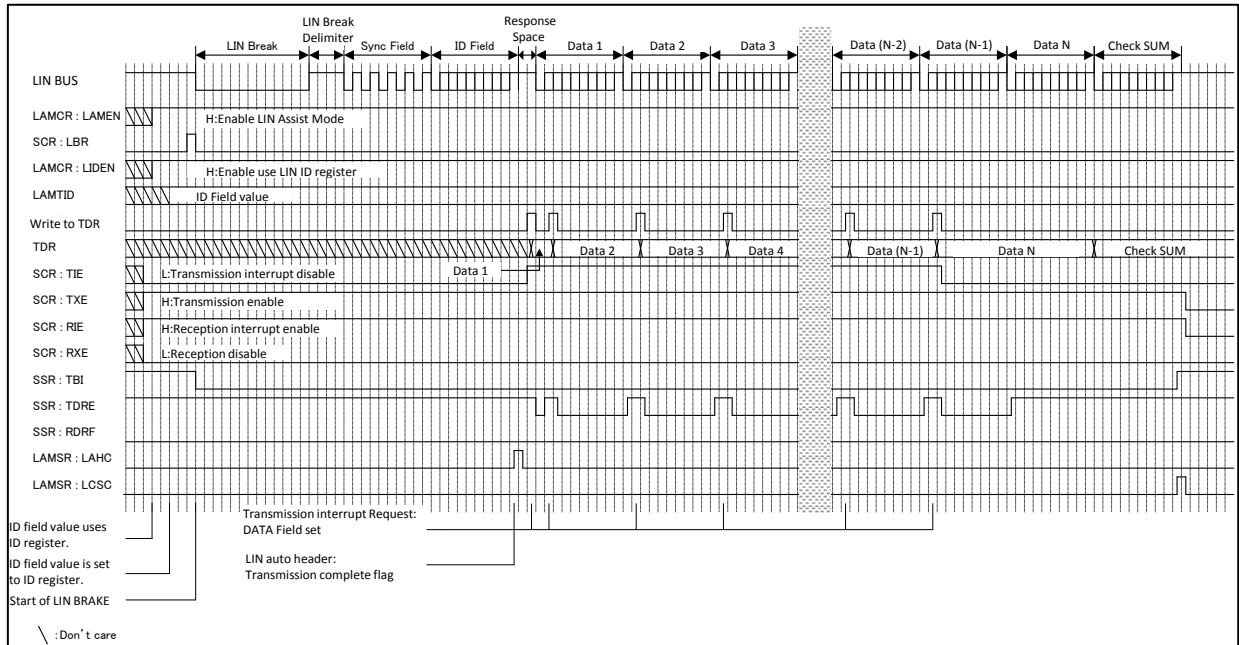


Figure 6-23 LIN Bus Timing (The ID Register is Not Used, the DATA Field is Transmitted, and a FIFO is Not Used)

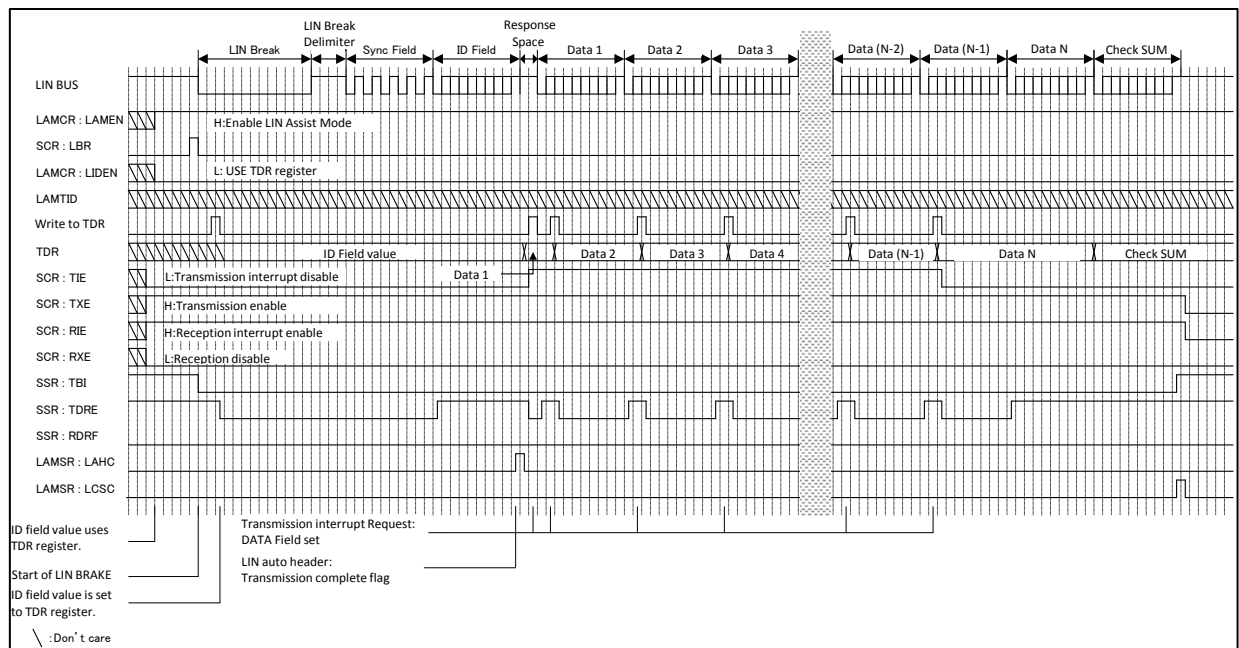






Figure 6-24 LIN Bus Timing (The ID Register is Used, the DATA Field is Received, and a FIFO is Not Used)

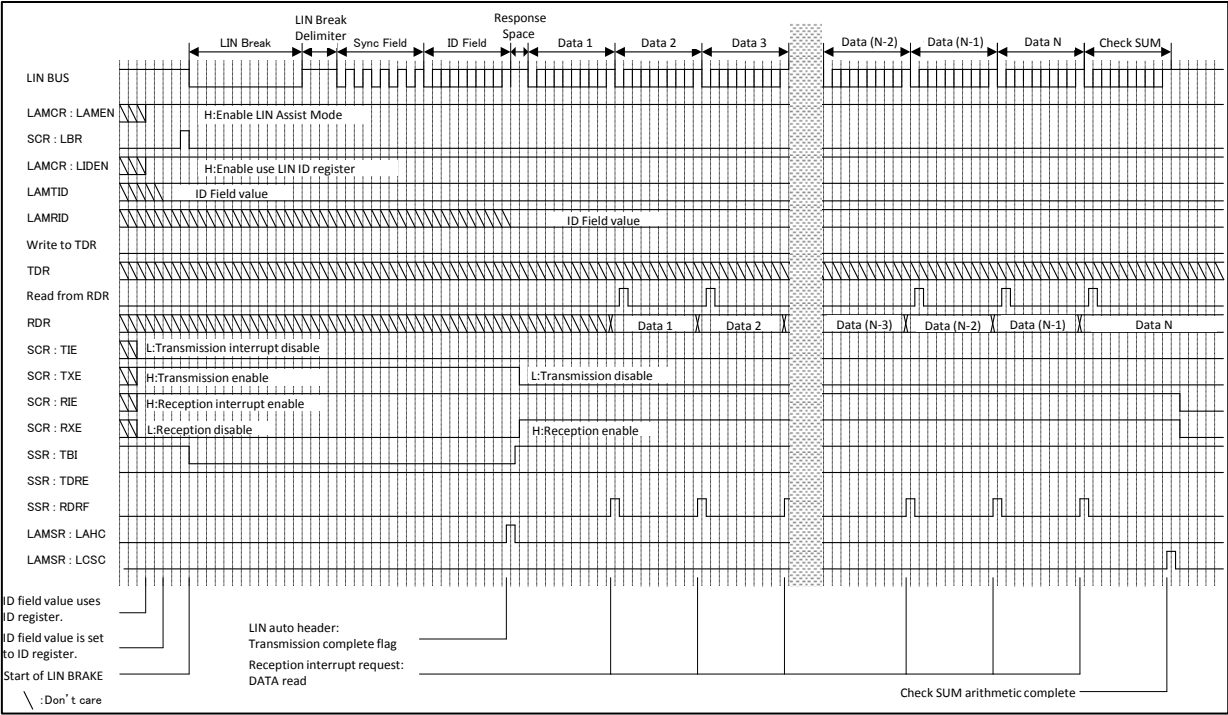


Figure 6-25 LIN Bus Timing (The ID Register is Not Used, the DATA Field is Received, and a FIFO is Not Used)

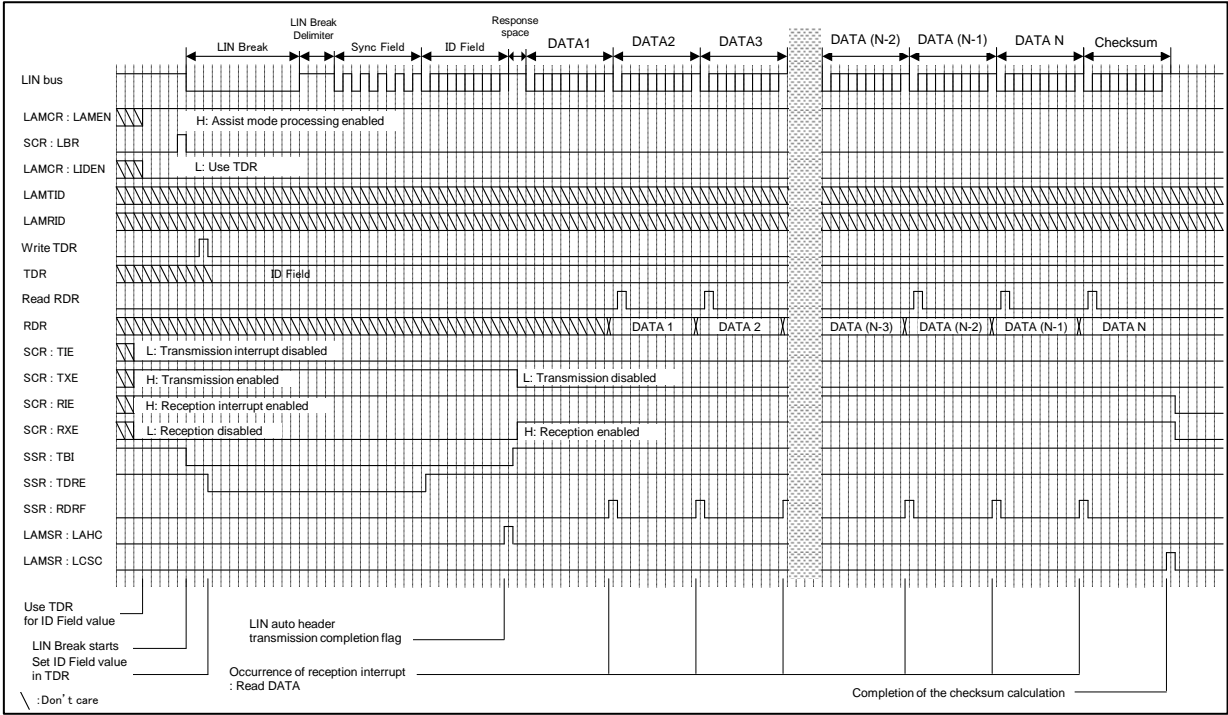






Figure 6-26 LIN Bus Timing (The ID Register is Used, the DATA Field is Transmitted, and a FIFO is Used)

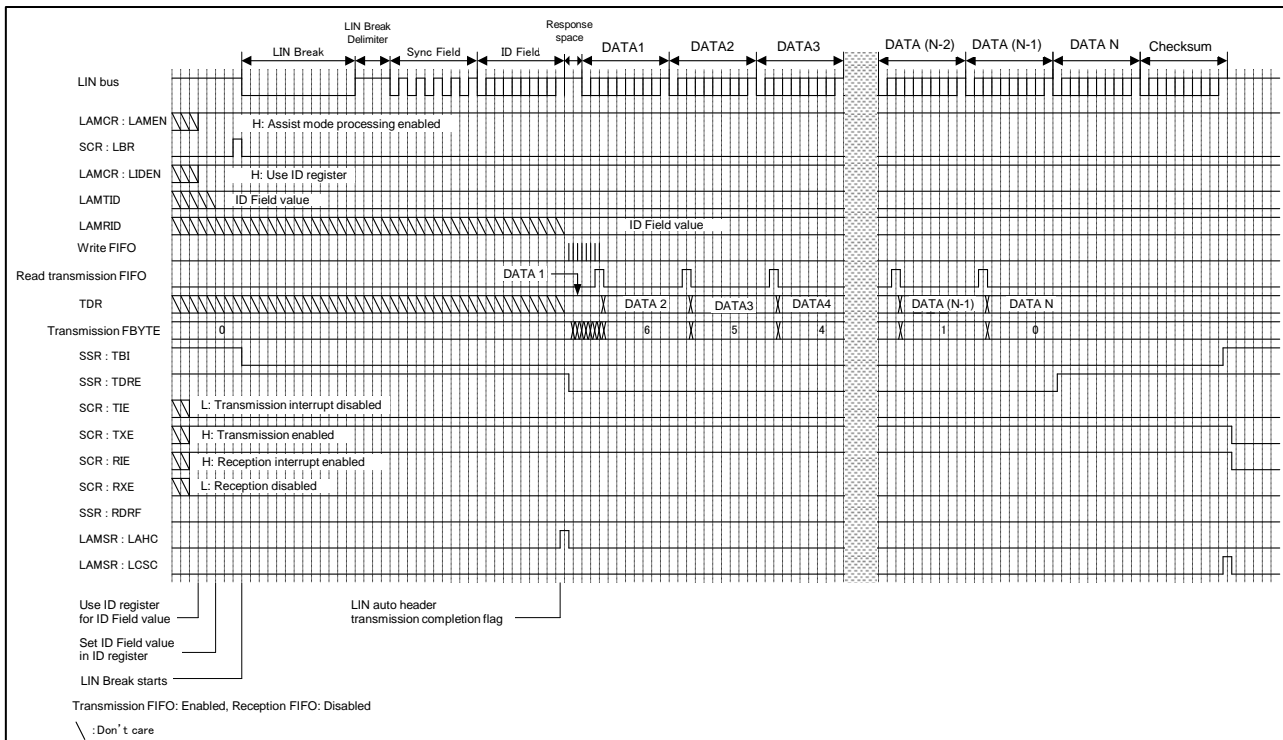


Figure 6-27 LIN Bus Timing (The ID Register is Not Used, the DATA Field is Transmitted, and a FIFO is Used)

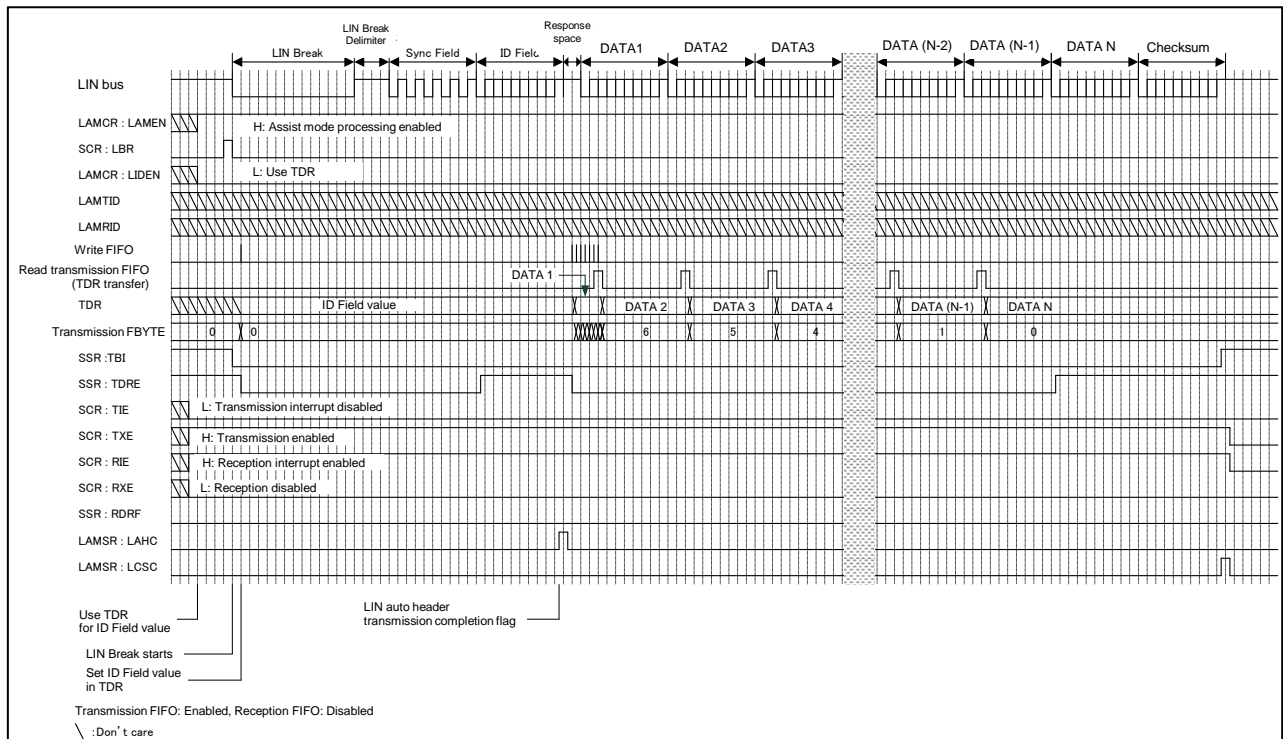


Figure 6-28 LIN Bus Timing (The ID Register is Used, the DATA Field is Received, and a FIFO is Used)

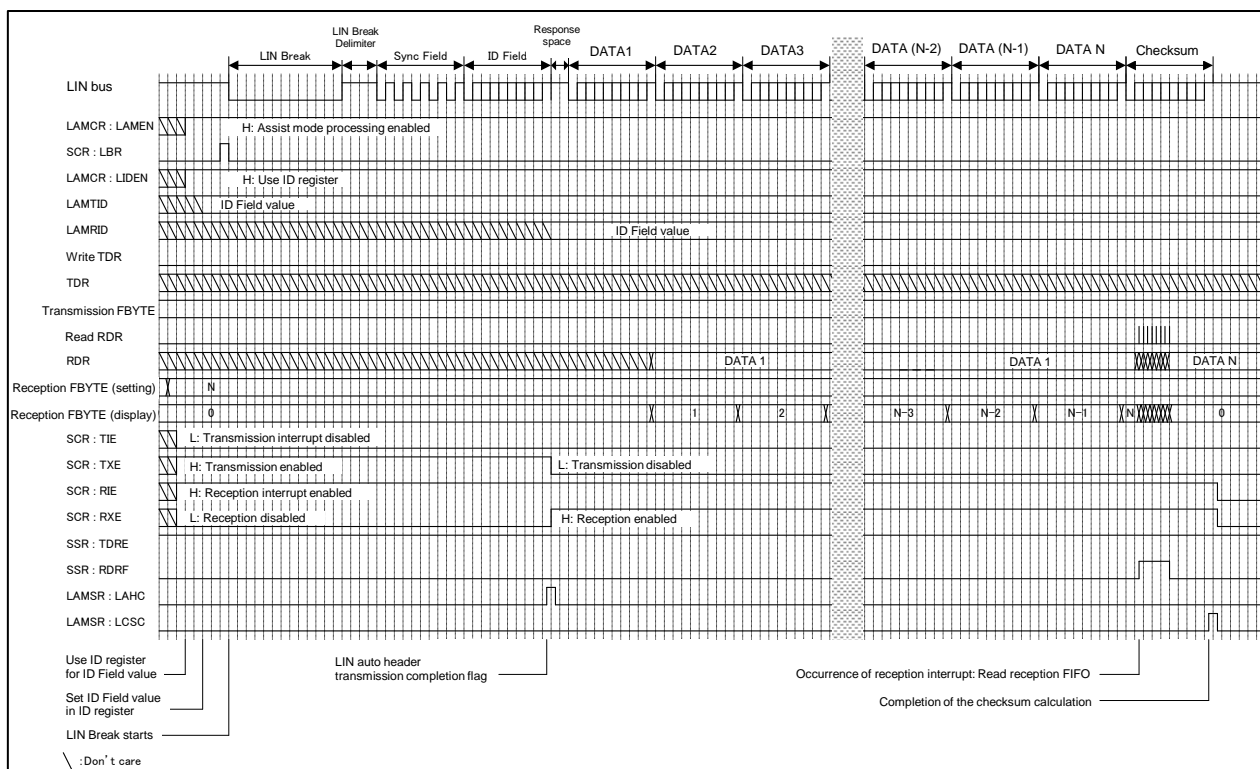
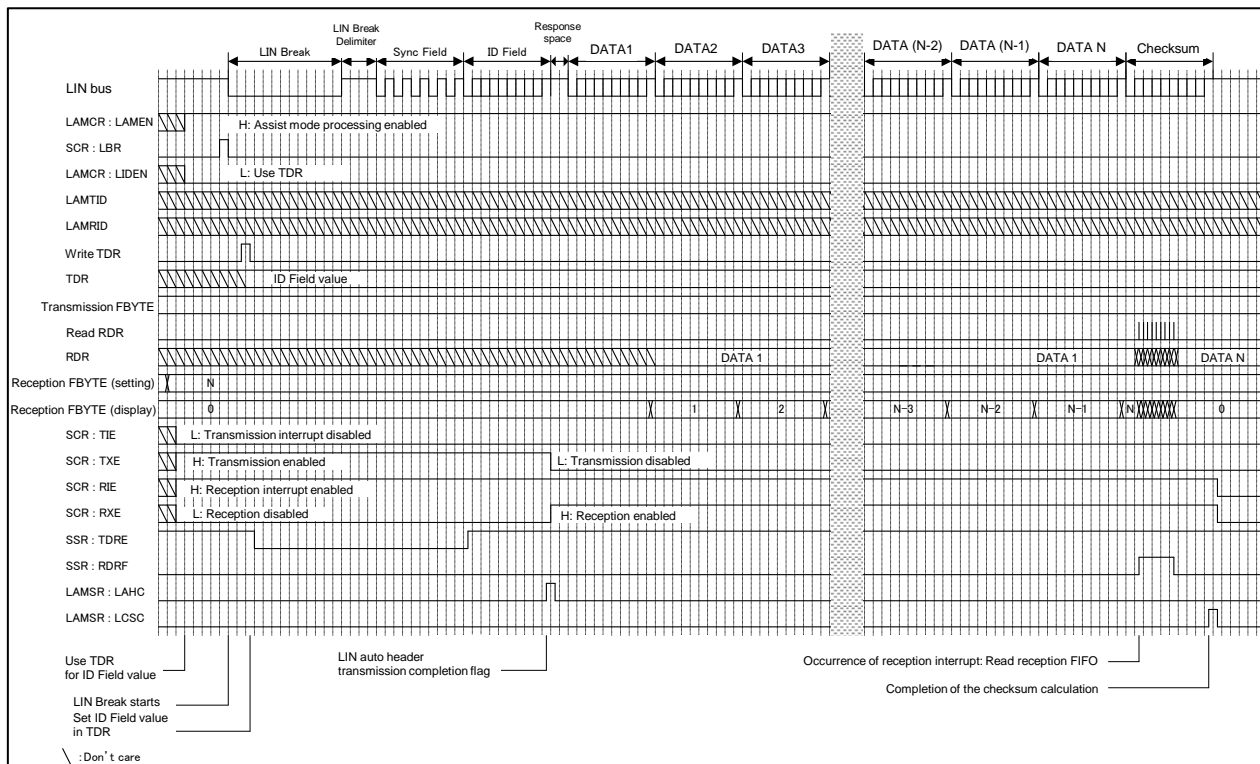


Figure 6-29 LIN Bus Timing (The ID Register is Not Used, the DATA Field is Received, and a FIFO is Used)





## (2) Slave Operation

### a) Auto Header Reception Setting

To process auto header reception using assist mode, set the following:

- To enable operation as a slave, set the SCR:MS bit (master/slave function selection bit) to "1".
- To enable operation using LIN assist mode, set the LAMCR:LAMEN bit (LIN assist mode processing enable bit) to "1".
- Select and set the stop bit length (SMR:SBL and ESCR:ESBL).
- When the assist mode reception ID register (LAMRID) is used for ID reception, set the LAMCR: LIDEN bit (LIN ID register enable bit) to "1".  
When the data reception register (RDR) is used for ID reception, set the LAMCR:LIDEN bit (LIN ID register use enable bit) to "0".
- To enable auto baud rate adjustment, set the SACSR:AUTE bit (auto baud rate adjustment bit) to "1".
- Set the reception enable bit (SCR:RXE) to "0" (reception disabled).

### b) LIN Break Field Reception to ID Field Reception

- (1) When the LIN Break Field is entered, the LIN Break Field is detected (SSR:LBD=1) at the 11th bit. At this time, a status interrupt occurs if the ESCR:LBIE bit is set to "1".

The following operations are performed as part of auto baud rate adjustment.

- (2) When the LIN Interface (v2.1) detects the 1st falling edge of the Sync Field, it initializes the serial timer register (STMR) to 0.
- (3) When the 5th falling edge of the Sync Field is detected, the Sync Field detection flag (SACSR:SFD) is set to "1". At this time, a status interrupt occurs if the SACSR:SFDE bit is set to "1".
- (4) When the falling edge of the 5th Sync Field is detected, the following operation is performed depending on the value of the serial timer register (STMR).
  - When the value of the serial timer register (STMR) is between that of the Sync Field lower limit register (SFLR) and Sync Field upper limit register (SFUR), the value of the serial timer register (STMR) is set in the baud rate generator register (BGR), and the baud rate setting flag (SACSR:BST) is set to "1".
  - When the value of the serial timer register (STMR) is below that of the Sync Field lower limit register (SFLR) or exceeds that of the Sync Field upper limit register (SFUR), the baud rate generator register (BGR) will not be changed and the baud rate setting flag (SACSR:BST) is reset to "0".

Figure 6-30 LIN Break Field to ID Field Reception (When STMR is between SFUR and SFLR)

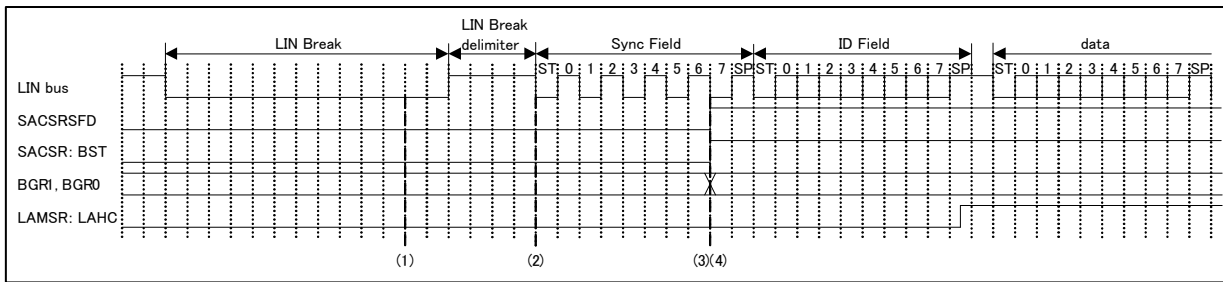
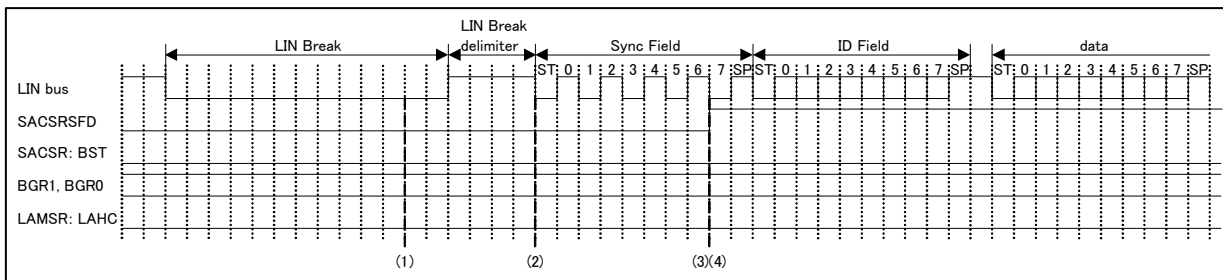


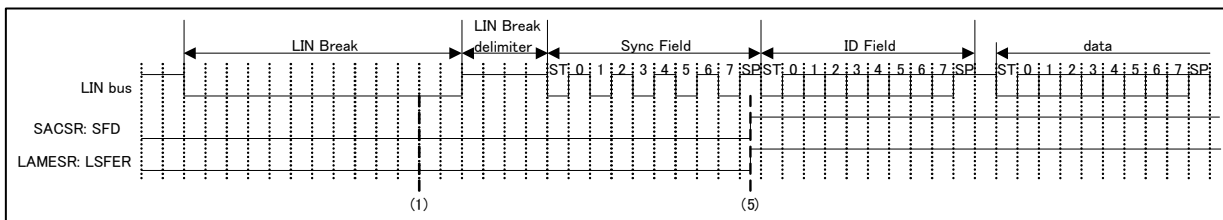
Figure 6-31 LIN Break Field to ID Field Reception (When STMR is Not between SFUR and SFLR)



The following operations are performed when auto baud rate adjustment is not conducted.

- (5) When auto baud rate adjustment is not conducted, set the SACSR:AUTE bit (auto baud rate adjustment bit) to "0". Treat the Sync Field value as data, and verify that the Sync Field value is 0x55. When it is 0x55, the Sync Field detection flag (SACSR:SFD) is set to "1". When it is other than 0x55, the Sync Data error flag (LAMESR:LSFER) is set to "1".

Figure 6-32 LIN Break Field to ID Field Reception (When Auto Baud Rate Adjustment is Not Performed)



The following operations are common both to when auto baud rate adjustment is performed and when it is not performed.

- (6) When auto header reception is completed in LIN assist mode, the LAMSR:LAHC bit (LIN auto header completion flag) is set to "1". If a LIN parity error (LAMESR:LPTE=1) occurs in the ID Field, the LAMSR:LAHC bit (LIN auto header completion flag) is set to "1". So, if the LAMSR:LAHC bit is set to "1", verify that the error is not detected.
- (7) When the ID Field is correctly received, set the checksum type (LAMCR:LCSTYP) and the LIN data length setting bit (LAMCR:LDL3 to LDL0).



Figure 6-33 LIN Break Field to ID Field Reception (When a Parity Error Occurs)

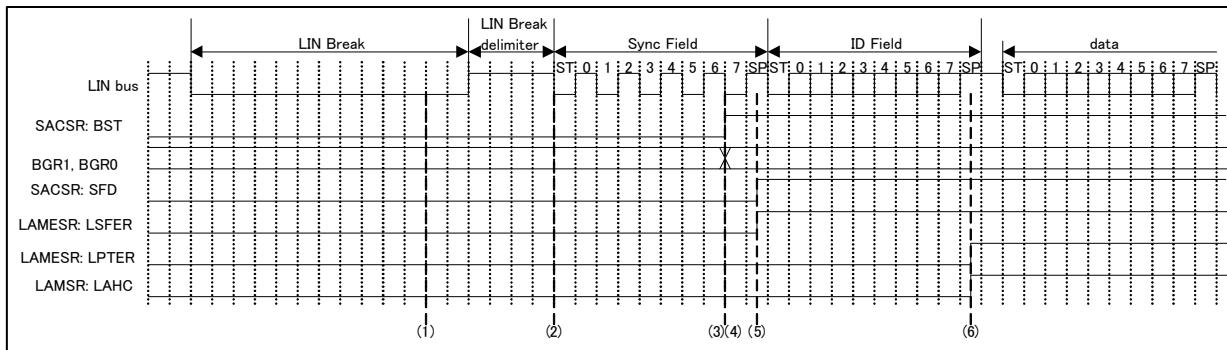
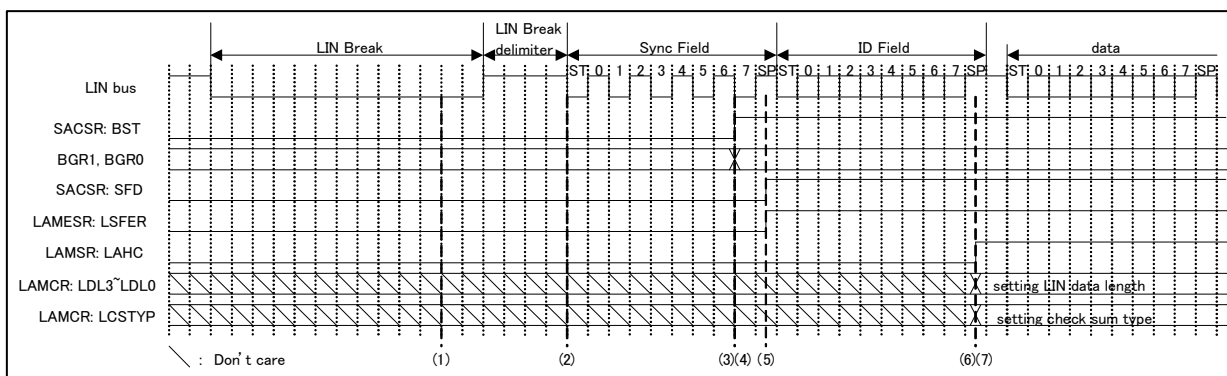


Figure 6-34 LIN Break Field to ID Field Reception (When the LIN Data Length is Set)

**Notes:**

- Disable reception ( $SCR:RXE=0$ ) while the slave header is being received in assist mode.
- The setting of the transmission enable bit ( $SCR:TXE$ ) is ignored while the slave header is received in assist mode.
- While the slave is operating in assist mode, the Sync Field value cannot be stored in the reception data register (RDR).
- During slave operation, suppose that the setting is such that the reception data register (RDR) is used for the reception of the ID Fields ( $LAMCR:LIDEN=0$ ). Then, when an ID Field is received, the received ID value is stored into the reception data register (RDR) and the reception data full flag bit is set ( $SSR:RDRF=1$ ). Check the ID value after the LIN auto header completion flag is set ( $LAMSR:LAHC=1$ ).
- Even when it is determined that there is no need for auto baud rate adjustment ( $SACSR:BST=0$ ), there are cases that the reception of the ID Field continues, and LIN auto header completion flag ( $LAMSR:LAHC=1$ ) and error flag may be set. Therefore, when the LIN auto header completion flag ( $LAMSR:LAHC$ ) is set to "1", verify the baud rate setting flag.

### c) ID Field Reception to DATA Field Transmission and Reception

After the reception of the ID Field, it is possible to select whether the DATA Field is to be transmitted to or received by the master device.

(When the DATA Field is transmitted)

- Using the received ID Field value, set the checksum type (LAMCR:LCSTYP) and LIN data length setting bit (LAMCR:LDL3 to LDL0).
- After receiving the ID Field, write the data into the transmission data register (TDR). In this case, enable transmission (SCR:TXE=1) and the transmission interrupt (SCR:TIE=1).
- Perform the checksum calculation based on the LIN data length setting bit (LAMCR:LDL3 to LDL0), and automatically transmit the checksum after transmitting the final data.
- For the checksum calculation, the calculation method can be selected with the LIN checksum type selection bit (LAMCR:LCSTYP).
- After the reception of checksum data, set the checksum calculation completion flag (LAMSR:LCSC). In this case, when the checksum calculation completion interrupt enable bit is set (LAMIER:LCSCIE=1), a status interrupt occurs.
- After the completion of the checksum calculation (LAMSR:LCSC=1), disable transmission (SCR:TXE=0).

#### Notes:

- While a response is being transmitted during an assist mode operation, disable reception (SCR:RXE=0).
- During assist mode operations, response transmission data (data, checksum) cannot be stored in the reception data register (RDR).
- When the LIN data length is set to 0 bytes (LAMCR:LDL3 to LDL0 =0000) for response transmission, write dummy data to the TDR register to automatically start the checksum calculation and transmit the data. (Any value can be written.) The TDR setting value does not affect the checksum calculation.
- When the LIN data length is set to "0" (LAMCR:LDL3 to LDL0 =0000), the checksum values will be as follows:
  - With the standard checksum setting (LAMCR:LCSTYP=0), the checksum value is 0xFF.
  - With the extended checksum setting (LAMCR:LCSTYP=1), the checksum value is the inverse value of the ID Field.

Figure 6-35 ID Field Reception to DATA Field Transmission (When the ID Register is Used)

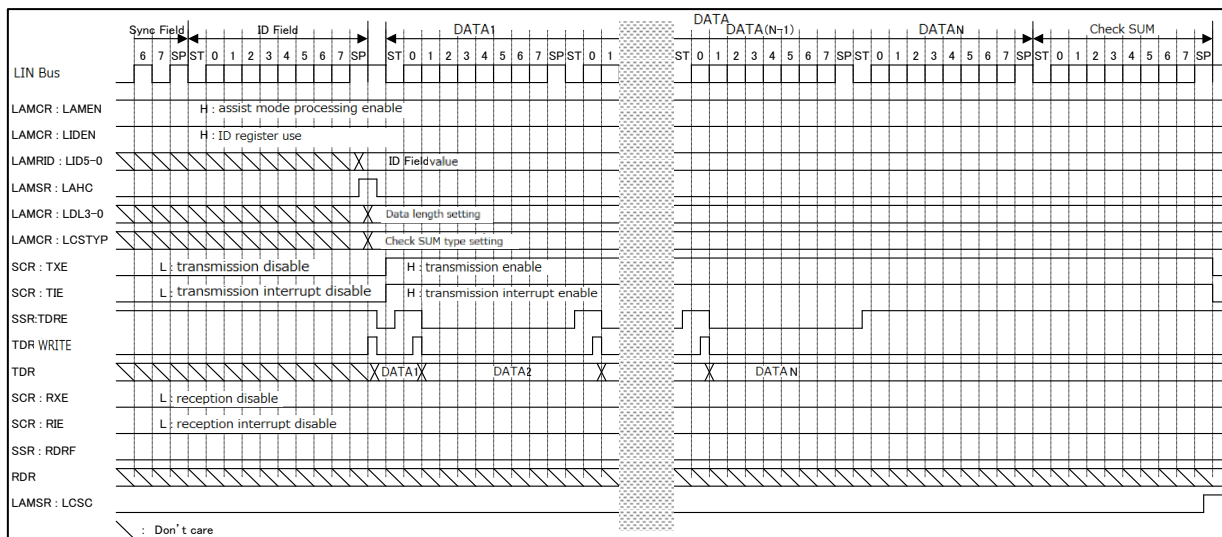
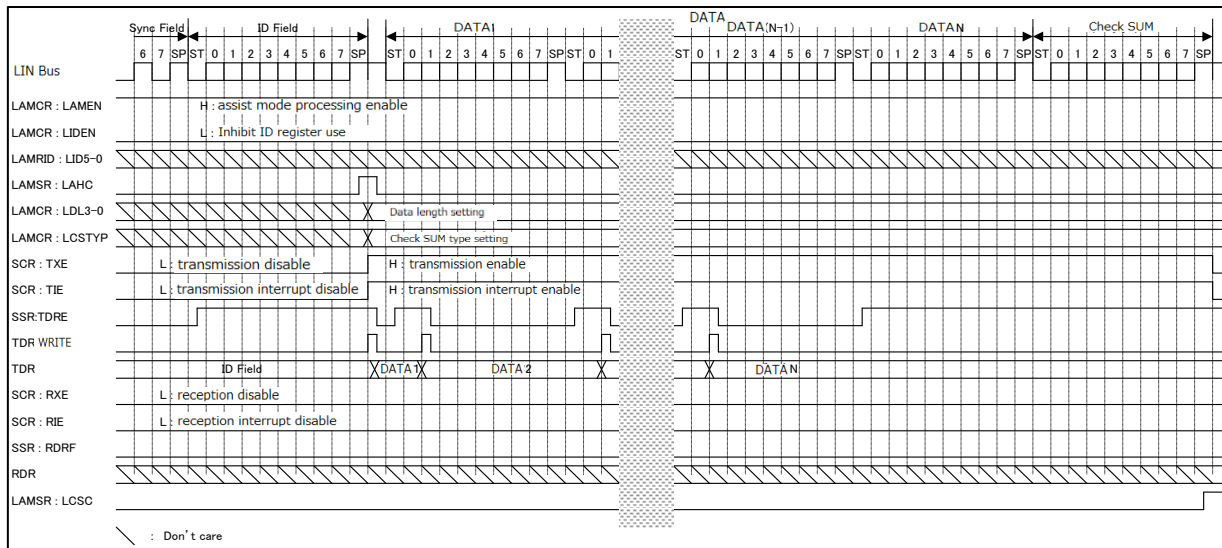






Figure 6-36 ID Field Reception to DATA Field Transmission (When the ID Register is Not Used)



(When the DATA Field is received)

- Using the received ID Field value, set the checksum type (LAMCR:LCSTYP) and LIN data length setting bit (LAMCR:LDL3 to LDL0).
- Enable reception (SCR:RXE=1).
- The checksum calculation is performed automatically. For the checksum calculation, the calculation method can be selected with the LIN checksum type selection bit (LAMCR:LCSTYP).
- For each reception of DATA Field, SSR:RDRF is set to "1". A reception interrupt occurs if the reception interrupt is enabled (SCR:RIE=1).
- When the checksum calculation completion flag is set (LCSC=1), check for a checksum error. In this case, when the checksum calculation completion interrupt is enabled, a status interrupt occurs. When the checksum error interrupt is enabled, a reception interrupt occurs.
- After the completion of checksum reception (LAMSR:LCSC=1), disable reception (SCR:RXE=0).

#### Notes:

- During data reception, the following occurs if a falling edge of serial data is detected at the same time as, or 1 or 2 bus clocks earlier than, the sampling point: The edge becomes invalid and the frame that follows cannot be received normally. When frames are successively output, it is recommended that intervals be provided between the frames.
- The checksum value of response reception during assist mode operations is not stored in the reception data register (RDR).
- When the LIN data length is set to "0" (LAMCR:LDL3 to LDL0 = 0b0000), the checksum values will be as follows:
  - With the standard checksum setting (LAMCR:LCSTYP=0), the checksum value is 0xFF.
  - With the extended checksum setting (LAMCR:LCSTYP=1), the checksum value is the inverse value of the ID Field.
- For slave operation, when the LIN communication speed is 19.2kbps, within about 25μs after the auto header completion flag is set (LAMSR:LAHC=1), set the checksum type (LAMCR:LCSTYP) and the LIN data length (LAMCR:LDL[3:0]), and disable the transmission (SCR:TXE), the transmission interrupt (SCR:TIE), the reception (SCR:RXE) and the reception interrupt (SCR:RIE). (Set to a value less than half of one cycle time for LIN communication.)

Figure 6-37 ID Field Reception to DATA Field Reception (When the ID Register is Used)

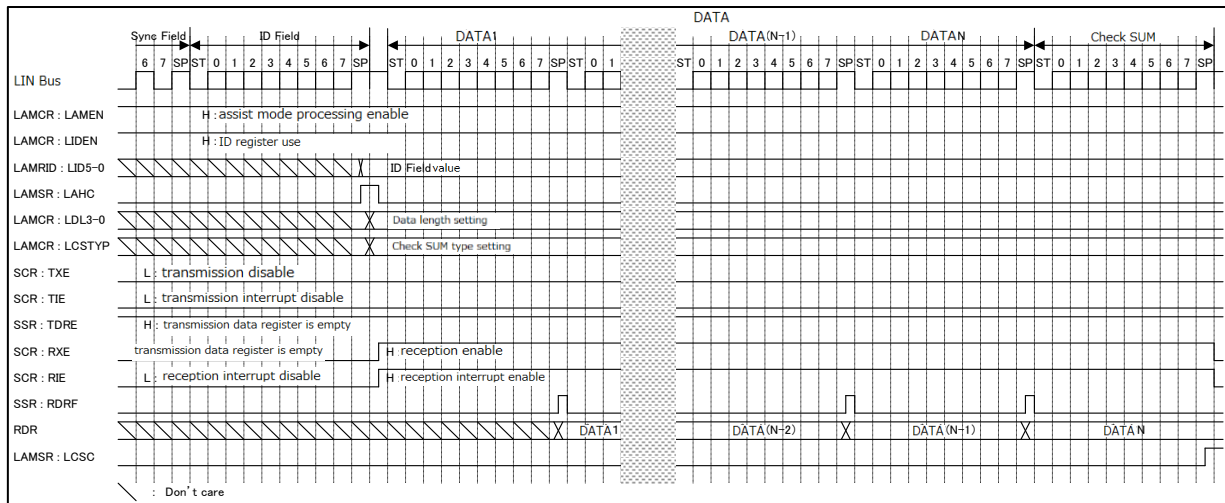
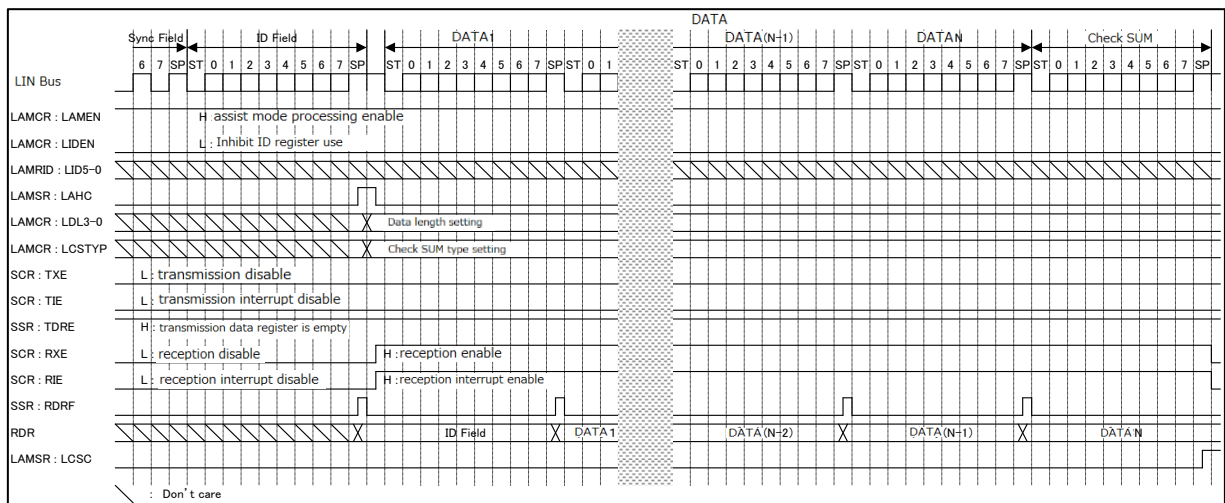


Figure 6-38 ID Field Reception to DATA Field Reception (When the ID Register is Not Used)



#### d) LIN Break Field Reception During Assist Mode Processing

When the LIN Break Field (SSR:LBD=1) was detected from Sync Field to Checksum, the following procedure is required:

- Disable reception (SCR:RXE=0) and transmission (SCR:TXE=0).
- Clear the error flag.
- Abandon any data received before LIN Break Field was detected.
  - When a reception FIFO is used, after disabling the reception FIFO operation (FCR0:FE1=0 or FCR0:FE2=0), reset the reception FIFO (FCR0:FCL1=1 or FCR0:FCL2=1).
  - Then, to clear the reception data register, read the RDR register.
- Abandon any data transmitted before the LIN Break Field is detected.
  - When a transmission FIFO is used, after disabling the transmission FIFO operation (FCR0:FE1=0 or FCR0:FE2=0), reset the transmission FIFO (FCR0:FCL1=1 or FCR0:FCL2=1).
  - Then, execute transmission data register clear (LAMCR:LTDRCL=1) to attain the transmission bus idle status.





e) Slave Operation Timing Chart

Figure 6-39 LIN Bus Timing (When the DATA Field is Transmitted: A FIFO is Not Used, SACSAR:AUTE=1, and an ID Register is Used)

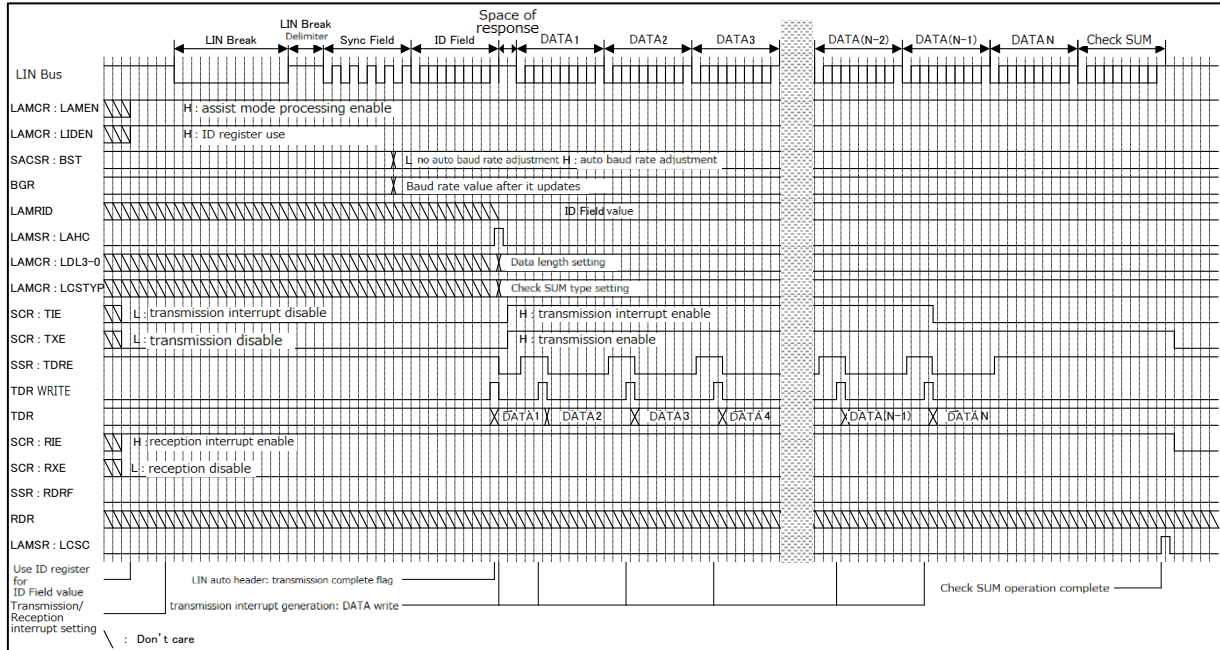
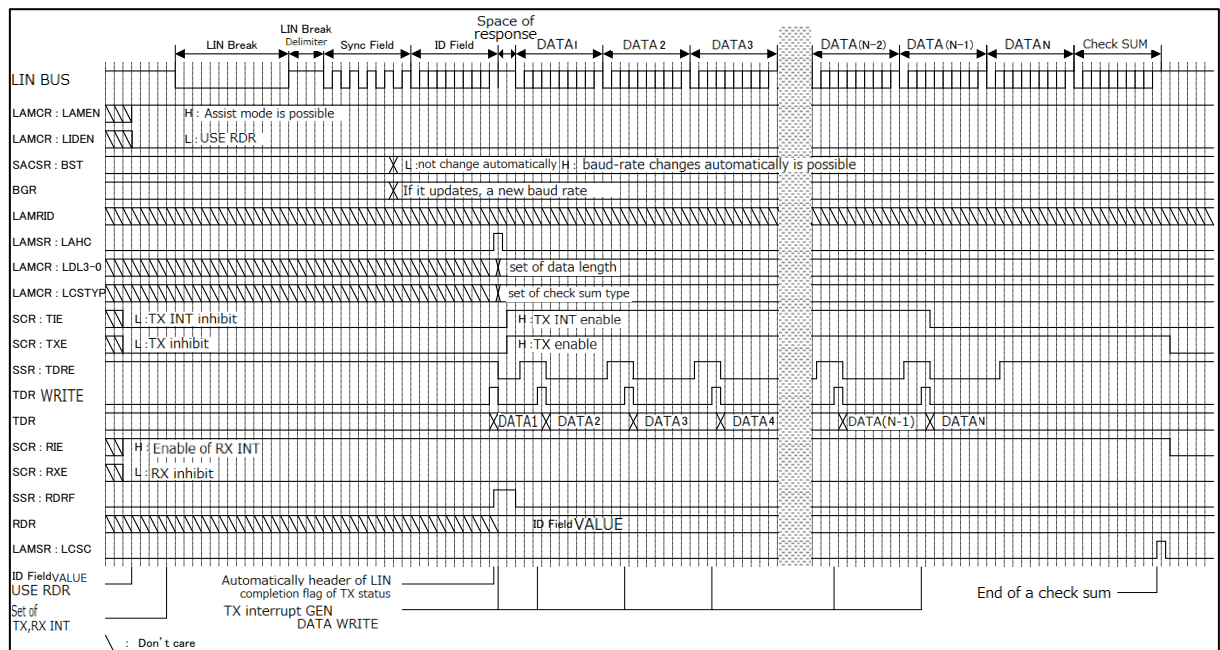
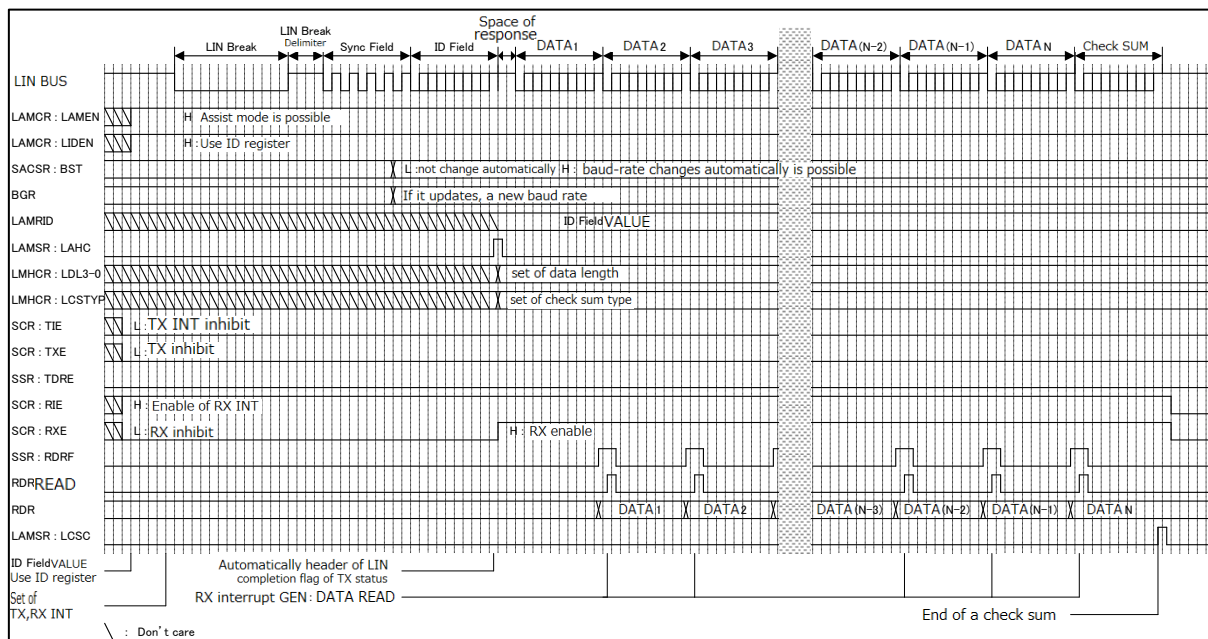


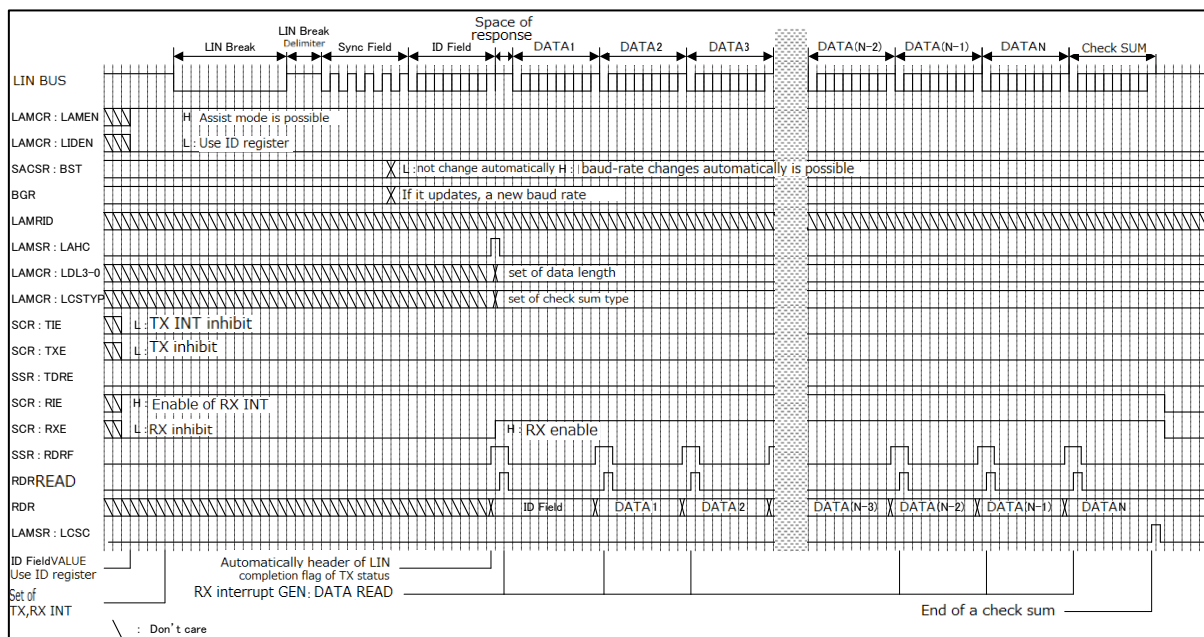
Figure 6-40 LIN Bus Timing (When the DATA Field is Transmitted: A FIFO is Not Used, SACSAR:AUTE=1, and the ID Register is Not Used)



**Figure 6-41 LIN Bus Timing (When the DATA Field is Received: A FIFO is Not Used, SACSAR:AUTE=1, and an ID Register is Used)**

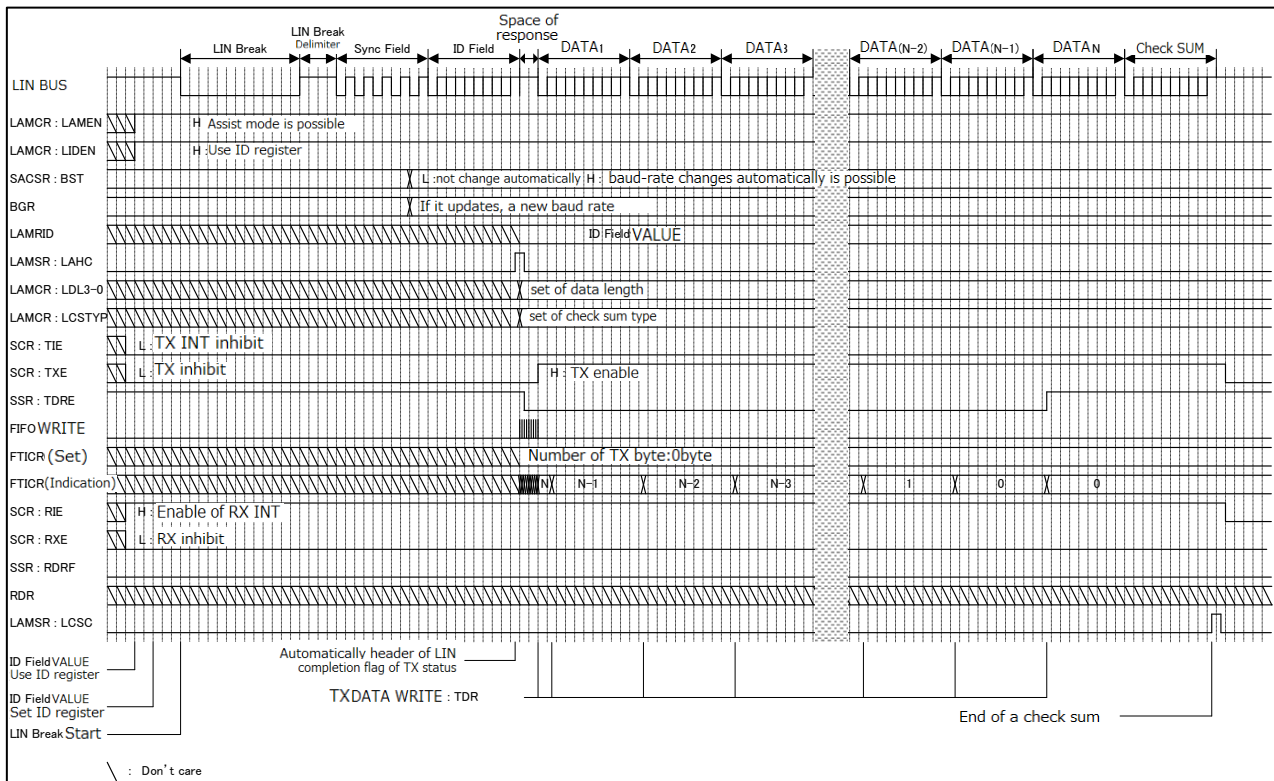


**Figure 6-42 LIN Bus Timing (When the DATA Field is Received: A FIFO is Not Used, SACSAR:AUTE = 1, and an ID Register is Not Used)**

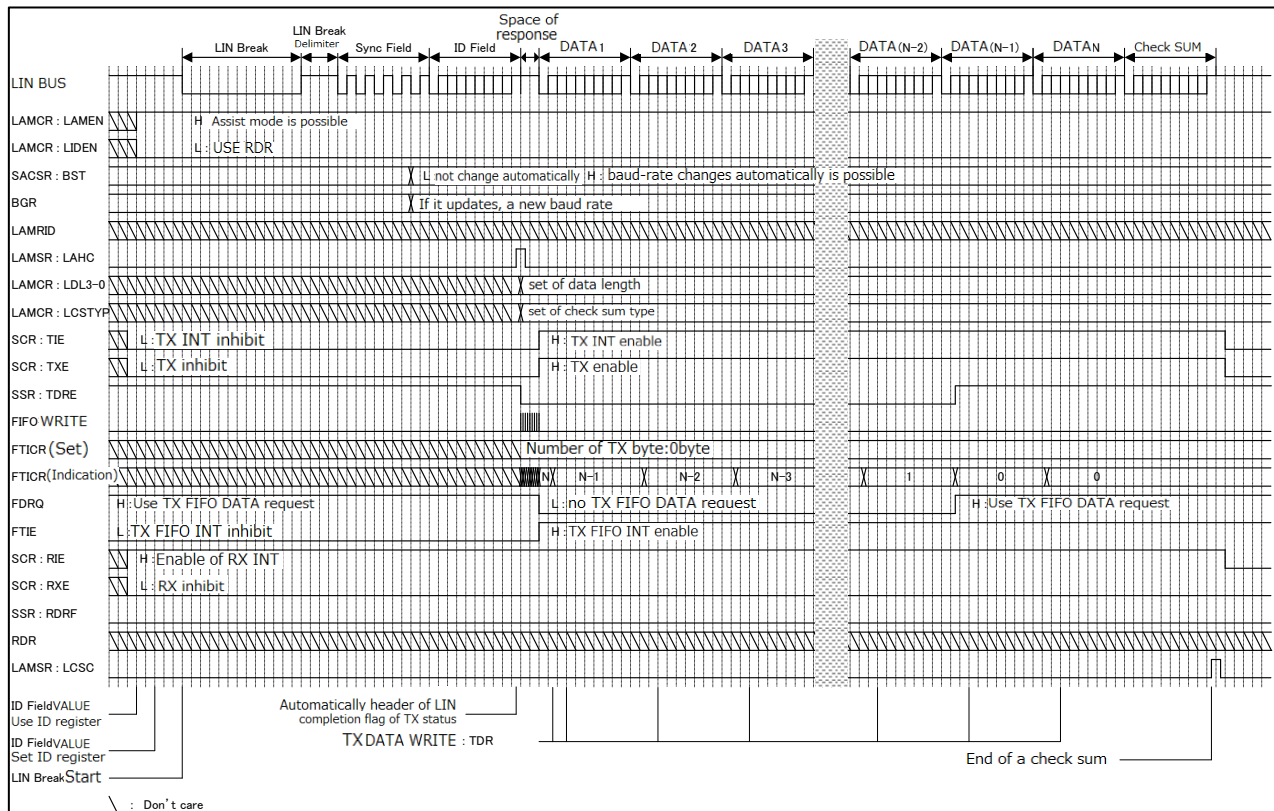




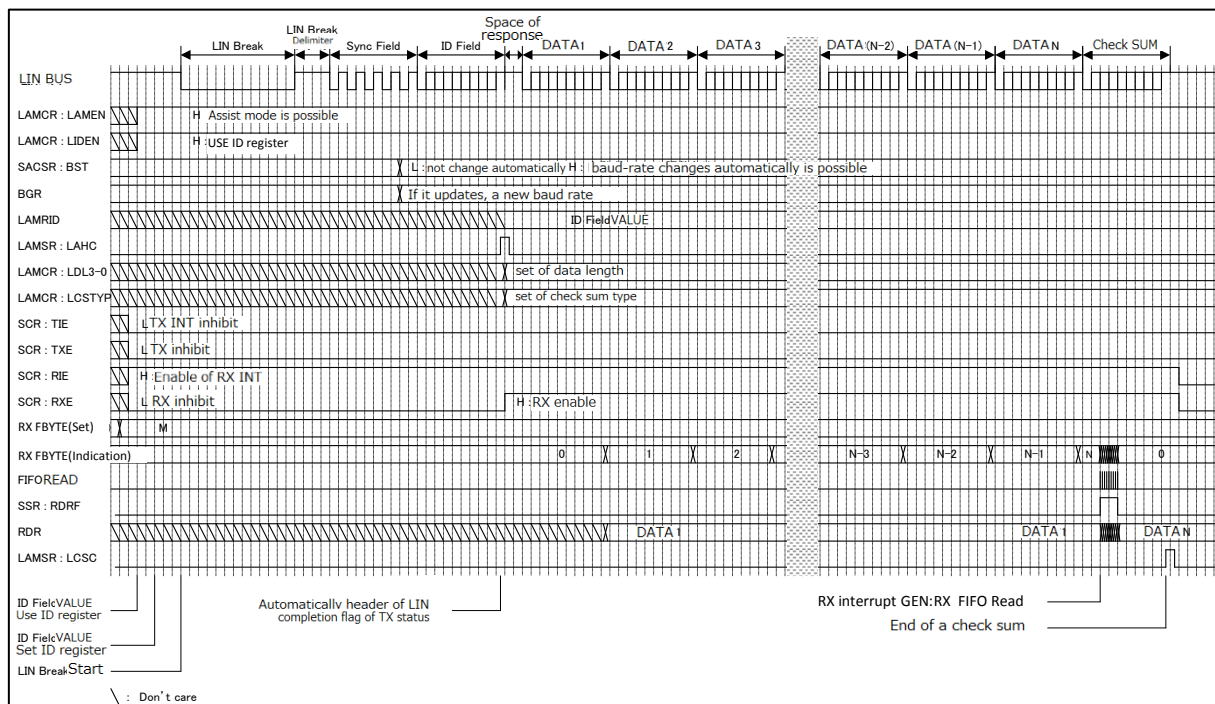
**Figure 6-43 LIN Bus Timing (When the DATA Field is Transmitted: A FIFO is Used, SACSAR:AUTE=1, and the ID Register is Used)**



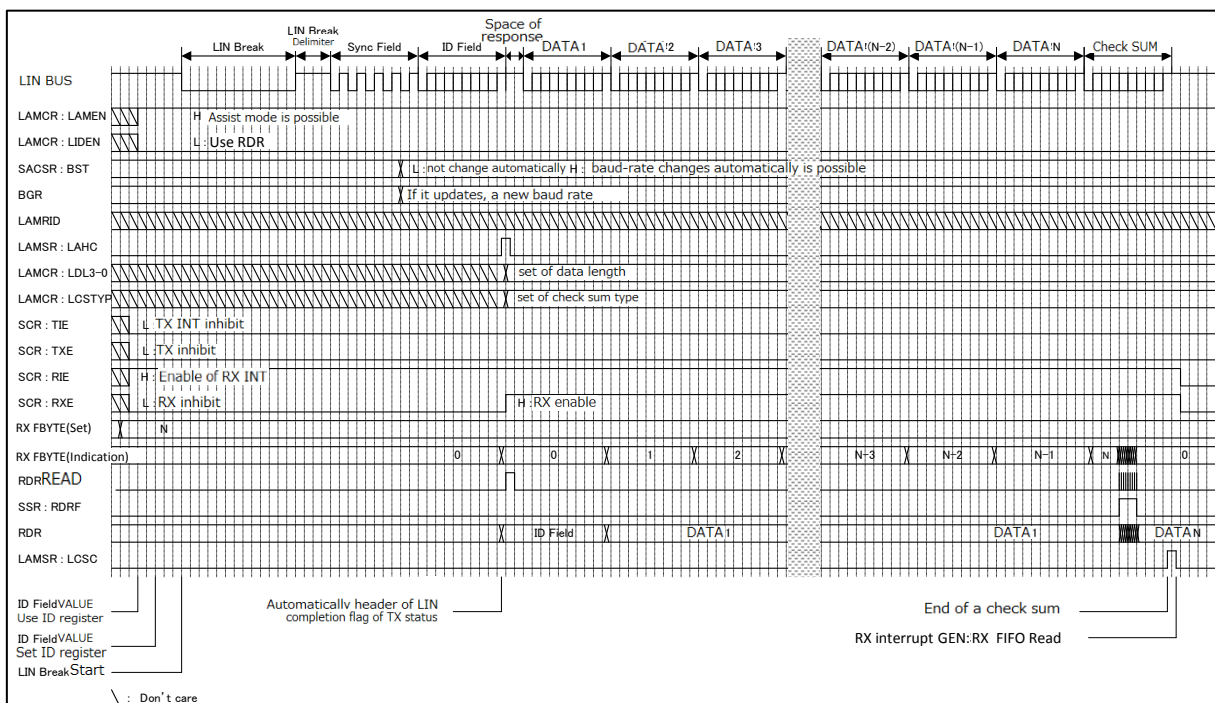
**Figure 6-44 LIN Bus Timing (When the DATA Field is Transmitted: A FIFO is Used, SACSAR:AUTE=1, and the ID Register is Not Used)**



**Figure 6-45 LIN Bus Timing (When the DATA Field is Received: A FIFO is Used, SACS: AUTE=1, and the ID Register is Used)**



**Figure 6-46 LIN Bus Timing (When the DATA Field is Received: A FIFO is Used, SACS: AUTE=1, and the ID Register is Not Used)**



## 7. Setup Procedure and Program Flow for Operation Mode 3 (LIN Communication Mode)

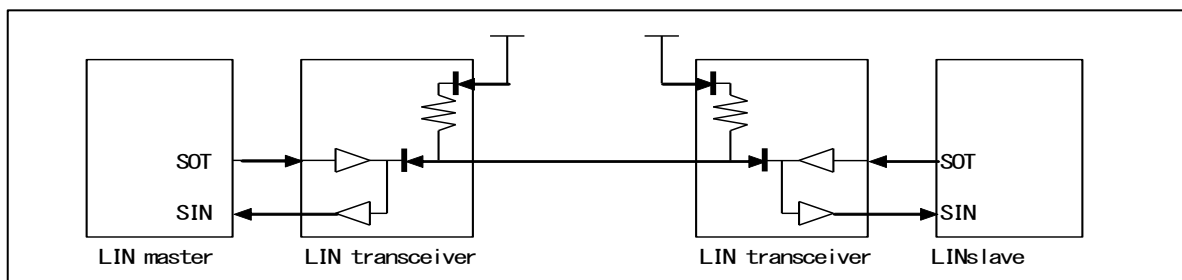
Operation mode 3 (LIN communication mode) can be used for either the LIN master system or the LIN slave system.

### Register Setting

#### Connection between MCUs

Figure 7-1 illustrates the communication system between a LIN master and a LIN slave. The LIN interface (v2.1) can operate as either the LIN master or the LIN slave.

Figure 7-1 Example of LIN Bus System Communication





## 7.1. Manual Mode

This section provides an example of a flow chart for the master or slave side in manual mode.

### Flow Chart Example

#### a) Master Operation

Figure 7-2 Example of LIN Communication Master Mode (A FIFO is Not Used)

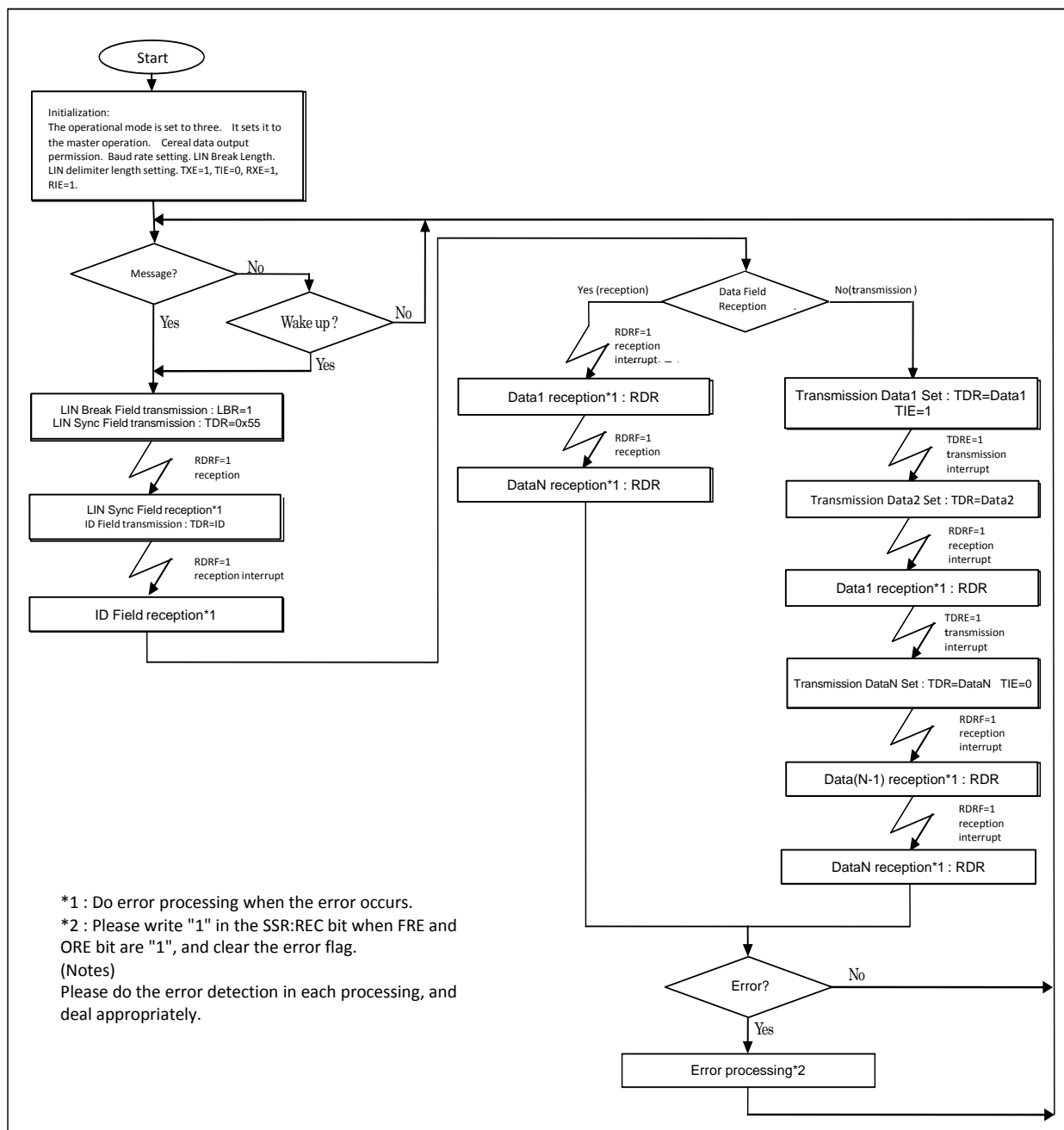
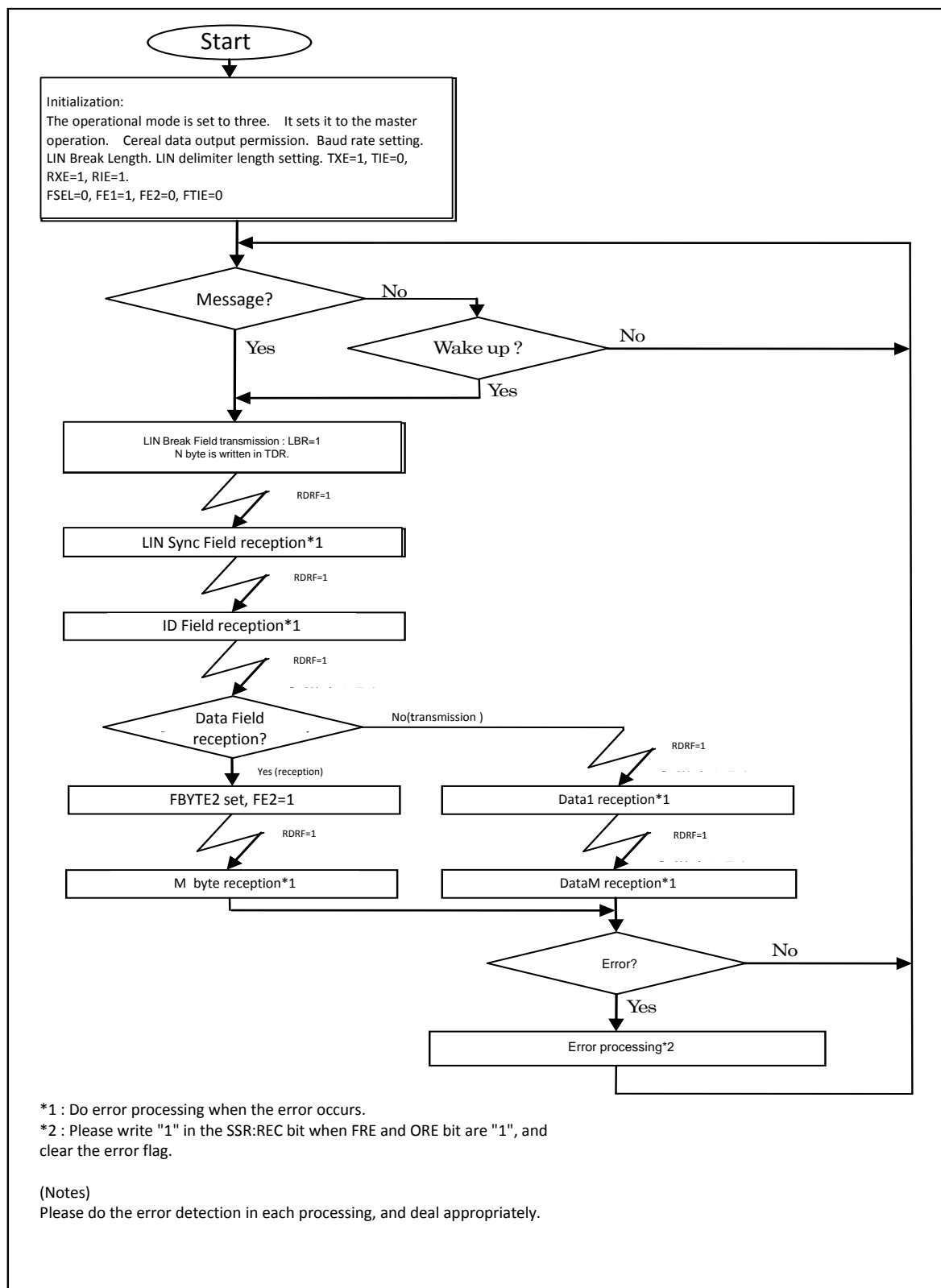


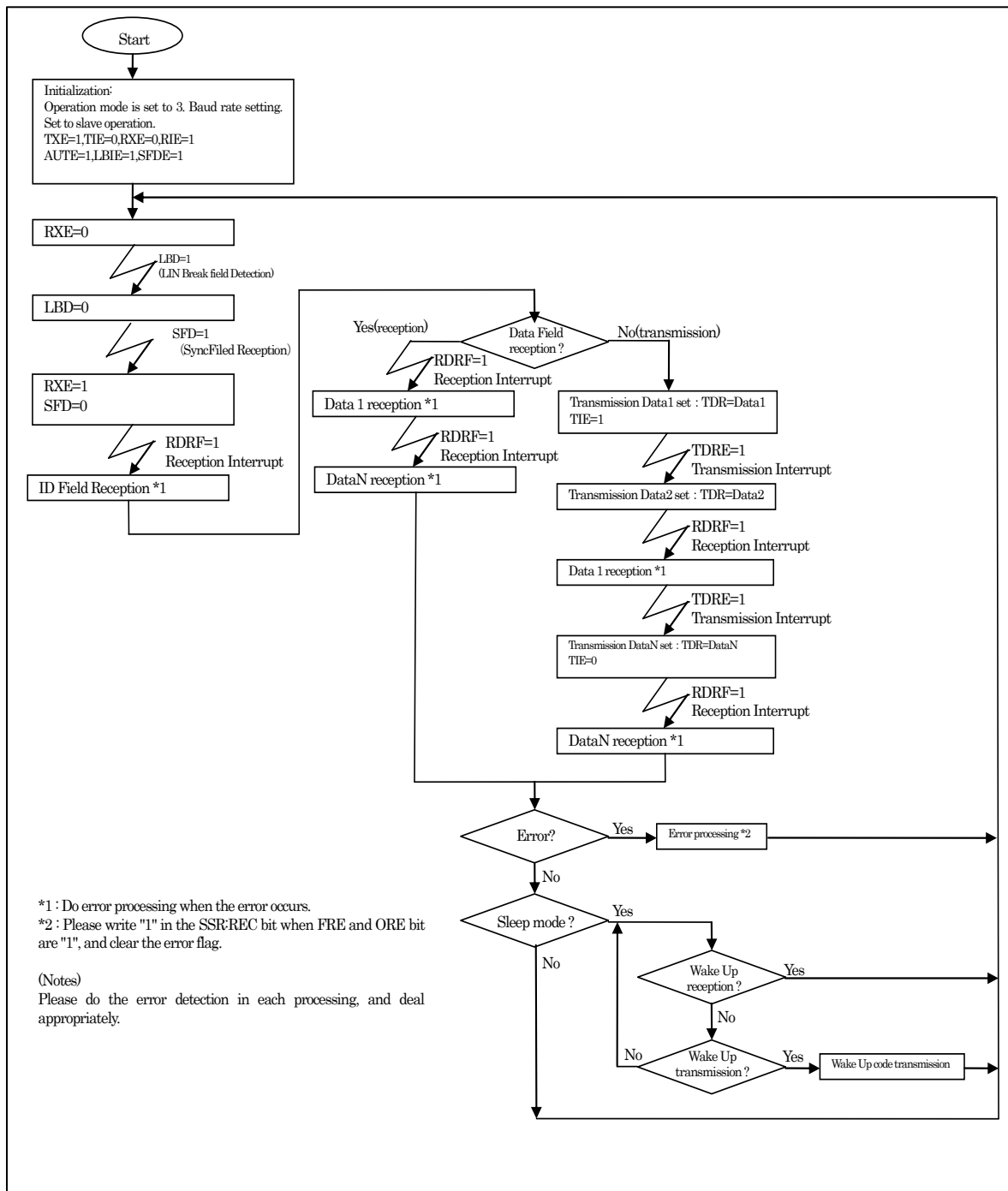


Figure 7-3 Example of LIN Communication Master Mode (A FIFO is Used)



## b) Slave Operation

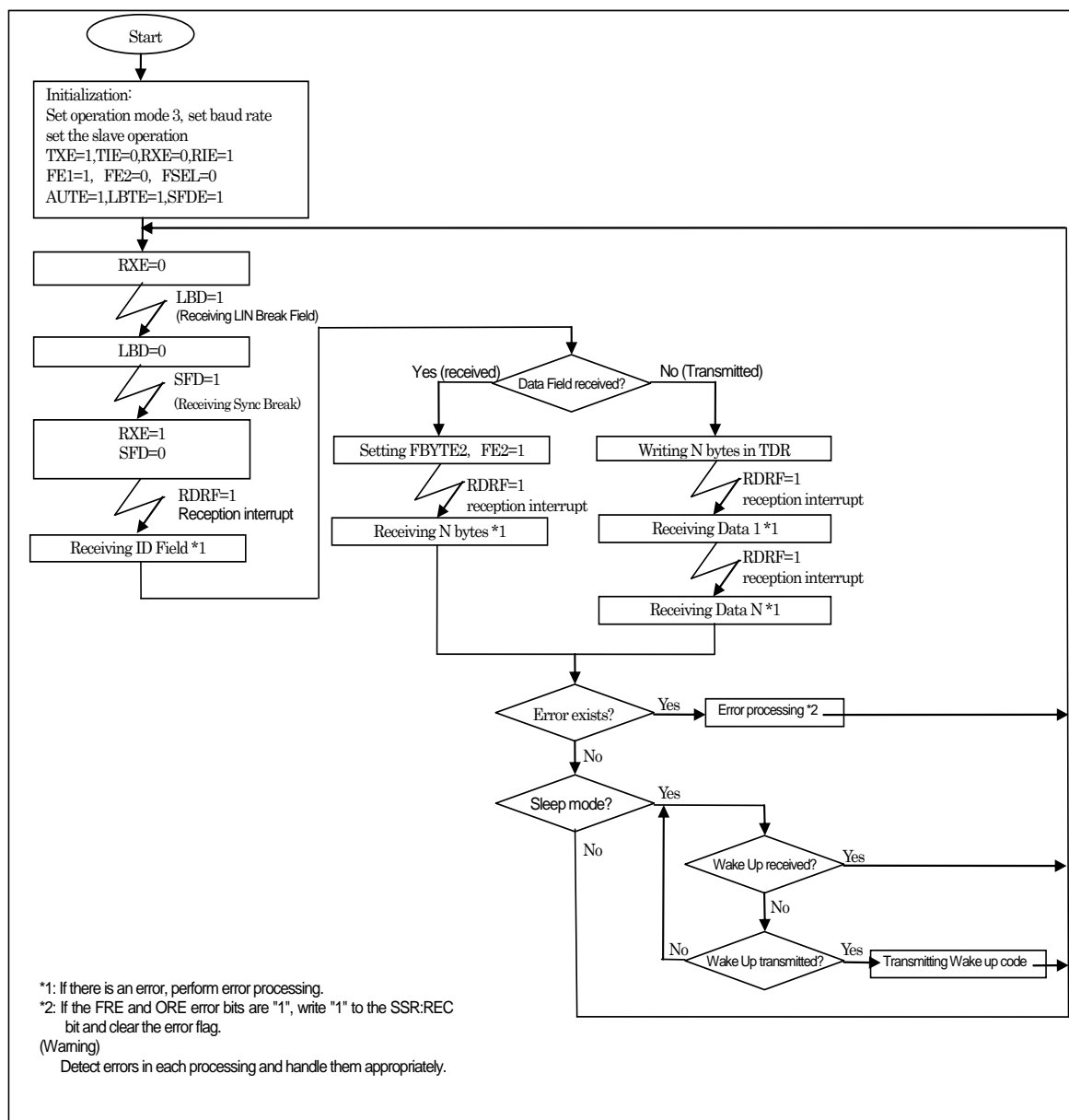
**Figure 7-4 Example of LIN Communication Slave Mode (FIFO is Not Used, Auto Baud Rate Adjustment is Allowed (SACSR:AUTE=1))**







**Figure 7-5 Example of LIN Communication Slave Mode (A FIFO is Used, Auto Baud Rate Adjustment Allowed (SACSR:AUTE=1))**



## 7.2. Assist Mode

This section provides an example of a flow chart of the master or slave side in assist mode.

### Flow Chart Example

The example of the flow chart is described from next page.

## a) Master Operation

Figure 7-6 Example of LIN Communication Master Mode (Assist Mode/a FIFO is Not Used)

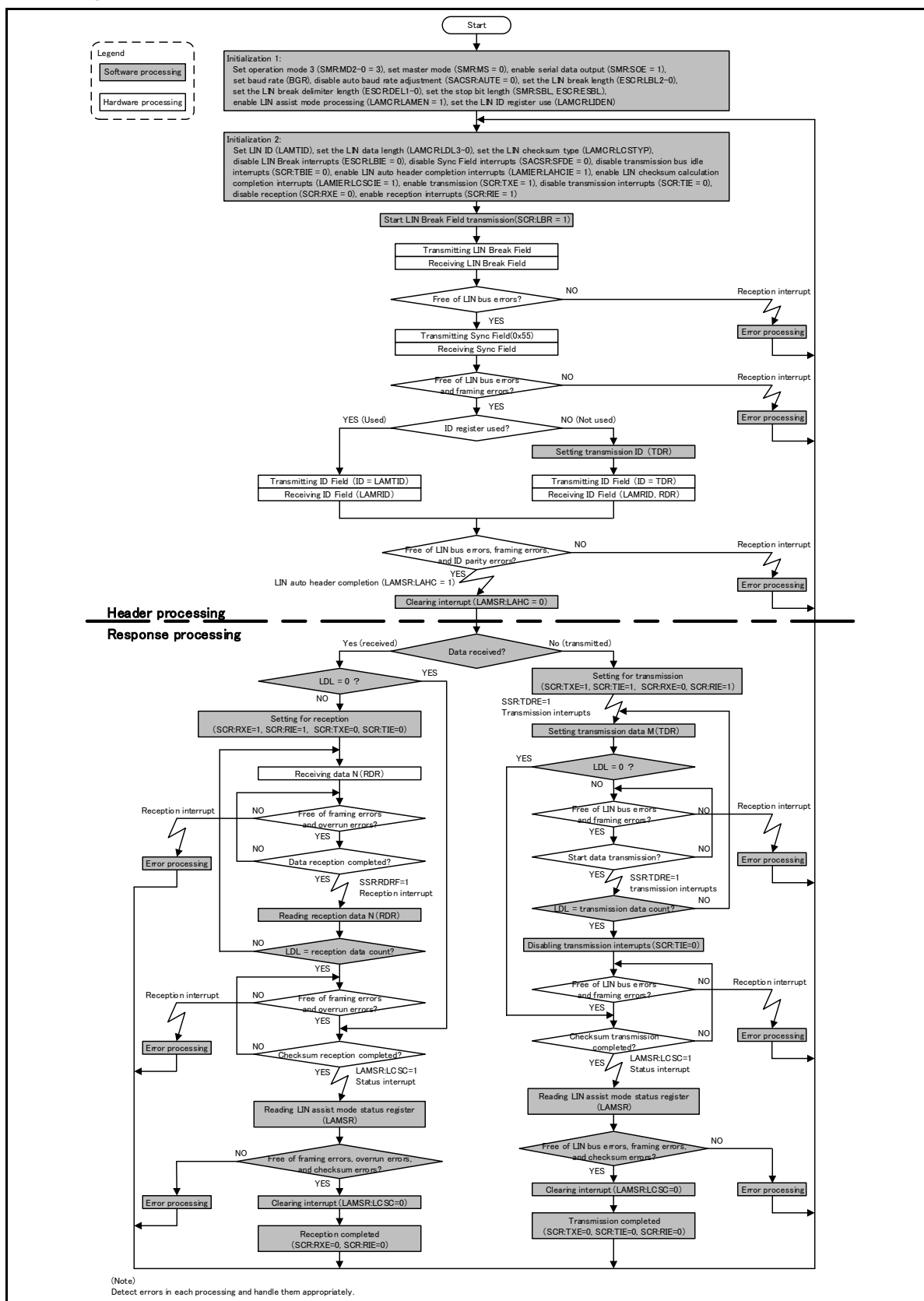
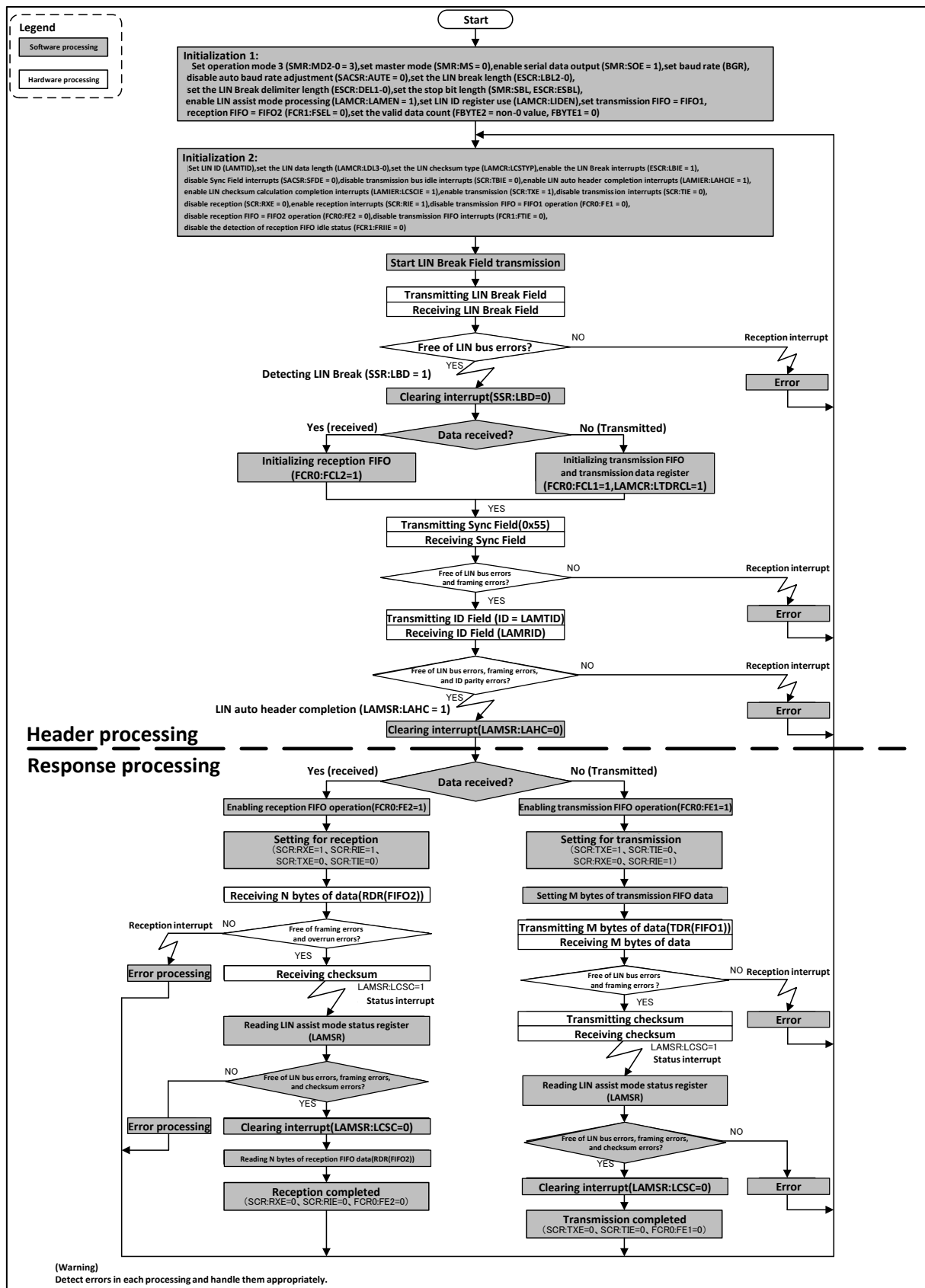


Figure 7-7 Example of LIN Communication Master Mode (Assist Mode/A FIFO is Used)



## b) Slave Operation

Figure 7-8 Example of LIN Communication Slave Mode (Assist Mode/A FIFO is Not Used)

★ Example of LIN communication slave mode (FIFO not used, auto baud rate adjustment enabled)

■ In assist mode, a FIFO is not used, and auto baud rate adjustment is enabled.

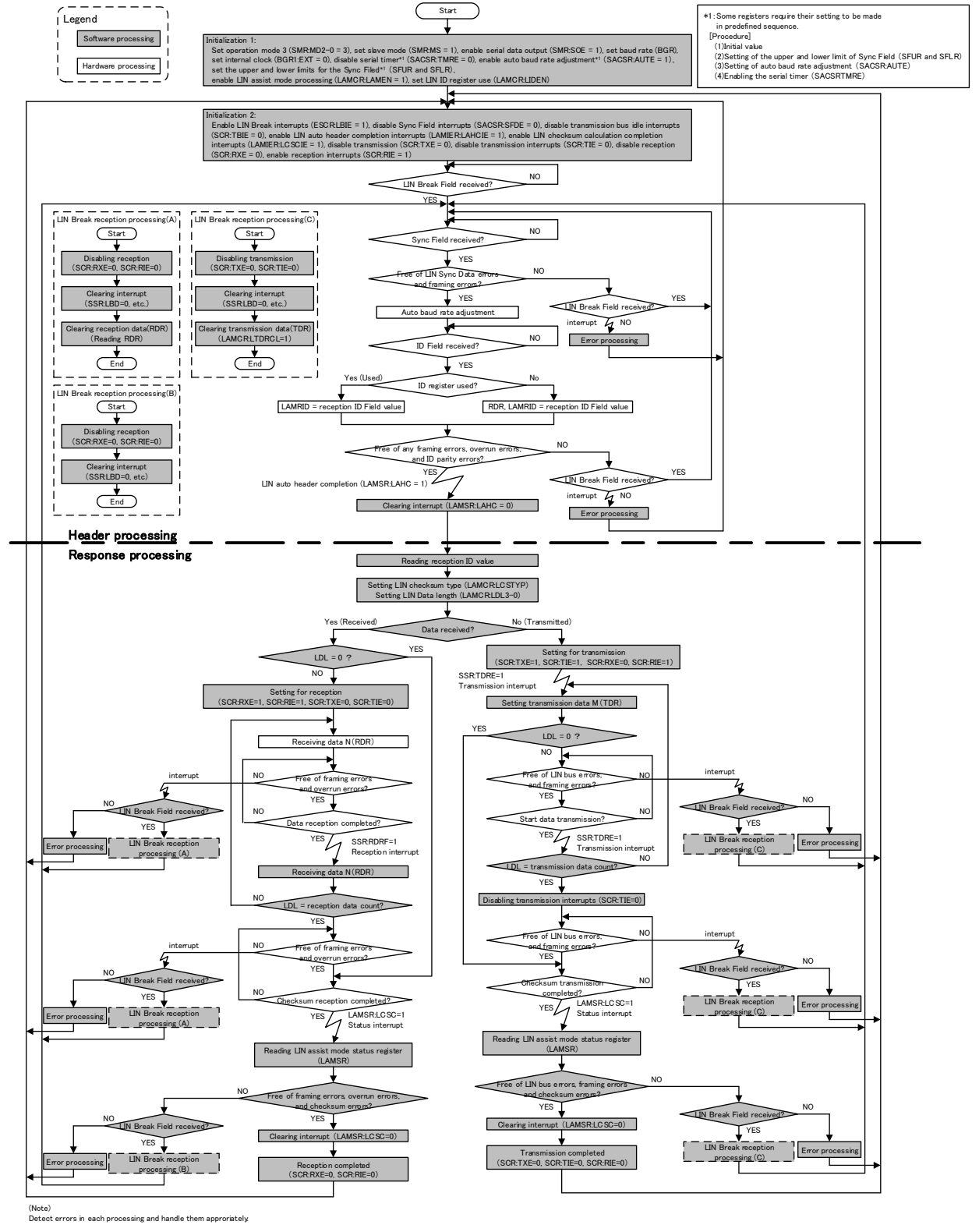
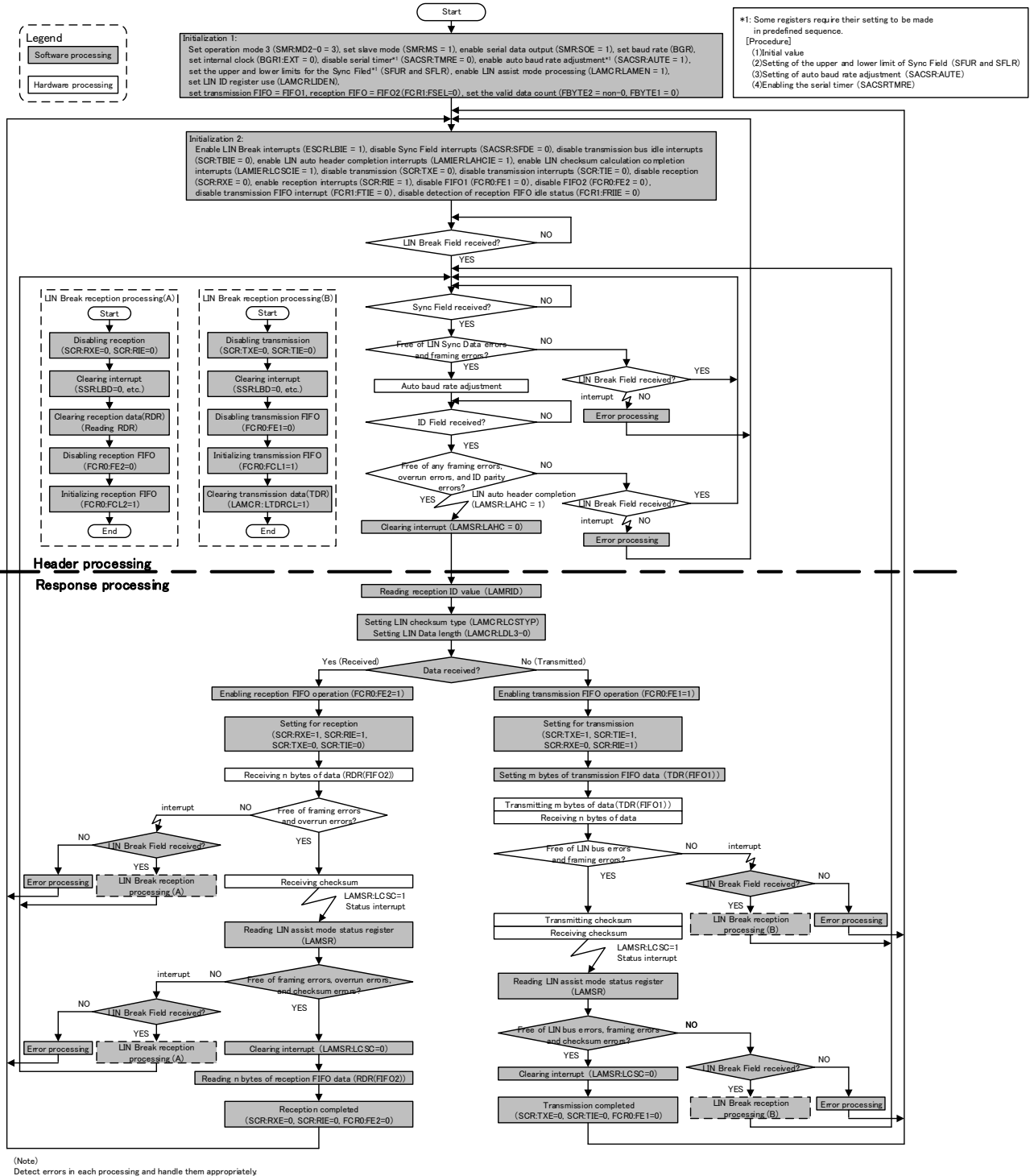


Figure 7-9 Example of LIN Communication Slave Mode (Assist Mode/A FIFO is Used)

- ★ Example of LIN communication slave mode (FIFO used, auto baud rate adjustment enabled)
- In assist mode, a FIFO is used, and auto baud rate adjustment is enabled.





## 8. Registers

This section provides a list of the registers of the LIN interface (v2.1).

All registers are prefixed by "MFSxx\_". xx is the channel number (00, 01, 02, 03, 04).

**Table 8-1 List of LIN Interface (V2.1) Registers**

Abbreviated Register Name	Register Name	See
SCR	Serial Control Register	8.1
SMR	Serial Mode Register	8.2
SSR	Serial Status Register	8.3
ESCR	Extended Communication Control Register	8.4
RDR/TDR	Reception Data Register/Transmission Data Register	8.5
SACSR	Serial Auxiliary Control Status Register	8.6
STMR	Serial Timer Register	8.7
STMCR	Serial Timer Comparison Register	8.8
SFUR	Sync Field Upper Limit Register	8.9
SFLR	Sync Field Lower Limit Register	8.10
BGR1/0	Baud Rate Generator Register 1/0	8.11
LAMSR	LIN Assist Mode Status Register	8.12
LAMCR	LIN Assist Mode Control Register	8.13
LAMIER	LIN Assist Mode Interrupt Enable Register	8.14
LAMTID/LAMRID	LIN Assist Mode Transmission/Reception ID Register	8.15
LAMESR	LIN Assist Mode Error Status Register	8.16
LAMERT	LIN Assist Mode Error Test Register	8.17
FCR1	FIFO Control Register 1	8.18
FCR0	FIFO Control Register 0	8.19
FBYTE	FIFO Byte Register	8.20
FTICR	Transmission FIFO Interrupt Control Register	8.21
SCRC	Serial Control Clear Register	8.22
SMRC	Serial Mode Clear Register	8.23
SSRC	Serial Status Clear Register	8.24
ESCRC	Extended Communication Control Clear Register	8.25
SACSRC	Serial Auxiliary Control Status Clear Register	8.26
LAMSRC	LIN Assist Mode Status Clear Register	8.27
LAMCRC	LIN Assist Mode Control Clear Register	8.28
LAMIERC	LIN Assist Mode Interrupt Enable Clear Register	8.29
LAMESRC	LIN Assist Mode Error Status Clear Register	8.30

Abbreviated Register Name	Register Name	See
FCR1C	FIFO Control Clear Register 1	8.31
FCR0C	FIFO Control Clear Register 0	8.32
SCRS	Serial Control Set Register	8.33
SMRS	Serial Mode Set Register	8.34
SSRS	Serial Status Set Register	8.35
ESCRS	Extended Communication Control Set Register	8.36
SACRS	Serial Auxiliary Control Status Set Register	8.37
LAMCRS	LIN Assist Mode Control Set Register	8.38
LAMIERS	LIN Assist Mode Interrupt Enable Set Register	8.39
FCR1S	FIFO Control Set Register 1	8.40
FCR0S	FIFO Control Set Register 0	8.41





## 8.1. Serial Control Register (SCR)

The serial control register (SCR) is used to enable/disable transmission/reception interrupts, enable/disable transmission idle interrupts, and enable/disable transmission/reception operations. This register also includes the settings for LIN Break Field generation and LIN interface (v2.1) reset.

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	UPCL	MS	LBR	RIE	TIE	TBIE	RXE	TXE
ACCESS_TYPE	R0,W	R/W	R0,W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

\* Lower byte [bit7:0] of this register is serial mode register (SMR).

### [bit15] UPCL: Programmable clear bit

This bit initializes the internal status of the LIN interface (v2.1).

This bit is set when the SCRS:UPCLS bit in the set register is set to "1".

When "1" is set:

- The LIN interface (v2.1) is directly reset (software reset). However, the register settings are retained. At this time, transmission and reception communications are immediately disconnected.
- The baud rate generator reloads the setting values of the BGR1/0 registers and then restarts.
- All transmission/reception and status interrupt factors (SSR:TDRE, TBI, RDRF, FRE, ORE, LBD, SACSR:TINT, SFD, LAMSR:LER, SER, RDRF, TDRE, TBI, LCSC, LAHC, LAMESR:LCSE, LPTER, LSFER, and LBSER) are initialized.
- The baud rate setting flag (SACSR:BST) is initialized.

When "0" is set:

There is no effect.

Value	Description	
	Write	Read
0	No effect	"0" is always read.
1	Programmable clear	

#### Notes:

- Execute programmable clear after setting interrupt prohibition.
- When a FIFO is used, prohibit the FIFO (FCR0:FE2, FE1=0) before executing programmable clear.
- Programmable clear does not clear the transmission/reception FIFOs.
- Executing programmable clear (SCR:UPCL=1) does not clear the value of the serial timer register (STMR).

### [bit14] MS: Master/Slave function select bit

This bit selects either master or slave mode.

- This bit is reset when the SCRC:MSC bit in the clear register is set to "1".
- This bit is set when the SCRS:MSS bit in the set register is set to "1".

Value	Description
0	Master mode
1	Slave mode

**[bit13] LBR: LIN Break Field setting bit (effective only for master operation)**

- In LIN manual mode operation (LAMCR:LAMEN=0):
  - Setting this bit to "1" generates the LIN Break Field and LIN Break delimiter in the lengths specified by the ESCR:LBL1/0 and ESCR:DEL1/0 bits.
- In LIN assist mode operation (LAMCR:LAMEN=1):
  - Setting this bit to "1" generates the LIN Break Field and LIN Break delimiter in the lengths specified by the ESCR:LBL2/1/0 and ESCR:DEL1/0 bits, and then transmits the Sync Field and the ID Field.
- This bit is set when the SCRS:LBRS bit in the set register is set to "1".

When writing:

- Writing "0": No effect
- Writing "1": In LIN manual mode operation (LAMCR: LAMEN=0), a LIN Break Field is generated.  
In LIN assist mode operation (LAMCR: LAMEN=1), a LIN Break Field is generated, and a Sync Field and an ID Field are transmitted.

When reading:

"0" is always read.

Value	Description	
	Write	Read
0	No effect	"0" always read
1	In LIN manual mode, a LIN Break Field is generated. In LIN assist mode, the LIN Break Field to ID Field are transmitted.	

**Notes:**

- This bit is effective only for master operation (MS=0).
- This bit must not be set to "1" when a header is being transmitted or when a response is being transmitted or received.
- If Programmable clear (writing UPCL="1") and LBR Setting (writing LBR="1") are attempted simultaneously, the programmable clear takes precedence.

**[bit12] RIE: Reception interrupt enable bit**

- This bit enables/disables reception interrupt request output to the CPU.
- A reception interrupt request is issued when the RIE bit and the reception data flag bit (SSR:RDRF) are "1" or when one of the error flag bits (SSR:FRE, ORE) is "1".
- This bit is reset when the SCRC:RIEC bit in the clear register is set to "1".
- This bit is set when the SCRS:RIES bit in the set register is set to "1".

Value	Description
0	Disable reception interrupts.
1	Enable reception interrupts.

**[bit11] TIE: Transmission interrupt enable bit**

- This bit enables/disables transmission interrupt request output to the CPU.
- A transmission interrupt request is issued when the TIE bit and the SSR:TDRE bit are set to "1".
- This bit is reset when the SCRC:TIEC bit in the clear register is set to "1".
- This bit is set when the SCRS:TIES bit in the set register is set to "1".

Value	Description
0	Disable transmission interrupts.
1	Enable transmission interrupts.

**[bit10] TBIE: Transmission bus idle interrupt enable bit**

- This bit enables/disables transmission bus idle interrupt request output to the CPU.
- A transmission bus idle interrupt request is output when the TBIE bit and the SSR:TBI bit are "1".
- This bit is reset when the SCRC:TBIEC bit in the clear register is set to "1".
- This bit is set when the SCRS:TBIES bit in the set register is set to "1".

Value	Description
0	Disable transmission bus idle interrupts.
1	Enable transmission bus idle interrupts.

**[bit9] RXE: Reception operation enable bit**

This bit controls enable/disable of the receiving operation of the LIN interface (v2.1).

- This bit is reset when the SCRC:RXEC bit in the clear register is set to "1".
- This bit is set when the SCRS:RXES bit in the set register is set to "1".
- When set to "0": Data frame reception operation is disabled.
- When set to "1": Data frame reception operation is enabled.

Value	Description
0	Disable reception.
1	Enable reception.

**Notes:**

- Even if reception operation is enabled (RXE=1), reception operation does not start unless a falling edge of a start bit is input.
- In master operation, even if reception operation is enabled (RXE=1), data is not received while LIN Break Field is being transmitted.
- In manual mode (LAMCR:LAMEN=0), disabling reception operation (RXE=0) during reception immediately stops reception operation.
- In assist mode (LAMCR:LAMEN=1), disable reception (RXE=0) during transmission or reception of a header and during response transmission.
- In assist mode (LAMCR:LAMEN=1), even if reception operation is disabled (RXE=0) during header reception, header reception operation does not stop. To stop the reception, disable the reception (RXE=0) and set manual mode (LAMCR:LAMEN=0).
- In assist mode (LAMCR:LAMEN=1), disabling reception operation (RXE=0) during reception of a response immediately stops reception operation.
- Framing error is detected (SSR:FRE=1) when a LIN Break Field is received and reception is enabled (RXE=1).

**[bit8] TXE: Transmission operation enable bit**

This bit enables/disables transmission by the LIN interface (v2.1).

- This bit is reset when the SCRC:TXEC bit in the clear register is set to "1".
- This bit is set when the SCRS:TXES bit in the set register is set to "1".
- When set to "0": Data frame transmission operation is disabled.
- When set to "1": Data frame transmission operation is enabled.

Value	Description
0	Disable transmission.
1	Enable transmission.

**Notes:**

- In manual mode (LAMCR:LAMEN=0), disabling transmission operation (TXE=0) during transmission immediately stops transmission operation.
- In assist mode (LAMCR:LAMEN=1), even if transmission operation is disabled (TXE=0) during header transmission, header transmission operation does not stop. To stop the transmission, disable the transmission (TXE=0) and set manual mode (LAMCR:LAMEN=0).
- In assist mode (LAMCR:LAMEN=1), disabling transmission operation (TXE=0) during transmission of a response immediately stops transmission operation.



## 8.2. Serial Mode Register (SMR)

The serial mode register (SMR) is responsible for the following: 1. Operation mode setting, 2. Selection of the transfer direction, the data length, and the stop bit length, and 3. Setting of enabling/disabling of output on the serial data and clock pins.

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	MD2	MD1	MD0	Reserved	SBL	Reserved		SOE
ACCESS_TYPE	R/W	R/W	R/W	R/W0	R/W	R0,W0		R/W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	00		0

### [bit7:5] MD2 to 0: Operation mode setting bits

These bits set the operation mode.

"0b000": Set operation mode 0 (asynchronous normal mode).

"0b001": Set operation mode 1 (asynchronous multi-processor mode).

"0b010": Set operation mode 2 (clock synchronous mode).

"0b011": Set operation mode 3 (LIN communication mode).

This chapter describes the register and its operations in operation mode 3 (LIN communication mode).

Value			Description
MD2	MD1	MD0	
0	0	0	Operation mode 0 (asynchronous normal mode)
0	0	1	Operation mode 1 (asynchronous multi-processor mode)
0	1	0	Operation mode 2 (clock synchronous mode)
0	1	1	Operation mode 3 (LIN communication mode)
Other than above			Setting is prohibited

- This section describes the register and its operations in operation mode 3.
- For operation mode 0/1, see "CHAPTER: UART". For operation mode 2, see "CHAPTER: CSIO".

### Notes:

- To switch the operation mode, execute programmable clear (SCR:UPCL=1) and then switch the operation mode.
- Set the registers only after setting an operation mode.

### [bit4] Reserved: Reserved bit

**[bit3] SBL: Stop bit length selection bit**

This bit sets the length of the stop bits (frame end mark of the transmission data).

- This bit is reset when the SMRC:SBLC bit in the clear register is set to "1".
- This bit is set when the SMRS:SBLS bit in the set register is set to "1".
- Settings where SBL is "0" and ESCR:ESBL is "0": The stop bit is set to 1 bit.
- Settings where SBL is "1" and ESCR:ESBL is "0": The stop bit is set to 2 bits.
- Settings where SBL is "0" and ESCR:ESBL is "1": The stop bit is set to 3 bits.
- Settings where SBL is "1" and ESCR:ESBL is "1": The stop bit is set to 4 bits.

Value	Description	
0	ESCR:ESBL=0	1 bit
	ESCR:ESBL=1	3 bits
1	ESCR:ESBL=0	2 bits
	ESCR:ESBL=1	4 bits

**Notes:**

- Reception always detects only the first stop bit.
- Set this bit when transmission is disabled (SCR:TXE = 0).

**[bit2:1] Reserved: Reserved bits**

**[bit0] SOE: Serial data output enable bit**

This bit enables/disables the output of serial data.

- This bit is reset when the SMRC:SOEC bit in the clear register is set to "1".
- This bit is set when the SMRS:SOES bit in the set register is set to "1".
- "0": Set the SOUT pin as a general-purpose I/O port.
- "1": Set the SOUT pin as the serial data output pin (SOUT).

Value	Description
0	General-purpose I/O port
1	Serial data output pin

**Note:**

- The SOT pin must also be set as the resource output pin. For how it is set, see "CHAPTER: I/O Port".



### 8.3. Serial Status Register (SSR)

The serial status register (SSR) checks the transmission and reception status, checks the reception error flag, detects the LIN Break Field, or clears the reception error flag.

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	REC	Reserved	LBD	FRE	ORE	RDRF	TDRE	TBI
ACCESS_TYPE	R0,W	R0,W0	R,W	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	1	1

\* Lower byte [7:0] of this register is extended communication control register (ESCR).

#### [bit15] REC: Reception error flag clear bit

This bit clears the FRE and ORE bits in the serial status register (SSR) and the SER flag in the LIN assist mode status register (LAMSR).

- This bit is set when the SSRS:RECS bit in the set register is set to "1".
- Writing "1" clears the error flag.
- Writing "0" does not have any effect.

Value	Description	
	Write	Read
0	No effect	"0" is always read.
1	Clear the reception error flags (FRE, ORE, and LAMSR:SER).	

#### [bit14] Reserved: Reserved bit

#### [bit13] LBD: LIN Break Field detection flag bit

This bit indicates LIN Break Field detection.

The LBD bit is set to "1" when the serial input (SIN) has "L" input for an interval of 11 bits or more. At this time, a status interrupt occurs if the LIN Break Field interrupt enable bit (LBIE) is set to "1".

This bit is reset when the SSRC:LBDC bit in the clear register is set to "1".

(When reading)

"1": Detect a LIN Break Field.

"0": Do not detect a LIN Break Field.

(When writing)

Writing "0" clears LBD bit.

Writing "1" does not have any effect.

Value	Description	
	Write	Read
0	Clear the LBD flag	No LIN Break Field detected
1	No effect	LIN Break Field detected

**[bit12] FRE: Framing error flag bit**

- This bit is set to "1" when a framing error occurs during reception. This bit is cleared by writing "1" to the REC bit in the serial status register (SSR).
- A reception interrupt request is issued when the FRE bit and the RIE bit are "1".
- If this flag is set, the reception data register (RDR) data will be invalid.
- If this flag is set when a reception FIFO is used, the reception FIFO enable bits are cleared, and no reception data is stored to the reception FIFO.

Value	Description
0	No framing error
1	Framing error found

**Notes:**

- *During reception of a LIN Break Field, a framing error is detected before detection of a LIN Break Field if the reception enable setting (SCR:RXE=1) is made. However, header reception is performed normally without being stopped.*
- *In assist mode (LAMCR:LAMEN), suppose that a LIN Break Field is detected and a new LIN Break is then transmitted from the master before the completion of ID Field reception. Then, a framing error is detected by the "L" level at the 10th bit in the new LIN Break Field regardless of the reception disable setting (SCR:RXE=0). However, header reception is performed normally without being stopped.*

**[bit11] ORE: Overrun error flag bit**

- This bit is set to "1" if an overrun occurs during reception. This bit is cleared by writing "1" to the REC bit in the serial status register (SSR).
- A reception interrupt request is issued when the ORE bit and the RIE bit are "1".
- If this flag is set, the reception data register (RDR) data will be invalid.
- If this flag is set when a reception FIFO is used, the reception FIFO enable bits are cleared, and no reception data is stored to the reception FIFO.

Value	Description
0	No overrun error
1	Overrun error found

**[bit10] RDRF: Reception data full flag bit**

- This flag indicates the status of the reception data register (RDR).
- This bit is set to "1" when reception data is loaded into RDR. Reading the reception data register (RDR) causes this bit to be cleared to "0".
- A reception interrupt request is issued when the RDRF bit and the RIE bit are "1".
- When the reception FIFO is used, if the reception FIFO receives a prescribed number of data items, RDRF is set to "1".
- when the reception FIFO is used, this bit is cleared to "0" when the reception FIFO becomes empty as a result of data being read from it.

Value	Description
0	The reception data register (RDR) is empty.
1	The reception data register (RDR) contains data.



**Notes:**

- When the reception FIFO is used and RDRF has become "1", resetting the reception FIFO (FCR0:FCL2, FCL1="1") does not cause RDRF to be set to "0". Therefore, in order to set RDRF to "0" after resetting the reception FIFO, perform a dummy reading of the reception data register in reception disable status (SCR:RXE="0").
- During slave operation (SCR:MS = 1) in assist mode (LAMCR:LAMEN=1), suppose that the setting is such that the reception data register (RDR) is used for the reception of ID Fields (LAMCR:LIDEN=0). Then, the loading of an ID Field into the reception data register (RDR) sets the reception data full flag bit (RDRF=1). The LIN auto header completion flag is also set (LAMSR:LAHC=1) at this time.
- During slave operation (SCR:MS = 1) in assist mode (LAMCR:LAMEN=1), suppose that the setting is such that the LIN assist mode reception ID register is used for the reception of ID Fields (LAMCR:LIDEN = 1). Then, when an ID Field is received, the received ID value is not stored into the reception data register (RDR) and the reception data full flag bit (SSR:RDRF) is not set.
- The Sync Field and the checksum are not stored into the reception data register (RDR) and the reception data full flag bit (SSR:RDRF) is not set.
- The transmitted fields are not stored in the reception data register (RDR) and the reception data full flag bit (SSR:RDRF) is not set.

**[bit9] TDRE: Transmission data empty flag bit**

- This bit indicates the status of the transmission data register (TDR).
- Writing transmission data to TDR sets this bit to "0", indicating that TDR contains valid data. Loading data into the transmission shift register to start transmission sets this bit to "1", indicating that TDR does not contain valid data.
- A transmission interrupt request is issued when the TDRE bit and the TIE bit are "1".
- The TDRE bit is set to "1" when "1" is written to the UPCL bit in the serial control register (SCR).
- During master operation (SCR:MS=0) in assist mode (LAMCR:LAMEN=1), suppose that the setting is such that the transmission data register (TDR) is used for the transmission of ID Fields (LAMCR:LIDEN=0). Then, transmission of the first bit in the ID Field sets the transmission data empty flag (TDRE=1).
- For details on the setting and clearing timing of the TDRE bit when the transmission FIFO is used, see Section "2.1.4. Occurrence of Interrupts and Flag Set Timing When Using Transmission FIFO".

Value	Description
0	The transmission data register (TDR) holds data.
1	The transmission data register is empty.

**[bit8] TBI: Transmission bus idle flag bit**

- This bit indicates that the LIN interface (v2.1) is not performing transmission.
- This bit is set to "0" when transmission data is written to the transmission data register (TDR).
- This bit is set to "1" during those periods in which the transmission data register (TDR) is empty (TDRE = 1) and transmission is not performed.
- In manual mode (LAMCR:LAMEN=0):
  - Setting the LIN Break Field (SMR:LBR=1) sets this bit to "0".
  - Once transmission of the LIN Break Field is complete and the transmission data register is empty, this bit is set to "1".
- In assist mode (LAMCR:LAMEN=1):
  - This bit is set to "0" during transmission of a header in the master operation (SCR:MS=0).
  - Once the header transmission (ID Field transmission) is complete and the transmission data register is empty, this bit is set to "1".
  - This bit is set to "0" during the transmission of a response.
  - Once response transmission (checksum transmission) is complete and the transmission data register is empty, this bit is set to "1".
- A transmission interrupt request is issued if this bit is set to "1" and transmission bus idle interrupt is enabled (SCR:TBIE=1).

Value	Description
0	Transmission in progress
1	No transmission



## 8.4. Extended Communication Control Register (ESCR)

The extended communication control register (ESCR) enables/disables LIN Break Field interrupts, detects LIN Break Fields, and selects lengths for the LIN Break Field, the LIN Break delimiter, and the stop bit.

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved	ESBL	LBL2	LBIE	LBL1	LBL0	DEL1	DEL0
ACCESS_TYPE	R0,W0	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

**[bit7] Reserved: Reserved bit**

### [bit6] ESBL: Extended stop bit length selection bit

This bit selects the length of the stop bits (frame end mark of the transmission data).

- This bit is reset when the ESCRC:ESBLC bit in the clear register is set to "1".
- This bit is set when the ESCRS:ESBLS bit in the set register is set to "1".

Settings where SBL is "0" and ESCR:ESBL is "0": The stop bit is set to 1 bit.

Settings where SBL is "1" and ESCR:ESBL is "0": The stop bit is set to 2 bits.

Settings where SBL is "0" and ESCR:ESBL is "1": The stop bit is set to 3 bits.

Settings where SBL is "1" and ESCR:ESBL is "1": The stop bit is set to 4 bits.

Value	Description	
0	SMR:SBL = 0	1 bit
	SMR:SBL = 1	2 bits
1	SMR:SBL = 0	3 bits
	SMR:SBL = 1	4 bits

#### Notes:

- Reception always detects only the first stop bit.
- Set this bit when transmission is disabled (SCR:TXE=0).
- In assist mode (LAMCR:LAMEN=1), set this bit before setting the LIN Break Field (SCR:LBR=1).

### [bit4] LBIE: LIN Break Field detection interrupt enable bit

This bit enables/disables LIN Break Field detection interrupts.

When the LIN Break Field detection flag (LBD) is "1", enabling interrupts (LBIE=1) generates a reception interrupt.

- This bit is reset when the ESCRC:LBIEC bit in the clear register is set to "1".
- This bit is set when the ESCRS:LBIES bit in the set register is set to "1".

Value	Description
0	Disable LIN Break Field detection interrupts.
1	Enable LIN Break Field detection interrupts.

**[bit5, bit3:2] LBL 2/1/0: LIN Break Field length selection bits (effective only for master operation)**

- These bits set the LIN Break Field duration as a number of bits.
- Set these bits before setting the LBR bit in the serial control register (SCR) to "1" (LIN Break Field transmission).
- During slave operation, a LIN Break Field is detected at the 11th bit regardless of the setting of these bits.

Value			Description
LBL2	LBL1	LBL0	
0	0	0	13-bit length
0	0	1	14-bit length
0	1	0	15-bit length
0	1	1	16-bit length
1	0	0	17-bit length
1	0	1	18-bit length
1	1	0	19-bit length
1	1	1	20-bit length

**Notes:**

- This function is effective only during master operation (SCR:MS=0).
- Set these bits before setting the LIN Break Field (SCR:LBR=1).
- In manual mode (LAMCR:LAMEN=0), only 13-bit to 16-bit lengths are effective. Therefore, always set LBL2 to "0".

**[bit1:0] DEL1 to 0: LIN Break delimiter length selection bits (effective only for master operation)**

- These bits set the length of the LIN Break delimiter as a number of bits.
- Set these bits before setting the LBR bit in the serial control register (SCR) to "1" (LIN Break Field transmission).

Value	Description
00	1-bit length
01	2-bit length
10	3-bit length
11	4-bit length

**Notes:**

- This function is effective only during master operation (SCR:MS=0).
- In assist mode (LAMCR:LAMEN=1), set this bit before setting the LIN Break Field (SCR:LBR=1).



## 8.5. Reception Data Register/Transmission Data Register (RDR/TDR)

The reception data and transmission data registers are placed at the same address. If read, the register functions as a reception data register, and if written to, it functions as a transmission register.

### (1) Reception Data Register (RDR)

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	D7	D6	D5	D4	D3	D2	D1	D0
ACCESS_TYPE	R	R	R	R	R	R	R	R
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

#### [bit7:0] D7 to 0: Reception data

The reception data register (RDR) is the data buffer register used for the reception of serial data.

- Serial data signals transmitted to the serial input pin (SIN pin) are converted by the shift register, and are then stored into the reception data register (RDR).
- The reception data full flag bit (SSR:RDRF) is set to "1" when reception data is stored into the reception data register (RDR). If reception interrupts are enabled (SCR:RIE=1), a reception interrupt request is generated.
- Read the reception data register (RDR) when the reception data full flag bit (SSR:RDRF) is "1". The reception data full flag bit (SSR:RDRF) is automatically cleared to "0" when reading data from the serial reception data register (RDR).
- If a reception error occurs (when SSR:ORE or FRE becomes "1"), the data in the reception data register (RDR) will be invalid.

#### Notes:

- If a reception error occurs, the data in the reception data register (RDR) will be invalid.
- The operation in assist mode (LAMCR:LAMEN=1) is as follows:
  - During slave operation, suppose that the setting is such that the LIN assist mode reception ID register is used for the reception of ID Fields (LAMCR:LIDEN=1). Then, when an ID Field is received, the received ID value is not stored into the reception data register (RDR) and the reception data full flag bit (SSR:RDRF) is not set.
  - During slave operation, suppose that the setting is such that the reception data register (RDR) is used for the reception of ID Fields (LAMCR:LIDEN=0). Then, when an ID Field is received, the received ID value is stored into the reception data register (RDR) and the reception data full flag bit is set (SSR:RDRF=1). Check the ID value after the LIN auto header completion flag is set (LAMSR:LAHC=1).
  - The Sync Field and the checksum are not stored into the reception data register (RDR) and the reception data full flag bit (SSR:RDRF) is not set.
  - The transmitted fields are not stored into the reception data register (RDR) and the reception data full flag bit (SSR:RDRF) is not set.
  - Transmission and reception processing for assist mode stops when a reception error occurs (SSR:FRE, ORE, LAMESR:LCSE, LSFER, LBSE, or LPTER). At this time, the response reception processing stops the storing of the reception data in the reception data register, regardless of the reception enable setting (SCR:RXE=1).
- The operation when the reception FIFO is used is as follows:
  - When the reception FIFO receives a predefined data count, the RDRF to "1".
  - When the reception FIFO becomes empty, RDRF is cleared to "0".

- If a reception error occurs (when SSR:ORE, FRE becomes "1"), the enable bit in the reception FIFO is cleared and the reception data is not stored into the reception FIFO.

## (2) Transmission Data Register (TDR)

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	D7	D6	D5	D4	D3	D2	D1	D0
ACCESS_TYPE	W	W	W	W	W	W	W	W
PROT_TYPE	-							
INITIAL_VALUE	1	1	1	1	1	1	1	1

### [bit7:0] D7 to 0: Transmission data

The transmission data register (TDR) is the data buffer register used for the transmission of serial data.

- If transmission operation is enabled (SCR:TXE = 1), and the data to transmit is written to the transmission data register (TDR), the transmission data is transferred to the transmission shift register, is converted into serial data, and is then sent out from the serial data output pin (SOUT pin).
- The transmission data empty flag (SSR:TDRE) is cleared to "0" when the transmission data is written to the serial transmission data register (TDR).
- If the transmission FIFO is disabled or empty, the transmission data empty flag (SSR:TDRE) is set to "1" when transmission data is transferred to the transmission shift register to start transmission.
- If the transmission data empty flag (SSR:TDRE) is "1", the next data for transmission can be written. If transmission interrupts are enabled, a transmission interrupt is generated. Write the next transmission data after a transmission interrupt is generated or when the transmission data empty flag (SSR:TDRE) is "1".
- When the transmission data empty flag (SSR:TDRE) is "0" and the transmission FIFO is disabled or full, transmission data cannot be written to the transmission data register (TDR).

### Notes:

- The transmission data register is a write-only register, and the reception data register is a read-only register. The two registers are placed at the same address, so the written value differs from the read value.
- For details on the set timing of the transmission data empty flag (SSR:TDRE) when the transmission FIFO is used, see Section "2.1.4 Occurrence of Interrupts and Flag Set Timing When Using Transmission FIFO".
- The upper 2 bits of data are ignored when all of the following conditions are satisfied.
  - LIN hardware assist mode (LAMCR:LAMEN=1)
  - ID DATA register not used (LAMCR:LIDEN=0)
  - ID DATA being transmitted



## 8.6. Serial Auxiliary Control Status Register (SACSR)

The serial auxiliary control status register (SACSR) controls the serial test operation, enables/disables auto baud rate adjustment, enables/disables Sync Field interrupts, selects a serial timer activation method, enables/disables timer interrupts, sets the division value of the serial timer operation clock, and enables/disables the serial timer.

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	STST	BST	SFD	SFDE	AUTE	Reserved		TINT
ACCESS_TYPE	R/W	R,WX	R,W	R/W	R/W	R/W0		R,W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	00		0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	TINTE	Reserved	Reserved	TDIV3	TDIV2	TDIV1	TDIV0	TMRE
ACCESS_TYPE	R/W	R0,W0	R/W0	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

### [bit15] STST: Serial test bit

This bit enables or disables serial test mode.

When serial test mode is enabled, SOUT and SIN are connected together inside the multi-function serial interface, so that the data transmitted from SOUT can be directly received from SIN.

When serial test mode is enabled, the SOUT pin is fixed to "H", and any data input to the SIN pin is ignored.

- This bit is reset when the SACSRC:STSTC bit in the clear register is set to "1".
- This bit is set when the SACSRS:STSTS bit in the set register is set to "1".

Value	Description
0	Disable serial test mode.
1	Enable serial test mode.

#### Note:

- This bit can be changed only when transmission/reception are disabled (SCR:TXE=0, SCR:RXE=0).

### [bit14] BST: Baud rate setting flag

This bit indicates that auto baud rate adjustment has been performed upon the reception of a Sync Field.

This bit is updated upon the detection of a 5th falling edge of the LIN bus in the Sync Field.

Value	Description	
	Write	Read
0	No effect	Without auto baud rate adjustment
1		With auto baud rate adjustment

#### Notes:

- This bit is fixed to "0" when auto baud rate adjustment is disabled (AUTE=0).
- A software reset (SCR:UPCL="1") resets this bit to "0".
- This bit is effective only when the Sync Field detection flag (SACSR:SFD) is "1".
- Writing to this bit is no effect.

**[bit13] SFD: Sync Field detection flag**

This bit indicates that a Sync Field has been detected.

This bit is set to "1" upon the detection of a 5th falling edge of the LIN bus in the Sync Field.

When this bit is "1" and the Sync Field detection interrupt enable bit (SFDE) is "1", a status interrupt request is issued.

Writing to this bit resets this bit to "0".

This bit is reset when the SACSRC:SFDC bit in the clear register is set to "1".

Value	Description	
	Write	Read
0	Clear	Sync Field not detected
1	No effect	Sync Field detected

**Notes:**

- A software reset (SCR:UPCL="1") resets this bit to "0".
- Writing "1" to this bit has no effect.
- This bit is effective both in master mode (SCR:MS=0) and in slave mode (SCR:MS=1).

**[bit12] SFDE: Sync Field detection interrupt enable bit**

This bit enables/disables Sync Field interrupt to the CPU.

When this bit is "1" and the Sync Field detection flag (SFD) is "1", a status interrupt request is issued.

- This bit is reset when the SACSRC:SFDEC bit in the clear register is set to "1".
- This bit is set when the SACSRS:SFDES bit in the set register is set to "1".

Value	Description
0	Disable interrupt of Sync Field detection.
1	Enable interrupt of Sync Field detection.

**[bit11] AUTE: Auto baud rate adjustment bit**

This bit enables/disables auto baud rate adjustment.

- This bit is reset when the SACSRC:AUTEC bit in the clear register is set to "1".
- This bit is set when the SACSRS:AUTES bit in the set register is set to "1".

Value	Description
0	Disable auto baud rate adjustment.
1	Enable auto baud rate adjustment.

**Notes:**

- This bit is internally fixed to "0" in master mode (SCR:MS = 0).
- Setting this bit to "1" sets the timer operation clock division bits (TDIV3 to TDIV0) to 3 (division by 8).
- These bits can be changed from "0" to "1" only when the serial timer enable bit (TMRE) is "0".

**[bit10:9] Reserved: Reserved bits**



**[bit8] TINT: Timer interrupt flag**

If the serial timer register (STMR) and the serial timer comparison register (STMCR) match, the serial timer register (STMR) is set to "0", and this bit is set to "1".

When this bit is "1" and the timer interrupt enable bit (TINTE) is "1", a status interrupt request is issued.

- Writing to this bit resets this bit to "0".
- Writing "1" to this bit has no effect.

This bit is reset when the SACSRC:TINTC bit in the clear register is set to "1".

Value	Description	
	Write	Read
0	Clear	No timer interrupt request
1	No effect	Timer interrupt request issued

**Note:**

- A software reset (SCR:UPCL = "1") resets this bit to "0".

**[bit7] TINTE: Timer interrupt enable bit**

This bit enables/disables timer interrupt to the CPU.

When this bit is "1" and the timer interrupt flag (TINT) is "1", a status interrupt request is issued.

- This bit is reset when the SACSRC:TINTEC bit in the clear register is set to "1".
- This bit is set when the bit SACSRS:TINTES in the set register is set to "1".

Value	Description
0	Disable interrupt triggered by the serial timer.
1	Enable interrupt triggered by the serial timer.

**[bit6:5] Reserved: Reserved bits****[bit4:1] TDIV3 to 0: Timer operation clock division bits**

These bits set the division ratio of the serial timer.

Value				Description						
TDIV3	TDIV2	TDIV1	TDIV0	Division Ratio	$\phi = 8 \text{ MHz}$	$\phi = 10 \text{ MHz}$	$\phi = 16 \text{ MHz}$	$\phi = 20 \text{ MHz}$	$\phi = 24 \text{ MHz}$	$\phi = 32 \text{ MHz}$
0	0	0	0	$\phi$	125ns	100ns	62.5ns	50ns	41.67ns	31.25ns
0	0	0	1	$\phi/2$	250ns	200ns	125ns	100ns	83.33ns	62.5ns
0	0	1	0	$\phi/4$	500ns	400ns	250ns	200ns	166.67ns	125ns
0	0	1	1	$\phi/8$	1us	800ns	500ns	400ns	333.33ns	250ns
0	1	0	0	$\phi/16$	2us	1.6us	1us	800ns	666.67ns	500ns
0	1	0	1	$\phi/32$	4us	3.2us	2us	1.6us	1.33us	1us
0	1	1	0	$\phi/64$	8us	6.4us	4us	3.2us	2.67us	2us
0	1	1	1	$\phi/128$	16us	12.8us	8us	6.4us	5.33us	4us
1	0	0	0	$\phi/256$	32us	25.6us	16us	12.8us	10.67us	8us

$\phi$ : Bus clock

**Notes:**

- These bits can be changed only when the serial timer enable bit (TMRE) is "0".
- Settings other than the above are prohibited.

**[bit0] TMRE: Serial timer enable bit**

This bit enables or disables the serial timer operation.

- This bit is reset when the SACSRC:TMREC bit in the clear register is set to "1".
- This bit is set when the SACSRS:TMRES bit in the set register is set to "1".

Value	Description
0	Stop serial timer operation. When stopped, the value of the serial timer register (STMR) is retained.
1	Changing this bit from "0" to "1" initializes the serial timer register (STMR) to "0" and starts serial timer operation.



## 8.7. Serial Timer Register (STMR)

The serial timer register (STMR) represents the timer value of the serial timer.

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	TM15	TM14	TM13	TM12	TM11	TM10	TM9	TM8
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	TM7	TM6	TM5	TM4	TM3	TM2	TM1	TM0
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

### [bit15:0] TM15 to 0: Timer data bits

These bits indicate the timer value of the serial timer.

During timer operation, the timer value of the serial timer is incremented by 1 for each timer operation clock (specified by SACSR:TDIV3 to TDIV0).

#### Note:

- These bits are initialized to "0" when the timer operation starts.

## 8.8. Serial Timer Comparison Register (STMCR)

The serial timer comparison register (STMCR) sets the timer comparison value of the serial timer.

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	TC15	TC14	TC13	TC12	TC11	TC10	TC9	TC8
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	TC7	TC6	TC5	TC4	TC3	TC2	TC1	TC0
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

### [bit15:0] TC15 to 0: Compare bits

These bits set a comparison value for the serial timer.

These bits are compared with the serial timer register (STMR). If the serial timer register (STMR) coincides with these bits when updated, the serial timer register is set to "0". At this time, the timer interrupt flag (SACSR:TINT) is set to "1".

#### Notes:

- The timer interrupt flag (SACSR:TINT) is fixed to "1" when this register is set to 0x0000, the timer is operating, and the division value (SACSR:TDIV) of the timer operation clock is set to 0b0000.
- This register can be changed only when the serial timer is disabled (SACSR:TMRE="0").
- The serial timer register (STMR) may be reset to 0x0000 before baud rate adjustment when all of the following conditions are satisfied. For this reason, set these bits to a value larger than that set for the Sync Field upper limit bits (SFUR) when the auto baud rate adjustment bit (SACSR:AUTE) is "1".
  - The auto baud rate adjustment bit (SACSR:AUTE) is "1".
  - The value of these bits is equal to or less than the value set for the Sync Field upper limit bits (SFUR).



## 8.9. Sync Field Upper Limit Register (SFUR)

The Sync Field upper limit register (SFUR) sets the upper limit on the values that can be set in the baud rate generator register for auto baud rate adjustment.

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved	TU14	TU13	TU12	TU11	TU10	TU9	TU8
ACCESS_TYPE	R0, W0	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	TU7	TU6	TU5	TU4	TU3	TU2	TU1	TU0
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

**[bit15] Reserved: Reserved bit**

### **[bit14:0] TU14 to 0: Upper limit bits**

These bits set the upper limit on the values that can be set in the baud rate generator register (BGR) for auto baud rate adjustment.

Suppose that the auto baud rate adjustment bit (SACSR:AUTE) is "1" and that operation is being performed in slave mode (SCR:MS=1). Also suppose that the value of the serial timer register (STMR) after receiving Sync Field does not exceed the value of these bits and is not less than the value of the Sync Field lower limit register (SFLR). Then, the baud rate generator register (BGR) is set to the value of the serial timer register (STMR).

**Note:**

- These bits can be changed when the auto baud rate adjustment bit (SACSR:AUTE) is "0".

## 8.10. Sync Field Lower Limit Register (SFLR)

The Sync Field lower limit register (SFLR) sets the lower limit for the values that can be set in the baud rate generator register for auto baud rate adjustment.

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved	TL14	TL13	TL12	TL11	TL10	TL9	TL8
ACCESS_TYPE	R0, W0	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	TL7	TL6	TL5	TL4	TL3	TL2	TL1	TL0
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

**[bit15] Reserved: Reserved bit**

### **[bit14:0] TL14 to 0: Lower limit bits**

These bits set the lower limit on the values that can be set in the baud rate generator register (BGR) for auto baud rate adjustment.

Suppose that the auto baud rate adjustment bit (SACSR:AUTE) is "1" and that operation is being performed in slave mode (SCR:MS=1). Also suppose that the value of the serial timer register (STMR) after receiving Sync Field does not exceed the value of the Sync Field upper limit register (SFUR) and is not less than the value of these bits. Then, the baud rate generator register (BGR) is set to the value of the serial timer register (STMR).

**Note:**

- These bits can be changed when the auto baud rate adjustment bit (SACSR:AUTE) is "0".



### 8.11. Baud Rate Generator Register 1, 0 (BGR1 to 0)

Baud rate generator register 1, 0 (BGR1, BGR0) sets the division ratio of the serial clock. An external clock can also be selected as the clock source of the reload counter.

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	EXT	BGR1						
ACCESS_TYPE	R/W	R/W						
PROT_TYPE	-							
INITIAL_VALUE	0	0000000						

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	BGR0							
ACCESS_TYPE	R/W							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

#### [bit15] EXT: External clock selection bit

The EXT bit, bit15, selects the internal clock or an external clock as the clock source of the reload counter. EXT = 0 selects the internal clock. EXT = 1 selects an external clock.

Value	Description
0	Use the internal clock.
1	Use an external clock.

#### [bit14:8] BGR1: Baud rate generator register 1

#### [bit7:0] BGR0: Baud rate generator register 0

- The baud rate generator registers set the division ratio of the serial clock.
- BGR1 represents the higher bits while BGR0 represents the lower bits. The reload value for counting can be written to these registers, and the set reload values can be read from these registers.
- The reload counter starts counting when a reload value is written in baud rate generator register 1, 0 (BGR1, BGR0).

bit14:8	Description
Write	Write to reload counter bits 8 to 14.
Read	Read the setting value of BGR1.

bit7:0	Description
Write	Write to reload counter bits 0 to 7.
Read	Read the setting value of BGR0.

**Notes:**

- Use 16-bit access for writing a value to the baud rate generator register (BGR1, BGR0).
- If the values set in the baud rate generator registers (BGR1, BGR0) are changed, the new values are reloaded when the counter value becomes "0x00000". Therefore, to enable the new value immediately, execute programmable clear (UPCL) after changing the values of BGR1/0.
- When the reload value is even, the "L" width of the serial clock is 1-bus-clock-cycle longer than the "H" width. When the reload value is odd, the "L" width and the "H" width of the serial clock are equal.
- Set the reload value to 3 or larger. However, data may not be received normally depending on baud rate errors and the reload value setting.
- To change the clock setting to that of the external clock (EXT=1) when the baud rate generator is operating, perform the following: 1. Write "0" to baud rate generators 1, 0 (BGR1, BGR0). 2. Execute programmable clear (UPCL). 3. Set external clock mode (EXT=1).





## 8.12. LIN Assist Mode Status Register (LAMSR)

The LIN assist mode status register (LAMSR) checks the auto header transmission/reception status and the reception error flag.

BITS_OFFSET	15	14	13	12	11	10	9	8
BITS_NAME	LER	SER	RDRF	TDRE	TBI	LCSC	Reserved	LAHC
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,W	R0,W0	R,W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	1	1	0	0	0

\* Lower byte [bit7:0] of this register is LIN assist mode control register (LAMCR).

### [bit15] LER: LIN representative error flag bit

- This bit is set to "1" when any of the following errors occurs. For details on the setting and clearing conditions for the error flag bit, see the explanations of the individual bits of the LIN assist mode error status register (LAMESR).
  - LIN bus error flag bit (LBSE)
  - LIN Sync Data error flag bit (LSFER)
  - LIN ID parity error flag bit (LPTE)
  - LIN checksum error flag bit (LCSE)

Value	Description
0	There is no error.
1	An error has occurred.

#### Note:

- In manual mode (LAMCR:LAMEN=0), this bit is always read as "0".

### [bit14] SER: Serial interface representative error flag bit

- This bit is set to "1" when any of the following errors occurs. For details on the setting and clearing conditions for the error flag bit, see the explanations of the individual bits of the serial status register (SSR).
  - Framing error flag bit (FRE)
  - Overrun error flag bit (ORE)

Value	Description
0	There is no error.
1	An error has occurred.

#### Note:

- This bit is effective in LIN mode.

### [bit13] RDRF: Reception data full flag bit

- This bit is the same as the reception data full flag bit (RDRF) in the serial status register (SSR). For an explanation of this bit, see the description of the serial status register (SSR).

#### Note:

- This bit is effective in LIN mode.

**[bit12] TDRE: Transmission data empty flag bit**

- This bit is the same as the transmission data empty flag bit (TDRE) in the serial status register (SSR). For an explanation of this bit, see the description of the serial status register (SSR).

**Note:**

- *This bit is effective in LIN mode.*

**[bit11] TBI: Transmission bus idle flag bit**

- This bit is the same as the transmission bus idle flag bit (TBI) in the serial status register (SSR). For an explanation of this bit, see the description of the serial status register (SSR).

**Note:**

- *This bit is effective in LIN mode.*

**[bit10] LCSC: LIN checksum calculation completion flag bit**

- This bit indicates the completion of checksum calculation.
- This bit is set to "1" upon the completion of checksum calculation, which happens when data of the configured length (LAMCR:LDL3 to LDL0) and a checksum are received during assist mode (LAMCR:LAMEN = 1) reception operation.
- A status interrupt request is issued when the LIN checksum calculation completion flag bit (LCSC) and the checksum calculation completion interrupt enable bit (LCSCIE) are "1".

(When reading)

"1": Checksum calculation completion has been detected.

"0": Checksum calculation completion has not been detected.

(When writing)

Writing "0" clears the LCSC bit.

Writing "1" does not have any effect.

Value	Description	
	Write	Read
0	Clear LCSC flag	Checksum calculation is in progress. Or Checksum calculation start is being awaited.
1	No effect	Checksum calculation has been completed.

**Notes:**

- *This bit is reset when the LAMSRC:LCSCC bit in the clear register is set to "1".*
- *This bit is effective in assist mode (LAMCR:LAMEN=1).*

**[bit9] Reserved: Reserved bit**

**[bit8] LAHC: LIN auto header completion flag bit**

- This bit indicates the LIN auto header status.
- This bit is set to "1" when a LIN header is received in assist mode (LAMCR:LAMEN=1).
- A status interrupt request is issued when the LIN auto header completion flag bit (LAHC) and the LIN auto header completion interrupt enable bit (LAHCIE) are "1".
- After this bit is set to "1", reading the LIN auto header reception ID register (LAMRID) sets this bit to "0".

(When reading)

"1": LIN auto header completion has been detected.

"0": LIN auto header completion has not been detected.

(When writing)

Writing "0" clears the LAHC bit.

Writing "1" does not have any effect.

Value	Description	
	Write	Read
0	Clear LAHC flag	LIN auto header is being received. Or, its reception is being awaited.
1	No effect	LIN auto header has been received.

**Notes:**

- This bit is reset when the LAMSRC:LAHCC bit in the clear register is set to "1".
- This bit is effective in assist mode (LAMCR:LAMEN=1).

### 8.13. LIN Assist Mode Control Register (LAMCR)

The LIN assist mode control register (LAMCR) enables LIN auto header processing, enables the use of the LIN ID register, selects a LIN checksum type, clears TDR, and sets the LIN data length in LIN assist mode.

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	LDL3	LDL2	LDL1	LDL0	LTDRCL	LCSTYP	LIDEN	LAMEN
ACCESS_TYPE	R/W	R/W	R/W	R/W	R0,W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

#### [bit7:4] LDL3 to 0: LIN data length setting bits

- These bits set the length of the LIN response data within a range of 0 to 8 bytes.
- The setting value should represent the data length.
- Data of the length specified by these bits is sent in the transmission operation.
- Data of the length specified by these bits is received in the reception operation.

Value				Description
LDL3	LDL2	LDL1	LDL0	
0	0	0	0	0 byte
0	0	0	1	1 byte
0	0	1	0	2 bytes
0	0	1	1	3 bytes
0	1	0	0	4 bytes
0	1	0	1	5 bytes
0	1	1	0	6 bytes
0	1	1	1	7 bytes
1	0	0	0	8 bytes
Other than the above				Setting prohibited

#### Notes:

- This function is effective only in LIN assist mode (LAMEN=1).
- During master mode operation, set these bits before the generation of LIN Break Fields (SCR:LBR=1).
- During slave mode operation, set these bits before the start of response transmission/reception.

#### [bit3] LTDRCL: LIN transmission data register clear bit

- This bit clears the transmission data register (TDR).
- When set to "1": The transmission data empty flag bit (SSR:TDRE) and the transmission bus idle flag bit (SSR:TBI) are set to "1".
- When set to "0": No effect on the operation.
- When it is read, "0" is always read.
- This bit is set when the LAMCRS:LTDRCLS bit in the set register is set to "1".

Value	Description	
	Write	Read
0	No effect	"0" is always read.
1	Set the transmission data empty flag bit (SSR:TDRE) and the transmission bus idle flag bit (SSR:TBI) to "1".	

**Notes:**

- Clearing the transmission data register does not reset the transmission FIFO.
- When the transmission FIFO is used, clear the transmission FIFO (FCR0:FCL1 or FCR0:FCL2) and then clear the transmission data register.
- Do not set this bit to "1" during transmission.

**[bit2] LCSTYP: LIN checksum type selection bit**

This bit selects the LIN checksum type.

- This bit is reset when the LAMCRC:LCSTYPC bit in the clear register is set to "1".
- This bit is set when the LAMCRS:LCSTYPS bit in the set register is set to "1".

"0": The standard checksum is selected.

"1": The extended checksum is selected.

Value	Description
0	Standard checksum
1	Extended checksum

**Notes:**

- This function is effective only in LIN assist mode (LAMEN=1).
- During master mode operation, set these bits before the generation of LIN Break Fields (SCR:LBR=1).
- During slave mode operation, set these bits before the start of response transmission/reception.

**[bit1] LIDEN: LIN ID register use enable bit**

This bit sets whether to use the LIN assist mode transmission/reception ID register (LAMTID/LAMRID).

- This bit is reset when the LAMCRC:LIDENC bit in the clear register is set to "1".
- This bit is set when the LAMCRS:LIDENS bit in the set register is set to "1".

(In master mode (SCR:MS=0))

When set to "0": Data written to the transmission data register (TDR) as the transmission data of LIN ID Field is used.

When set to "1": Data written to the LIN assist mode transmission ID register (LAMTID) as the transmission data of LIN ID Field is used.

(In slave mode (SCR:MS=1))

When set to "0": The received data of LIN ID Field is stored into the reception data register (RDR).

When set to "1": The received data of LIN ID Field is stored into the LIN assist mode reception ID register (LAMRID).

Value	Description	
	Master	Slave
0	Use the transmission data register (TDR).	Use the reception data register (RDR).
1	Use the LIN assist mode transmission ID register (LAMTID).	Use the LIN assist mode reception ID register (LAMRID).

**Note:**

- This function is effective only in LIN assist mode (LAMEN=1).

**[bit0] LAMEN: LIN assist mode processing enable bit**

This bit sets whether LIN assist mode is used.

- This bit is reset when the LAMCRC:LAMENC bit in the clear register is set to "1".
- This bit is set when the LAMCRS:LAMENS bit in the set register is set to "1".

When set to "0": LIN manual mode is selected.

When set to "1": LIN assist mode is selected.

Value	Description
0	Manual mode
1	Assist mode

**Notes:**

- In manual mode, change this bit when LIN transmission/reception is disabled (SCR:RXE=0, SCR:TXE=0).
- In assist mode, do not change this bit when LIN is operating.
- When changing this bit, execute programmable clear (SCR:UPCL=1) immediately after changing the bit.



### 8.14. LIN Assist Mode Interrupt Enable Register (LAMIER)

The LIN assist mode interrupt enable register (LAMIER) enables/disables LIN auto header completion interrupt, LIN checksum calculation completion interrupt, LIN bus error interrupt, LIN ID parity error interrupt, LIN Sync Data error interrupt, and LIN checksum error interrupt.

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved	LCSERIE	LPTERIE	LSFERIE	LBSERIE	LCSCIE	Reserved	LAHCIE
ACCESS_TYPE	R0,W0	R/W	R/W	R/W	R/W	R/W	R0,W0	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

\* Lower byte [bit7:0] of this register is LIN assist mode transmission/reception register (LAMTID/LAMRID).

**Note:**

- Please use 8-bit access to access this register.

#### [bit15] Reserved: Reserved bit

#### [bit14] LCSERIE: LIN checksum error interrupt enable bit

- This bit enables/disables LIN checksum error interrupt request output to the CPU.
- A reception interrupt request is issued when the LCSERIE bit and the LAMESR:LCSESR bit are "1".
- This bit is reset when the LAMIERC:LCSERIEC bit in the clear register is set to "1".
- This bit is set when the LAMIERC:LCSERIES bit in the set register is set to "1".

Value	Description
0	Disable LIN checksum error interrupt.
1	Enable LIN checksum error interrupt.

#### [bit13] LPTERIE: LIN ID parity error interrupt enable bit

- This bit enables/disables LIN ID parity error interrupt request output to the CPU.
- A reception interrupt request is issued when the LPTERIE bit and the LAMESR:LPTESR bit are "1".
- This bit is reset when the LAMIERC:LPTERIEC bit in the clear register is set to "1".
- This bit is set when the LAMIERC:LPTERIES bit in the set register is set to "1".

Value	Description
0	Disable LIN ID parity error interrupt.
1	Enable LIN ID parity error interrupt.

#### [bit12] LSFERIE: LIN Sync Data error interrupt enable bit

- This bit enables/disables LIN Sync Data error interrupt request output to the CPU.
- A reception interrupt request is issued when the LSFERIE bit and the LAMESR:LSFESR bit are "1".
- This bit is reset when the LAMIERC:LSFERIEC bit in the clear register is set to "1".
- This bit is set when the LAMIERC:LSFERIES bit in the set register is set to "1".

Value	Description
0	Disable LIN Sync Data error interrupt.
1	Enable LIN Sync Data error interrupt.

**[bit11] LBSERIE: LIN bus error interrupt enable bit**

- This bit enables/disables LIN bus error interrupt request output to the CPU.
- A reception interrupt request is issued when the LBSERIE bit and the LAMESR:LBSE bit are "1".
- This bit is reset when the LAMIERC:LBSEIEC bit in the clear register is set to "1".
- This bit is set when the LAMIERC:LBSEIES bit in the set register is set to "1".

Value	Description
0	Disable LIN bus error interrupt.
1	Enable LIN bus error interrupt.

**[bit10] LCSCIE: LIN checksum calculation completion interrupt enable bit**

- This bit enables/disables LIN checksum calculation completion interrupt request output to the CPU.
- A status interrupt request is issued when the LCSCIE bit and LAMSR:LCSC bit are "1".
- This bit is reset when the LAMIERC:LCSCIEC bit in the clear register is set to "1".
- This bit is set when the LAMIERC:LCSCIES bit in the set register is set to "1".

Value	Description
0	Disable LIN checksum calculation completion interrupt.
1	Enable LIN checksum calculation completion interrupt.

**[bit9] Reserved: Reserved bit**

**[bit8] LAHCIE: LIN auto header completion interrupt enable bit**

- This bit enables/disables LIN auto header completion interrupt request output to the CPU.
- A status interrupt request is issued when the LAHCIE bit and the LAMSR:LAHC bit are "1".
- This bit is reset when the LAMIERC:LAHCIEC bit in the clear register is set to "1".
- This bit is set when the LAMIERC:LAHCIES bit in the set register is set to "1".

Value	Description
0	Disable LIN auto header completion interrupt.
1	Enable LIN auto header completion interrupt.





### 8.15. LIN Assist Mode Transmission/Reception ID Register (LAMTID/LAMRID)

The LIN assist mode transmission/reception ID register (LAMTID/LAMRID) displays the reception LIN ID parity, sets the transmission LIN ID, and displays the reception ID.

#### (1) LIN Assist Mode Transmission ID Register (LAMTID)

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved		LID5	LID4	LID3	LID2	LID1	LID0
ACCESS_TYPE	W0		W	W	W	W	W	W
PROT_TYPE	-							
INITIAL_VALUE	00		0	0	0	0	0	0

**[bit7:6] Reserved: Reserved bits**

#### **[bit5:0] LID5 to 0: LIN ID setting bits**

(When writing)

When assist mode is set to master and the LIN ID register use enable bit (LIDEN) is set to enabled, these bits set LIN ID Field data.

**Note:**

- This function is effective only in LIN assist mode (LAMCR:LAMEN=1).

**(2) LIN Assist Mode Reception ID Register (LAMRID)**

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	P1	P0	LID5	LID4	LID3	LID2	LID1	LID0
ACCESS_TYPE	R	R	R	R	R	R	R	R
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

**[bit7:6] P1 to 0: LIN ID parity display bits**

(When reading)

When assist mode is set to slave and the LIN ID register use enable bit (LIDEN) is set to enabled, these bits indicate the parity value of the received LIN ID Field.

**[bit5:0] LID5 to 0: LIN ID setting/display bits**

(When reading)

When assist mode is set to slave and the LIN ID register use enable bit (LIDEN) is set to enabled, these bits indicate the data of the received LIN ID Field.

**Notes:**

- This function is effective only in LIN assist mode (LAMCR:LAMEN=1).
- These bits indicate the data of the ID Field received in this register even if a LIN ID parity error occurs.



## 8.16. LIN Assist Mode Error Status Register (LAMESR)

The LIN assist mode error status register (LAMESR) checks the flag of a LIN checksum error, LIN Sync Data error, LIN ID parity error, and LIN bus error.

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved	LCSER	LPTR	LSFER	LBSER	Reserved		
ACCESS_TYPE	R0,W0	R,W	R,W	R,W	R,W	R0,W0		
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	000		

\* Lower byte [7:0] of this register is LIN assist mode error test register (LAMERT).

### Note:

- This register just can be accessed with 8bit access or 16-bit access.

### [bit15] Reserved: Reserved bit

### [bit14] LCSEr: LIN checksum error flag bit

- This bit is set to "1" when a LIN checksum error occurs.
- Writing "0" to this bit resets this error flag bit to "0".
- A reception interrupt request is issued when the LCSEr bit and the LCSErIE bit are "1".
- This bit is reset when the LAMESRC:LCSErC bit in the clear register is set to "1".

Value	Description	
	Write	Read
0	Clear the error flag	There is no error.
1	No effect	An error has occurred.

### Note:

- This function is effective only in LIN assist mode (LAMCR:LAMEN=1).

**[bit13] LPTER: LIN ID parity error flag bit**

- This bit is set to "1" if a LIN ID parity error occurs.
- Writing "0" to this bit resets this error flag bit to "0".
- A reception interrupt request is issued when the LPTER bit and the LPTER IE bit are "1".
- When this flag is set, the received ID Field data is displayed in the LIN reception ID register (LAMRID) or the reception data register (RDR).
- This bit is reset when the LAMESRC:LPTERC bit in the clear register is set to "1".

Value	Description	
	Write	Read
0	Clear the error flag	There is no error.
1	No effect	An error has occurred.

**Note:**

- This function is effective only in LIN assist mode (LAMCR:LAMEN=1).

**[bit12] LSFER: LIN Sync Data error flag bit**

- This bit detects whether the Sync Field value is 0x55 when auto baud rate adjustment is disabled (SACSR:AUTE = 0) in slave mode (SCR:MS=1).
- A LIN Sync Data error is set to "1" when auto baud rate adjustment is disabled (SACSR:AUTE=0) in slave mode (SCR:MS=1).
- Writing "0" to this bit resets this error flag bit to "0".
- A reception interrupt request is issued when the LSFER bit and the LSFERIE bit are "1".
- This bit is reset when the LAMESRC:LSFERC bit in the clear register is set to "1".

Value	Description	
	Write	Read
0	Clear the error flag	There is no error.
1	No effect	An error has occurred.

**Note:**

- This function is effective only in LIN assist mode (LAMCR:LAMEN=1).

**[bit11] LBSER: LIN bus error flag bit**

- This bit is set to "1" when a LIN bus error occurs.
- Writing "0" to this bit resets this error flag bit to "0".
- A reception interrupt request is issued when the LBSER bit and the LBSER IE bit are "1".
- Reception data with an error is stored in the reception data register (RDR), when this flag is set in the ID Field and data field.
- This bit is reset when the LAMESRC:LBSERC bit in the clear register is set to "1".

Value	LIN Bus Error Flag Bit	
	Write	Read
0	Clear the error flag	There is no error.
1	No effect	An error has occurred.

**Note:**

- *This function is effective only in LIN assist mode (LAMCR:LAMEN=1).*

**[bit10:8] Reserved: Reserved bits**

## 8.17. LIN Assist Mode Error Test Register (LAMERT)

The LIN assist mode error test register (LAMERT) sets the pseudo error setting for a framing error, LIN bus error, LIN Sync Data error, LIN ID parity error, and LIN checksum error based on the settings of the key code control bit and pseudo error setting bit.

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	KEY1	KEY0	Reserved	LCSERT	LPTERT	LSFERT	LBSERT	FRET
ACCESS_TYPE	R0,W	R0,W	R0,W0	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

### Notes:

- In manual mode (LAMCR:LAMEN=0), the setting of pseudo error test mode is prohibited.
- This register just can be accessed with 8bit access or 16bit access.

### [bit7:6] KEY1 to 0: Key code control bits

- This bit is a key code register that enables the following pseudo error settings.
  - Framing error pseudo error setting bit (FRET)
  - LIN bus error pseudo error setting bit (LBSERT)
  - LIN Sync Data error pseudo error setting bit (LSFERT)
  - LIN ID parity error pseudo error setting bit (LPTERT)
  - LIN checksum error pseudo error setting bit (LCSERT)
- To set the pseudo error setting, write a value according to the following procedure.
  - (KEY1 to 0 = 0b00) + write a pseudo error setting value
  - (KEY1 to 0 = 0b01) + write the pseudo error setting value (same as the previous value)
  - (KEY1 to 0 = 0b10) + write the pseudo error setting value (same as the previous value)
  - (KEY1 to 0 = 0b11) + write the pseudo error setting value (same as the previous value)
  - The pseudo error setting value is enabled when the value is written for the 4th time.
- The value written to this register is invalid if you do not follow this setting procedure (if a value is written or read to/from another register during the writing process, if the value written is incorrect, or if a value is read from this register during the writing process).
- To clear a pseudo error setting, follow the same procedure as that for making a pseudo error setting.

### Note:

- Assist mode is stopped when any of the following errors occurs.
  - LIN bus error
  - LIN framing error
  - LIN Sync Data error
  - LIN ID parity error
  - LIN checksum error

### [bit5] Reserved: Reserved bit

**[bit4] LCSERT: LIN checksum error pseudo error setting bit**

- This bit controls the occurrence of a LIN checksum error.
- In assist mode, set this bit to "1" (an error has occurred) before sending response data. When sending a checksum, it is output after being inverted. When an inverted checksum is received, a LIN checksum error occurs and the flag bit (LAMESR:LCSER) is set to "1".
- The pseudo error function is enabled and the error continues to occur until the setting of this bit is disabled (= "0").

Value	Description
0	No error
1	An error has occurred.

**[bit3] LPTERT: LIN ID parity error pseudo error setting bit**

- This bit controls the occurrence of a LIN ID parity error.
- Set this bit to "1" (an error has occurred) before setting the Lin Break Field (SCR:LBR=0) when assist mode is set to master (SCR:MS=0). When sending the ID Field, the ID parity bit (2-bit) is output after being inverted. When the ID Field of an inverted ID parity is received, a LIN ID parity error occurs and the flag bit (LAMESR:LPTER) is set to "1".
- The pseudo error function is enabled and the error continues to occur until the setting of this bit is disabled (= "0").

Value	Description
0	No error
1	An error has occurred.

**[bit2] LSFERT: LIN Sync Data error pseudo error setting bit**

- This bit controls the occurrence of a LIN Sync Data error.
- Set this bit to "1" (an error has occurred) before setting the Lin Break Field (SCR:LBR=0) when assist mode is set to master (SCR:MS=0). All bits in the LIN Sync Field are output after being inverted.
- The pseudo error function is enabled and this bit continues to output the inverted Sync Field data until this bit set is disabled (= "0").

Value	Description
0	No error
1	An error has occurred.

**Note:**

- Set the LIN bus error pseudo error setting bit (LBSERT=1) at the same time as this bit (LSFERT=1) is set.

**[bit1] LBSERT: LIN bus error pseudo error setting bit**

- This bit controls the occurrence of a LIN bus error.
- When assist mode is set to master and when this bit is set to "1" (an error has occurred) in each field (Sync Field, ID Field, data, and checksum) to which data has been sent, a LIN bus error occurs and the flag bit (LAMESR:LBSER) is set to "1".
- When assist mode is set to slave and when this bit is set to "1" (an error has occurred) in each field (data and checksum) to which a response has been sent, a LIN bus error occurs and the flag bit (LAMESR:LBSER) is set to "1".
- The pseudo error function is enabled and the error continues to occur until the setting of this bit is disabled (= "0").

Value	Description
0	No error
1	An error has occurred.

**Note:**

- The LIN bus error pseudo error setting cannot be set for the LIN Break Field.

**[bit0] FRET: Framing error pseudo error setting bit**

- This bit controls the occurrence of a LIN framing error.
- In assist mode, when this bit is set to "1" (an error has occurred) in each field (Sync Field, ID Field, data, and checksum), the stop bit is output after being inverted. If an inverted stop bit is received, a framing error occurs and the flag bit (SSR:FRE) is set to "1".
- The pseudo error function is enabled and the error continues to occur until the setting of this bit is disabled (= "0").

Value	Description
0	No error
1	An error has occurred.





### 8.18. FIFO Control Register 1 (FCR1)

The FIFO control register 1 (FCR1) selects transmission and reception FIFOs, enables transmission FIFO interrupts, and controls the interrupt flag.

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved		Reserved	FLSTE	FRIIE	FDRQ	FTIE	FSEL
ACCESS_TYPE	R/W0		R0,WX	R/W	R/W	R,W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	00		0	0	0	1	0	0

\* Lower byte [7:0] of this register is FIFO control register 0 (FCR0).

**[bit15:13] Reserved: Reserved bits**

#### **[bit12] FLSTE: Retransmission data lost detection enable bit**

This bit enables FLST bit detection.

- This bit is reset when the FCR1C:FLSTEC bit in the clear register is set to "1".
- This bit is set when the FCR1S:FLSTES bit in the set register is set to "1".

Value	Description
0	Disable data lost detection. (Disable FLST bit detection)
1	Enable data lost detection. (Enable FLST bit detection)

**Note:**

- To set this bit to "1", first set the FSET bit to "1" and then set this bit to "1".

**[bit11] FRIIE: Reception FIFO idle detection enable bit**

When the reception FIFO contains valid data, this bit enables or disables the detection of continuation of reception idle status for an 8-bit time or longer. If the reception interrupt is enabled (SCR:RIE=1), the detection of reception idle status triggers a reception interrupt.

- This bit is reset when the FCR1C:FRIIEC bit in the clear register is set to "1".
- This bit is set when the bit FCR1S:FRIIES in the set register is set to "1".

"0": Disable detection of reception idle status.

"1": Enable detection of reception idle status.

Value	Description
0	Disable detection of reception FIFO idle.
1	Enable detection of reception FIFO idle.

**Note:**

- To use the reception FIFO, set this bit to "1".

**[bit10] FDRQ: Transmission FIFO data request bit**

This bit requests transmission FIFO data.

Value "1" of this bit indicates that transmission data is requested. At this time, a transmission interrupt request is issued if transmission interrupt is enabled (FTIE=1).

FDRQ set conditions

- When transmission FIFO interrupt control is not used
  - FBYTE (for transmission)=0 (Transmission FIFO is empty.)
  - Reset of the transmission FIFO
- When transmission FIFO interrupt control is used
  - FTICR setting value  $\geq$  FTICR read value (The amount of data in the transmission FIFO is equal to the interrupt trigger level or lower.)
  - Reset of the transmission FIFO

FDRQ clearing conditions

- Writing "0" to this bit.
- The transmission FIFO is full.

Value	Description
0	No transmission FIFO data request
1	Transmission FIFO data request issued

**Notes:**

- The FSEL bit cannot be changed when this bit is "0".
- Writing "0" to this bit is prohibited when the set value is equal to or less than a setting value.
- When you write required data to the transmission FIFO after a transmission interrupt occurs, write "0" to the FIFO transmission data request bit (FCR1:FDRQ) to clear the interrupt.

**[bit9] FTIE: Transmission FIFO interrupt enable bit**

This bit enables a transmission FIFO interrupt. If this bit is set to "1", a transmission interrupt occurs when the FDRQ bit is "1".

- This bit is reset when the FCR1C:FTIEC bit in the clear register is set to "1".
- This bit is set when the FCR1S:FTIES bit in the set register is set to "1".

Value	Description
0	Disable transmission FIFO interrupt.
1	Enable transmission FIFO interrupt.

**[bit8] FSEL: FIFO selection bit**

This bit selects the transmission and reception FIFO.

- This bit is reset when the FCR1C:FSELC bit in the clear register is set to "1".
- This bit is set when the FCR1S:FSELS bit in the set register is set to "1".

"0": Allocate as transmission FIFO: FIFO1 and as reception FIFO: FIFO2.

"1": Allocate as transmission FIFO: FIFO2 and as reception FIFO: FIFO1.

Value	Description
0	Transmission FIFO: FIFO1, reception FIFO: FIFO2
1	Transmission FIFO: FIFO2, reception FIFO: FIFO1

**Notes:**

- This bit is not cleared by a FIFO reset (FCL2, FCL1=1).
- To change the value of this bit, disable the FIFO operation (FE2, FE1=0) first.
- This bit cannot be changed when FDRQ=0.
- Set the FIFO selection bit (FSEL) before setting the FIFO byte register (FBYTE) and the transmission FIFO interrupt control register (FTICR).
- Access cannot be performed at the same time as the FIFO byte register (FBYTE).

## 8.19. FIFO Control Register 0 (FCR0)

FIFO control register 0 (FCR0) enables/disables FIFO operation, resets a FIFO, saves the read pointer, and makes the retransmission setting.

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved	FLST	FLD	FSET	FCL2	FCL1	FE2	FE1
ACCESS_TYPE	R0,W0	R,WX	R,W	R,W	R0,W	R0,W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

**[bit7] Reserved: Reserved bit**

### **[bit6] FLST: FIFO retransmission data lost flag bit**

This bit indicates that the retransmission data for the transmission FIFO has been lost.

FLST set condition

- Data is written to the FIFO when the FLSTE bit in the FIFO control register 1 (FCR1) is "1" and the write pointer of the transmission FIFO and the read pointer saved by the FSET bit coincide.

FLST clearing conditions

- FIFO reset (writing "1" to FCL)
- Writing "1" to the FSET bit

If "1" is set in this bit, the data pointed to by the read pointer saved with the FSET bit is overwritten. For this reason, re-transmission with the FLD bit cannot be set even if an error occurs. When performing retransmission with this bit set to "1", reset the FIFO and then write the data to the FIFO again.

Value	Description
0	Data has not been lost.
1	Data has been lost.

**[bit5] FLD: FIFO pointer reload bit**

This bit reloads, to the read pointer, the data saved to the transmission FIFO by the FSET bit. This bit is used for retransmission in cases such as communication errors.

This bit is set to "0" when retransmission setting has been completed.

- This bit is set when the FCR0S:FLDS bit in the set register is set to "1".

Value	Description
0	Do not execute reload.
1	Execute reload.

**Notes:**

- While this bit is set to "1", reloading to the read pointer is in progress, so writing is prohibited except in the case of a FIFO reset.
- Setting this bit to "1" is prohibited when FIFO is enabled or transmission is in progress.
- To set the TIE and TBIE bits to "1", first set them to "0" and write "1" to this bit, and then enable the transmission FIFO and set the TIE and TBIE bits to "1".

**[bit4] FSET: FIFO pointer saving bit**

This bit stores the read pointer of the transmission FIFO.

If the read pointer is saved before transmission starts and then a communication error occurs, retransmission is possible if the FLST bit is "0".

- This bit is set when the FCR0S:FSETS bit in the set register is set to "1".

"1": Save the current read pointer value.

"0": No effect.

Value	Description
0	Do not save.
1	Save the read pointer value.

**Note:**

- Set this bit to "1" when the transmission byte count (FBYTE) is 0.

**[bit3] FCL2: FIFO2 reset bit**

This bit resets FIFO2.

If this bit is set to "1", the internal status of FIFO2 is initialized.

Only the FCR0:FLST bit is initialized. The values of the other bits in the FCR1/0 registers are retained.

- This bit is set when the FCR0S:FCL2S bit in the set register is set to "1".

Value	Description	
	Write	Read
0	No effect	"0" is always read.
1	Reset FIFO2.	

**Notes:**

- Disable transmission/reception first and then execute a FIFO2 reset.
- Execute the reset after setting the transmission FIFO interrupt enable bit to "0".
- The valid data count for the FBYTE2 register is set to 0.
- The TDR register and RDR register are not initialized.

**[bit2] FCL1: FIFO1 reset bit**

This bit resets FIFO1.

If this bit is set to "1", the internal status of FIFO1 is initialized.

Only the FCR0:FLST bit is initialized. The values of the other bits in the FCR1/0 registers are retained.

- This bit is set when the FCR0S:FCL1S bit in the set register is set to "1".

Value	Description	
	Write	Read
0	No effect	"0" is always read.
1	Reset FIFO1.	

**Notes:**

- Disable transmission/reception first and then execute a FIFO1 reset.
- Execute the reset after setting the transmission FIFO interrupt enable bit to "0".
- The valid data count of the FBYTE1 register is set to 0.
- The TDR register and RDR register are not initialized.

**[bit1] FE2: FIFO2 operation enable bit**

This bit enables or disables the operation of FIFO2.

- Set this bit to "1" when using FIFO2.
- When FIFO2 is specified as the transmission FIFO and "1" is written to this bit, transmission starts immediately if FIFO2 contains data and LIN interface (v2.1) transmission is enabled (SCR:TXE = 1). At this time, to set the TIE and TBIE bits to "1", first set them to "0" and write "1" to this bit, and then set the TIE and TBIE bits to "1".
- A reception error causes this bit to be cleared to "0" if the FIFO is selected as the reception FIFO by the FSEL bit. After that, this bit cannot be set to "1" unless the reception error is cleared.
- To use FIFO2 as the transmission or reception FIFO, set this bit to "1" or "0" when the transmission or reception buffer is empty ((SSR:TDRE=1) or (SSR:RDRF=0)), respectively.
- To use FIFO2 as the reception FIFO, first disable reception (SCR:RXE=0) and then set this bit to "0" when the reception buffer is empty (SSR:RDRF=0) and the reception FIFO does not contain valid data (FBYTE2=0).
- To use FIFO2 as the reception FIFO, first disable reception (SCR:RXE=0) and then set this bit to "1" when the reception buffer is empty (SSR:RDRF=0).
- Disabling FIFO2 does not change the state of FIFO2.
- This bit is reset when the FCR0C:FE2C bit in the clear register is set to "1".
- This bit is set when the FCR0S:FE2S bit in the set register is set to "1".

Value	Description
0	Disable FIFO2 operation.
1	Enable FIFO2 operation.

**[bit0] FE1: FIFO1 operation enable bit**

This bit enables or disables the operation of FIFO1.

- Set this bit to "1" when using FIFO1.
- When FIFO1 is specified as the transmission FIFO and "1" is written to this bit, transmission starts immediately if FIFO1 contains data and LIN interface (v2.1) transmission is enabled (SCR:TXE=1). At this time, to set the TIE and TBIE bits to "1", first set them to "0" and write "1" to this bit, and then set the TIE and TBIE bits to "1".
- A reception error causes this bit to be cleared to "0" if the FIFO is selected as the reception FIFO by the FSEL bit. After that, this bit cannot be set to "1" unless the reception error is cleared.
- To use FIFO1 as the transmission or reception FIFO, set this bit to "1" or "0" when the transmission or reception buffer is empty ((SSR:TDRE=1) or (SSR:RDRF=0)), respectively.
- To use FIFO1 as the reception FIFO, first disable reception (SCR:RXE=0) and then set this bit to "0" when the reception buffer is empty (SSR:RDRF=0) and the reception FIFO does not contain valid data (FBYTE1=0).
- To use FIFO1 as the reception FIFO, first disable reception (SCR:RXE=0) and then set this bit to "1" when the reception buffer is empty (SSR:RDRF=0).
- Disabling FIFO1 does not change the state of FIFO1.
- This bit is reset when the FCR0C:FE1C bit in the clear register is set to "1".
- This bit is set when the FCR0S:FE1S bit in the set register is set to "1".

Value	Description
0	Disable FIFO1 operation.
1	Enable FIFO1 operation.





## 8.20. FIFO Byte Register (FBYTE)

The FIFO byte register (FBYTE) indicates the valid data count in the FIFO. This register also specifies whether a reception interrupt is generated when the predefined amount of data is received by the reception FIFO.

BITS_OFFSET	15	14	13	12	11	10	9	8
BITS_NAME	FBYTE2							
ACCESS_TYPE	R,W							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

BITS_OFFSET	7	6	5	4	3	2	1	0
BITS_NAME	FBYTE1							
ACCESS_TYPE	R,W							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

**[bit15:8] FBYTE2 [7:0]: FIFO2 data number indication bits**

**[bit7:0] FBYTE1 [7:0]: FIFO1 data number indication bits**

The FBYTE register indicates the valid data count for the FIFO. The table below shows the setting conditions for different FCR1:FSEL bit values.

Value	FIFO Selection	Data Number Indication
0	FIFO2: Reception FIFO, FIFO1: Transmission FIFO	FIFO2:FBYTE2, FIFO1:FBYTE1
1	FIFO2: Transmission FIFO, FIFO1: Reception FIFO	FIFO2:FBYTE2, FIFO1:FBYTE1

- The initial transfer count value for the FBYTE register is 0x08. Therefore, set 0x00 as the transfer count of the FBYTE to the transmission FIFO selected by the FSEL.
- Set the data count, on which the generation of the reception interrupt flag is based, as the transfer count of the FBYTE for the reception FIFO. When the transferred data count and the data count indication in the FBYTE register coincide, the interrupt flag (RDRF) is set to "1".
- When both of the following conditions are satisfied, the continuation of the reception idle status for 8 baud rate clocks or longer sets the interrupt flag (RDRF) to "1".
  - Reception FIFO idle detection enable bit (FRIIE) is "1".
  - The number of data items in the reception FIFO does not reach the transfer count.
- During an 8-clock count, the counter is reset to 0 when RDR is read, and the system starts counting the 8 clocks again. The counter is reset to 0 when the reception FIFO is disabled. If the reception FIFO is enabled when data remains in the reception FIFO, counting restarts.

**FBYTE2, FBYTE1: FIFO2 Data Count Indication Bit, FIFO1 Data Count Indication Bit**

bit	Description
Write	Set the transfer count.
Read	Read valid data count.

Read (valid data count)

Transmission: Data count that was written but not sent to the transmission FIFO

Reception: Data count that has been received but not read by the reception FIFO

Write (transfer count)

Transmission: Set 0x00.

Reception: Set the data count that triggers reception interrupts.

**Table 8-2 Data Count Stored in FIFO**

FIFO Capacity	Max FBYTE Count	Data Count that Can Be Stored
16 bytes	16	16
32 bytes	32	32
64 bytes	64	64
128 bytes	128	128

**Notes:**

- Set 0x00 to FBYTE for the transmission FIFO.
- Set "1" or a larger value in FBYTE for the reception FIFO.
- Change the value after disabling the reception operation.
- Any setting that would exceed the FIFO capacity is prohibited.
- Set the FIFO byte register (FBYTE) after setting the FIFO selection bit (FCR1:FSEL).
- The FIFO selection bit (FCR1:FSEL) and the FIFO byte register (FBYTE) cannot be set at the same time.
- As the FIFO data count for transmission, the transmission data count that has been written, minus 1, is displayed as the valid data count. This is because an attempt to write transmission data to the TDR register stores the data in the transmission FIFO if the TDR register contains data that has yet to be transmitted. When the data in the TDR register is transmitted, the data in the transmission FIFO that is yet to be transmitted is transferred to the TDR register.
- The FIFO data count for reception represents the data count received by the reception FIFO but which has not yet been read. The data count does not include the data being received by the RDR register.



## 8.21. Transmission FIFO Interrupt Control Register (FTICR)

The transmission FIFO interrupt control register (FTICR) sets the condition for interrupts triggered by the valid data count of the FIFO transmission.

BITS_OFFSET	15	14	13	12	11	10	9	8
BITS_NAME	FTICR2							
ACCESS_TYPE	R,W							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

BITS_OFFSET	7	6	5	4	3	2	1	0
BITS_NAME	FTICR1							
ACCESS_TYPE	R,W							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

[bit15:8] FTICR2[7:0]: FIFO2 data number indication bits

[bit7:0] FTICR1[7:0]: FIFO1 data number indication bits

The FTICR register sets the trigger level for interrupts by the valid transmission data number (remaining amount) in the transmission FIFO. The table below shows the setting by FCR1:FSEL bit.

Value	Description	
0	FIFO1	FTICR1
1	FIFO2	FTICR2

- The initial value of the valid data count for triggering interrupts of the FTICR register is 0x00.
- Set the number of data items that generate a transmission interrupt to the FTICR of the transmission FIFO.
- Set FTICR so that it satisfies "FTICR ≤ FIFO capacity - 2".
- The read value indicates the valid data count of the FIFO.
- Transmission FIFO: The data count written in the transmission FIFO that has not yet been transmitted.
- Reception FIFO: The data count received by the reception FIFO that has not yet been read.

FTICR2, FTICR1: FIFO2 Data Count Indication Bit, FIFO1 Data Count Indication Bit

bit	Description
Write	Set the number of valid data items that generates an interrupt.
Read	Read valid data count.

**Notes:**

- Any setting that would exceed the FIFO capacity is prohibited.
- The setting value cannot be read.
- As the FIFO data count for transmission, the transmission data count that has been written, minus 1, is displayed as the valid data count. This is because an attempt to write transmission data to the TDR register stores the data in the transmission FIFO if the TDR register contains data that has yet to be transmitted. When the data in the TDR register is transmitted, the data in the transmission FIFO that is yet to be transmitted is transferred to the TDR register.
- The FIFO data count for reception represents the data count received by the reception FIFO but which has not yet been read. The data count does not include the data being received by the RDR register.



## 8.22. Serial Control Clear Register (SCRC)

The serial control clear register (SCRC) can clear a bit in the serial control register (SCR).

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved	MSC	Reserved	RIEC	TIEC	TBIEC	RXEC	TXEC
ACCESS_TYPE	R0,W0	R0,W	R0,W0	R0,W	R0,W	R0,W	R0,W	R0,W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

\* Lower byte [7:0] of this register is serial mode clear register (SMRC).

### [bit15] Reserved: Reserved bit

### [bit14] MSC: Clearing the master/slave function selection bit.

Writing "1" to this bit resets the SCR:MS to "0".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

### [bit13] Reserved: Reserved bit

### [bit12] RIEC: Clearing the reception interrupt enable bit.

Writing "1" to this bit resets the SCR:RIE to "0".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

### [bit11] TIEC: Clearing the transmission interrupt enable bit

Writing "1" to this bit resets the SCR:TIE to "0".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

### [bit10] TBIEC: Clearing the transmission bus idle interrupt enable bit

Writing "1" to this bit resets the SCR:TBIE to "0".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

### [bit9] RXEC: Clearing the reception operation enable bit

Writing "1" to this bit resets the SCR:RXE to "0".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

**[bit8] TXEC: Clearing the transmission operation enable bit**

Writing "1" to this bit resets the SCR:TXE to "0".

Writing "0" to this bit is invalid.

"0" is always read from this bit.



### 8.23. Serial Mode Clear Register (SMRC)

The serial mode clear register (SMRC) can clear a bit in the serial mode register (SMR).

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved				SBLC	Reserved		SOEC
ACCESS_TYPE	R0, W0				R0,W	R0,W0		R0,W
PROT_TYPE	-							
INITIAL_VALUE	0000				0	00		0

#### [bit7:4] Reserved: Reserved bits

#### [bit3] SBLC: Clearing the stop bit length selection bit

Writing "1" to this bit resets the SMR:SBL to "0".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

#### [bit2:1] Reserved: Reserved bits

#### [bit0] SOEC: Clearing the serial data output enable bit

Writing "1" to this bit resets the SMR:SOE to "0".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

## 8.24. Serial Status Clear Register (SSRC)

The serial status clear register (SSRC) can clear a bit in the serial status register (SSR).

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved		LBDC	Reserved				
ACCESS_TYPE	R0,W0		R0,W	R0,W0				
PROT_TYPE	-							
INITIAL_VALUE	00		0	00000				

\* Lower byte [7:0] of this register is extended communication control clear register (SMRC).

**[bit15:14] Reserved: Reserved bits**

**[bit13] LBDC: Clearing the LIN Break Field detection flag bit**

Writing "1" to this bit resets the SSR:LBD to "0".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

**[bit12:8] Reserved: Reserved bits**





## 8.25. Extended Communication Control Clear Register (ESCRC)

The extended communication control clear register (ESCRC) can be used to clear a bit in the extended communication control register (ESCR).

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved	ESBLC	Reserved	LBIEC	Reserved			
ACCESS_TYPE	R0,W0	R0,W	R0,W0	R0,W	R0,W0			
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0000			

**[bit7] Reserved: Reserved bit**

**[bit6] ESBLC: Clearing the extended stop bit length selection bit**

Writing "1" to this bit resets the ESCR:ESBL to "0".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

**[bit5] Reserved: Reserved bit**

**[bit4] LBIEC: Clearing the LIN Break Field detection interrupt enable bit**

Writing "1" to this bit resets the ESCR:LBIE to "0".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

**[bit3:0] Reserved: Reserved bits**

## 8.26. Serial Auxiliary Control Status Clear Register (SACSRC)

The serial auxiliary control status clear register (SACSRC) can clear the bits in the serial auxiliary control status register (SACSR).

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	STSTC	Reserved	SFDC	SFDEC	AUTEC	Reserved		TINTC
ACCESS_TYPE	R0,W	R0,W0	R0,W	R0,W	R0,W	R0,W0		R0,W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	00		0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	TINTEC	Reserved						TMREC
ACCESS_TYPE	R0,W	R0,W0						R0,W
PROT_TYPE	-							
INITIAL_VALUE	0	000000						0

### [bit15] STSTC: Clearing the serial test bit

Writing "1" to this bit resets SACSR:STST to "0".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

### [bit14] Reserved: Reserved bit

### [bit13] SFDC: Clearing the Sync Field detection flag

Writing "1" to this bit resets SACSR:SFD to "0".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

### [bit12] SFDEC: Clearing the Sync Field detection interrupt enable bit

Writing "1" to this bit resets SACSR:SFDE to "0".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

### [bit11] AUTEC: Clearing the auto baud rate adjustment bit

Writing "1" to this bit resets SACSR:AUTE to "0".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

### [bit10:9] Reserved: Reserved bits

**[bit8] TINTC: Clearing the timer interrupt flag**

Writing "1" to this bit resets SACS: TINT to "0".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

**[bit7] TINTE: Clearing the timer interrupt enable bit**

Writing "1" to this bit resets SACS: TINTE to "0".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

**[bit6:1] Reserved: Reserved bits****[bit0] TMREC: Clearing the serial timer enable bit**

Writing "1" to this bit resets SACS: TMRE to "0".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

## 8.27. LIN Assist Mode Status Clear Register (LAMSRC)

The LIN assist mode status clear register (LAMSRC) can be used to clear a bit in the LIN assist mode status register (LAMSR).

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved					LCSCC	Reserved	LAHCC
ACCESS_TYPE	R0,W0					R0,W	R0,W0	R0,W
PROT_TYPE	-							
INITIAL_VALUE	00000					0	0	0

\* Lower byte [7:0] of this register is LIN assist mode control clear register (LAMCRC).

**[bit15:11] Reserved: Reserved bits**

### **[bit10] LCSCC: Clearing the LIN checksum calculation completion flag**

Writing "1" to this bit resets LAMSR:LCSC to "0".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

**[bit9] Reserved: Reserved bit**

### **[bit8] LAHCC: Clearing the LIN auto header completion flag**

Writing "1" to this bit resets the LAMSR:LAHC to "0".

Writing "0" to this bit is invalid.

"0" is always read from this bit.



## 8.28. LIN Assist Mode Control Clear Register (LAMCRC)

The LIN assist mode control clear register (LAMCRC) can be used to clear a bit in the LIN assist mode control register (LAMCR).

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved					LCSTYPC	LIDENC	LAMENC
ACCESS_TYPE	R0,W0					R0,W	R0,W	R0,W
PROT_TYPE	-							
INITIAL_VALUE	00000					0	0	0

**[bit7:3] Reserved: Reserved bits**

### **[bit2] LCSTYPC: Clearing the LIN checksum type selection bit**

Writing "1" to this bit resets the LAMCR:LCSTYP to "0".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

### **[bit1] LIDENC: Clearing the LIN ID register use enable bit**

Writing "1" to this bit resets LAMCR:LIDEN to "0".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

### **[bit0] LAMENC: Clearing the LIN assist mode processing enable bit**

Writing "1" to this bit resets the LAMCR:LAMEN to "0".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

## 8.29. LIN Assist Mode Interrupt Enable Clear Register (LAMIERC)

The LIN assist mode interrupt enable clear register (LAMIERC) can be used to clear a bit in the LIN assist mode interrupt enable register (LAMIER).

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved	LCSERIEC	LPTERIEC	LSFERIEC	LBSERIEC	LCSCIEC	Reserved	LAHCIEC
ACCESS_TYPE	R0, W0	R0,W	R0,W	R0,W	R0,W	R0,W	R0, W0	R0,W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved							
ACCESS_TYPE	R0,W0							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

**[bit15] Reserved: Reserved bit**

**[bit14] LCSERIEC: Clearing the LIN checksum error interrupt enable bit**

Writing "1" to this bit resets LAMIER:LCSERIE to "0".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

**[bit13] LPTERIEC: Clearing the LIN ID parity error interrupt enable bit**

Writing "1" to this bit resets the LAMIER:LPTERIE to "0".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

**[bit12] LSFERIEC: Clearing the LIN Sync Data error interrupt enable bit**

Writing "1" to this bit resets the LAMIER:LSFERIE to "0".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

**[bit11] LBSEIEC: Clearing the LIN bus error interrupt enable bit**

Writing "1" to this bit resets LAMIER:LBSEIE to "0".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

**[bit10] LCSCIEC: Clearing the LIN checksum calculation completion interrupt enable bit**

Writing "1" to this bit resets LAMIER:LCSCIE to "0".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

**[bit9] Reserved: Reserved bit****[bit8] LAHCIEC: Clearing the LIN auto header completion interrupt enable bit**

Writing "1" to this bit resets LAMIER:LAHCIE to "0".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

**[bit7:0] Reserved: Reserved bits**

### 8.30. LIN Assist Mode Error Status Clear Register (LAMESRC)

The LIN assist mode error status clear register (LAMESRC) can be used to clear a bit in the LIN assist mode error status register (LAMESR).

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved	LCSERC	LPTEC	LSFERC	LBSERC	Reserved		
ACCESS_TYPE	R0, W0	R0,W	R0,W	R0,W	R0,W	R0,W0		
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	000		

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved							
ACCESS_TYPE	R0,W0							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

**[bit15] Reserved: Reserved bit**

**[bit14] LCSERC: Clearing the LIN checksum error flag**

Writing "1" to this bit resets LAMESR:LCSER to "0".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

**[bit13] LPTEC: Clearing the LIN ID parity error flag**

Writing "1" to this bit resets LAMESR:LPTEC to "0".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

**[bit12] LSFERC: Clearing the LIN Sync Data error flag**

Writing "1" to this bit resets LAMESR:LSFER to "0".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

**[bit11] LBSERC: Clearing the LIN bus error flag**

Writing "1" to this bit resets LAMESR:LBSER to "0".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

**[bit10:0] Reserved: Reserved bits**





### 8.31. FIFO Control Clear Register 1 (FCR1C)

FIFO control clear register 1 (FCR1C) can clear a bit in FIFO control register 1 (FCR1).

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved			FLSTEC	FRIIEC	FDRQC	FTIEC	FSEL
ACCESS_TYPE	R0,W0			R0,W	R0,W	R0,W	R0,W	R0,W
PROT_TYPE	-							
INITIAL_VALUE	000			0	0	0	0	0

**[bit15:13] Reserved: Reserved bits**

**[bit12] FLSTEC: Clearing the retransmission data lost detection enable bit**

Writing "1" to this bit resets FCR1:FLSTE to "0".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

**[bit11] FRIIEC: Clearing the reception FIFO idle detection enable bit**

Writing "1" to this bit resets FCR1:FRIIE to "0".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

**[bit10] FDRQC: Clearing the transmission FIFO data request bit**

Writing "1" to this bit resets FCR1:FDRQ to "0".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

**[bit9] FTIEC: Clearing the transmission FIFO interrupt enable bit**

Writing "1" to this bit resets FCR1:FTIE to "0".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

**[bit8] FSEL: Clearing the FIFO selection bit**

Writing "1" to this bit resets FCR1:FSEL to "0".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

### 8.32. FIFO Control Clear Register 0 (FCR0C)

FIFO control clear register 0 (FCR0C) can clear a bit in FIFO control register 0 (FCR0).

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved						FE2C	FE1C
ACCESS_TYPE	R0,W0						R0,W	R0,W
PROT_TYPE	-							
INITIAL_VALUE	000000						0	0

**[bit7:2] Reserved: Reserved bits**

#### **[bit1] FE2C: Clearing the FIFO2 operation enable bit**

Writing "1" to this bit resets FCR0:FE2 to "0".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

#### **[bit0] FE1C: Clearing the FIFO1 operation enable bit**

Writing "1" to this bit resets FCR0:FE1 to "0".

Writing "0" to this bit is invalid.

"0" is always read from this bit.



### 8.33. Serial Control Set Register (SCRS)

The serial control set register (SCRS) can set a bit in the serial control register (SCR).

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	UPCLS	MSS	LBRS	RIES	TIES	TBIES	RXES	TXES
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

\* Lower byte [7:0] of this register is serial mode set register (SMRS).

#### [bit15] UPCLS: Setting the programmable clear bit

Writing "1" to this bit sets SCR:UPCL to "1".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

#### [bit14] MSS: Setting the master/slave function selection bit

Writing "1" to this bit sets SCR:MS to "1".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

#### [bit13] LBRS: Setting the LIN Break Field setting bit (effective only for the master operation)

Writing "1" to this bit sets SCR:LBR to "1".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

#### [bit12] RIES: Setting the reception interrupt enable bit

Writing "1" to this bit sets SCR:RIE to "1".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

#### [bit11] TIES: Setting the transmission interrupt enable bit

Writing "1" to this bit sets SCR:TIE to "1".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

#### [bit10] TBIES: Setting the transmission bus idle interrupt enable bit

Writing "1" to this bit sets SCR:TBIE to "1".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

**[bit9] RXES: Setting the reception operation enable bit**

Writing "1" to this bit sets the SCR:RXE to "1".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

**[bit8] TXES: Setting the transmission operation enable bit**

Writing "1" to this bit sets SCR:TXE to "1".

Writing "0" to this bit is invalid.

"0" is always read from this bit.



### 8.34. Serial Mode Set Register (SMRS)

The serial mode set register (SMRS) can set a bit in the serial mode register (SMR).

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved				SBLS	Reserved		SOES
ACCESS_TYPE	R0,W0				R0,W	R0,W0		R0,W
PROT_TYPE	-							
INITIAL_VALUE	0000				0	00		0

**[bit7:5] Reserved: Reserved bits**

**[bit4] WUCRS: Setting the wake up control bit**

Writing "1" to this bit sets SMR:WUCR to "1".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

**[bit3] SBLS: Setting the stop bit length selection bit**

Writing "1" to this bit sets the SMR:SBL to "1".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

**[bit2:1] Reserved: Reserved bits**

**[bit0] SOES: Setting the serial data output enable bit**

Writing "1" to this bit sets SMR:SOE to "1".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

### 8.35. Serial Status Set Register (SSRS)

The serial status set register (SSRS) can set a bit in the serial status register (SSR).

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	RECS	Reserved						
ACCESS_TYPE	R0,W	R0,W0						
PROT_TYPE	-							
INITIAL_VALUE	0	0000000						

\* Lower byte [7:0] of this register is extended communication control set register (ESCRS).

#### [bit15] RECS: Setting the reception error flag clear bit

Writing "1" to this bit sets SSR:REC to "1".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

#### [bit14:8] Reserved: Reserved bits



### 8.36. Extended Communication Control Set Register (ESCRS)

The extended communication control set register (ESCRS) can be used to set a bit in the extended communication control register (ESCR).

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved	ESBLS	Reserved	LBIES	Reserved			
ACCESS_TYPE	R0,W0	R0,W	R0,W0	R0,W	R0,W0			
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0000			

**[bit7] Reserved: Reserved bit**

**[bit6] ESBLS: Setting the extended stop bit length selection bit**

Writing "1" to this bit sets the ESCR:ESBL to "1".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

**[bit5] Reserved: Reserved bit**

**[bit4] LBIES: Setting the LIN Break Field detection interrupt enable bit**

Writing "1" to this bit sets the ESCR:LBIE to "1".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

**[bit3:0] Reserved: Reserved bits**

### 8.37. Serial Auxiliary Control Status Set Register (SACSR)

The serial auxiliary control status set register (SACSR) can set the bits in the serial auxiliary control status register (SACSR).

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	STSTS	Reserved		SFDES	AUTES	Reserved		
ACCESS_TYPE	R0,W	R0,W0		R0,W	R0,W	R0,W0		
PROT_TYPE	-							
INITIAL_VALUE	0	00		0	0	000		

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	TINTES							TMRES
ACCESS_TYPE	R0,W	R0,W0						R0,W
PROT_TYPE	-							
INITIAL_VALUE	0	000000						0

#### [bit15] STSTS: Setting the serial test bit

Writing "1" to this bit sets SACSR:STST to "1".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

#### [bit14:13] Reserved: Reserved bits

#### [bit12] SFDES: Setting the Sync Field detection interrupt enable bit

Writing "1" to this bit sets the SACSR:SFDE to "1".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

#### [bit11] AUTES: Setting the auto baud rate adjustment bit

Writing "1" to this bit sets SACSR:AUTE to "1".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

#### [bit10:8] Reserved: Reserved bits

#### [bit7] TINTES: Setting the timer interrupt enable bit

Writing "1" to this bit sets SACSR:TINTE to "1".

Writing "0" to this bit is invalid.

"0" is always read from this bit.





**[bit6] Reserved: Reserved bit**

**[bit5] TRGES: Setting the external trigger enable bit**

Writing "1" to this bit sets SACSR:TRGE to "1".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

**[bit4:1] Reserved: Reserved bits**

**[bit0] TMRES: Setting the serial timer enable bit**

Writing "1" to this bit sets SACSR:TMRE to "1".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

### 8.38. LIN Assist Mode Control Set Register (LAMCRS)

The LIN assist mode control set register (LAMCRS) can be used to set a bit in the LIN assist mode control register (LAMCR).

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved							
ACCESS_TYPE	R0,W0							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved				LTDRCLS	LCSTYPS	LIDENS	LAMENS
ACCESS_TYPE	R0,W0				R0,W	R0,W	R0,W	R0,W
PROT_TYPE	-							
INITIAL_VALUE	0000				0	0	0	0

#### [bit15:4] Reserved bits

#### [bit3] LTDRCLS: Setting the LIN transmission data register clear bit

Writing "1" to this bit sets LAMCR:LTDRCCL to "1".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

#### [bit2] LCSTYPS: Setting the LIN checksum type selection bit

Writing "1" to this bit sets LAMCR:LCSTYP to "1".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

#### [bit1] LIDENS: Setting the LIN ID register use enable bit

Writing "1" to this bit sets LAMCR:LIDEN to "1".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

#### [bit0] LAMENS: Setting the LIN assist mode processing enable bit

Writing "1" to this bit sets LAMCR:LAMEN to "1".

Writing "0" to this bit is invalid.

"0" is always read from this bit.



### 8.39. LIN Assist Mode Interrupt Enable Set Register (LAMIERS)

The LIN assist mode interrupt enable set register (LAMIERS) can be used to set a bit in the LIN assist mode interrupt enable register (LAMIER).

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved	LCSERIES	LPTERIES	LSFERIES	LBSERIES	LCSCIES	Reserved	LAHCIES
ACCESS_TYPE	R0,W0	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W0	R0,W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved							
ACCESS_TYPE	R0,W0							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

**[bit15] Reserved: Reserved bit**

**[bit14] LCSERS: Setting the LIN checksum error interrupt enable bit**

Writing "1" to this bit sets LAMIER:LCSERIE to "1".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

**[bit13] LPTERS: Setting the LIN ID parity error interrupt enable bit**

Writing "1" to this bit sets LAMIER:LPTERIE to "1".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

**[bit12] LSFERS: Setting the LIN Sync Data error interrupt enable bit**

Writing "1" to this bit sets LAMIER:LSFERIE to "1".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

**[bit11] LBSERS: Setting the LIN bus error interrupt enable bit**

Writing "1" to this bit sets LAMIER:LBSERIE to "1".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

**[bit10] LCSCIES: Setting the LIN checksum calculation completion interrupt enable bit**

Writing "1" to this bit sets LAMIER:LCSCIE to "1".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

**[bit9] Reserved: Reserved bit**

**[bit8] LAHCIES: Setting the LIN auto header completion interrupt enable bit**

Writing "1" to this bit sets LAMIER:LAHCIE to "1".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

**[bit7:0] Reserved: Reserved bits**



## 8.40. FIFO Control Set Register 1 (FCR1S)

FIFO control set register 1 (FCR1S) can set a bit in FIFO control register 1 (FCR1).

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved			FLSTES	FRIIES	Reserved	FTIES	FSELS
ACCESS_TYPE	R0,W0			R0,W	R0,W	R0, W0	R0,W	R0,W
PROT_TYPE	-							
INITIAL_VALUE	000			0	0	0	0	0

\* Lower byte [7:0] of this register is FIFO control set register 0 (FCRS0).

**[bit15:13] Reserved: Reserved bits**

**[bit12] FLSTES: Setting the retransmission data lost detection enable bit**

Writing "1" to this bit sets FCR1:FLSTE to "1".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

**[bit11] FRIIES: Setting the reception FIFO idle detection enable bit**

Writing "1" to this bit sets FCR1:FRIIE to "1".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

**[bit10] Reserved: Reserved bit**

**[bit9] FTIES: Setting the transmission FIFO interrupt enable bit**

Writing "1" to this bit sets FCR1:FTIE to "1".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

**[bit8] FSELS: Setting the FIFO selection bit**

Writing "1" to this bit sets FCR1:FSEL to "1".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

## 8.41. FIFO Control Set Register 0 (FCR0S)

FIFO control set register 0 (FCR0S) can set a bit in FIFO control register 0 (FCR0).

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved		FLDS	FSETS	FCL2S	FCL1S	FE2S	FE1S
ACCESS_TYPE	R0,W0		R0,W	R0,W	R0,W	R0,W	R0,W	R0,W
PROT_TYPE	-							
INITIAL_VALUE	00		0	0	0	0	0	0

**[bit7:6] Reserved: Reserved bits**

**[bit5] FLDS: Setting the FIFO pointer reload bit**

Writing "1" to this bit sets FCR0:FLD to "1".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

**[bit4] FSETS: Setting the FIFO pointer saving bit**

Writing "1" to this bit sets FCR0:FSET to "1".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

**[bit3] FCL2S: Setting the FIFO2 reset bit**

Writing "1" to this bit sets FCR0:FCL2 to "1".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

**[bit2] FCL1S: Setting the FIFO1 reset bit**

Writing "1" to this bit sets FCR0:FCL1 to "1".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

**[bit1] FE2S: Setting the FIFO2 operation enable bit**

Writing "1" to this bit sets FCR0:FE2 to "1".

Writing "0" to this bit is invalid.

"0" is always read from this bit.

**[bit0] FE1S: Setting the FIFO1 operation enable bit**

Writing "1" to this bit sets FCR0:FE1 to "1".

Writing "0" to this bit is invalid.

"0" is always read from this bit.



## 9. Precautions for Using

Precautions for using LIN are provided below.

### Notes on DMA Transfer

By using the interrupt factor generation of LIN, the DMA controller can be activated.

Before activating the DMA controller from LIN, make settings for the DMA controller. If DMA transfer is performed, only 1 is supported as the entire length (block count) of 1 block. (DMAi\_An:BC=0)

For the settings of the DMA controller and for the details, see "CHAPTER: DMA Controller".

## CHAPTER 38: Base Timer

This chapter explains the functions and operations of the base timer.

---

1. Overview
2. Configuration
3. Operation
4. 32-bit Mode Operation
5. Interrupts
6. Start of DMA Controller (DMAC)
7. Registers
8. Notes on Using
9. Description by Function Mode





## 1. Overview

Only one of the following timer functions can be selected for the base timer in the FMD2 to FMD0 bit settings in the timer control register (TMCR): reset mode, 16-bit PWM timer, 16-bit PPG timer, 16/32-bit reload timer, and 16/32-bit PWC timer. This section provides an overview of the various selectable timers.

### (1) Relationship between Mode Settings and Various Timer Functions

Table 1-1 Mode Settings and Various Timer Functions

FMD2 to FMD0 Bit Settings	Function
000	Reset mode
001	16-bit PWM timer
010	16-bit PPG timer
011	16/32-bit reload timer
100	16/32-bit PWC timer

### (2) Reset Mode

Reset mode is the state in which the base timer macros have been reset (to the initial values in each register). To use another timer function or switch the T32 bit setting, first enter this mode and then set the other timer function or the T32 bit. However, after a macro reset, it is possible to set a timer function and the T32 bit without entering this mode.

### (3) 16-bit PWM Timer

This timer consists of a 16-bit down counter, a 16-bit data register with a cycle setting buffer, a 16-bit compare register with a duty setting buffer, and a pin controller.

The registers with buffers store the cycle and duty data, enabling rewriting while the timer is operating.

The counter clock of the 16-bit down counter can be selected from eight types of internal clocks (internal clock divided by 1/4/16/128/256/512/1024/2048) and three types of external clocks (rising-edge, falling-edge, and both-edges detection).

One-shot mode and continuous mode can be selected. In one-shot mode, counting stops when an underflow occurs. In continuous mode, counting is repeated following a reload.

The start of the 16-bit PWM timer can be selected from a software trigger and three types of external events (rising-edge, falling-edge, and both-edges detection).

### (4) 16-bit PPG Timer

This timer consists of a 16-bit down counter, a 16-bit data register for the H width setting, a 16-bit data register for the L width setting, and a pin controller.

The count clock of the 16-bit down counter can be selected from eight types of internal clocks (internal clock divided by 1/4/16/128/256/512/1024/2048) and three types of external clocks (rising-edge, falling-edge, and both-edges detection).

One-shot mode and continuous mode can be selected. In one-shot mode, counting stops when an underflow occurs. In continuous mode, counting is repeated following a reload.

The start of the 16-bit PPG timer can be selected from a software trigger and three types of external events (rising-edge, falling-edge, and both-edges detection).

#### **(5)16/32-bit Reload Timer**

This timer consists of a 16-bit down counter, a 16-bit reload register, and a pin controller.

The count clock of the 16-bit down counter can be selected from eight types of internal clocks (internal clock divided by 1/4/16/128/256/512/1024/2048) and three types of external clocks (rising-edge, falling-edge, and both-edges detection).

One-shot mode and continuous mode can be selected. In one-shot mode, counting stops when an underflow occurs. In continuous mode, counting is repeated following a reload.

The start of the 16/32-bit reload timer can be selected from a software trigger and three types of external events (rising-edge, falling-edge, and both-edges detection).

#### **(6) 16/32-bit PWC Timer**

This timer consists of a 16-bit up counter, measurement input pins, and a control register.

With the input of external pulses, the timer measures the time between events.

The reference count clock can be selected from eight types of internal clocks (divided by 1/4/16/128/256/512/1024/2048).

Each measurement mode: H pulse width (rising to falling) / L pulse width (falling to rising)

Rising cycle (rising to rising) / falling cycle (falling to falling)

Edge-to-edge measurement (rising or falling to falling or rising)

An interrupt request can be generated at the measurement end time.

The measurement is 1-time only or continuous. Either can be selected.



2. Configuration

Figure 2-1 to Figure 2-4 are block diagrams of the base timer in each mode.

Figure 2-1 Block Diagram of the 16-bit PWM Timer

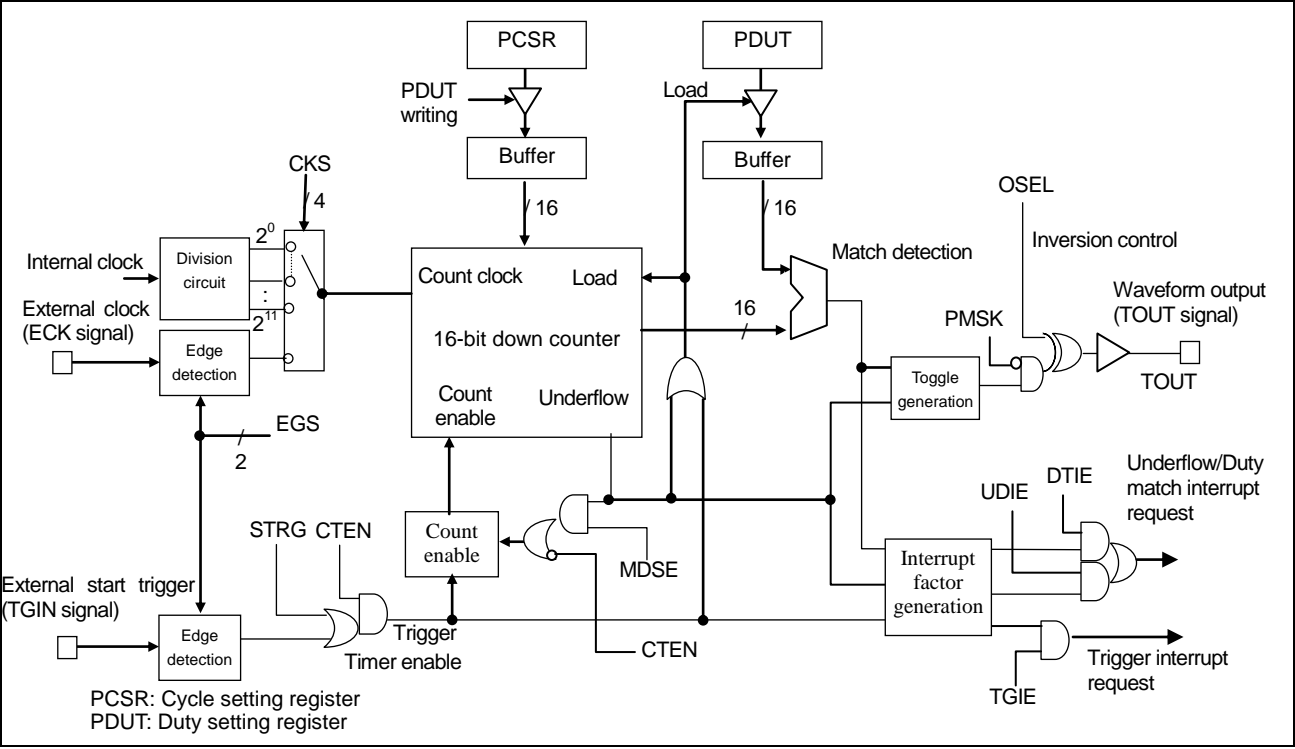


Figure 2-2 Block Diagram of the 16-bit PPG Timer

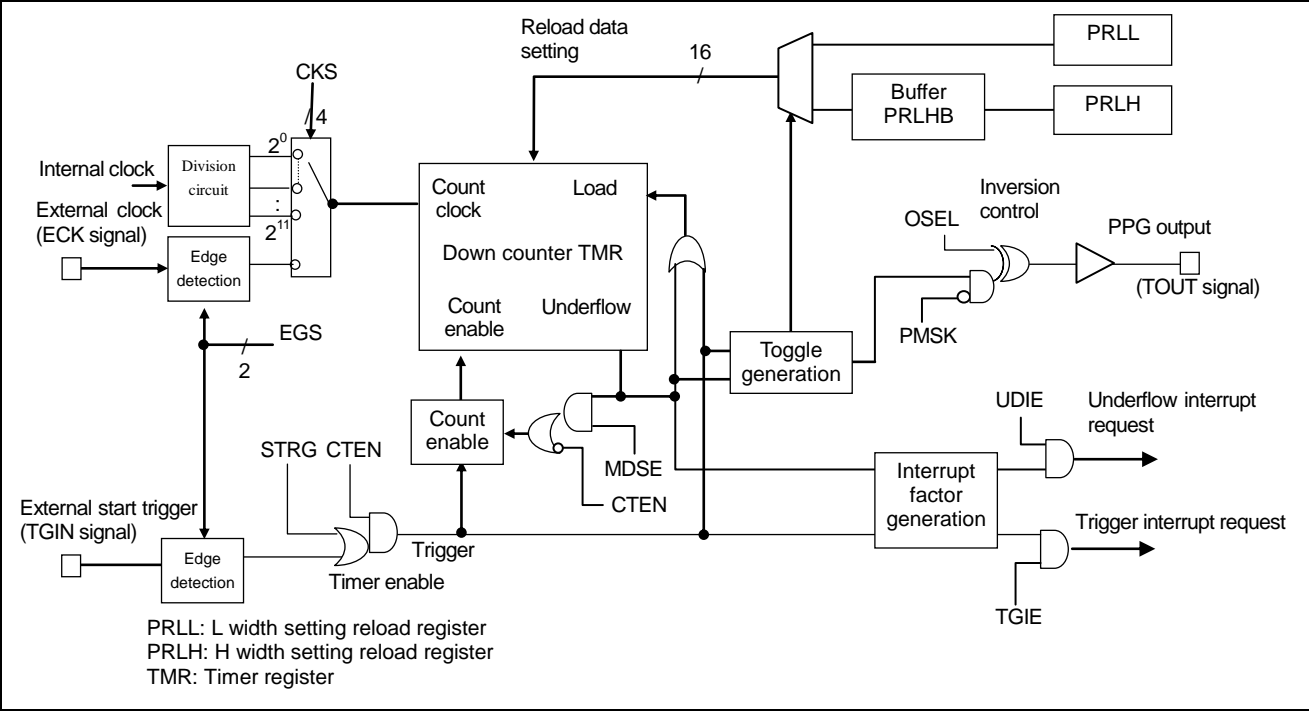


Figure 2-3 Block Diagram of the 16/32-bit Reload Timer (ch.1, ch.0)

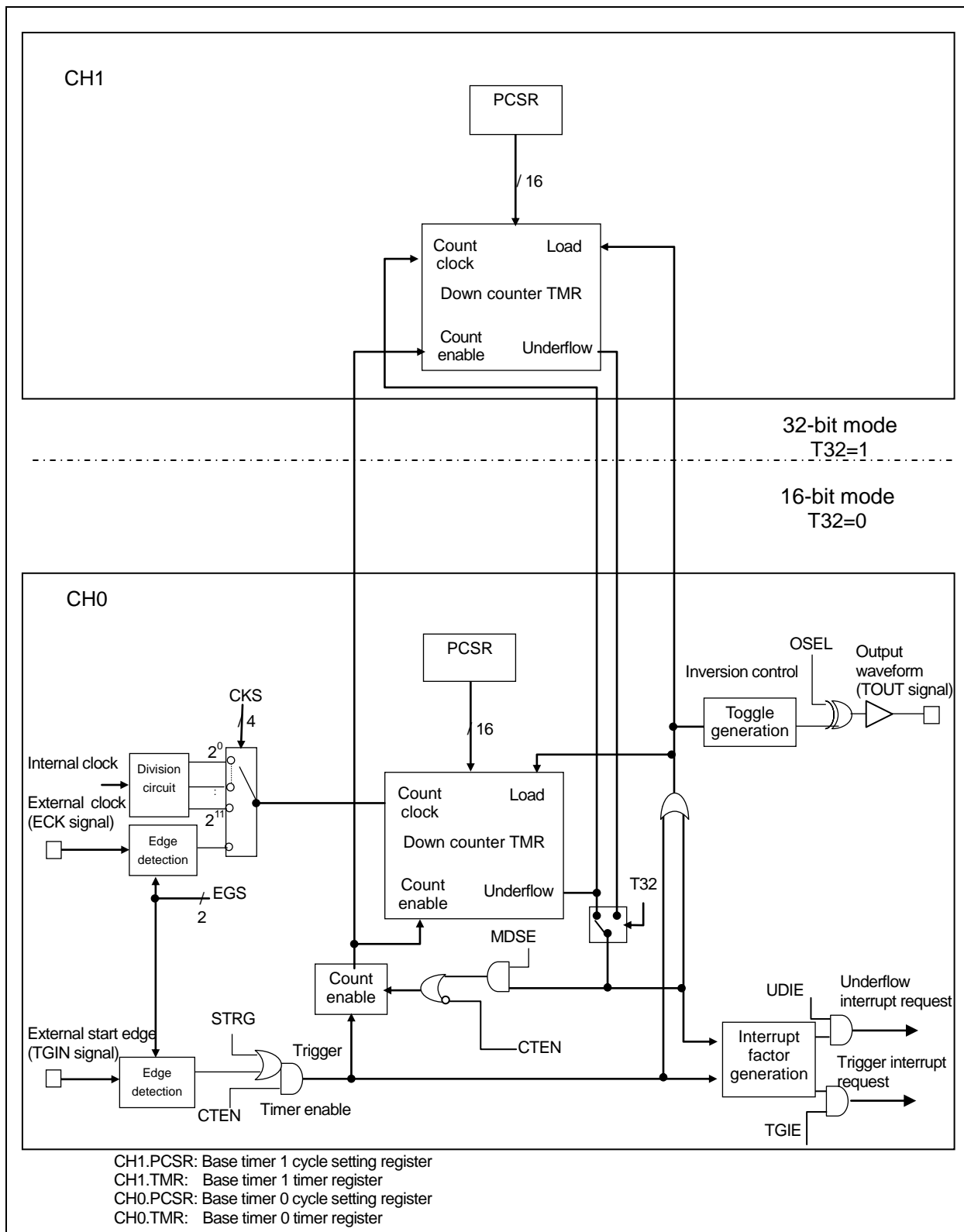
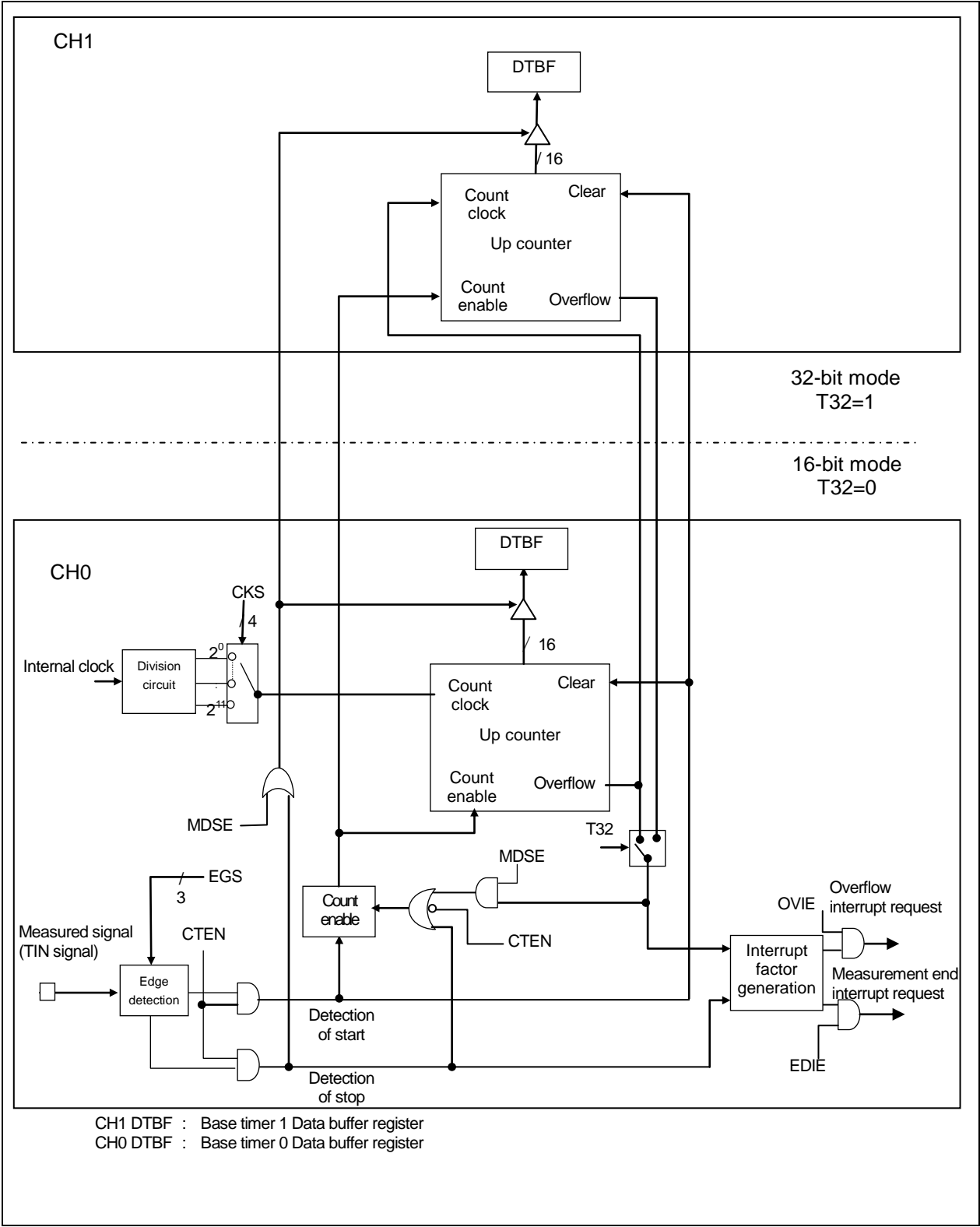




Figure 2-4 Block Diagram of the 16/32-bit PWC Timer (ch.1, ch.0)



### 3. Operation

This section explains the operations of the base timer.

#### Operations of Base Timer

##### a) Reset Mode

Reset mode is the state in which the base timer macros have been reset (to the initial values in each register). To use another timer function or switch the T32 bit setting, first enter this mode and then set the other timer function or the T32 bit. However, after a macro reset, it is possible to set a timer function and the T32 bit without entering this mode. If this mode is set for the even-numbered channel when 32-bit mode is set, the odd-numbered channel is reset at the same time, so reset mode need not be set for the odd-numbered channel.

##### b) 16-bit PWM Timer

The 16-bit PWM timer starts counting down from the set cycle value upon trigger activation. The first output at that time is the L level. If the 16-bit down counter matches the set value in the duty setting register, the output is inverted to the H level. Then, the output is inverted to the L level again when the counter underflows. Thus, this timer can generate a waveform with an arbitrary cycle and duty.

##### c) 16-bit PPG Timer

The 16-bit PPG timer starts counting down from the set value in the L width setting reload register upon trigger activation. The first output at that time is the L level. The output is inverted to the H level when the counter underflows. Subsequently, the counter starts counting down from the set value in the H width setting reload register. The output is inverted to the L level when the counter underflows. Thus, this timer can generate a waveform with an arbitrary L width and H width.

##### d) 16-bit Reload Timer

The 16-bit reload timer starts counting down from the set cycle value upon trigger activation. An interrupt flag is set to "1" when the 16-bit down counter underflows. The output level is either toggle output or pulse output. Toggle output is the output inverted for each underflow by the MDSE bit setting. Pulse output is the output of "H" due to the start of counting or "L" due to an underflow.

##### e) 32-bit Reload Timer

With the same basic operation as the 16-bit reload timer, this timer uses two channels, an even-numbered channel and an odd-numbered channel, to operate as a 32-bit reload timer. The even-numbered channel performs the lower 16-bit timer operations, and the odd-numbered channel performs the upper 16-bit timer operations. The interrupt controller and output waveform control conform to the settings of only the even-numbered channel. To set a cycle, write it first to the upper register (odd-numbered channel) and then to the lower register (even-numbered channel).

To read the timer value, read it first from the lower register (even-numbered channel) and then from the upper register (odd-numbered channel).

##### f) 16-bit PWC Timer

The PWC timer starts the 16-bit up counter upon input of the set measurement start edge. The timer stops the counter upon the detection of a measurement end edge. The count value at this time is stored as a pulse width in the data buffer register.



**g) 32-bit PWC Timer**

With the same basic operation as the 16-bit PWC timer, this timer uses two channels, an even-numbered channel and an odd-numbered channel, to operate as a 32-bit PWC timer. The even-numbered channel performs the lower 16-bit count operations, and the odd-numbered channel performs the upper 16-bit count operations. The interrupt controller conforms to the settings of only the even-numbered channel. To read a measurement value or count value, read it first from the lower register (even-numbered channel) and then from the upper register (odd-numbered channel).

## 4. 32-bit Mode Operation

The reload timer and PWC are capable of 32-bit mode operation using two channels. This section shows the basic functions/operations of the 32-bit mode function.

### (1) 32-bit Mode Function

This function realizes the operation of the 32-bit data reload timer or 32-bit data PWC timer by using two base timer channels. The value of the timer or counter in operation can also be read when the lower 16-bit timer or counter value of the even-numbered channel is read. This is because the upper 16-bit timer or counter value of the odd-numbered channel is also fetched at the time.

### (2) 32-bit Mode Settings

First, reset the state by setting the FMD2 to FMD0 bits in the TMCR register of the even-numbered channel to "000" for reset mode. Then, in the same way as in 16-bit mode, make reload timer or PWC timer selection and operation settings. To set 32-bit operation mode at this time, write "1" to the T32 bit in the TMCR register. Keep the T32 bit of the odd-numbered channel at "0". There is no need to set reset mode. For the reload timer, set the upper 16-bit reload value of the 32 bits in the cycle setting register of the odd-numbered channel. Then, set the lower 16-bit reload value in the cycle setting register of the even-numbered channel.

After the writing of the T32 bit, the change to 32-bit operation mode is reflected immediately, so change the settings for both channels in the count stopped state.

To change from 32-bit mode to 16-bit mode, set the FMD2 to FMD0 bits in the TMCR register of the even-numbered channel to "000" for reset mode. This resets the states of both the even-numbered and odd-numbered channels. As a result, the settings for 16-bit mode can be made for each of the channels.

### (3) 32-bit Mode Operation

After 32-bit mode is set, if the reload timer or PWC timer is started with even-numbered channel control, the even-numbered channel timer/counter performs the lower 16-bit operations. The odd-numbered channel timer/counter performs the upper 16-bit operations.

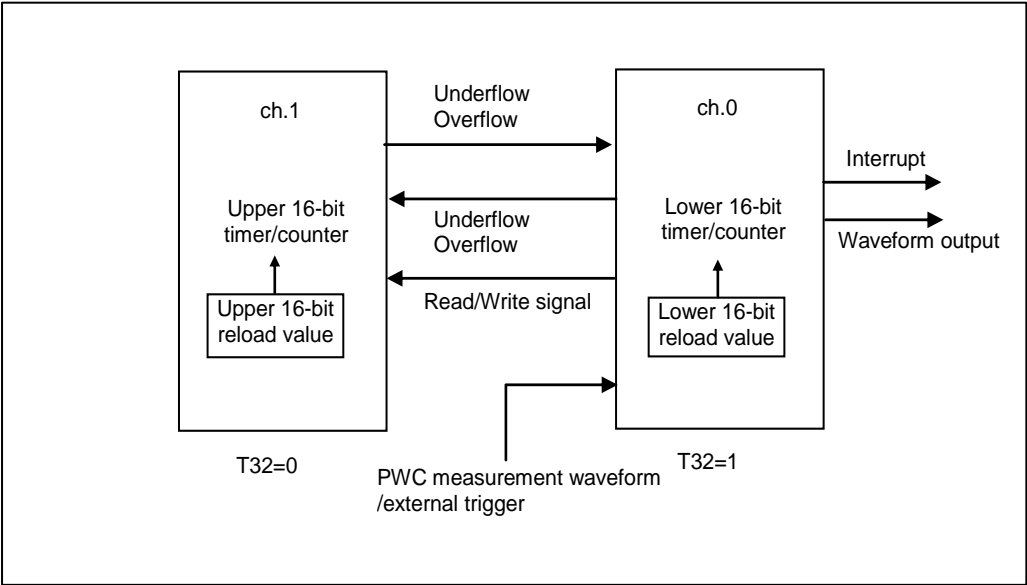
Operation in 32-bit mode conforms to the settings made for the even-numbered channel. Thus, the settings made for the odd-numbered channel (excluding those in the cycle setting register at the reload timer time) are ignored. Timer start, waveform output, and interrupt signals for the even-numbered channel are valid. (The odd-numbered channel is fixed at "L" and masked.)

Figure 4-1 shows the configuration of ch.0 and ch.1.





Figure 4-1 Configuration of 32-bit Mode Operation (for ch.0 and ch.1)



## 5. Interrupts

This section shows a list summarizing the interrupt request flags, interrupt enable bits, and interrupt factors in each function of the base timer.

### Interrupt Control Bits and Interrupt Factors of each Function

Table 5-1 shows the interrupt control bits and interrupt factors of each function.

**Table 5-1 Interrupt Controller Bits and Interrupt Factors in Each Mode**

	Status Control Register (STC)		
	Interrupt Request Flag Bit	Interrupt Request Enable Bit	Interrupt Factor
PWM Timer Function	UDIR : bit0	UDIE : bit4	Detection of underflow
	DTIR : bit1	DTIE : bit5	Detection of duty match
	TGIR : bit2	TGIE : bit6	Detection of timer activation trigger
PPG Timer Function	UDIR : bit0	UDIE : bit4	Detection of underflow
	TGIR : bit2	TGIE : bit6	Detection of timer activation trigger
Reload Timer Function	UDIR : bit0	UDIE : bit4	Detection of underflow
	TGIR : bit2	TGIE : bit6	Detection of timer activation trigger
PWC Timer Function	OVIR : bit0	OVIE : bit4	Detection of overflow
	EDIR : bit2	EDIE : bit6	Detection of measurement end

## 6. Start of DMA Controller (DMAC)

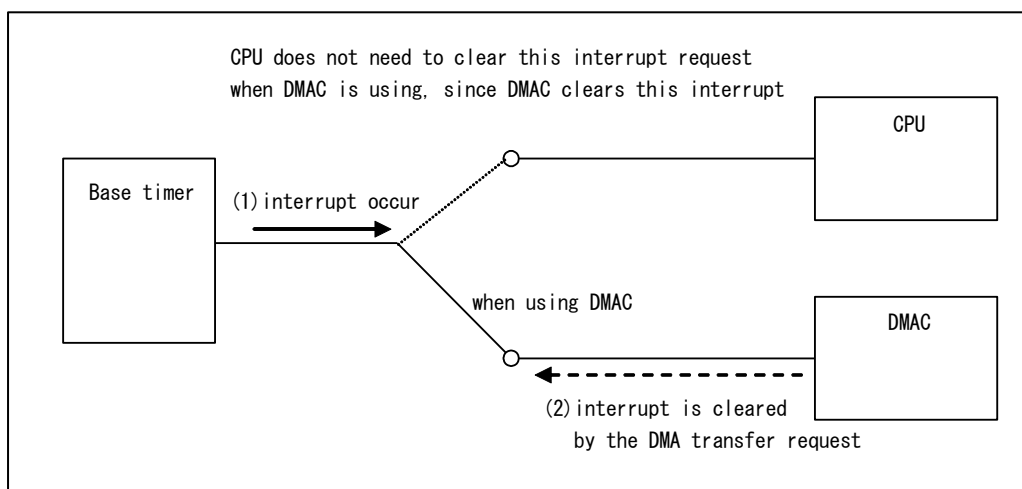
The generation of a base timer interrupt request can be used to start the DMAC.

### DMA Transfer Operation Using Base Timer Interrupt Factor

The generation of a base timer interrupt factor can be used to start the DMAC.

Figure 6-1 shows an overview of DMAC start with the base timer.

Figure 6-1 Overview of DMAC Start with the Base Timer



Configure the DMAC before starting it with the base timer. For details on DMAC settings, see "CHAPTER: DMA Controller".

## 7. Registers

This section lists the registers of each mode of the base timer.

All registers are prefixed by "BTxx\_". xx is the channel number (00 to 11).

**Table 7-1 List of Registers When the 16-bit PWM Timer is Selected**

Abbreviated Register Name	Register Name	See
TMCR	Timer Control Register	9.1.6
TMCR2	Timer Control Register 2	9.1.6
STC	Status Control Register	9.1.6
STCC	Status Control Clear Register	9.1.6
STCS	Status Control Set Register	9.1.6
PCSR	PWM Cycle Setting Register	9.1.7
PDUT	PWM Duty Setting Register	9.1.8
TMR	Timer Register	9.1.9

**Table 7-2 List of Registers When the 16-bit PPG Timer is Selected**

Abbreviated Register Name	Register Name	See
TMCR	Timer Control Register	9.2.6
TMCR2	Timer Control Register 2	9.2.6
STC	Status Control Register	9.2.6
STCC	Status Control Clear Register	9.2.6
STCS	Status Control Set Register	9.2.6
PRL	L Width Setting Reload Register	9.2.7
PRLH	H Width Setting Reload Register	9.2.8
TMR	Timer Register	9.2.9



**Table 7-3 List of Registers When the Reload Timer is Selected**

<b>Abbreviated Register Name</b>	<b>Register Name</b>	<b>See</b>
TMCR	Timer Control Register	9.3.3
TMCR2	Timer Control Register 2	9.3.3
STC	Status Control Register	9.3.3
STCC	Status Control Clear Register	9.3.3
STCS	Status Control Set Register	9.3.3
PCSR	Cycle Setting Register	9.3.4
TMR	Timer Register	9.3.5

**Table 7-4 List of Registers When the PWC Timer is Selected**

<b>Abbreviated Register Name</b>	<b>Register Name</b>	<b>See</b>
TMCR	Timer Control Register	9.4.2
TMCR2	Timer Control Register 2	9.4.2
STC	Status Control Register	9.4.2
STCC	Status Control Clear Register	9.4.2
STCS	Status Control Set Register	9.4.2
DTBF	Data Buffer Register	9.4.3

## 8. Notes on Using

This section explains precautions on use of the base timer.

### (1) Notes to Observe When Accessing Register

#### Status Control Register (STC) Access

- This register supports writing from the bit-band alias area. For details on the bit-band alias area, see "CHAPTER: Bit-band Unit".
- To clear a specific bit in this register, write "1" to the corresponding bit in the status control clear register (STCC).
- To set a specific bit in this register, write "1" to the corresponding bit in the status control set register (STCS).
- Direct writing to this register is possible only during writing to all bits.

### (2) Notes to Observe on Configuration Using Program, Common to Use of Timers

- Rewriting of the following bits in the TMCR2 register and TMCR register during operation is prohibited. Be sure that such rewriting is done either before the timer starts or after it stops.

[TMCR2 bit8], [TMCR bit14,13,12]	CKS3 to CKS0: Clock selection bits
[bit10,9,8]	EGS2, EGS1, EGS0: Measurement edge selection bits
[bit7]	T32: 32-bit timer selection bit (when the reload timer and PWC function are selected)
[bit6,5,4]	FMD2 to FMD0: Timer function selection bits
[bit2]	MDSE: Measurement mode (single/continuous) selection bit

- All the registers of the base timer are initialized when the FMD2 to FMD0 bits in the TMCR register are set to "000" for reset mode. For this reason, all the registers must be reconfigured.
- The settings for bits other than the FMD2 to FMD0 bits in the TMCR register are ignored and initialized when the FMD2 to FMD0 bits in the TMCR register are set to "000" for reset mode.

### (3) Notes on Using the 16-bit PWM/PPG/Reload Timer

- If the interrupt request flag set timing and clear timing overlap, the flag set has priority, and the clear operation is disabled.
- If the load timing and count timing overlap, the load operation has priority for the down counter.
- After configuring the timer function with the FMD2 to FMD0 bits in the TMCR register, set the cycle, duty, H width, and L width.
- If a restart is detected at the count end time in one-shot mode, the count value is reloaded. Then, the restart begins.



#### (4) Notes on Using the PWC Timer

- If "1" is written to the count start enable bit (CTEN), the counter is cleared. Thus, any data in the counter before the start is enabled is invalid.
- If the PWC mode setting (FMD2 to FMD0 = "100") and the measurement start setting (CTEN="1") are made from a system reset and reset mode at the same time, the resulting operation may depend on the state of the immediately preceding measurement signal.
- If a measurement start edge is detected at the same time that a restart is set in continuous measurement mode, the timer starts counting immediately from "0x0001".
- If a restart is performed after a count operation begins, operations such as the following may occur, depending on the timing.
  - For a restart at the same time as a measurement end edge in pulse-width single measurement mode, the restart is performed and the measurement start edge wait state begins, but the measurement end flag (EDIR) is set to "1".
  - For a restart at the same time as a measurement end edge in pulse-width continuous measurement mode, the restart is performed and the measurement start edge wait state begins, but the measurement end flag (EDIR) is set to "1" and the measurement results at that point are transferred to DTBF.

Note the operation of the flag when using the interrupt controller, etc. at the restart time during operation as described above.

## 9. Description by Function Mode

This section explains each function of the base timer.

### Functions of the Base Timer

1. PWM Timer Function
2. PPG Timer Function
3. Reload Timer Function
4. PWC Timer Function





## 9.1. PWM Timer Function

Only one of the following timer functions can be selected for the base timer in the FMD2 to FMD0 bit settings in the timer control register (TMCR): 16-bit PWM timer, 16-bit PPG timer, 16/32-bit reload timer, and 16/32-bit PWC timer. This section explains the timer function with the PWM setting.

1. 16-bit PWM Timer Operation
2. One-shot Operation
3. Interrupt Factors and Timing Chart
4. Output Waveform
5. PWM Timer Operation Flow
6. Timer Control Registers (TMCR, TMCR2), Status Control Register (STC), Status Control Clear Register (STCC), and Status Control Set Register (STCS)
7. PWM Duty Setting Register (PDUT)
8. Timer Register (TMR)

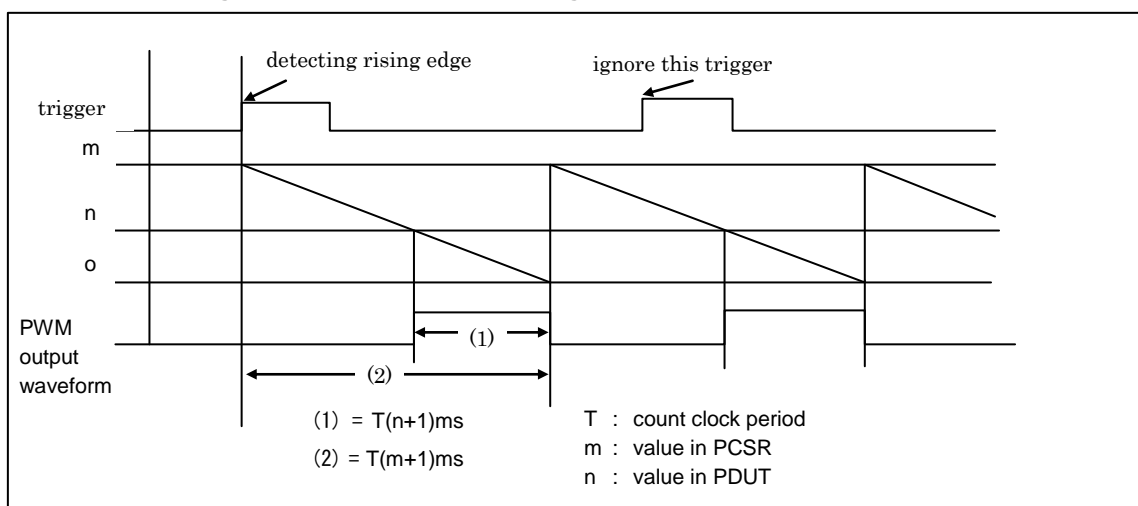
### 9.1.1. 16-bit PWM Timer Operation

In PWM operation, waveforms of the set cycle can be output either singly or continuously upon the detection of a trigger. The cycle of the output pulse can be controlled through PCSR value changes. The duty ratio can be controlled through PDUT value changes. After writing data to PCSR, be sure to write to PDUT.

#### Continuous Operation

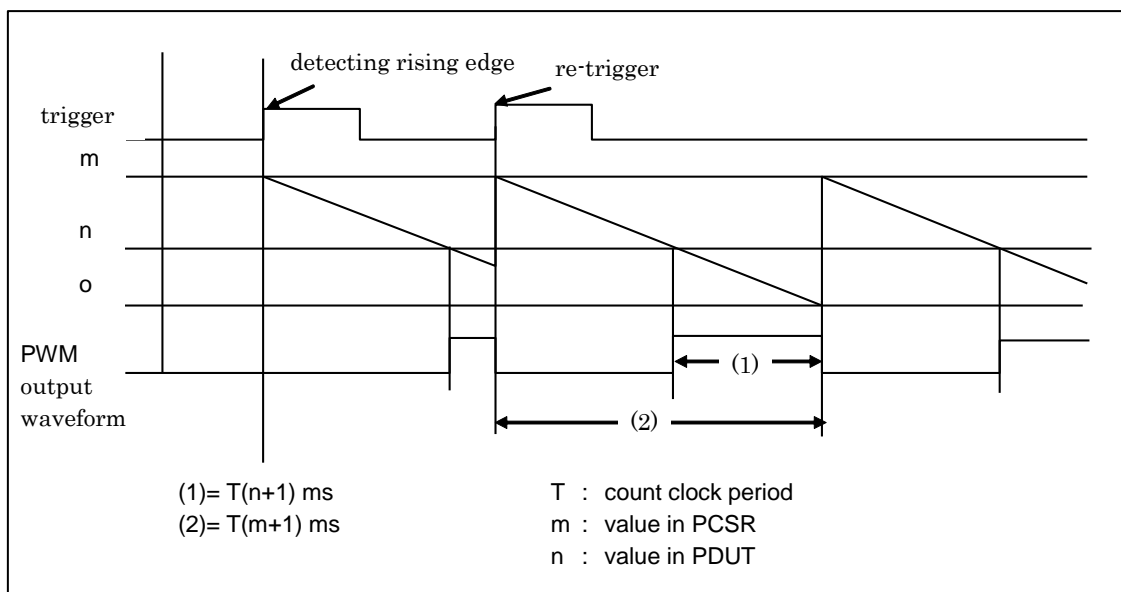
##### a) When Restart is Disabled (RTGEN=0)

Figure 9-1 PWM Operation Timing Chart (When Restart is Disabled)



##### b) When Restart is Enabled (RTGEN=1)

Figure 9-2 PWM Operation Timing Chart (When Restart is Enabled)



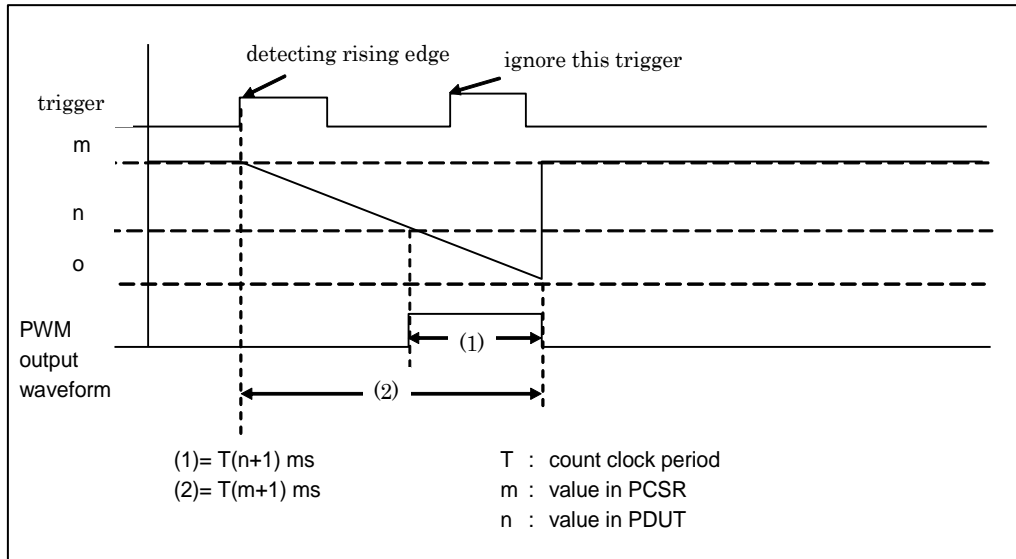
### 9.1.2. One-shot Operation

In one-shot operation, a trigger can cause the output of a single pulse of any width. If restart is enabled, the counter is reloaded when an edge is detected during operation.

#### One-shot Operation

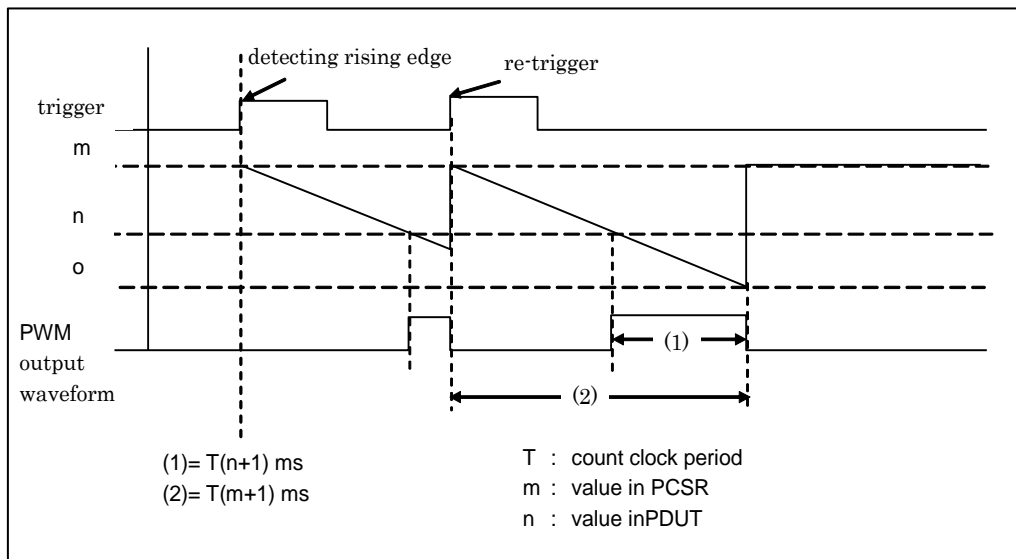
##### a) When Restart is Disabled (RTGEN=0)

Figure 9-3 One-shot Operation Timing Chart (Trigger Restart Disabled)



##### b) When Restart is Enabled (RTGEN=1)

Figure 9-4 One-shot Operation Timing Chart (Trigger Restart Enabled)



9.1.3. Interrupt Factors and Timing Chart

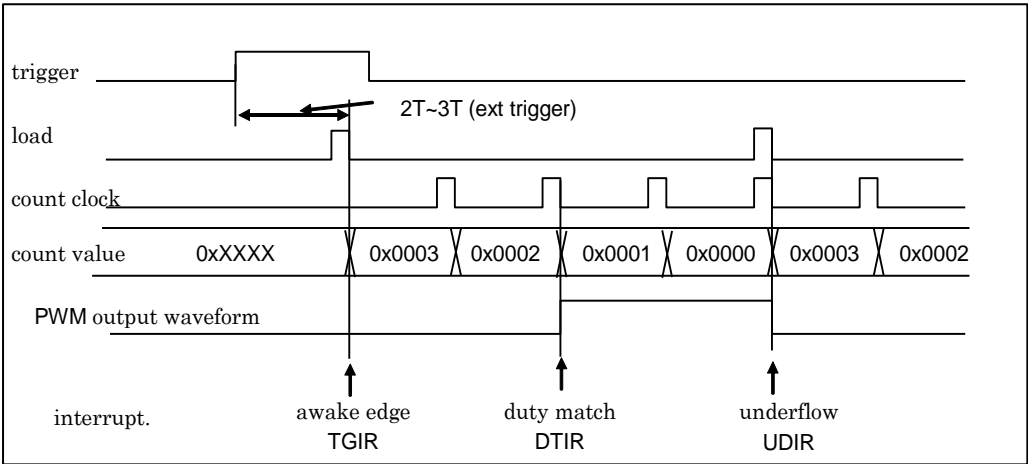
This section shows interrupt factors and a timing chart.

Interrupt Factors and Timing Chart (PWM Output: Normal Polarity)

As the time from the input of a trigger until the loading of a counter value, the required software trigger time is T, and the required external trigger time is 2T to 3T (T: internal clock cycle).

Figure 9-5 shows the interrupt factors and a timing chart where the cycle setting value is 3 and the duty value is 1.

Figure 9-5 PWM Timer Interrupt Factors and Timing Chart



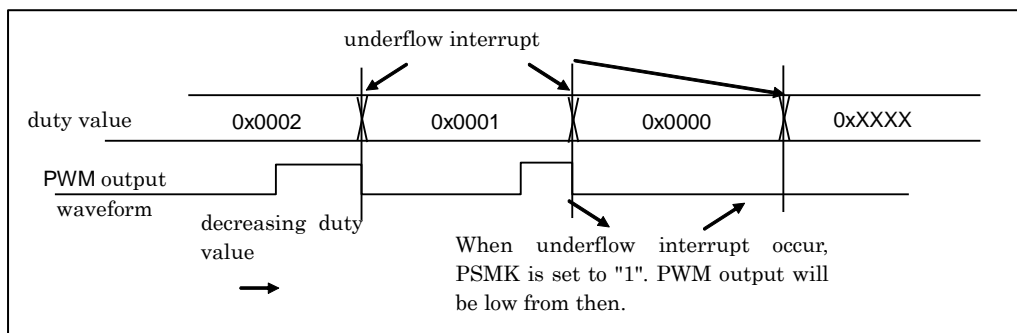
### 9.1.4. Output Waveform

This section shows PWM output.

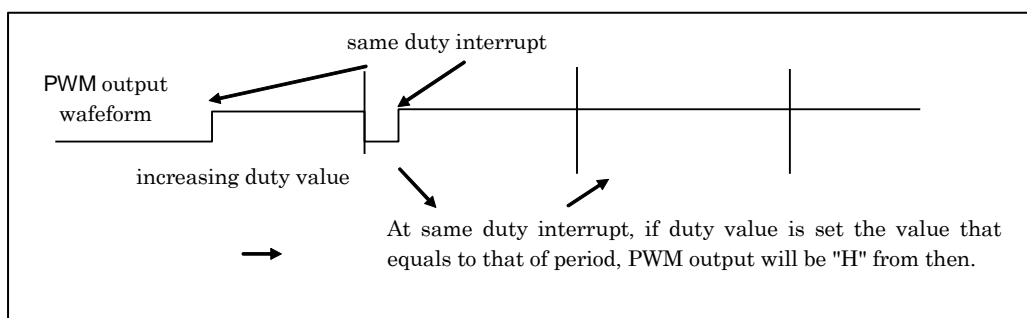
Output Methods for All "L" or All "H" in PWM Output

Figure 9-6 shows an output method where the PWM output is all "L". Figure 9-7 shows an output method where it is all "H".

**Figure 9-6 Example where PWM Output is All "L" Level**



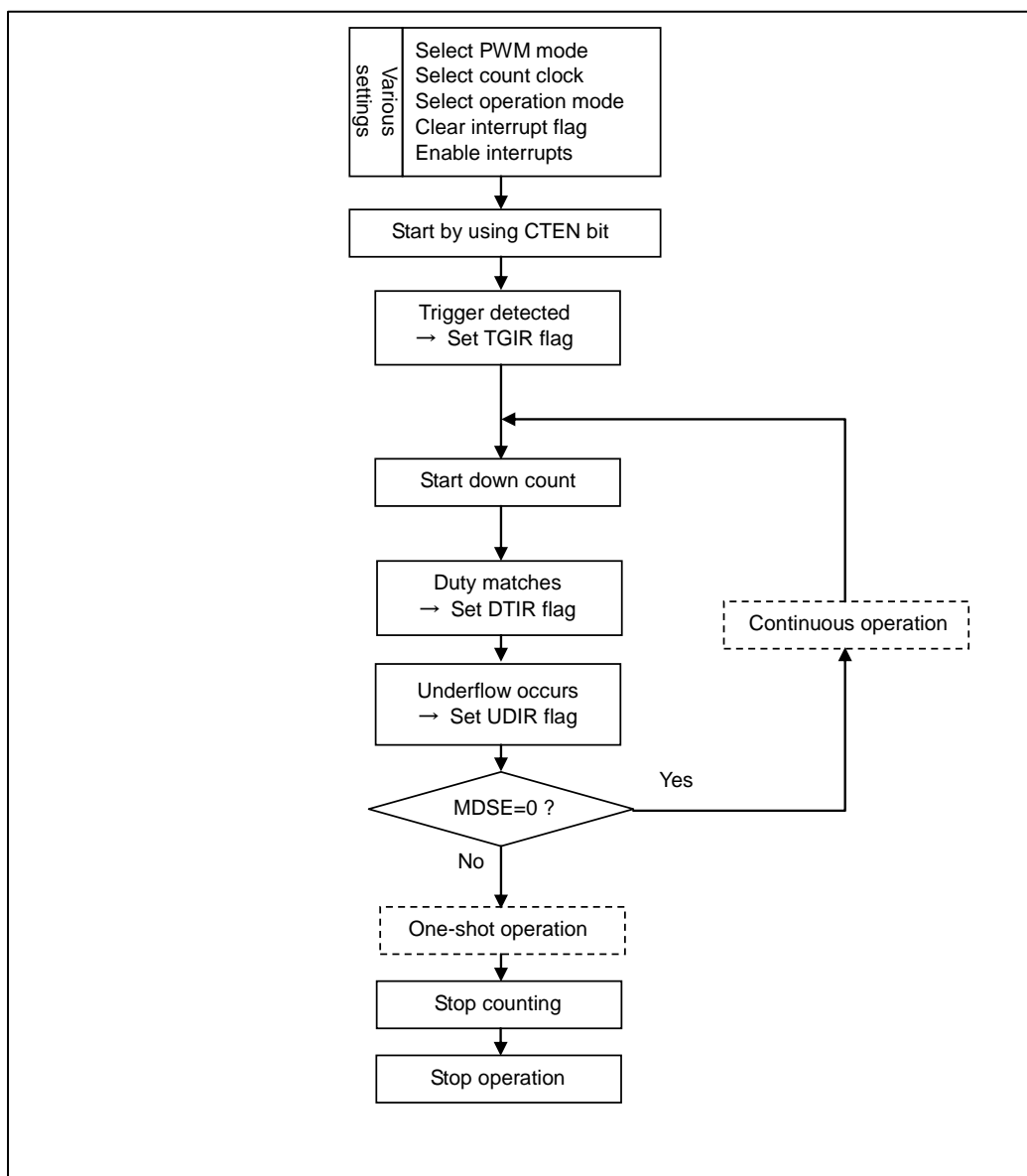
**Figure 9-7 Example where PWM Output is All "H" Level**



### 9.1.5. PWM Timer Operation Flow

This section shows the PWM timer operation flow.

Figure 9-8 PWM Timer Operation Flow





### 9.1.6. Timer Control Registers (TMCR, TMCR2), Status Control Register (STC), Status Control Clear Register (STCC), and Status Control Set Register (STCS)

The timer control register (TMCR) controls the PWM timer. Note that there are bits that cannot be rewritten during PWM timer operation.

For details on writing to the status control register (STC), see Section "8. Notes on Using".

#### (1) Timer Control Register (Upper byte of TMCR)

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved	CKS2	CKS1	CKS0	RTGEN	PMSK	EGS1	EGS0
ACCESS_TYPE	R0,W0	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit15] Reserved: Reserved bit

#### [bit14:12, TMCR2:bit8] CKS3 to CKS0: Count clock selection bits

- These bits select a count clock for the 16-bit down counter
- Any modification to the count clock is reflected immediately after the setting is changed. Therefore, modify CKS3 to CKS0 while counting is stopped (CTEN="0"). However, note that the bits can be modified at the same time as "1" is written to the count operation enable bit (CTEN).

Value				Description
CKS3	CKS2	CKS1	CKS0	
0	0	0	0	$\phi$
0	0	0	1	$\phi / 4$
0	0	1	0	$\phi / 16$
0	0	1	1	$\phi / 128$
0	1	0	0	$\phi / 256$
0	1	0	1	External clock (rising-edge event)
0	1	1	0	External clock (falling-edge event)
0	1	1	1	External clock (both-edges event)
1	0	0	0	$\phi / 512$
1	0	0	1	$\phi / 1024$
1	0	1	0	$\phi / 2048$
Other than above				Setting prohibited

$\phi$ : peripheral clock

#### [bit11] RTGEN: Restart enable bit

This bit enables restart by a software trigger or trigger input.

Value	Description
0	Disable restart.
1	Enable restart.

**[bit10] PMSK: Pulse output mask bit**

- This bit controls the output waveform level of the PWM output waveform.
- When the bit is "0", the PWM waveform is output as is.
- When the bit is "1", PWM output is masked to L output regardless of the cycle or duty setting value.

Value	Description
0	Normal output
1	Fixed at L output

**Note:**

- If the output polarity specification bit (OSEL) in the timer control register (lower byte of TMCR) is set for inverted output, setting the PMSK bit to "1" results in masking to H output.

**[bit9:8] EGS1 to EGS0: Trigger input edge selection bits**

- These bits select the valid edge for an input waveform as an external start factor, and they set trigger conditions.
- For the initial value or the "00" setting, no valid edge is selected for an input waveform, so no external waveform causes a start.
- Modify EGS1 and EGS0 while counting is stopped (CTEN="0"). However, note that the bits can be modified at the same time as "1" is written to the CTEN bit.

Value	Description
00	Trigger input invalid
01	Rising edge
10	Falling edge
11	Both edges

**Note:**

- If "1" is written to the STRG bit, the software trigger becomes valid regardless of the EGS1 and EGS0 settings.





## (2) Timer Control Register (Lower byte of TMCR)

BITS_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved	FMD2	FMD1	FMD0	OSEL	MDSE	CTEN	STRG
ACCESS_TYPE	R0,W0	R/W	R/W	R/W	R/W	R/W	R/W	R0,W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit7] Reserved: Reserved bit





### [bit6:4] FMD2 to FMD0: Timer function selection bits

- These bits select the timer function.
- If "001" is set in the FMD2 to FMD0 bits, the PWM function is selected.
- Modify these bits while the timer is stopped (CTEN="0"). However, note that the bits can be modified at the same time as "1" is written to the CTEN bit.

Value			Description
0	0	0	Reset mode
0	0	1	16-bit PWM timer
0	1	0	16-bit PPG timer
0	1	1	16/32-bit reload timer
1	0	0	16/32-bit PWC timer
1	0	1	Setting prohibited
1	1	0	
1	1	1	

### [bit3] OSEL: Output polarity specification bit

- This bit sets the PWM output polarity.

Polarity	After Reset	Duty Match	Underflow
Normal	"L" output		
Inverse	"H" output		

Value	Description
0	Normal polarity
1	Inverse polarity

**[bit2] MDSE: Mode selection bit**

- This bit selects either operation for continuous pulse output or one-shot operation for single-pulse output.
- Modify the bit while the timer is stopped (CTEN="0"). However, note that the bits can be modified at the same time as "1" is written to the CTEN bit.

Value	Description
0	Continuous operation
1	One-shot operation

**[bit1] CTEN: Count operation enable bit**

- This bit enables operation of the down counter.
- If "0" is written to this bit when counter operation is enabled (CTEN bit is "1"), the counter stops.

Value	Description
0	Stop operation.
1	Enable operation.

**[bit0] STRG: Software trigger bit**

- If "1" is written to the STRG bit when the CTEN bit is "1", a software trigger is applied.
- The read value of the STRG bit is always "0".

**Notes:**

- *Even if "1" is written to the CTEN and STRG bits at the same time, a software trigger is applied.*
- *If "1" is written to the STRG bit, the software trigger becomes valid regardless of the EGS1 and EGS0 settings.*



### (3) Timer Control Register 2 (TMCR2)

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved							CKS3
ACCESS_TYPE	R0,W0							R/W
PROT_TYPE	-							
INITIAL_VALUE	0000000							0

Note: This register is located in the upper byte of the STC register.

**[bit15:9] Reserved: Reserved bits**

**[bit8] CKS3: Count clock selection bit**

See Section "Count clock selection bits" in "(1) Timer Control Register (Upper byte of TMCR)".

#### (4) Status Control Register (STC)

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved	TGIE	DTIE	UDIE	Reserved	TGIR	DTIR	UDIR
ACCESS_TYPE	R0,W0	R/W	R/W	R/W	R0,W0	R,W	R,W	R,W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

Note: The TMCR2 register is located in the upper byte of this register.

#### [bit7] Reserved: Reserved bit

#### [bit6] TGIE: Trigger interrupt request enable bit

- This bit controls interrupt requests of the trigger interrupt request bit (bit2 TGIR).
- If the TGIE bit is enabled and the TGIR bit is set to "1", an interrupt request is issued to the CPU.
- Writing "1" to the STCC:TGIEC bit clears this bit.
- Writing "1" to the STCS:TGIES bit sets this bit.

Value	Description
0	Disable interrupt requests.
1	Enable interrupt requests.

#### [bit5] DTIE: Duty match interrupt request enable bit

- This bit controls interrupt requests of the duty match interrupt request bit (bit1 DTIR).
- If the DTIE bit is enabled and the DTIR bit is set to "1", an interrupt request is issued to the CPU.
- Writing "1" to the STCC:DTIEC bit clears this bit.
- Writing "1" to the STCS:DTIES bit sets this bit.

Value	Description
0	Disable interrupt requests.
1	Enable interrupt requests.

#### [bit4] UDIE: Underflow interrupt request enable bit

- This bit controls interrupt requests of the underflow interrupt request bit (bit0 UDIR).
- If the UDIE bit is enabled and the UDIR bit is set to "1", an interrupt request is issued to the CPU.
- Writing "1" to the STCC:UDIEC bit clears this bit.
- Writing "1" to the STCS:UDIES bit sets this bit.

Value	Description
0	Disable interrupt requests.
1	Enable interrupt requests.

#### [bit3] Reserved: Reserved bit



**[bit2] TGIR: Trigger interrupt request bit**

- The TGIR bit is set to "1" when a software trigger or trigger input is detected.
- Writing "0" clears the TGIR bit.
- Writing "1" to the STCC:TGIRC bit clears this bit.
- Writing "1" to the TGIR bit has no effect on the bit value.

Value	Description
0	The interrupt factor is cleared.
1	Detect the interrupt factor.

**[bit1] DTIR: Duty match interrupt request bit**

- The DTIR bit is set to "1" when the count value matches the duty setting value.
- Writing "0" clears the DTIR bit.
- Writing "1" to the STCC:DTIRC bit clears this bit.
- Writing "1" to the DTIR bit has no effect on the bit value.

Value	Description
0	The interrupt factor is cleared.
1	Detect the interrupt factor.

**[bit0] UDIR: Underflow interrupt request bit**

- The UDIR bit is set to "1" when the count value underflows from "0x0000" to "0xFFFF".
- Writing "0" clears the UDIR bit.
- Writing "1" to the STCC:UDIRC bit clears this bit.
- Writing "1" to the UDIR bit has no effect on the bit value.

Value	Description
0	The interrupt factor is cleared.
1	Detect the interrupt factor.

### (5) Status Control Clear Register (STCC)

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved	TGIEC	DTIEC	UDIEC	Reserved	TGIRC	DTIRC	UDIRC
ACCESS_TYPE	R0,W0	R0,W	R0,W	R0,W	R0,W0	R0,W	R0,W	R0,W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

Note: Reserved bits are located in the upper byte of this register.

#### [bit7] Reserved: Reserved bit

#### [bit6] TGIEC: Trigger interrupt request enable clear bit

If "1" is written to this bit, the STC:TGIE bit is cleared to "0".

Value	Description
0	Invalid
1	Clear the TGIE bit.

#### [bit5] DTIEC: Duty match interrupt request enable clear bit

If "1" is written to this bit, the STC:DTIE bit is cleared to "0".

Value	Description
0	Invalid
1	Clear the DTIE bit.

#### [bit4] UDIEC: Underflow interrupt request enable clear bit

If "1" is written to this bit, the STC:UDIE bit is cleared to "0".

Value	Description
0	Invalid
1	Clear the UDIE bit.

#### [bit3] Reserved: Reserved bit

#### [bit2] TGIRC: Trigger interrupt request clear bit

If "1" is written to this bit, the STC:TGIR bit is cleared to "0".

Value	Description
0	Invalid
1	Clear the TGIR bit.

#### [bit1] DTIRC: Duty match interrupt request clear bit

If "1" is written to this bit, the STC:DTIR bit is cleared to "0".

Value	Description
0	Invalid
1	Clear the DTIR bit.



**[bit0] UDIRC: Underflow interrupt request clear bit**

If "1" is written to this bit, the STC:UDIR bit is cleared to "0".

Value	Description
0	Invalid
1	Clear the UDIR bit.

**(6) Status Control Set Register (STCS)**

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved	TGIES	DTIES	UDIES	Reserved			
ACCESS_TYPE	R0,W0	R0,W	R0,W	R0,W	R0,W0			
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0000			

**[bit7] Reserved: Reserved bit**

**[bit6] TGIES: Trigger interrupt request enable set bit**

- If "1" is written to this bit, the STC:TGIE bit is set to "1".

Value	Description
0	Invalid
1	Set the TGIE bit.

**[bit5] DTIES: Duty match interrupt request enable set bit**

- If "1" is written to this bit, the STC:DTIE bit is set to "1".

Value	Description
0	Invalid
1	Set the DTIE bit.

**[bit4] UDIES: Underflow interrupt request enable set bit**

- If "1" is written to this bit, the STC:UDIE bit is set to "1".

Value	Description
0	Invalid
1	Set the UDIE bit.

**[bit3:0] Reserved: Reserved bits**





### 9.1.7. PWM Cycle Setting Register (PCSR)

The PWM cycle setting register (PCSR) is a register with a buffer for setting a cycle. There is a transfer to the timer register at the start time and at the underflow time.

BIT_OFFSET	15-0
BIT_NAME	PCSR
ACCESS_TYPE	R/W
PROT_TYPE	-
INITIAL_VALUE	XXXXXXXX_XXXXXXXX

#### [bit15:0] PCSR[15:0]: PWM cycle setting register

These bits compose the register with a buffer for setting a cycle. There is a transfer to the timer register at the start time and at the underflow time.

When initializing and rewriting the cycle setting register, be sure to write to the duty setting register after writing to the cycle setting register.

- Use 16-bit data access for the PCSR register.
- After configuring the PWM function with the FMD2 to FMD0 bits in the TMCR register, set a cycle in the PCSR register.

### 9.1.8. PWM Duty Setting Register (PDUT)

The PWM duty setting register (PDUT) is a register with a buffer for setting a duty. An underflow causes a buffer transfer.

BIT_OFFSET	15-0
BIT_NAME	PDUT
ACCESS_TYPE	R/W
PROT_TYPE	-
INITIAL_VALUE	XXXXXXXX_XXXXXXXX

#### [bit15:0] PDUT[15:0]: PWM duty setting register

These bits compose the register with a buffer for setting a duty. An underflow causes a transfer from the buffer.

If the set values of the cycle setting register and duty setting register are the same value, the output for normal polarity is all "H", and the output for inverse polarity is all "L".

If  $PCSR < PDUT$  in the set values, the output for normal polarity is all "L", and the output for inverse polarity is all "H".

- Use 16-bit data access for the PDUT register.
- After configuring the PWM function with the FMD2 to FMD0 bits in the TMCR register, set a duty in the PDUT register.



### 9.1.9. Timer Register (TMR)

The timer register (TMR) can read the value of the 16-bit down counter.

BIT_OFFSET	15-0
BIT_NAME	TMR
ACCESS_TYPE	R,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000

#### [bit15:0] TMR[15:0]: Timer register

The timer register can read the value of the 16-bit down counter.

Use 16-bit data access for the TMR register.

## 9.2. PPG Timer Function

Only one of the following timer functions can be selected for the base timer in the FMD2 to FMD0 bit settings in the timer control register (TMCR): 16-bit PWM timer, 16-bit PPG timer, 16/32-bit reload timer, and 16/32-bit PWC timer. This section explains the timer function with the PPG setting.

1. 16-bit PPG Timer Operation
2. Continuous Operation
3. One-shot Operation
4. Interrupt Factors and Timing Chart
5. PPG Timer Operation Flow
6. Timer Control Registers (TMCR, TMCR2), Status Control Register (STC), Status Control Clear Register (STCC), and Status Control Set Register (STCS) When the PPG Timer is Selected
7. L Width Setting Reload Register (PRLL)
8. H Width Setting Reload Register (PRLH)
9. Timer Register (TMR)

### 9.2.1. 16-bit PPG Timer Operation

PPG timer operation can control output pulses by using the L width and H width settings for the output pulses in the respective reload registers.

#### (1) Overview of Operation

There are two 16-bit reload registers for the L width and H width settings and one buffer for the H width setting (PRLH, PRLH, and PRLHB).

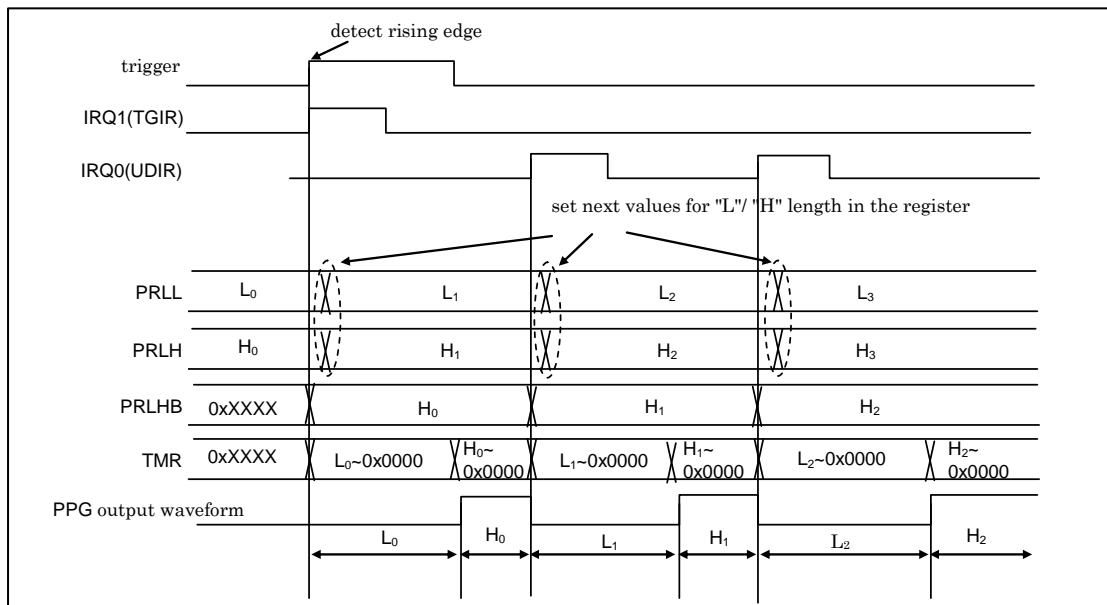
An activation trigger first causes loading of the set value of PRLH into the 16-bit down counter. At the same time, the set value of PRLH is transferred to PRLHB. The PPG output level changes to L, and each count clock counts down. The detection of an underflow causes reloading of the PRLHB value into the counter and inversion of the PPG output waveform. Meanwhile, each count clock counts down. The detection of another underflow causes inversion of the PPG output waveform, reloading of the set value of PRLH into the counter, and transfer of the set value of PRLH to PRLHB.

The pulse output by the output waveform from this operation has an L width and H width corresponding to each reload register value.

#### (2) Timing of Writing to Reload Registers

Data is written to the reload registers PRLH and PRLH when an activation trigger is detected during the period after the underflow interrupt factor (UDIR) is set to "1" and before the change to the next cycle. The data that is set at this time is the settings for the next cycle. The set data in PRLH and PRLH is automatically transferred to TMR and PRLHB, respectively, when an activation trigger is detected or when an underflow occurs at the H width count end time. The data transferred to PRLHB is automatically reloaded into TMR when an underflow occurs at the L width count end time.

Figure 9-9 Timing Chart of Writing to Reload Registers



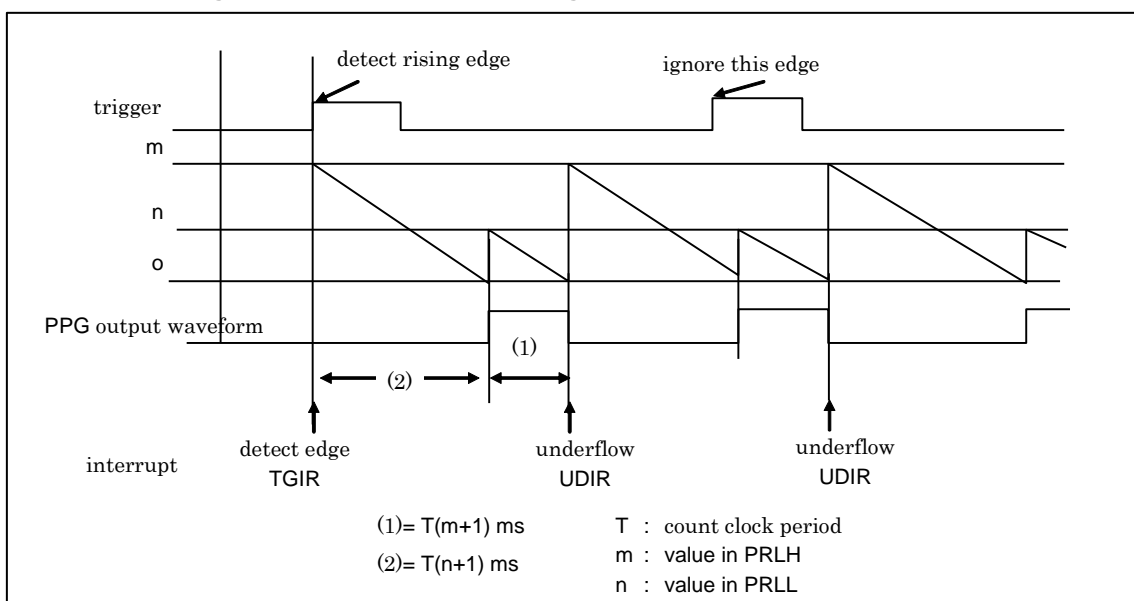
## 9.2.2. Continuous Operation

In continuous operation, pulses can be output continuously through updating of the L width and H width at the set timing for each interrupt factor. If restart is enabled, the counter is reloaded when an edge is detected during operation.

### Continuous Operation

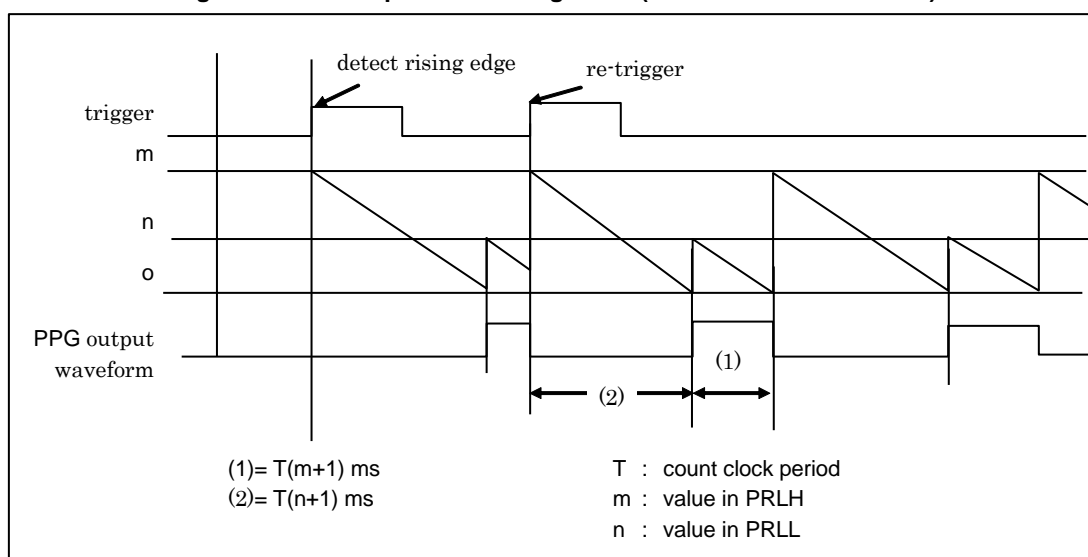
#### a) When Restart is Disabled (RTGEN=0)

Figure 9-10 PPG Operation Timing Chart (When Restart is Disabled)



#### b) When Restart is Enabled (RTGEN=1)

Figure 9-11 PPG Operation Timing Chart (When Restart is Enabled)



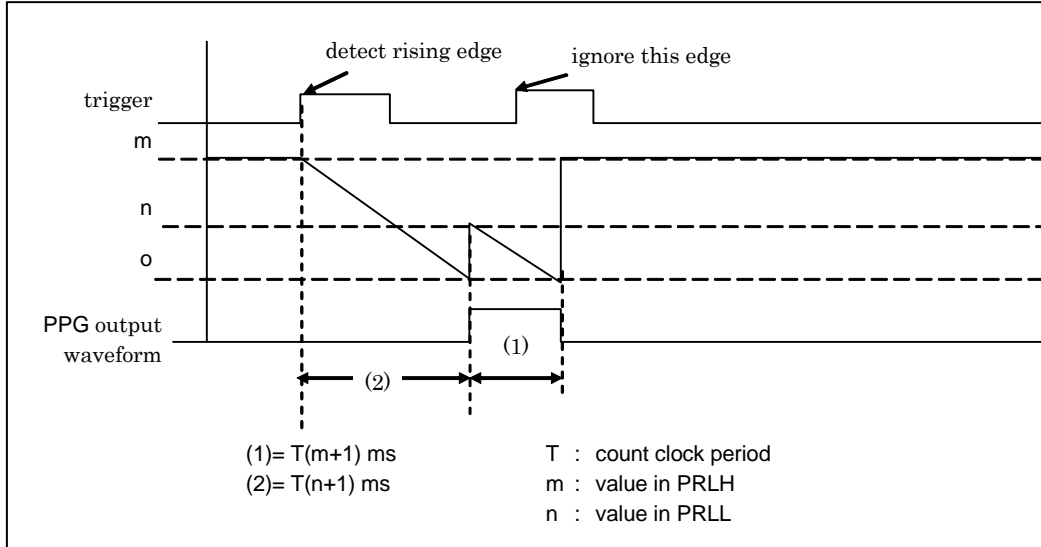
### 9.2.3. One-shot Operation

In one-shot operation, a trigger can cause the output of a single pulse of any width. If restart is enabled, the counter is reloaded when an edge is detected during operation.

#### (1) One-shot Operation

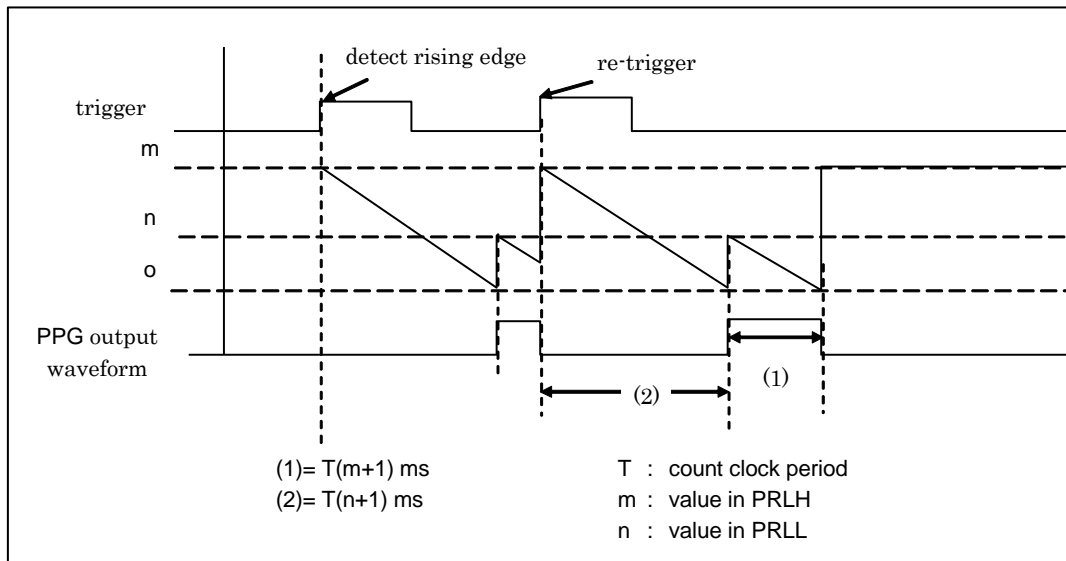
##### a) When Restart is Disabled (RTGEN=0)

Figure 9-12 One-shot Operation Timing Chart (Trigger Restart Disabled)



##### b) When Restart is Enabled (RTGEN=1)

Figure 9-13 One-shot Operation Timing Chart (Trigger Restart Enabled)



## **(2) Relationship between Reload Value and Pulse Width**

The output pulse width is the following value: 1 is added to the value written to the 16-bit reload register, and the sum is multiplied by the cycles of the count clock. Therefore, the pulse width of 1 cycle of the count clock is the value when the reload register value is "0x0000". The pulse width of 65536 cycles of the count clock is the value when the reload register value is "0xFFFF". The pulse width calculation formula is as follows.

$$\begin{aligned} PL &= T \times (L + 1) & PL: L \text{ pulse width} \\ PH &= T \times (H + 1) & PH: H \text{ pulse width} \end{aligned}$$

$T$ : Count clock cycle  
 $L$ : PRL value  
 $H$ : PRLH value



### 9.2.4. Interrupt Factors and Timing Chart

This section shows interrupt factors and a timing chart.

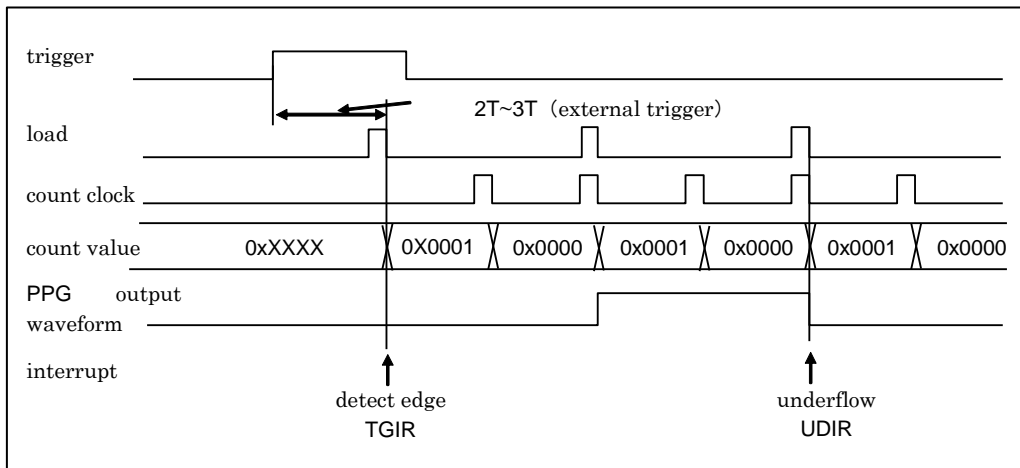
#### Interrupt Factors and Timing Chart (PPG Output: Normal Polarity)

For the period from trigger application until the loading of a counter value, the required software trigger time is  $T$ , and the required external trigger time is  $2T$  to  $3T$  ( $T$ : Internal clock cycle).

An interrupt factor is generated when a PPG activation trigger is detected or when an underflow is detected at the H level output time.

Figure 9-14 shows the interrupt factors and a timing chart where the L width setting value is 1 and the H width setting value is 1.

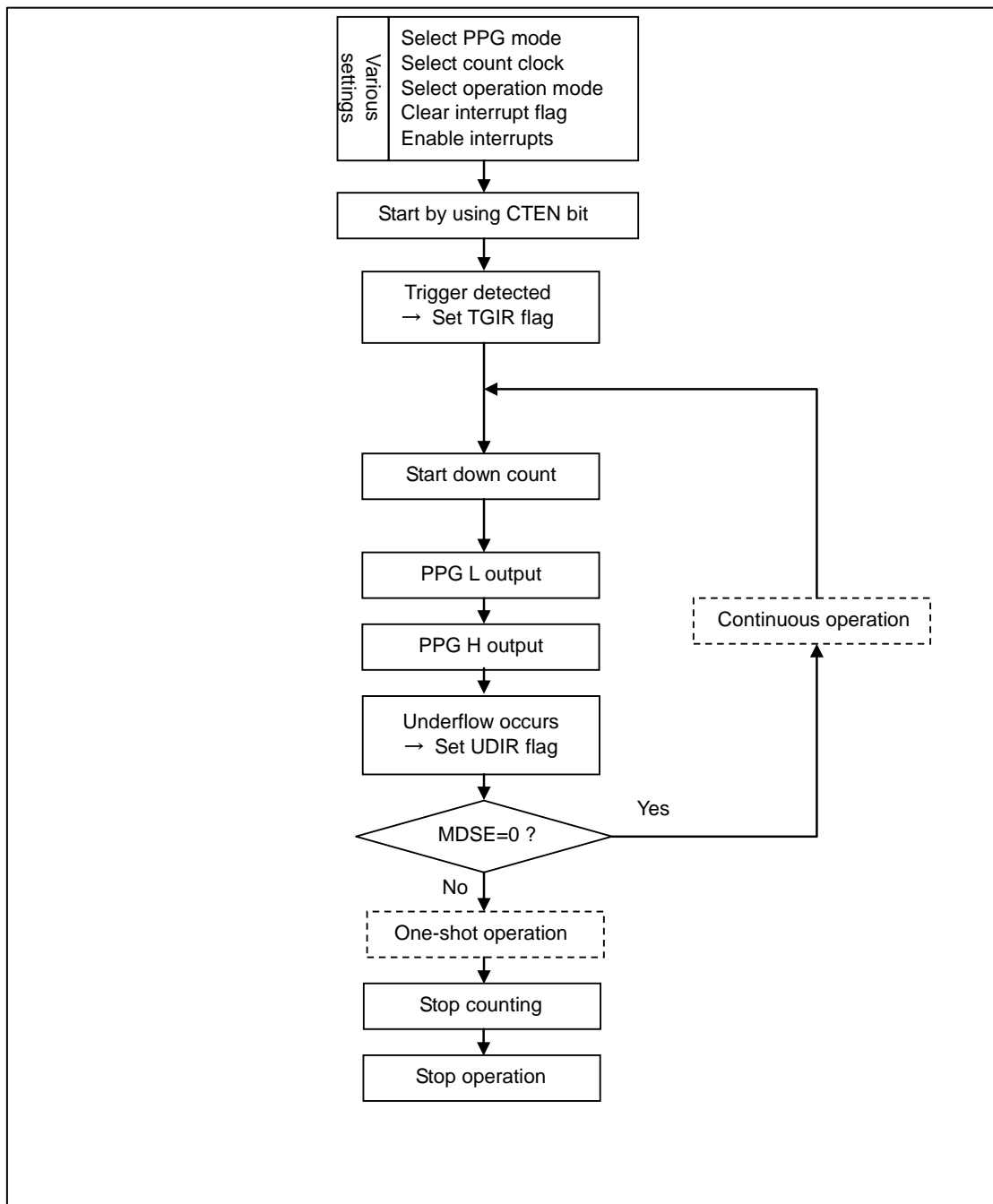
Figure 9-14 PPG Timer Interrupt Factors and Timing Chart



### 9.2.5. PPG Timer Operation Flow

This section shows the PPG timer operation flow.

Figure 9-15 PPG Timer Operation Flow





## 9.2.6. Timer Control Registers (TMCR, TMCR2), Status Control Register (STC), Status Control Clear Register (STCC), and Status Control Set Register (STCS) When the PPG Timer is Selected

For details on writing to the status control register (STC), see Section "8. Notes on Using".

### (1) Timer Control Register (Upper Byte of TMCR)

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved	CKS2	CKS1	CKS0	RTGEN	PMSK	EGS1	EGS0
ACCESS_TYPE	R0,W0	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit15] Reserved: Reserved bit

#### [bit14:12, TMCR2:bit8] CKS3 to CKS0: Count clock selection bits

- These bits select a count clock for the 16-bit down counter.
- Any modification to the count clock is reflected immediately after the setting is changed. Therefore, modify CKS3 to CKS0 while counting is stopped (CTEN="0"). However, note that the bits can be modified at the same time as "1" is written to the CTEN bit.

Value				Description
CKS3	CKS2	CKS1	CKS0	
0	0	0	0	$\phi$
0	0	0	1	$\phi / 4$
0	0	1	0	$\phi / 16$
0	0	1	1	$\phi / 128$
0	1	0	0	$\phi / 256$
0	1	0	1	External clock (rising-edge event)
0	1	1	0	External clock (falling-edge event)
0	1	1	1	External clock (both-edges event)
1	0	0	0	$\phi / 512$
1	0	0	1	$\phi / 1024$
1	0	1	0	$\phi / 2048$
Other than above				Setting prohibited

$\phi$ : peripheral clock

#### [bit11] RTGEN: Restart enable bit

This bit enables restart by a software trigger or trigger input.

Value	Description
0	Disable restart.
1	Enable restart.

**[bit10] PMSK: Pulse output mask bit**

- This bit controls the output waveform level of the PPG output waveform.
- When the bit is "0", the PPG waveform is output as is.
- When the bit is "1", PPG output is masked to L output regardless of the cycle or duty setting value.

Value	Description
0	Normal output
1	Fixed at L output

**Note:**

- *If the output polarity specification bit (OSEL) in the timer control register (lower byte of TMCR) is set for inverted output, setting the PMSK bit to "1" results in masking to H output.*

**[bit9:8] EGS1 to EGS0: Trigger input edge selection bits**

- These bits select the valid edge for an input waveform as an external start factor, and they set trigger conditions.
- For the initial value or the "00" setting, no valid edge is selected for an input waveform, so no external waveform causes a start.
- Modify EGS1 and EGS0 while counting is stopped (CTEN="0"). However, note that the bits can be modified at the same time as "1" is written to the CTEN bit.

Value	Description
00	Trigger input invalid
01	Rising edge
10	Falling edge
11	Both edges

**Note:**

- *If "1" is written to the STRG bit, the software trigger becomes valid regardless of the EGS1 and EGS0 settings.*



## (2) Timer Control Register (Lower Byte of TMCR)

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved	FMD2	FMD1	FMD0	OSEL	MDSE	CTEN	STRG
ACCESS_TYPE	R0,W0	R/W	R/W	R/W	R/W	R/W	R/W	R0,W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit7] Reserved: Reserved bit

### [bit6:4] FMD2 to FMD0: Timer function selection bits

- These bits select the timer function.
- If "010" is set in the FMD2 to FMD0 bits, the PPG function is selected.
- Modify these bits while the timer is stopped (CTEN="0"). However, note that the bits can be modified at the same time as "1" is written to the CTEN bit.

Value			Description
0	0	0	Reset mode
0	0	1	16-bit PWM timer
0	1	0	16-bit PPG timer
0	1	1	16/32-bit reload timer
1	0	0	16/32-bit PWC timer
1	0	1	Setting prohibited
1	1	0	
1	1	1	

### [bit3] OSEL: Output polarity specification bit

This bit sets the PPG output polarity.

Polarity	After Reset	L Width Count End	H Width Count End
Normal	"L" output		
Inverse	"H" output		

Value	Description
0	Normal polarity
1	Inverse polarity

**[bit2] MDSE: Mode selection bit**

- This bit selects either operation for continuous pulse output or one-shot operation for single-pulse output.
- Modify the bit while the timer is stopped (CTEN="0"). However, note that the bits can be modified at the same time as "1" is written to the CTEN bit.

Value	Description
0	Continuous operation
1	One-shot operation

**[bit1] CTEN: Count operation enable bit**

- This bit enables operation of the down counter.
- If "0" is written to this bit when counter operation is enabled (CTEN bit is "1"), the counter stops.

Value	Description
0	Stop operation.
1	Enable operation.

**[bit0] STRG: Software trigger bit**

If "1" is written to the STRG bit when the CTEN bit is "1", a software trigger is applied.

Value	Description
0	Invalid
1	Startup by software

**Notes:**

- Even if "1" is written to the CTEN and STRG bits at the same time, a software trigger is applied.
- If "1" is written to the STRG bit, the software trigger becomes valid regardless of the EGS1 and EGS0 settings.



### (3) Timer Control Register 2 (TMCR2)

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved							CKS3
ACCESS_TYPE	R0,W0							R/W
PROT_TYPE	-							
INITIAL_VALUE	0000000							0

**[bit15:9] Reserved: Reserved bits**

**[bit8] CKS3: Count clock selection bit**

See Section "Count clock selection bits" in "(1) Timer Control Register (Upper Byte of TMCR)".

#### (4) Status Control Register (STC)

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved	TGIE	Reserved	UDIE	Reserved	TGIR	Reserved	UDIR
ACCESS_TYPE	R0,W0	R/W	R0,W0	R/W	R0,W0	R,W	R0,W0	R,W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

Note: The TMCR2 register is located in the upper byte of this register.

##### [bit7] Reserved: Reserved bit

##### [bit6] TGIE: Trigger interrupt request enable bit

- This bit controls interrupt requests of the trigger interrupt request bit (bit2 TGIR).
- If the TGIE bit is enabled and the TGIR bit is set to "1", an interrupt request is issued to the CPU.
- Writing "1" to the STCC:TGIEC bit clears this bit.
- Writing "1" to the STCS:TGIES bit sets this bit.

Value	Description
0	Disable interrupt requests.
1	Enable interrupt requests.

##### [bit5] Reserved: Reserved bit

##### [bit4] UDIE: Underflow interrupt request enable bit

- This bit controls interrupt requests of the underflow interrupt request bit (bit0 UDIR).
- If the UDIE bit is enabled and the UDIR bit is set to "1", an interrupt request is issued to the CPU.
- Writing "1" to the STCC:UDIEC bit clears this bit.
- Writing "1" to the STCS:UDIES bit sets this bit.

Value	Description
0	Disable interrupt requests.
1	Enable interrupt requests.

##### [bit3] Reserved: Reserved bit

##### [bit2] TGIR: Trigger interrupt request bit

- The TGIR bit is set to "1" when a software trigger or trigger input is detected.
- Writing "0" clears the TGIR bit.
- Writing "1" to the STCC:TGIRC bit clears this bit.
- Writing "1" to the TGIR bit has no effect on the bit value.

Value	Description
0	The interrupt factor is cleared.
1	Detect the interrupt factor.

##### [bit1] Reserved: Reserved bit





**[bit0] UDIR: Underflow interrupt request bit**

- During counting from the set H width value, the UDIR bit is set to "1" when the count value underflows and changes from "0x0000" to "0xFFFF".
- Writing "0" clears the UDIR bit.
- Writing "1" to the STCC:UDIRC bit clears this bit.
- Writing "1" to the UDIR bit has no effect on the bit value.

Value	Description
0	The interrupt factor is cleared.
1	Detect the interrupt factor.

### (5) Status Control Clear Register (STCC)

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved	TGIEC	Reserved	UDIEC	Reserved	TGIRC	Reserved	UDIRC
ACCESS_TYPE	R0,W0	R0,W	R0,W0	R0,W	R0,W0	R0,W	R0,W0	R0,W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

**[bit7] Reserved: Reserved bit**

**[bit6] TGIEC: Trigger interrupt request enable clear bit**

If "1" is written to this bit, the STC:TGIE bit is cleared to "0".

Value	Description
0	Invalid
1	Clear the TGIE bit.

**[bit5] Reserved: Reserved bit**

**[bit4] UDIEC: Underflow interrupt request enable clear bit**

If "1" is written to this bit, the STC:UDIE bit is cleared to "0".

Value	Description
0	Invalid
1	Clear the UDIE bit.

**[bit3] Reserved: Reserved bit**

**[bit2] TGIRC: Trigger interrupt request clear bit**

If "1" is written to this bit, the STC:TGIR bit is cleared to "0".

Value	Description
0	Invalid
1	Clear the TGIR bit.

**[bit1] Reserved: Reserved bit**

**[bit0] UDIRC: Underflow interrupt request clear bit**

If "1" is written to this bit, the STC:UDIR bit is cleared to "0".

Value	Description
0	Invalid
1	Clear the UDIR bit.



**(6) Status Control Set Register (STCS)**

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved	TGIES	Reserved	UDIES	Reserved			
ACCESS_TYPE	R0,W0	R0,W	R0,W0	R0,W	R0,W0			
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0000			

**[bit7] Reserved: Reserved bit**

**[bit6] TGIES: Trigger interrupt request enable set bit**

If "1" is written to this bit, the STC:TGIE bit is set to "1".

Value	Description
0	Invalid
1	Set the TGIE bit.

**[bit5] Reserved: Reserved bit**

**[bit4] UDIES: Underflow interrupt request enable set bit**

If "1" is written to this bit, the STC:UDIE bit is set to "1".

Value	Description
0	Invalid
1	Set the UDIE bit.

**[bit3:0] Reserved: Reserved bits**

### 9.2.7. L Width Setting Reload Register (PRL)

The L width setting reload register (PRL) is the register used to set the L width of the PPG output waveform. An underflow at the activation trigger detection time or after the end of H width counting causes a transfer to the timer register.

BIT_OFFSET	15-0
BIT_NAME	PRL
ACCESS_TYPE	R/W
PROT_TYPE	-
INITIAL_VALUE	XXXXXXXX_XXXXXXXX

These bits compose the register used to set the L width of the PPG output waveform. An underflow at the activation trigger detection time or H width count end time causes a transfer to the timer register.

- Use 16-bit data access for the PRL register.
- After configuring the PPG function with the FMD2 to FMD0 bits in the TMCR register, set the L width in the PRL register.



### 9.2.8. H Width Setting Reload Register (PRLH)

The H width setting reload register (PRLH) is the register with a buffer used to set the H width of the PPG output waveform. An underflow when an activation trigger is detected or after H width counting ends causes a transfer from PRLH to the buffer register. An underflow at the L width count end time causes a transfer from the buffer register to the timer register.

BIT_OFFSET	15-0
BIT_NAME	PRLH
ACCESS_TYPE	R/W
PROT_TYPE	-
INITIAL_VALUE	XXXXXXXX_XXXXXXXX

These bits compose the register used to set the H width of the PPG output waveform. An underflow at the activation trigger detection time or H width count end time causes a transfer from PRLH to the buffer register. An underflow at the L width count end time causes a transfer from the buffer register to the timer register.

- Use 16-bit data access for the PRLH register.
- After configuring the PPG function with the FMD2 to FMD0 bits in the TMCR register, set the H width in the PRLH register.

### 9.2.9. Timer Register (TMR)

The timer register (TMR) can read the value of the 16-bit down counter.

BIT_OFFSET	15-0
BIT_NAME	TMR
ACCESS_TYPE	R,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000

#### [bit15:0] TMR: Timer register

The timer register can read the value of the 16-bit down counter.

Use 16-bit data access for the TMR register.



### 9.3. Reload Timer Function

Only one of the following timer functions can be selected for the base timer in the FMD2 to FMD0 bit settings in the timer control register (TMCR): 16-bit PWM timer, 16-bit PPG timer, 16/32-bit reload timer, and 16/32-bit PWC timer. This section explains the timer function with the reload timer setting.

1. Operations of 16-bit Reload Timer
2. Reload Timer Operation Flow
3. Timer Control Registers (TMCR, TMCR2), Status Control Register (STC), Status Control Clear Register (STCC) and Status Control Set Register (STCS), When the Reload Timer is Selected
4. Cycle Setting Register (PCSR)
5. Timer Register (TMR)

9.3.1.      **Operations of 16-bit Reload Timer**

The reload timer operates by performing a countdown from the set value in the cycle setting register, in synchronization with the count clock. When the count value becomes 0, the counting ends, or the timer automatically loads the cycle setting to continue operating until the countdown is stopped.

**(1) Count Operation When Internal Clock is Selected**

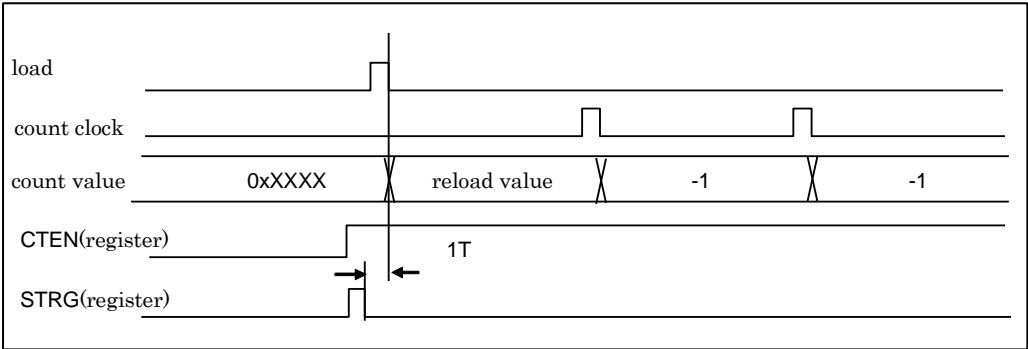
To start a count operation at the same time as counting is enabled, write "1" to both the CTEN bit and STRG bit in the timer control register. In a state where the timer has started (CTEN="1"), trigger input with the STRG bit is always enabled regardless of the operation mode.

With count operations enabled, the start of the timer by a software trigger or external trigger results in the cycle setting register value being loaded into the counter and a countdown being started.

The time required from the generation of a counter start trigger until the loading of cycle setting register data into the counter is 1T (T: Internal clock cycle).

Figure 9-16 shows the start and operation of the counter using a software trigger.

**Figure 9-16 Count Operation When the Internal Clock is Selected**





## (2) Underflow Operation

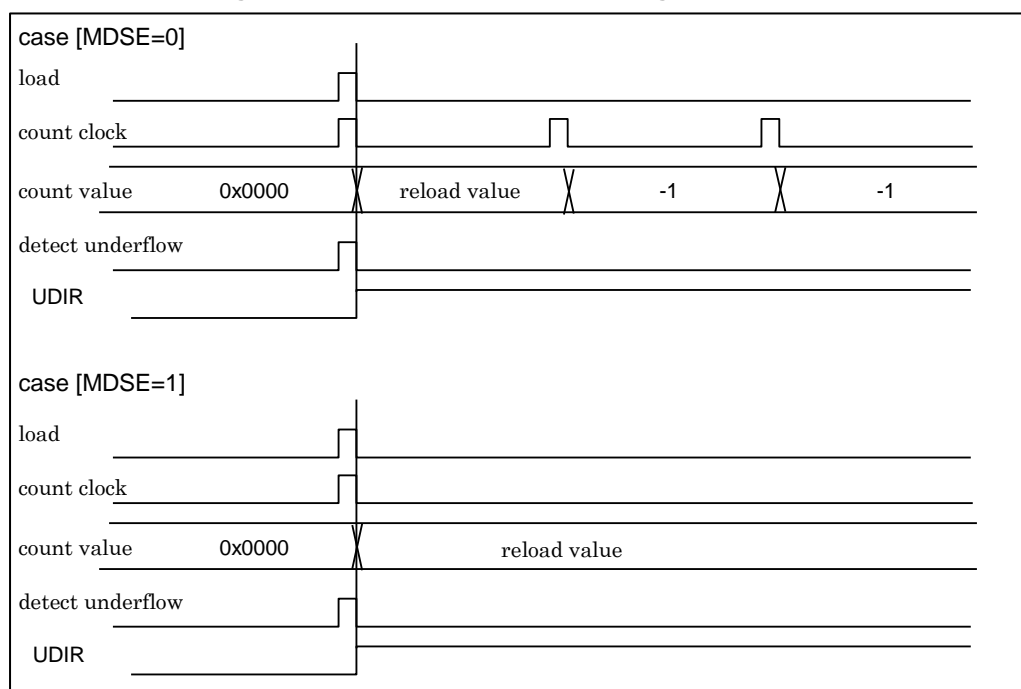
If the counter value changes from 0x0000 to 0xFFFF, an underflow has occurred. Therefore, an underflow occurs at the count of [set value of cycle setting register + 1].

The cycle setting register (PCSR) contents are loaded into the counter when an underflow occurs. If the MDSE bit in the timer control register (TMCR) is "0", the count operation continues. If the MDSE bit is "1", the count operation stops with the loaded counter value unchanged.

The UDIR bit in the status control register (STC) is set to "1" by an underflow. If the UDIE bit is "1" at this time, an interrupt request is generated.

Figure 9-17 shows an underflow operation timing chart.

Figure 9-17 Underflow Operation Timing Chart



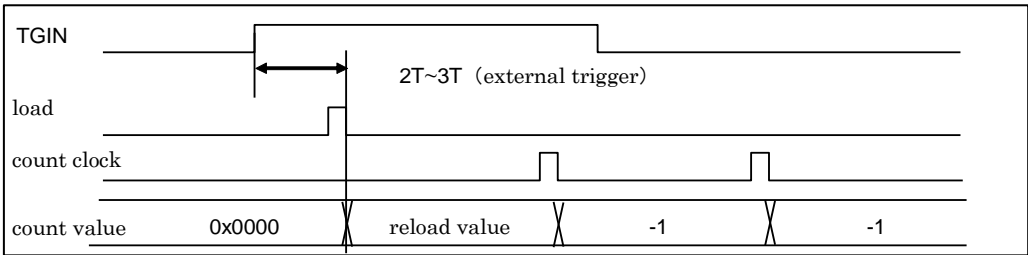


(3) Operation of Input Pin Function

The TGIN pin can be used as trigger input. When a valid edge is input to the TGIN pin, the cycle setting register contents are loaded into the counter, and a count operation begins. For the period from trigger application until the loading of a counter value, the required time is 2T to 3T (T: internal clock cycle).

Figure 9-18 shows trigger input operation where the valid edge specification is a rising edge.

Figure 9-18 Trigger Input Operation

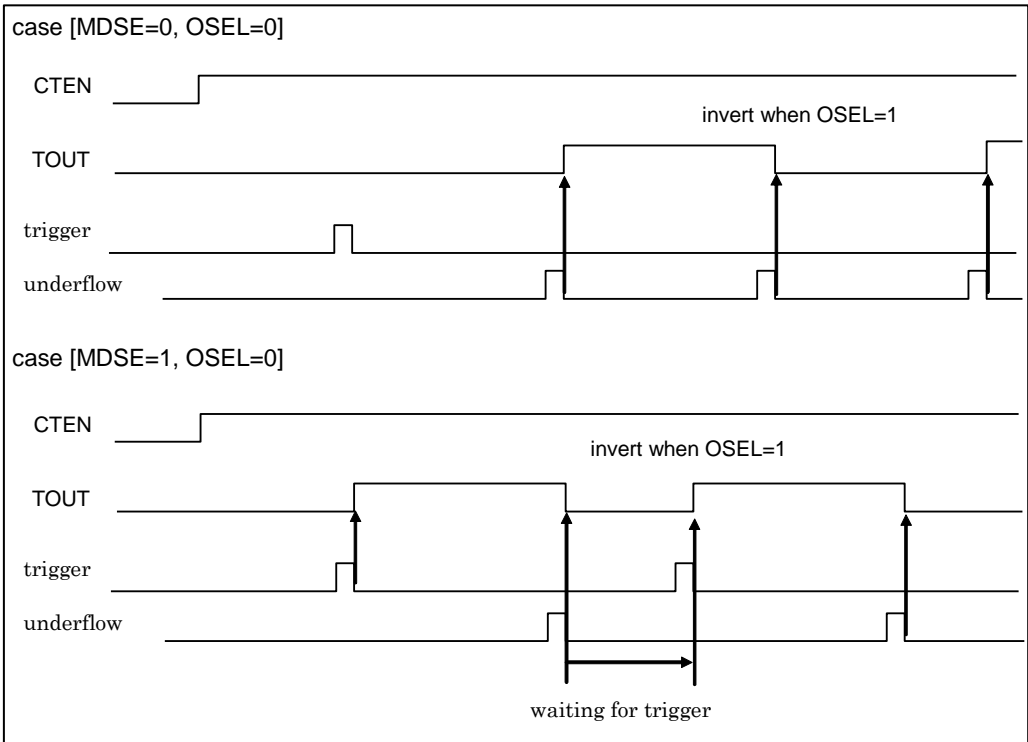


(4) Operation of Output Pin Function

The TOUT output pin functions as toggle output in reload mode and as pulse output in one-shot mode. The toggle output is inverted by an underflow. The pulse output indicates that counting is in progress. The OSEL bit in the timer control register (TMCR) can set the output polarity. If OSEL is "0", the initial value is "0" for toggle output, and "1" is output during counting for one-shot pulse output. If OSEL is "1", the output waveform is inverted.

Figure 9-19 shows an output pin function operation timing chart.

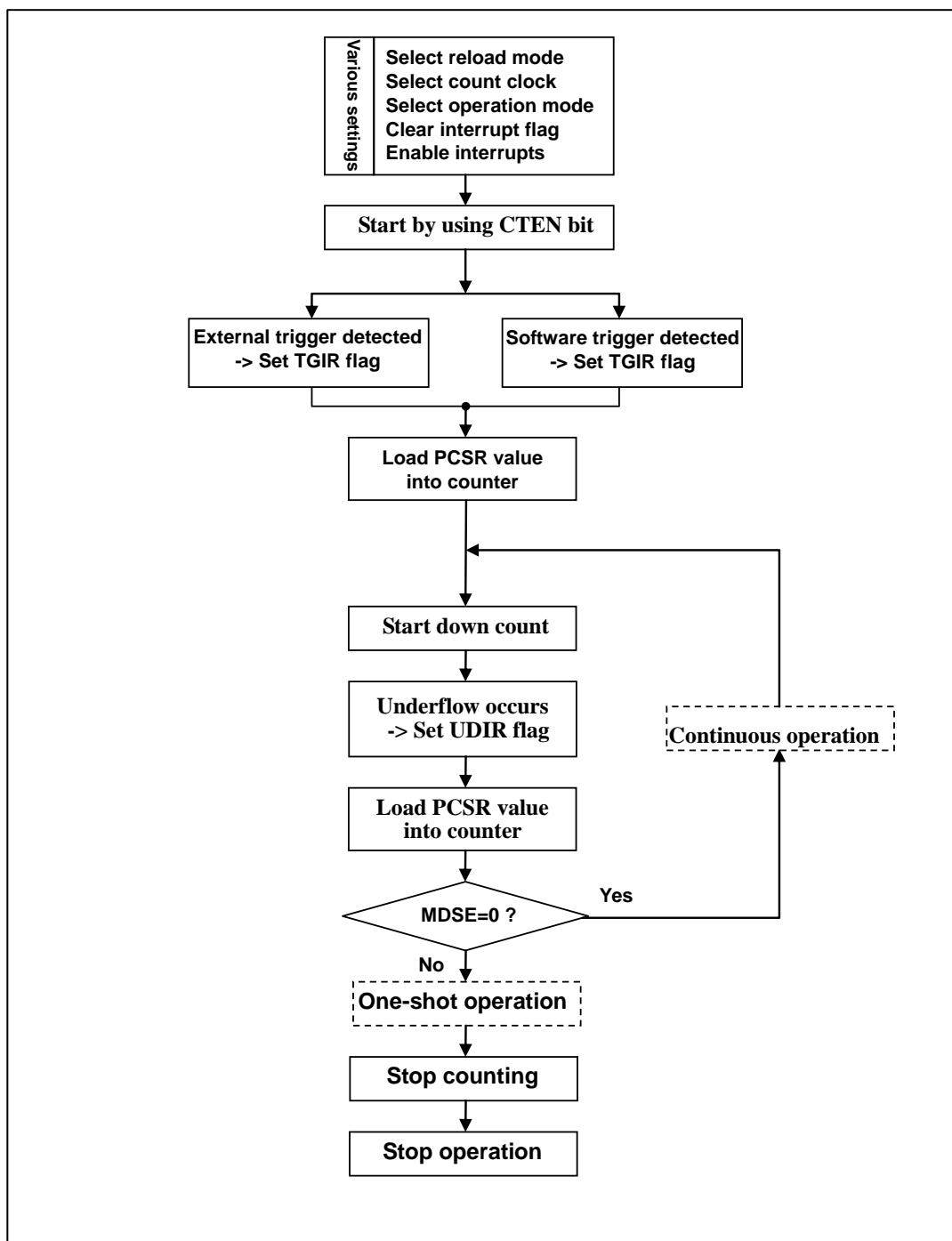
Figure 9-19 Output Pin Function Operation Timing Chart



### 9.3.2. Reload Timer Operation Flow

This section shows the reload timer operation flow.

Figure 9-20 Reload Timer Operation Flow



### 9.3.3. Timer Control Registers (TMCR, TMCR2), Status Control Register (STC), Status Control Clear Register (STCC) and Status Control Set Register (STCS), When the Reload Timer is Selected

The timer control register (TMCR) controls timer operation.

For details on writing to the status control register (STC), see Section "8. Notes on Using".

#### (1) Timer Control Register (Upper Byte of TMCR)

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved	CKS2	CKS1	CKS0	Reserved		EGS1	EGS0
ACCESS_TYPE	R0,W0	R/W	R/W	R/W	R0,W0		R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	00		0	0

[bit15] Reserved: Reserved bit

#### [bit14:12, TMCR2:bit8] CKS3 to CKS0: Count clock selection bits

- These bits select a count clock for the 16-bit down counter.
- Any modification to the count clock is reflected immediately after the setting is changed. Therefore, modify CKS3 to CKS0 while counting is stopped (CTEN="0"). However, note that the bits can be modified at the same time as "1" is written to the CTEN bit.

Value				Description
CKS3	CKS2	CKS1	CKS0	
0	0	0	0	$\phi$
0	0	0	1	$\phi / 4$
0	0	1	0	$\phi / 16$
0	0	1	1	$\phi / 128$
0	1	0	0	$\phi / 256$
0	1	0	1	External clock (rising-edge event)
0	1	1	0	External clock (falling-edge event)
0	1	1	1	External clock (both-edges event)
1	0	0	0	$\phi / 512$
1	0	0	1	$\phi / 1024$
1	0	1	0	$\phi / 2048$
Other than above				Setting prohibited

$\phi$ : peripheral clock

[bit11:10] Reserved: Reserved bits

#### [bit9:8] EGS1 to EGS0: Trigger input edge selection bits

- These bits select the valid edge for an input waveform as an external start factor, and they set trigger conditions.
- For the initial value or the "00" setting, no valid edge is selected for an input waveform, so no external waveform causes a start.
- Modify EGS1 and EGS0 while counting is stopped (CTEN="0"). However, note that the bits can be modified at the same time as "1" is written to the CTEN bit.



Value	Description
00	Trigger input invalid
01	External trigger(rising edge)
10	External trigger(falling edge)
11	External trigger(both edges)

**Note:**

- If "1" is written to the STRG bit, the software trigger becomes valid regardless of the EGS1 and EGS0 settings.

## (2) Timer Control Register (Lower Byte of TMCR)

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	T32	FMD2	FMD1	FMD0	OSEL	MDSE	CTEN	STRG
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R0,W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

### [bit7] T32: 32-bit timer selection bit

- This bit selects the 32-bit timer function.
- If the reload timer function is selected with "011" set in the FMD2 to FMD0 bits, setting the T32 bit to "1" results in operation in 32-bit timer mode.
- Modify these bits while the timer is stopped (CTEN="0"). However, note that the bit can be modified at the same time as "1" is written to the CTEN bit. (See Section "4. 32-bit Mode Operation".)

Value	Description
0	16-bit timer mode
1	32-bit timer mode

### [bit6:4] FMD2 to FMD0: Timer function selection bits

- These bits select the timer function.
- If "011" is set in the FMD2 to FMD0 bits, the reload timer function is selected.
- Modify these bits while the timer is stopped (CTEN="0"). However, note that the bits can be modified at the same time as "1" is written to the CTEN bit.

Value	Description
0 0 0	Reset mode
0 0 1	16-bit PWM timer
0 1 0	16-bit PPG timer
0 1 1	16/32-bit reload timer
1 0 0	16/32-bit PWC timer
Other than above	Setting prohibited

### [bit3] OSEL: Output polarity specification bit

- This bit selects either normal or inverse output as the timer output level.
- When combined with the mode selection bit (bit2 MDSE), this bit generates an output waveform as follows.

MDSE	OSEL	Output Waveform
0	0	Toggle output of "L" at the count start time
0	1	Toggle output of "H" at the count start time
1	0	Rectangular output of "H" during counting
1	1	Rectangular output of "L" during counting

Value	Description
0	Normal polarity
1	Inverse polarity



**[bit2] MDSE: Mode selection bit**

- If the MDSE bit is set to "0", reload mode is selected. The cycle setting register (PCSR) value is loaded into the counter at the same time that the count value underflows from "0x0000" to "0xFFFF", and the count operation continues.
- If the MDSE bit is set to "1", one-shot mode is selected. When the count value underflows from "0x0000" to "0xFFFF", operation stops.
- Modify these bits while the timer is stopped (CTEN="0"). However, note that the bits can be modified at the same time as "1" is written to the CTEN bit.

Value	Description
0	Reload mode
1	One-shot mode

**[bit1] CTEN: Timer enable bit**

- This bit enables operation of the down counter.
- If "0" is written to this bit when counter operation is enabled (CTEN bit is "1"), the counter stops.

Value	Description
0	Stop operation.
1	Enable operation.

**[bit0] STRG: Software trigger bit**

- If "1" is written to the STRG bit when the CTEN bit is "1", a software trigger is applied.
- The read value of the STRG bit is always "0".

**Notes:**

- *Even if "1" is written to the CTEN and STRG bits at the same time, a software trigger is applied.*
- *If "1" is written to the STRG bit, the software trigger becomes valid regardless of the EGS1 and EGS0 settings.*

**(3) Timer Control Register 2 (TMCR2)**

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved							CKS3
ACCESS_TYPE	R0,W0							R/W
PROT_TYPE	-							
INITIAL_VALUE	0000000							0

Note: This register is located in the upper byte of the STC register.

**[bit15:9] Reserved: Reserved bits**

**[bit8] CKS3: Count clock selection bit**

See Section "Count clock selection bits" in "(1) Timer Control Register (Upper Byte of TMCR)".





#### (4) Status Control Register (STC)

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved	TGIE	Reserved	UDIE	Reserved	TGIR	Reserved	UDIR
ACCESS_TYPE	R0,W0	R/W	R0,W0	R/W	R0,W0	R,W	R0,W0	R,W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

Note: The TMCR2 register is located in the upper byte of this register.

#### [bit7] Reserved: Reserved bit

#### [bit6] TGIE: Trigger interrupt request enable bit

- This bit controls interrupt requests of the trigger interrupt request bit (bit2 TGIR).
- If the TGIE bit is enabled and the TGIR bit is set to "1", an interrupt request is issued to the CPU.
- Writing "1" to the STCC:TGIEC bit clears this bit.
- Writing "1" to the STCS:TGIES bit sets this bit.

Value	Description
0	Disable interrupt requests.
1	Enable interrupt requests.

#### [bit5] Reserved: Reserved bit

#### [bit4] UDIE: Underflow interrupt request enable bit

- This bit controls interrupt requests of the underflow interrupt request bit (bit0 UDIR).
- If the UDIE bit is enabled and the UDIR bit is set to "1", an interrupt request is issued to the CPU.
- Writing "1" to the STCC:UDIEC bit clears this bit.
- Writing "1" to the STCS:UDIES bit sets this bit.

Value	Description
0	Disable interrupt requests.
1	Enable interrupt requests.

#### [bit3] Reserved: Reserved bit

**[bit2] TGIR: Trigger interrupt request bit**

- The TGIR bit is set to "1" when a software trigger or trigger input is detected.
- Writing "0" clears the TGIR bit.
- Writing "1" to the STCC:TGIRC bit clears this bit.
- Writing "1" to the TGIR bit has no effect on the bit value.

Value	Description
0	The interrupt factor is cleared.
1	Detect the interrupt factor.

**[bit1] Reserved: Reserved bit**

**[bit0] UDIR: Underflow interrupt request bit**

- The UDIR bit is set to "1" when the count value underflows and changes from "0x0000" to "0xFFFF".
- Writing "0" clears the UDIR bit.
- Writing "1" to the STCC:UDIRC bit clears this bit.
- Writing "1" to the UDIR bit has no effect on the bit value.

Value	Description
0	The interrupt factor is cleared.
1	Detect the interrupt factor.



### (5) Status Control Clear Register (STCC)

BITS_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved	TGIEC	Reserved	UDIEC	Reserved	TGIRC	Reserved	UDIRC
ACCESS_TYPE	R0,W0	R0,W	R0,W0	R0,W	R0,W0	R0,W	R0,W0	R0,W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

**[bit7] Reserved: Reserved bit**

#### **[bit6] TGIEC: Trigger interrupt request enable clear bit**

If "1" is written to this bit, the STC:TGIE bit is cleared to "0".

Value	Description
0	Invalid
1	Clear the TGIE bit.

**[bit5] Reserved: Reserved bit**

#### **[bit4] UDIEC: Underflow interrupt request enable clear bit**

If "1" is written to this bit, the STC:UDIE bit is cleared to "0".

Value	Description
0	Invalid
1	Clear the UDIE bit.

**[bit3] Reserved: Reserved bit**

#### **[bit2] TGIRC: Trigger interrupt request clear bit**

If "1" is written to this bit, the STC:TGIR bit is cleared to "0".

Value	Description
0	Invalid
1	Clear the TGIR bit.

**[bit1] Reserved: Reserved bit**

#### **[bit0] UDIRC: Underflow interrupt request clear bit**

- If "1" is written to this bit, the STC:UDIR bit is cleared to "0".
- The read value of this bit is always "0".

Value	Description
0	Invalid
1	Clear the UDIR bit.

### (6) Status Control Set Register (STCS)

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved	TGIES	Reserved	UDIES	Reserved			
ACCESS_TYPE	R0,W0	R0,W	R0,W0	R0,W	R0,W0			
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0000			

**[bit7] Reserved: Reserved bit**

**[bit6] TGIES: Trigger interrupt request enable set bit**

- If "1" is written to this bit, the STC:TGIE bit is set to "1".
- The read value of this bit is always "0".

Value	Description
0	Invalid
1	Set the TGIE bit.

**[bit5] Reserved: Reserved bit**

**[bit4] UDIES: Underflow interrupt request enable set bit**

- If "1" is written to this bit, the STC:UDIE bit is set to "1".
- The read value of this bit is always "0".

Value	Description
0	Invalid
1	Set the UDIE bit.

**[bit3:0] Reserved: Reserved bits**



### 9.3.4. Cycle Setting Register (PCSR)

The cycle setting register (PCSR) is a register that retains the initial count value. In 32-bit mode, this value for the even-numbered channel is the initial value of the lower 16-bit count. This value for the odd-numbered channel is the initial value of the upper 16-bit count. The initial value at the reset time is undefined. Be sure to access this register with 16-bit data transfer instructions.

BIT_OFFSET	15-0
BIT_NAME	PCSR
ACCESS_TYPE	R/W
PROT_TYPE	-
INITIAL_VALUE	XXXXXXXX_XXXXXXXX

#### [bit15:0] PCSR[15:0]: Cycle setting register

These bits compose the register used to set a cycle. An underflow causes a transfer to the timer register.

- Use 16-bit data access for the PCSR register.
- After configuring the reload timer function with the FMD2 to FMD0 bits in the TMCR register, set a cycle in the PCSR register.
- To write data to the PCSR register in 32-bit mode, access first the upper 16-bit data (data for the odd-numbered channel) and then the lower 16-bit data (data for the even-numbered channel).

### 9.3.5. Timer Register (TMR)

The timer register (TMR) is a register that can read the count value of a timer. In 32-bit mode, this value for the even-numbered channel is the value of the lower 16-bit count. This value for the odd-numbered channel is the value of the upper 16-bit count. The initial value is undefined.

Be sure to read this register with 16-bit data transfer instructions.

BIT_OFFSET	15-0
BIT_NAME	TMR
ACCESS_TYPE	R,WX
PROT_TYPE	-
INITIAL_VALUE	XXXXXXXX_XXXXXXXX

#### [bit15:0] TMR[15:0]: Timer register

The timer register can read the value of the 16-bit down counter.

- Use 16-bit data access for the TMR register.
- To read the TMR register in 32-bit mode, access first the lower 16-bit data (data for the even-numbered channel) and then the upper 16-bit data (data for the odd-numbered channel).



## 9.4. PWC Timer Function

Only one of the following timer functions can be selected for the base timer in the FMD2 to FMD0 bit settings in the timer control register (TMCR): 16-bit PWM timer, 16-bit PPG timer, 16/32-bit reload timer, and 16/32-bit PWC timer. This section explains the timer function with the PWC setting.

1. Operations of PWC Timer
2. Timer Control Registers (TMCR, TMCR2), Status Control Register (STC), Status Control Clear Register (STCC) and Status Control Set Register (STCS) When the PWC Timer is Selected
3. Data Buffer Register (DTBF)







## (2) Selection of Count Clock

Eight types of count clocks can be selected for the counter through the setting of TMCR2 register bit8 (CKS3) and TMCR register bit14 to 12 (CKS2, CKS1, and CKS0).

The following count clocks can be selected.

TMCR2 and TMCR Registers CKS3, CKS2, CKS1, and CKS0 Bits	Internal Count Clock Selected
0000	Internal clock [initial value]
0001	Internal clock divided by 4
0010	Internal clock divided by 16
0011	Internal clock divided by 128
0100	Internal clock divided by 256
0101	Setting prohibited
0110	
0111	
1000	Internal clock divided by 512
1001	Internal clock divided by 1024
1010	Internal clock divided by 2048
Other than above	Setting prohibited

The initial value selected after a reset is the internal clock.  
Be sure to select a count clock before starting the counter.

### (3) Selection of Operation Mode

To select each operation mode/measurement mode, set TMCR.

Operation mode setting ... TMCR bit10 to 8: EGS2, EGS1, EGS0 (Select a measurement edge.)

Measurement mode setting ... TMCR bit2: MDSE (Select single measurement/continuous measurement.)

The following table lists a selection of operation modes.

Operation Mode		MDSE	EGS2	EGS1	EGS0
Rising to falling H pulse width measurement	Continuous measurement mode: Buffer enabled	0	0	0	0
	Single measurement mode: Buffer disabled	1	0	0	0
Rising to rising Cycle measurement between rising edges	Continuous measurement mode: Buffer enabled	0	0	0	1
	Single measurement mode: Buffer disabled	1	0	0	1
Falling to falling Cycle measurement between falling edges	Continuous measurement mode: Buffer enabled	0	0	1	0
	Single measurement mode: Buffer disabled	1	0	1	0
Measurement between rising or falling and rising or falling	Continuous measurement mode: Buffer enabled	0	0	1	1
	Single measurement mode: Buffer disabled	1	0	1	1
Falling to rising L pulse width measurement	Continuous measurement mode: Buffer enabled	0	1	0	0
	Single measurement mode: Buffer disabled	1	1	0	0
Setting prohibited		0	1	0	1
		1	1	0	1
		0	1	1	0
		1	1	1	0
		0	1	1	1
		1	1	1	1

The initial values selected after a reset are H pulse width measurement and continuous measurement mode.  
Be sure to select an operation mode before starting the counter.



#### (4) Starting and Stopping Pulse Width Measurement

To start/restart/forcibly stop each operation, set bit1 (CTEN bit) in TMCR.

Pulse width measurement is started/restarted by the writing of "1" to the CTEN bit and forcibly stopped by the writing of "0" to the CTEN bit.

CTEN	Function
1	Start/Restart pulse width measurement.
0	Stop pulse width measurement.

#### (5) Post-start Operation

The operations after the start of pulse width measurement mode do not include counting until the measurement start edge is input. After the detection of the measurement start edge, the 16-bit up counter starts counting from "0x0001".

#### (6) Restart

After the start, a repeated start performed during operation (writing "1" again in a state where the CTEN bit is "1") is called a "restart." The operation in any such restart is described below.

- In the measurement start edge wait state:  
There is no effect on operation.
- During measurement:  
The count is cleared to "0x0000". Then, the measurement start edge wait state begins again. If the measurement end edge is detected at the same time as a restart, the measurement end flag (EDIR) is set to "1". Then, if the mode is continuous measurement mode, the measurement results are transferred to DTBF.

#### (7) About Stopping

In single measurement mode, the count operation is automatically stopped by a counter overflow or the end of measurement, so stopping it does not need to be a concern. In continuous measurement mode, to stop counting before counting is automatically stopped, you need to forcibly stop it.

#### (8) Counter Clearing and Initial Value

The 16-bit up counter is cleared to "0x0000" in the following cases.

- Upon a reset
- When "1" is written to bit1 (CTEN bit) in TMCR (even including restart times)

The 16-bit up counter is initialized to "0x0001" in the following case.

- When the measurement start edge is detected

## **(9) Pulse Width Measurement Operation Details**

### **a) Single Measurement and Continuous Measurement**

The modes for pulse width measurement are a mode where measurement is 1-time only and a mode where measurement is continuous. Each mode is selected with the MDSE bit in TMCR. (See Section "(3) Selection of Operation ".) The modes differ as described below.

Single measurement mode:

The first input of the measurement end edge stops the counting by the counter and sets the measurement end flag (EDIR) in STC to "1". No subsequent measurements are made.

However, if a restart is performed at the same time, the mode enters the measurement start wait state.

Continuous measurement mode:

The input of the measurement end edge stops the counting by the counter and sets the measurement end flag (EDIR) in STC to "1". The counting remains stopped until the measurement start edge is input again. The counter is initialized to "0x0001" and measurement begins when the measurement start edge is input again. The measurement results of the counter are transferred to DTBF at the measurement end time.

Be sure to select/change the measurement mode while the counter is stopped.

### **b) Measurement Result Data**

The handling of measurement results and counter values and the function of DTBF differ between single measurement mode and continuous measurement mode. The measurement results from each mode differ as described below.

Single measurement mode:

If DTBF is read during operation, the count value being measured is obtained.

If DTBF is read after measurement ends, the measurement result data is obtained.

Continuous measurement mode:

The measurement results of the counter are transferred to DTBF at the measurement end time.

The immediately preceding measurement results are obtained when DTBF is read, and the last measurement results are retained even during a measurement operation. The count value being measured cannot be read.

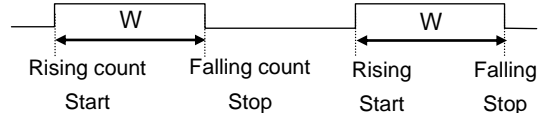
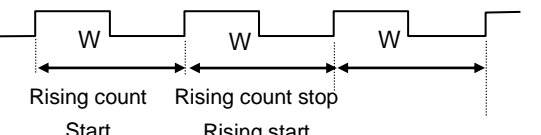
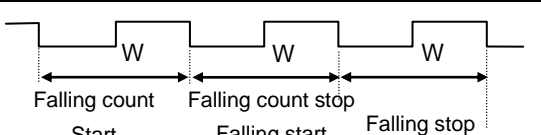
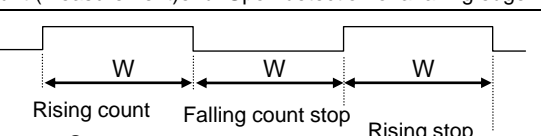
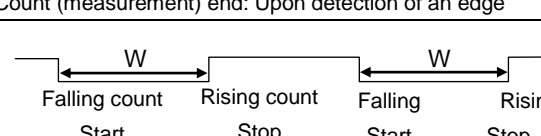
In continuous measurement mode, if the next measurement ends before the measurement results are read, the last measurement results are overwritten by the new measurement results. In such cases, the error flag (ERR) is set to "1" in STC. The error flag (ERR) is automatically cleared when DTBF is read.



### c) Measurement Modes and Count Operations

The measurement mode can be selected from the following five types, depending on where the input pulse is measured. The following table explains them.

**Table 9-1 Measurement Modes and Count Operations**

Measurement Mode	EGS2, 1, 0	Measurement Description (W: Pulse Width Measured)
H pulse width measurement	000	 <p>The width of the "H" period is measured. Count (measurement)start: Upon detection of a rising edge Count (measurement)end: Upon detection of a falling edge</p>
Cycle measurement between rising edges	001	 <p>The cycle between rising edges is measured. Count (measurement)start: Upon detection of a rising edge Count (measurement) end: Upon detection of a rising edge</p>
Cycle measurement between falling edges	010	 <p>The cycle between falling edges is measured. Count (measurement) start: Upon detection of a falling edge Count (measurement)end: Upon detection of a falling edge</p>
Pulse width measurement between all edges	011	 <p>The width between continuously input edges is measured. Count (measurement) start: Upon detection of an edge Count (measurement) end: Upon detection of an edge</p>
L pulse width measurement	100	 <p>The width of the "L" period is measured. Count (measurement) start: Upon detection of a falling edge Count (measurement) end: Upon detection of a rising edge</p>

In either measurement mode, after the counter is cleared to "0x0000" by the start of measurement, the counter does not do any counting until the measurement start edge is input. From the input of the measurement start edge, the counter continues counting up for each count clock until the measurement end edge is input.

For measurements such as pulse width measurement and cycle measurement between all edges in continuous measurement mode, the end edge is the next measurement start edge.

#### d) Pulse Width/Cycle Calculation Method

The method of calculating the pulse width/cycle from the obtained measurement result data in DTBF after measurement ends is shown below.

$$TW = n \times t$$

TW: Measured pulse width/cycle

n: Measurement result data in DTBF

t: Count clock cycle

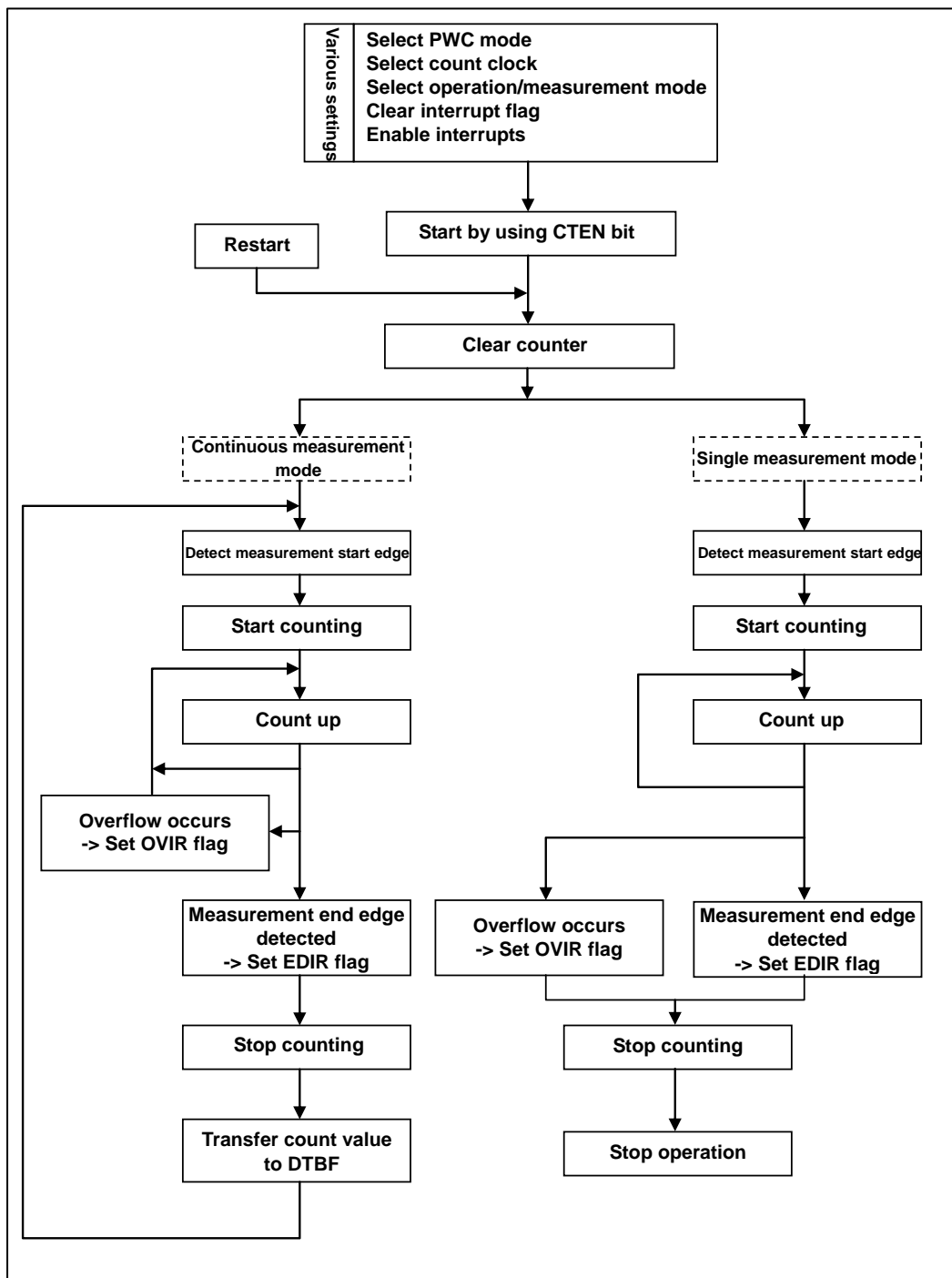
#### e) Interrupt Request Generation

The following two interrupt requests can be generated.

- Interrupt request due to a counter overflow
- If counting up during measurement causes an overflow, the overflow flag (OVIR) is set to "1". Furthermore, if overflow interrupt requests are enabled, an interrupt request is generated.
- Interrupt request due to measurement end
- If the measurement end edge is detected, the measurement end flag (EDIR) in STC is set to "1". Furthermore, if measurement end interrupt requests are enabled, an interrupt request is generated.
- The measurement end flag (EDIR) is automatically cleared by the reading of the measurement results, DTBF.

f) Pulse Width Measurement Operation Flow

Figure 9-23 Pulse Width Measurement Operation Flow



### 9.4.2. Timer Control Registers (TMCR, TMCR2), Status Control Register (STC), Status Control Clear Register (STCC) and Status Control Set Register (STCS) When the PWC Timer is Selected

The timer control register (TMCR) controls timer operation.

For details on writing to the status control register (STC), see Section "8. Notes on Using".

#### (1) Timer Control Register (Upper Byte of TMCR)

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved	CKS2	CKS1	CKS0	Reserved	EGS2	EGS1	EGS0
ACCESS_TYPE	R0,W0	R/W	R/W	R/W	R0,W0	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit15] Reserved: Reserved bit

#### [bit14:12, TMCR2:bit8] CKS3 to CKS0: Count clock selection bits

- These bits select a count clock for the 16-bit down counter.
- Any modification to the count clock is reflected immediately after the setting is changed. Therefore, modify CKS3 to CKS0 while counting is stopped (CTEN="0"). However, note that the bits can be modified at the same time as "1" is written to the CTEN bit.

Value				Description
CKS3	CKS2	CKS1	CKS0	
0	0	0	0	$\phi$
0	0	0	1	$\phi / 4$
0	0	1	0	$\phi / 16$
0	0	1	1	$\phi / 128$
0	1	0	0	$\phi / 256$
0	1	0	1	Setting prohibited
0	1	1	0	
0	1	1	1	
1	0	0	0	$\phi / 512$
1	0	0	1	$\phi / 1024$
1	0	1	0	$\phi / 2048$
Other than above				Setting prohibited

$\phi$ : peripheral clock

[bit11] Reserved: Reserved bit





**[bit10:8] EGS2 to EGS0: Measurement edge selection bits**

- These bits set a measurement edge condition.
- Modify EGS2 to EGS0 while counting is stopped (CTEN="0"). However, note that the bits can be modified at the same time as "1" is written to the CTEN bit.

Value	Description
000	"H" pulse width measurement (rising to falling)
001	Cycle measurement between rising edges (rising to rising)
010	Cycle measurement between falling edges (falling to falling)
011	Pulse width measurement between all edges (rising or falling to falling or rising)
100	"L" pulse width measurement (falling to rising)
101	Setting prohibited
110	
111	

## (2) Timer Control Register (Lower Byte of TMCR)

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	T32	FMD2	FMD1	FMD0	Reserved	MDSE	CTEN	Reserved
ACCESS_TYPE	R/W	R/W	R/W	R/W	R0,W0	R/W	R/W	R0,W0
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

### [bit7] T32: 32-bit timer selection bit

- This bit selects the 32-bit timer function.
- If the PWC function is selected with "100" set in the FMD2 to FMD0 bits, setting the T32 bit to "1" results in operation in 32-bit PWC mode.
- Modify these bits while the timer is stopped (CTEN="0"). However, note that the bit can be modified at the same time as "1" is written to the CTEN bit. (See Section "4. 32-bit Mode Operation".)

Value	Description
0	16-bit timer mode
1	32-bit timer mode

### [bit6:4] FMD2 to FMD0: Timer function selection bits

- These bits select the timer function.
- If "100" is set in the FMD2 to FMD0 bits, the PWC timer function is selected.
- Modify these bits while the timer is stopped (CTEN="0"). However, note that the bits can be modified at the same time as "1" is written to the CTEN bit.

Value			Description
0	0	0	Reset mode
0	0	1	16-bit PWM timer
0	1	0	16-bit PPG timer
0	1	1	16/32-bit reload timer
1	0	0	16/32-bit PWC timer
1	0	1	Setting prohibited
1	1	0	
1	1	1	

### [bit3] Reserved: Reserved bit



**[bit2] MDSE: Mode selection bit**

- Modify these bits while the timer is stopped (CTEN="0"). However, note that the bits can be modified at the same time as "1" is written to the CTEN bit.

Value	Description
0	Continuous measurement mode (buffer register enabled)
1	Single measurement mode (stop after 1 measurement)

**[bit1] CTEN: Timer enable bit**

- This bit enables the starting or restarting of the up counter.
- If "1" is written with the counter in the operation enabled state (CTEN bit is "1"), a restart is assumed, the counter is cleared, and operation enters the measurement start edge wait state.
- If "0" is written to this bit when counter operation is enabled (CTEN bit is "1"), the counter stops.
- After measurement ends in single measurement mode, CTEN is cleared.

Value	Description
0	Stop operation.
1	Enable operation.

**[bit0] Reserved: Reserved bit**

**(3) Timer Control Register 2 (TMCR2)**

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved							CKS3
ACCESS_TYPE	R0,W0							R/W
PROT_TYPE	-							
INITIAL_VALUE	0000000							0

Note: This register is located in the upper byte of the STC register.

**[bit15:9] Reserved: Reserved bits**

**[bit8] CKS3: Count clock selection bit**

See Section "Count clock selection bits" in "(1) Timer Control Register (Upper Byte of TMCR)".



#### (4) Status Control Register (STC)

BITS_OFFSET	7	6	5	4	3	2	1	0
BITS_NAME	ERR	EDIE	Reserved	OVIE	Reserved	EDIR	Reserved	OVIR
ACCESS_TYPE	R,WX	R/W	R0,W0	R/W	R0,W0	R,WX	R0,W0	R,W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

Note: The TMCR2 register is located in the upper byte of this register.

##### [bit7] ERR: Error flag bit

- This bit is a flag indicating that, in continuous measurement mode, the next measurement has ended before the measurement results in the DTBF register have been read. In this case, the DTBF register values are updated with the new measurement results, so the immediately preceding measurement results are lost.
- Measurement continues regardless of the ERR bit value.
- The ERR bit is read-only. Writing to the bit has no effect on the bit value.
- Reading the measurement results (DTBF) clears the ERR bit.

Value	Description
0	Normal state
1	Overwrite any unread measurement results with the next measurement results.

##### [bit6] EDIE: Measurement end interrupt request enable bit

- This bit controls interrupt requests of the measurement end interrupt request bit (bit2 EDIR).
- If the EDIE bit is enabled and the EDIR bit is set to "1", an interrupt request is issued to the CPU.
- Writing "1" to the STCC:EDIEC bit clears this bit.
- Writing "1" to the STCS:EDIES bit sets this bit.

Value	Description
0	Disable interrupt requests.
1	Enable interrupt requests.

##### [bit5] Reserved: Reserved bit

##### [bit4] OVIE: Overflow interrupt request enable bit

- This bit controls interrupt requests of the overflow interrupt request bit (bit0 OVIR).
- If the OVIE bit is enabled and the OVIR bit is set to "1", an interrupt request is issued to the CPU.
- Writing "1" to the STCC:OVIEC bit clears this bit.
- Writing "1" to the STCS:OVIES bit sets this bit.

Value	Description
0	Disable interrupt requests.
1	Enable interrupt requests.

##### [bit3] Reserved: Reserved bit

**[bit2] EDIR: Measurement end interrupt request bit**

- This bit indicates that measurement has ended by setting the flag to "1" at the end time.
- Reading the measurement results (DTBF) clears the EDIR bit. The EDIR bit is read-only. Writing to the bit has no effect on the bit value.

Value	Description
0	Read measurement results (DTBF).
1	Detect the interrupt factor.

**[bit1] Reserved: Reserved bit**

**[bit0] OVIR: Overflow interrupt request bit**

- The flag is set to "1" when the count value overflows from 0xFFFF to 0x0000.
- Writing "0" clears the OVIR bit.
- Writing "1" to the STCC:OVIRC clears this bit.
- Writing "1" to the OVIR bit has no effect on the bit value.

Value	Description
0	The interrupt factor is cleared.
1	Detect the interrupt factor.



### (5) Status Control Clear Register (STCC)

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved	EDIEC	Reserved	OVIEC	Reserved			OVIRC
ACCESS_TYPE	R0,W0	R0,W	R0,WX	R0,W	R0,W0			R0,W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	000			0

**[bit7] Reserved: Reserved bit**

#### **[bit6] EDIEC: Measurement end interrupt request enable clear bit**

If "1" is written to this bit, the STC:EDIE bit is cleared to "0".

Value	Description
0	Invalid
1	Clear the EDIE bit.

**[bit5] Reserved: Reserved bit**

#### **[bit4] OVIEC: Overflow interrupt request enable clear bit**

If "1" is written to this bit, the STC:OVIE bit is cleared to "0".

Value	Description
0	Invalid
1	Clear the OVIE bit.

**[bit3:1] Reserved: Reserved bits**

#### **[bit0] OVIRC: Overflow interrupt request clear bit**

If "1" is written to this bit, the STC:OVIR bit is cleared to "0".

Value	Description
0	Invalid
1	Clear the OVIR bit.

### (6) Status Control Set Register (STCS)

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved	EDIES	Reserved	OVIES	Reserved			
ACCESS_TYPE	R0,W0	R0,W	R0,W0	R0,W	R0,W0			
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0000			

**[bit7] Reserved: Reserved bit**

#### **[bit6] EDIES: Measurement end interrupt request enable set bit**

If "1" is written to this bit, the STC:EDIE bit is set to "1".

Value	Description
0	Invalid
1	Set the EDIE bit.

**[bit5] Reserved: Reserved bit**

#### **[bit4] OVIES: Overflow interrupt request enable set bit**

If "1" is written to this bit, the STC:OVIE bit is set to "1".

Value	Description
0	Invalid
1	Set the OVIE bit.

**[bit3:0] Reserved: Reserved bits**





### 9.4.3. Data Buffer Register (DTBF)

The data buffer register (DTBF) is a register that can read the measurement value or count value of the PWC timer. In 32-bit mode, the value of the lower 16 bits is read for the even-numbered channel, and the value of the upper 16 bits for the odd-numbered channel.

Be sure to read this register with 16-bit data transfer instructions.

BIT_OFFSET	15-0
BIT_NAME	DTBF
ACCESS_TYPE	R,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000

#### [bit15:0] DTBF[15:0]: Data buffer register

- These bits compose the DTBF register as a read-only register in either continuous measurement mode or single measurement mode. Writing does not change the register value.
- In continuous measurement mode (TMCR bit2 MDSE = "0"), the register is used as a buffer register to store the last measurement results.
- In single measurement mode (TMCR bit2 MDSE = "1"), the up counter is directly accessed with the DTBF register. Reading is allowed even during counting so that the count value can be read. After measurement ends, the measurement results are stored as is.
- Use 16-bit data access for the DTBF register.



## CHAPTER 39: Base Timer I/O Selection Function

This chapter explains the base timer I/O selection function.

---

1. Overview
2. Configuration
3. Operation
4. Registers



## 1. Overview

This section provides an overview of the base timer I/O selection function.

The base timer I/O selection function is a function for selecting a signal I/O method for the base timer by setting an I/O mode.

By switching the timer function, the base timer mounted in a channel can be used as any of the timers described below for each channel. The I/O method for the respective functions can be selected.

The following 7 patterns can be selected for I/O pin connections.

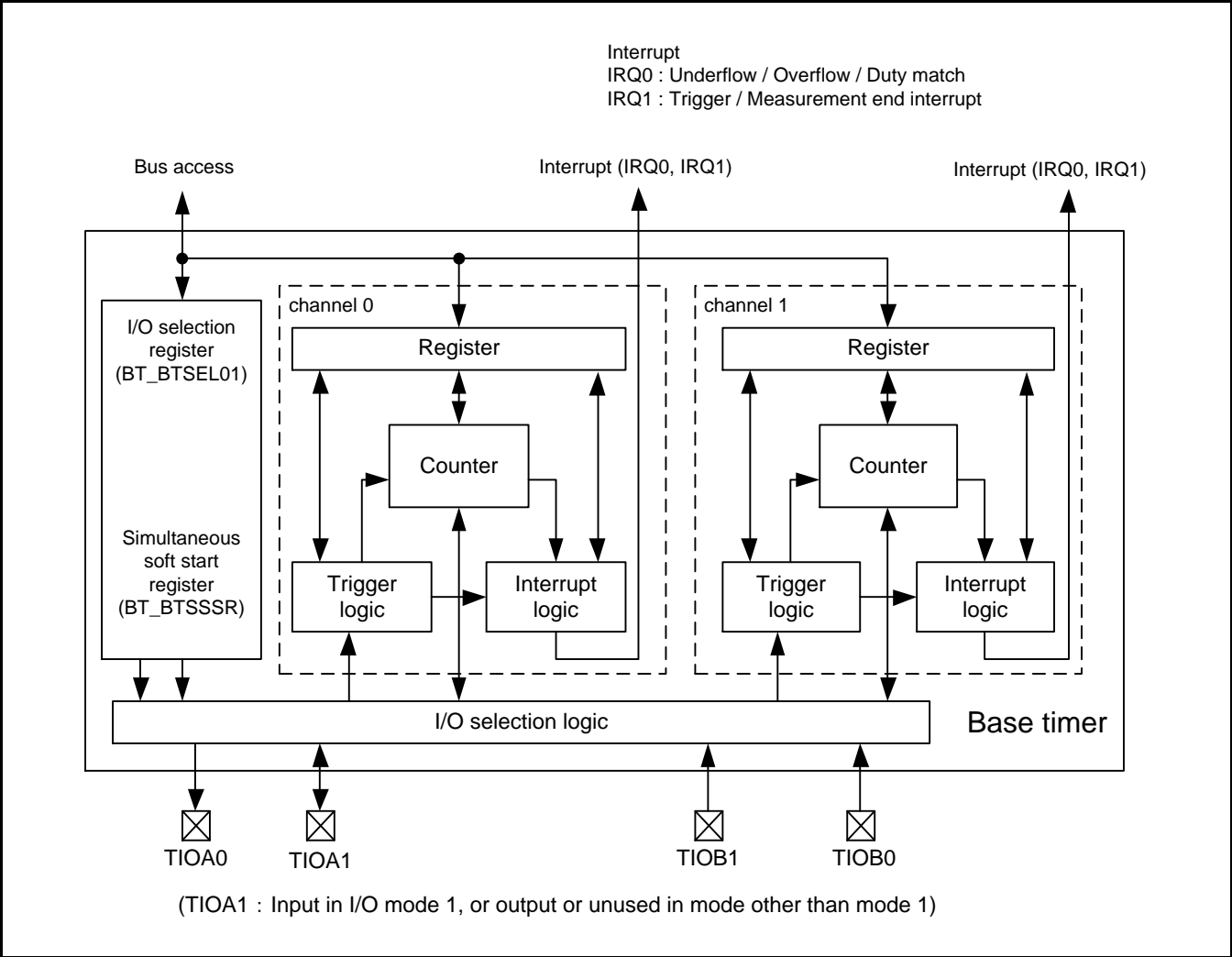
- 16-bit timer standard mode
- 32-bit timer full-function mode
- PPG trigger 2-channel sharing mode
- Timer start/stop mode
- Simultaneous soft start mode
- Timer start/stop and simultaneous soft start mode
- Timer start mode

The 32-bit reload timer and 32-bit PWC timer can be implemented by using 2 channels of the mounted base timer. These 2 channels are channel m (m is an even number) and channel n ( $n=m+1$ ).

2. Configuration

This section explains the configuration of the base timer and I/O selection function.

Figure 2-1 Block Diagram of I/O Selection Function



Configuration for channel 0 and channel 1



### 3. Operation

This section explains base timer I/O assignment.

Before using a timer, make an I/O setting for the base timer by using the I/O mode selection bit (BTSEL01). You can select one of the following 7 modes.

- I/O mode 0: 16-bit timer standard mode  
In this mode, the base timer in 1 channel operates individually and separately from the others.
- I/O mode 1: 32-bit timer full-function mode  
The signals of the even-numbered channels of the base timer are individually assigned to external pins in operation in this mode.
- I/O mode 2: External trigger sharing mode  
This mode enables simultaneous input of external start triggers to the base timers of 2 channels. The base timers of the 2 channels can be started simultaneously.
- I/O mode 4: Timer start/stop mode  
In this mode, an even-numbered channel controls the start/stop of an odd-numbered channel. The odd-numbered channel is started by the rising edge\* of an output signal from the even-numbered channel, and stopped by the falling edge\*.
- I/O mode 5: Simultaneous soft start mode  
In this mode, software starts multiple channels simultaneously.
- I/O mode 6: Soft start timer start/stop mode  
In this mode, an even-numbered channel controls the start/stop of an odd-numbered channel. Software starts the even-numbered channel. The odd-numbered channel is started by the rising edge\* of an output signal from the even-numbered channel, and stopped by the falling edge\*.
- I/O mode 7: Timer start mode  
In this mode, an even-numbered channel controls the start of an odd-numbered channel. The odd-numbered channel is started by the rising edge\* of an output signal from the even-numbered channel.

\*: Use the trigger input selection bit (BTxx\_TMCR:EGS) for the setting.

Figure 3-1 Block Diagram of I/O Mode 0 (16-bit Timer Standard Mode)

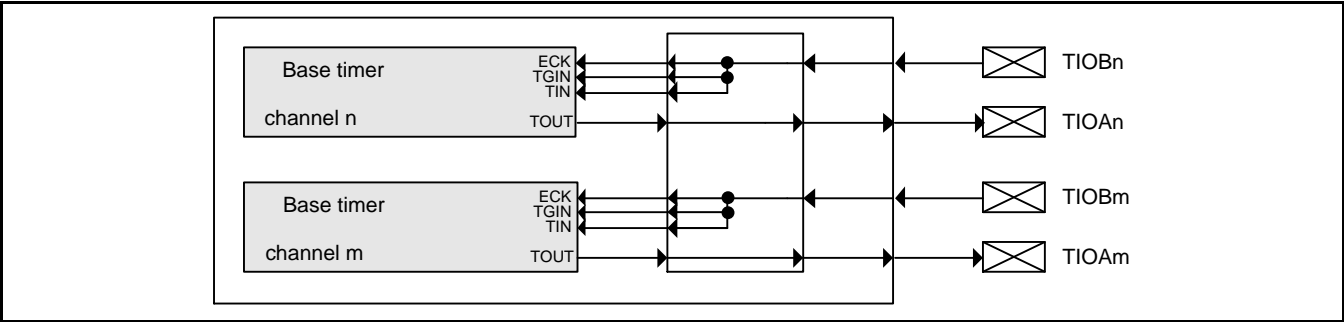


Figure 3-2 Block Diagram of I/O Mode 1 (32-bit Timer Full-function Mode)

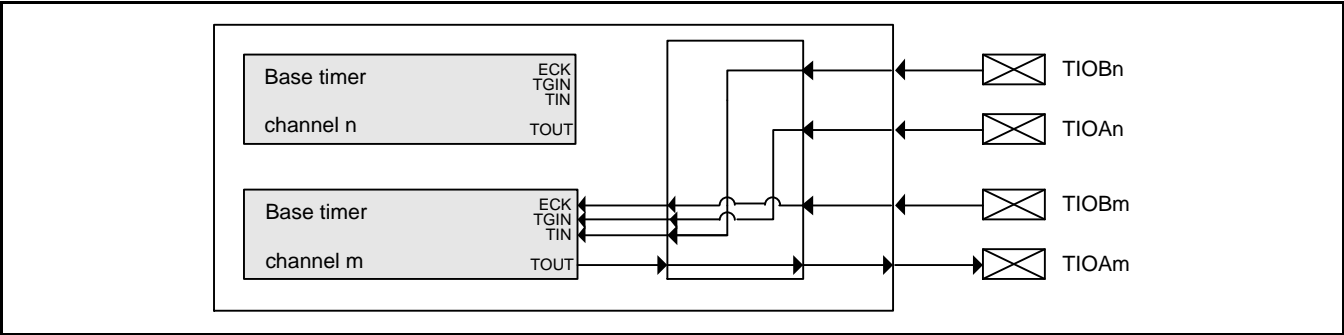


Figure 3-3 Block Diagram of I/O Mode 2 (PPG Trigger 2-channel Sharing Mode)

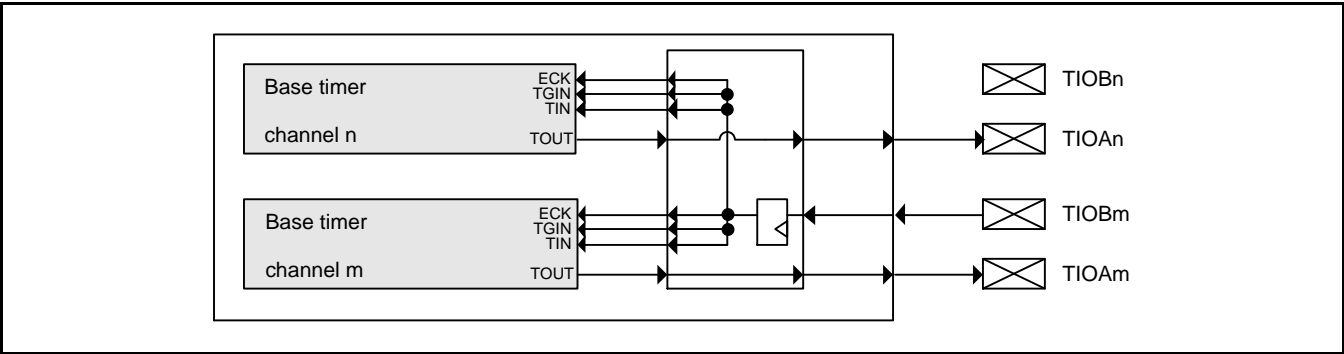


Figure 3-4 Block Diagram of I/O Mode 4 (Timer Start/Stop Mode)

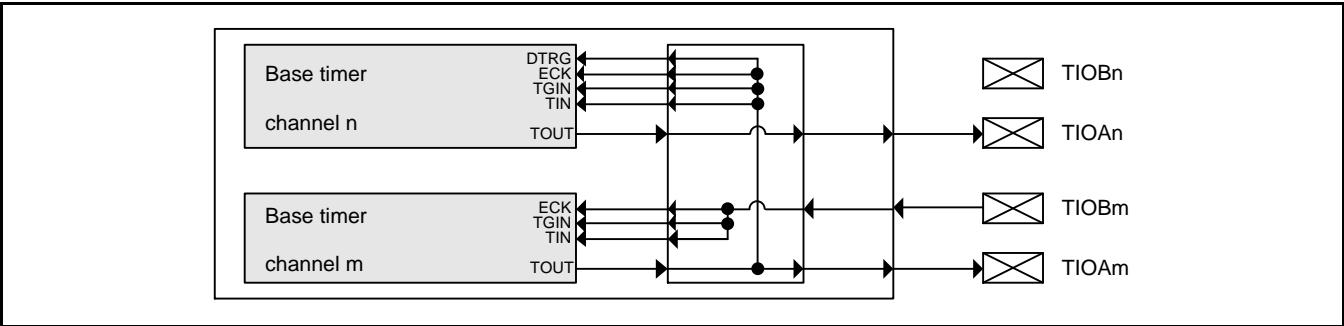


Figure 3-5 Block Diagram of I/O Mode 5 (Simultaneous Soft Start Mode)

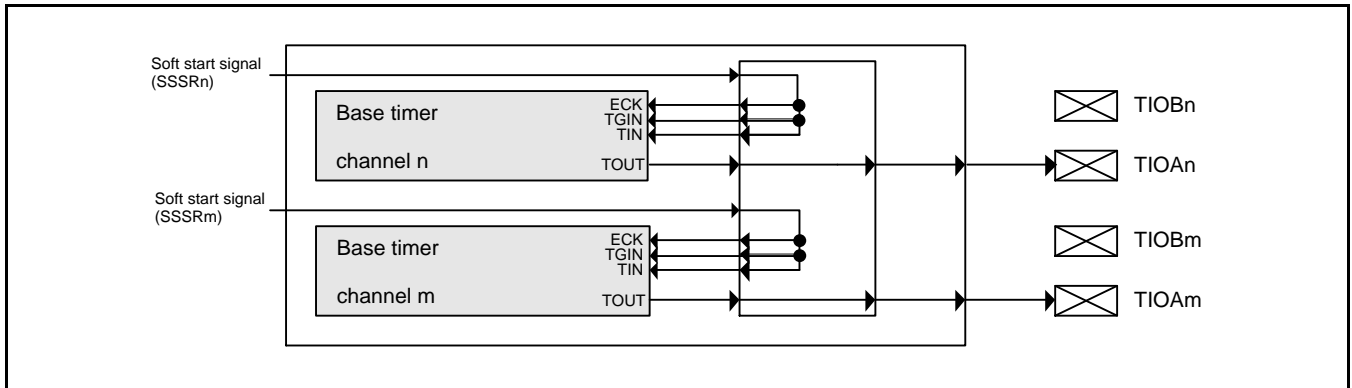


Figure 3-6 Block Diagram of I/O Mode 6 (Timer Start/Stop and Simultaneous Soft Start Mode)

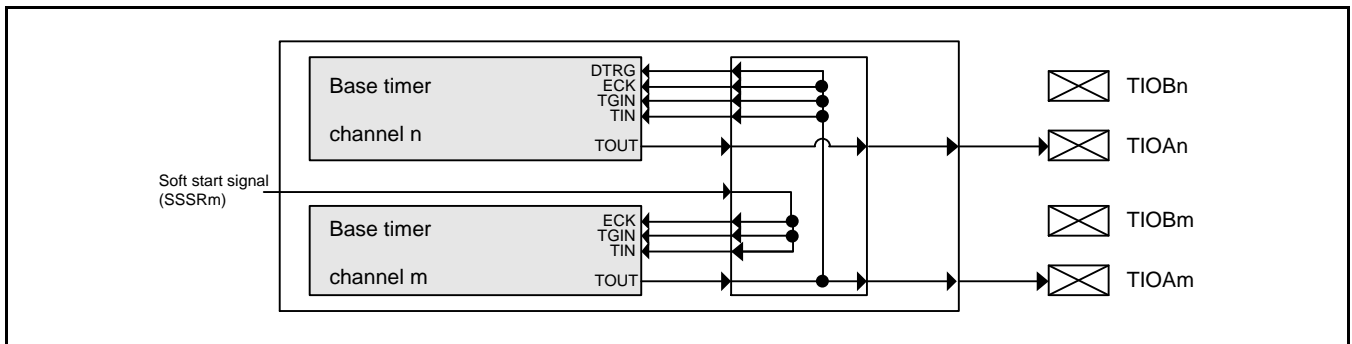
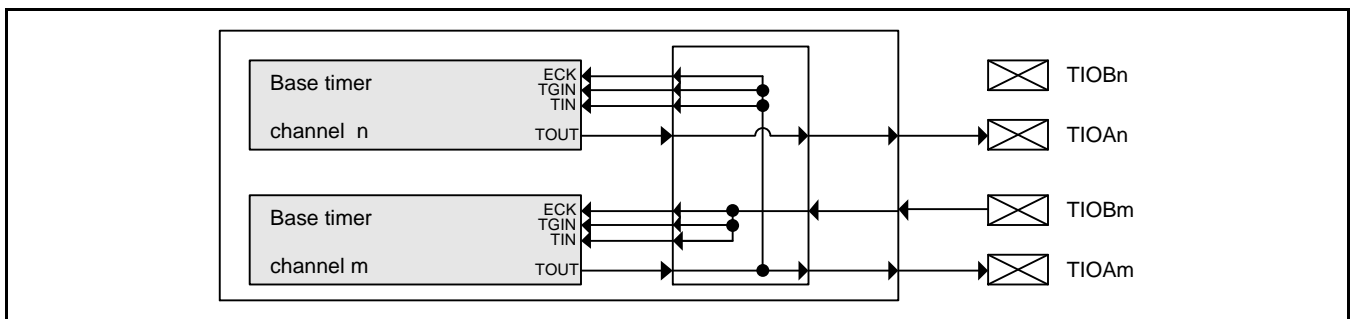


Figure 3-7 Block Diagram of I/O Mode 7 (Timer Start Mode)



**Note:**

- If I/O mode 1 is set, set TIOAn of the corresponding odd-numbered channel to port input mode by using the GPIO setting.

## 4. Registers

This section explains the base timer I/O selection function registers.

**Table 4-1 List of Base Timer I/O Selection Registers**

Abbreviated Register Name	Register Name	See
BT_BTSEL01	I/O Selection Register (channel 0, 1)	4.1
BT_BTSEL23	I/O Selection Register (channel 2, 3)	4.1
BT_BTSEL45	I/O Selection Register (channel 4, 5)	4.1
BT_BTSEL67	I/O Selection Register (channel 6, 7)	4.1
BT_BTSEL89	I/O Selection Register (channel 8, 9)	4.1
BT_BTSEL1011	I/O Selection Register (channel 10, 11)	4.1
BT_BTSSSR	Simultaneous Soft Start Register	4.2





#### 4.1. I/O Selection Registers (BT\_BTSEL01, BT\_BTSEL23, BT\_BTSEL45, BT\_BTSEL67, BT\_BTSEL\_89, BT\_BTSEL1011)

This section shows the bit configuration of the I/O selection registers.

These bits set the I/O modes of the 2 channels of base timer channel m (m is 0 or an even number) and channel n (n=m+1: odd number) for the following connection.

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R1,WX
PROT_TYPE	-
INITIAL_VALUE	11111111_11111111_11111111

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved				BTSEL01			
ACCESS_TYPE	R1,WX				R/W			
PROT_TYPE	-							
INITIAL_VALUE	1111				0000			

**[bit31:4] Reserved: Reserved bits**

**[bit3:0] BTSEL01[3:0]: I/O mode selection bits**

These bits set the I/O modes of the 2 channels of base timer channel m and channel n for the following connection.

Value	Description
0000	I/O mode 0: 16-bit timer standard mode
0001	I/O mode 1: 32-bit timer full-function mode
0010	I/O mode 2: PPG trigger 2-channel sharing mode
0011	Setting prohibited
0100	I/O mode 4: Timer start/stop mode
0101	I/O mode 5: Simultaneous soft start mode
0110	I/O mode 6: Timer start/stop and simultaneous soft start mode
0111	I/O mode 7: Timer start mode
1XXX	Setting prohibited

X: Don't care

**Note:**

- You cannot initialize this register by setting reset mode (TMCR:FMD2 to 0="0b000"). After setting reset mode, rewrite this register.

## 4.2. Simultaneous Soft Start Register (BT\_BTSSSR)

This section shows the bit configuration of the simultaneous soft start register.

These bits represent input signals in I/O modes 5 and 6. This register can be used to generate triggers for all channels simultaneously.

BIT_OFFSET	31-16
BIT_NAME	Reserved
ACCESS_TYPE	R1,WX
PROT_TYPE	-
INITIAL_VALUE	11111111_11111111

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved				SSSR11	SSSR10	SSSR9	SSSR8
ACCESS_TYPE	R1,WX				R1,W	R1,W	R1,W	R1,W
PROT_TYPE	-							
INITIAL_VALUE	1111				1	1	1	1

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	SSSR7	SSSR6	SSSR5	SSSR4	SSSR3	SSSR2	SSSR1	SSSR0
ACCESS_TYPE	R1,W	R1,W	R1,W	R1,W	R1,W	R1,W	R1,W	R1,W
PROT_TYPE	-							
INITIAL_VALUE	1	1	1	1	1	1	1	1

**[bit31:12] Reserved: Reserved bits**

### **[bit11:0] SSSR11 to SSSR0: Simultaneous soft start bits**

These bits represent input signals in I/O modes 5 and 6. For details on connections, see the block diagram of each I/O mode in "3. Operation".

Writing "1" starts the corresponding channel, and writing "0" is invalid. Up to 12 channels with channel numbers 0 to 11 can be started simultaneously.

Value	Description
0	No operation
1	Assigns the "1" pulse to input and starts the corresponding channel.

[x] represents the channel number of the base timer. It is a value from 0 to 11.





## CHAPTER 40: 32-bit Free-run Timer

This chapter describes the functions of the 32-bit free-run timer.

---

1. Overview
2. Configuration
3. Operation
4. Registers
5. Precautions for Using



## 1. Overview

The 32-bit free-run timer supports the 32-bit up count mode or up/down count mode. This timer can be used with the 32-bit input capture. This timer can measure the input pulse width and external clock cycle.

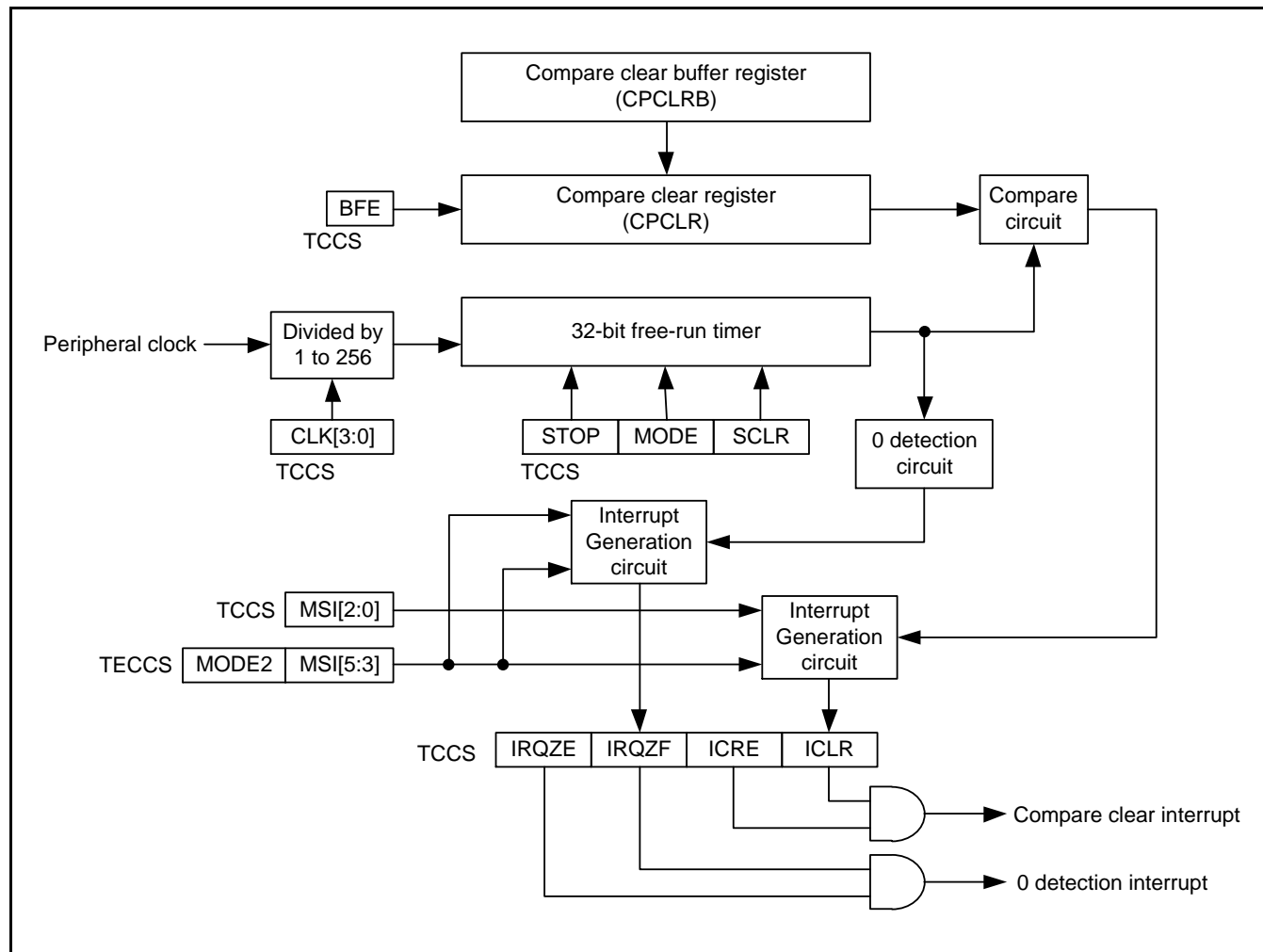
### (1) Functions of the 32-bit Free-run Timer

- The 32-bit free-run timer is composed of the 32-bit up/down counter, control register, 32-bit compare clear register, 32-bit compare clear buffer register, and prescaler.
- The 9 types of counter operation clocks ( $\phi$ ,  $\phi/2$ ,  $\phi/4$ ,  $\phi/8$ ,  $\phi/16$ ,  $\phi/32$ ,  $\phi/64$ ,  $\phi/128$ , and  $\phi/256$ ) ( $\phi$ : peripheral clock) can be selected.
- A compare clear interrupt is generated when the values of the compare clear register and the 32-bit counter are compared and their values match. A 0 detection interrupt is generated when the 32-bit counter detects the count value "0x00000000".
- The compare clear register has a buffer register (the data written to this buffer register is transferred to the compare clear register). When the 32-bit counter is stopped, the data is transferred as soon as data is written to the buffer. When the 32-bit counter is operating, data is transferred from the buffer upon detecting the count value "0x00000000".
- The count value is reset to "0x00000000" when a hardware reset occurs, software clears the timer, or the value of the compare clear register and the count value match in up count mode.
- The output value of the 32-bit counter can be used as a clock count of the input capture.

## 2. Configuration

Figure 2-1 is a configuration diagram of the 32-bit free-run timer.

**Figure 2-1 Block Diagram of 32-bit Free-run Timer**



### 3. Operation

This section provides a summary of the operation of the 32-bit free-run timer.

#### (1) Operation of the 32-bit Free-run Timer

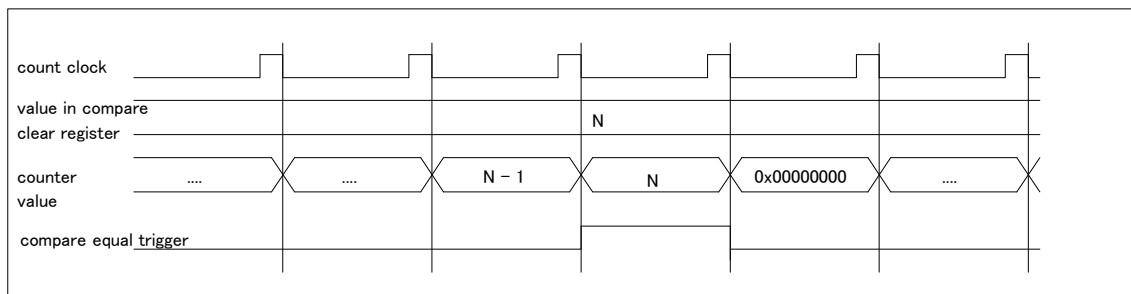
The 32-bit free-run timer starts counting from the value set in the timer data register (TCDT) after the timer is set to enabled (TCCS:STOP). When the 32-bit input capture is connected to the free-run timer, then the timer count value is used as base time of the 32-bit input capture.

#### (2) Counter Clear

The count value of the free-run timer is cleared to 0 when any of the following conditions is met.

- When the count value matches the value of the compare clear register (CPCLR) in the up count mode (MODE=0 of the timer state control register (TCCS))
- When "1" is written to the timer clear bit (SCLR) of the timer state control register (TCCS) while the free-run timer is operating (STOP=0 of the timer state control register (TCCS))
- When "0x00000000" is written to the timer data register (TCDT) while the free-run timer is stopped (STOP=1 of the timer state control register (TCCS))
- When the hardware is reset. Upon reset, the counter is cleared immediately.
- When "1" is written to the timer clear bit (SCLR) of the timer state control register (TCCS) or when the count value matches the value of the compare clear register, the counter is cleared synchronously with the count timing.

Figure 3-1 Clear Timing of 32-bit Free-run Timer



**Note:**

- The count value of the free-run timer is not cleared if "1" is written to the timer clear bit (SCLR) of the timer state control register (TCCS) while the timer is stopped.

### (3) Timer Mode

Either of the following modes can be selected for the free-run timer.

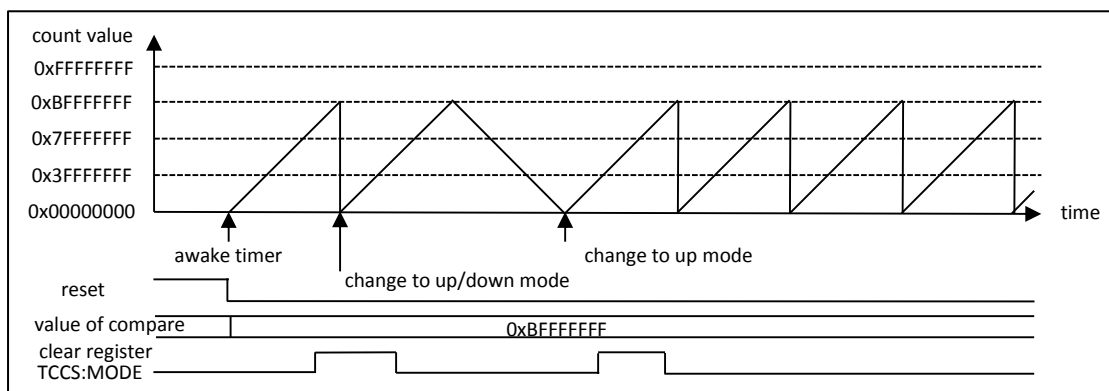
- Up count mode (MODE=0 of the timer state control register (TCCS))
- Up/down count mode (MODE=1 of the timer state control register (TCCS))

In the up count mode, the counter starts counting from the timer data register (TCDT) that is set in advance. The counter continues to count up until the count value matches the value of the compare clear register (CPCLR). Then, the counter is cleared to "0x00000000" and starts counting up again.

In the up/down count mode, the counter starts counting from the timer data register (TCDT) that is set in advance. The counter continues to count up until the count value matches the value of the compare clear register (CPCLR). Then, the counter changes the counting direction from up count to down count. It continues to count down until the counter value reaches "0x00000000" and starts counting up again.

A value can be written to the timer count mode bit (MODE) of the timer state control register (TCCS) at any time, even when the timer is operating or stopped. The value written to this bit during the timer operation is stored in a buffer. The count mode changes when the count value becomes "0x00000000".

**Figure 3-2 Change of Timer Mode (While Timer is Operating)**

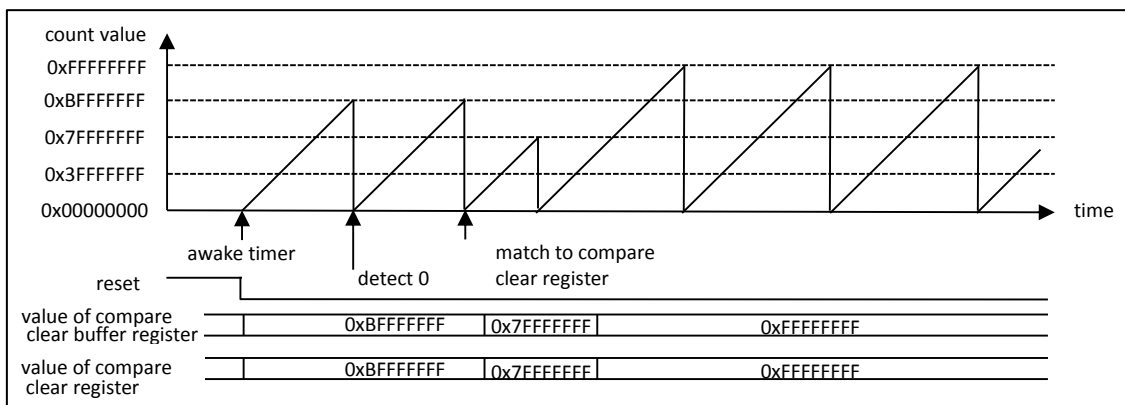




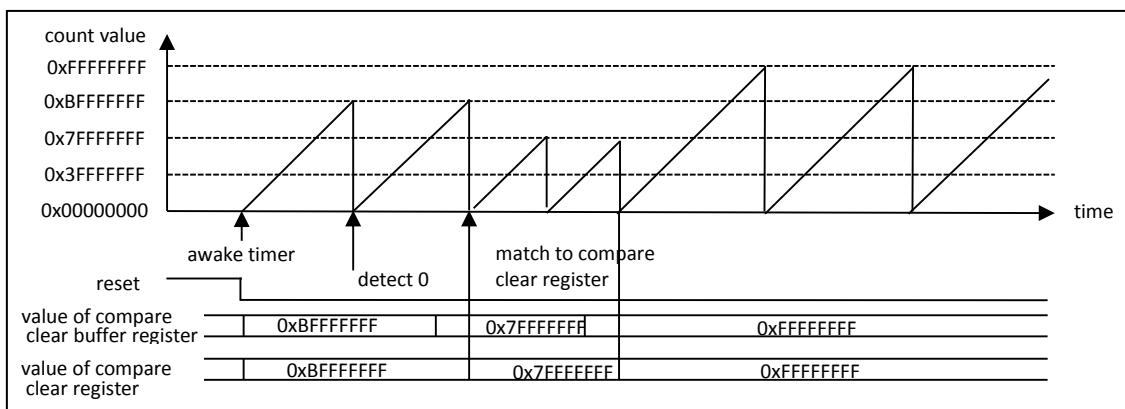
#### (4) Compare Clear Buffer

The compare clear register (CPCLR) has a buffer function that can be enabled or disabled. When the buffer function is enabled (BFE=1 of the timer state control register (TCCS)), the data written to the compare clear buffer register (CPCLRB) is transferred to the CPCLR register upon detecting the count value 0. When the buffer function is disabled (BFE=0 of the timer state control register (TCCS)), the data can be directly written to the compare clear register (CPCLR).

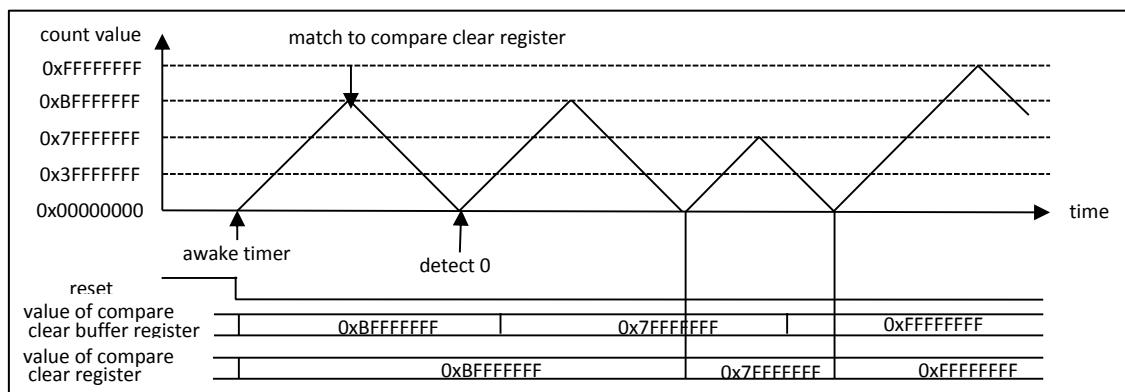
**Figure 3-3 Up Count Mode Operation When Compare Clear Buffer is Disabled (BFE=0 of Timer State Control Register (TCCS))**



**Figure 3-4 Up Count Mode Operation When Compare Clear Buffer is Enabled (BFE=1 of Timer State Control Register (TCCS))**



**Figure 3-5 Up/Down Count Mode Operation When Compare Clear Buffer is Enabled (BFE=1 of Timer State Control Register (TCCS))**





(5) Timer Interrupt

The 32-bit free-run timer can generate the following 2 interrupts.

- Compare clear interrupt
- 0 detection interrupt

The compare clear interrupt is generated when the count value matches the value of the compare clear register. The 0 detection interrupt is generated when the count value reaches "0x00000000".

**Note:**

- The 0 detection interrupt is not generated when the timer is cleared (SCLR=1 of the timer state control register (TCCS)).

Figure 3-6 Interrupt Generated in Up Count Mode (MODE=0 of the Timer State Control Register (TCCS))

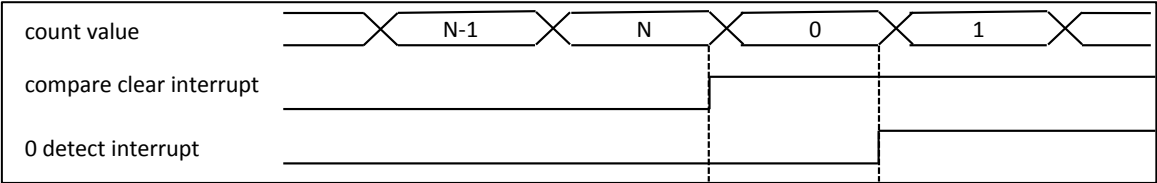
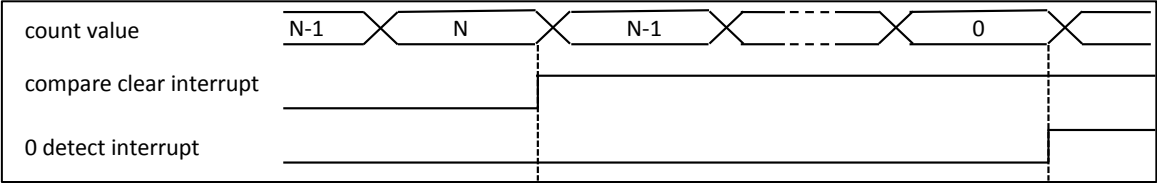


Figure 3-7 Interrupt Generated in Up/Down Count Mode (MODE=1 of the Timer State Control Register (TCCS))



## (6) Interrupt Mask Function

Either the 0 detection interrupt or the compare clear interrupt, or both interrupts can be masked.

The following describes the case when either interrupt is masked.

- An interrupt request flag can be masked by setting the interrupt mask selection bits (MSI2 to MSI0) of the timer state control register (TCCS). The interrupt mask selection bits (MSI2 to MSI0) are the 3-bit reload down register that reloads the value when the mask count value reaches "0b000". The mask count value can also be loaded by directly writing data to the interrupt mask selection bits (MSI2 to MSI0). The number of mask counts is the value set in the interrupt mask selection bits (MSI2 to MSI0). An interrupt request flag is not masked when the mask count value (MSI2 to MSI0) becomes "0b000".
- The mask control of an interrupt request varies depending on the count mode (MODE of the timer state control register (TCCS)). In the up count mode (MODE=0), only the compare clear interrupt request flag can be masked and the 0 detection interrupt is generated every time a timer counter value of 0 is detected. In the up/down count mode (MODE=1), only the 0 detection interrupt request flag can be masked.

The following describes the case when both interrupt requests are masked.

- Both interrupts can be masked only when the free-run timer is in the up/down count mode (MODE=1) and when the timer extended state control register (TECCS) is set to MODE2=1 and the timer state control register (TCCS) is set to MODE=1.
- MSI2 to MSI0 bits of the timer state control register (TCCS) are used to mask the 0 detection interrupt. MSI5 to MSI3 bits of the timer extended state control register (TECCS) are used to mask the compare clear interrupt.

### Note:

- The 0 detection interrupt is not generated when the timer is cleared (SCLR=1 of the timer state control register (TCCS)).

Figure 3-8 Compare Clear Interrupts to Be Masked in Up Count Mode

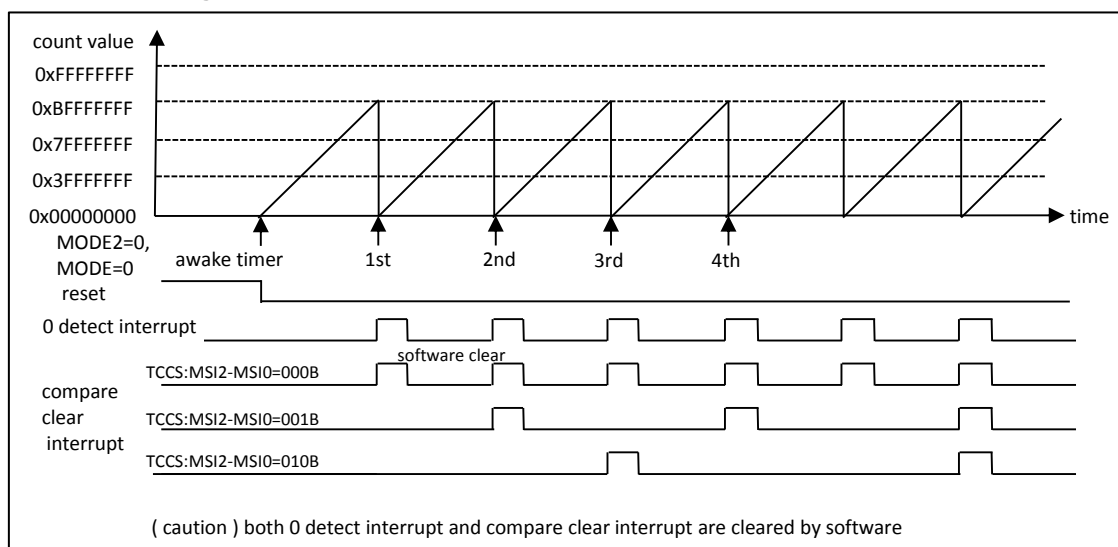


Figure 3-9 0 Detection Interrupts to Be Masked in Up/Down Count Mode

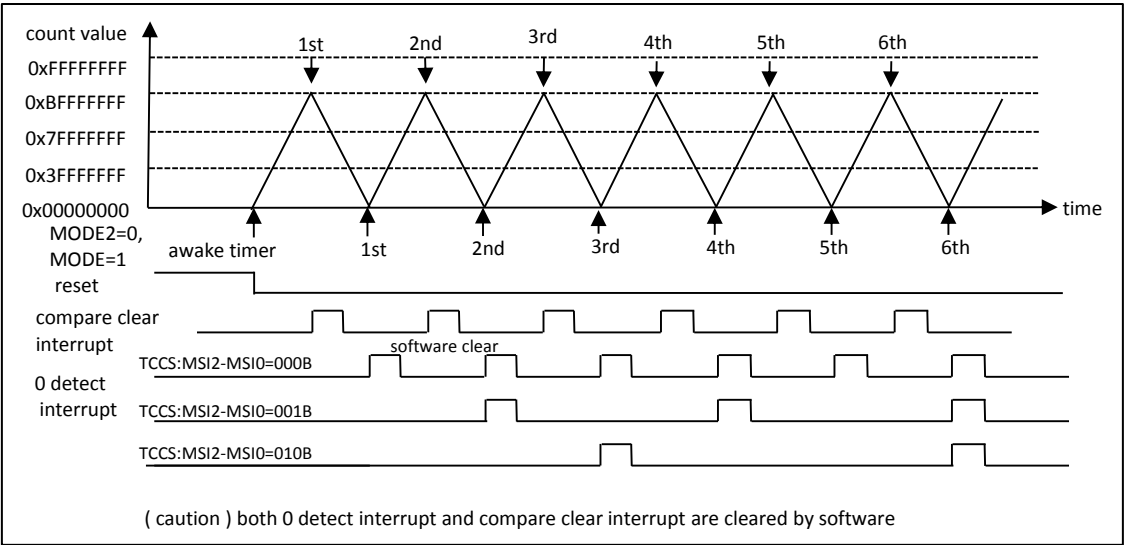
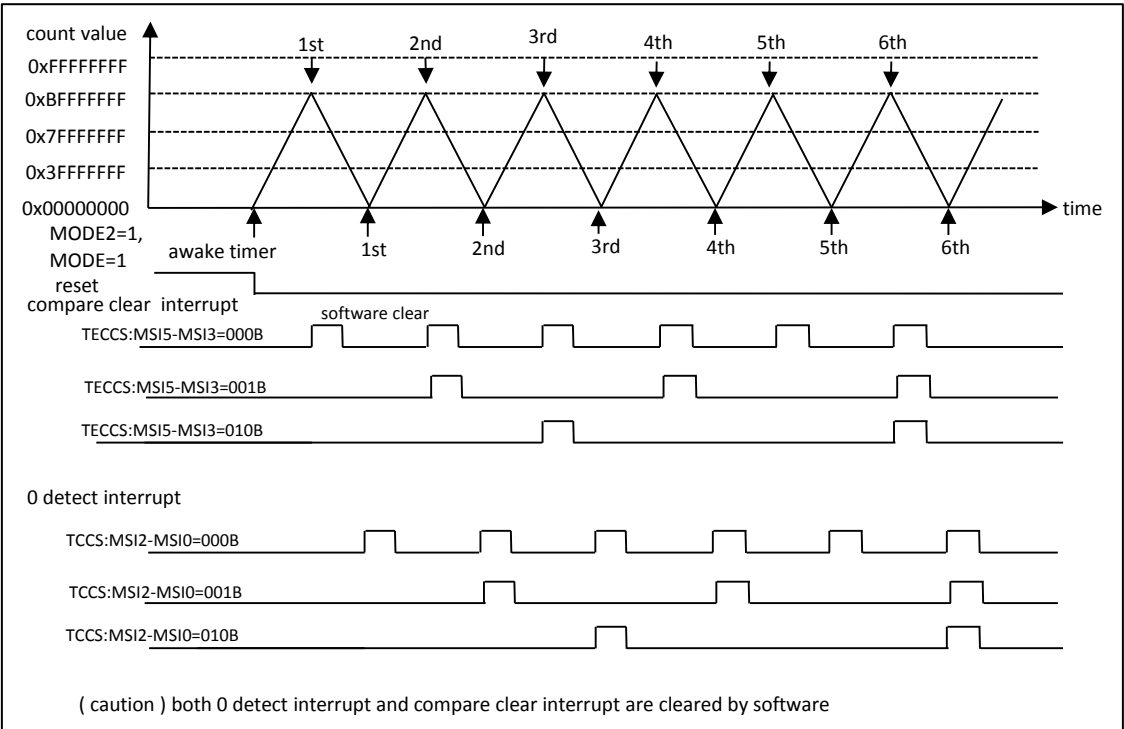


Figure 3-10 0 Detection Interrupts and Compare Clear Interrupts to Be Masked in Up/Down Count Mode





### 3.1. Interrupts of the 32-bit Free-run Timer

There are the 2 types of interrupts as the interrupts of the 32-bit free-run timer: compare interrupt and 0 detection interrupt.

Table 3-1 shows the interrupt control bits and the interrupt factor of the free-run timer.

**Table 3-1 Interrupt Control Bits and Interrupt Factor of Free-run Timer**

Control Bit and Factor	Free-run Timer	
	Compare Clear	0 Detection
Interrupt request flag bit	Compare clear interrupt flag bit (ICLR) of the timer state control register (TCCS)	0 detection interrupt flag bit (IRQZF) of the timer state control register (TCCS)
Interrupt request enable bit	Compare clear interrupt request enable bit (ICRE) of the timer state control register (TCCS)	0 detection interrupt request enable bit (IRQZE) of the timer state control register (TCCS)
Interrupt Factor	The value of the free-run timer matches the value of the compare clear register (CPCLR).	The free-run timer value becomes "0".

"1" is set to the compare clear interrupt flag (ICLR) of the timer state control register (TCCS) when the value of the free-run timer matches the value of the compare clear register (CPCLR). When the interrupt request is set to enabled (ICRE=1 of the timer state control register (TCCS)) in this state, then the interrupt request signal (MIRQ) becomes "H".

"1" is set to the 0 detection interrupt flag (IRQZF) of the timer state control register (TCCS) when the value of the free-run timer becomes "0x00000000". When the interrupt request is set to enabled (IRQZE=1 of the timer state control register (TCCS)) in this state, then the interrupt request signal (ZIRQ) becomes "H".

## 4. Registers

This section provides the register list of the 32-bit free-run timer. All registers are prefixed by "FRTxx\_". xx is the channel number (00, 01, 02, 03, 04).

**Table 4-1 List of 32-bit Free-run Timer Registers**

Abbreviated Register Name	Register Name	See
CPCLRB/CPCLR	Compare Clear Buffer Register / Compare Clear Register	4.1
TCDT	Timer Data Register	4.2
TCCS	Timer State Control Register	4.3
TECCS	Timer Extended State Control Register	4.4
TCCSC	Timer State Control Clear Register	4.5
TCCSS	Timer State Control Set Register	4.6



## 4.1. Compare Clear Buffer Register (CPCLRB)/ Compare Clear Register (CPCLR)

The compare clear buffer register (CPCLRB) is the buffer register of the compare clear register (CPCLR). Both registers are located in the same address.

### (1) Compare Clear Buffer Register (CPCLRB)

BIT_OFFSET	31-0
BIT_NAME	CL
ACCESS_TYPE	W
PROT_TYPE	-
INITIAL_VALUE	11111111_11111111_11111111_11111111

#### [bit31:0] CL[31:0]: Compare clear value buffer bits

The compare clear buffer register (CPCLRB) is a buffer register that is located in the same address as the compare clear register (CPCLR).

The value of the compare clear buffer register (CPCLRB) is immediately transferred to the compare clear register (CPCLR) when the buffer function is set to disabled (BFE=0 of the timer state control register (TCCS)) or the free-run timer is stopped.

The value of the compare clear buffer register (CPCLRB) is transferred to the compare clear register (CPCLR) when the buffer function is set to enabled (BFE=1 of the timer state control register (TCCS)) and 0 is detected as the count value of the free-run timer.

#### Note:

- Do not set "0x00000000" in the compare clear buffer register (CPCLRB).
- For access to this register, use word access instructions.

## (2) Compare Clear Register (CPCLR)

BIT_OFFSET	31-0
BIT_NAME	CL
ACCESS_TYPE	R
PROT_TYPE	-
INITIAL_VALUE	11111111_11111111_11111111_11111111

### [bit31:0] CL[31:0]: Compare clear value bits

The compare clear register (CPCLR) is used to compare the count value of the free-run timer.

In the up count mode (MODE=0 of the timer state control register (TCCS)), the count value of the free-run timer is reset to "0x00000000" when the value of this register matches the count value of the free-run timer.

In the up/down count mode (MODE=1 of the timer state control register (TCCS)), the free-run timer changes the counting direction from up count to down count when the value of the compare clear register (CPCLR) matches the count value of the free-run timer. The counting direction is changed from down count to up count when 0 is detected.

#### **Note:**

- For access to this register, use word access instructions.





## 4.2. Timer Data Register (TCDT)

The timer data register (TCDT) reads the count value of the free-run timer. This register can also be used to set the count value of the free-run timer.

BIT_OFFSET	31-0
BIT_NAME	T
ACCESS_TYPE	R,W
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000_00000000

### [bit31:0] T[31:0]: Timer data value bits

The timer data register (TCDT) is used to read the count value of the free-run timer.

The count value can be set by writing a value to this register. However, a value needs to be written while the free-run timer is stopped (STOP=1 of the timer state control register (TCCS)).

The count value of free-run timer is cleared to "0x00000000" under any of the following conditions.

- Hardware reset (after reset, the counter was cleared immediately)
- The timer clear bit (SCLR) of the timer state control register (TCCS) is "1" while the free-run timer is operating (STOP=0 of the timer state control register (TCCS)).
- The value of the compare clear register (CPCLR) matches the timer count value in the up count mode (MODE=0 of the timer state control register (TCCS)).
- The timer data register (TCDT) is written "0x00000000" while the free-run timer is stopping (STOP=1 of the timer state control register (TCCS)).

When timer clear bit (SCLR) of the timer state control register (TCCS) is written "1" or the value of the compare clear register matches the timer count value, the counter was cleared synchronizing with count timing.

### Notes:

- *The free-run timer is not cleared to "0x00000000" even if the timer clear bit (SCLR) of the timer state control register (TCCS) is set to "1" while the free-run timer is stopped (STOP=1 of the timer state control register (TCCS)).*
- *For access to this register, use word access instructions.*

### 4.3. Timer State Control Register (TCCS)

The timer state control register (TCCS) is a register that is used to control the operation of the free-run timer. For details on writing to this register, see "5.Precautions for Using."

BIT_OFFSET	31-16
BIT_NAME	Reserved
ACCESS_TYPE	R1,WX
PROT_TYPE	-
INITIAL_VALUE	11111111_11111111

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved	IRQZF	IRQZE	MSI2	MSI1	MSI0	ICLR	ICRE
ACCESS_TYPE	R/W0	R,W	R/W	R,W	R,W	R,W	R,W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	BFE	STOP	MODE	SCLR	CLK			
ACCESS_TYPE	R/W	R/W	R/W	R0,W	R/W			
PROT_TYPE	-							
INITIAL_VALUE	0	1	0	0	0000			

**[bit31:15] Reserved: Reserved bits**

#### **[bit14] IRQZF: 0 detection interrupt flag bit**

This bit is set to "1" when the count value of the free-run timer is 0.

When "0" is written to this bit, this bit is cleared to "0".

When "1" is written to this bit, it is not affected.

This bit is cleared by writing "1" to the IRQZFC bit of the TCCSC register.

Value	Description	
	Read	Write
0	0 is not detected.	Clear this bit.
1	0 is detected.	Do not affect this bit.

#### **Notes:**

- This bit is not set to "1" when the timer is cleared ("1" is written to the SCLR) while the free-run timer is operating (STOP=0 of the timer enable bit).
- In the up/down count mode (MODE=1), this bit is set to "1" when an interrupt that is set by the interrupt mask selection bits (MSI2 to MSI0) occurs. This bit is not set to "1" when no interrupt occurs.
- In the up count mode (MODE=0), this bit is set to "1" every time 0 is detected regardless of the value of the interrupt mask selection bits (MSI2 to MSI0).

### [bit13] IRQZE: 0 detection request enable bit

When this bit is set to "1" and the 0 detection interrupt flag bit (IRQZF) is set to "1", an interrupt request to the CPU is generated.

This bit is cleared to "0" by writing "1" to the IRQZEC bit of the TCCSC register. This bit is set to "1" by writing "1" to the IRQZES bit of the TCCSS register.

Value	Description
0	Disable interrupt requests.
1	Enable interrupt requests.

### [bit12:10] MSI2 to MSI0: Interrupt mask selection bits

For MODE2=0 of the timer extended state control register (TECCS)

- In the up count mode (MODE=0), these bits are used to set the number of times to mask the compare clear interrupt flag. In the up/down count mode bit (MODE=1), these bits are used to set the number of times to mask the 0 detection interrupt flag.
- When these bits are set to "0b000", the interrupt flags are not masked.
- For MODE2=1 of the timer extended state control register (TECCS)
- In the up/down count mode (MODE=1), these bits are used to set the number of times to mask the 0 detection interrupt flag.
- In the up count mode (MODE=0), it is prohibited to use these bits to make this setting.

Value			Description
MSI2	MSI1	MSI0	
0	0	0	Generate an interrupt flag at the first match occurrence.
0	0	1	Generate an interrupt flag at the second match occurrence.
0	1	0	Generate an interrupt flag at the third match occurrence.
0	1	1	Generate an interrupt flag at the fourth match occurrence.
1	0	0	Generate an interrupt flag at the fifth match occurrence.
1	0	1	Generate an interrupt flag at the sixth match occurrence.
1	1	0	Generate an interrupt flag at the seventh match occurrence.
1	1	1	Generate an interrupt flag at the eighth match occurrence.

#### Notes:

- The value read is a mask counter value. The mask counter is a decrement counter.
- The written data is written to the mask register.
- While the free-run timer is operating (STOP=0 of the timer enable bit), the value written to the mask register is reloaded to the counter only when the mask counter becomes 0.
- When the free-run timer is stopped (STOP=1 of the timer enable bit), the value written to the mask register is immediately reloaded to the mask counter.

**[bit9] ICLR: Compare clear interrupt flag bit**

This bit is set to "1" when the value of the compare clear register (CPCLR) and the value of the free-run timer match.

When "0" is written to this bit, this bit is cleared to "0".

When "1" is written to this bit, it is not affected.

This bit is cleared by writing "1" to the ICLRC bit of the TCCSC register.

Value	Description	
	Read	Write
0	No compare clear match	Clear this bit.
1	Compare clear match	Do not affect this bit.

**Notes:**

- In the up count mode ( $MODE=0$ ), this bit is set to "1" when the interrupt flag set by the interrupt mask selection bits ( $MSI2$  to  $MSI0$ ) occurs. This bit is not set to "1" when no interrupt occurs.
- In the up/down count mode ( $MODE=1$ ), regardless of the value of the interrupt mask selection bits ( $MSI2$  to  $MSI0$ ), this bit is set to "1" every time the compare clear occurs.

**[bit8] ICRE: Compare clear interrupt request enable bit**

When this bit is set to "1" and the compare clear interrupt flag bit (ICLR) is set to "1", an interrupt request to the CPU is generated.

This bit is cleared to "0" by writing "1" to the ICREC bit of the TCCSC register.

This bit is set to "1" by writing "1" to the ICRES bit of the TCCSS register.

Value	Description
0	Disable interrupt requests.
1	Enable interrupt requests.

**[bit7] BFE: Compare clear buffer enable bit**

This bit is used to enable the compare clear buffer register (CPCLRB).

When this bit is set to "0":

The compare clear buffer register (CPCLRB) becomes invalid. Thus, data can be written directly to the compare clear register (CPCLR).

When this bit is set to "1":

The compare clear buffer register (CPCLRB) becomes valid. The data written and stored in the compare clear buffer register (CPCLRB) is transferred to the compare clear register (CPCLR) when "0" is detected as the count value of the free-run timer.

This bit is cleared to "0" by writing "1" to the BFEC bit of the TCCSC register.

This bit is set to "1" by writing "1" to the BFES bit of the TCCSS register.

Value	Description
0	Disable the compare clear buffer.
1	Enable the compare clear buffer.

#### [bit6] STOP: Timer enable bit

This bit is used to stop/start counting of the free-run timer.

When this bit is set to "0":

The free-run timer starts counting.

When this bit is set to "1":

The free-run timer stops counting.

This bit is cleared to "0" by writing "1" to the STOPC bit of the TCCSC register.

This bit is set to "1" by writing "1" to the STOPS bit of the TCCSS register.

Value	Description
0	Enable counting (start counting).
1	Disable counting (stop counting).

#### [bit5] MODE: Timer count mode bit

This bit is used to select a count mode of the free-run timer.

When this bit is set to "0":

The up count mode is selected. The timer continues to count up until the count value matches the value of the compare clear register (CPCLR) and is reset to 0. After that, it starts counting up again.

When this bit is set to "1":

The up/down count mode is selected. The timer continues to count up until the count value matches the value of the compare clear register (CPCLR). After that, the count direction changes to down count. When the count value reaches 0, the count direction changes to up count again.

A value can be written to this bit regardless of whether the timer is operating (STOP=0 of the timer enable bit) or stopped (STOP=1). When the timer is operating, the value written to this bit is stored in a buffer.

After that, when the timer value becomes 0, the count mode is set based on the value stored in the buffer.

This bit is cleared to "0" by writing "1" to the MODEC bit of the TCCSC register.

This bit is set to "1" by writing "1" to the MODES bit of the TCCSS register.

Value	Description
0	Up count mode
1	Up/down count mode

#### [bit4] SCLR: Timer clear bit

This bit is used to initialize the free-run timer.

Initializing the value of the free-run timer:

The free-run timer is initialized to 0 at the next count clock when "1" is written to this bit while the free-run timer is operating (STOP=0 of the timer enable bit).

The free-run timer is not initialized if "1" is written to this bit when the free-run timer is stopped (STOP=1 of the timer enable bit).

Initializing the count direction of the free-run timer:

The free-run timer always starts the operation from up count when its operation is started again (STOP=0 of the timer enable bit) after "1" is written to this bit.

The free-run timer starts the operation from up count even if it stops the operation when counting down (STOP=1 of the timer enable bit), and then starts operating again (STOP=0 of the timer enable bit) after "1" is written to this bit.

This bit is set to "1" by writing "1" to the SCLRS bit of the TCCSS register.

The read value is always "0".

Value	Description	
	Read	Write
0	"0" always read	Do not initialize the counter.
1		Initialize the counter to "0x00000000".

**Notes:**

- The 0 detection interrupt is not generated even if this bit is set to "1".
- The timer is not cleared when "0" is written to this bit before the next count clock after "1" is written to this bit.
- In the up/down count mode, when "0" is written to the SCLR bit before updating the timer count after "1" is written to the SCLR bit while the timer is counting down, the count value is not updated and the count direction is changed to up count.

**[bit3:0] CLK[3:0]: Clock frequency selection bits**

These bits are used to select the count clock frequency of the free-run timer.

The clock frequency is changed immediately upon setting these bits.

Value	Description					
	Count Clock	$\phi = 40\text{MHz}$	$\phi = 20\text{MHz}$	$\phi = 10\text{MHz}$	$\phi = 5\text{MHz}$	$\phi = 2.5\text{MHz}$
0000	$\phi$	25ns	50ns	100ns	200ns	400ns
0001	$\phi/2$	50ns	100ns	200ns	400ns	800ns
0010	$\phi/4$	100ns	200ns	400ns	800ns	1.6 $\mu\text{s}$
0011	$\phi/8$	200ns	400ns	800ns	1.6 $\mu\text{s}$	3.2 $\mu\text{s}$
0100	$\phi/16$	400ns	800ns	1.6 $\mu\text{s}$	3.2 $\mu\text{s}$	6.4 $\mu\text{s}$
0101	$\phi/32$	800ns	1.6 $\mu\text{s}$	3.2 $\mu\text{s}$	6.4 $\mu\text{s}$	12.8 $\mu\text{s}$
0110	$\phi/64$	1.6 $\mu\text{s}$	3.2 $\mu\text{s}$	6.4 $\mu\text{s}$	12.8 $\mu\text{s}$	25.6 $\mu\text{s}$
0111	$\phi/128$	3.2 $\mu\text{s}$	6.4 $\mu\text{s}$	12.8 $\mu\text{s}$	25.6 $\mu\text{s}$	51.2 $\mu\text{s}$
1000	$\phi/256$	6.4 $\mu\text{s}$	12.8 $\mu\text{s}$	25.6 $\mu\text{s}$	51.2 $\mu\text{s}$	102.4 $\mu\text{s}$
Other settings are prohibited.	-	-	-	-	-	-

$\phi$ : Peripheral clock.



## 4.4. Timer Extended State Control Register (TECCS)

The timer extended state control register (TECCS) is an extended control register that controls the operation of the free-run timer.

BIT_OFFSET	31-16
BIT_NAME	Reserved
ACCESS_TYPE	R1,WX
PROT_TYPE	-
INITIAL_VALUE	11111111_11111111

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved				MODE2	MSI5	MSI4	MSI3
ACCESS_TYPE	R0,WX				R/W	R,W	R,W	R,W
PROT_TYPE	-							
INITIAL_VALUE	0000				0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved							
ACCESS_TYPE	R1,WX							
PROT_TYPE	-							
INITIAL_VALUE	11111111							

[bit31:12] Reserved: Reserved bits

### [bit11] MODE2: Interrupt mask mode bit 2

This bit is used to independently mask the 0 detection interrupt and the compare clear interrupt when the free-run timer is in the up/down count mode (MODE=1 of the timer state control register (TCCS)).

If "1" is written to this bit when the free-run timer is in the up/down count mode (MODE=1 of the timer state control register (TCCS)), the value set to the compare clear interrupt mask selection bits (MSI5 to MSI3) of this register is enabled and the compare clear interrupt flag is masked the specified number of times. The value set to the interrupt mask selection bits (MSI2 to MSI0) of the timer state control register (TCCS) becomes valid as the number of times to mask the 0 detection interrupt flag.

Value		Description
MODE2	MODE*	
0	0	The setting values of MSI5 to MSI3 are invalid.
0	1	The setting values of MSI5 to MSI3 are invalid.
1	0	The setting is prohibited (the operation is not guaranteed).
1	1	The setting values of MSI5 to MSI3 are valid.

\*: bit5 of the timer state control register (TCCS)

#### Note:

- The operation when "1" is written to this bit is not guaranteed when the free-run timer is in the up count mode.

**[bit10:8] MSI5 to MSI3: Compare clear interrupt mask selection bits**

- These bits are valid only when the interrupt mask mode bit 2 is "1" (MODE2=1) and the free-run timer is in the up/down count mode (MODE=1 of the timer state control register (TCCS)). These bits are used to set the number of times to mask the compare clear interrupt flag.  
The number of times to mask the 0 detection interrupt flag is set by MSI2 to MSI0 of the timer state control register (TCCS).
- The compare clear interrupt flag is not masked when these bits are set to "0b000".

Value			Description
MSI5	MSI4	MSI3	
0	0	0	Generate an interrupt flag at the first match occurrence.
0	0	1	Generate an interrupt flag at the second match occurrence.
0	1	0	Generate an interrupt flag at the third match occurrence.
0	1	1	Generate an interrupt flag at the fourth match occurrence.
1	0	0	Generate an interrupt flag at the fifth match occurrence.
1	0	1	Generate an interrupt flag at the sixth match occurrence.
1	1	0	Generate an interrupt flag at the seventh match occurrence.
1	1	1	Generate an interrupt flag at the eighth match occurrence.

**Notes:**

- The value read is a mask counter value. The mask counter is a decrement counter.
- The written data is written to the mask register.
- While the free-run timer is operating (the timer enable bit STOP=0 of the timer state control register (TCCS)), the value written to the mask register is reloaded to the counter only when the mask counter becomes 0.
- When the free-run timer is stopped (the timer enable bit STOP=1 of the timer state control register (TCCS)), the value written to the mask register is immediately reloaded to the mask counter.

**[bit7:0] Reserved: Reserved bits**





## 4.5. Timer State Control Clear Register (TCCSC)

The timer state control clear register (TCCSC) is a register that is used to clear bits of the timer state control register (TCCS).

BIT_OFFSET	31-16
BIT_NAME	Reserved
ACCESS_TYPE	R0,W0
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved	IRQZFC	IRQZEC	Reserved			ICLRC	ICREC
ACCESS_TYPE	R0,WX	R0,W	R0,W	R0,W0			R0,W	R0,W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	000			0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	BFEC	STOPC	MODEC	Reserved				
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W0				
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	00000				

**[bit31:15] Reserved: Reserved bits**

### [bit14] IRQZFC: IRQZF clear bit

When "1" is written to this bit, the 0 detection interrupt flag bit (IRQZF) of the timer state control register (TCCS) is cleared.

Value	Description
0	Affect neither this bit nor the 0 detection interrupt flag bit (IRQZF) of the timer state control register (TCCS).
1	Clear the 0 detection interrupt flag bit (IRQZF) of the timer state control register (TCCS).

### [bit13] IRQZEC: IRQZE clear bit

When "1" is written to this bit, the 0 detection request enable bit (IRQZE) of the timer state control register (TCCS) is cleared.

Value	Description
0	Affect neither this bit nor the 0 detection request enable bit (IRQZE) of the timer state control register (TCCS).
1	Clear the 0 detection request enable bit (IRQZE) of the timer state control register (TCCS).

**[bit12:10] Reserved: Reserved bits**

**[bit9] ICLRC: ICLR clear bit**

When "1" is written to this bit, the compare clear interrupt flag bit (ICLR) of the timer state control register (TCCS) is cleared.

Value	Description
0	Affect neither this bit nor the compare clear interrupt flag bit (ICLR) of the timer state control register (TCCS).
1	Clear the compare clear interrupt flag bit (ICLR) of the timer state control register (TCCS).

**[bit8] ICREC: ICRE clear bit**

When "1" is written to this bit, the compare clear interrupt request enable bit (ICRE) of the timer state control register (TCCS) is cleared.

Value	Description
0	Affect neither this bit nor the compare clear interrupt request enable bit (ICRE) of the timer state control register (TCCS).
1	Clear the compare clear interrupt request enable bit (ICRE) of the timer state control register (TCCS).

**[bit7] BFEC: BFE clear bit**

When "1" is written to this bit, the compare clear buffer enable bit (BFE) of the timer state control register (TCCS) is cleared.

Value	Description
0	Affect neither this bit nor the compare clear buffer enable bit (BFE) of the timer state control register (TCCS).
1	Clear the compare clear buffer enable bit (BFE) of the timer state control register (TCCS).

**[bit6] STOPC: STOP clear bit**

When "1" is written to this bit, the timer enable bit (STOP) of the timer state control register (TCCS) is cleared.

Value	Description
0	Affect neither this bit nor the timer enable bit (STOP) of the timer state control register (TCCS).
1	Clear the timer enable bit (STOP) of the timer state control register (TCCS).

**[bit5] MODEC: MODE clear bit**

When "1" is written to this bit, the timer count mode bit (MODE) of the timer state control register (TCCS) is cleared.

Value	Description
0	Affect neither this bit nor the timer count mode bit (MODE) of the timer state control register (TCCS).
1	Clear the timer count mode bit (MODE) of the timer state control register (TCCS).

**[bit4:0] Reserved: Reserved bits**



## 4.6. Timer State Control Set Register (TCCSS)

The timer state control set register (TCCSC) is a register that is used to set bits of the timer state control register (TCCS).

BIT_OFFSET	31-16
BIT_NAME	Reserved
ACCESS_TYPE	R0,W0
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved		IRQZES	Reserved				ICRES
ACCESS_TYPE	R0,W0		R0,W	R0,W0				R0,W
PROT_TYPE	-							
INITIAL_VALUE	00		0	0000				0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	BFES	STOPS	MODES	SCLRS	Reserved			
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W	R0,W0			
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0000			

**[bit31:14] Reserved: Reserved bits**

**[bit13] IRQZES: IRQZE set bit**

When "1" is written to this bit, the 0 detection request enable bit (IRQZE) of the timer state control register (TCCS) is set to "1".

Value	Description
0	Affect neither this bit nor the 0 detection request enable bit (IRQZE) of the timer state control register (TCCS).
1	Set the 0 detection request enable bit (IRQZE) of the timer state control register (TCCS).

**[bit12:9] Reserved: Reserved bits**

**[bit8] ICRES: ICRE set bit**

When "1" is written to this bit, the compare clear interrupt request enable bit (ICRE) of the timer state control register (TCCS) is set to "1".

Value	Description
0	Affect neither this bit nor the compare clear interrupt request enable bit (ICRE) of the timer state control register (TCCS).
1	Set the compare clear interrupt request enable bit (ICRE) of the timer state control register (TCCS).

**[bit7] BFES: BFE set bit**

When "1" is written to this bit, the compare clear buffer enable bit (BFE) of the timer state control register (TCCS) is set to "1".

Value	Description
0	Affect neither this bit nor the compare clear buffer enable bit (BFE) of the timer state control register (TCCS).
1	Set the compare clear buffer enable bit (BFE) of the timer state control register (TCCS).

**[bit6] STOPS: STOP set bit**

When "1" is written to this bit, the timer enable bit (STOP) of the timer state control register (TCCS) is set to "1".

Value	Description
0	Affect neither this bit nor the timer enable bit (STOP) of the timer state control register (TCCS).
1	Set the timer enable bit (STOP) of the timer state control register (TCCS).

**[bit5] MODES: MODE set bit**

When "1" is written to this bit, the timer count mode bit (MODE) of the timer state control register (TCCS) is set to "1".

Value	Description
0	Affect neither this bit nor the timer count mode bit (MODE) of the timer state control register (TCCS).
1	Set the timer count mode bit (MODE) of the timer state control register (TCCS).

**[bit4] SCLRS: SCLR set bit**

When "1" is written to this bit, the timer clear bit (SCLR) of the timer state control register (TCCS) is set to "1". "0" is always read from this bit.

Value	Description
0	Affect neither this bit nor the timer clear bit (SCLR) of the timer state control register (TCCS).
1	Set the timer clear bit (SCLR) of the timer state control register (TCCS).

**[bit3:0] Reserved: Reserved bits**



## 5. Precautions for Using

The following shows the notes when using the 32-bit free-run timer.

### (1) Notes to Observe when Accessing a Register

#### a) When Accessing the Compare Clear Register (CPCLR) or the Compare Clear Buffer Register (CPCLRB)

Use word access instructions to the compare clear register (CPCLR) and the compare clear buffer register (CPCLRB).

#### b) When Accessing the Timer State Control Register (TCCS)

- This register supports writing from the bit-band alias area. For the bit-band alias area, see "CHAPTER: Bit-band Unit".
- To clear a specified bit of this register, clear the bit by writing "1" to the applicable bit of the timer state control clear register (TCCSC).
- To set a specified bit of this register, set the bit by writing "1" to the applicable bit of the timer state control set register (TCCSS).
- Data can be written directly to this register only when writing to all bits.
- In the normal reading mode, the interrupt mask counter value is read from MSI2 to MSI0.

#### c) When Accessing the Timer Extended State Control Register (TECCS)

- In the normal reading mode, the interrupt mask counter value is read from MSI5 to MSI3.

### (2) Notes when Operating the Free-run Timer

#### When Setting Using a Program

- When the hardware is reset, the count value becomes "0x00000000", but the 0 detection interrupt flag is not set.
- Because the timer mode bit (MODE of the timer state control register (TCCS)) has a buffer, a specified timer mode is enabled after 0 is detected.
- The timer clear bit (SCLR=1 of the timer state control register (TCCS)) initializes the timer. However, this bit does not generate the 0 detection interrupt.
- The compare clear flag is not set if the timer starts counting when the value of the compare clear register (CPCLR) matches the count value.
- Set any values other than 0 to the compare clear register (CPCLR). Note that the following operations occur if 0 is set.
  - When the timer mode bit (MODE of the timer state control register (TCCS)) is in the up count mode (MODE=0), the count value is updated to 0 and fixed to 0. Then, the 0 detection interrupt flag and the compare clear flag are set at every count clock.
  - When the timer count mode bit (MODE of the timer state control register (TCCS)) is in the up/down count mode (MODE=1), the count value is counted up from "0x00000000" to "0xFFFFFFFF", and this up count operation is repeated. The 0 detection interrupt flag and the compare clear flag are set to "1" when the count value becomes 0.



## CHAPTER 41: 32-bit Input Capture

This chapter describes the functions of the 32-bit input capture.

---

1. Overview
2. Configuration
3. Operation
4. Registers
5. Precautions for Using



## 1. Overview

The 32-bit input capture can measure the input pulse width and external clock cycle based on the value of the 32-bit free-run timer.

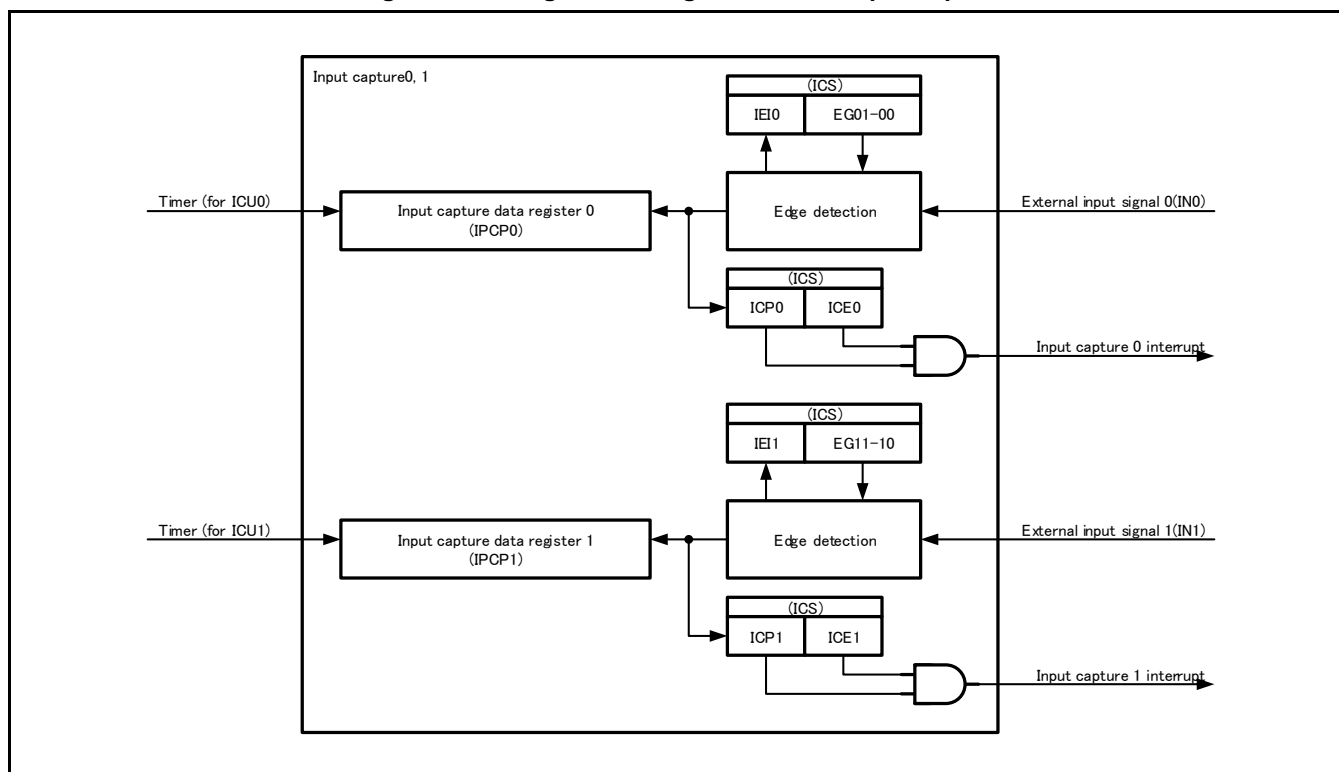
### Functions of the 32-bit Input Capture

- The 32-bit input capture is composed of 2 input captures. Each channel can be operated individually.
- Each input capture is composed of 2 independent external input pins (IN0 and IN1), capture registers that correspond to the pins, and the capture control register. When an edge signal is detected on an external input pin, the value of the free-run timer can be stored in the capture register. At the same time, an interrupt is generated.
- 3 types of trigger edges (rising edge, falling edge, and both edges) of an external input signal can be selected. There is a register that indicates whether a trigger edge is a rising or falling edge.
- An interrupt is generated when a valid edge is detected from an external input signal.

## 2. Configuration

Figure 2-1 is a configuration diagram of the 32-bit input capture.

### Figure 2-1 Configuration Diagram of 32-bit Input Capture





### 3. Operation

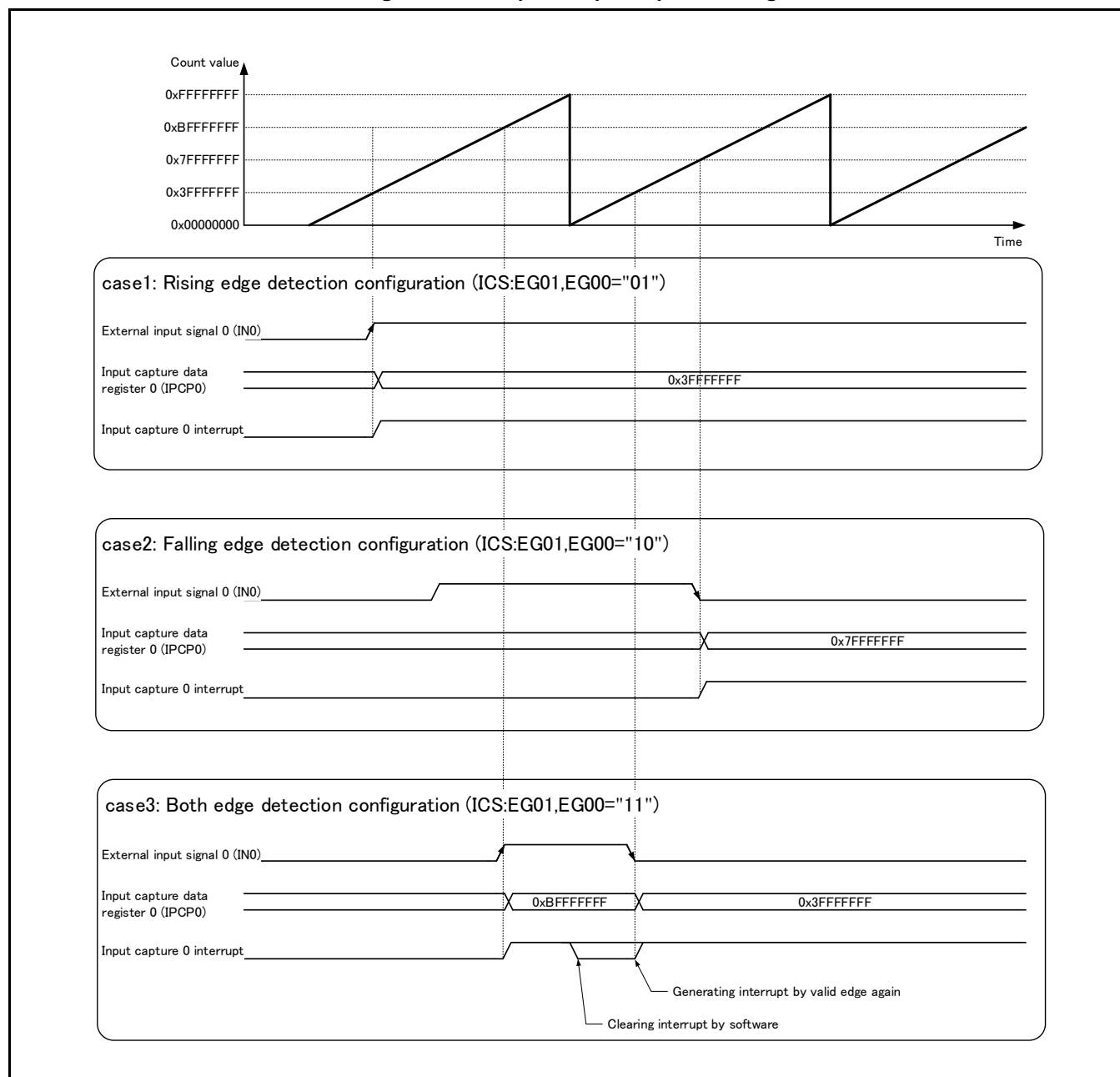
This section provides an overview of 32-bit input capture operation.

#### (1) Operation of the 32-bit Input Capture

The 32-bit input capture is used to detect a specified valid edge. When a valid edge is detected, an interrupt flag is set. Then the value of the 32-bit free-run timer is loaded to the input capture register.

#### (2) Input Capture Operation

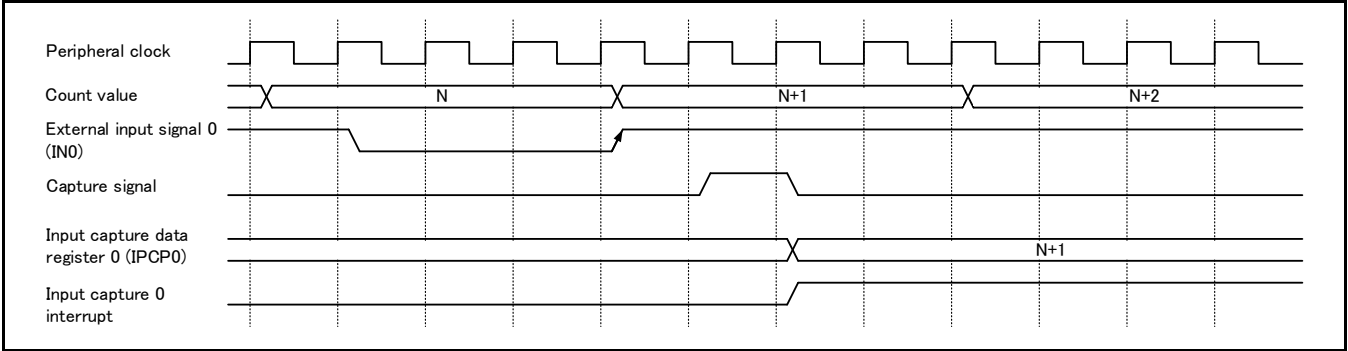
Figure 3-1 Example of Input Capture Timing





(3) Input Capture Input Timing

Figure 3-2 Example of Input Capture Timing for Input Signals





### 3.1. Interrupt

As the interrupt of the 32-bit input capture, there is an input capture interrupt triggered by an external input signal.

#### Input Capture Interrupt

Table 3-1 shows the interrupt control bits and interrupt factor of the input capture.

**Table 3-1 Interrupt Control Bits and Interrupt Factor of Input Capture 1, 0**

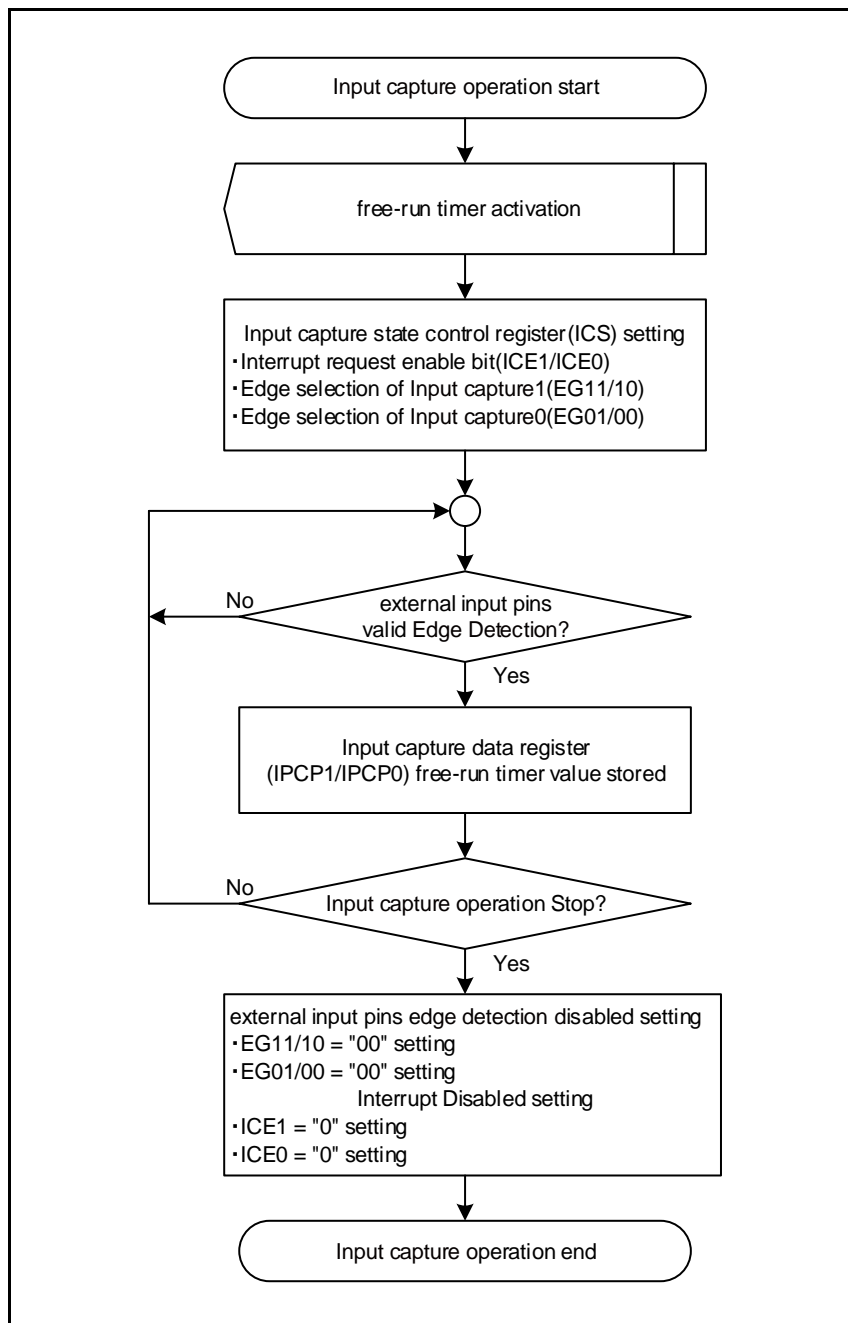
Control Bit and Factor	Input Capture 1/0
Interrupt Request Flag Bit	The interrupt request flag bits ICP1, 0 in the input capture state control register (ICS)
Interrupt Request Enable Bit	The interrupt request enable bits ICE1, 0 in the input capture state control register (ICS)
Interrupt Factor	A valid edge is detected on the external input pin (IN1, IN0).

For the interrupt capture, when a valid edge is detected on the external input pin (IN1, IN0), the interrupt request flag (ICP1, ICP0) in the input capture state control register (ICS) is set to "1". When an interrupt request is set to enabled (ICE1, ICE0 = "0b11" in the input capture state control register (ICS)) in this state, then the interrupt request is output to the interrupt controller.

## 3.2. Setting Procedure Example

This section explains a setting procedure example of 32-bit input capture.

Figure 3-3 Setting Procedure Example of 32-bit Input Capture





## 4. Registers

This section provides the register list of the 32-bit input capture.

All registers are prefixed by "ICUxx\_". xx is the channel number (00, 02, 04).

**Table 4-1 Register List of the 32-bit Input Capture**

Abbreviated Register Name	Register Name	See
IPCP0/IPCP1	Input Capture Data Registers 0/1	4.1
ICS	Input Capture State Control Register	4.2
ICSC	Input Capture State Control Clear Register	4.3
ICSS	Input Capture State Control Set Register	4.4

## 4.1. Input Capture Data Registers 0, 1 (IPCP0, IPCP1)

Input capture data registers 0, 1 (IPCP0, IPCP1) are used to store the count value of the free-run timer when a valid edge of an external input signal is detected.

### (1) Input Capture Data Register (IPCP0)

BIT_OFFSET	31-0
BIT_NAME	CP
ACCESS_TYPE	R,WX
PROT_TYPE	-
INITIAL_VALUE	XXXXXXXX_XXXXXXXX_XXXXXXXX_XXXXXXXX

#### [bit31:0] CP31 to CP00: Input capture data value bits

The input capture data register 0 (IPCP0) is used to store the value of the free-run timer when a valid edge of the external input pin IN0 signal is detected.

The free-run timer that is mentioned here indicates the operating state of the free-run timer that is connected to the input capture.

**Note:**

- For access to this register, use word access instructions.



## (2) Input Capture Data Register (IPCP1)

BIT_OFFSET	31-0
BIT_NAME	CP
ACCESS_TYPE	R,WX
PROT_TYPE	-
INITIAL_VALUE	XXXXXXXX_XXXXXXXX_XXXXXXXX_XXXXXXXX

### [bit31:0] CP31 to CP00: Input capture data value bits

Input capture data register 1 (IPCP1) is used to store the value of the free-run timer when a valid edge of the external input pin IN1 signal is detected.

The free-run timer that is mentioned here indicates the operating state of the free-run timer that is connected to the input capture.

#### **Note:**

- For access to this register, use word access instructions.

## 4.2. Input Capture State Control Register (ICS)

The input capture state control register (ICS) is used to select an edge, enable an interrupt request, and control an interrupt request flag. This register is also used to indicate a valid edge detected in the input captures 0, 1.

For details on writing to this register, see "5. Precautions for Using".

BIT_OFFSET	31-16
BIT_NAME	Reserved
ACCESS_TYPE	R1,WX
PROT_TYPE	-
INITIAL_VALUE	11111111_11111111

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved						IEI1	IEI0
ACCESS_TYPE	R1,WX						R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	111111						0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	ICP1	ICP0	ICE1	ICE0	EG11	EG10	EG01	EG00
ACCESS_TYPE	R,W	R,W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

**[bit31:10] Reserved: Reserved bits**

### **[bit9] IEI1: Valid edge indication bit (input capture 1)**

- This bit is a valid edge indication bit in input capture data register 1. This bit indicates that a rising or falling edge is detected.
- When a falling edge is detected, this bit is set to "0".
- When a rising edge is detected, this bit is set to "1".
- This bit is read-only.

Value	Description
0	A falling edge is detected.
1	A rising edge is detected.

**Note:**

- The value read is meaningless if the edge selection bit (EG11, EG10) is set to "0b00".
- If the edge selection bit (EG11, EG10) is set to any value other than "0b00", then the value is updated when the interrupt request flag (ICP1) is set.



**[bit8] IEI0: Valid edge indication bit (input capture 0)**

- This bit is a valid edge indication bit in input capture data register 0. This bit indicates that a rising or falling edge is detected.
- When a falling edge is detected, this bit is set to "0".
- When a rising edge is detected, this bit is set to "1".
- This bit is a read-only bit.

Value	Description
0	A falling edge is detected.
1	A rising edge is detected.

**Notes:**

- The value read is meaningless if the edge selection bit (EG01, EG00) is set to "0b00".
- If the edge selection bit (EG01, EG00) is set to any value other than "0b00", then the value is updated when the interrupt request flag (ICP0) is set.

**[bit7] ICP1: Interrupt request flag bit (input capture 1)**

- This bit is used as an interrupt request flag of input capture 1.
- This bit is set to "1" immediately upon detecting a valid edge on the external input pin (IN1).
- An interrupt is generated as soon as this bit is set to "1" when the interrupt request enable bit (ICE1) is "1".
- When this bit is set to "0", this bit is cleared to "0".
- When this bit is set to "1", this bit is not affected.
- This bit is cleared to "0" by writing "1" to the ICP1C bit in the ICSC register.

Value	Description	
	Read	Write
0	No valid edge is detected.	This bit is cleared.
1	A valid edge is detected.	This bit is not affected.

**[bit6] ICP0: Interrupt request flag bit (input capture 0)**

- This bit is used as an interrupt request flag of input capture 0.
- This bit is set to "1" immediately upon detecting a valid edge on the external input pin (IN0).
- An interrupt is generated as soon as this bit is set to "1" when the interrupt request enable bit (ICE0) is "1".
- When this bit is set to "0", this bit is cleared to "0".
- When this bit is set to "1", this bit is not affected.
- This bit is cleared to "0" by writing "1" to the ICP0C bit in the ICSC register.

Value	Description	
	Read	Write
0	No valid edge is detected.	This bit is cleared.
1	A valid edge is detected.	This bit is not affected.

**[bit5] ICE1: Interrupt request enable bit (input capture 1)**

- This bit is used to enable an interrupt request of input capture1.
- An interrupt of input capture 1 is generated when the interrupt request flag bit (ICP1) is set while this bit is "1".
- This bit is cleared to "0" by writing "1" to the ICE1C bit in the ICSC register.
- This bit is set to "1" by writing "1" to the ICE1S bit in the ICSS register.

Value	Description
0	Disable interrupt requests.
1	Enable interrupt requests.

**[bit4] ICE0: Interrupt request enable bit (input capture 0)**

- This bit is used to enable an interrupt request of input capture 0.
- An interrupt of input capture 0 is generated when the interrupt request flag bit (ICP0) is set while this bit is "1".
- This bit is cleared to "0" by writing "1" to the ICE0C bit in the ICSC register.
- This bit is set to "1" by writing "1" to the ICE0S bit in the ICSS register.

Value	Description
0	Disable interrupt requests.
1	Enable interrupt requests.

**[bit3:2] EG11, EG10: Edge selection bits (input capture 1)**

These bits are used to enable the operation of input capture 1. These bits specify a valid edge of the external input (IN1).

Value		Description
EG11	EG10	
0	0	No edge is detected (stop).
0	1	A rising edge is detected.
1	0	A falling edge is detected.
1	1	Both edges are detected.

**[bit1:0] EG01, EG00: Edge selection bits (input capture 0)**

These bits are used to enable the operation of input capture 0. These bits specify a valid edge of the external input (IN0).

Value		Description
EG01	EG00	
0	0	No edge is detected (stop).
0	1	A rising edge is detected.
1	0	A falling edge is detected.
1	1	Both edges are detected.



### 4.3. Input Capture State Control Clear Register (ICSC)

The input capture state control clear register (ICSC) is a register to clear a bit in the input capture state control register (ICS).

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,W0
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	ICP1C	ICP0C	ICE1C	ICE0C	Reserved			
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W	R0,W0			
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0000			

**[bit31:8] Reserved: Reserved bits**

#### **[bit7] ICP1C: ICP1 clear bit**

Writing "1" to this bit clears the ICP1 bit in the ICS register.

Value	Description
0	Do not affect this bit and the interrupt request flag bit (input capture1) (ICP1) in the input capture state control register (ICS).
1	Clear the interrupt request flag bit (input capture1) (ICP1) in the input capture state control register (ICS).

#### **[bit6] ICP0C: ICP0 clear bit**

Writing "1" to this bit clears the ICP0 bit in the ICS register.

Value	Description
0	Do not affect this bit and the interrupt request flag bit (input capture0) (ICP0) in the input capture state control register (ICS).
1	Clear the interrupt request flag bit (input capture0) (ICP0) in the input capture state control register (ICS).

#### **[bit5] ICE1C: ICE1 clear bit**

Writing "1" to this bit clears the ICE1 bit in the ICS register.

Value	Description
0	Do not affect this bit and the interrupt request enable bit (input capture 1) (ICE1) in the input capture state control register (ICS).
1	Clear the interrupt request enable bit (input capture 1) (ICE1) in the input capture state control register (ICS).

**[bit4] ICE0C: ICE0 clear bit**

Writing "1" to this bit clears the ICE0 bit in the ICS register.

Value	Description
0	Do not affect this bit and the interrupt request enable bit (input capture 0) (ICE0) in the input capture state control register (ICS).
1	Clear the interrupt request enable bit (input capture 0) (ICE0) in the input capture state control register (ICS).

**[bit3:0] Reserved: Reserved bits**



#### 4.4. Input Capture State Control Set Register (ICSS)

The input capture state control set register (ICSS) is a register that sets the bit in the input capture state control register (ICS).

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,W0
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved		ICE1S	ICE0S	Reserved			
ACCESS_TYPE	R0,W0		R0,W	R0,W	R0,W0			
PROT_TYPE	-							
INITIAL_VALUE	00		0	0	0000			

**[bit31:6] Reserved: Reserved bits**

##### **[bit5] ICE1S: ICE1 set bit**

Writing "1" to this bit sets the ICE1 bit in the ICS register to "1".

Value	Description
0	Do not affect this bit and the interrupt request enable bit (input capture 1) (ICE1) in the input capture state control register (ICS).
1	Set the interrupt request enable bit (input capture 1) (ICE1) in the input capture state control register (ICS).

##### **[bit4] ICE0S: ICE0 set bit**

Writing "1" to this bit sets the ICE0 bit in the ICS register to "1".

Value	Description
0	Do not affect this bit and the interrupt request enable bit (input capture 0) (ICE0) in the input capture state control register (ICS).
1	Set the interrupt request enable bit (input capture 0) (ICE0) in the input capture state control register (ICS).

**[bit3:0] Reserved: Reserved bits**

## 5. Precautions for Using

The following shows the notes when using the 32-bit input capture.

### (1) Notes when Accessing a Register

#### a) When Accessing the Input Capture Data Registers 0, 1 (IPCP0, 1)

Use word access instructions for the input capture data registers 0, 1 (IPCP0, 1).

#### b) When Accessing the Input Capture State Control Register (ICS)

- This register supports writing from the bit-band alias area. For the bit-band alias area, see "CHAPTER: Bit-band Unit".
- To clear a specified bit in this register, clear the bit by writing "1" to the applicable bit in the input capture state control clear register (ICSC).
- To set a specified bit in this register, set the bit by writing "1" to the applicable bit in the input capture state control set register (ICSS).
- Data can be written directly to this register only when writing to all bits.

#### c) Notes on Interrupt Processing

- The valid edge indication bit (IEI1, IEI0) in the input capture state control register (ICS) indicates a detected latest edge when the level of the external input pin (IN0, IN1) switches while an interrupt routine is processing after the interrupt request flag (ICP1, ICP0) in the input capture state control register (ICS) is set to "1".

### (2) Notes on Input Capture Operation

#### About Capture Timing

- Capture resolution is 1 peripheral clock because the input capture operates according to the peripheral clock timing. The resolution is 1 timer count because the capture data is the timer counter value of the free-run timer.





## CHAPTER 42: FlexRay/RDC Dedicated Clock

This chapter explains FlexRay/RDC dedicated clock.

---

1. Overview
2. Configuration
3. Operation
4. Setting Procedure Example
5. Registers
6. Precautions for Using





## 1. Overview

This section gives a brief overview of FlexRay/RDC dedicated clock.

This product includes a PLL for FlexRay/RDC separately from the PLL for the CPU core source clock.

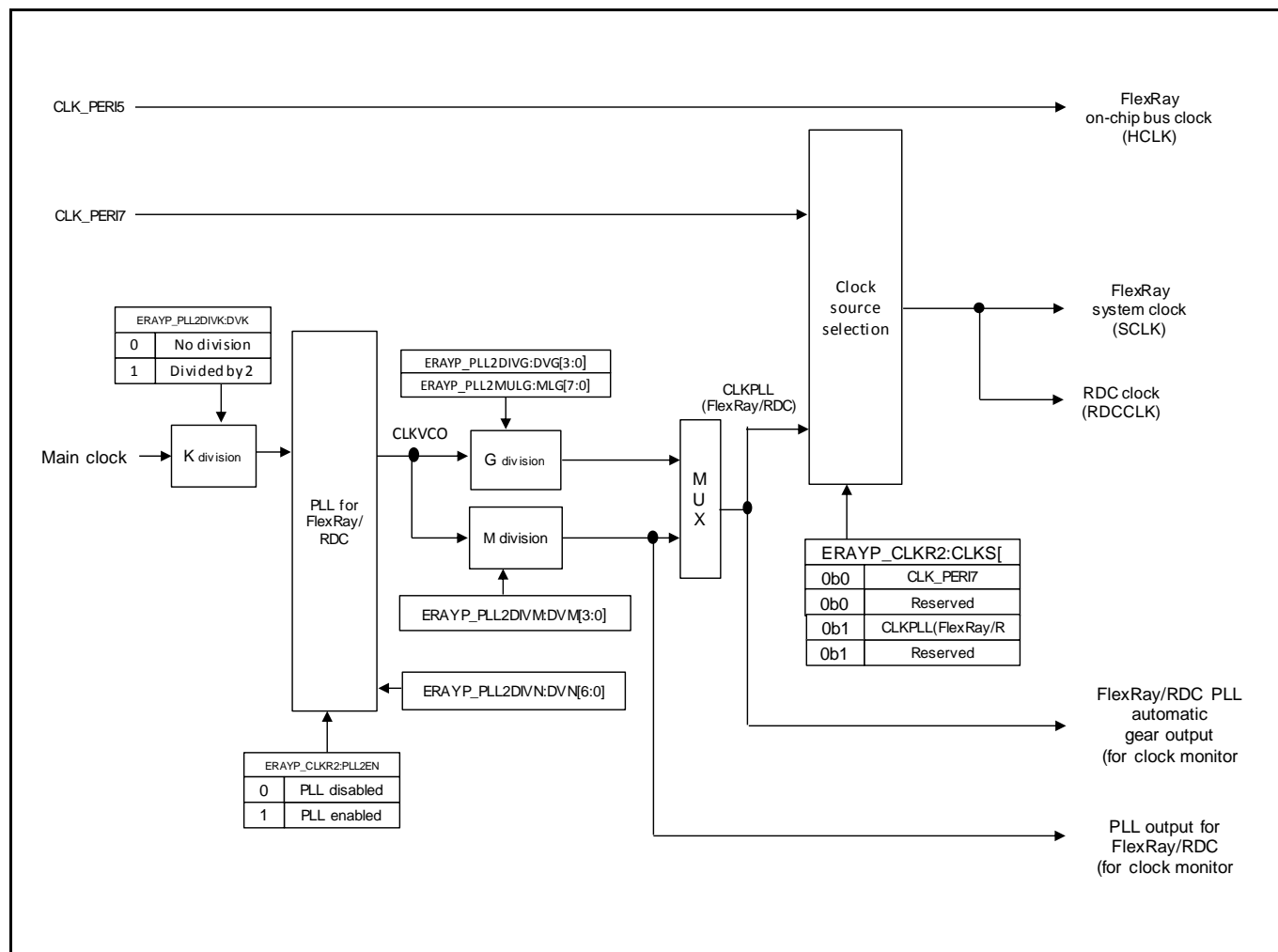
This module controls PLL oscillation and the clock for FlexRay/RDC.

- PLL multiplication rate can be set freely
- Clock automatic gear up/down function for preventing voltage drops and voltage surges
- FlexRay system clock (SCLK) and RDC clock (RDCCLK) source selection function
- Interrupt generation function triggered by the detection of the deadlock state of the FlexRay/RDC PLL macro

## 2. Configuration

This section shows a block diagram of FlexRay/RDC dedicated clock.

Figure 2-1 Block Diagram of FlexRay/RDC Dedicated Clock





### 3. Operation

This section describes the operation of FlexRay/RDC dedicated clock.

#### (1) FlexRay/RDC PLL Control

After initialization, FlexRay/RDC PLL oscillation is stopped. While stopped, FlexRay/RDC PLL output cannot be selected as the clock source. After starting the program, first set the multiplier for FlexRay/RDC PLL for the clock source, and wait until FlexRay/RDC PLL is locked. Then change the clock source. While waiting for FlexRay/RDC PLL to lock, wait for 200 $\mu$ s or more. When FlexRay/RDC PLL output is selected as the clock source, FlexRay/RDC PLL cannot be stopped.

Writing to the register has no affect. Select the CLK\_PERI7 as the clock source before operations such as stopping the FlexRay/RDC PLL for changing to the stop mode.

#### (2) FlexRay/RDC PLL Multiplier

Changing the FlexRay/RDC PLL multiplier setting to a value other than the initial value must be performed during or before enabling FlexRay/RDC PLL after the program is started.

After changing the multiplier setting, wait for FlexRay/RDC PLL lock time, and then switch the clock source. While waiting for FlexRay/RDC PLL to lock, it is recommended to use the main clock timer interrupt.

To change the FlexRay/RDC PLL multiplier setting during normal operation, first change the clock source to one other than FlexRay/RDC PLL. As with the above, after changing the multiplier setting, wait for the FlexRay/RDC PLL to lock, and then switch the clock source.

#### (3) Clock Automatic Gear Up/Down

In order to avoid voltage drops or surges when the clock source is switched from oscillation to high frequency PLL output (or vice versa), a circuit is mounted in the PLL interface for the FlexRay/RDC so that clock gear-up and gear-down can be executed smoothly.

Main functions are implemented using two division counters (M division counter and G division counter).

In the M division counter, the target frequency is designated for PLL feedback.

In the G division counter, the frequency increases from the programmable frequency division designated in the G division setting (ERAYP\_PLL2DIVG:DVG[3:0]) to the target frequency designated in the M division setting (DIVM), and decreases from M division setting (ERAYP\_PLL2DIVM:DVM[3:0]) to the programmable end frequency (ERAYP\_PLL2DIVG:DVG[3:0]).

When the system clock is changed from a lower frequency to a higher frequency (gear up), or from a higher frequency to a lower frequency (gear down), only the DIVG > DIVM setting becomes the effective clock gear specification.

Frequency step is conducted using the PLL output frequency multiplication value as shown below.

Oscillator=4 MHz, M=4, N=80 (Ex.: If PLL output = 320 MHz and frequency output to FlexRay/RDC=80MHz, the frequency multiplication value will be N = 80)

The gear divider can be set to any even number divider.

#### (4) CSV Control of FlexRay/RDC PLL

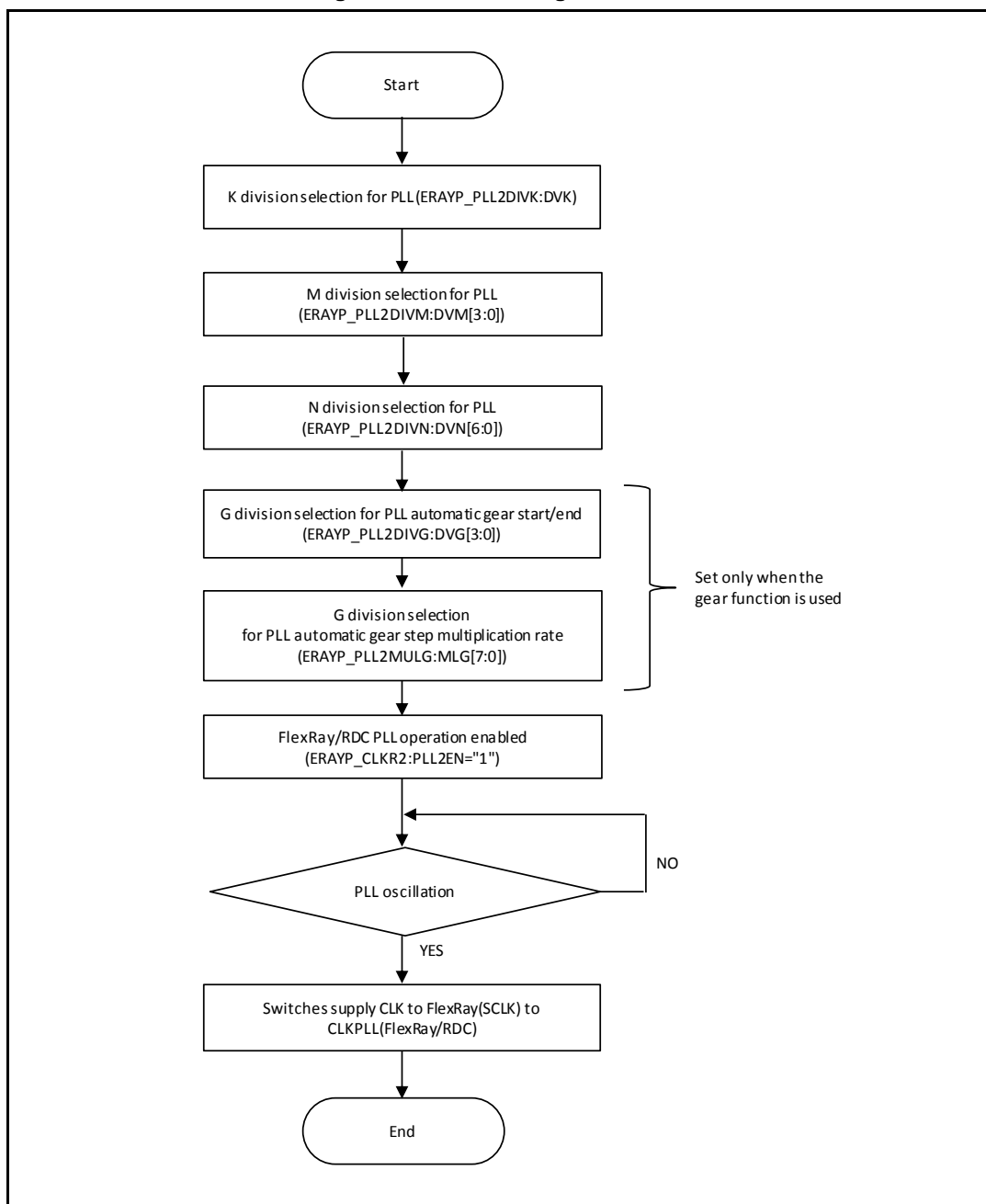
About CSV control for FlexRay/RDC, see Section configuration of sub-system PLL clock supervisor in "CHAPTER: Clock Supervisor".

## 4. Setting Procedure Example

This section shows an example of the FlexRay/RDC dedicated clock setting procedure.

### (1) Clock Setting Procedures

Figure 4-1 Clock Setting Procedures



Use the main timer for the stabilization wait time of FlexRay/RDC PLL, and wait for the switching time.

**Note:**

- If CLKPLL (FlexRay/RDC) is selected as the clock source (ERAYP\_CLKR2:CLKS[1:0]="0b10"), the register values of ERAYP\_PLL2DIVK, ERAYP\_PLL2DIVM, ERAYP\_PLL2DIVN, ERAYP\_PLL2DIVG, and ERAYP\_PLL2MULG cannot be changed.



## (2) Recommended Setting

Table 4-1 Recommended Setting

Main Clock (MCLK) [MHz]	Frequency Parameter			PLL Output for FlexRay/RDC (CLKVCO) [MHz]	FlexRay/RDC Clock (SCLK/RDCCLK) [MHz]
	DVK	DVM[3:0]	DVN[6:0]		
4	0	0011	100_1111	320	80
20	1	0011	100_1111	320	80

To use FlexRay/RDC, it is recommended to set the values as shown in the above table.

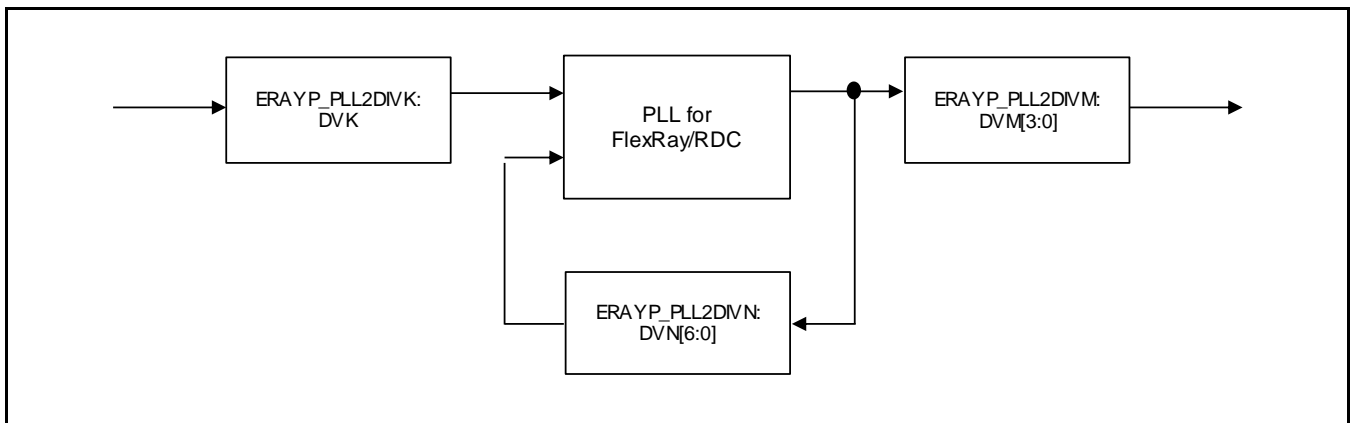
### Note:

- Set the FlexRay/RDC dedicated clock to 80MHz.

## (3) Calculating Frequency

- PLL input frequency for FlexRay/RDC = (Main clock frequency) / (ERAYP\_PLL2DIVK:DVK division ratio)
- PLL multiplication rate for FlexRay/RDC = (ERAYP\_PLL2DIVN:DVN[6:0] multiplication rate)
- PLL output frequency for FlexRay/RDC = (PLL input clock frequency for FlexRay/RDC) × PLL multiplication rate for FlexRay/RDC
- FlexRay/RDC clock frequency = (PLL macro oscillating clock frequency for FlexRay/RDC) / (ERAYP\_PLL2DIVM:DVM[3:0] division ratio)

Figure 4-2 Configuration of FlexRay/RDC Dedicated Clock



### Note:

- The PLL macro oscillating clock frequency for FlexRay/RDC has upper and lower limits. Set the PLL multiplication rate within the following range.
- $200 \text{ MHz} \leq \text{PLL macro oscillating clock frequency} \leq 400 \text{ MHz}$

#### (4) Clock Automatic Gear Procedure Example

1. Set the FlexRay/RDC PLL interface register (ERAYP\_PLL2DIVN, ERAYP\_PLL2DIVM, ERAYP\_PLL2DIVG, ERAYP\_PLL2MULG) according to the selected frequency and gear time.
2. Set the FlexRay/RDC PLL to ON (ERAYP\_CLKR2:PLL2EN=1).
3. When an interrupt was received after switching the gear up or down, enable the relevant interrupt (ERAYP\_PLL2CTRL:IEUP, ERAYP\_PLL2CTRL:IEDN).
4. Wait for the PLL stabilization wait time.
5. Switch the clock source to CLKPLL (FlexRay/RDC) (ERAYP\_CLKR2:CLKS[1:0]="0b00" → "0b10").
6. Wait for the ERAYP\_PLL2CTRL:GRUP gear up flag before switching the clock source back to CLK\_PERI7, or check the ERAYP\_PLL2CTRL:GRUP=1 setting before changing the bit in the ERAYP\_CLKR2 register.
7. Switch the clock source to CLK\_PERI7 (ERAYP\_CLKR2:CLKS[1:0]="0b10" → "0b00").
8. Wait for the PLL2CTRL:GRDN gear down flag before switching the clock source back to CLKPLL, or check the ERAYP\_PLL2CTRL:GRDN=1 setting before changing the bit in the ERAYP\_CLKR2 register.
9. Set the FlexRay/RDC PLL to OFF (ERAYP\_CLKR2:PLL2EN=0).

#### (5) Clock Automatic Gear Up/Down Setting Example

If ERAYP\_PLL2DIVG:DVG[3:0]=4 and ERAYP\_PLL2MULG:MLG[7:0]=20, the following gear up is executed when switching from oscillation to PLL.

1. Step: 1-cycle 16.0MHz (16.0MHz will become 20-cycle PLL output)
  2. Step: 2-cycle 16.8MHz (16.8MHz will become 19-cycle PLL output)
  3. Step: 3-cycle 17.8MHz (17.8MHz will become 18-cycle PLL output)
  - :
  16. Step: 16-cycle 64.0MHz (64.0MHz will become 5-cycle PLL output)
  17. Step: 17-cycle 80.0MHz (80.0MHz will become 4-cycle PLL output)
  18. Step: 18-cycle 106.7MHz (106.7MHz will become 3-cycle PLL output)
  19. Step: 19-cycle 160.0MHz (160.0MHz will become 2-cycle PLL output)
- The target frequency reached in the final transition step (16. to 17. in this example)

When a multiplication value is set in the gear multiplication rate register, each step will be multiplied.

The time after the start frequency is generated until the target frequency is reached can be calculated by the following formula.

$$duration = mul \cdot t \cdot \left[ \sum_{k=1}^i k \cdot (i - k + 1) - \sum_{k=j+1}^i k \cdot (i - k + 1) \right]$$

This formula is the same as the following formula (the finite arithmetic series in the first sum term comes down to the following).

$$duration = mul \cdot t \cdot \left[ \frac{i \cdot (i + 1) \cdot (i + 2)}{6} - \sum_{k=j+1}^i k \cdot (i - k + 1) \right]$$

i = G, j = G - M, Mul = MULG, t = 1/f (PLLOUT)

The setting above becomes equal to the 1483PLL output clock cycle, for which the time from the start frequency to the target frequency is 9,262,500ps (approximately 9.3 μs).



## 5. Registers

This section describes the registers of FlexRay/RDC dedicated clock.

**Table 5-1 List of FlexRay/RDC Dedicated Clock Registers**

Abbreviated Register Name	Register Name	See
ERAYP_CSVR	FlexRay/RDC PLL CSV Control Register	5.1
ERAYP_PLL2DIVM	FlexRay/RDC PLL Multiplication Rate (M Division) Selection Register	5.2
ERAYP_PLL2DIVN	FlexRay/RDC PLL Multiplication Rate (N Division) Selection Register	5.3
ERAYP_PLL2DIVG	FlexRay/RDC PLL Automatic Gear Multiplication Rate (G Division) Selection Register	5.4
ERAYP_PLL2MULG	FlexRay/RDC PLL Step Multiplication Rate (G division) Selection Register	5.5
ERAYP_PLL2CTRL	Automatic Gear Control Register	5.6
ERAYP_PLL2DIVK	FlexRay/RDC PLL Multiplication Rate (K Division) Selection Register	5.7
ERAYP_CLKR2	FlexRay/RDC PLL Clock Output Control Register	5.8
ERAYP_PLL2CTRLF	Automatic Gear Control Flag Register	5.9
ERAYP_CLKR2F	FlexRay/RDC PLL Clock Output Control Flag Register	5.10
ERAYP_PLL2CTRLC	Automatic Gear Control Clear Register	5.11
ERAYP_CLKR2C	FlexRay/RDC PLL Clock Output Control Clear Register	5.12

## 5.1. FlexRay/RDC PLL CSV Control Register (ERAYP\_CSVR)

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved						CSVSEL	CSVEN
ACCESS_TYPE	R0,WX						R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	000000						0	0

**[bit7:2] Reserved: Reserved bits**

### **[bit1] CSVSEL: CSV selection**

This bit select NMI is generated or reset is generated when CSV for FlexRay/RDC detects abnormal state.

Value	Description
0	NMI is generated at the abnormal state detection.
1	Reset is generated at the abnormal state detection.

### **[bit0] CSVEN: CSV enable**

This bit enables/disables CSV for FlexRay/RDC.

Value	Description
0	CSV for FlexRay/RDC PLL is disabled.
1	CSV for FlexRay/RDC PLL is enabled.





## 5.2. FlexRay/RDC PLL Multiplication Rate (M Division) Selection Register (ERAYP\_PLL2DIVM)

This register selects the FlexRay/RDC PLL clock multiplication rate.

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved				DVM			
ACCESS_TYPE	R0,WX				R/W			
PROT_TYPE	-							
INITIAL_VALUE	0000				0000			

[bit7:4] Reserved: Reserved bits

[bit3:0] DVM[3:0]: CLKVCO M division selection

Value	Description
0000	CLKVCO(no division) (initial value)
0001	CLKVCO / 2
0010	CLKVCO / 3
0011	CLKVCO / 4
0100	CLKVCO / 5
0101	CLKVCO / 6
0110	CLKVCO / 7
0111	CLKVCO / 8
...	...
1111	CLKVCO / 16

**Note:**

- No division (:1) can be selected for the M division counter. However this setting is not recommended. The generated output clock should be an odd number of the clock duty ratio (PLL direct output).
- Always select a division ratio that is one or higher, and an even number of division ratio (2, 4, 6, etc.).
- An odd number of division ratio (3, 5, 7, etc.) can be selected for the M division counter. However this setting is not recommended. The generated output clock should be an odd number of the clock duty ratio. Always select an even number of division ratio (2, 4, 6, etc.).
- If CLKPLL (FlexRay/RDC) is selected as the clock source(ERAYP\_CLKR2:CLKS[1:0]="0b10"), the register value cannot be changed.
- When the ERAYP\_PLL2DIVM and ERAYP\_PLL2DIVN registers are changed, it is recommended to disable PLL (ERAYP\_CLKR2: PLL2EN="0") and enable PLL (ERAYP\_CLKR2: PLL2EN="1") later.

### 5.3. FlexRay/RDC PLL Multiplication Rate (N Division) Selection Register (ERAYP\_PLL2DIVN)

This register selects the multiplication rate from the PLL input clock to the FlexRay PLL clock.

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved	DVN						
ACCESS_TYPE	R0,WX	R/W						
PROT_TYPE	-							
INITIAL_VALUE	0	0000000						

[bit7] Reserved: Reserved bits

[bit6:0] DVN[6:0]: N division selection for main clock

Value	Description
000_0000	Main clock (No division) (initial value)
000_0001	Main clock divided by 2
000_0010	Main clock divided by 3
000_0011	Main clock divided by 4
000_0100	Main clock divided by 5
000_0101	Main clock divided by 6
000_0110	Main clock divided by 7
000_0111	Main clock divided by 8
...	...
111_1111	Main clock divided by 128

**Notes:**

- If CLKPLL (FlexRay/RDC) is selected as the clock source (CLKR2:CLKS[1:0]="0b10"), the register value cannot be changed.
- When the ERAYP\_PLL2DIVM and ERAYP\_PLL2DIVN registers are changed, it is recommended to disable PLL for FlexRay/RDC (ERAYP\_CLKR2:PLL2EN="0") and enable PLL for FlexRay/RDC (ERAYP\_CLKR2:PLL2EN=1) later.



## 5.4. FlexRay/RDC PLL Automatic Gear Multiplication Rate (G Division) Selection Register (ERAYP\_PLL2DIVG)

This register selects the FlexRay PLL clock gear multiplication rate.

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved				DVG			
ACCESS_TYPE	R0,WX				R/W			
PROT_TYPE	-							
INITIAL_VALUE	0000				0000			

[bit7:4] Reserved: Reserved bits

[bit3:0] DVG[3:0]: G division selection for PLL automatic gear start/end

Value	Description
0000	Automatic gear disabled (initial value)
0001	CLKVCO: 2 (Divided by 2)
0010	CLKVCO: 3 (Divided by 3)
0011	CLKVCO: 4 (Divided by 4)
0100	CLKVCO: 5 (Divided by 5)
0101	CLKVCO: 6 (Divided by 6)
0110	CLKVCO: 7 (Divided by 7)
0111	CLKVCO: 8 (Divided by 8)
...	...
1111	CLKVCO: 16 (Divided by 16)

### Notes:

- For details on using this function, see Section "(3) Clock Automatic Gear Up/Down".
- An odd number of division ratio (3, 5, 7, etc.) can be selected for the G division counter. However this setting is not recommended. Always select an even number of division ratio (2, 4, 6, etc.).
- If CLKPLL (FlexRay/RDC) is selected as the clock source (ERAYP\_CLKR2:CLKS[1:0]="0b10"), the register value cannot be changed.

## 5.5. FlexRay/RDC PLL Step Multiplication Rate (G division) Selection Register (ERAYP\_PLL2MULG)

This register selects the automatic gear step multiplication rate.

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	MLG							
ACCESS_TYPE	R/W							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

### [bit7:0] MLG[7:0]: G division selection for PLL automatic gear step multiplication rate

Value	Description
0000_0000	G division step $\times 1$ (Multiplied by 1)
0000_0001	G division step $\times 2$ (Multiplied by 8)
0000_0010	G division step $\times 3$ (Multiplied by 8)
0000_0011	G division step $\times 4$ (Multiplied by 8)
0000_0100	G division step $\times 5$ (Multiplied by 8)
0000_0101	G division step $\times 6$ (Multiplied by 8)
0000_0110	G division step $\times 7$ (Multiplied by 8)
0000_0111	G division step $\times 8$ (Multiplied by 8)
...	...
1111_1111	G division step $\times 256$ (Multiplied by 256)

#### Notes:

- For details on using this function, see "(3) Clock Automatic Gear Up/Down".
- If CLKPLL (FlexRay/RDC) is selected as the clock source (ERAYP\_CLKR2:CLKS[1:0]="0b10"), the register value cannot be changed.



## 5.6. Automatic Gear Control Register (ERAYP\_PLL2CTRL)

This register sets the automatic gear operation control.

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved				IEDN	Reserved	IEUP	Reserved
ACCESS_TYPE	R0,WX				R/W	R0,WX	R/W	R0,WX
PROT_TYPE	-							
INITIAL_VALUE	0000				0	0	0	0

**[bit7:4] Reserved: Reserved bits**

**[bit3] IEDN: Interrupt enable gear down**

Value	Description
0	Gear down interrupt disabled (initial value)
1	Gear down interrupt enabled

To receive interrupt after switching the gear down, enable interrupt.

**[bit2] Reserved: Reserved bit**

**[bit1] IEUP: Interrupt enable gear up**

Value	Description
0	Gear up interrupt disabled (initial value)
1	Gear up interrupt enabled

To receive interrupt after switching the gear up, enable interrupt.

**[bit0] Reserved: Reserved bit**

## 5.7. FlexRay/RDC PLL Multiplication Rate (K Division) Selection Register (ERAYP\_PLL2DIVK)

This register selects the FlexRay/RDC PLL clock division.

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved							DVK
ACCESS_TYPE	R0,WX							R/W
PROT_TYPE	-							
INITIAL_VALUE	0000000							0

**[bit7:1] Reserved: Reserved bit**

### **[bit0] DVK: K division selection for main clock**

This bit selects the main clock (MCLK) division for the FlexRay/RDC PLL input clock as follows.

Value	Description
0	Main clock/ 1 (no division) (initial value)
1	Main clock/ 2 (divided by 2)

#### **Notes:**

- If CLKPLL (FlexRay/RDC) is selected as the clock source (ERAYP\_CLKR2:CLKS[1:0]="0b10"), the register value cannot be changed.
- To set the main clock for the FlexRay/RDC PLL input clock to 20MHz, set this bit to "1". For a setting example, see Section "4. Setting Procedure Example".



## 5.8. FlexRay/RDC PLL Clock Output Control Register (ERAYP\_CLKR2)

This register sets FlexRay operation control.

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	FPOVF	Reserved	FPOVIE	Reserved	Reserved	PLL2EN	CLKS	
ACCESS_TYPE	R,WX	R0,WX	R/W	R0,W0	R/W0	R/W	R/W	
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	00	

### [bit7] FPOVF: FlexRay/RDC PLL alarm flag

When a deadlock state of FlexRay/RDC PLL macro is detected, this flag is raised.

Value	Description
0	Ordinary lock state
1	Deadlock state

### [bit6] Reserved: Reserved bit

### [bit5] FPOVIE: FlexRay/RDC PLL alarm interrupt request enabled

When FlexRay/RDC PLL alarm interrupt request flag bit becomes "1", this bit determines whether to generate FlexRay/RDC PLL alarm interrupt request.

Value	Description
0	Interrupt request disabled
1	Interrupt request enabled

### [bit4] Reserved: Reserved bit

### [bit3] Reserved: Reserved bit

### [bit2] PLL2EN: FlexRay/RDC PLL selection enabled

This bit sets FlexRay PLL operation as follows.

Value	Description
0	Main clock/ 1 (no division) (initial value)
1	Main clock/ 2 (divided by 2)

It is prohibited to change the FlexRay/RDC PLL operation enable bit(PLL2EN) when CLKPLL(FlexRay/RDC) is selected as the clock source(CLKS[1:0]="0b10").

### [bit1:0] CLKS[1:0]: SCLK output selection

These bits set the selection of SCLK and RDCCLK output from FlexRay PLL-I/F as follows.

Value	Description
00	CLK_PERI7 (initial value)
01	Reserved
10	CLKPLL(FlexRay/RDC)
11	Reserved

To use FlexRay/RDC, set CLKS[1:0]="0b10".

**Note:**

- When FlexRay/RDC PLL selection enabled bit (*PLL2EN*) is "1", SCLK output selection bits (*CLKS[1:0]*) is writable.





## 5.9. Automatic Gear Control Flag Register (ERAYP\_PLL2CTRLF)

This register is used to indicate gear up interrupt flag and gear down interrupt flag.

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved					GRDN	Reserved	GRUP
ACCESS_TYPE	R0,WX					R,WX	R0,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	00000					0	0	0

**[bit7:3] Reserved: Reserved bits**

**[bit2] GRDN: Interrupt flag gear down**

Value	Description
0	Gear down interrupt inactive (initial value)
1	Gear down interrupt active

Once the G division counter reaches the programmed end value, this flag is set when the clock source: CLKPLL (FlexRay/RDC) is changed to the clock source: CLK\_PERI7.

**[bit1] Reserved: Reserved bit**

**[bit0] GRUP: Interrupt flag gear up**

Value	Description
0	Gear up interrupt inactive (initial value)
1	Gear up interrupt active

Once the G division counter reaches the end value defined in the M division counter, this flag is set when the clock source: CLK\_PERI7 is changed to the clock source: CLKPLL (FlexRay/RDC).

## 5.10. FlexRay/RDC PLL Clock Output Control Flag Register (ERAYP\_CLKR2F)

This register is used to indicate FlexRay/RDC PLL alarm interrupt request flag.

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved	FPOVIR	Reserved					
ACCESS_TYPE	R0,WX	R,WX	R0,WX					
PROT_TYPE	-							
INITIAL_VALUE	0	0	000000					

**[bit7] Reserved: Reserved bit**

### **[bit6] FPOVIR: FlexRay/RDC PLL alarm interrupt request flag**

- FlexRay/RDC PLL macro alarm interrupt request is indicated with this flag.
- While this bit is set to "1" and FlexRay/RDC PLL alarm interrupt request (FPOVIE) is set to "1", FlexRay/RDC PLL alarm interrupt is generated.

Value	Description
0	Ordinary lock state (initial value)
1	Deadlock state

**[bit5:0] Reserved: Reserved bits**



## 5.11. Automatic Gear Control Clear Register (ERAYP\_PLL2CTRLC)

This register is used to clear the bit in the automatic gear control register.

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved					GRDNC	Reserved	GRUPC
ACCESS_TYPE	R0,W0					R0,W	R0,W0	R0,W
PROT_TYPE	-							
INITIAL_VALUE	00000					0	0	0

**[bit7:3] Reserved: Reserved bits**

### **[bit2] GRDNC: Interrupt flag gear down clear**

Writing "1" to this bit clears the GRDN bit in the ERAYP\_PLL2CTRL register.

Value	Description
0	No effect
1	Clear the interrupt flag gear down

**[bit1] Reserved: Reserved bit**

### **[bit0] GRUPC: Interrupt flag gear up clear**

Writing "1" to this bit clears the GRUP bit in the ERAYP\_PLL2CTRL register.

Value	Description
0	No effect
1	Clear the interrupt flag gear up

## 5.12. FlexRay/RDC PLL Clock Output Control Clear Register (ERAYP\_CLKR2C)

This register is used to clear the bit in the FlexRay/RDC PLL clock output control register.

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved	FPOVIRC	Reserved					
ACCESS_TYPE	R0,W0	R0,W	R0,W0					
PROT_TYPE	-							
INITIAL_VALUE	0	0	000000					

**[bit7] Reserved: Reserved bit**

### **[bit6] FPOVIRC: FlexRay/RDC PLL alarm interrupt request flag clear**

Writing "1" to this bit clears the FPOVIR bit in the ERAYP\_CLKR2 register.

Value	Description
0	No effect
1	Clear the FlexRay/RDC PLL alarm interrupt request flag

**[bit5:0] Reserved: Reserved bits**



## 6. Precautions for Using

This section explains precautions for using the FlexRay/RDC dedicated clock

### (1) Clock

When configuring the FlexRay, do not stop CLK\_PER17.

### (2) Clock Automatic Gear

When using the clock automatic gear function, it is recommended to assess the current state of this function by using the gear up and gear down flags (ERAYP\_PLL2CTRL:GRUP, ERAYP\_PLL2CTRL:GRDN). In this way, it is possible to avoid a malfunction that could occur in the clock system by a setting change before completion.



## CHAPTER 43: Clock Monitor

This chapter explains clock monitor.

---

1. Overview
2. Configuration
3. Operation
4. Registers
5. Notes



## 1. Overview

This section gives a brief overview of clock monitor.

The clock monitor is a macro that outputs internal clock signals to external pins. The clock monitor has a function for dividing the frequency of a clock signal before output to the pin, allowing clock signals to be used for synchronization of external circuits with MCU functions.

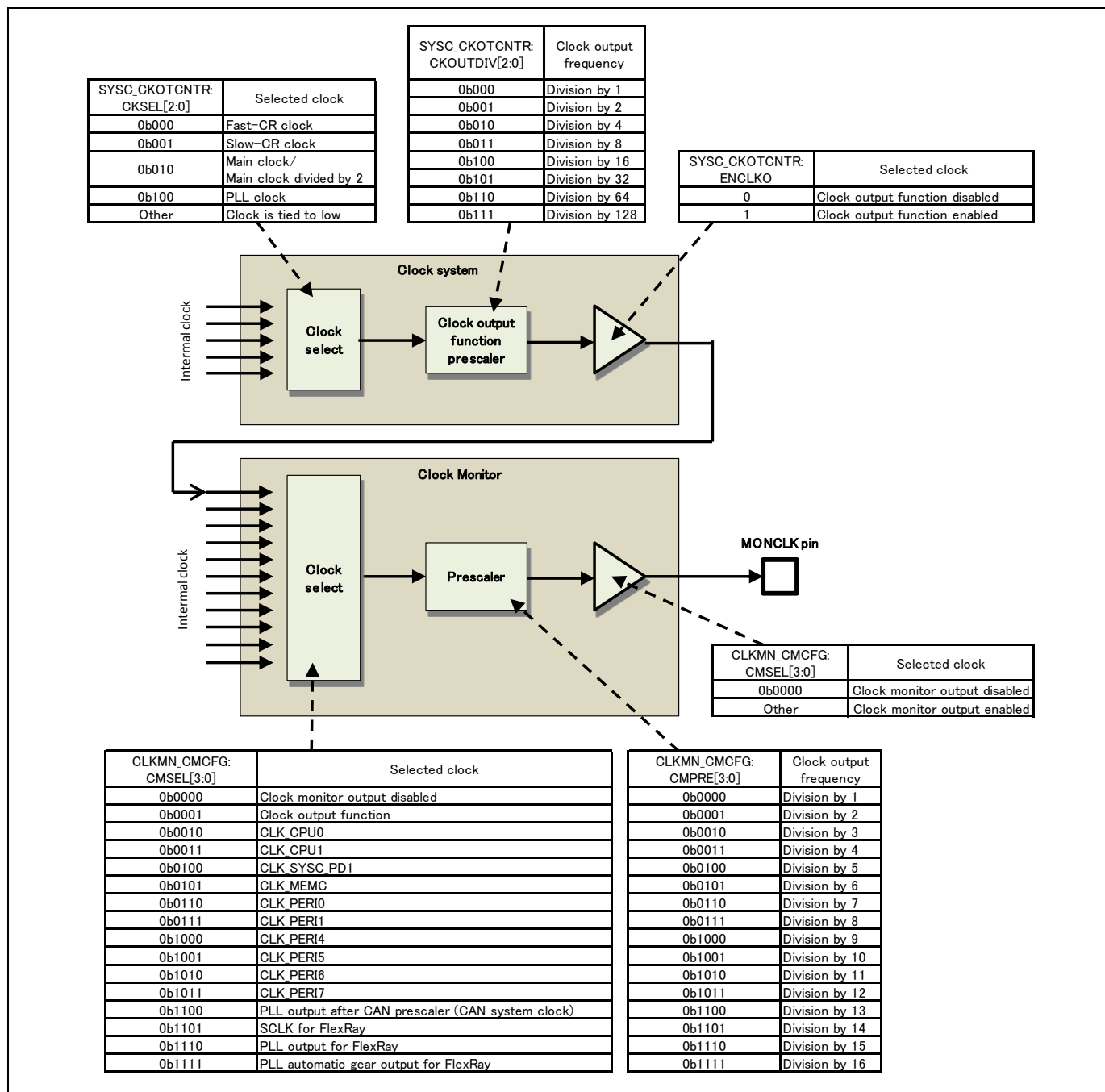
### Features

- Format : Divide the internal clock signal and output to a pin (MONCLK)
- Channels : 1
- Division ratio: CLK/1 to CLK/16
- Fast-CR clock, Slow-CR clock, Main clock/
- Allows for glitchless output
- Programmable mark level (outputs "L" or "H" before the clock output is enabled)
- Interrupts : None
- Stops clock output in stop mode and becomes high impedance

## 2. Configuration

This section shows a block diagram of clock monitor.

Figure 2-1 Configuration Diagram of Clock Monitor



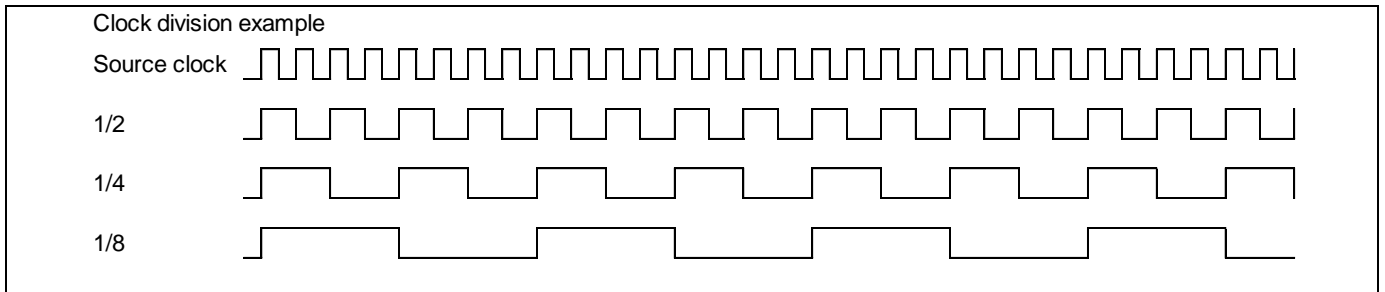




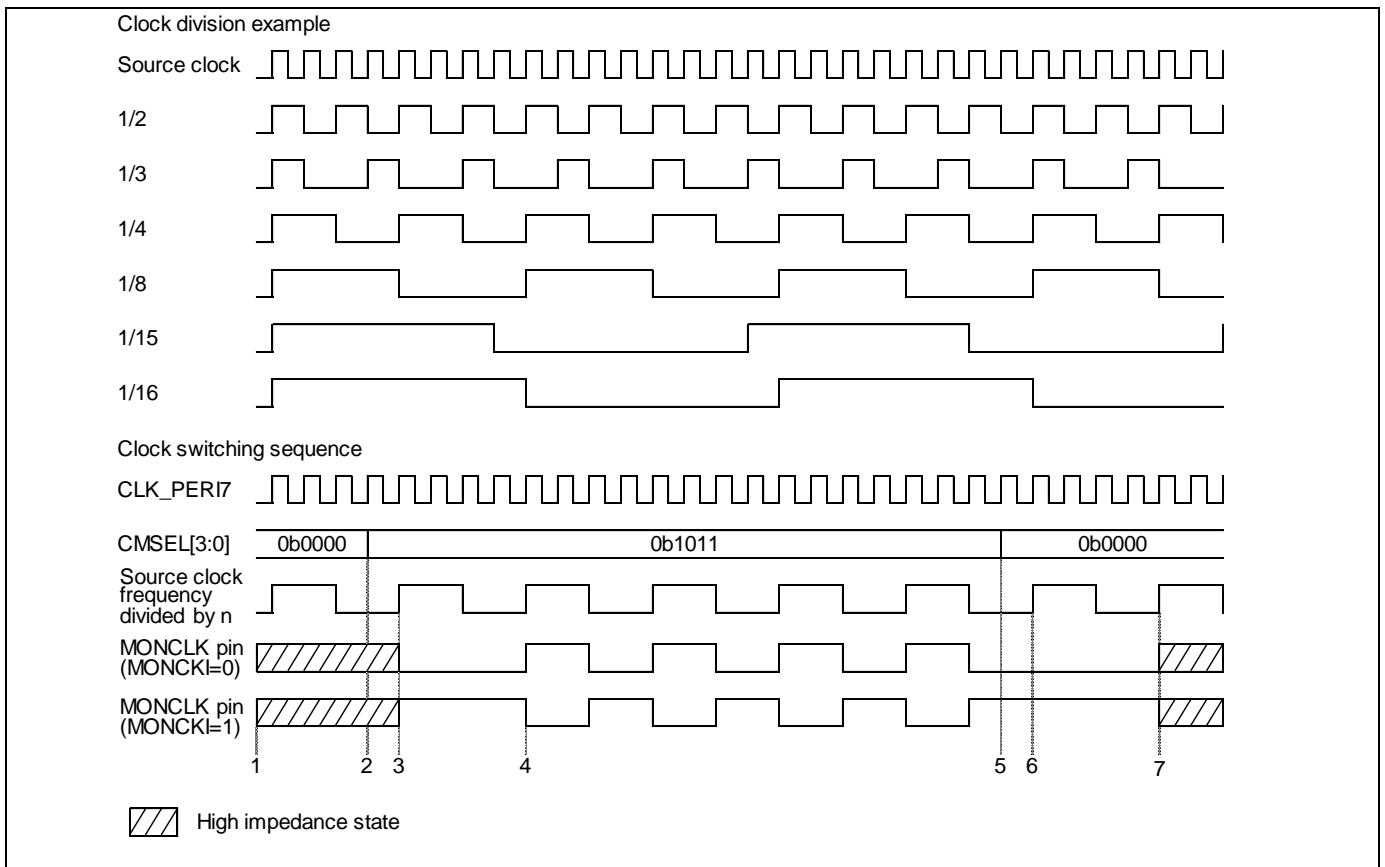
### 3. Operation

This section describes the operation of clock monitor.

**Figure 3-1 Clock Output Function**



**Figure 3-2 Clock Monitor**



1. The MONCLK pin is in the high impedance state.
2. CMSEL[3:0] is set to the selected clock (prescaler) from 0b0000 (no clock selected).
3. The MONCLK pin is set to the output "L" status (or "H" output if MONCKI is set to "1") for the duration of one internal (prescaled) clock.
4. After one period of the selected (prescaler) internal clock, MONCLK outputs the selected (prescaler) internal clock.
5. CMSEL[3:0] is set to 0b0000 (no clock selected) from the selected clock (prescaler).
6. The MONCLK pin is set to the output "L" status (or "H" output if MONCKI is set to "1") for the duration of one internal (prescaled) clock.
7. The MONCLK pin switches to the high impedance state.



4. Registers

This section describes the registers of Clock Monitor.

Table 4-1 List of Clock Monitor Registers

Abbreviated Register Name	Register Name	See
CLKMN_CMCFG	Clock Monitor Control Register	4.1
CLKMN_CSCFG	Clock Control Register	4.2

For configuration of clock output function, see clock output function register (SYSC\_CKOTCNTR) in "CHAPTER: Clock System".



## 4.1. Clock Monitor Control Register (CLKMN\_CMCFG)

This register is used to select monitoring clock and set clock divider.

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	CMPRE				CMSEL			
ACCESS_TYPE	R/W				R/W			
PROT_TYPE	-							
INITIAL_VALUE	0000				0000			

### [bit7:4] CMPRE[3:0]: Output frequency prescaler bits

These bits specify the output frequency of the clock signal to the clock monitor pin.

Value	Description
0000	Source clock divided by 1 (selected by CMSEL) (Initial value)
0001	Source clock divided by 2 (selected by CMSEL)
0010	Source clock divided by 3 (selected by CMSEL)
0011	Source clock divided by 4 (selected by CMSEL)
0100	Source clock divided by 5 (selected by CMSEL)
0101	Source clock divided by 6 (selected by CMSEL)
0110	Source clock divided by 7 (selected by CMSEL)
0111	Source clock divided by 8 (selected by CMSEL)
1000	Source clock divided by 9 (selected by CMSEL)
1001	Source clock divided by 10 (selected by CMSEL)
1010	Source clock divided by 11 (selected by CMSEL)
1011	Source clock divided by 12 (selected by CMSEL)
1100	Source clock divided by 13 (selected by CMSEL)
1101	Source clock divided by 14 (selected by CMSEL)
1110	Source clock divided by 15 (selected by CMSEL)
1111	Source clock divided by 16 (selected by CMSEL)

**[bit3:0] CMSEL[3:0]: Output source clock selection bits**

Value	Description
0000	MONCLK output disabled (high impedance state) (initial value)
0001	Output clock of clock output function
0010	CLK_CPU0
0011	CLK_CPU1
0100	CLK_SYSC_PD1
0101	CLK_MEMC
0110	CLK_PERI0
0111	CLK_PERI1
1000	CLK_PERI4
1001	CLK_PERI5
1010	CLK_PERI6
1011	CLK_PERI7
1100	PLL output after CAN prescaler (CAN system clock)
1101	SCLK for FlexRay
1110	PLL output for FlexRay
1111	PLL automatic gear output for FlexRay

**Note:**

- For details on writing to this register, see Section "5. Notes".



## 4.2. Clock Control Register (CLKMN\_CSCFG)

This register is used to configuration mark level of clock monitor pin.

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved	Reserved	Reserved	MONCKI	Reserved			
ACCESS_TYPE	R/W0	R0,WX	R/W0	R/W	R/W0			
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0000			

**[bit15] Reserved: Reserved bit**

**[bit14] Reserved: Reserved bit**

**[bit13] Reserved: Reserved bit**

**[bit12] MONCKI: Clock monitor MONCLK inverter**

Value	Description
0	MONCLK mark level is low level (initial value)
1	MONCLK mark level is high level

**[bit11:8] Reserved**

**Note:**

- For details on writing to this register, see Section "5. Notes".

## 5. Notes

The following shows the notes when using the clock monitor.

In order to achieve glitch-free switching, use the following procedure when changing the clock source (CMSEL[3:0]) or prescaler ratio (CMPRE[3:0]).

- The CMPRE[3:0] bits can be written only when the CMSEL[3:0] bits are "0b0000".
- The CMPRE[3:0] bits can be written only when "0b0000" is written to the CMSEL[3:0] registers during the same write access.
- Do not change CLKMN\_CMCFG:CMSEL[3:0] and CLKMN\_CSCFG:MONCKI within 2 cycle of monitor clock division after setting CLKMN\_CMCFG:CMSEL[3:0].
- When selecting another effective clock while something is already selected as the clock source (CMSEL is not "0b0000"), first set CMSEL to "0b0000" and check that CMSEL returns "0b0000" on read before writing the target clock setting value to CMSEL.
- If the clock selected as the monitor clock is stopped during monitoring, rewriting to any registers have no effect until the selected clock is started again or the unit is reset.

(Access example)

### 1. Access

CLKMN\_CMCFG:CMSEL[3:0]=0b0000

CLKMN\_CMCFG:CMSEL[3:0]=Prescaler

### 2. Access

CLKMN\_CMCFG:CMSEL[3:0]=Clock

The CSCFG:MONCKI flag can also be written the same as above only when CMSEL[3:0] are "0b0000".

Frequency which can output as monitor output is up to 50MHz. When monitoring clock with 50MHz or more, always divide clock.





## CHAPTER 44: FlexRay Controller

This chapter describes FlexRay controller.

---

1. Overview
2. Configuration
3. Operation
4. Registers





## 1. Overview

The FlexRay controller communicates according to the FlexRay Protocol Specifications V2.1. The bit rate is set to 10 Mbit/s by specifying the maximum system clock.

### Overview of the FlexRay Controller

A message buffer with a length of up to 254 data bytes can be located for FlexRay network communications. The message storage area consists of a single-port message RAM that holds up to 128 message buffers. The message handler provides functions for all message processing. The functions are as follows:

- Acceptance filter
- Message transfer between 2 FlexRay channel protocol controllers and message RAM
- Transmission schedule management
- Providing message status information

The registers of the FlexRay controller are accessible from the host. These registers are used to set/control/monitor the following:

- FlexRay channel protocol controller
- Message handler
- Global time unit
- System universal control
- Frame and symbol processing
- Network management
- Interrupt control
- Access to message RAM via input/output buffer

The FlexRay controller supports the following functions:

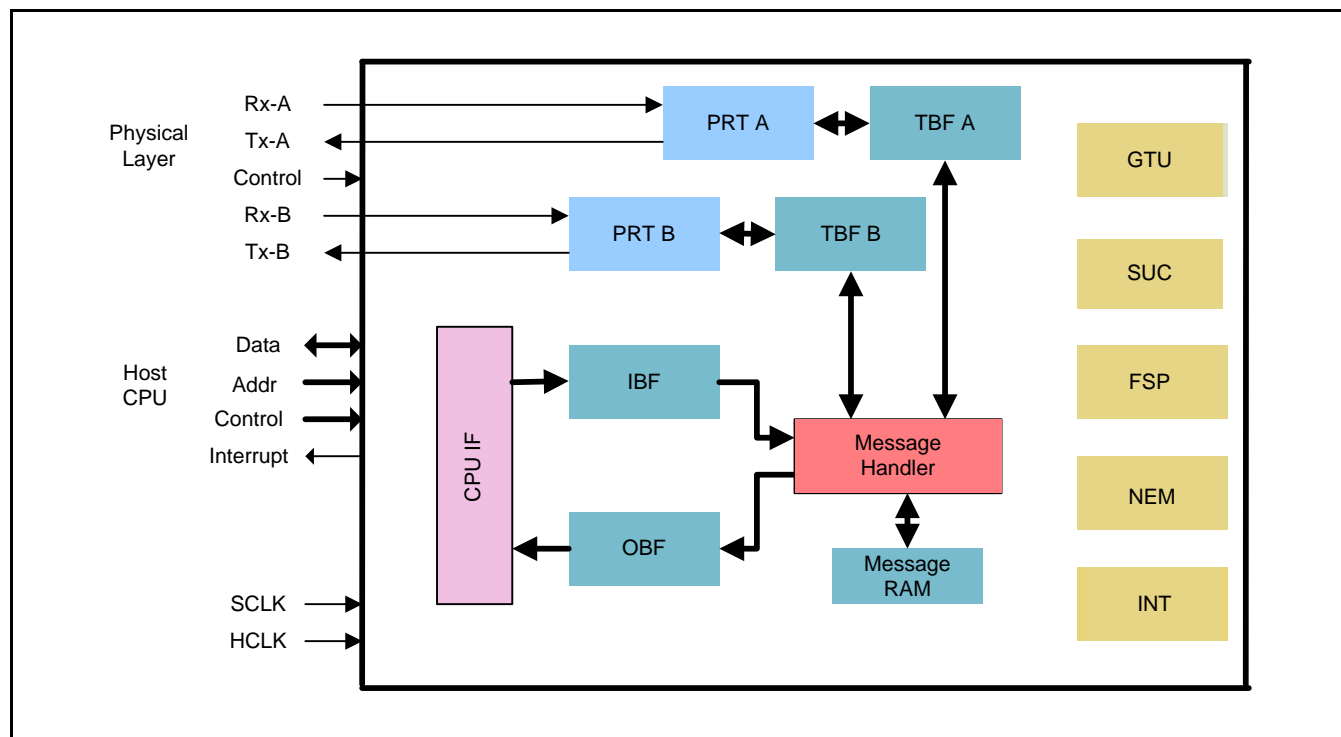
- Complies with the FlexRay Protocol Specifications V2.1
- Up to 10 Mbit/s bit rate on each channel
- Up to 128 message buffers configurable
- 8-Kbyte message RAM (equivalent to the following storage capacity)
  - 128 message buffers with max. 48-byte data section
  - 30 message buffers with max. 254-byte data section
- Variable-length message buffer configuration
- One configurable reception FIFO
- Each message buffer configurable as a reception buffer, as a transmission buffer, or as part of the reception FIFO
- Host access to message buffers via input and output buffers
  - Input buffer: Holds messages to be transferred to the message RAM
  - Output buffer: Holds messages read from the message RAM
- Slot counter, cycle counter, and channel filtering
- Maskable interrupts

Network management support

## 2. Configuration

This section shows the configuration of the FlexRay controller.

Figure 2-1 FlexRay Controller Configuration



Functional Description of Each Block

### (1) CPU Interface (CIF)

This interface connects the host CPU to the FlexRay controller.

### (2) Input Buffer (IBF)

This buffer is used to write to the message buffers configured in the message RAM.

The host CPU can write header and data sections from the input buffer to a specific message buffer.

The message handler transfers data from the input buffer to the selected message buffer in message RAM.

### (3) Output Buffer (OBF)

This buffer is used to read the message buffers configured in the message RAM.

The message handler transfers data from the selected message buffer to the output buffer.

Once the data transfer is complete, the host CPU can read the header section and data section of the message buffer that was transferred from the output buffer.

### (4) Message Handler (MHD)

The message handler controls the data transfers between the following components:

- Input/Output buffer and message RAM
- Transient buffer RAM of 2 FlexRay protocol controllers and message RAM



#### **(5) Message RAM (MRAM)**

The message RAM consists of a single-port RAM that stores configuration data (header and data) for up to 128 FlexRay message buffers.

#### **(6) Transient Buffer RAM (TBF A/B)**

This RAM stores the data sections of 2 messages.

#### **(7) FlexRay Channel Protocol Controller (PRT A/B)**

The FlexRay channel protocol controller consists of a shift register and the FlexRay protocol FSM.

The protocol controller provides the following functions:

- Checking and controlling bit timings
- Receiving/Transmitting FlexRay frames and symbols
- Checking the header CRC
- Generating/Checking frame CRC
- Connecting to the bus driver

In addition, this block connects to the following blocks:

- Physical layer (bus driver)
- Transient buffer RAM
- Message handler
- Global time unit
- System universal control
- Frame and symbol processing
- Network management
- Interrupt control

#### **(8) Global Time Unit (GTU)**

The global time unit provides the following functions:

- Microtick generation
- Macrotick generation
- Fault-tolerant clock synchronization using the FTM algorithm
  - Rate correction
  - Offset correction
- Cycle counter
- Static segment timing control
- Dynamic segment (minislot) timing control
- Support for external clock correction

#### **(9) System Universal Control (SUC)**

The system universal control controls the following functions:

- Configuration
- Wakeup
- Startup
- Normal operation
- Passive operation
- Monitor mode

### **(10) Frame and Symbol Processing (FSP)**

Frame and symbol processing controls the following functions:

- Checking to ensure that the timing of frames and symbols is correct
- Testing the syntactic and semantic validity of reception frames
- Setting the slot status flag

### **(11) Network Management (NEM)**

The network management provides the following function:

- Handling of the network management vector

### **(12) Interrupt Control (INT)**

The interrupt control provides the following functions:

- Provision of error and interrupt flags
- Controlling the enabling/disabling of interrupt factors
- Controlling the allocation of interrupt factors to 2 module interrupt lines
- Enabling/disabling 2 module interrupt lines
- Managing 2 interrupt timers
- Stopping watch time capturing



### 3. Operation

This section describes the function of the FlexRay protocol. For more detailed information on the FlexRay protocol, see the FlexRay Protocol Specifications V2.1.

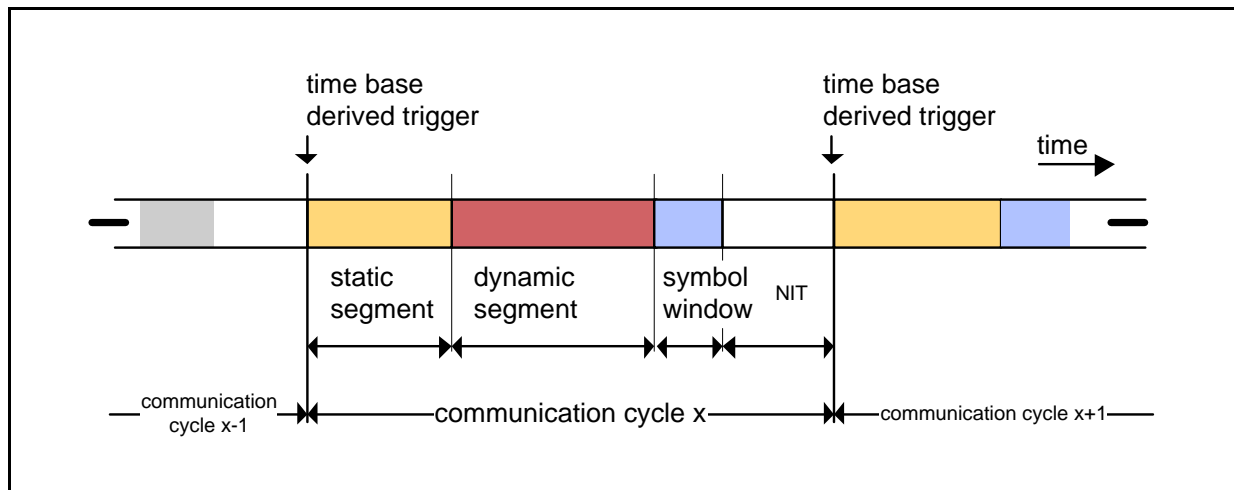
#### 3.1. Communication Cycle

The FlexRay communication cycle consists of the following elements:

- Static segment
- Dynamic segment (optional)
- Symbol window (optional)
- Network idle time (NIT)

The network communication time (NCT) consists of a static segment, dynamic segment, and symbol window. Starting at 1, the slot counter for each communication channel is incremented by the end of the dynamic segment. Also, both channels use the same synchronized macrotick.

**Figure 3-1 Structure of Communication Cycle**



##### (1) Static Segment

The static segment has the following features.

- The bus guardian (if available) protects slots.
- Frame transmission begins at the action point of each static slot.
- The payload length is the same in all the frames of both channels.

Parameters:

Number of static slots, GTUC7:NSS[9:0]

Static slot length, GTUC7:SSL[9:0]

Static frame data length, MHDC:SFDL[6:0]

Action point offset, GTUC9:APO[5:0]

## (2) Dynamic Segment

The dynamic segment has the following features.

- The bus guardian (even if available) is disabled, and all controllers have bus access.
- The slot length is variable and different in both channels.
- Transmission begins at a minislot action point.

Parameters:

Number of minislots, GTUC8:NMS[12:0]

Minislot length, GTUC8:MSL[5:0]

Minislot action point offset, GTUC9:MAPO[4:0]

Transmission-end minislot value, MHDC:SLT[12:0]

## (3) Symbol Window

The FlexRay Protocol Specifications V2.1 defines 3 symbols.

- Wakeup symbol (WUS): Transmitted only in the WAKEUP state
- Collision avoidance symbol (CAS): Transmitted only in the STARTUP state
- Media access test symbol (MTS): Transmitted in the NORMAL\_ACTIVE state to test the bus guardian

During the symbol window period, 1 MTS symbol is transmitted per channel.

The symbol window has the following features.

- 1 symbol is transmitted.
- MTS symbol transmission begins at a symbol window action point.

Parameters:

Action point offset, GTUC9:APO[5:0]

Network idle time start, GTUC4:NIT[13:0]

## (4) Network Idle Time (NIT)

During the network idle time (NIT), the FlexRay controller performs the following tasks:

- Calculating the clock correction time (offset and rate)
- Performing offset correction for each macrotick after the start of offset correction
- Performing cluster cycle-related tasks

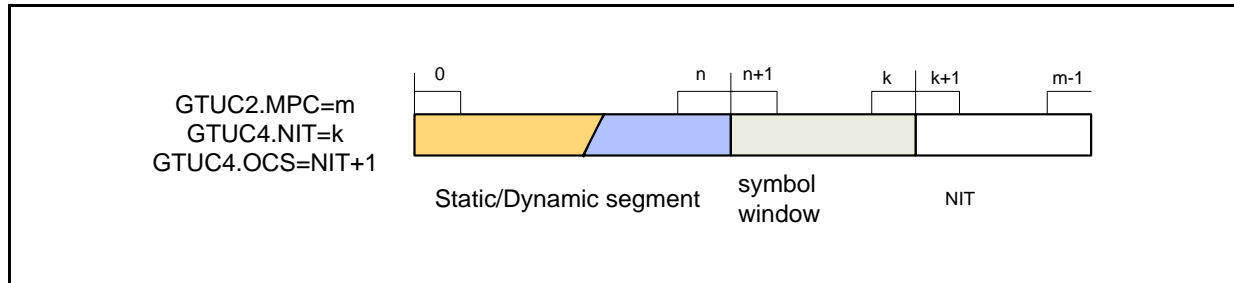
Parameters:

Network idle time start, GTUC4:NIT[13:0]

Offset correction start, GTUC4:OCS[13:0]

## (5) NIT Start and Offset Correction Start Settings

Figure 3-2 NIT Start and Offset Correction Start Settings



The setting of GTUC2:MPC=m assumes that the number of microticks per cycle, gMacroPerCycle, is m. Also, the static/dynamic segment is assumed to start at macrotick 0 and end at macrotick n.

$n = \text{static segment length} + \text{dynamic segment offset} + \text{dynamic segment length} - 1 \text{ MT}$   
 $= g\text{NumberOfStaticSlots} \times g\text{dStaticSlot} + \text{dynamic segment offset}$   
 $+ g\text{NumberOfMinislots} \times g\text{dMinislot} - 1 \text{ MT}$

The static segment length is set in GTUC7:SSL and GTUC7:NSS.

The dynamic segment length is set in GTUC8:MSL and GTUC8:NMS.

The dynamic segment offset can be calculated as follows:

if  $g\text{dActionPointOffset} \leq g\text{dMinislotActionPointOffset}$ :

Dynamic segment offset = 0 MT

else if  $g\text{dActionPointOffset} > g\text{dMinislotActionPointOffset}$ :

Dynamic segment offset =  $g\text{dActionPointOffset} - g\text{dMinislotActionPointOffset}$

If NIT starts at macrotick k+1 and ends at the final macrotick of m-1 cycles, the setting is as follows:

GTUC4:NIT = k

Also, the offset correction start is configured to satisfy the following condition:

$GTUC4:OCS \geq GTUC4:NIT + 1 = k + 1$

The symbol window length between the end of the static/dynamic segment and the NIT start is calculated as k-n.

## 3.2. Communication Modes

FlexRay protocol v2.1 supports time-triggered distributed (TT-D) mode. This section describes the communication modes that use time-triggered distributed (TT-D) mode.

### **Time-triggered Distributed Mode (TT-D: Time-triggered Distributed)**

The following communication modes are available in TT-D mode:

- Pure static: Minimum of 2 static slots + symbol window (optional)
- Mixed static/dynamic: Minimum of 2 static slots + dynamic segment + symbol window (optional)

Operation in time-triggered distributed mode requires a minimum of 2 cold start nodes. Also, cluster startup requires 2 fault-free cold start nodes. Each startup frame must be a synchronization frame. All the cold start nodes are synchronization nodes.





### 3.3. Clock Synchronization

TT-D mode uses distributed clock synchronization. Each node synchronizes with the cluster by measuring the reception timing of synchronization frames from other nodes.

#### 3.3.1. Global Time

Each node operates according to the concept of global time, although the node has its own clock. Global time consists of a vector of 2 values: cycle (cycle counter), and cycle time (macrotick counter).

- Macrotick (MT) = Base unit of FlexRay network time measurement  
(A macrotick is an integral multiple of a microtick ( $\mu T$ ).)
- Cycle = Unit that represents the duration of 1 communication cycle  
(A cycle is expressed in macroticks (MT).)

#### 3.3.2. Local Time

The node operation time is regulated to an accuracy in units of microticks inside the node. A microtick is a controller-specific unit of time obtained from the system clock of individual nodes.

For that reason, different controllers of the same node may have different times. The accuracy of local time error measurement is in units of microticks ( $\mu T$ ).

- Microtick generation order: System clock  $\rightarrow$  prescaler  $\rightarrow$  microtick ( $\mu T$ )
- $\mu T$  = Base unit of time measurement in the FlexRay controller (The clock is corrected in units of  $\mu T$ .)
- Cycle counter + macrotick counter = Local view of the global time of the node

#### 3.3.3. Synchronization Process

Synchronization frames are used as a means of clock synchronization. Only the synchronization nodes configured in advance can transmit synchronization frames. A synchronization node in a 2-channel cluster will need to transmit a synchronization frame to both channels.

FlexRay has the following restrictions for synchronization.

- There is a maximum of 1 synchronization frame per node in 1 communication cycle.
- There is a maximum of 15 synchronization frames per cluster in 1 communication cycle.
- The number of synchronization frames (GTUC2:SNM[3:0]) that are set in advance on all nodes must be used for clock synchronization.
- A minimum of 2 synchronization nodes are required for clock synchronization and startup.

The time deviation between the expected reception time and observed reception time of a synchronization frame received during the static segment period is measured for clock synchronization. The correction time is calculated with the FTM algorithm during the NIT period (offset: all cycles; rate: odd-numbered cycles).

For details, see Chapter 8 in the FlexRay Protocol Specifications V2.1.

#### (1) Offset (Phase) Correction

- The time deviation of the current cycle time is measured.
- For a node with 2 channels, the smaller of the values measured in the channels is adopted as the calculation value.
- This correction is calculated for the NIT periods of all communication cycles.
- The offset correction value calculated with an even-numbered cycle is used only to check for errors.
- Comparisons are performed with the limit value to check for errors.
- The correction value is an integer followed by the symbol  $\mu T$ .
- The correction is performed in an odd-numbered cycle. The offset correction is distributed over each microtick from the start to end of the cycle (end of NIT), and the start position of the next cycle on each node is shifted to lengthen or shorten the current cycle by a number of microticks.

#### (2) Rate (Frequency) Correction

- The difference in the time deviation (time deviation difference) for even-numbered cycles and odd-numbered cycles is measured.
- For a node with 2 channels, the average time deviation difference measured in the channels is adopted as the calculation value.
- This correction is calculated for the NIT periods of odd-numbered communication cycles.
- Cluster drift damping is performed using the global damping value.
- Comparisons are performed with the limit value to check for errors.
- The correction value is an integer followed by the symbol  $\mu T$ .
- The correction is performed in the next even/odd-numbered cycle pair. The correction is distributed over each microtick that makes up 1 cycle, and the start position of the next cycle pair on each node is shifted to lengthen or shorten the current cycle by a number of microticks.

#### (3) Sync Frame Transmission

Sync frames can be transmitted only from buffer 0 and 1. Message buffer 1 is used to transmit a sync frame when the sync frame has different payloads on 2 channels. In this case, the MRC:SPLM bit must be set to "1".

The message buffer used to transmit sync frames must be configured with a key slot ID, which can be set only in the DEFAULT\_CONFIG or CONFIG state.

Nodes that transmit sync frames have the SUCC1:TXSY setting of "1".

### 3.3.4. External Clock Synchronization

There may be significant drifting in independent clusters during normal operation. For a required synchronization operation within an independent cluster, external clock synchronization is necessary even if the synchronization is done on a node in the cluster. The offset correction time and rate correction time for the cluster are inferred by the host, enabling the operation to be accomplished.

- The external offset/rate correction value is a signed integer.
- The external offset/rate correction value is added to the calculated offset/rate correction value.
- The total offset/rate correction time (external and internal) is checked against the set limit value.



### 3.4. Error Handling

The error handling implemented in FlexRay assumes that communication between unaffected nodes is guaranteed during periods where nodes have a lower layer protocol error. In some cases, an operation to restart normal operation of the FlexRay controller must be implemented in an application program.

EIR:PEMC is set to "1" during a transition of the error handling state. Then, an interrupt is generated if the interrupt would be valid. CCEV:ERRM[1:0] indicates the actual error mode.

**Table 3-1 POC Error Modes**

Error Mode	Description
ACTIVE (green)	<p>Full operation State: NORMAL_ACTIVE</p> <p>The FlexRay controller is completely synchronized and supports clock synchronization in the entire cluster. The error status and status change information can be obtained from reading the error interrupt flag and status interrupt flag from the EIR register and SIR register, respectively. An interrupt is generated if the interrupt would be valid.</p>
PASSIVE (yellow)	<p>Limited operation State: NORMAL_PASSIVE, FlexRay controller self-recovery available</p> <p>The FlexRay controller stops transmitting frames and symbols but can process received frames. Clock synchronization continues based on the received frames, but active clock synchronization for the entire cluster is not performed.</p> <p>The error status and status change information can be obtained from reading the error interrupt flag and status interrupt flag from the EIR register and SIR register, respectively. An interrupt is generated if the interrupt would be valid.</p>
COMM_HALT (red)	<p>Operation stopped State: HALT, FlexRay controller self-recovery unavailable</p> <p>The FlexRay controller stops frame and symbol processing, clock synchronization processing, and macrotick generation.</p> <p>The error status and status change information can be obtained from reading the error interrupt flag and status interrupt flag from the EIR register and SIR register, respectively. The bus driver is stopped.</p>

### 3.4.1. Clock Correction Failure Counter

The NORMAL\_ACTIVE state transitions to the NORMAL\_PASSIVE state when the clock correction failure counter reaches the maximum PASSIVE transition time without clock correction, SUCC3:WCP[3:0]. The NORMAL\_ACTIVE/NORMAL\_PASSIVE state transitions to the HALT state when the counter reaches the maximum HALT transition time without clock correction, SUCC3:WCF[3:0].

After passing the startup phase, the clock correction failure counter, CCEV:CCFC[3:0], can monitor the periods during which the node clock correction time cannot be calculated. If either the missing offset correction signal, SFS:MOCS, or the missing rate correction signal, SFS:MRCS, is set to "1", the clock correction failure counter is incremented at the end of an odd-numbered communication cycle.

If neither the missing offset correction signal, SFS:MOCS, nor the missing rate correction signal, SFS:MRCS, is set to "1", the clock correction failure counter is set to "0" at the end of an odd-numbered communication cycle.

The clock correction failure counter stops incrementing when it reaches the maximum HALT transition time without clock correction, SUCC3:WCF[3:0]. (In other words, the counter does not return to 0 even when incremented to its maximum value.) The clock correction failure counter is set to "0" when the CONFIG state transitions to the READY or NORMAL\_ACTIVE state.

**Note:**

- There is no transition to the HALT state when SUCC1:HCSE has not been set.

### 3.4.2. Counter for Cycle Pair Number Required for State Transition from Passive to Active

The passive to active count, SUCC1:PTAC[4:0], controls the transition of POC from the NORMAL\_PASSIVE state to the NORMAL\_ACTIVE state. SUCC1:PTA[4:0] defines the number of even/odd-numbered cycle pairs that must have the valid clock correction time before the transition from the NORMAL\_PASSIVE state to the NORMAL\_ACTIVE state. If SUCC1:PTA[4:0] is set to "0", the transition from the NORMAL\_PASSIVE state to the NORMAL\_ACTIVE state is not possible.

### 3.4.3. HALT Command

The setting where SUCC1:CMD[3:0] is "0110" (CHI command HALT) enables a transition to the HALT state when the host detects an error state.

If the command is executed in the NORMAL\_ACTIVE or NORMAL\_PASSIVE state, POC transitions to the HALT state at the end of the current cycle. If it is executed in any other state, SUCC1:CMD[3:0] becomes "0000", which is command\_not\_accepted, and EIR:CNA is set to "1". An interrupt is generated if the interrupt would be valid.

### 3.4.4. FREEZE Command

The setting where SUCC1:CMD[3:0] is "0111" (CHI command FREEZE) enables a transition to the HALT state when the host detects a severe error state. This command triggers a transition to the HALT state regardless of the current POC state.

The transition to the HALT state can be read from CCSV:PSL[5:0].

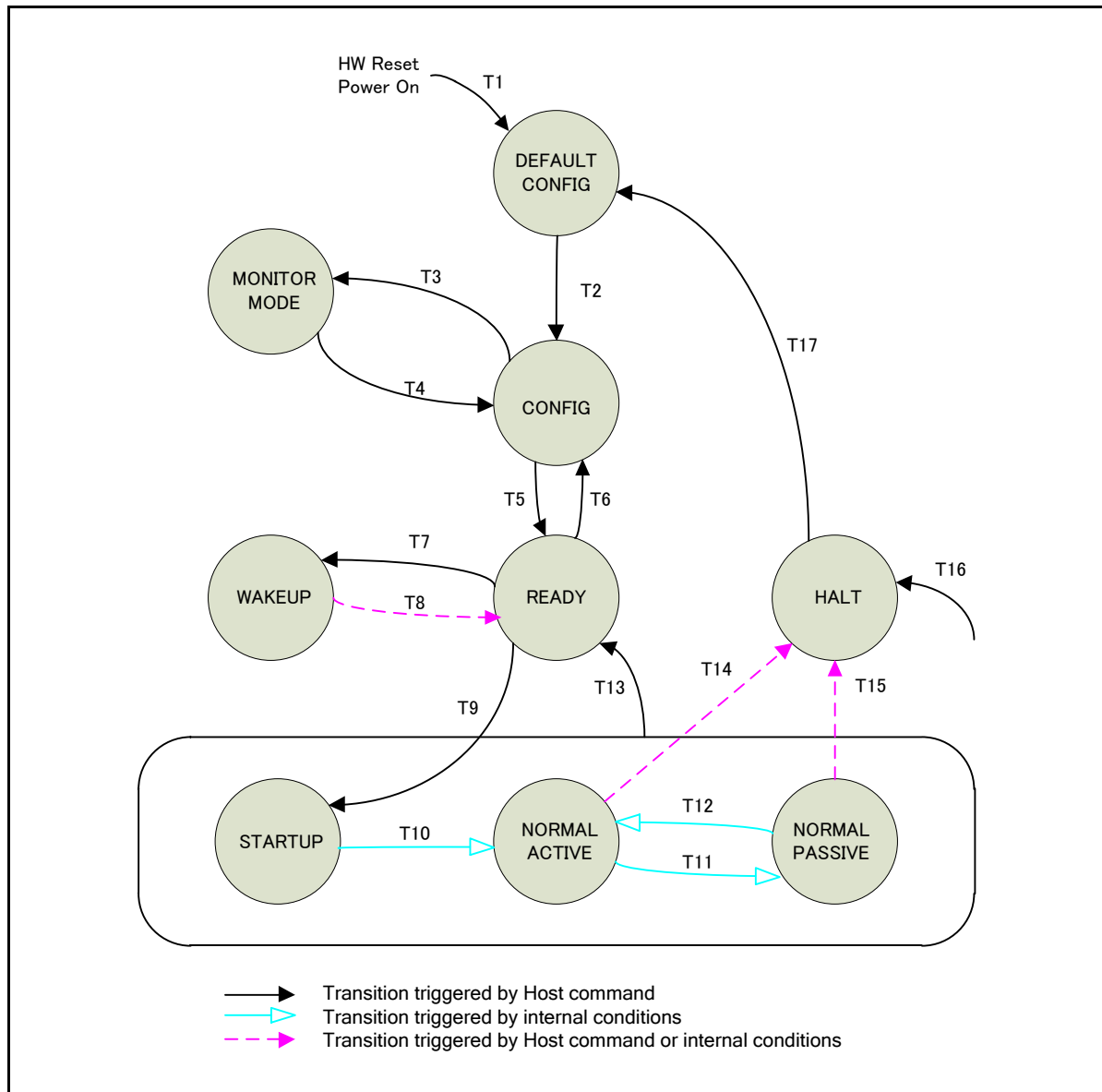
**Note:**

- There is no transition to the HALT state when SUCC1:HCSE has not been set.

## 3.5. Communication Controller States

### 3.5.1. Communication Controller State

Figure 3-3 State Diagram of the Communication Controller (CC)



State transitions are controlled by software resets by the host, external pins/RXDA/RXDB, the POC state machine, and the CHI command vector, SUCC1:CMD[3:0].

If the SUCC1:CMD[3:0] setting is "0111" (CHI command FREEZE), all states transition to the HALT state.

**Table 3-2 FlexRay Controller State Transitions**

Tn	State	From	To
1	- Hard reset	All states	DEFAULT_CONFIG
2	- CONFIG command Setting of SUCC1:CMD[3:0]="0001" (CHI command CONFIG)	DEFAULT_CONFIG	CONFIG
3	- Unlock sequence (by MONITOR_MODE command) Setting of SUCC1:CMD[3:0]="1011" (CHI command MONITOR_MODE)	CONFIG	MONITOR_MODE
4	- CONFIG command Setting of SUCC1:CMD[3:0]="0001" (CHI command CONFIG)	MONITOR_MODE	CONFIG
5	- Unlock sequence (by READY command) Setting of SUCC1:CMD[3:0]="0010" (CHI command READY)	CONFIG	READY
6	- CONFIG command Setting of SUCC1:CMD[3:0]="0001" (CHI command CONFIG)	READY	CONFIG
7	- WAKEUP command Setting of SUCC1:CMD[3:0]="0011" (CHI command WAKEUP)	READY	WAKEUP
8	- Wakeup pattern transmitted - WUP received - Frame header received - Wakeup collision occurred - READY command Setting of SUCC1:CMD[3:0]="0010" (CHI command READY)	WAKEUP	READY
9	- RUN command Setting of SUCC1:CMD[3:0]="0100" (CHI command RUN)	READY	STARTUP
10	- Startup successful	STARTUP	NORMAL_ACTIVE
11	- Clock correction failure counter reached set value of SUCC3:WCP[3:0]	NORMAL_ACTIVE	NORMAL_PASSIVE
12	- Number of valid cycle pairs in terms of clock correction time reached set value of UCC1:PTA[4:0]	NORMAL_PASSIVE	NORMAL_ACTIVE
13	- READY command Setting of SUCC1:CMD[3:0]="0010" (CHI command READY)	STARTUP, NORMAL_ACTIVE, NORMAL_PASSIVE	READY
14	- SUCC1:HCSE set to "1" after clock correction failure counter reached set value of SUCC3:WCF[3:0] - HALT command Setting of SUCC1:CMD[3:0]="0110" (command HALT)	NORMAL_ACTIVE	HALT
15	- SUCC1:HCSE set to "1" after clock correction failure counter reached set value of SUCC3:WCF[3:0] - HALT command Setting of SUCC1:CMD[3:0]="0110" (command HALT)	NORMAL_PASSIVE	HALT
16	- FREEZE command Setting of SUCC1:CMD[3:0]="0111" (CHI command FREEZE)	All states	HALT
17	- CONFIG command Setting of SUCC1:CMD[3:0]="0001" (CHI command CONFIG)	HALT	DEFAULT_CONFIG



### 3.5.2. DEFAULT\_CONFIG State

The FlexRay controller is stopped in the DEFAULT\_CONFIG state. All configuration registers are accessible, but the RXDA/RXDB/TXDA/TXDB/TXEN pins are inactive.

There is a transition to this state in the following cases:

- When a hard reset is performed
- When there is a transition from the HALT state

To transition from the DEFAULT\_CONFIG state to the CONFIG state, write "0001" in SUCC1:CMD.

### 3.5.3. CONFIG State

The FlexRay controller is stopped in the CONFIG state. All configuration registers are accessible, but the RXDA/RXDB/TXDA/TXDB/TXEN pins are inactive. This state is used to initialize the FlexRay controller settings.

There is a transition to this state in the following cases:

- When there is a transition from the DEFAULT\_CONFIG state
- When there is a transition from the MONITOR\_MODE or READY state

If there is a transition to this state via the HALT state and DEFAULT\_CONFIG state, status information and settings can be analyzed. Confirm that no settings are missing before a transition from the CONFIG state.

A transition from the CONFIG state requires the execution of the unlock sequence described in Section "4.2.1. Lock Register". After the unlocking of the CONFIG state, writing to SUCC1:CMD is necessary for a transition to the next state.

**Note:**

- *The transition from the CONFIG state to the READY state does not affect the message buffer status registers (MHDS, TXRQ1/2/3/4, NDAT1/2/3/4, and MBSC1/2/3/4) and status data stored in the message RAM.*

### 3.5.4. MONITOR\_MODE

After the unlocking of the CONFIG state and writing of SUCC1:CMD to "1011", there is a transition to MONITOR\_MODE. FlexRay frames and wakeup patterns can be received in this mode. The integrity of the time of the received frames is not checked. The consistency of temporarily received frames is not checked. Consequently, cycle counter filtering is not supported. This mode can be used for debugging purposes, such as to prepare in case FlexRay network startup fails. After the writing of SUCC1:CMD to "0001", there is a transition to the CONFIG state.

In MONITOR\_MODE, the first operation is invalid. This means that a receiving message buffer may simply be created to receive on 1 channel. A received frame is stored in the message buffer of the frame ID, and it receives the channel. Invalid frames are handled in the same way as data frames. The MBS:VFRA, MBS:VFRB, MBS:MLST, MBS:RCIS, MBS:SFIS, MBS:SYNS, MBS:NFIS, MBS:PPIS, and MBS:RESS state bit values become valid only after a frame is received.

The reception FIFO cannot be used in MONITOR\_MODE.

In MONITOR\_MODE, the CAS symbol and MTS symbol cannot be distinguished from each other. If either of these symbols is detected on the channel, either SIR:MTSA or SIR:MTSB is set. SIR:CAS does not function in MONITOR\_MODE.

### 3.5.5. READY State

After the unlocking of the CONFIG state and writing of SUCC1:CMD to "0010", there is a transition to the READY state. The following is possible: cluster wakeup through a transition from this state to the WAKEUP state, a cold start through a transition from this state to the STARTUP state, or integration into the running cluster from this state.

Each of the following states transitions to the READY state through the writing of SUCC1:CMD to "0010" (CHI command READY):

- CONFIG state
- WAKEUP state
- STARTUP state
- NORMAL\_ACTIVE state
- NORMAL\_PASSIVE state

The READY state transitions to the following respective states as a result of writing:

- To the CONFIG state through the writing of SUCC1:CMD to "0001" (CHI command CONFIG)
- To the WAKEUP state through the writing of SUCC1:CMD to "0011" (CHI command WAKEUP)
- To the STARTUP state through the writing of SUCC1:CMD to "0100" (CHI command RUN)

**Note:**

- *The transition of POC from the READY state to the STARTUP state does not affect status bits (MHDS[14:0]), registers (TXRQ1/2/3/4), and status data stored in the message RAM.*





### 3.5.6. WAKEUP State

This section describes the wakeup settings for the FlexRay controller.

The READY state transitions to the WAKEUP state under the following conditions.

- The WAKEUP state transitions to the READY state through the writing of SUCC1:CMD[3:0] to "0011" (CHI command WAKEUP).
- After transmission of a normal wakeup pattern is completed
- After a WUP is received
- After a WUP collision is detected
- After a frame header is received
- When SUCC1:CMD[3:0] of "0010" is written (CHI command READY)

To execute WAKEUP on a cluster, WAKEUP must be executed before communication startup. When receiving a wakeup pattern on a channel, the bus driver wakes up the other components of the node. At least 1 node in a cluster generates the wakeup pattern.

The host controls the whole wakeup procedure. First, it references the cluster states from the bus driver and FlexRay controller, configures the FlexRay controller (and the bus guardian if available), and wakes up the cluster. This configuring of the FlexRay controller enables transmission of a separate special wakeup pattern to each available channel. The FlexRay controller should recognize the wakeup pattern only during the WAKEUP state.

WAKEUP can be executed on only 1 channel at a time. The wakeup channel must be configured with the writing of SUCC1:WUCS during the CONFIG state. Although the communication in process on this channel can be guaranteed to be unaffected, it is impossible to confirm normal operation on all nodes from the startup phase. Therefore, wakeup pattern transmission cannot be guaranteed to wake up all nodes connected to the wakeup channel. Also, a wakeup pattern can be transmitted to only 1 channel in a two-channel system. A cold start node, which requires system startup, wakes up the remaining channels before starting communication startup.

This wakeup procedure has a state where 1 node transmits a wakeup pattern even when several nodes connected to a single channel are transmitting wakeup patterns simultaneously. The wakeup pattern can also wake up other nodes even after a collision by 2 nodes transmitting wakeup patterns simultaneously since recovery from the signal collision would be quick.

After a transition to the READY state following WAKEUP, the FlexRay controller reports the change in the WAKEUP status by setting the SIR:WST flag to "1". The WAKEUP status vector can be read from CCSV:WSV[2:0]. If the received wakeup pattern is valid, either the SIR:WUPA or SIR:WUPB flag is set to "1".

Figure 3-4 POC Configuration in WAKEUP State

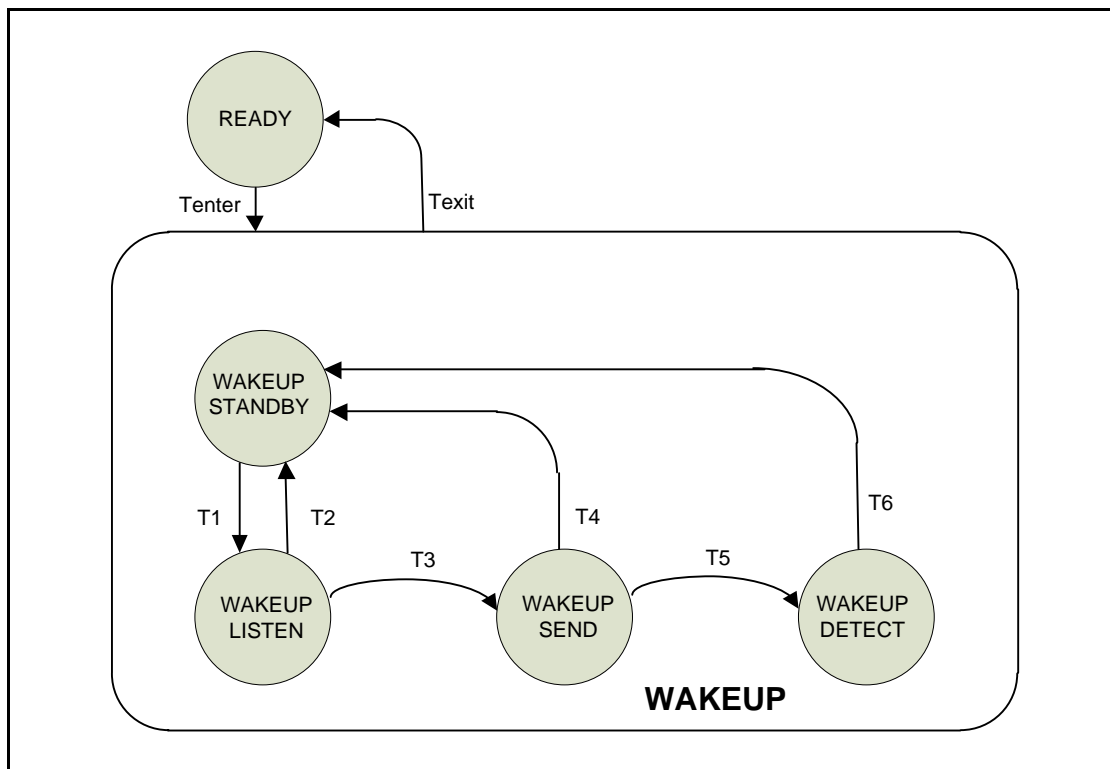


Table 3-3 WAKEUP State Transitions

Tn	State	From	To
Enter	<ul style="list-style-type: none"> <li>WAKEUP command</li> <li>Setting of SUCC1:CMD[3:0]="0011" (CHI command WAKEUP)</li> </ul>	READY	WAKEUP
1	<ul style="list-style-type: none"> <li>WAKEUP command</li> <li>Setting of SUCC1:CMD[3:0]="0011" (CHI command WAKEUP)</li> </ul>	WAKEUP_STANDBY	WAKEUP_LISTEN
2	<ul style="list-style-type: none"> <li>A WUP is received on the wakeup channel selected by SUCC1:WUCS.</li> <li>A frame header is received on either of the valid channels.</li> </ul>	WAKEUP_LISTEN	WAKEUP_STANDBY
3	<ul style="list-style-type: none"> <li>A timer event has occurred.</li> </ul>	WAKEUP_LISTEN	WAKEUP_SEND
4	<ul style="list-style-type: none"> <li>Transmission of a wakeup pattern has completed normally.</li> </ul>	WAKEUP_SEND	WAKEUP_STANDBY
5	<ul style="list-style-type: none"> <li>A collision is detected.</li> </ul>	WAKEUP_SEND	WAKEUP_DETECT
6	<ul style="list-style-type: none"> <li>The wakeup timer has timed out.</li> <li>A WUP is received on the wakeup channel selected by SUCC1:WUCS.</li> <li>A frame header is received on either of the valid channels.</li> </ul>	WAKEUP_DETECT	WAKEUP_STANDBY
Exit	<ul style="list-style-type: none"> <li>Wakeup has completed (after T2, T4, and T6).</li> <li>READY command</li> <li>Setting of SUCC1:CMD[3:0]="0010" (CHI command READY)</li> <li>(There is a simultaneous reset to the WAKEUP_STANDBY state by this CHI command.)</li> </ul>	WAKEUP	READY



The wakeup timer and wakeup noise timer control the WAKEUP\_LISTEN state. These 2 timers are controlled with parameters: the listen timeout value, SUCC2:LT[20:0], and the listen timeout noise value, SUCC2:LTN[3:0]. Listen Timeout is effective for quick cluster wakeup in a noise-free environment. Listen Timeout Noise is effective for wakeup in an environment with a high level of noise interference.

In the WAKEUP\_SEND state, wakeup patterns are transmitted on set channels to check for their collisions. After a transition from the WAKEUP state to the READY state, CMD[3:0] must be "0100" (CHI command RUN) for a transition to the STARTUP state.

In the WAKEUP\_DETECT state, the cause of a wakeup collision detected in the WAKEUP\_SEND state can be identified. The identification is stopped when the Listen Timeout setting in SUCC2:LT[20:0] is exceeded. There is a direct transition to the READY state upon either detection of a wakeup pattern from another node or reception of a frame header. If neither that detection nor reception happens, there is a transition from the WAKEUP\_DETECT state after Listen Timeout elapses. In such cases, the cause of the wakeup collision is unknown.

The host must recognize possible failures during wakeup and take the necessary actions. If a node causes a wakeup during its startup, we recommend that the start must be delayed for the minimum time until another cold start node wakes up and is initialized.

The FlexRay Protocol Specifications V2.1 recommends waking up 2 channels using 2 different FlexRay controllers.

### (1) Host Operations

The host must adjust the wakeup of 2 channels and determine whether to wake up a specific channel. The host starts transmission of wakeup patterns. The bus driver at the opposite end detects the wakeup patterns and notifies the local host.

The host controls the following wakeup procedure (the wakeup procedure for 1 channel).

- Configure the FlexRay controller in the CONFIG state.
  - Select the wakeup channel that is set by the SUCC1:WUCS bit.
- Check the bus driver to see whether a WUP was received.
- Start the bus driver on the selected wakeup channel.
- Write SUCC1:CMD[3:0] of "0010" for a transition to the READY state.
- Write SUCC1:CMD[3:0] of "0011" to start waking up the set channel.
  - The FlexRay controller transitions to the WAKEUP state.
  - After completing a wakeup, the FlexRay controller transitions to the READY state and displays the wakeup status (CCSV:POCS[5:0]).
- Wait for the previously determined duration, until another node can be woken up and configured.
- Perform the following procedure for a cold start node.
  - Wait until the other channel becomes a WUP, in a 2-channel cluster configuration.
  - Write SUCC1:CMD[3:0] of "1001" (CHI command ALLOW\_COLDSTART) to reset the cold start inhibit flag, CCSV:CSI.
- Write SUCC1:CMD[3:0] of "0100" (CHI command RUN) for a transition to the STARTUP state.

The bus driver triggers the following wakeup procedure.

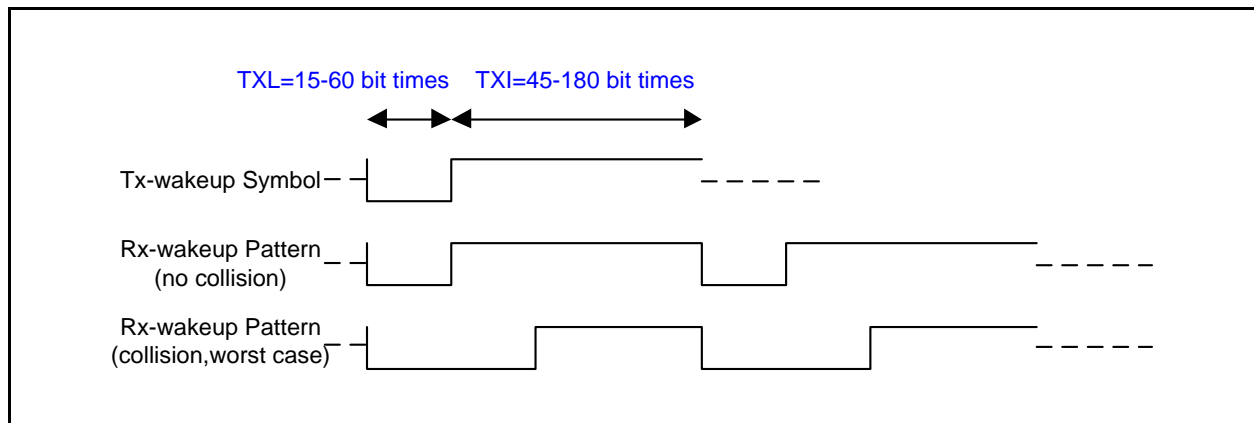
- The bus driver identifies the wakeup.
- The bus driver notifies the host of the wakeup event.
- The host configures the FlexRay controller.
- The host may execute the following as necessary:
  - Wakeup command for the second channel
  - Wait of the duration determined beforehand so that another node can be woken up and configured
- Write SUCC1:CMD[3:0] of "0100" (CHI command RUN) for a transition to the STARTUP state.

## (2) Wakeup Pattern (WUP)

A wakeup pattern (WUP) consists of at least 2 wakeup symbols (WUS). The wakeup symbol and wakeup pattern are set by the PRTC1 register and PRTC2 register, respectively.

- A single-channel wakeup and wakeup symbol cannot be transmitted to both channels at the same time.
- Wakeup symbol recovery from a signal collision is quick in an environment where a node transmits at least 2 wakeup patterns.  
(2 overlapping wakeup symbols are always identifiable.)
- The wakeup symbol must be identical in all nodes of the cluster.
- The Low time of the wakeup symbol is set by PRTC2:TXL[5:0].
- The wakeup symbol idle time, which is used to listen to activity on the bus, is set by PRTC2:TXI[7:0].
- The wakeup pattern consists of at least 2 transmission wakeup symbols necessary for wakeup.
- The number of repetitions (between 2 and 63 repetitions) can be set by PRTC1:RWP[5:0].
- The length of the wakeup symbol reception window is set by PRTC1:RXW[8:0].
- The Low time of wakeup reception is set by PRTC2:RXL[5:0].
- The wakeup reception idle phase time is set by PRTC2:RXI[5:0].

Figure 3-5 Wakeup Pattern Timing





### 3.5.7. STARTUP State

The initial confirmation for cold starting a node should be to confirm that both connected channels have been woken up in the STARTUP state.

The time required for completing wakeup and configuration of all nodes and clusters cannot be estimated. The time required for all the nodes and stars to complete wake up and setting cannot be assumed. Since at least two nodes are required for the startup of cluster communications, it is recommended to postpone the startup of the node on which a wake up is to be generated for the duration of the minimum time required for another cold started node to complete the wake up, initial setting, and startup.

The time delay due to the completion of wakeup and configuration of all nodes and clusters is approximately several 100 ms as a rough standard. (However, this delay depends on the hardware used.)

Startup is executed on all channels simultaneously. Nodes transmit the startup frames only during startup.

Procedures for distributed startup that is robust to errors have been prepared in advance for initialization synchronization on all nodes. Generally, nodes transition to the NORMAL\_ACTIVE state through the following procedures. (See Figure 3-6)

- Cold start procedure to start schedule synchronization (the leading cold start node)
- Cold start procedure for the participation of another cold start node (the following cold start node)
- Integration procedure for integration into the existing communication schedule (all other nodes)

An attempt at a cold start starts with transmission of a collision avoidance symbol (CAS). Only the cold start node that transmitted the CAS transmits frames during the first 4 cycles after the CAS. Afterward, another cold start node participates, and then all other nodes participate in the cluster.

The cold start node transmits synchronization frames to the key slot by setting "1" in SUCC1:TXST and SUCC1:TXSY. Message buffer 0 contains the key slot ID that defines the slot number to which the startup frame is transmitted. The startup frame indicator is set to "1" in the frame header of the startup frame.

After the transmission of the startup frame, the transmission request flag TXRQ1:TXR0 of message buffer 0 is reset to "0". To transmit data frames from message buffer 0, "1" must be set in TXRQ1:TXR0 via the IBCR register after a transition to the NORMAL\_ACTIVE state. Without that setting being made, a null frame will be transmitted to the slot corresponding to the frame ID stored in message buffer 0.

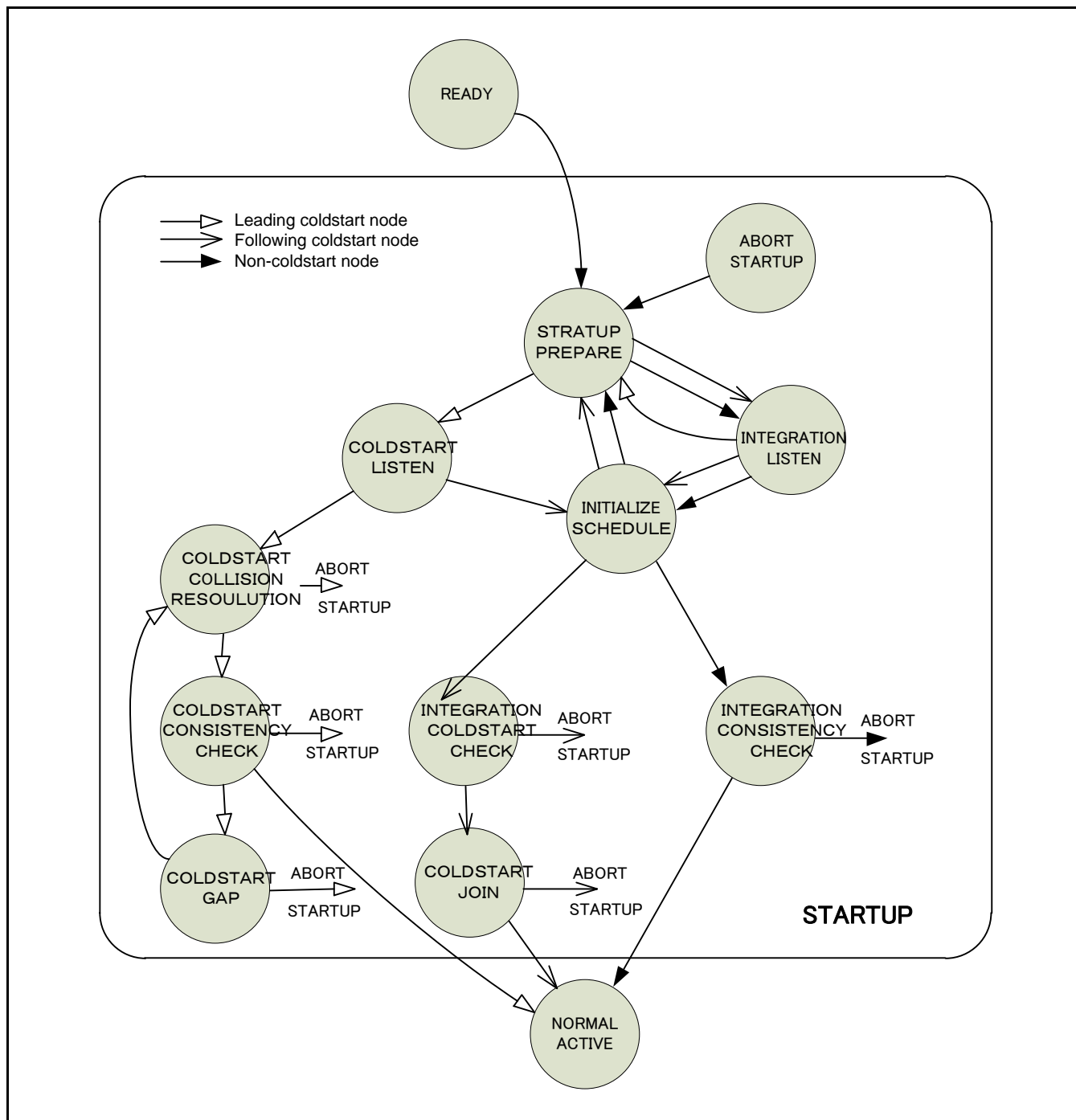
A cluster consisting of 3 or more nodes must be configured such that at least 3 nodes are cold start nodes. A cluster consisting of 2 nodes must have both nodes configured as cold start nodes. At least 2 fault-free cold start nodes are required for startup of the cluster.

Each startup frame should also be a synchronization frame. Therefore, all cold start nodes are also synchronization nodes. The number of cold start attempts is set by SUCC1:CSA[4:0].

To integrate non-cold start nodes into a cluster, at least 2 startup frames from other nodes are required. Integration of the non-cold start nodes may begin before startup of the cold start nodes is completed. However, startup of those non-cold start nodes is never completed until startup of at least 2 cold start nodes has completed.

If both a non-cold start node and a cold start node obtain TDMA (time division multiple access) schedule information and then receive a synchronization frame, they immediately start passive integration through the integration procedure. During the integration, the nodes adjust their own clock to the global clock (rate and offset) and their own cycle time to the network global cycle. Afterward, these settings are checked for consistency on all the available network nodes. Nodes can actively participate in communications only when they have passed these checks.

Figure 3-6 State Diagram of the Time-triggered Startup



### (1) Cold Start Inhibit Mode

Nodes cannot initialize cluster communication in cold start inhibit mode (CCSV:CSI is "1"). This means that startup using the cold start procedure is prohibited from beginning. Such nodes can either be integrated into the running cluster or transmit startup frames after another cold start node has started initializing cluster communication.

The cold start inhibit bit, CCSV:CSI, can be set while POC is in the READY state. Clear this bit by setting SUCC1:CMD[3:0] to "1001" (CHI command ALLOW\_COLDSTART).



## (2) Startup Timeout

The FlexRay controller provides 2 different  $\mu$ T timers that support 2 timeout values (startup timeout and startup noise timeout). These 2 timers start at a transition to the COLDSTART\_LISTEN state. When either of these 2 timers ends, the node terminates the detection phase of the other node (a transition from the COLDSTART\_LISTEN state to another state) in order to start communication.

### Note:

- The startup timer and startup noise timer are the same as the wakeup timer and wakeup noise timer, respectively, and they use SUCC2:LT[20:0] and SUCC2:LTN[3:0], respectively.

### a) Startup Timeout

The startup timeout limits the listening time used by a node in order to determine whether communication has been established between other nodes, or whether at least 1 cold start node is requesting integration with other nodes. The startup timer is set by SUCC2:LT[20:0] (pdListenTimeout).

The startup timer is pdListenTimeout, which is SUCC2:LT[20:0].

The startup timer is restarted under the following conditions.

- There is a transition to the COLDSTART\_LISTEN state.
- Both channels reach the idle state in the COLDSTART\_LISTEN state.

The startup timer is stopped under the following conditions.

- Communication is detected on 1 channel among the set channels while in the COLDSTART\_LISTEN state.
- There is a transition from the COLDSTART\_LISTEN state to another state.

Once the startup timer passes the restricted time, neither a timer overflow nor a periodic restart occurs. The timer status is retained for future processes.

### b) Startup Noise Timeout

The startup timer and startup noise timer start at the time of a transition from the STARTUP\_PREPARE state to the COLDSTART\_LISTEN state. The startup noise timeout is used to improve the reliability of the startup procedure in an environment with noise.

The startup noise timeout is determined by SUCC2:LTN[3:0].

The startup noise timer is:

$$\text{pdListenTimeout} \times \text{gListenNoise} = \text{SUCC2:LT}[20:0] \times (\text{SUCC2:LTN}[3:0] + 1)$$

The startup noise timeout is calculated as follows:

$$\text{SUCC2:LT}[20:0] + (\text{SUCC2:LTN}[3:0] \times \text{SUCC2:LT}[20:0])$$

The startup noise timer is restarted under the following conditions.

- There is a transition to the COLDSTART\_LISTEN state.
- A header or CAS symbol that is decoded normally in the COLDSTART\_LISTEN state is received.

The startup noise timer stops at a transition from the COLDSTART\_LISTEN state to another state.

Once the startup timer passes the restricted time, neither a timer overflow nor a periodic restart occurs. The timer status is retained for future processes. If communication is detected in any channel, the startup noise timer does not restart. In other words, this timeout is defined as a worst-case solution that has been prepared to guarantee that a node can start the communication cluster even in an environment with noise.

### **(3) Startup Process of Leading Cold Start Node (Starting Cold Start)**

A cold start node in the COLDSTART\_LISTEN state monitors the states of the connected channels.

If no communication is detected, the node transitions to the COLDSTART\_COLLISION\_RESOLUTION state and starts a cold start. The CAS symbol is first transmitted in the first normal cycle. This cycle is called cycle 0.

The node transmits its startup frame from cycle 0. Before each of the cold start nodes becomes available for a cold start, some nodes may transmit CAS symbols simultaneously for startup to begin following the cold start procedure. This state is resolved in the first 4 cycles after CAS transmission.

If any of the nodes that has started a cold start receives a CAS symbol or a frame header within these 4 cycles, the node transitions again to the COLDSTART\_LISTEN state. As a result, only 1 node within the cluster continues the cold start procedure. The other cold start nodes start transmitting their own startup frames in cycle 4.

A node that starts a cold start after 4 cycles in the COLDSTART\_COLLISION\_RESOLUTION state transitions to the COLDSTART\_CONSISTENCY\_CHECK state. This node collects all startup frames from cycle 4 and cycle 5, and corrects the clock. After the clock is corrected without an error and at least 1 valid startup frame pair is received, it transitions from the COLDSTART\_CONSISTENCY\_CHECK state to the NORMAL\_ACTIVE state.

The number of cold start attempts is set by SUCC1:CSA[4:0]. The remaining number of cold start attempts can be read from CCSV:RCA[4:0]. The remaining number of cold start attempts is decremented each time that an attempt is made. A transition to the COLDSTART\_LISTEN state is possible when the remaining number of attempts is greater than 1. A transition to the COLDSTART\_COLLISION\_RESOLUTION state is possible when the remaining number of attempts is greater than 0. Integration into a cluster is possible when the remaining number of cold start attempts is 1, but a cold start is prohibited.

### **(4) Startup Process of Following Cold Start Node (in Response to Leading Cold Start Node)**

When transitioning to the COLDSTART\_LISTEN state, a cold start node attempts to receive a valid pair of startup frames to obtain the cycle schedule and clock correction from the leading cold state node.

Upon receiving the first valid startup frame, it immediately transitions to the INITIALIZE\_SCHEDULE state. Upon receiving the second valid startup frame and obtaining the cycle schedule, it transitions to the INTEGRATION\_COLDSTART\_CHECK state.

In the INTEGRATION\_COLDSTART\_CHECK state, the following is guaranteed: clock correction can be done correctly, and the leading cold start node is available for use. (The following cold start node initializes its schedule according to this leading cold start node.)

The following cold start node collects all synchronization frames and corrects the clock in the following cycle pair. The node transitions to the COLDSTART\_JOIN state when the clock correction does not show an error and the node continues to receive sufficient frames from the same node.

The following cold start node starts transmitting its own startup frames in the COLDSTART\_JOIN state and continues transmitting these frames in the next cycle. This enables the leading cold start node and participating nodes to check whether their cycle schedules are synchronized with each other. If an error is detected in clock correction, the participating nodes stop cluster integration. If the node in this state receives at least 1 valid startup frame in an even-numbered cycle and at least 1 valid pair of startup frames in all cycle pairs, the node transitions from the COLDSTART\_JOIN state to the NORMAL\_ACTIVE state. Therefore, the following cold start node transitions from the STARTUP state to the NORMAL\_ACTIVE state at least 1 cycle later than the leading cold start node.





### (5) Startup Process of Non-cold Start Node

A non-cold start node in the INTEGRATION\_LISTEN state monitors the states of the connected channels.

Upon receiving the first valid startup frame, it immediately transitions to the INITIALIZE\_SCHEDULE state.

Upon receiving the second valid startup frame and obtaining the cycle schedule, it transitions to the INTEGRATION\_CONSISTENCY\_CHECK state.

In the INTEGRATION\_CONSISTENCY\_CHECK state, the non-cold start node checks whether clock correction is working correctly and whether enough cold start nodes (at least 2) are transmitting startup frames synchronized to the cycle schedule. The integration is stopped when any error is detected while clock correction is in operation.

Be sure that this non-cold start node receives either 2 valid startup frames or a valid startup frame from other integrated nodes within the first even-numbered cycle in this state. Otherwise, the node aborts the integration.

Be sure that this non-cold start node receives either 2 valid pairs of startup frames or a valid pair of startup frames from other integrated nodes within the first cycle pair in this state. Otherwise, the node aborts the integration.

If 2 or more valid startup frames are not received within the even-numbered cycle after the first cycle pair, or if 2 or more valid pairs of startup frames are not received within 1 cycle pair, the startup is aborted.

For the node in this state, a transition from the STARTUP state to the NORMAL\_ACTIVE state requires that the node receive 2 valid pairs of startup frames each in 2 cycle pairs. Therefore, the node transitions from the STARTUP state to the NORMAL\_ACTIVE state at least 1 cycle pair after a node that has started a cold start, or at the end of an odd-numbered cycle.

## 3.5.8. NORMAL\_ACTIVE State

The startup phase of the entire cluster ends immediately after the transition of the node that transmitted the first CAS symbol and 1 additional node to the NORMAL\_ACTIVE state. The transmission timing of all transmission messages is scheduled in the NORMAL\_ACTIVE state. This includes all data frames in the same way as synchronization frames. Rate and offset measurement begins in all even-numbered cycles. (Even/Odd-numbered cycle pairs are required.)

The FlexRay controller supports the normal communication functions in the NORMAL\_ACTIVE state.

- Transmission and reception on the FlexRay bus are performed according to settings.
- Clock synchronization is in operation.

The FlexRay controller transitions from the NORMAL\_ACTIVE state to the following states:

- To the HALT state after the end of the current cycle through the writing of SUCC1:CMD[3:0] to "0110" (CHI command HALT)
- Immediately to the HALT state through the writing of SUCC1:CMD[3:0] to "0111" (CHI command FREEZE)
- To the HALT state through an error state change from ACTIVE to COMM\_HALT
- To the NORMAL\_PASSIVE state through an error state change from ACTIVE to PASSIVE
- To the READY state through the writing of SUCC1:CMD[3:0] to "0010" (CHI command READY)

### 3.5.9. NORMAL\_PASSIVE State

The NORMAL\_ACTIVE state transitions to the NORMAL\_PASSIVE state when the error state changes from ACTIVE to PASSIVE.

Nodes can receive all frames in the NORMAL\_PASSIVE state. (Nodes are completely synchronized, and clock synchronization is possible.) However, nodes do not actively participate in communication, as compared with the NORMAL\_ACTIVE state. This means that neither symbols nor frames are transmitted.

The following operations are performed in the NORMAL\_PASSIVE state.

- Frames are received on the FlexRay bus.
- Neither frames nor symbols are transmitted to the FlexRay bus.
- Clock synchronization is in operation.

The FlexRay controller transitions from the NORMAL\_PASSIVE state to the following states:

- To the HALT state after the end of the current cycle through the writing of SUCC1:CMD[3:0] to "0110" (CHI command HALT)
- Immediately to the HALT state through the writing of SUCC1:CMD[3:0] to "0111" (CHI command FREEZE)
- To the HALT state through an error state change from PASSIVE to COMM\_HALT
- To NORMAL\_PASSIVE through an error state change from PASSIVE to ACTIVE (This error state change occurs when CCEV:PTAC[4:0] equals SUCC1:PTA[4:0]–1.)
- To the READY state through the writing of SUCC1:CMD[3:0] to "0010" (CHI command READY)

### 3.5.10. HALT State

All communications (transmission and reception) are halted in this state.

The FlexRay controller transitions to the HALT state in the following cases:

- Transition from the NORMAL\_ACTIVE or NORMAL\_PASSIVE state when SUCC1:CMD[3:0] of "0110" (CHI command HALT) is written
- Transition from all states when SUCC1:CMD[3:0] of "0111" (CHI command FREEZE) is written
- Transition from the NORMAL\_ACTIVE state when the clock correction fatal counter reaches the maximum HALT transition time without clock correction, WCF[3:0]. SUCC1:HCSE is set to "1".
- Transition from the NORMAL\_PASSIVE state when the clock correction fatal counter reaches the maximum HALT transition time without clock correction, WCF[3:0]. SUCC1:HCSE is set to "1".

The FlexRay controller transitions from this state to the DEFAULT\_CONFIG state in the following case:

- When SUCC1:CMD[3:0] of "0001" (CHI command CONFIG) is written

When SUCC1:CMD[3:0] of "0110" (CHI command HALT) is written, the FlexRay controller sets the CCSV:HRQ bit to "1" and transitions to the HALT state after the next cycle ends.

When SUCC1:CMD[3:0] of "0111" (CHI command FREEZE) is written, the controller immediately transitions to the HALT state and sets the CCSV:FSI bit to "1".

The status of the transition to the HALT state can be read from CCSV:PSL[5:0].



### 3.6. Network Management

Generated network management (NM) vectors can be read from the NMV1 to NMV3 registers. The FlexRay controller performs a bit OR operation on all NM vectors in all valid reception NM frames where the payload preamble indicator (PPI) is set. Only static frames can be configured as NM frames. Also, after each cycle is completed, NM vectors are updated.

An NM vector length of between 0 and 12 bytes can be set by NEMC:NML[3:0]. Be sure that the set NM vector length is the same on all nodes in the cluster.

The PPIT bit in the header section of each transmission buffer must be set through WRHS1:PPIT to configure the frame transmission buffer that is set by the PPI bit. Also, NM information must be written in the data section of each transmission buffer.

A mechanism for evaluating the NM vectors must be implemented in an application.

**Note:**

- *If the message buffer is configured to transmit/receive network management frames, the set payload length in header 2 of the message buffer must be equal to or greater than the set NM vector length in NEMC:NML[3:0].  
The cycle count does not increase when the HALT state is passed, so the NM vectors are not updated. In this case, the NMV1 to NMV3 registers retain the values from the previous cycle.*

### 3.7. Filtering and Masking

Filtering is done through a comparison of the settings of the current slot, cycle counter value, channel ID (channel A and channel B), and message buffer. If this comparison finds an information match, the message buffer is updated or transmitted.

Filtering is applied to the following:

- Slot counter
- Cycle counter
- Channel ID

The following filter combinations can be used for filtering at the transmission or reception time:

- Slot counter + channel ID
- Slot counter + channel ID + cycle counter

To store received messages in the message buffer, all the set filters must be matched against information on the received messages.

**Note:**

- *The FIFO rejection filter and FIFO rejection filter mask configure the FIFO acceptance filter.*

A message is transmitted to the time slot corresponding to the set frame ID in the set channel. If cycle counter filtering is enabled, the set cycle filter value must also match.

#### 3.7.1. Slot Counter Filtering

All the transmission and reception buffers contain a frame ID in the header section. The frame ID is compared with the current slot counter value to assign it to the slot corresponding to the transmission or reception buffer.

If multiple buffers are configured with the same frame ID and same channel ID, and if the buffers have cycle counter filter values corresponding to the same slot, the message buffer with the smallest buffer number is used.

#### 3.7.2. Cycle Counter Filtering

Cycle counter filtering is based on the concept of a cycle set. A match to a filter is detected when 1 element of the cycle set matches. The cycle set is defined by the cycle code field in header section 1 in each message buffer.

When configuring message buffer 0 to store startup/synchronization frames or single-slot frames by setting the respective SUCC1:TXST, SUCC1:TXSY, and SUCC1:TSM bits, disable cycle counter filtering in message buffer 0.

**Note:**

- *Static time slot sharing by cycle counter filtering between different nodes in the FlexRay network is not permitted.*



Table 3-4 describes settings of the number of cycles belonging to a cycle set.

**Table 3-4 Cycle Set Definitions**

Cycle Code	Match with Cycle Counter Values
0b000000x	all Cycles
0b000001c	every second Cycle at $(\text{Cycle Count}) \bmod 2 = c$
0b00001cc	every fourth Cycle at $(\text{Cycle Count}) \bmod 4 = cc$
0b0001ccc	every eighth Cycle at $(\text{Cycle Count}) \bmod 8 = ccc$
0b001cccc	every sixteenth Cycle at $(\text{Cycle Count}) \bmod 16 = cccc$
0b01ccccc	every thirty-second Cycle at $(\text{Cycle Count}) \bmod 32 = ccccc$
0b1cccccc	every sixty-fourth Cycle at $(\text{Cycle Count}) \bmod 64 = ccccccc$

Table 3-5 below shows some examples of valid cycle sets used for cycle counter filtering.

**Table 3-5 Examples of Valid Cycle Sets**

Cycle Code	Match with Cycle Counter Values
0b0000011	1-3-5-7-...-63
0b0000100	0-4-8-12-...-60
0b0001110	6-14-22-30-...-62
0b0011000	8-24-40-56
0b0100011	3-35
0b1001001	9

A received message is stored only if the cycle counter value during the cycle in which the message was received matches an element of the cycle set of the reception buffer. Other filter criteria must also be satisfied.

The contents of the transmission buffer are transmitted to the set channel when an element of the cycle set matches the current cycle counter value. Other filter criteria must also be satisfied.

### 3.7.3. Channel ID Filtering

The header section of each message buffer in the message RAM has 2-bit channel filtering fields (CHA and CHB). They act as filters for the reception buffer and as control fields for the transmission buffer. (See Table 3-6 below)

Table 3-6 Channel Filtering Settings

CHA	CHB	Transmission Buffer (Transmission Frame)	Reception Buffer (Storage of Received Frame)
1	1	on both channels (static segment only)	received on channel A or B (store first semantically valid frame, static segment only)
1	0	on channel A	received on channel A
0	1	on channel B	received on channel B
0	0	no transmission	ignore frame

The contents of the transmission buffer are transmitted to the channel specified by the channel filtering field when slot counter filtering and cycle counter filtering criteria are satisfied. However, setup (CHA and CHB settings) for transmission to both channels is permitted only in the static segment.

If the channel specified by the channel filtering field receives valid received frames when slot counter filtering and cycle counter filtering criteria are satisfied, the frames are stored. However, setup (CHA and CHB settings) for frame reception by both channels is permitted only in the static segment.

**Note:**

- If the message buffer is configured for dynamic segments and both channel filtering field bits (CHA and CHB) are set to "1", frames are not transmitted, and received frames are ignored. (The function is the same as when CHA and CHB are "0".)

### 3.7.4. FIFO Filtering

FIFO filtering has 1 rejection filter and 1 rejection filter mask prepared for it. The rejection filter consists of the channel filter FRF:CH[1:0], frame ID filter FRF:FID[10:0], and cycle counter filter FRF:CYF[6:0]. The FRF register and FRFM register can be configured only in the DEFAULT\_CONFIG or CONFIG state. The filter settings in the header section of a message buffer belonging to a FIFO group are ignored.

FRF:CYF[6:0] is a 7-bit cycle counter filter that specifies a cycle set and determines the communication cycle to which to apply the frame ID filter and channel filter. The cycle set specified by this register does not apply the frame ID filter and channel filter to all cycles. During those non-application cycles, no frames are received.

If the filtering by the set rejection filter and rejection filter mask does not reject the channel ID, frame ID, and cycle counter of a valid received frame that has no match with the dedicated reception buffer, the frame is stored in the FIFO.



## 3.8. Transmission Procedure

### 3.8.1. Static Segment

If several messages are suspended from transmission in the static segment, the message with the frame ID corresponding to the next transmission slot is selected as the next message to transmit.

The data section of the transmission buffer assigned to the static segment can be updated by the end of the previous time slot. This means that message transmission from the input buffer must be started through writing to the input buffer command request register at the end of the time slot.

### 3.8.2. Dynamic Segment

If several messages are suspended from transmission in the dynamic segment, the message with the highest priority (the smallest frame ID) is selected as the next message to transmit. Also, different slot counter columns may be generated for channel A and channel B in the dynamic segment (for simultaneous transmission with different frame IDs from both channels).

The data section of the transmission buffer assigned to the dynamic segment can be updated by the end of the previous slot. This means that message transmission from the input buffer must be started through writing to the input buffer command request register at the end of the time slot.

The transmission completion minislot value, MHDC:SLT[12:0], defines the maximum minislot value that can be transmitted before frame transmission in the dynamic segment of the current cycle is prohibited.

### 3.8.3. Transmission Buffer

Each message buffer can be used as a transmission buffer when the CFG bit in the header section of the message buffer is set to "1" through WRHS1.

A transmission buffer can be assigned to a FlexRay controller channel in the following ways:

- Static segment: Channel A or channel B  
Channel A and channel B
- Dynamic segment: Channel A or channel B

Message buffer 0 is used as a dedicated buffer for storing startup frames and synchronization frames or as a dedicated buffer for the specified single-slot frames, as set by SUCC1:TXST, SUCC1:TXSY, and SUCC1:TSM, respectively. In these cases, message buffer 0 can be reconfigured only in the DEFAULT\_CONFIG or CONFIG state. This ensures that 1 startup frame/synchronization frame per communication cycle is mostly transmitted, no matter which node transmits them. No startup frame/synchronization frame can be transmitted from other message buffers.

Except buffer 0, all message buffers configured for static segment or dynamic segment transmission can be reconfigured during execution according to the MRC:SEC[1:0] setting. (See Section "3.11.1. Message Buffer Reconfiguration") However, the data pointer in the header partition references the data partition in the message RAM. Therefore, if the payload length and data pointer in the header section of a message buffer are set again, the message buffer structure may be incorrect.

If a message buffer is reconfigured during execution (the header section is updated), the message buffer may not be transmitted in each communication cycle.

The header CRC must be provided to all transmission buffers because the FlexRay controller has no function for calculating the header CRC. If network management is required, the host must set the PPIT bit in the header section of each message buffer to "1" and write network management information to the data section of the message buffer. (See Section "3.6. Network Management").

The payload length field stores a payload length in 2-byte units. If the payload length of the set static transmission buffer is shorter than the setting in MHDC:SFDL[6:0], padding bytes are inserted to guarantee the payload length of the static frame. The padding byte is indicated by "0".

**Note:**

- In cases with an odd-numbered payload length ( $PLC=1, 3, 5, \dots$ ), a 16-bit zero must be written at the end of the message buffers.

The transmission mode can be set on each transmission buffer by the transmission mode flag, TXM. If this bit is set to "1", the transmission message is transmitted in single-shot mode. If the bit is set to "0", it is transmitted in continuous mode.

In single-shot mode, the TXR flag of each message buffer is cleared to "0" after transmission is completed. Then, the transmission buffer can be overwritten by the next message to be transmitted.

In continuous mode, the TXR flag of each message buffer is not cleared to "0" after transmission is completed. In this case, a frame is transmitted each time that the filter criteria match. Writing each message buffer number to the IBCR register while the IBCM:STXRH bit setting is "0" can clear the TXR flag to "0".

If multiple transmission buffers satisfy the filter criteria, the transmission buffer with the lowest buffer number is used for transmission in each slot.

### 3.8.4. Frame Transmission

The following procedure is required for preparing a message buffer for transmission.

- Configure the transmission buffer in the message RAM through WRHS1, WRHS2, and WRHS3.
- Write data to the data section of the transmission buffer through WRDSn.
- Write the target buffer number to the IBCR register to set the input buffer to the message RAM and transfer message data.
- If the IBCM register is configured for message transmission, the transmission request flag TXR of each message buffer is set to "1" as soon as the transfer from the input buffer is completed, and the message buffer waits for transmission.
- A check of each TXR bit (TXR is "0") in the TXRQ1/2/3/4 register (only in single-shot mode) can verify whether transmission of the message buffer has completed.

After the transmission is completed (in single-shot mode), each TXR flag of the TXRQ1/2/3/4 register is cleared to "0". Also, if the MBI bit in the header section of the message buffer is set to "1", SIR:TXI is set to "1". An interrupt is generated if the interrupt would be valid.

### 3.8.5. Null Frame Transmission

If the transmission request flag is not set to "1" in the static segment before the transmission time and none of the other transmission buffers satisfies the filter criteria, the FlexRay controller transmits a null frame with the null frame indicator set to "0" and payload data cleared to "0".

A null frame is transmitted in the following cases.

- The transmission request flag is not set (TXR is "0") on the message buffer that matches the filter criteria and has the smallest buffer number.
- All the transmission buffers have cycle counter filters, but none of them matches the current cycle. In this case, the message buffer status (MBS) is not updated.

No null frame is transmitted in the dynamic segment.





## 3.9. Reception Procedure

### 3.9.1. Reception Buffer

Each message buffer can be used as a dedicated reception buffer when the CFG bit in the header section of each message buffer is set to "0" through WRHS1.

A reception buffer can be assigned to a FlexRay controller channel in the following ways:

- Static segment: Channel A or channel B  
Channel A and channel B
- Dynamic segment: Channel A or channel B

The FlexRay controller stores all the elements (except the frame CRC) of the frame that matches the filter criteria for reception buffers.

All reception message buffers configured for static segments or dynamic segments can be reconfigured during execution, through the MRC:SEC[1:0] setting. (See Section "3.11.1. Message Buffer Reconfiguration") However, if the header section of the message buffer is reconfigured during execution, the messages received in each communication cycle may be lost.

If multiple buffers match the filter criteria, the reception buffer with the smallest message buffer number is updated by the received message.

### 3.9.2. Frame Reception

The following procedure is required for preparing a message buffer for reception.

- Configure the reception buffer in the message RAM through WRHS1, WRHS2, and WRHS3.
- Transfer settings from the input buffer to the message RAM by writing the target message buffer number to the IBCR register.

When these steps are performed, the message buffer starts functioning as the active reception buffer, and the acceptance filtering process is executed each time that a message is received. The reception buffer that matches the filter criteria first is updated by the received message.

If a valid payload segment is stored in the data section of the message buffer, each ND flag of the NDAT1/2/3/4 register is set to "1". Also, if the MBI bit of the header section of the message buffer is set to "1", the SIR:RXI flag is set to "1". An interrupt is generated if the interrupt would be valid.

If the ND bit is already set to "1" when the message buffer is updated, MBS:MLST in the reception message buffer is set, and unprocessed message data is lost. If a slot has no frame in it or receives a null frame or corrupted frame, the data section of the message buffer configured for this slot is not updated. In this case, only the individual message buffer status flags are updated.

Each MBS flag of the MBSC1/2/3/4 register is set to "1" when the status flag in the header section of the message buffer is updated. If the MBI bit of the header section of the message buffer is set to "1", the SIR:MBSI flag is set to "1". An interrupt is generated if the interrupt would be valid.

If the payload length PLR[6:0] of a received frame is longer than the set value in PLC[6:0] in the header section of each message buffer, the data field stored in the message buffer is truncated to that length.

For data transfer between the output buffer (OBF) and the message RAM, follow the procedure described in "b) Data Transmission from Message RAM to Output Buffer" of Section "3.11.2. Host Access to Message RAM".

**Note:**

- The ND and MBS flags are cleared to "0" when the payload data and header, respectively, of the received message are transferred to the output buffer.

### 3.9.3. Null Frame Reception

The reception buffer does not reflect the payload segment of a received null frame. If a null frame is received, the header section of the reception buffer is updated by the received null frame. The null frame indicator, NFI, in the header section of the reception message buffer is cleared to "0".

Each MBS flag of the MBSC1/2/3/4 register is set to "1" when the status flag in the header section of the message buffer is updated. If the MBI bit of the header section of the message buffer is set to "1", the SIR:MBSI flag is set to "1". An interrupt is generated if the interrupt would be valid.



## 3.10. FIFO Function

### 3.10.1. Details

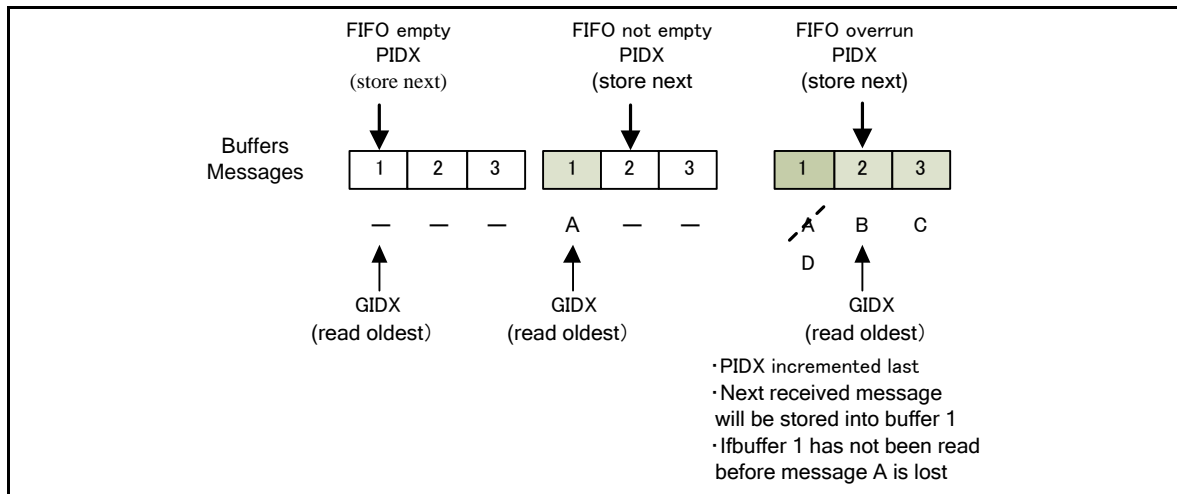
You can configure 1 group of message buffers as the FIFO buffer. Message buffers belonging to a FIFO message group are adjacent to one another in the register map, starting with the message buffer referenced by MRC:FFB[7:0] and ending with the message buffer referenced by MRC:LCB[7:0]. A maximum of 127 message buffers can be assigned to the FIFO.

The FIFO stores all valid received messages that do not match the filter criteria of dedicated reception buffers and match the set FIFO filter criteria. In such cases, the frame ID, payload length, receive cycle count, and status bit of the specified FIFO message buffer are overwritten by a received frame. Also, the SIR:RFNE bit indicates that the FIFO is not empty, and the SIR:RFF bit is set to "1" when the reception FIFO is becoming full, and the EIR:RFO bit indicates the detection of a FIFO overrun. An interrupt is generated if the interrupt would be valid.

The FIFO has 2 index registers tied to the FIFO itself: the PUTIndex (PIDX) register, and the GETIndex (GIDX) register. The PIDX register indicates the place to store the next message in the FIFO. When a new message is received, the message is written to the message buffer specified by the PIDX register. Then, the PIDX register value is incremented to indicate the message buffer for storing the next message. If the PIDX register value exceeds the largest message buffer number in the FIFO, the PIDX register value becomes the smallest message buffer number in the FIFO. The GIDX register is used to specify the next message buffer to read in the FIFO. After transfer of the contents of the message buffer belonging to the FIFO message group to the output buffer, the GIDX register value is incremented. The PIDX register and GIDX register are not accessible.

The FIFO becomes full when the PIDX register value reaches the GIDX register value. If a new message is written before the oldest message in the FIFO is read, both the PIDX register value and the GIDX register value are incremented, and the new message overwrites the oldest message in the FIFO. As a result, the EIR:RFO flag is set to "1"

Figure 3-7 FIFO Status: Empty, Not Empty, and Overrun



If the PIDX register value differs from the GIDX register value, the FIFO not-empty status is detected, and the SIR:RFNE flag is set to "1". This indicates that the FIFO has at least 1 received message. Figure 3-7 shows the FIFO empty status, FIFO not-empty status, and FIFO overrun status, where the FIFO is assumed to contain 3 message buffers.

The FIFO rejection filter (FRF) defines a filter pattern for rejecting messages. The filter consists of the channel filter, frame ID, and cycle counter filter. If the FRF:RSS bit is set to "1", the FIFO filter rejects all the messages received in the static segment. If the FRF:RNF bit is set to "1", the FIFO does not store received null frames.

The FIFO rejection filter mask (FRFM) specifies which frame ID filter bit in the FIFO rejection filter register is not used for rejection filtering.

### 3.10.2. FIFO Settings

Set the payload length PLC[6:0] to the same value on all message buffers belonging to the FIFO, through WRHS2. Also, set the data pointer to the first 32-bit word of the data section of each message buffer in the message RAM through WRHS3.

All the information required for the acceptance filter is set by the FIFO rejection filter and FIFO rejection filter mask. Therefore, there is no need to set the filter criteria in the header section of each message buffer belonging to the FIFO.

**Note:**

- The setting of "0" for WRHS1:MBI and the MBI bit programmed in message buffers as the FIFO should be performed to prevent RX interrupts from being generated.  
If the payload length of a received frame is longer than the set value of PLC[6:0] in the header section of each message buffer, the data field stored in the FIFO message buffer is truncated to the length of PLC[6:0].

### 3.10.3. Access to FIFO

By writing the first FIFO message buffer number (referenced by MRC:FFB[7:0]) to the OBCR register, transfer it from the message RAM to the output buffer in order to read from the FIFO. That operation transfers the message buffer specified by the GIDX register to the output buffer. After this transfer, the GIDX register value is incremented.



### 3.11. Message Handling

The message handler controls data transfer between the input/output buffer and message RAM, and between the message RAM and 2 transient buffer RAM units. Any access to the internal RAM is in units of 32+1 bits. The additional bit is used for a parity check.

The message buffer stored in the message RAM is accessed under the control of the message handler state machine. This prevents access collisions between the hosts to 2 FlexRay channel protocol controllers and the message RAM.

The frame ID of the message buffer assigned to the static segment must be in a range of 1 to GTUC7:NSS[9:0]. The frame ID of the message buffer assigned to the dynamic segment must be in a range of GTUC7:NSS[9:0]+1 to 2047.

Any received message that does not match the filter criteria of the dedicated reception buffer (static segment or dynamic segment) but matches the filter criteria of the FIFO rejection filter is stored in the reception FIFO (if it is so set).

#### 3.11.1. Message Buffer Reconfiguration

For an application that requires more than 128 message buffers, the static message buffer and dynamic message buffer may be reconfigured while the FlexRay controller is still operating. This can be done by updating the header section of each message buffer via the WRHS1 to WRHS3 input buffer registers.

The MRC:SEC[1:0] control bits in the message RAM setting register must enable that reconfiguration.

If the message buffer is not updated by a received frame or a transmission message is not transmitted from the message buffer before the reconfiguration begins, the message is lost.

Setting the frame ID of the reconfigured message buffer again enables that transmission or reception. The transmission or reception enable timing depends on the state of the current slot counter when the update of the header section is completed. Therefore, depending on the message buffer reconfiguration cycle, the received frame may fail to update the message buffer or the transmission message in the message buffer may not be transmitted.

A message RAM scan ends at the start of NIT even if it has not been completed. The message RAM scan for 2 to 15 slots starts at the beginning of slot 1 of the actual cycle. The message RAM scan on slot 1 is performed before the cycle, in parallel with the check by each scan on the message RAM even when a message buffer is configured for slot 1 of the next cycle.

The first dynamic message buffer number is set by MRC:FDB[7:0]. If the message RAM scan starts while CC is in the dynamic segment, the scan starts at the set message buffer number in MRC:FDB[7:0].

The following procedure is required for reconfiguring the message buffer for use in slot 1 of the next cycle.

- To reconfigure a "static buffer" as the message buffer for slot 1, reconfigure the message buffer before it is evaluated by the final message RAM scan in the static segment of the actual cycle.
- To reconfigure a "static buffer + dynamic buffer" as the message buffer for slot 1, reconfigure the message buffer before it is evaluated by the final message RAM scan in the actual cycle.
- The message RAM scan finishes at the start of NIT. If the message RAM scan has not yet evaluated the reconfigured message buffer by that time, the message buffer is not considered during the next cycle.

**Note:**

- *Reconfiguration of a message buffer may lead to a loss of messages, so great care must be taken in the reconfiguration. If a message buffer is reconfigured in continuous cycles, a received frame may not update the message buffer, or a transmission message in the message buffer may not be transmitted.*

### 3.11.2. Host Access to Message RAM

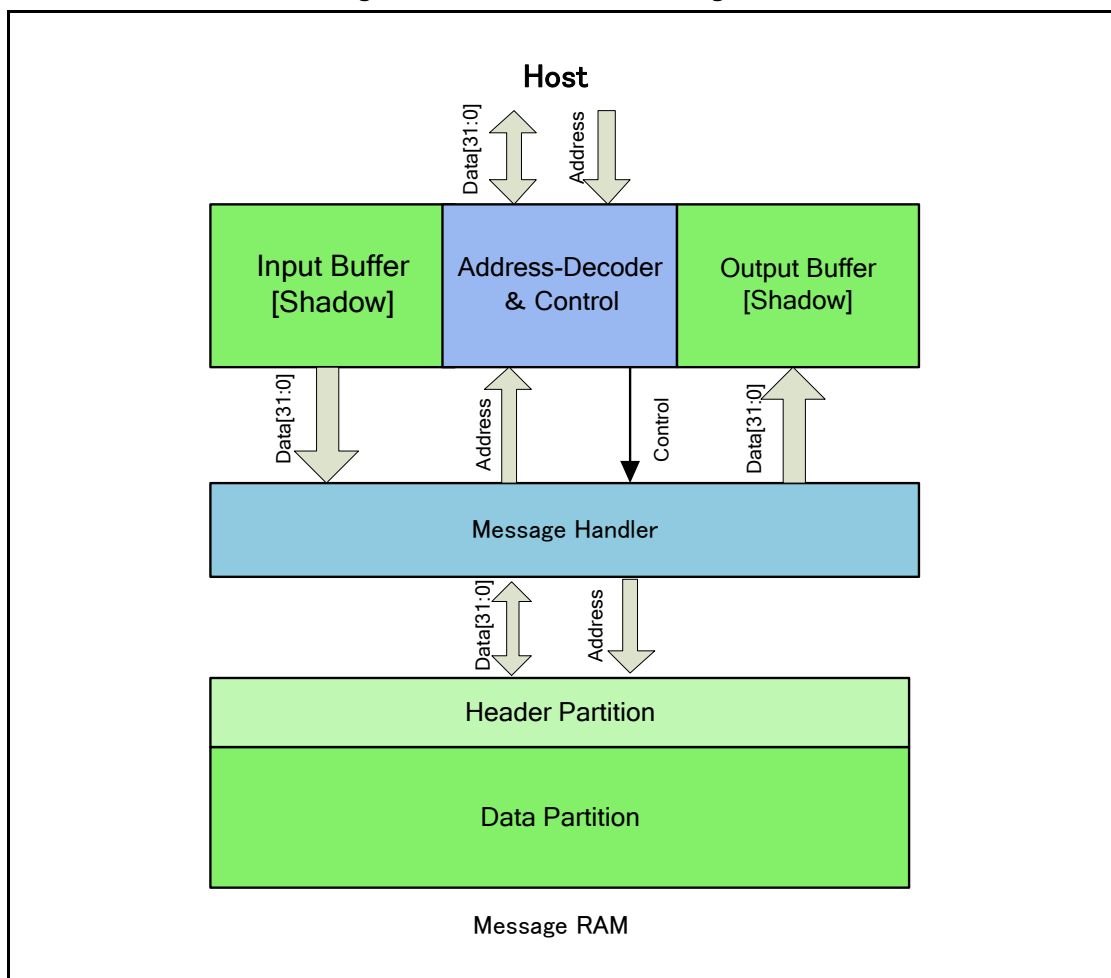
Message transfer between the input buffer and message RAM or between the message RAM and output buffer begins when the transfer target message buffer number is written to the IBCR register or OBCR register, respectively.

The IBCM register and OBCM register can be used separately to read/write the header section and data section of the selected message buffer.

If "1" is set in the IBCM:STXRH bit, the IBCM:STXRS bit is set to "1", and the selected message buffer is updated. Then, the transmission request flag TXR of the message buffer is automatically set to "1". If "0" is set in the IBCM:STXRH bit, the IBCM:STXRS bit is cleared to "0", and the transmission request flag TXR of the selected message buffer is cleared to "0". This clearing operation can be used to stop transmission from a message buffer operating in continuous mode.

The input buffer (IBF) and output buffer (OBF) compose a double buffer. The IBF host/OBF host in this double-buffer configuration can be accessed from the host. In contrast, the IBF shadow/OBF shadow is accessed by the message handler to transfer data between the IBF/OBF and message RAM.

Figure 3-8 Host Access to Message RAM

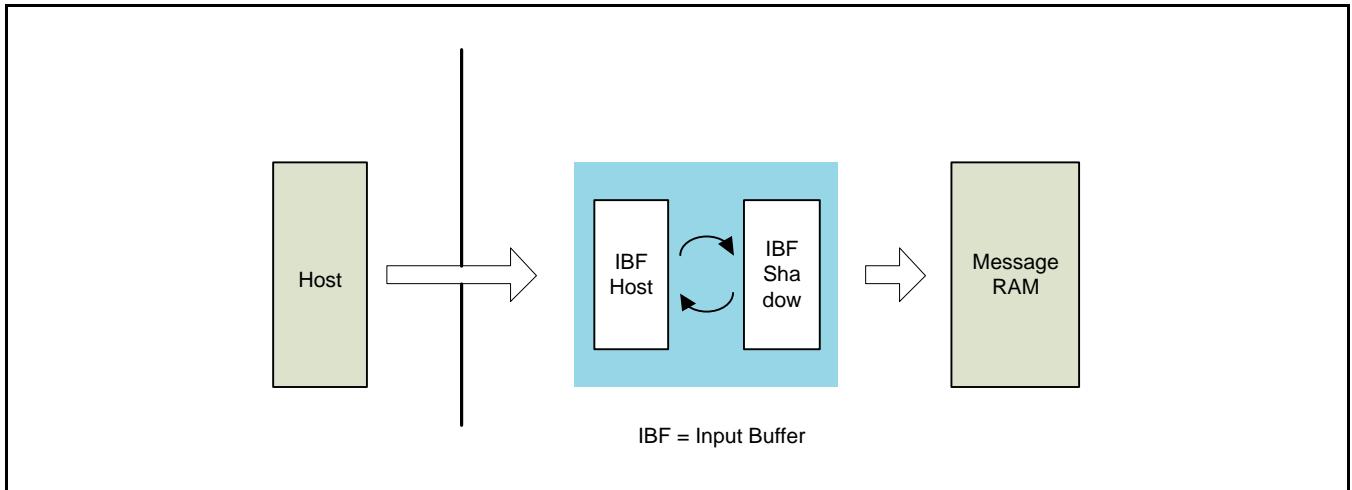


#### a) Data Transfer from Input Buffer to Message RAM

To configure/update the message buffers in the message RAM, the data must be written to WRDSn, and the header must be written to WRHS1 to WRHS3. A specific operation is selected with the setting of IBCM.

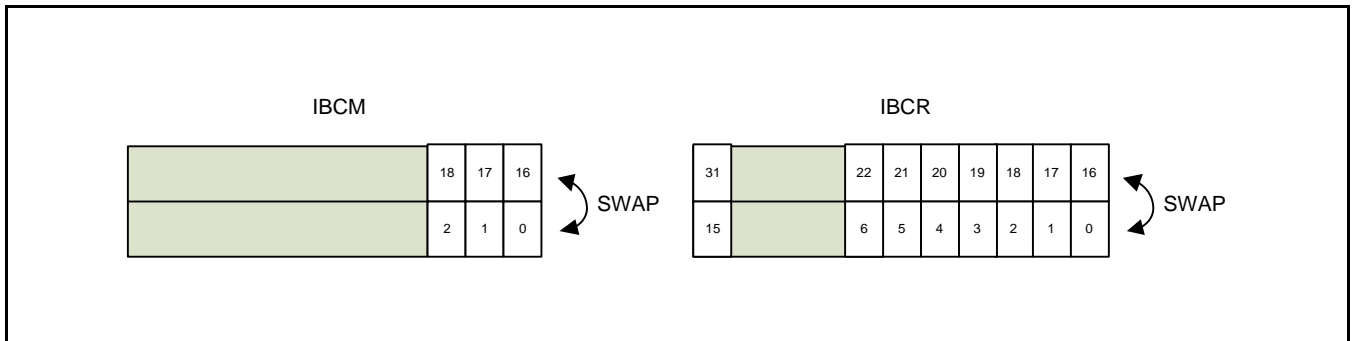
The IBF host and IBF shadow are swapped with each other through the writing of the target message buffer number of the message RAM to IBCR:IBRH[6:0]. (See Figure 3-9)

Figure 3-9 Double-buffer Structure of Input Buffer



Also, bits in the IBCM register and IBCR register are swapped with each other to retain the association with each section in the IBF. (See Figure 3-10)

Figure 3-10 IBCM Register and IBCR Register Bit Swapping



This write operation sets IBCR:IBSYS to "1". The message handler starts transmitting the contents of the IBF shadow to the message buffer in the message RAM selected by IBCR:IBRS[6:0].

The next message can be written to the IBF host while data is transferred from the IBF shadow to the target message buffer in the message RAM. After the transfer between the IBF shadow and message RAM is completed and the IBCR:IBSYS bit is cleared to "0", the next transfer to the message RAM begins when the next target message buffer number is written to IBCR:IBRH[6:0].

If anything is written to IBCR:IBRH[6:0] while IBCR:IBSYS is "1", IBCR:IBSYH is set to "1". When data transfer from the IBF shadow to the message RAM is completed, IBCR:IBSYH is cleared to "0", "1" is retained in IBCR:IBSYS, and then the next transfer to the message RAM begins. At the same time, the message buffer number and command mask flag stored in IBCR:IBRH[6:0] and IBCR:IBRS[6:0], respectively, are swapped with each other.

Example of the input buffer setting procedure:

Configure/Update the first message buffer via the IBF.

Write the data section to WRDSn.

Write the header section to WRHS1 to WRHS3.

Command mask writing: Write to IBCM:LHSH, IBCM:LDSH, and IBCM:STXRH.

Data transfer request to the target message buffer: Write to IBCR:IBRH[6:0].

Configure/Update the second message buffer via the IBF.

Write the data section to WRDSn.

Write the header section to WRHS1 to WRHS3.

Command mask writing: Write to IBCM:LHSH, IBCM:LDSH, and IBCM:STXRH.

Data transfer request to the target message buffer: Write to IBCR:IBRH[6:0] after IBCR:IBSYH is cleared to "0".

Configure/Update the third message buffer via the IBF.

... (Subsequently, repeat the same procedure to configure/update the second message buffer.)

**Note:**

- If IBCR:IBSYH is "1", access to the input buffer sets the EIR:IBA error flag to "1". In this case, access is disabled.

**Table 3-7 Assignment of Input Buffer Command Mask Bits**

Position	Access	Bit	Function
18	r	STXRS	Sets the start or end of the transmission request shadow.
17	r	LDSS	Reads the start or end of the data section shadow.
16	r	LHSS	Reads the start or end of the header section shadow.
2	r/w	STXRH	Sets the transmission request host.
1	r/w	LDSH	Reads the data section host.
0	r/w	LHSH	Reads the header section host.

**Table 3-8 Assignment of Input Buffer Request Mask Bits**

Position	Access	Bit	Function
31	r	IBSYS	IBF busy shadow, Signal for starting transfer from the running IBF shadow to the message RAM
22...16	r	IBRS[6:0]	IBF request shadow, Message buffer number for the current or final update
15	r	IBSYH	IBF busy host, Transfer request pending for the message buffer referenced by IBRH6:0
6...0	r/w	IBRH[6:0]	IBF request host, Message buffer number for the next update

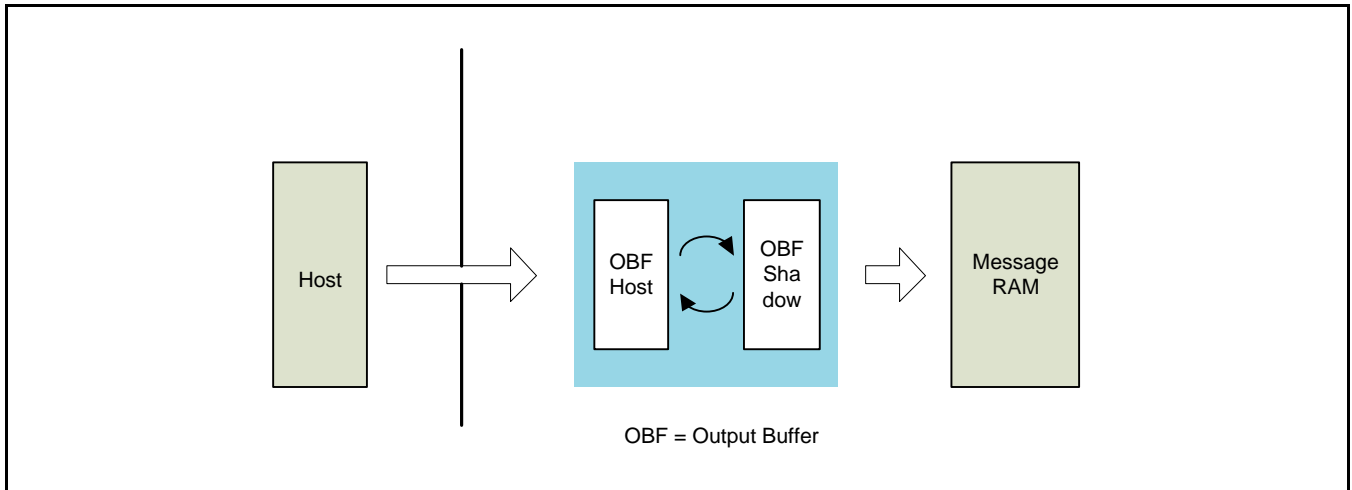


#### b) Data Transmission from Message RAM to Output Buffer

To read the message buffer from the message RAM, writing to the OBCR register is required to trigger data transfer as configured in OBCM. After the transfer is completed, the data transferred from RDDSn, RDHS1 to RDHS3, and MBS can be read.

OBCR:OBR[6:0] sets the buffer number of the transfer source message buffer in the message RAM.

**Figure 3-11 Double-buffer Structure of Output Buffer**

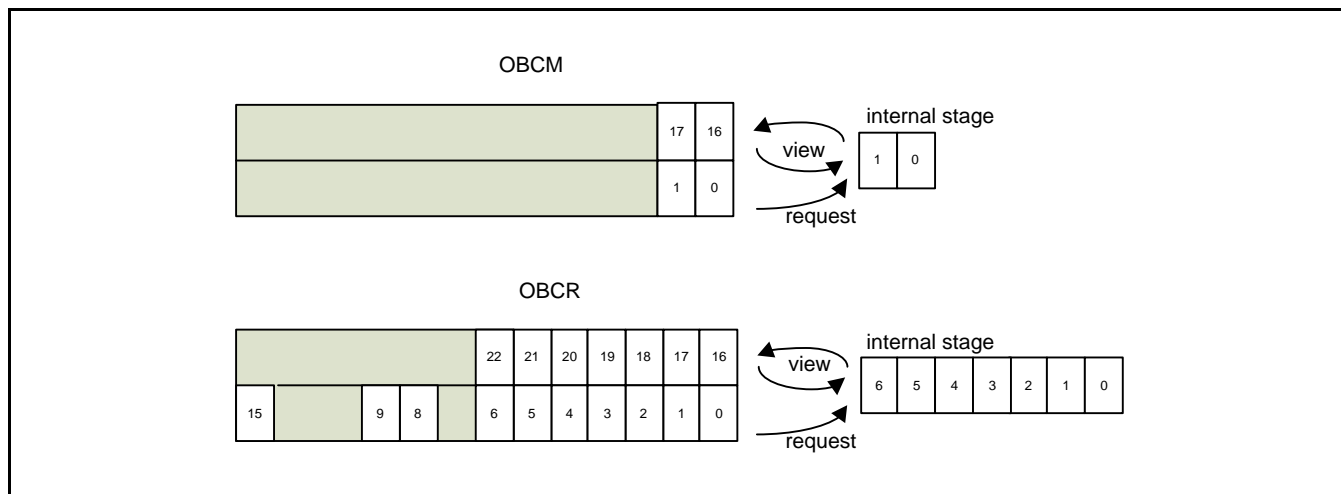


As well as each OBCM:RHSS, OBCM:RDSS, OBCM:RHS, and OBCM:RDSH bit and each OBCR:OBR[6:0] and OBCR:OBRH[6:0] bit, the OBF host and OBF shadow are swapped with each other according to the OBCR:VIEW and OBCR:REQ bit settings.

With the OBCR:REQ bit set to "1", each OBCM:RHSS, OBCM:RDSS, and OBCR:OBR[6:0] bit is copied to the internal registers. (See Figure 3-12)

After OBCR:REQ is set to "1", OBCR:OBSYS is set to "1", and transfer begins from the message RAM of the message buffer selected by OBCR:OBR[6:0] to the OBF shadow. After the transfer from the message RAM to the OBF shadow is completed, the OBCR:OBSYS bit is cleared to "0". OBCR:REQ and OBCR:VIEW can be set to "1" while OBCR:OBSYS is "0".

Figure 3-12 OBCM Register and OBCR Register Bit Swapping



If "1" is set in OBCR:VIEW while the OBCR:OBSYS bit is "0", the OBF host and OBF shadow can be swapped with each other. (See Figure 3-11) Also, the OBCR:OBRH[6:0], OBCM:RSH, and OBCM:RDSH bits are swapped with the internal registers. The internal registers store the contents copied from OBCR:OBRH[6:0], OBCM:RHSS, and OBCM:RDSS, before "1" is set in OBCR:VIEW. Swapping them guarantees that the message buffer number readable from OBCR:OBRH[6:0] and the mask settings readable from OBCM:RSH and OBCM:RDSH match the transfer data readable from the OBF host. (See Figure 3-12)

After the swap, the already transferred message buffer can be read from the OBF host, and the message handler can transfer the next message from the message RAM to the OBF shadow.

If both REQ and VIEW are set to "1" simultaneously while OBSYS is "0", OBSYS is set to "1".

Subsequently, the OBF host and OBF shadow are swapped with each other. Also, the OBCM:RDSH and OBCM:RSH mask bits are swapped with the internal registers to handle each output buffer transfer. After that, OBRH[6:0] is copied to the internal register. Then, transfer from the message RAM of the selected message buffer to the OBF shadow begins. While the transfer is in progress, the CPU can read the message buffer previously transferred from the OBF host. The OBSYS bit is cleared to "0" when the transfer between the message RAM and OBF shadow is completed.

Example of the output buffer setting procedure:

Make a request to the first message buffer for transfer to the OBF shadow.

Command mask writing: Write to OBCM:RHSS and OBCM:RDSS.

Transfer request to the first message buffer: Write to OBCR:OBRH[6:0] and OBCR:REQ.

Wait until OBCR:OBSYS is cleared to "0".

Make a request to the second message buffer for transfer to the OBF shadow, and read the first message buffer from the OBF host.

Command mask writing: Write to OBCM:RHSS and OBCM:RDSS.

Swapping of the OBF host and shadow with each other for the first message, and transfer request for the second message: Write to OBCR:VIEW, OBCR:REQ, and OBCR:OBRH[6:0].

Read the first message.

Wait until OBCR:OBSYS is cleared to "0".



Make a request to the third message buffer for transfer to the OBF shadow, and read the second message buffer from the OBF host.

Command mask writing: Write OBCM:RHSS and OBCM:RDSS.

Swapping of the OBF host and shadow with each other for the second message, and transfer request for the third message: Write to OBCR:VIEW, OBCR:REQ, and OBCR:OBR[6:0].

Read the second message.

Wait until OBCR:OBSYS is cleared to "0".

... (Repeat the same steps.)

Read the n-th message buffer from the OBF host (assuming no more requests for message buffer transfer).

Swapping of the OBF host and shadow with each other for the n-th message: Write to OBCR:VIEW (do not write to OBCR:OBR[6:0]).

Read the n-th message.

**Table 3-9 Assignment of Output Buffer Command Mask Bits**

Position	Access	Bit	Function
17	r	RDSH	Data section that can be used for host access
16	r	RHSH	Header section that can be used for host access
1	r/w	RDSS	Reads the data section shadow.
0	r/w	RHSS	Reads the header section shadow.

**Table 3-10 Assignment of Output Buffer Request Mask Bits**

Position	Access	Bit	Function
22...16	r	OBRH[6:0]	OBF request host, Message buffer number that can be used for host access
15	r	OBSYS	OBF busy shadow, Signal for starting the transfer from the message RAM to the running OBF shadow
9	r/w	REQ	Requests transfer from the message RAM to the OBF shadow.
8	r/w	VIEW	Displays the OBF shadow, and swaps the OBF shadow and OBF host with each other.
6...0	r/w	OBR[6:0]	OBF request shadow, Message buffer number for the next request

### 3.11.3. FlexRay Protocol Controller Access to Message RAM

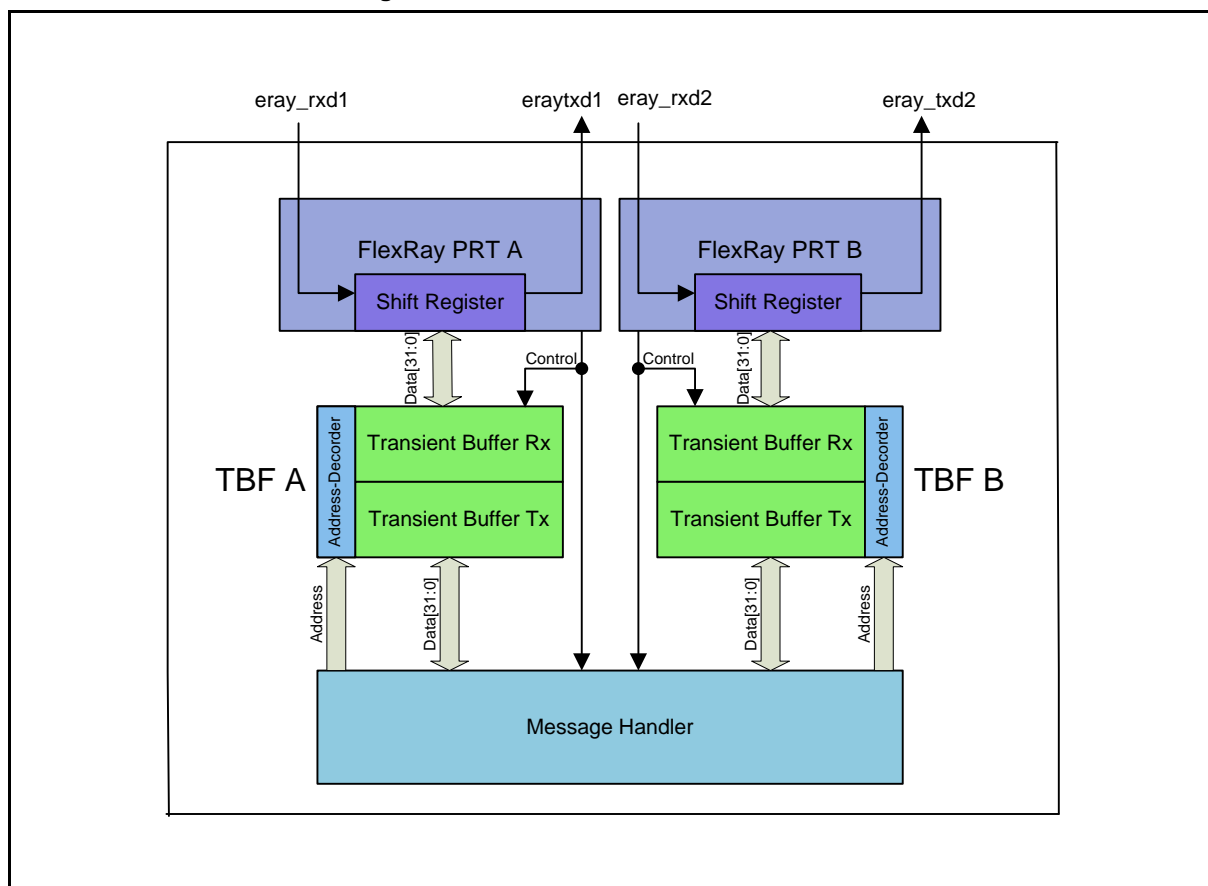
Transfer between 2 FlexRay channel protocol controllers and the message RAM uses 2 transient buffer RAM units (TBF A and TBF B) to buffer data.

Those 2 transient buffer RAM units compose a double buffer, and they can store 2 complete FlexRay messages. With 1 buffer accessible to the message handler, the other buffer is assigned to the corresponding channel protocol controller.

For example, if the message handler writes a transmission message to the transient buffer Tx, the FlexRay channel protocol controller can store the currently received message in the transient buffer Rx. While the message stored in the transient buffer Tx is being transmitted, the message handler transfers the latest received message (if it passes the acceptance filter) stored in the transient buffer Rx to the message RAM and updates the message buffer.

The data transfer between the transient buffer RAM and the shift register of the FlexRay channel protocol controller is in units of 32-bit words. This enables the use of an independent 32-bit shift register whose length is equal to the FlexRay message length.

Figure 3-13 Access to Transient Buffer RAM



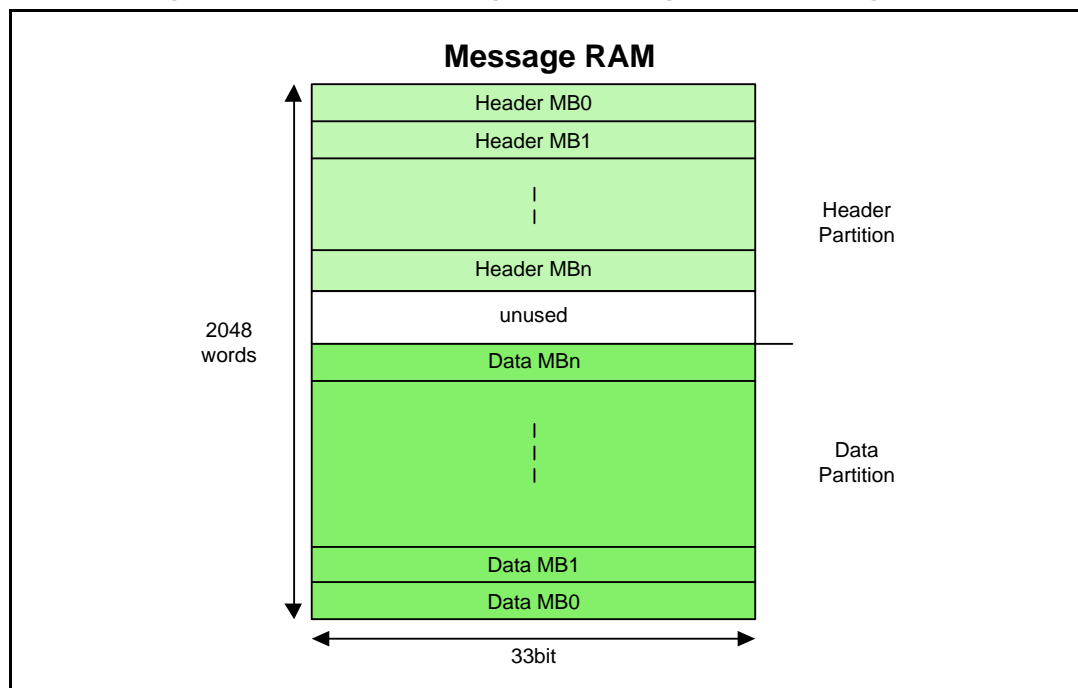
### 3.12. Message RAM

Direct access to the message buffer in the message RAM is disabled to prevent collisions between host access to the message RAM and transmission and reception of FlexRay messages. Access is processed via the input buffer and output buffer. The message RAM can store up to 128 message buffers.

The message RAM consists of 2048 bytes × 33 bits, which equals 67,584 bits, and a parity bit protects each piece of 32-bit data. The structure of the message RAM can have a variable (0 to 254) number of data bytes per FlexRay frame, as shown in Figure 3-14.

The data partition starts at  $(MRC:LCB+1) \times 4$  words in the message RAM (1 word=32+1 bits).

**Figure 3-14 Example of Message Buffer Configuration in Message RAM**



#### Header Partition

The header partition stores the header section of a configured message buffer.

A maximum of 128 message buffers are supported.

Each message buffer has a 4-word (1 word=32+1 bits) header section.

Header 3 of each message buffer has an 11-bit data pointer for each data section of the data partition.

#### Data Partition

The data partition is a variable-length memory area that stores data sections with different data lengths. The maximum number of message buffers for various data lengths are as follows.

If each data section is 254 bytes, the maximum is 30 message buffers.

If each data section is 128 bytes, the maximum is 56 message buffers.

If each data section is 48 bytes, the maximum is 128 message buffers.

#### Note:

- Be sure to configure the area used for the header partition + data partition as an area of 2048 words (1 word = 33 bits) or less.

### 3.12.1. Header Partition

The header partition of the message RAM stores the message buffer status and message buffer setting elements as shown in Figure 3-15 below. The header section of the message buffer is configured via the IBF (WRHS1 to WRHS3), and the header section is read via the OBF (RDHS1 to RDHS3 + MBS). Set the data pointer in the header section to define the start position of the data section of each message buffer. Also, do not modify the data pointer during execution. Configure (reconfigure) the message buffers belonging to the FIFO message group in the DEFAULT\_CONFIG or CONFIG state.

The header section of each message buffer occupies 4 words (1 word=32+1 bits) in the header partition of the message RAM. The header section of message buffer 0 starts at the top of the message RAM.

Obtain the header CRC of the transmission buffer through calculations.

Valid received frames (including valid null frames) update the reception payload length, PLR[6:0], the reception cycle counter, RCC[5:0], the reception channel indicator, RCI, the startup frame indicator, SFI, the sync frame indicator, SYN, the null frame indicator, NFI, the payload preamble indicator, PPI, and the reserved bit, RES.

A 4-word area in the header of each configured message buffer contains the message buffer status, MBS.

Figure 3-15 Header Section of Message Buffer in Message RAM

Bit Word	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	P			M B I	T X M	P P I T	C F G	C H B	C H A																								
1	P																																
2	P																																
3	P																																
...	P																																
...	P																																

	Frame Configuration	(PPIT, CFG, FrameID, Payload, Length Configured)
	Filter Configuration	(CHB, CHA, Cycle Code)
	Message Buffer Control	(MIB, TXM)
	Message RAM Configuration	(Data Pointer)
	Updated from received Frame	(Payload Length Received, RES, PPI, NFI, SYN, SFI and RCI Receive Cycle Count)
	Message Buffer Status MBS	(MLST, ESB, ESA, TCIB, TCIA, SVOB, SVOA, CEOB, CEOA, SEOB, SEOA, VFRB, VFRA, RESS, PPIS, NFIS, SYNS, SFIS, RCIS, Cycle Count Status, FTA, FTB)
	Parity bit	
	Unused	



#### Header 1

The following parameters are written via WRHS1 and read via RDHS1.

Frame ID	Slot counter filtering setting
Cycle code	Cycle counter filtering setting
CHA, CHB	Channel filtering settings
CFG	Message buffer setting: Reception/Transmission
PPIT	Payload preamble indicator transmission
TXM	Transmission mode setting: Single-shot/Continuous
MBI	Enable flag for message buffer transmission/reception interrupts

#### Header 2

The following parameters are written via WRHS2 and read via RDHS2.

Header CRC	Transmission buffer: Calculated from the frame header segment Reception buffer: Updated by the received frame
Configured payload length	Set data section length (in 2-byte units)
Reception payload length	Payload segment length stored in the received frame (in 2-byte units)

#### Header 3

The following parameters are written via WRHS3 and read via RDHS3.

Data pointer	Pointer to the start position of the data section corresponding to the data partition
--------------	---

The following parameters are read via RDHS3.

They are valid only for reception buffers and updated by received frames.

Receive cycle count	Stored cycle count value from the received frame
RCI	Reception channel indicator
SFI	Startup frame indicator
SYN	Sync frame indicator
NFI	Null frame indicator
PPI	Payload preamble indicator
RES	Reserved bit

#### Header 4

Header 4 is read via MBS. It is updated when the set slot is finished.

The following parameters are valid for the transmission buffer and reception buffer.

VFRA	Channel A reception valid frame
VFRB	Channel B reception valid frame
SEOA	Channel A syntax error
SEOB	Channel B syntax error
CEOA	Channel A content error
CEOB	Channel B content error
SVOA	Channel A slot boundary violation
SVOB	Channel B slot boundary violation

The following parameters are valid only for the transmission buffer.

TCIA	Channel A transmission conflict indicator
TCIB	Channel B transmission conflict indicator

The following parameters are valid only for the reception buffer.

ESA	Channel A empty slot
ESB	Channel B empty slot
MLST	Message lost
FTA	Channel A frame transmission
FTB	Channel B frame transmission
Cycle Count Status	Actual cycle count at the status update time
RCIS	Channel indicator reception
SFIS	Startup frame indicator status
SYNS	Sync frame indicator status
NFIS	Null frame indicator status
PPIS	Payload preamble indicator status
RESS	Reserved bit status

### 3.12.2. Data Partition

The data partition in the message RAM stores the data section of the message buffer configured for reception or transmission as defined in the header partition. The number of data bytes that can be set for each message buffer ranges from 0 to 254 bytes. The bit width of the message RAM is set in 32 bits + 1 parity bit to optimize the data transfer between the host interface and message RAM and the data transfer between the shift register of 2 FlexRay channel protocol controllers and the message RAM.

The data partition starts immediately after the header partition. Set the data pointer to point to an address within the data partition when configuring the message buffer in the message RAM. Figure 3-16 below shows an example of how the data section of a configured message buffer is stored in the data partition in the message RAM.

The start and end positions of the data section of the message buffer are determined by the set data pointer and payload length in the header section of the message buffer. This enables flexible use of RAM space with various data lengths in a message buffer.

If the data section size is an odd number of a 2-byte unit, the remaining 16 bits in the last 32-bit word are not used. (See Figure 3-16)

**Figure 3-16 Example of Data Section Structure in Message RAM**

Bit Word	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
...	P	unused								unused								unused								unused							
...	P	unused								unused								unused								unused							
...	P	MBn Data3								MBn Data2								MBn Data1								MBn Data0							
...	P	...								...								...								...							
...	P	...								...								...								...							
...	P	MBn Data(m)								MBn Data(m-1)								MBn Data(m-2)								MBn Data(m-3)							
...	P	...								...								...								...							
...	P	...								...								...								...							
...	P	...								...								...								...							
...	P	MB1 Data3								MB1 Data2								MB1 Data1								MB1 Data0							
...	P	...								...								...								...							
...	P	MBn Data(K)								MBn Data(K-1)								MBn Data(K-2)								MBn Data(K-3)							
2046	P	MB0 Data3								MB0 Data2								MB0 Data1								MB0 Data0							
2047	P	unused								unused								MB0 Data5								MB0 Data4							



### 3.12.3. Parity Check

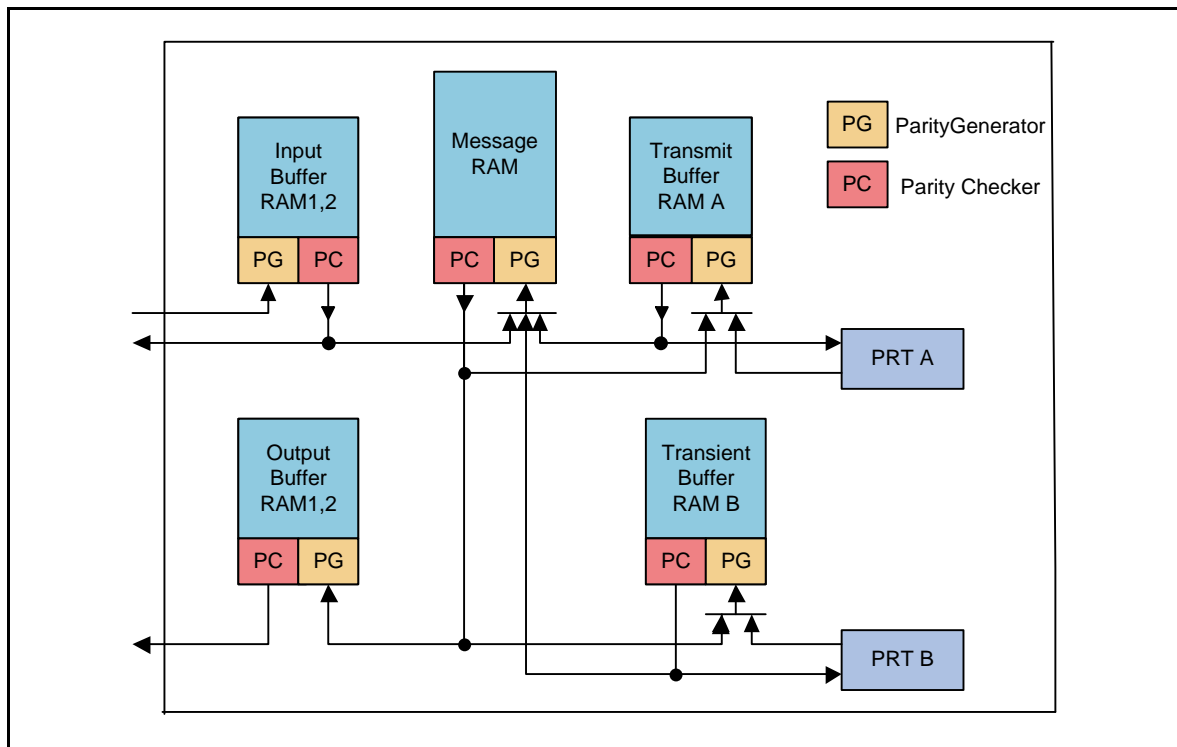
The FlexRay controller is equipped with a parity check mechanism to guarantee the integrity of data stored in 7 RAM blocks. These RAM blocks include the parity generator/checker connected as shown in Figure 3-17, and the parity generator generates the parity bit when data is written to a RAM block. The FlexRay controller uses even-numbered parity. (The parity bit is generated as "0" when there is an even number of "1"s in a 32-bit word.)

The parity bit is stored in each piece of data. Also, the parity is checked when data is read from a RAM block. The bit width of the internal data bus of the FlexRay controller is 32 bits.

The respective error flag is set to "1" when a parity error is detected. The message handler status register contains the parity error flags (MHDS:PIBF, MHDS:POBF, MHDS:PMR, MHDS:PTBF1, and MHDS:PTBF2) and the faulty message buffer indicators (MHDS:FMBD, MHDS:MFMB, and MHDS:FMB[6:0]). These error flags control the EIR:PERR error interrupt flag.

Figure 3-17 shows the data paths between the RAM blocks and the parity generator/checker.

Figure 3-17 Parity Generation and Check



**Note:**

- The parity generator and parity checker are blocks independent from the RAM block.

The following operations are executed when a parity error is detected.

All cases:

- The corresponding parity error flag of the MHDS register is set.
- The parity error flag, EIR:PERR, is set. An interrupt is generated if the interrupt would be valid.

Special cases:

1. Parity error during data transfer from input buffer RAM 1 or 2 to the message RAM

a) Transfer of the header and data, or transfer of data:

- The MHDS:PIBF bit is set.
- The MHDS:FMBD bit is set to indicate that MHDS:FMB[6:0] was updated.
- MHDS:FMB[6:0] indicates the message buffer number with the error.
- No transmission request bit is set for the transmission buffer where the parity error occurred.

b) Transfer of data

Parity error when the header of each message is read from the message RAM

- The MHDS:PMR bit is set.
- The MHDS:FMBD bit is set to indicate that MHDS:FMB[6:0] was updated.
- MHDS:FMB[6:0] indicates the message buffer number with the error.
- Data in the message buffer is not updated.
- No transmission request bit is set for the transmission buffer where the parity error occurred.

2. Parity error during data transfer from input buffer RAM 1 or 2 to the host

- The MHDS:PIBF bit is set.

3. Parity error during scanning of the header section of the message RAM

- The MHDS:PMR bit is set.
- The MHDS:FMBD bit is set to indicate that MHDS:FMB[6:0] was read.
- MHDS:FMB[6:0] indicates the message buffer number with the error.
- The message buffer with the parity error is ignored.

4. Parity error during data transfer from the message RAM to transient buffer RAM 1 or 2

- MHDS:PMR is set.
- MHDS:FMBD is set to indicate that MHDS:FMB[6:0] was read.
- MHDS:FMB[6:0] indicates the message buffer number with the error.
- Frame transmission from the message buffer number with the error is stopped.

5. Parity error during data transfer from transient buffer RAM 1 or 2 to channel protocol controller 1 or 2

- MHDS:PTBF1 and MHDS:PTBF2 are set.
- Frame transmission from the transient buffer number with the error is stopped.

6. Parity error during data transfer from transient buffer RAM 1 or 2 to the message RAM

- MHDS:PTBF1 and MHDS:PTBF2 are set.
- MHDS:FMBD is set to indicate that MHDS:FMB[6:0] was updated.
- MHDS:FMB[6:0] indicates the message buffer number with the error.

7. Parity error during data transfer from the message RAM to the output buffer RAM

- MHDS:PMR is set.
- MHDS:FMBD is set to indicate that MHDS:FMB[6:0] was read.
- MHDS:FMB[6:0] indicates the message buffer number with the error.

8. Parity error during data transfer from the output buffer RAM to the host

- MHDS:POBF is set.

9. Parity error while data is read from transient buffer RAM 1 or 2



If a parity error occurs when the message handler reads a frame with network management information (PPI is "1") from transient buffer RAM 1 or 2, the MV1 to MV3 network management vectors corresponding to the frame are not updated.

### 3.12.4. Parity Error Handling

Restoration from a parity error is done through a transfer.

#### (1) Self-restore

- Input buffer RAM 1 and 2
- Output buffer RAM 1 and 2
- Data in message RAM
- Transient buffer RAM A
- Transient buffer RAM B

Self-restoration from the occurrences of the aforementioned parity errors is possible through overwriting by CPU access in FlexRay communication.

#### (2) CLEAR RAM Command

For the DEFAULT\_CONFIG or CONFIG state, the CLEAR\_RAM command initializes the RAM in all modules to zero.

#### (3) Temporary Unlocking of Header Section

Restoration from a parity error in the header section of a locked message buffer is possible with a transfer from the input buffer to the header section of the locked buffer.

In this transfer, write access to IBCR (specifying the message buffer number) must precede the unlock operation from the CONFIG state. (See Section "4.2.1. Lock Register".)

In that single transfer, each message buffer header is unlocked and data is updated regardless of whether it belongs to the FIFO or whether the lock operation belongs to it according to MRC:SEC[1:0].

### 3.13. Interrupts

Interrupt lines to generate interrupts immediately are provided for the occurrence of the following events: error occurrence, detection of a status change, frame transmission/reception, and a timer event. This enables very quick response to any error condition, status change, or timer event. However, if too many interrupts are generated, the operation rate required for an application may not be met. Therefore, the FlexRay controller supports a function for enabling/disabling operation per interrupt.

An interrupt is generated in the following cases.

- An error is detected.
- A status flag is set.
- The timer reaches the set value.
- A message is transferred from the input buffer to the message buffer or from the message RAM to the output buffer.
- A stop watch event occurs.

Event display and interrupt generation for status changes or error occurrences work as 2 independent tasks. Each event is displayed regardless of whether the interrupt is enabled. The current error information and status information can be obtained from the reading of the EIR register and SIR register, respectively.

**Table 3-11 Module Interrupt Flags and Interrupt Line Valid Flags**

Register	Bit	Function
EIR	PEMC	POC error mode changed flag
	CNA	Invalid command notification flag
	SFBM	Frames below minimum flag
	SFO	Synchronization frame overflow flag
	CCF	Clock correction failure flag
	CCL	CHI command locked flag
	PERR	Parity error flag
	RFO	Reception FIFO overrun flag
	EFA	Empty FIFO access flag
	IIBA	Illegal input buffer access flag
	IOBA	Illegal output buffer access flag
	MHF	Message handler constraints flag
	EDA	Error detected on channel A flag
	LTVA	Channel A transmission failure detection flag
	TABA	Transmission across channel A slot boundary detection flag
	EDB	Error detected on channel B flag
	LTVB	Channel B transmission failure detection flag
	TABB	Transmission across channel B slot boundary detection flag
SIR	WST	Wakeup status flag
	CAS	Collision avoidance symbol flag
	CYCS	Communication cycle start flag
	TXI	Transmission completion flag
	RXI	Reception completion flag
	RFNE	Reception FIFO flag
	RFF	Reception FIFO full flag
	NMVC	Network management vector changed flag
	TI0	Timer 0 flag
	TI1	Timer 1 flag
	TIBC	Transfer input buffer completed flag
	TOBC	Transfer output buffer completed flag



Register	Bit	Function
SIR	SWE	Stop watch event flag
	SUCS	Startup completed successfully flag
	MBSI	Message buffer status changed flag
	SDS	Start of dynamic segment flag
	WUPA	Channel A wakeup pattern reception flag
	MTSA	MTS received on channel A flag
	WUPB	Channel B wakeup pattern recognition flag
	MTSB	MTS received on channel B flag
ILE	EINT0	Interrupt line INT0 enable flag
	EINT1	Interrupt line INT1 enable flag

The INT0 and INT1 interrupt lines are controlled by valid interrupts. Also, these 2 interrupt lines, INT0 and INT1, can be enabled or disabled separately with the ILE:EINT0 and ILE:EINT1 settings, respectively.

The 2 timer interrupts generated by interrupt timer 0 and interrupt timer 1 can be used by the INT2 line in 16-bit non-multiplex bus mode and the INT2 and INT3 lines in 16-bit multiplex bus mode. They can be configured using the T0C register and T1C register.

The STOPWT input pin generates the stop watch event.

Each of the SIR:TIBC and SIR:TOBC bits is set to "1" when data transfer between the IBF/OBF and message RAM is completed.



4. Registers

The FlexRay controller has 2-Kbyte address space (from 0x0000 to 0x07FF) and its registers are configured as 32-bit registers. Host access to the message RAM (access from the host CPU) is performed through input buffers and output buffers. To avoid conflict between host access and message transmission/reception, these buffers are used to buffer data for transfer to the message RAM and data that has been transferred from the message RAM.

The number (N) of available message buffers depends on the payload length of the configured message buffer. The maximum number of message buffers is 128 and the maximum payload length is 254 bytes. The message buffers are allocated as shown in Figure 4-1. The message buffers are classified into three consecutive groups.

- |                          |   |
|--------------------------|---|
| Static buffers           | - Transmission and reception buffers allocated to static segments                     |
| Static + dynamic buffers | - Transmission and reception buffers allocated to static segments or dynamic segments |
| FIFO                     | - Reception FIFO  |

The allocation of message buffers can be changed by setting the message RAM configuration register (MRC) in the DEFAULT\_CONFIG or CONFIG state.

The first group operates as static message buffers.

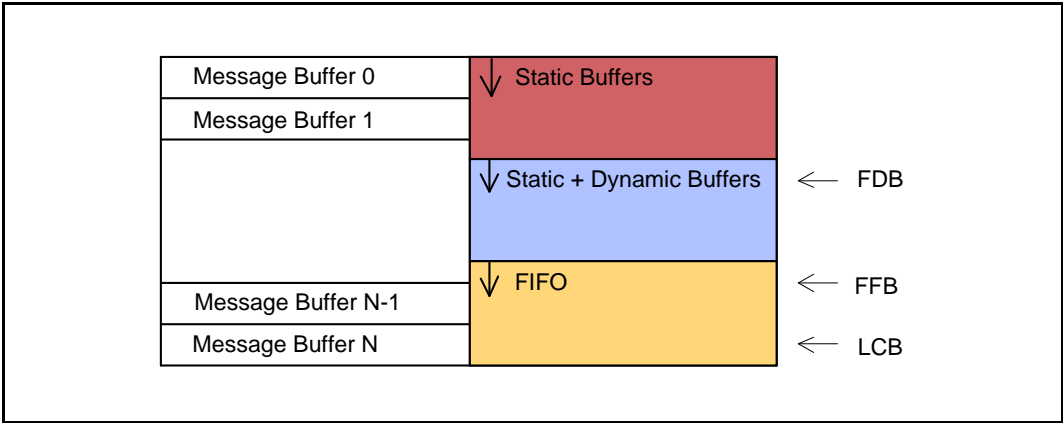
The second group operates as static/dynamic message buffers. The message buffers that belong to this group can be reconfigured during operation from dynamic segments to static segments, or from static segments to dynamic segments, depending on the state of MRC:SEC[1:0].

Message buffers that belong to the third group are connected to a reception FIFO.

Message buffer 0 is a static message buffer used to store and transmit the startup/synchronization frame, or single slot frame (a frame that is transmitted in SINGLE slot mode) depending on the setting of SUCC1:TXST, SUCC1:TXSY, and SUCC1:TSM. Message buffer 0 is required to have a key slot ID, which can be configured (or reconfigured) only in the DEFAULT\_CONFIG or CONFIG state.

FDB, FFB, and LCB in Figure 4-1 represent the first dynamic buffer number MRC:FDB[7:0], the first FIFO buffer number MRC:FFB[7:0], and the last message buffer number MRC:LCB[7:0], respectively.

Figure 4-1 Allocation of Message Buffers



**Note:**

- All FlexRay controller registers are accessed as 32-bit accesses.



**Table 4-1 List of FlexRay Controller Registers**

All FlexRay controller registers have prefix (FLXRY\_).

Abbreviated Register Name	Register Name		See
CIF0	Customer Registers	Version Information Register	4.1.1
CIF1		Control Register	4.1.2
CIF1F		Flag Register	4.1.3
CIF1C		Flag Clear Register	4.1.4
LCK	Special Register	Lock Register	4.2.1
EIR	Interrupt-related Registers	Error Interrupt Register	4.3.1
SIR		Status Interrupt Register	4.3.2
EILS		Error Interrupt Line Selection Register	4.3.3
SILS		Status Interrupt Line Selection Register	4.3.4
EIES		Error Interrupt Enable Register (set)	4.3.5
EIER		Error Interrupt Enable Register (reset)	4.3.5
SIES		Status Interrupt Enable Register (set)	4.3.6
SIER		Status Interrupt Enable Register (reset)	4.3.6
ILE		Interrupt Line Enable Register	4.3.7
T0C		Timer 0 Configuration Register	4.3.8
T1C		Timer 1 Configuration Register	4.3.9
STPW1		Stop Watch Register 1	4.3.10
STPW2		Stop Watch Register 2	4.3.11
SUCC1	Communication Controller (CC) Control Registers	SUC Configuration Register 1	4.4.1
SUCC2		SUC Configuration Register 2	4.4.2
SUCC3		SUC Configuration Register 3	4.4.3
NEMC		NEM Configuration Register	4.4.4
PRTC1		PRT Configuration Register 1	4.4.5
PRTC2		PRT Configuration Register 2	4.4.6
MHDC		MHD Configuration Register	4.4.7
GTUC1		GTU Configuration Register 1	4.4.8
GTUC2		GTU Configuration Register 2	4.4.9
GTUC3		GTU Configuration Register 3	4.4.10
GTUC4		GTU Configuration Register 4	4.4.11
GTUC5		GTU Configuration Register 5	4.4.12
GTUC6		GTU Configuration Register 6	4.4.13
GTUC7		GTU Configuration Register 7	4.4.14

Abbreviated Register Name	Register Name		See
GTUC8	Communication Controller (CC) Control Registers	GTU Configuration Register 8	4.4.15
GTUC9		GTU Configuration Register 9	4.4.16
GTUC10		GTU Configuration Register 10	4.4.17
GTUC11		GTU Configuration Register 11	4.4.18
CCSV	Communication Controller (CC) Status Registers	CC Status Vector Register	4.5.1
CCEV		CC Error Vector Register	4.5.2
SCV		Slot Counter Value Register	4.5.3
MTCCV		Macrotick and Cycle Counter Value Register	4.5.4
RCV		Rate Correction Value Register	4.5.5
OCV		Offset Correction Value Register	4.5.6
SFS		Sync Frame Status Register	4.5.7
SWNIT		Symbol Window and NIT Status Register	4.5.8
ACS		Aggregated Channel Status Register	4.5.9
ESID1		Even Cycle Sync Frame ID Register 1	4.5.10
ESID2		Even Cycle Sync Frame ID Register 2	4.5.10
ESID3		Even Cycle Sync Frame ID Register 3	4.5.10
ESID4		Even Cycle Sync Frame ID Register 4	4.5.10
ESID5		Even Cycle Sync Frame ID Register 5	4.5.10
ESID6		Even Cycle Sync Frame ID Register 6	4.5.10
ESID7		Even Cycle Sync Frame ID Register 7	4.5.10
ESID8		Even Cycle Sync Frame ID Register 8	4.5.10
ESID9		Even Cycle Sync Frame ID Register 9	4.5.10
ESID10		Even Cycle Sync Frame ID Register 10	4.5.10
ESID11		Even Cycle Sync Frame ID Register 11	4.5.10
ESID12		Even Cycle Sync Frame ID Register 12	4.5.10
ESID13		Even Cycle Sync Frame ID Register 13	4.5.10
ESID14		Even Cycle Sync Frame ID Register 14	4.5.10
ESID15		Even Cycle Sync Frame ID Register 15	4.5.10
OSID1		Odd Cycle Sync Frame ID Register 1	4.5.11
OSID2		Odd Cycle Sync Frame ID Register 2	4.5.11
OSID3		Odd Cycle Sync Frame ID Register 3	4.5.11
OSID4		Odd Cycle Sync Frame ID Register 4	4.5.11
OSID5		Odd Cycle Sync Frame ID Register 5	4.5.11
OSID6		Odd Cycle Sync Frame ID Register 6	4.5.11
OSID7		Odd Cycle Sync Frame ID Register 7	4.5.11
OSID8		Odd Cycle Sync Frame ID Register 8	4.5.11
OSID9		Odd Cycle Sync Frame ID Register 9	4.5.11





Abbreviated Register Name	Register Name		See
OSID10	Communication Controller (CC) Status Registers	Odd Cycle Sync Frame ID Register 10	4.5.11
OSID11		Odd Cycle Sync Frame ID Register 11	4.5.11
OSID12		Odd Cycle Sync Frame ID Register 12	4.5.11
OSID13		Odd Cycle Sync Frame ID Register 13	4.5.11
OSID14		Odd Cycle Sync Frame ID Register 14	4.5.11
OSID15		Odd Cycle Sync Frame ID Register 15	4.5.11
NMV1		Network Management Register 1	4.5.12
NMV2		Network Management Register 2	4.5.12
NMV3		Network Management Register 3	4.5.12
MRC	Message Buffer Control Registers	Message RAM Configuration Register	4.6.1
FRF		FIFO Rejection Filter Register	4.6.2
FRFM		FIFO Rejection Filter Mask Register	4.6.3
FCL		FIFO Critical Level Register	4.6.4
MHDS	Message Buffer Status Registers	Message Handler Status Register	4.7.1
LDS		Last Dynamic Transmission Slot Register	4.7.2
FSR		FIFO Status Register	4.7.3
MHDF		Message Handler Constraints Flags	4.7.4
TXRQ1		Transmission Request Register 1	4.7.5
TXRQ2		Transmission Request Register 2	4.7.5
TXRQ3		Transmission Request Register 3	4.7.5
TXRQ4		Transmission Request Register 4	4.7.5
NDAT1		New Data Register 1	4.7.6
NDAT2		New Data Register 2	4.7.6
NDAT3		New Data Register 3	4.7.6
NDAT4		New Data Register 4	4.7.6
MBSC1		Message Buffer Status Changed Register 1	4.7.7
MBSC2		Message Buffer Status Changed Register 2	4.7.7
MBSC3		Message Buffer Status Changed Register 3	4.7.7
MBSC4		Message Buffer Status Changed Register 4	4.7.7
CREL	Identification Registers	Core Release Register	4.8.1
ENDN		Endian Register	4.8.2
WRDSn	Input Buffer	Write Data Section Register [1 to 64]	4.9.1
WRHS1		Write Header Section Register 1	4.9.2
WRHS2		Write Header Section Register 2	4.9.3
WRHS3		Write Header Section Register 3	4.9.4
IBCM		Input Buffer Command Mask Register	4.9.5
IBCR		Input Buffer Command Request Register	4.9.6

Abbreviated Register Name	Register Name		See
RDDSn	Output Buffer	Read Data Section Register [1 to 64]	4.10.1
RDHS1		Read Header Section Register 1	4.10.2
RDHS2		Read Header Section Register 2	4.10.3
RDHS3		Read Header Section Register 3	4.10.4
MBS		Message Buffer Status Register	4.10.5
OBCM		Output Buffer Command Mask Register	4.10.6
OBCR		Output Buffer Command Request Register	4.10.7



## 4.1. Customer Registers

These registers are assigned to the version information and FlexRay control (DMA support, interrupt register, FlexRay reset, and buffer data SWAP).

### 4.1.1. Version Information Register (CIF0)

BITS	31	30	29	28	27	26	25	24
BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	VERSION							
ACCESS_TYPE	R,WX							
PROT_TYPE	-							
INITIAL_VALUE	00000100							

BITS	23	22	21	20	19	18	17	16
BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	VERSION							
ACCESS_TYPE	R,WX							
PROT_TYPE	-							
INITIAL_VALUE	11111111							

BITS	15	14	13	12	11	10	9	8
BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	VERSION							
ACCESS_TYPE	R,WX							
PROT_TYPE	-							
INITIAL_VALUE	01110000							

BITS	7	6	5	4	3	2	1	0
BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	VERSION							
ACCESS_TYPE	R,WX							
PROT_TYPE	-							
INITIAL_VALUE	11111111							

#### [bit31:24] Manufacturer ID code bits

These bits have the Manufacturer ID code set. 0x04 is read. Writing to these bits has no effect.

#### [bit23:16] Version bits

These bits indicate the LSI version. 0xFF is read. Writing to these bits has no effect.

#### [bit15:8] LSI identification number bits

These bits indicate the LSI identification number. Three-digit numbers of bottom of the part number are shown in hexadecimal. 0x7B is read. Writing to these bits has no effect.

#### [bit7:0] FlexRay IP identification bits

These bits indicate the FlexRay IP identification number. 0xFF is read. When these bits indicate 0xFF, read the CREL register if necessary, since the register contains IP information. Writing to these bits has no effect.

### 4.1.2. Control Register (CIF1)

BITS_OFFSET	31	30	29	28	27	26	25	24
BITS_NAME	Reserved	DLVLO	DMODO	DENBO	Reserved	DLVLI	DMODI	DENBI
ACCESS_TYPE	R0,W0	R/W	R/W	R/W	R0,W0	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BITS_OFFSET	23	22	21	20	19	18	17	16
BITS_NAME	Reserved							
ACCESS_TYPE	R0,W0							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved	RESET		SWAP	Reserved	TENB1	Reserved	TENB0
ACCESS_TYPE	R0,W0	R0,W		R/W	R0,W0	R/W	R0,W0	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	00		0	0	0	0	0

BITS_OFFSET	7	6	5	4	3	2	1	0
BITS_NAME	Reserved							
ACCESS_TYPE	R0,W0							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

**[bit31] Reserved: Reserved bit**

#### **[bit30] DLVLO: DMA level/edge selection bit for output buffer**

The DREQO register indicates the following when register DMODO is 0:

When this bit is set to "0": DREQO indicates output buffer busy.

When this bit is set to "1": DREQO indicates inversion of output buffer busy.

Edge detection method for output buffer busy when register DMODO is 1:

When this bit is set to "0": Falling edge detection of output buffer busy

When this bit is set to "1": Rising edge detection of output buffer busy

Value	Description	
	DMODO=0	DMODO=1
0	DMA request level is output buffer busy	DMA request is falling edge of output buffer busy
1	DMA request level is inversion of output buffer busy	DMA request is rising edge of output buffer busy

#### **Note:**

- The result of edge detection of output buffer busy that corresponds to the setting of DLVLO ("1" when the edge is detected) is retained. The edge detection result cannot be referenced while DMODO is "0".



**[bit29] DMOD0: DMA trigger mode selection bit for output buffer**

This bit selects information output to the DREQO register and a DMA request.

When this bit is set to "0": Output buffer busy is output.

When this bit is set to "1": State of edge detection of output buffer busy is output.

Value	Description
0	Busy level of output buffer
1	Busy edge of output buffer

**Notes:**

- DMOD0 is a selector that selects information output to the DREQO register and a DMA request.
- The function to disable edge detection is not provided. Edge detection is always enabled.

**[bit28] DENBO: DMA request output enable bit for output buffer**

When this bit is set to "0": DMA request output disabled.

When this bit is set to "1": DMA request output enabled.

Value	Description
0	Disabled
1	Enabled

**[bit27] Reserved: Reserved bit**

**[bit26] DLVLI: DMA level/edge selection bit for input buffer host**

The DREQI register indicates the following when register DMODI is 0:

When this bit is set to "0": DREQI indicates input buffer host busy.

When this bit is set to "1": DREQI indicates inversion of input buffer host busy.

Edge detection method for input buffer host busy when register DMODI is 1:

When this bit is set to "0": Falling edge detection of input buffer host busy

When this bit is set to "1": Rising edge detection of input buffer host busy

Value	Description	
	DMODI=0	DMODI=1
0	DMA request level is input buffer host busy	DMA request is falling edge of input buffer host busy
1	DMA request level is inversion of input buffer host busy	DMA request is rising edge of input buffer host busy

**Note:**

- The result of detecting the edge of input buffer host busy that corresponds to the setting of DLVLI ("1" when the edge is detected) is retained. The edge detection result cannot be referenced while DMODI is "0".

**[bit25] DMODI: DMA trigger mode selection bit for input buffer host**

This bit selects information output to the DREQI register and a DMA request.

When this bit is set to "0": Input buffer host busy is output.

When this bit is set to "1": State of edge detection of input buffer host busy is output.

Value	Description
0	Busy level of input buffer host
1	Busy edge of input buffer host

**Note:**

- *DMODI is a selector that selects information output to the DREQI register and a DMA request. The function to disable edge detection is not provided. Edge detection is always enabled.*

**[bit24] DENBI: DMA request output enable bit for input buffer host**

When this bit is set to "0": DMA request output disabled.

When this bit is set to "1": DMA request output enabled.

Value	Description
0	Disabled
1	Enabled

**[bit23:16] Reserved: Reserved bits**

**[bit15] Reserved: Reserved bit**

**[bit14:13] RESET[1:0]: FlexRay reset bits**

Key codes are supported.

If "0b00", "0b01", "0b10", and "0b11" are written consecutively to the reset register, reset is output to the FlexRay macro.

**Notes:**

- *If read access or writing to an address that does not include a reset bit is performed in the middle of key code writing, the key code writing is considered suspended. To perform reset again, write the values from "0b00" again.*
- *Be careful when writing a value to a bit other than reset because no bit-masking function is provided. For bits for which the setting value needs to be retained, write the last value. However, TREQ1, TREQ0, DREQ0, and DREQI clear interrupts when "0" is written. Therefore, if there is no need to clear interrupts, writing "1" should be performed.*
- *When a key code is written before a DMA transfer that targets FlexRay address space is completed, key code writing is suspended by DMA that is started during the key code writing.*
- *When "00" is written, it is always determined to be the start of a key code. If key code writing is suspended by writing "00", it is determined to be the start of a new key code writing at the same time.*
- *Example)*

Start                      Suspend & Start                      Output reset  
↓                                      ↓                                      ↓  
"00" -> "01" -> "00" -> "01" -> "10" -> "11"



**[bit12] SWAP: Buffer data SWAP enable bit**

This bit selects byte swapping.

The following settings are reflected in the RAM area:

When this bit is set to "0": Swapping disabled ([31:24] [23:16] [15:8] [7:0])

When this bit is set to "1": Swapping enabled ([7:0] [15:8] [23:16] [31:24])

This bit has no effect on register access.

Value	Description
0	SWAP off
1	SWAP on

**[bit11] Reserved: Reserved bit**

**[bit10] TENB1: Timer 1 interrupt enable bit**

When this bit is set to "0": Interrupt disabled.

When this bit is set to "1": Interrupt enabled.

Value	Description
0	Output mask (fix to "0")
1	Select the rising edge of timer 1

**[bit9] Reserved: Reserved bit**

**[bit8] TENB0: Timer 0 interrupt enable bit**

When this bit is set to "0": Interrupt disabled.

When this bit is set to "1": Interrupt enabled.

Value	Description
0	Output mask (fix to "0")
1	Select the rising edge of timer 0

**[bit7:0] Reserved: Reserved bits**

### 4.1.3. Flag Register (CIF1F)

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,W0
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved				DREQO	DREQI	TREQ1	TREQ0
ACCESS_TYPE	R0,W0				R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0000				0	0	0	0

**[bit31:4] Reserved: Reserved bits**

#### **[bit3] DREQO: DMA request flag bit for output buffer**

Reading this bit indicates whether there is a DMA request.

DMODO = 0: Level of output buffer busy displayed (inverted by DLVLO).

DMODO = 1: DMA request displayed where edge detection of output buffer busy is performed (level output).

Writing to this bit has no effect. Flag bit cleared in CIF1C register.

Value	Description	
	Read	Write
0	No DMA request	No effect
1	DMA request	

#### **Notes:**

- Writing "1" to DREQOC to clear a DMA request is only effective for the DMA request by edge detection.
- The DMA request by edge detection is also cleared to "0" when DMA transfer occurs.
- The state of "1" for the DMA request by edge detection is maintained until it is cleared. It needs to be cleared before edge detection.
- The DMA request by edge detection is retained. When DMODO is set to 1, a DMA request occurs based on the retained data. If DMODO is changed from 0 to 1, the edge detection result detected while DMODO is 0 is output as is when DMODO is changed to 1. If you want edge detection obtained after DMODO is changed to 1, a clear must be performed before DMODO is changed.
- If an edge detection for a DMA request and a DREQOC clear occur at the same time, the DMA request takes precedence and the clear is ignored. To clear DREQO, the clear operation must be performed for DREQOC again.
- When DENBO is 1, the content read from DREQO is the same as the content of the DMA request.





**[bit2] DREQI: DMA request flag bit for input buffer host**

Reading this bit indicates whether there is a DMA request.

DMODI = 0: Level of input buffer host busy indicated (inverted by DLVLI).

DMODI = 1: DMA request indicated where edge detection of input buffer host busy is performed (level output).

Writing to this bit has no effect. Flag bit cleared in CIF1C register.

Value	Description	
	Read	Write
0	No DMA request	No effect
1	DMA request	

**Notes:**

- Writing "0" to DREQIC to clear a DMA request is only effective for the DMA request by edge detection.
- The DMA request by edge detection is also cleared to "0" when DMA transfer occurs.
- The state of "1" for the DMA request by edge detection is maintained until it is cleared. It needs to be cleared before edge detection.
- The DMA request by edge detection is retained. When DMODI is set to 1, a DMA request occurs based on the retained data. If DMODI is changed from 0 to 1, the edge detection result detected while DMODI is 0 is output as is when DMODI is changed to 1. If you want edge detection obtained after DMODI is changed to 1, a clear must be performed before DMODI is changed.
- If an edge detection for a DMA request and a DREQIC clear occur at the same time, the DMA request takes precedence and the clear is ignored. To clear DREQI, the clear operation must be performed for DREQIC again.
- When DENBI is 1, the content read from DREQI is the same as the content of the DMA request.

**[bit1] TREQ1: Timer 1 interrupt request bit**

When this bit is read, it indicates a TREQ1 interrupt request.

TENB0 = 0: Level of timer 1 indicated.

TENB0 = 1: "1" is indicated by rising edge detection of timer 1 (level output).

Writing to this bit has no effect. Flag bit cleared in CIF1C register.

Value	Description	
	Read	Write
0	No timer 1 interrupt request	No effect
1	Timer 1 interrupt request	

**Notes:**

- The interrupt request by edge detection is retained. When TENB1 is set to 1, an interrupt request occurs based on the retained data. If TENB1 is changed from 0 to 1, the edge detection result detected while TENB1 is 0 is output as is when TENB1 is changed to 1. If you want edge detection obtained after TENB1 is changed to 1, a clear must be performed before TENB1 is changed.
- If a TREQ1 interrupt request and a TREQ1C clear occur at the same time, the clear is ignored. To clear TREQ1, the clear operation must be performed for TREQ1C again.

**[bit0] TREQ0: Timer 0 interrupt request bit**

When this bit is read, it indicates a TREQ0 interrupt request.

TENB1 = 0: Level of timer 0 indicated.

TENB1 = 1: "1" is indicated by rising edge detection of timer 1 (level output).

Writing to this bit has no effect. Flag bit cleared in CIF1C register.

Value	Description	
	Read	Write
0	No timer 0 interrupt request	No effect
1	Timer 0 interrupt request	

**Notes:**

- The interrupt request by edge detection is retained. When TENB0 is set to 1, an interrupt request occurs based on the retained data. If TENB0 is changed from 0 to 1, the edge detection result detected while TENB0 is 0 is output as is when TENB0 is changed to 1. If you want edge detection obtained after TENB0 is changed to 1, a clear must be performed before TENB0 is changed.
- If a TREQ0 interrupt request and a TREQ0C clear occur at the same time, the clear is ignored. To clear TREQ0, the clear operation must be performed for TREQ0C again.



#### 4.1.4. Flag Clear Register (CIF1C)

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,W0
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved				DREQOC	DREQIC	TREQ1C	TREQ0C
ACCESS_TYPE	R0,W0				R0,W	R0,W	R0,W	R0,W
PROT_TYPE	-							
INITIAL_VALUE	0000				0	0	0	0

**[bit31:4] Reserved: Reserved bits**

##### **[bit3] DREQOC: DMA request clear bit for output buffer**

Reading this bit always returns "0". Read the status of flag in CIF1F register.

Writing to this bit clears the DMA request by edge detection.

When this bit is set to "0": No effect

When this bit is set to "1": DMA request cleared

Value	Description
0	No effect
1	Clear DMA request

##### **Notes:**

- If there is no need to clear the DMA request by edge detection, write "0" during the write operation.
- Writing "1" to DREQOC to clear a DMA request is only effective for the DMA request by edge detection.
- The DMA request by edge detection is also cleared to "0" when DMA transfer occurs.
- The state of "1" for the DMA request by edge detection is maintained until it is cleared. It needs to be cleared before edge detection.
- The DMA request by edge detection is retained. When DMODO is set to 1, a DMA request occurs based on the retained data. If DMODO is changed from 0 to 1, the edge detection result detected while DMODO is 0 is output as is when DMODO is changed to 1. If you want edge detection obtained after DMODO is changed to 1, a clear must be performed before DMODO is changed.
- If an edge detection for a DMA request and a DREQOC clear occur at the same time, the DMA request takes precedence and the clear is ignored. To clear DREQO, the clear operation must be performed for DREQOC again.

**[bit2] DREQIC: DMA request clear bit for input buffer host**

Reading this bit always returns "0". Read the status of flag in CIF1F register.

Writing to this bit clears the DMA request by edge detection.

When this bit is set to "0": No effect

When this bit is set to "1": DMA request cleared

Value	Description
0	No effect
1	Clear DMA request

**Notes:**

- If there is no need to clear the DMA request by edge detection, write "0" during write operation.
- Writing "1" to DREQIC to clear a DMA request is only effective for the DMA request by edge detection.
- The DMA request by edge detection is also cleared to "0" when DMA transfer occurs.
- The state of "1" for the DMA request by edge detection is maintained until it is cleared. It needs to be cleared before edge detection.
- The DMA request by edge detection is retained. When DMODI is set to 1, a DMA request occurs based on the retained data. If DMODI is changed from 0 to 1, the edge detection result detected while DMODI is 0 is output as is when DMODI is changed to 1. If you want edge detection obtained after DMODI is changed to 1, a clear must be performed before DMODI is changed.
- If an edge detection for a DMA request and a DREQIC clear occur at the same time, the DMA request takes precedence and the clear is ignored. To clear DREQI, the clear operation must be performed for DREQIC again.



**[bit1] TREQ1C: Timer 1 interrupt request clear bit**

Reading this bit always returns "0". Read the status of flag in CIF1F register.

Writing to this bit clears the TREQ1 interrupt request.

When this bit is set to "0": No effect

When this bit is set to "1": Timer interrupt request cleared

Value	Description
0	No effect
1	Clear timer 1 interrupt request

**Notes:**

- The interrupt request by edge detection is retained. When TENB1 is set to 1, an interrupt request occurs based on the retained data. If TENB1 is changed from 0 to 1, the edge detection result detected while TENB1 is 0 is output as is when TENB1 is changed to 1. If you want edge detection obtained after TENB1 is changed to 1, a clear must be performed before TENB1 is changed.
- If a TREQ1 interrupt request and a TREQ1C clear occur at the same time, the clear is ignored. To clear TREQ1, the clear operation must be performed for TREQ1C again.

**[bit0] TREQ0C: Timer 0 interrupt request clear bit**

Reading this bit always returns "0". Read the status of flag in CIF1F register.

Writing to this bit clears the TREQ0 interrupt request.

When this bit is set to "0": Timer interrupt request cleared

When this bit is set to "1": No effect

Value	Description
0	No effect
1	Clear timer 0 interrupt request

**Notes:**

- The interrupt request by edge detection is retained. When TENB0 is set to 1, an interrupt request occurs based on the retained data. If TENB0 is changed from 0 to 1, the edge detection result detected while TENB0 is 0 is output as is when TENB0 is changed to 1. If you want edge detection obtained after TENB0 is changed to 1, a clear must be performed before TENB0 is changed.
- If a TREQ0 interrupt request and a TREQ0C clear occur at the same time, the clear is ignored. To clear TREQ0, the clear operation must be performed for TREQ0C again.

## 4.2. Special Register

### 4.2.1. Lock Register (LCK)

The lock register is write-only. Reading the register returns 0x00000000.

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,W0
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	CLK							
ACCESS_TYPE	R0,W							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

**[bit31:8] Reserved: Reserved bits**

#### **[bit7:0] CLK[7:0]: Configuration lock key bits**

Before exiting the CONFIG state by writing SUCC1:CMD[3:0]=0010 (READY command), 2 consecutive write operations (unlock sequence) must be performed on CLK [7:0]. The writing procedure is described below. If a different write access is made during the write access procedure, the procedure below must be repeated because these bits are still in the CONFIG state.

First writing: LCK:CLK[7:0]=1100 1110 (0xCE)

Second writing: LCK:CLK[7:0]=0011 0001 (0x31)

Third writing: SUCC1:CMD[3:0]=0010 (CHI command READY)

**Note:**

- The host uses 32-bit access for read from/write to all bit fields.



### 4.3. Interrupt-related Registers

#### 4.3.1. Error Interrupt Register (EIR)

When any of the errors described below is detected, the flag corresponding to the error is set to "1". The flag is cleared to "0" by writing "1" to the corresponding bit. Until then, it maintains the set value. Writing "0" has no effect. When a hard reset is performed, this register is cleared to "0".

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved					TABB	LTVB	EDB
ACCESS_TYPE	R0,W0					R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	00000					0	0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved					TABA	LTVA	EDA
ACCESS_TYPE	R0,W0					R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	00000					0	0	0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved				MHF	IOBA	IIBA	EFA
ACCESS_TYPE	R0,W0				R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0000				0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	RFO	PERR	CCL	CCF	SFO	SFBM	CNA	PEMC
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:27] Reserved: Reserved bits

[bit26] TABB: Transmission across channel B slot boundary detection flag (Transmission Across Boundary Channel B) bit

This bit reports that transmission across the slot boundary occurred on channel B.

Value	Description
0	No transmission across slot boundary detected on channel B
1	Transmission across slot boundary detected on channel B

**[bit25] LTVB: Channel B transmission failure detection flag (Latest Transmit Violation Channel B) bit**

This bit indicates the latest transmission failure detection on channel B.

Value	Description
0	No transmission failure detected on channel B
1	Latest transmission failure detected on channel B

**[bit24] EDB: Error detected on channel B flag (Error Detected on Channel B) bit**

When any of ACS:SEDB, ACS:CEDB, ACS:CIB, or ACS:SBVB is changed from "0" to "1", this bit is set to "1".

Value	Description
0	No error detected on channel B
1	Error detected on channel B

**[bit23:19] Reserved: Reserved bits**

**[bit18] TABA: Transmission across channel A slot boundary detection flag (Transmission Across Boundary Channel A) bit**

This bit reports that transmission across the slot boundary occurred on channel A.

Value	Description
0	No transmission across slot boundary detected on channel A
1	Transmission across slot boundary detected on channel A

**[bit17] LTVA: Channel A transmission failure detection flag (Latest Transmit Violation Channel A) bit**

This bit indicates the latest transmission failure on channel A.

Value	Description
0	No transmission failure detected on channel A
1	Latest transmission failure detected on channel A

**[bit16] EDA: Error detected on channel A flag (Error Detected on Channel A) bit**

When any of the ACS:SEDA, ACS:CEDA, ACS:CIA, or ACS:SBVA changes from "0" to "1", this bit is set to "1".

Value	Description
0	No error detected on channel A
1	Error detected on channel A

**[bit15:12] Reserved: Reserved bits**





**[bit11] MHF: Message handler constraints flag bit**

This flag indicates the constraint state of the message handler. The flag is set when any of the MHDF:SNUA, MHDF:SNUB, MHDF:FNFA, MHDF:FNFB, MHDF:TBFA, MHDF:TBFB, or MHDF:WAHP flag changes from "0" to "1".

Value	Description
0	No message handler failure detected
1	Message handler failure detected

**[bit10] IOBA: Illegal output buffer access flag (Illegal Output buffer Access) bit**

When the host requests a message buffer transfer from the message RAM to the output buffer while OBCR:OBSYS is "1", this bit is set to "1".

Value	Description
0	No illegal host access to the output buffer occurred
1	Illegal host access to the output buffer occurred

**[bit9] IIBA: Illegal input buffer access flag (Illegal Input Buffer Access) bit**

When the host requests a change of the message buffer through the input buffer, and when the following conditions occur, this bit is set to "1":

- 1) When the host writes data to the input buffer command request register to make any of the following changes in a state other than CONFIG or DEFAULT\_CONFIG
  - In cases where message buffer 0 is configured for key slot transmission (startup frame/synchronization frame transmission, or frame transmission in SINGLE slot mode), change of the header section of the buffer
  - During MRC:SEC[1:0] = "01", change of the header section of a static message buffer whose buffer number is smaller than MRC:FDB[7:0]
  - During MRC:SEC[1:0] = "1x", change of the header section of a static/dynamic message buffer
  - Change of the header section or data section of a message buffer that belongs to the reception FIFO
- 2) When the host writes data to a register of the input buffer while IBCR:IBSYH is set to "1"

Value	Description
0	No illegal host access to input buffer occurred
1	Illegal host access to input buffer occurred

**[bit8] EFA: Empty FIFO access flag (Empty FIFO Access) bit**

If the host requests a message transfer from the reception FIFO through the output buffer when the reception FIFO is empty, this bit is set to "1".

Value	Description
0	No host access occurred when reception FIFO was empty
1	Host access occurred when reception FIFO was empty

**[bit7] RFO: Reception FIFO overrun flag (Receive FIFO Overrun) bit**

If a reception FIFO overrun is detected, this bit is set to "1". This flag is cleared when the reception FIFO is read.

Value	Description
0	No reception FIFO overrun detected
1	Reception FIFO overrun detected

**[bit6] PERR: Parity error flag (Parity Error) bit**

This bit reports a parity error. If a parity error is detected while reading one of the RAM blocks of the FlexRay controller, this flag is set to "1". When the parity error flag in the MHDS register is cleared to "0", this flag is cleared to "0". See Section "4.7.1. Message Handler Status Register (MHDS)".

Value	Description
0	No parity error detected
1	Parity error detected

**[bit5] CCL: CHI command locked flag (CHI Command Locked) bit**

This bit indicates that SUCC1:CMD[3:0] was reset to "0000" because the execution of the previous CHI command has not been completed. In this case, the CNA bit is also set to "1".

Value	Description
0	CHI command accepted
1	CHI command not accepted

**[bit4] CCF: Clock correction failure flag (Clock Correction Failure) bit**

When any of the following errors occurs, this bit is set to "1" at the end of a cycle:

- Rate correction lost.
- Offset correction lost.
- Clock correction limit exceeded.

The clock correction status can be monitored by the CCEV and SFS registers. This flag may be set during startup. So, clear the flag after state transition to the NORMAL\_ACTIVE state occurs.

Value	Description
0	No clock correction error occurred
1	Clock correction failed



**[bit3] SFO: Synchronization frame overflow flag (Sync Frame Overflow) bit**

If either the number of synchronization frames received during the previous communication cycle or the number of different synchronization frame IDs received during a double cycle (even/odd) exceeds the maximum number of synchronization frames defined by GTUC2:SNM[3:0], this bit is set to "1".

Value	Description
0	Number of received synchronization frames is equal to or less than setting value of GTUC2:SNM[3:0]
1	Number of received synchronization frames is greater than setting value of GTUC2:SNM[3:0]

**[bit2] SFBM: Synchronization frames below minimum flag (Sync Frames Below Minimum) bit**

If the number of synchronization frames received during the previous communication cycle is below the minimum value required by the FlexRay protocol, this bit is set to "1". This flag may be set during startup. So, clear the flag after state transition to the NORMAL\_ACTIVE state occurs.

Value	Description
0	Synchronous node: 1 or more synchronization frames received Asynchronous node: 2 or more synchronization frames received
1	Number of received synchronization frames is below required minimum value

**[bit1] CNA: Invalid command notification flag (Command Not Accepted) bit**

This bit indicates that SUCC1:CMD[3:0] was reset to "0000" because the requested command could not be used in the current POC state or the CHI command was locked (CCL="1").

Value	Description
0	CHI command accepted
1	CHI command not accepted

**[bit0] PEMC: POC error mode changed flag (POC Error Mode Changed) bit**

When the error mode indicated by CCEV:ERRM[1:0] is changed, this bit is set to "1".

Value	Description
0	Error mode not changed
1	Error mode changed

### 4.3.2. Status Interrupt Register (SIR)

When any of the events described below is detected, the flag corresponding to the event is set to "1". The flag is cleared by writing "1" to the corresponding bit. Until then, it maintains the set value. Writing "0" has no effect. When a hard reset is performed, this register is cleared.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved						MTSB	WUPB
ACCESS_TYPE	R0,W0						R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	000000						0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved						MTSA	WUPA
ACCESS_TYPE	R0,W0						R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	000000						0	0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	SDS	MBSI	SUCS	SWE	TOBC	TIBC	TI1	TI0
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	NMVC	RFCL	RFNE	RXI	TXI	CYCS	CAS	WST
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

**[bit31:26] Reserved: Reserved bits**

**[bit25] MTSB: MTS received on channel B flag (MTS Received on Channel B) bit (vSS!ValidMTSB)**

This bit indicates that a media access test symbol (MTS) was received on channel B during the last symbol window. This bit is updated at the end of a symbol window.

Value	Description
0	No MTS symbol received on channel B
1	MTS symbol received on channel B

**[bit24] WUPB: Channel B wakeup pattern reception flag (Wakeup Pattern Channel B) bit**

If CC is in the wakeup, ready, or startup state, or in monitor mode, this bit is set to "1" when a wakeup pattern is received on channel B.

Value	Description
0	No wakeup pattern received on channel B
1	Wakeup pattern received on channel B



**[bit23:18] Reserved: Reserved bits**

**[bit17] MTSA: MTS received on channel A flag (MTS Received on Channel A) bit (vSSIValidMTSA)**

This bit indicates that a media access test symbol (MTS) was received on channel A during the last symbol window. This bit is updated at the end of a symbol window.

Value	Description
0	No MTS symbol received on channel A
1	MTS symbol received on channel A

**[bit16] WUPA: Channel A wakeup pattern reception flag (Wakeup Pattern Channel A) bit**

If CC is in the wakeup, ready, or startup state, or in monitor mode, this bit is set to "1" when a wakeup pattern is received on channel A.

Value	Description
0	No wakeup pattern received on channel A
1	Wakeup pattern received on channel A

**[bit15] SDS: Start of dynamic segment flag (Start of Dynamic Segment) bit**

When the dynamic segment is started, this bit is set to "1".

Value	Description
0	Dynamic segment not started
1	Dynamic segment started

**[bit14] MBSI: Message buffer status changed flag (Message Buffer Status Interrupt) bit**

When the MBI bit of a message buffer is set to "1", and the message buffer status (MBS) is changed (See Figure 3-15), this bit is set to "1".

Value	Description
0	Status of a message buffer whose MBI is set to "1" is not changed
1	Status of at least one message buffer whose MBI is set to "1" is changed

**[bit13] SUCS: Startup completed successfully flag (Startup Completed Successfully) bit**

When startup is completed successfully and the NORMAL\_ACTIVE state is set, this bit is set to "1".

Value	Description
0	Startup not completed successfully
1	Startup completed successfully

**[bit12] SWE: Stop watch event flag (Stop Watch Event) bit**

After the stop watch starts, the cycle counter and the macrotick value are stored in the stop watch register. (See Section "4.3.10. Stop Watch Register 1 (STPW1)")

Value	Description
0	No stop watch event occurred
1	Stop watch event occurred

**[bit11] TOBC: Transfer output buffer completed flag (Transfer Output Buffer Completed) bit**

When transfer from the message RAM to the output buffer is completed, and OBCR:OBSYS is reset, this bit is set to "1".

Value	Description
0	Transfer between message RAM and output buffer not completed
1	Transfer between message RAM and output buffer completed

**[bit10] TIBC: Transfer input buffer completed flag (Transfer Input Buffer Completed) bit**

When transfer from the input buffer to the message RAM is completed and IBCR:IBSYS is reset, this bit is set to "1".

Value	Description
0	Transfer between input buffer and message RAM not completed
1	Transfer between input buffer and message RAM completed

**[bit9] T1I: Timer 1 flag (Timer Interrupt 1) bit**

When the value of timer 1 matches the value of T1C, this bit is set to "1".

Value	Description
0	Value of timer 1 does not match value of T1C
1	Value of timer 1 matches value of T1C

**[bit8] T0I: Timer 0 flag (Timer Interrupt 0) bit**

When the value of timer 0 matches the value of T0C, this bit is set to "1".

Value	Description
0	Value of timer 0 does not match value of T0C
1	Value of timer 0 matches value of T0C

**[bit7] NMVC: Network management vector changed flag (Network Management Vector Changed) bit**

This bit indicates whether the network management vector is changed.

Value	Description
0	Network management vector not changed
1	Network management vector changed

**[bit6] RFCL: Reception FIFO full flag (Receive FIFO Critical Level) bit**

When the reception FIFO level (FSR:RFFL[7:0]) is equal to or above the critical level (FCL:CL[7:0]), this bit is set to "1".

Value	Description
0	Reception FIFO level is below the critical level
1	Reception FIFO level is equal to or above the critical level



**[bit5] RFNE: Reception FIFO flag (Receive FIFO Not Empty) bit**

When a valid frame is stored in the reception FIFO, this bit is set to "1".

Value	Description
0	Reception FIFO empty
1	Reception FIFO not empty

**[bit4] RXI: Reception completion flag (Receive Interrupt) bit**

If "1" is set in the MBI bit of each message buffer, this bit is set to "1" when the payload segment of the received valid frame is stored in the reception buffer. (See Figure 3-16)

Value	Description
0	No data section updated in reception buffer whose MBI bit is set to "1"
1	At least one of data sections in reception buffer whose MBI bit is set to "1" is updated

**[bit3] TXI: Transmission completion flag (Transmit Interrupt) bit**

If "1" is set in the MBI bit of each message buffer, this bit is set to "1" after frame transmission is successfully completed. (See Figure 3-16)

Value	Description
0	No frame transmitted from transmission buffer whose MBI bit is set to "1"
1	At least one frame successfully transmitted from transmission buffer whose MBI bit is set to "1"

**[bit2] CYCS: Communication cycle start flag (Cycle Start Interrupt) bit**

When a communication cycle starts, this bit is set to "1".

Value	Description
0	No communication cycle started
1	Communication cycle started

**[bit1] CAS: Collision avoidance symbol flag (Collision Avoidance Symbol) bit**

When CAS is received, this bit is set to "1".

Value	Description
0	No collision avoidance symbol received
1	Collision avoidance symbol received

**[bit0] WST: Wakeup status flag (Wakeup Status) bit**

When CCSV:WSV[2:0] changes to a value other than UNDEFINED, this flag is set to "1".

Value	Description
0	No wakeup status transition occurred
1	Wakeup status transition occurred

### 4.3.3. Error Interrupt Line Selection Register (EILS)

This register determines to which of the following interrupt lines an interrupt generated by the error interrupt flag in the EIR register should be assigned:

1 = Interrupt assigned to INT1 line.

0 = Interrupt assigned to INT0 line.

BITS	31	30	29	28	27	26	25	24
BIT_OFFSET								
BIT_NAME	Reserved					TABBL	LTVBL	EDBL
ACCESS_TYPE	R0,W0					R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	00000					0	0	0

BITS	23	22	21	20	19	18	17	16
BIT_OFFSET								
BIT_NAME	Reserved					TABAL	LTVAL	EDAL
ACCESS_TYPE	R0,W0					R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	00000					0	0	0

BITS	15	14	13	12	11	10	9	8
BIT_OFFSET								
BIT_NAME	Reserved				MHFL	IOBAL	IIBAL	EFAL
ACCESS_TYPE	R0,W0				R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0000				0	0	0	0

BITS	7	6	5	4	3	2	1	0
BIT_OFFSET								
BIT_NAME	RFOL	PERRL	CCLL	CCFL	SFOL	SFBML	CNAL	PEMCL
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:27] Reserved: Reserved bits

[bit26] TABBL: Transmission across channel B slot boundary detected interrupt line selection (Transmission Across Boundary Channel B Interrupt Line) bit

Value	Description
0	Interrupt assigned to INT0 line
1	Interrupt assigned to INT1 line

[bit25] LTVBL: Channel B transmission failure detected interrupt line selection (Latest Transmit Violation Channel B Interrupt Line) bit

Value	Description
0	Interrupt assigned to INT0 line
1	Interrupt assigned to INT1 line





**[bit24] EDBL: Error detected on channel B interrupt line selection (Error Detected on Channel B Interrupt Line) bit**

Value	Description
0	Interrupt assigned to INT0 line
1	Interrupt assigned to INT1 line

**[bit23:19] Reserved: Reserved bits**

**[bit18] TABAL: Transmission across channel A slot boundary detected interrupt line selection (Transmission Across Boundary Channel A Interrupt Line) bit**

Value	Description
0	Interrupt assigned to INT0 line
1	Interrupt assigned to INT1 line

**[bit17] LTVAL: Channel A transmission failure detected interrupt line selection (Latest Transmit Violation Channel A Interrupt Line) bit**

Value	Description
0	Interrupt assigned to INT0 line
1	Interrupt assigned to INT1 line

**[bit16] EDAL: Error detected on channel A interrupt line selection (Error Detected on Channel A Interrupt Line) bit**

Value	Description
0	Interrupt assigned to INT0 line
1	Interrupt assigned to INT1 line

**[bit15:12] Reserved: Reserved bits**

**[bit11] MHFL: Message handler constraints flag interrupt line selection (Message Handler Constraints Flag Interrupt Line) bit**

Value	Description
0	Interrupt assigned to INT0 line
1	Interrupt assigned to INT1 line

**[bit10] IOBAL: Illegal output buffer access interrupt line selection (Illegal Output Buffer Access Interrupt Line) bit**

Value	Description
0	Interrupt assigned to INT0 line
1	Interrupt assigned to INT1 line

**[bit9] IIBAL: Illegal input buffer access interrupt line selection (Illegal Input Buffer Access Interrupt Line) bit**

Value	Description
0	Interrupt assigned to INT0 line
1	Interrupt assigned to INT1 line

**[bit8] EFAL: Empty FIFO access interrupt line selection (Empty FIFO Access Interrupt Line) bit**

Value	Description
0	Interrupt assigned to INT0 line
1	Interrupt assigned to INT1 line

**[bit7] RFOL: Reception FIFO overrun interrupt line selection (Receive FIFO Overrun Interrupt Line) bit**

Value	Description
0	Interrupt assigned to INT0 line
1	Interrupt assigned to INT1 line

**[bit6] PERRL: Parity error interrupt line selection (Parity Error Interrupt Line) bit**

Value	Description
0	Interrupt assigned to INT0 line
1	Interrupt assigned to INT1 line

**[bit5] CCLLCHI: Command locked interrupt line selection (CHI Command Locked Interrupt Line) bit**

Value	Description
0	Interrupt assigned to INT0 line
1	Interrupt assigned to INT1 line

**[bit4] CCFL: Clock correction failure interrupt line selection (Clock Correction Failure Interrupt Line) bit**

Value	Description
0	Interrupt assigned to INT0 line
1	Interrupt assigned to INT1 line

**[bit3] SFOL: Sync frame overflow interrupt line selection (Sync Frame Overflow Interrupt Line) bit**

Value	Description
0	Interrupt assigned to INT0 line
1	Interrupt assigned to INT1 line



**[bit2] SFBML: Sync frames below minimum interrupt line selection (Sync Frames Below Minimum Interrupt Line) bit**

Value	Description
0	Interrupt assigned to INT0 line
1	Interrupt assigned to INT1 line

**[bit1] CNAL: Invalid command notification interrupt line selection (Command Not Accepted Interrupt Line) bit**

Value	Description
0	Interrupt assigned to INT0 line
1	Interrupt assigned to INT1 line

**[bit0] PEMCLPOC: Error mode changed interrupt line selection (POC Error Mode Changed Interrupt Line) bit**

Value	Description
0	Interrupt assigned to INT0 line
1	Interrupt assigned to INT1 line

#### 4.3.4. Status Interrupt Line Selection Register (SILS)

This register determines to which of the following interrupt lines an interrupt generated by the status interrupt flag in the SIR register should be assigned:

1 = Interrupt assigned to INT1 line.

0 = Interrupt assigned to INT0 line.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved						MTSBL	WUPBL
ACCESS_TYPE	R0,W0						R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	000000						1	1

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved						MTSAL	WUPAL
ACCESS_TYPE	R0,W0						R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	000000						1	1

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	SDSL	MBSIL	SUCSL	SWEL	TOBCL	TIBCL	TI1L	TI0L
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	1	1	1	1	1	1	1	1

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	NMVCL	RFCLL	RFNEL	RXIL	TXIL	CYCSL	CASL	WSTL
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	1	1	1	1	1	1	1	1

[bit31:26] Reserved: Reserved bits

[bit25] MTSBL: Channel B MTS received interrupt line selection (Media Access Test Symbol Channel B Interrupt Line) bit

Value	Description
0	Interrupt assigned to INT0 line
1	Interrupt assigned to INT1 line

[bit24] WUPBL: Channel B wakeup pattern received interrupt line selection (Wakeup Pattern Channel B Interrupt Line) bit

Value	Description
0	Interrupt assigned to INT0 line
1	Interrupt assigned to INT1 line



[bit23:18] Reserved: Reserved bits

[bit17] MTSAL: Channel A MTS received interrupt line selection (Media Access Test Symbol Channel A Interrupt Line) bit

Value	Description
0	Interrupt assigned to INT0 line
1	Interrupt assigned to INT1 line

[bit16] WUPAL: Channel A wakeup pattern received interrupt line selection (Wakeup Pattern Channel A Interrupt Line) bit

Value	Description
0	Interrupt assigned to INT0 line
1	Interrupt assigned to INT1 line

[bit15] SDSL: Start of dynamic segment interrupt line selection (Start of Dynamic Segment Interrupt Line) bit

bit	Description
0	Interrupt assigned to INT0 line
1	Interrupt assigned to INT1 line

[bit14] MBSIL: Message buffer status changed interrupt line selection (Message Buffer Status Interrupt Line) bit

Value	Description
0	Interrupt assigned to INT0 line
1	Interrupt assigned to INT1 line

[bit13] SUCSL: Startup completed successfully interrupt line selection (Startup Completed Successfully Interrupt Line) bit

Value	Description
0	Interrupt assigned to INT0 line
1	Interrupt assigned to INT1 line

[bit12] SWEL: Stop watch event interrupt line selection (Stop Watch Event Interrupt Line) bit

Value	Description
0	Interrupt assigned to INT0 line
1	Interrupt assigned to INT1 line

[bit11] TOBCL: Transfer output buffer completed interrupt line selection (Transfer Output Buffer Completed Interrupt Line) bit

Value	Description
0	Interrupt assigned to INT0 line
1	Interrupt assigned to INT1 line

**[bit10] TIBCL: Transfer input buffer completed interrupt line selection (Transfer Input Buffer Completed Interrupt Line) bit**

Value	Description
0	Interrupt assigned to INT0 line
1	Interrupt assigned to INT1 line

**[bit9] TI1L: Timer 1 interrupt line selection (Timer Interrupt 1 Line) bit**

Value	Description
0	Interrupt assigned to INT0 line
1	Interrupt assigned to INT1 line

**[bit8] TI0L: Timer 0 interrupt line selection (Timer Interrupt 0 Line) bit**

Value	Description
0	Interrupt assigned to INT0 line
1	Interrupt assigned to INT1 line

**[bit7] NMVCL: Network management vector changed interrupt line selection (Network Management Vector Changed Interrupt Line) bit**

Value	Description
0	Interrupt assigned to INT0 line
1	Interrupt assigned to INT1 line

**[bit6] RFCLL: Reception FIFO critical level interrupt line selection (Receive FIFO Critical Level Interrupt Line) bit**

Value	Description
0	Interrupt assigned to INT0 line
1	Interrupt assigned to INT1 line

**[bit5] RFNEL: Reception FIFO interrupt line selection (Receive FIFO Not Empty Interrupt Line) bit**

Value	Description
0	Interrupt assigned to INT0 line
1	Interrupt assigned to INT1 line

**[bit4] RXIL: Reception completed interrupt line selection (Receive Interrupt Line) bit**

Value	Description
0	Interrupt assigned to INT0 line
1	Interrupt assigned to INT1 line

**[bit3] TXIL: Transmission completed interrupt line selection (Transmit Interrupt Line) bit**

Value	Description
0	Interrupt assigned to INT0 line
1	Interrupt assigned to INT1 line



**[bit2] CYCSL: Communication cycle start interrupt line selection (Cycle Start Interrupt Line) bit**

Value	Description
0	Interrupt assigned to INT0 line
1	Interrupt assigned to INT1 line

**[bit1] CASL: Collision avoidance symbol interrupt line selection (Collision Avoidance Symbol Interrupt Line) bit**

Value	Description
0	Interrupt assigned to INT0 line
1	Interrupt assigned to INT1 line

**[bit0] WSTL: Wakeup status interrupt line selection (Wakeup Status Interrupt Line) bit**

Value	Description
0	Interrupt assigned to INT0 line
1	Interrupt assigned to INT1 line

### 4.3.5. Error Interrupt Enable Register (EIES, EIER (Error Interrupt Enable Set/Reset))

Setting this register determines which status change in the error interrupt register (EIR) generates an interrupt.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved					TABBE	LTVBE	EDBE
ACCESS_TYPE	R0,W0					R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	00000					0	0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved					TABAE	LTVAE	EDAE
ACCESS_TYPE	R0,W0					R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	00000					0	0	0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved				MHFE	IOBAE	IIBAE	EFAE
ACCESS_TYPE	R0,W0				R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0000				0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	RFOE	PERRE	CCLE	CCFE	SFOE	SFBME	CNAE	PEMCE
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

The interrupt enable flag is set to enabled by writing "1" to EIES register and set to disabled by writing "1" to EIER register. Writing "0" to either of the registers does not change the enable flag.

The same value is read from both registers.

Value	Description
0	Interrupts disabled
1	Interrupts enabled

**[bit31:27] Reserved: Reserved bits**

**[bit26] TABBE: Transmission across channel B slot boundary detected interrupt enable flag (Transmission Across Boundary Channel B Interrupt Enable) bit**

**[bit25] LTVBE: Channel B transmission failure detected interrupt enable flag (Latest Transmit Violation Channel B Interrupt Enable) bit**





**[bit24] EDBE: Error detected on channel B interrupt enable flag (Error Detected on Channel B Interrupt Enable) bit**

**[bit23:19] Reserved: Reserved bits**

**[bit18] TABAE: Transmission across channel A slot boundary detected interrupt enable flag (Transmission Across Boundary Channel A Interrupt Enable) bit**

**[bit17] LTVAE: Channel A transmission failure detected interrupt enable flag (Latest Transmit Violation Channel A Interrupt Enable) bit**

**[bit16] EDAE: Error detected on channel A interrupt enable flag (Error Detected on Channel A Interrupt Enable) bit**

**[bit15:12] Reserved: Reserved bits**

**[bit11] MHFE: Message handler constraints flag interrupt enable flag (Message Handler Constraints Flag Interrupt Enable) bit**

**[bit10] IOBAE: Illegal output buffer access interrupt enable flag (Illegal Output Buffer Access Interrupt Enable) bit**

**[bit9] IIBAE: Illegal input buffer access interrupt enable flag (Illegal Input Buffer Access Interrupt Enable) bit**

**[bit8] EFAE: Empty FIFO access interrupt enable flag (Empty FIFO Access Interrupt Enable) bit**

**[bit7] RFOE: Reception FIFO overrun interrupt enable flag (Receive FIFO Overrun Interrupt Enable) bit**

**[bit6] PERRE: Parity error interrupt enable flag (Parity Error Interrupt Enable) bit**

**[bit5] CCLECHI: Command locked interrupt enable flag (CHI Command Locked Interrupt Enable) bit**

**[bit4] CCFE: Clock correction failure interrupt enable flag (Clock Correction Failure Interrupt Enable) bit**

**[bit3] SFOE: Sync frame overflow interrupt enable flag (Sync Frame Overflow Interrupt Enable) bit**

**[bit2] SFBME: Sync frames below minimum interrupt enable flag (Sync Frames Below Minimum Interrupt Enable) bit**

**[bit1] CNAE: Invalid command notification interrupt enable flag (Command Not Accepted Interrupt Enable) bit**

**[bit0] PEMCEPOC: Error mode changed interrupt enable flag (POC Error Mode Changed Interrupt Enable) bit**



### 4.3.6. Status Interrupt Enable Register (SIES, SIER (Status Interrupt Enable Set/Reset))

Setting this register determines which status change in the status interrupt register (SIR) generates an interrupt.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved						MTSBE	WUPBE
ACCESS_TYPE	R0,W0						R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	000000						0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved						MTSAE	WUPAE
ACCESS_TYPE	R0,W0						R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	000000						0	0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	SDSE	MBSTE	SUCSE	SWEE	TOBCE	TIBCE	TI1E	TI0E
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	NMVCE	RFCE	RFNEE	RXIE	TXIE	CYCSE	CNSE	WSTE
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

The interrupt enable flag is set to enabled by writing "1" to SIES register and set to disabled by writing "1" to SIER register. Writing "0" to either of the registers does not change the enable flag.

The same value is read from both registers.

Value	Description
0	Interrupts disabled
1	Interrupts enabled

[bit31:26] Reserved: Reserved bits

[bit25] MTSBE: MTS received on channel B interrupt enable flag (MTS Received on Channel B Interrupt Enable) bit

[bit24] WUPBE: Channel B wakeup pattern received interrupt enable flag (Wakeup Pattern Channel B Interrupt Enable) bit

**[bit23:18] Reserved: Reserved bits**

**[bit17] MTSAE: MTS received on channel A interrupt enable flag (MTS Received on Channel A Interrupt Enable) bit**

**[bit16] WUPAE: Channel A wakeup pattern received interrupt enable flag (Wakeup Pattern Channel A Interrupt Enable) bit**

**[bit15] SDSE: Start of dynamic segment interrupt enable flag (Start of Dynamic Segment Interrupt Enable) bit**

**[bit14] MBSIE: Message buffer status changed interrupt enable flag (Message Buffer Status Interrupt Enable) bit**

**[bit13] SUCSE: Startup completed successfully interrupt enable flag (Startup Completed Successfully Interrupt Enable) bit**

**[bit12] SWEE: Stop watch event interrupt enable flag (Stop Watch Event Interrupt Enable) bit**

**[bit11] TOBCE: Transfer output buffer completed interrupt enable flag (Transfer Output Buffer Completed Interrupt Enable) bit**

**[bit10] TIBCE: Transfer input buffer completed interrupt enable flag (Transfer Input Buffer Completed Interrupt Enable) bit**

**[bit9] TI1E: Timer 1 interrupt enable flag (Timer Interrupt 1 Enable) bit**

**[bit8] TI0E: Timer 0 interrupt enable flag (Timer Interrupt 0 Enable) bit**

**[bit7] NMVCE: Network management vector changed interrupt enable flag (Network Management Vector Changed Interrupt Enable) bit**

**[bit6] RFCLE: Reception FIFO critical level interrupt enable flag (Receive FIFO Critical Level Interrupt Enable) bit**

**[bit5] RFNEE: Reception FIFO interrupt enable flag (Receive FIFO Not Empty Interrupt Enable) bit**



**[bit4] RXIE: Reception completed interrupt enable flag (Receive Interrupt Enable) bit**

**[bit3] TXIE: Transmission completed interrupt enable flag (Transmit Interrupt Enable) bit**

**[bit2] CYCSE: Communication cycle start interrupt enable flag (Cycle Start Interrupt Enable) bit**

**[bit1] CASE: Collision avoidance symbol interrupt enable flag (Collision Avoidance Symbol Interrupt Enable) bit**

**[bit0] WSTE: Wakeup status interrupt enable flag (Wakeup Status Interrupt Enable) bit**

### 4.3.7. Interrupt Line Enable Register (ILE)

There are 2 interrupt lines (INT0 and INT1) that can be enabled/disabled separately by setting the EINT0 bit and EINT1 bit to "1".

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,W0
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved						EINT1	EINT0
ACCESS_TYPE	R0,W0						R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	000000						0	0

[bit31:2] Reserved: Reserved bits

[bit1] EINT1: Interrupt line INT1 enable flag (Enable Interrupt Line 1) bit

Value	Description
0	Interrupt line (INT1) disabled
1	Interrupt line (INT1) enabled

[bit0] EINT0: Interrupt line INT0 enable flag (Enable Interrupt Line 0) bit

Value	Description
0	Interrupt line (INT0) disabled
1	Interrupt line (INT0) enabled



### 4.3.8. Timer 0 Configuration Register (T0C)

This register specifies the time when the timer 0 interrupt is generated, in cycle count units and macrotick units. When a timer 0 interrupt is generated, interrupt output INT2 is set to "1" for 1 macrotick and SIR:TIO is set to "1".

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved		T0MO[13:8]					
ACCESS_TYPE	R0,W0		R/W					
PROT_TYPE	-							
INITIAL_VALUE	00		000000					

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	T0MO[7:0]							
ACCESS_TYPE	R/W							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved	T0CC						
ACCESS_TYPE	R0,W0	R/W						
PROT_TYPE	-							
INITIAL_VALUE	0	0000000						

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved						T0MS	T0RC
ACCESS_TYPE	R0,W0						R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	000000						0	0

**[bit31:30] Reserved: Reserved bits**

**[bit29:16] T0MO[13:0]: Timer 0 macrotick offset configuration (Timer 0 Macrotick Offset) bits**

These bits set in macrotick units how much offset time should elapse from the beginning of each of the cycles set in a cycle set before a timer 0 interrupt is generated. The cycle set is set by T0CC.

**[bit15] Reserved: Reserved bit**

**[bit14:8] T0CC[6:0]: Timer 0 cycle code configuration (Timer 0 Cycle Code) bits**

These bits comprise the code that determines the cycle set used to generate a timer 0 interrupt. For details on configuring the cycle code, see Section "3.7.2. Cycle Counter Filtering".

**[bit7:2] Reserved: Reserved bits**

**[bit1] T0MS: Timer 0 mode select bit**

Value	Description
0	Single-shot mode
1	Continuous mode

**[bit0] T0RC: Timer 0 run control bit**

Value	Description
0	Timer 0 stopped
1	Timer 0 running

**Note:**

- *In the event of a state transition from the NORMAL\_ACTIVE or NORMAL\_PASSIVE state to a different state, or when timer 0 is stopped by clearing T0RC to "0", interrupt output INT2 immediately outputs "L".*  
*Timer 0 is obtained as a value equivalent to a macrotick counter value. A dedicated counter for timer 0 is not provided.*





### 4.3.9. Timer 1 Configuration Register (T1C)

When timer 1 reaches the specified macrotick count, a timer 1 interrupt is generated. In addition, interrupt output INT3 is set to "1" for 1 macrotick, and SIR:TI1 is set to "1".

When the POC is in the NORMAL\_ACTIVE or NORMAL\_PASSIVE state, timer 1 can operate. In other states, timer 1 stops its operation.

To reconfigure timer 1, the timer must be stopped by writing "0" to the T1RC bit.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved		T1MC[13:8]					
ACCESS_TYPE	R0,W0		R/W					
PROT_TYPE	-							
INITIAL_VALUE	00		000000					

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	T1MC[7:0]							
ACCESS_TYPE	R/W							
PROT_TYPE	-							
INITIAL_VALUE	00000010							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved							
ACCESS_TYPE	R0,W0							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved						T1MS	T1RC
ACCESS_TYPE	R0,W0						R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	000000						0	0

**[bit31:30] Reserved: Reserved bits**

**[bit29:16] T1MC[13:0]: Timer 1 macrotick count bits**

When timer 1 matches the set macrotick count, a timer 1 interrupt is generated.

Valid value: From 2 to 16383 MT (continuous mode)

From 1 to 16383 MT (single-shot mode)

**[bit15:2] Reserved: Reserved bits**

**[bit1] T1MS: Timer 1 mode select bit**

Value	Description
0	Single-shot mode
1	Continuous mode

**[bit0] T1RC: Timer 1 run control bit**

Value	Description
0	Timer 1 stopped
1	Timer 1 running

**Note:**

- In the event of a state transition from the *NORMAL\_ACTIVE* or *NORMAL\_PASSIVE* state to a different state, or when timer 0 is stopped by clearing *T1RC* to "0", interrupt output *INT3* immediately outputs "L".



#### 4.3.10. Stop Watch Register 1 (STPW1)

The stop watch is activated by rising/falling edge input to the STOPWT pin, generation of interrupt 0 or 1, or writing "1" to the SSWT bit by the host. Macro tick counter addition starts following the activation of the stop watch. Actual cycle counter and macro tick values are stored in this stop watch register (stop watch event). These values can be read from the host.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved		SMTV[13:8]					
ACCESS_TYPE	R0,W0		R,WX					
PROT_TYPE	-							
INITIAL_VALUE	00		000000					

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	SMTV[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved		SCCV					
ACCESS_TYPE	R0,W0		R,WX					
PROT_TYPE	-							
INITIAL_VALUE	00		000000					

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved	EINT1	EINT0	EETP	SSWT	EDGE	SWMS	ESWT
ACCESS_TYPE	R0,W0	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

**[bit31:30] Reserved: Reserved bits**

**[bit29:16] SMTV[13:0]: Stop watch event occurrence macro tick value (Stopped Macro tick Value) bits**

These bits indicate the macro tick counter value when a stop watch event occurs. Valid values are from 0 to 15999.

**[bit15:14] Reserved: Reserved bits**

**[bit13:8] SCCV[5:0]: Stop watch event occurrence cycle counter value (Stopped Cycle Counter Value) bits**

These bits indicate the cycle counter value when a stop watch event occurs. Valid values are from 0 to 63.

**[bit7] Reserved: Reserved bit**

**[bit6] EINT1: Enable interrupt 1 trigger bit**

When ESWT is 1, an event of interrupt 1 is used as the stop watch trigger.

bit	Description
0	Disable
1	Enable

**[bit5] EINT0: Enable interrupt 0 trigger bit**

When ESWT is 1, an event of interrupt 0 is used as the stop watch trigger.

bit	Description
0	Disable
1	Enable

**[bit4] EETP: Enable external trigger pin bit**

When ESWT is 1, an edge signal on the input pin STOPWT is used as the stop watch trigger.

bit	Description
0	Disable
1	Enable

**[bit3] SSWT: Software stop watch trigger bit**

When the host sets this bit to "1", the stop watch is activated. This bit is cleared to "0" after actual cycle counter and macrotick values are stored in the stop watch register.

Data can be written to this bit while ESWT is 0.

Value	Description
0	Software trigger cleared
1	Stop watch activated by software trigger

**[bit2] EDGE: Stop watch trigger edge select bit**

Value	Description
0	Falling edge
1	Rising edge

**[bit1] SWMS: Stop watch mode select bit**

Value	Description
0	Single
1	Continuous



**[bit0] ESWT: Enable stop watch trigger bit**

When the stop watch trigger is enabled, the stop watch is activated by an edge signal on the STOPWT input pin or an interrupt 0 or 1 signal (rising edge of INT0 or INT1). In single-shot mode, this bit is set to "0" after the running cycle counter and the macrotick value are stored in the stop watch register.

Value	Description
0	Disable
1	Enable

**Note:**

- Bits ESWT and SSWT cannot be set to 1 simultaneously. In this case, the write access is ignored, and both bits keep their previous values. Either the external stop watch trigger or software stop watch trigger may be used.

### 4.3.11. Stop Watch Register 2 (STPW2)

The stop watch counter values of channels A and B can be read from the host.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved					SSCVB[10:8]		
ACCESS_TYPE	R0,W0					R,WX		
PROT_TYPE	-							
INITIAL_VALUE	00000					000		

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	SSCVB[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved					SCCVA[10:8]		
ACCESS_TYPE	R0,W0					R,WX		
PROT_TYPE	-							
INITIAL_VALUE	00000					000		

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	SCCVA[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

**[bit31:27] Reserved: Reserved bits**

**[bit26:16] SSCVB[10:0]: Channel B stop watch counter value (Stop Watch Captured Slot Counter Value Channel B) bits**

These bits indicate the stop watch counter value of channel B when an event occurs (0 to 2047).

**[bit15:11] Reserved: Reserved bits**

**[bit10:0] SCCVA[10:0]: Channel A stop watch counter value (Stop Watch Captured Slot Counter Value Channel A) bits**

These bits indicate the stop watch counter value of channel A when an event occurs (0 to 2047).



## 4.4. Communication Controller (CC) Control Registers

This section describes the registers that control FlexRay communication controller (CC). The FlexRay Protocol Specifications requires that application settings be configured in the CONFIG state. Note that writing to the configuration registers is not locked in the DEFAULT\_CONFIG state.

When hard reset is input, the state transits to the DEFAULT\_CONFIG state, and each register is initialized. To change the state of the protocol operation controller (POC) from the DEFAULT\_CONFIG state to the CONFIG state, set CMD[3:0]=0001 (CHI command CONFIG). To change the state from the CONFIG state to the READY state, follow the procedure described in Section "4.2.1. Lock Register".

All bits with an asterisk(\*) can be updated in the DEFAULT\_CONFIG or CONFIG state.

#### 4.4.1. SUC Configuration Register 1 (SUCC1)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved				CCHB*	CCHA*	MTSB*	MTSA*
ACCESS_TYPE	R0,W0				R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0000				1	1	0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	HCSE*	TSM*	WUCS*	PTA*				
ACCESS_TYPE	R/W	R/W	R/W	R/W				
PROT_TYPE	-							
INITIAL_VALUE	0	1	0	00000				

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	CSA*					Reserved	TXSY*	TXST*
ACCESS_TYPE	R/W					R0,W0	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	00010					0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	PBSY	Reserved			CMD			
ACCESS_TYPE	R,WX	R0,W0			R/W			
PROT_TYPE	-							
INITIAL_VALUE	0	000			0000			

**[bit31:28] Reserved: Reserved bits**

##### **[bit27] CCHB: Connected to channel B (pChannels) bit**

This bit sets whether to connect a node to channel B.

Value	Description
0	Do not connect node to channel B
1	Connect node to channel B

##### **[bit26] CCHA: Connected to channel A (pChannels) bit**

This bit sets whether to connect a node to channel A.

Value	Description
0	Do not connect node to channel B
1	Connect node to channel B





**[bit25] MTSB: MTS symbol transmission on channel B (Select Channel B for MTS Transmission) bit**

This bit selects whether to use channel B for MTS symbol transmission. By default, this bit is cleared to "0" and can be changed only in the DEFAULT\_CONFIG or CONFIG state.

Value	Description
0	Do not use channel B for MTS symbol transmission
1	Use channel B for MTS symbol transmission

**[bit24] MTSA: MTS symbol transmission on channel A (Select Channel A for MTS Transmission) bit**

This bit selects whether to use channel A for MTS symbol transmission. By default, this bit is cleared to "0" and can be changed only in the DEFAULT\_CONFIG or CONFIG state.

Value	Description
0	Do not use channel A for MTS symbol transmission
1	Use channel A for MTS symbol transmission

**Note:**

- If writing to MTSA and MTSB in the SUCC1 register is performed directly using the unlock sequence described in Section "4.2.1. Lock Register" they can be changed in a state other than the DEFAULT\_CONFIG and CONFIG states.

*This is combined with the CHI command SEND\_MTS.*

*If MTSA and MTSB are set at the same time, the MTS symbol is transmitted to both channels by writing CMD[3:0]=1000.*

**[bit23] HCSE: Halt due to clock sync error bit (pAllowHaltDueToClock)**

This bit controls state transition to the HALT state due to a clock synchronization error. This bit can be changed in the DEFAULT\_CONFIG or CONFIG state.

Value	Description
0	NORMAL_PASSIVE state maintained even when clock synchronization error occurs
1	Transition to HALT state occurs due to a clock synchronization error

**[bit22] TSM: Transmission slot mode selection (Transmission Slot Mode) bit (pSingleSlotEnabled)**

This bit selects the initial transmission slot mode. In the SINGLE slot mode, only the preconfigured key slot is used for transmission. A key slot ID consists of message buffers 0 and 1 that follow the MRC:SPLM bit and the header section of message buffer 0. When TSM is "1", message buffers 0 and 1 and message buffer 0 can configure only in the DEFAULT\_CONFIG or CONFIG state. The ALL slot mode allows transmission using all slots.

This bit can be changed only in the DEFAULT\_CONFIG or CONFIG state. However, writing CMD[3:0]=0101 in the NORMAL\_ACTIVE or NORMAL\_PASSIVE state to apply the ALL\_SLOT command enables transition to the ALL slot mode. TSM is a write-only bit. The current slot mode is monitored by CCSV:SLM[1:0].

Value	Description
0	ALL slot mode
1	SINGLE slot mode

**[bit21] WUCS: Wakeup pattern transmission channel selection (Wakeup Channel Select) bit (pWakeupChannel)**

This bit selects a channel to transmit a wakeup pattern. Change to this bit is ignored when the state is not the DEFAULT\_CONFIG or CONFIG state.

bit	Description
0	Transmit wakeup pattern on channel A
1	Transmit wakeup pattern on channel B

**[bit20:16] PTA[4:0]: Required cycle pairs for state transition between passive and active (Passive to Active) bits (pAllowPassiveToActive)**

These bits define the number of consecutive even/odd cycle pairs for the valid clock correction time required for transition from the NORMAL\_PASSIVE state to the NORMAL\_ACTIVE state. If these bits are set to "00000," transition from the NORMAL\_PASSIVE to the NORMAL\_ACTIVE state cannot be achieved. These bits can be changed only in the DEFAULT\_CONFIG or CONFIG state.

Valid values are from 0 to 31 even/odd cycle pairs.

**[bit15:11] CSA[4:0]: Cold start attempts bits (gColdStartAttempts)**

These bits define the maximum number of attempts allowed to repeat startup of the network of cold start nodes when valid response cannot be received from other nodes. This value can be changed only in the DEFAULT\_CONFIG or CONFIG state. The value must be the same across all nodes in a cluster. Valid values are from 2 to 31.

**[bit10] Reserved: Reserved bit**

**[bit9] TXSY: Synchronization frame transmission in key slot (Transmit Sync Frame in Key Slot) bit (pKeySlotUsedForSync)**

This bit defines whether the key slot is used to transmit synchronization frames. This bit can be changed only in the DEFAULT\_CONFIG or CONFIG state.

Value	Description
0	Key slot not used to transmit synchronization frames. Neither in sync nor cold-started
1	Key slot used to transmit synchronization frames. In sync

**[bit8] TXST: Startup frame transmission in key slot (Transmit Startup Frame in Key Slot) bit (pKeySlotUsedForStartup)**

This bit defines whether the key slot is used to transmit startup frames. This bit can be changed only in the DEFAULT\_CONFIG or CONFIG state.

Value	Description
0	Key slot not used to transmit startup frames. Not cold-started
1	Key slot used to transmit startup frames. Cold-started

**Note:**

- To transmit startup frames, set both TXST and TXSY to "1".



**[bit7] PBSY: POC busy bit**

This bit indicates that a command cannot be accepted because the POC is busy. When PBSY is 1, CMD[3:0] is locked against write access. After a hard reset, this bit is set to "1" during initialization of the internal RAM.

Value	Description
0	POC is idle. Writing to CMD[3:0] enabled
1	POC is busy. CMD[3:0] is locked

**[bit6:4] Reserved: Reserved bits**

#### [bit3:0] CMD[3:0]: CHI command vector bits

Although writing to this CHI command vector is always possible, some commands are valid only in a specific POC state. If a command is not valid, the command is not executed, CHI command vector CMD[3:0] is reset to "0000" (command\_not\_accepted), and EIR:CNA is set to "1".

If the last CHI command has not been completed and EIR:CCL is set to "1" together with EIR:CNA, the CHI command needs to be repeated. If a command to change the state to the same POC state as the POC is currently in (except for the HALT state) is applied, this command is ignored and EIR:CNA is not set.

Value	Description
0000	command_not_accepted
0001	CONFIG
0010	READY
0011	WAKEUP
0100	RUN
0101	ALL_SLOTS
0110	HALT
0111	FREEZE
1000	SEND_MTS
1001	ALLOW_COLDSTART
1010	RESET_STATUS_INDICATORS
1011	MONITOR_MODE
1100	CLEAR_RAMs
1101	Reserved
1110	Reserved
1111	Reserved

Reading CMD[3:0] indicates the last accepted CHI command. The actual POC state is monitored by CCSV:POCS[5:0]. The CHI command "Reserved" belongs to the hardware test function.

Generally, the host must check SUCC1:PBSY before setting a CHI command.

#### command\_not\_accepted

If any of the following items applies, a write of CMD[3:0]=0000 is reset to CMD[3:0]=0000:

- An invalid command is set.
- A command is set during a period of internal POC state change.
- A new command is set during execution of a CHI command.
- command\_not\_accepted is set.

If a command is not valid, the command is not executed, the CHI command vector CMD[3:0] is reset to "0000" (command\_not\_accepted), and EIR:CNA is set to "1". If interrupts are enabled, an interrupt is generated.



### CONFIG

When CMD[3:0]=0001 is set in the DEFAULT\_CONFIG, READY, or MONITOR\_MODE state, the state transits to the CONFIG state. When CMD[3:0]=0001 is set in the HALT state, the state transits to the DEFAULT\_CONFIG state. When CMD[3:0]=0001 is set in another state, it is reset to CMD[3:0]=0000 (command\_not\_accepted).

### READY

When CMD[3:0]=0010 is set in the CONFIG, NORMAL\_ACTIVE, NORMAL\_PASSIVE, STARTUP, or WAKEUP state, the state transits to the READY state. When CMD[3:0]=0010 is set in another state, it is reset to CMD[3:0]=0000 (command\_not\_accepted).

### WAKEUP

When CMD[3:0]=0011 is set in the READY state, the state transits to the WAKEUP state. When CMD[3:0]=0011 is set in another state, it is reset to CMD[3:0]=0000 (command\_not\_accepted).

### RUN

When CMD[3:0]=0100 is set in the READY state, the state transits to the STARTUP state. When CMD[3:0]=0100 is set in another state, it is reset to CMD[3:0]=0000 (command\_not\_accepted).

### ALL\_SLOTS

When CMD[3:0]=0101 is set in the NORMAL\_ACTIVE or NORMAL\_PASSIVE state, the mode changes from SINGLE slot mode to ALL slot mode after successful startup/integration at the next end of the cycle. When CMD[3:0]=0101 is set in another state, it is reset to CMD[3:0]= 0000 (command\_not\_accepted).

### HALT

When CMD[3:0]=0110 is set in the NORMAL\_ACTIVE or NORMAL\_PASSIVE state, the halt request bit CCSV:HRQ is set to "1" and the state transits to the HALT state at the next end of the cycle. When CMD[3:0]=0110 is set in another state, it is reset to CMD[3:0]=0000 (command\_not\_accepted).

### FREEZE

When CMD[3:0]=0111 is set, the freeze status indicator CCSV:FSI is set to "1" and the state immediately transits to the HALT state. This can be set in any state.

### SEND\_MTS

When ALL slot mode (CCSV:SLM[1:0]=11) is set, and then the CMD[3:0]=1000 is set in the NORMAL\_ACTIVE state, single MTS symbol is transmitted to the channel set by MTSA and MTSB during the next symbol window. When CMD[3:0]=1000 is set in another state, it is reset to CMD[3:0]=0000 (command\_not\_accepted).

### ALLOW\_COLDSTART

When CMD[3:0]=1001 is set in a state other than the DEFAULT\_CONFIG, CONFIG, or HALT, CCSV:CSI is cleared to "0" to enable the cold start of a node. When CMD[3:0]=1001 is set in the DEFAULT\_CONFIG state, CONFIG state, HALT state, or MONITOR\_MODE, it is reset to CMD[3:0]=0000 (command\_not\_accepted). In addition, to enable cold start, both TXST and TXY must be set.

### RESET\_STATUS\_INDICATORS

When CMD[3:0]=1010 is set, CCSV:CSNI, CCSV:CSAI, and the CCSV:WSV[2:0] status flag are reset. This command is executed in the READY and STARTUP states. If it is executed in another state, the set value is reset to CMD[3:0]=0000 (command\_not\_accepted).

### CLEAR\_RAMs

When CMD[3:0]=1100 is set in the DEFAULT\_CONFIG or CONFIG state, MHDS:CRAM is set to "1". When CMD[3:0]=1100 is set in another state, it is reset to CMD[3:0]=0000 (command\_not\_accepted). Even after a hard reset, MHDS:CRAM is set to "1". Setting MHDS:CRAM to "1" initializes all internal RAM blocks to 0. During RAM initialization, PBSY indicates POC busy. During execution of the CHI command CLEAR\_RAMs (CMD[3:0]="1100"), configuration and status registers are accessible.

Initialization of the internal RAM blocks of the FlexRay controller requires 2048 HCLK cycles. After a hard reset or setting of CMD[3:0]=1100 (CHI command CLEAR\_RAMs), do not access IBF or OBF during initialization of internal RAM blocks.

You must ensure that nothing is transferred between the message RAM and the IBF/OBF before setting CMD[3:0]=1100.

This setting resets the message buffer status registers (MHDS, TXRQ1/2/3/4, NDAT1/2/3/4, and MBSC1/2/3/4).

#### Note:

- An accepted command, except CLEAR\_RAM and SEND\_MTS commands, changes the POC state in the SCLK domain within 8 HCLK or SCLK cycles (whichever is slower) from a CHI falling, if the POC is not busy and remains unchanged in the frame. Reading from the CCSV register delays in synchronization from the SCLK domain to the HCLK domain and in the CPU interface. Maximum additional delay is 12 cycles of the HCLK or SCLK clock whichever is slower.

### MONITOR\_MODE

When CMD[3:0]=1011 is set in the CONFIG state, the state transits to the MONITOR\_MODE. In this mode, FlexRay frames and wakeup patterns can be received and coding errors can be detected. However, the integrity of time is not checked for reception frames. This mode can be used for debug purpose. For example, when startup of a FlexRay network fails, the mode is used for cause analysis. When CMD[3:0]=1011 is set in another state, it is reset to CMD[3:0]=0000 (command\_not\_accepted).

**Table 4-1 Correspondence between CHI Commands in FlexRay Protocol Specifications and CMD[3:0]**

CHI Command	Processing Location (POC State)	CHI Command Vector CMD[3:0]
ALL_SLOTS	POC:normal active, POC:normal passive	ALL_SLOTS
ALLOW_COLDSTART	All except POC:default config, POC:config, POC:halt	ALLOW_COLDSTART
CONFIG	POC:default config, POC:ready	CONFIG
CONFIG_COMPLETE	POC:config	Unlock sequence & READY
DEFAULT_CONFIG	POC:halt	CONFIG
FREEZE	All	FREEZE
HALT	POC:normal active, POC:normal passive	HALT
READY	All except POC:default config, POC:config, POC:ready, POC:halt	READY
RUN	POC:ready	RUN
WAKEUP	POC:ready	WAKEUP



#### 4.4.2. SUC Configuration Register 2 (SUCC2)

This register can be changed only in the DEFAULT\_CONFIG or CONFIG state.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved				LTN*			
ACCESS_TYPE	R0,W0				R/W			
PROT_TYPE	-							
INITIAL_VALUE	0000				0001			

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved			LT[20:16]*				
ACCESS_TYPE	R0,W0			R/W				
PROT_TYPE	-							
INITIAL_VALUE	000			00000				

BIT_OFFSET	15-0							
BIT_NAME	LT[15:0]*							
ACCESS_TYPE	R/W							
PROT_TYPE	-							
INITIAL_VALUE	00000101_00000100							

[bit31:28] Reserved: Reserved bits

[bit27:24] LTN[3:0]: Listen timeout noise value (Listen Timeout Noise) bits (gListenNoise-1)

These bits set the upper limit value for the startup and wakeup listen timeout in the presence of noise expressed as a multiple of pdListenTimeout. The available value range for "gListenNoise" is from 2 to 16. LTN[3:0] must be the same for all nodes of the cluster.

[bit23:21] Reserved: Reserved bits

[bit20:0] LT[20:0]: Listen timeout value (Listen Timeout) bits (pdListenTimeout)

These bits set the listen timeout of startup and wakeup in  $\mu$ T units.

The available value range for "pdListenTimeout" is from 1284 to 1283846  $\mu$ T.

**Note:**

- The wakeup and startup noise timeout is calculated as follows:  

$$pdListenTimeout \times gListenNoise = LT[20:0] \times (LTN[3:0] + 1)$$

### 4.4.3. SUC Configuration Register 3 (SUCC3)

This register can be changed only in the DEFAULT\_CONFIG or CONFIG state.

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,W0
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	WCF*				WCP*			
ACCESS_TYPE	R/W				R/W			
PROT_TYPE	-							
INITIAL_VALUE	0001				0001			

**[bit31:8] Reserved: Reserved bits**

**[bit7:4] WCF[3:0]: Maximum HALT transition time without clock correction (Maximum Without Clock Correction Fatal) bits (gMaxWithoutClockCorrectionFatal)**

These bits define the clock correction loss time inducing state transition from the NORMAL\_ACTIVE or NORMAL\_PASSIVE state to the HALT state in the continuous even/odd cycle pair numbers. It must be the same for all nodes of the cluster. Valid values are 1 to 15 cycle pair numbers.

**[bit3:0] WCP[3:0]: Maximum PASSIVE transition time without clock correction (Maximum Without Clock Correction Passive) bits (gMaxWithoutClockCorrectionPassive)**

These bits define the clock correction loss time inducing state transition from the NORMAL\_ACTIVE state to the NORMAL\_PASSIVE state at a continuous even/odd cycle pair number. It must be the same for all nodes of the cluster. Valid values are 1 to 15 cycle pair numbers.

**Note:**

- If SUCC1:HCSE is not set, there is no transition to the HALT state.





#### 4.4.4. NEM Configuration Register (NEMC)

This register can be changed only in the DEFAULT\_CONFIG or CONFIG state.

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,W0
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved				NML*			
ACCESS_TYPE	R0,W0				R/W			
PROT_TYPE	-							
INITIAL_VALUE	0000				0000			

**[bit31:4] Reserved: Reserved bits**

**[bit3:0] NML[3:0]: Network management vector length bits  
(gNetworkManagementVectorLength)**

These bits set the length of the network management vector. The configured length must be the same for all nodes of the cluster. Valid values are 0 to 12 bytes.

#### 4.4.5. PRT Configuration Register 1 (PRTC1)

This register can be changed only in the DEFAULT\_CONFIG or CONFIG state.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	RWP*						Reserved	RXW[8]*
ACCESS_TYPE	R/W						R0,W0	R/W
PROT_TYPE	-							
INITIAL_VALUE	000010						0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	RXW[7:0]*							
ACCESS_TYPE	R/W							
PROT_TYPE	-							
INITIAL_VALUE	01001100							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	BRP*		SPP*		Reserved	CASM6	CASM5*	CASM4*
ACCESS_TYPE	R/W		R/W		R0,W0	R,WX	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	00		00		0	1	1	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	CASM3*	CASM2*	CASM1*	CASM0	TSST*			
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W			
PROT_TYPE	-							
INITIAL_VALUE	0	0	1	1	0011			

**[bit31:26] RWP[5:0]: Wakeup pattern transmission count (Repetitions of Tx Wakeup Pattern) bits (pWakeupPattern)**

These bits set the number of times the wakeup symbol is transmitted. Valid values are 2 to 63.

**[bit25] Reserved: Reserved bit**

**[bit24:16] RXW[8:0]: Wakeup symbol reception window length (Wakeup Symbol Receive Window Length) bits (gdWakeupSymbolRxWindow)**

These bits set the window length of the wakeup pattern received by the node as a number of bit times. It must be the same for all nodes in a cluster. Valid values are 76 to 301 bit times.



**[bit15:14] BRP[1:0]: Baud Rate Prescaler bits (gdSampleClockPeriod, pSamplePerMicrotick)**

These bits set the baud rate on the FlexRay bus. 1 bit time is always composed of 8 samples (gdSampleClockPeriod X 8). For system clock SCLK setting, see "CHAPTER: FlexRay Dedicated Clock".

00:

gdSampleClockPeriod = 1/SCLK(s)  
pSamplesPerMicrotick = 2

01:

gdSampleClockPeriod = 2/SCLK(s)  
pSamplesPerMicrotick = 1

10, 11:

gdSampleClockPeriod = 4/SCLK(s)  
pSamplesPerMicrotick = 1

**[bit13:12] SPP[1:0]: Strobe Point Position bits**

These bits define the sample count number. The sampling is repeated for the number of times defined by SPP[1:0] and the bit values (High/Low) are determined by the majority of the observed sample values.

00, 11= Sample 5

01 = Sample 4

10 = Sample 6

**Note:**

- In FlexRay protocol 2.1, SPP[1:0]=00. The alternate strobe point location was available for the compensation of the asymmetry in the physical layer.

**[bit11] Reserved: Reserved bit**

**[bit10:4] CASM[6:0]: Collision avoidance symbol upper limit (Collision Avoidance Symbol Max) bits (gdCASRxLowMax)**

These bits define the upper limit of the acceptance window length used for collision avoidance symbol (CAS).

CASM bit6 is fixed to 1. Valid values are 67 to 99 bit times.

**[bit3:0] TSST[3:0]: Transmission start sequence time (Transmission Start Sequence Transmitter) bits (gdTSSTransmitter)**

These bits define the transmission start sequence (TSS) time in units of bit time (1 bit time=4  $\mu$ T=100ns@10Mbps). It must be the same for all nodes of the cluster. Valid values are 3 to 15 bit times.

#### 4.4.6. PRT Configuration Register 2 (PRTC2)

This register can be changed only in the DEFAULT\_CONFIG or CONFIG state.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved		TXL*					
ACCESS_TYPE	R0,W0		R/W					
PROT_TYPE	-							
INITIAL_VALUE	00		001111					

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	TXI*							
ACCESS_TYPE	R/W							
PROT_TYPE	-							
INITIAL_VALUE	00101101							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved		RXL*					
ACCESS_TYPE	R0,W0		R/W					
PROT_TYPE	-							
INITIAL_VALUE	00		001010					

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved		RXI*					
ACCESS_TYPE	R0,W0		R/W					
PROT_TYPE	-							
INITIAL_VALUE	00		001110					

**[bit31:30] Reserved: Reserved bits**

**[bit29:24] TXL[5:0]: Wakeup symbol transmission low time (Wakeup Symbol Transmit Low) bits (gdWakeupSymbolTxLow)**

These bits set the Low time of the wakeup symbol transmitted by the node as a number of bit times. It must be the same for all nodes in a cluster. Valid values are 15 to 60 bit times.

**[bit23:16] TXI[7:0]: Wakeup symbol transmission idle phase time (Wakeup Symbol Transmit Idle) bits (gdWakeupSymbolTxIdle)**

These bits set the idle phase time of the wakeup symbol transmitted by the node as a number of bit times. It must be the same for all nodes in a cluster. Valid values are 45 to 180 bit times.

**[bit15:14] Reserved: Reserved bits**



**[bit13:8] RXL[5:0]: Wakeup reception low time (Wakeup Symbol Receive Low) bits  
(gdWakeupSymbolRxLow)**

These bits set the Low time of the wakeup symbol received by the node as a number of bit times. It must be the same for all nodes in a cluster. Valid values are 10 to 55 bit times.

**[bit7:6] Reserved: Reserved bits**

**[bit5:0] RXI[5:0]: Wakeup reception idle phase time (Wakeup Symbol Receive Idle) bits  
(gdWakeupSymbolRxIdle)**

These bits set the idle phase time of the wakeup symbol received by the node as a number of bit times. It must be the same for all nodes in a cluster. Valid values are 14 to 59 bit times.

#### 4.4.7. MHD Configuration Register (MHDC)

This register can be changed only in the DEFAULT\_CONFIG or CONFIG state.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved			SLT[12:8]*				
ACCESS_TYPE	R0,W0			R/W				
PROT_TYPE	-							
INITIAL_VALUE	000			00000				

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	SLT[7:0]*							
ACCESS_TYPE	R/W							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved							
ACCESS_TYPE	R0,W0							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved	SFDL*						
ACCESS_TYPE	R0,W0	R/W						
PROT_TYPE	-							
INITIAL_VALUE	0	0000000						

**[bit31:29] Reserved: Reserved bits**

**[bit28:16] SLT[12:0]: Transmission end minislot value (Start of Latest Transmit) bits (pLatestTx)**

These bits set the maximum minislot value immediately before the frame transmission is prohibited by the dynamic segment. When SLT[12:0] is set to "0", data is not transmitted to the dynamic segment. Valid values are 0 to 7981 minislots.

**[bit15:7] Reserved: Reserved bits**

**[bit6:0] SFDL[6:0]: Static frame data length bits (gPayloadLengthStatic)**

These bits set the payload length of an entire cluster for all frames transmitted by the static segment. The actual payload length is double the byte length of the set value of this bit. The payload length must be the same for all nodes in a cluster. Valid values are 0 to 127.



#### 4.4.8. GTU Configuration Register 1 (GTUC1)

This register can be changed only in the DEFAULT\_CONFIG or CONFIG state.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,W0							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved				UT[19:16]*			
ACCESS_TYPE	R0,W0				R/W			
PROT_TYPE	-							
INITIAL_VALUE	0000				0000			

BIT_OFFSET	15-0							
BIT_NAME	UT[15:0]*							
ACCESS_TYPE	R/W							
PROT_TYPE	-							
INITIAL_VALUE	00000010_10000000							

[bit31:20] Reserved: Reserved bits

[bit19:0] UT[19:0]: Microtick (Microtick per Cycle) bits (pMicroPerCycle)

These bits set the microtick of the communication cycle. Valid values are 640 to 640000  $\mu$ T.

#### 4.4.9. GTU Configuration Register 2 (GTUC2)

This register can be changed only in the DEFAULT\_CONFIG or CONFIG state.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,W0							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved				SNM*			
ACCESS_TYPE	R0,W0				R/W			
PROT_TYPE	-							
INITIAL_VALUE	0000				0010			

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved		MPC[13:8]*					
ACCESS_TYPE	R0,W0		R/W					
PROT_TYPE	-							
INITIAL_VALUE	00		000000					

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	MPC[7:0]*							
ACCESS_TYPE	R/W							
PROT_TYPE	-							
INITIAL_VALUE	00001010							

**[bit31:20] Reserved: Reserved bits**

**[bit19:16] SNM[3:0]: Maximum sync node (Sync Node Max) bits (gSyncNodeMax)**

These bits set the maximum number of nodes that transmit a synchronous frame (the frame where "1" is set for synchronous frame indicator SYN). It must be the same for all nodes in a cluster. Valid values are 2 to 15.

**[bit15:14] Reserved: Reserved bits**

**[bit13:0] MPC[13:0]: Macrotick (Macrotick Per Cycle) bits (gMacroPerCycle)**

These bits set the macrotick of the communication cycle. The cycle length must be the same for all nodes in a cluster. Valid values are 10 to 16000 MT.





#### 4.4.10. GTU Configuration Register 3 (GTUC3)

This register can be changed only in the DEFAULT\_CONFIG or CONFIG state.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved	MIOB*						
ACCESS_TYPE	R0,W0	R/W						
PROT_TYPE	-							
INITIAL_VALUE	0	0000010						

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved	MIOA*						
ACCESS_TYPE	R0,W0	R/W						
PROT_TYPE	-							
INITIAL_VALUE	0	0000010						

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	UIOB*							
ACCESS_TYPE	R/W							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	UIOA*							
ACCESS_TYPE	R/W							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

**[bit31] Reserved: Reserved bit**

**[bit30:24] MIOB[6:0]: Channel B macrotick initial offset (Macrotick Initial Offset Channel B) bits (pMacroInitialOffset[B])**

These bits specify the number of macroticks between the macrotick boundary after the channel B secondary time reference point and the static slot boundary. This value is based on the nominal macrotick value. It must be the same for all nodes in a cluster. Valid values are 2 to 72 MT.

**[bit23] Reserved: Reserved bit**

**[bit22:16] MIOA[6:0]: Channel A macrotick initial offset (Macrotick Initial Offset Channel A) bits (pMacroInitialOffset[A])**

These bits specify the number of macroticks between the macrotick boundary after channel A secondary time reference point and the static slot boundary. This value is based on the nominal macrotick value. It must be the same for all nodes in a cluster. Valid values are 2 to 72 MT.

**[bit15:8] UIOB[7:0]: Channel B microtick initial offset (Microtick Initial Offset Channel B) bits (pMicroInitialOffset[B])**

These bits set the number of microticks between the macrotick boundary after the channel B secondary time reference point and the actual time reference point. The parameter depends on pDelayCompensation[B]. Therefore, the number must be set for each channel independently. Valid values are 0 to 240  $\mu$ T.

**[bit7:0] UIOA[7:0]: Channel A microtick initial offset (Microtick Initial Offset Channel A) bits (pMicroInitialOffset [A])**

These bits set the number of microticks between the macrotick boundary after the channel A secondary time reference point and the actual time reference point. The parameter depends on pDelayCompensation[A]. Therefore, the number must be set for each channel independently. Valid values are 0 to 240  $\mu$ T.



#### 4.4.11. GTU Configuration Register 4 (GTUC4)

This register can be changed only in the DEFAULT\_CONFIG or CONFIG state. For details of setting NIT[13:0] and OCS[13:0], see "(5) NIT Start and Offset Correction Start Settings" in Section "3.1 CHAPTER 44:3.1 Communication Cycle".

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved		OCS[13:8]*					
ACCESS_TYPE	R0,W0		R/W					
PROT_TYPE	-							
INITIAL_VALUE	00		000000					

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	OCS[7:0]*							
ACCESS_TYPE	R/W							
PROT_TYPE	-							
INITIAL_VALUE	00001000							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved		NIT[13:8]*					
ACCESS_TYPE	R0,W0		R/W					
PROT_TYPE	-							
INITIAL_VALUE	00		000000					

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	NIT[7:0]*							
ACCESS_TYPE	R/W							
PROT_TYPE	-							
INITIAL_VALUE	00000111							

**[bit31:30] Reserved: Reserved bits**

**[bit29:16] OCS[13:0]: Offset correction start bits (gOffsetCorrectionStart - 1)**

These bits determine the offset correction start position in the NIT phase. The position is calculated by counting from the cycle start position. It must be the same for all nodes in a cluster. Valid values are 8 to 15998 MT.

**[bit15:14] Reserved: Reserved bits**

**[bit13:0] NIT[13:0]: Network idle time start bits (gMacroPerCycle - gdNIT - 1)**

These bits set the start point of the network idle time NIT at the end of communication cycle indicated by the number of macroticks.

If the following condition is satisfied, NIT starts.

Macrotick = gMacroPerCycle - gdNIT - 1

It must be the same for all nodes in a cluster. Valid values are 7 to 15997MT.

#### 4.4.12. GTU Configuration Register 5 (GTUC5)

This register can be changed only in the DEFAULT\_CONFIG or CONFIG state.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	DEC*							
ACCESS_TYPE	R/W							
PROT_TYPE	-							
INITIAL_VALUE	00001110							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved			CDD*				
ACCESS_TYPE	R0,W0			R/W				
PROT_TYPE	-							
INITIAL_VALUE	000			00000				

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	DCB*							
ACCESS_TYPE	R/W							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	DCA*							
ACCESS_TYPE	R/W							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

##### [bit31:24] DEC[7:0]: Decoding correction value (Decoding Correction) bits (pDecodingCorrection)

These bits set the decoding correction value used to determine the primary time reference point. Valid values are 14 to 143  $\mu$ T.

##### [bit23:21] Reserved: Reserved bits

##### [bit20:16] CDD[4:0]: Cluster drift damping bits (pClusterDriftDamping)

These bits set the cluster drift damping used for clock synchronization in order to minimize accumulated rounding errors. Valid values are 0 to 20  $\mu$ T.

##### [bit15:8] DCB[7:0]: Channel B reception delay compensation bits (pDelayCompensation[B])

These bits are used to compensate for reception delays on channel B. This guarantees the propagation delay assumed within the range from 0.0125 to 0.05  $\mu$ s up to cPropagationDelayMax set in units of microticks. In reality, the minimum propagation delay time of all synchronous nodes must be applied. Valid values are 0 to 200  $\mu$ T.



**[bit7:0] DCA[7:0]: Channel A reception delay compensation (Delay Compensation Channel A) bits (pDelayCompensation[A])**

These bits are used to compensate for reception delays on channel A. This guarantees the propagation delay assumed within the range from 0.0125 to 0.05  $\mu$ s up to cPropagationDelayMax set in units of microticks. In reality, the minimum propagation delay time of all synchronous nodes must be applied. Valid values are 0 to 200  $\mu$ T.

#### 4.4.13. GTU Configuration Register 6 (GTUC6)

This register can be changed only in the DEFAULT\_CONFIG or CONFIG state.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved					MOD[10:8]*		
ACCESS_TYPE	R0,W0					R/W		
PROT_TYPE	-							
INITIAL_VALUE	00000					000		

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	MOD[7:0]*							
ACCESS_TYPE	R/W							
PROT_TYPE	-							
INITIAL_VALUE	00000010							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved					ASR[10:8]*		
ACCESS_TYPE	R0,W0					R/W		
PROT_TYPE	-							
INITIAL_VALUE	00000					000		

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	ASR[7:0]*							
ACCESS_TYPE	R/W							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

**[bit31:27] Reserved: Reserved bits**

**[bit26:16] MOD[10:0]: Maximum oscillator drift bits (pdMaxDrift)**

These bits set the maximum drift offset between 2 asynchronous nodes in 1 communication cycle in  $\mu\text{T}$  units. Valid values are 2 to 1923  $\mu\text{T}$ .

**[bit15:11] Reserved: Reserved bits**

**[bit10:0] ASR[10:0]: Acceptance startup range (Accepted Startup Range) bits (pdAcceptedStartupRange)**

These bits set the extended range of the measurement error to the startup frame as a number of microticks. Valid values are 0 to 1875  $\mu\text{T}$ .



#### 4.4.14. GTU Configuration Register 7 (GTUC7)

This register can be changed only in the DEFAULT\_CONFIG or CONFIG state.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved						NSS[9:8]*	
ACCESS_TYPE	R0,W0						R/W	
PROT_TYPE	-							
INITIAL_VALUE	000000						00	

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	NSS[7:0]*							
ACCESS_TYPE	R/W							
PROT_TYPE	-							
INITIAL_VALUE	00000010							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved						SSL[9:8]*	
ACCESS_TYPE	R0,W0						R/W	
PROT_TYPE	-							
INITIAL_VALUE	000000						00	

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	SSL[7:0]*							
ACCESS_TYPE	R/W							
PROT_TYPE	-							
INITIAL_VALUE	00000100							

**[bit31:26] Reserved: Reserved bits**

**[bit25:16] NSS[9:0]: Number of static slots bits (gNumberOfStaticSlots)**

These bits set the number of static slots in a cycle. At least 2 cold-start nodes must be configured to start up a FlexRay network. The number of static slots must be the same for all nodes in a cluster. Valid values are 2 to 1023.

**[bit15:10] Reserved: Reserved bits**

**[bit9:0] SSL[9:0]: Static slot length bits (gdStaticSlot)**

These bits set the static slot period in macroticks. The static slot length must be the same for all nodes in a cluster. Valid values are 4 to 659 MT.

#### 4.4.15. GTU Configuration Register 8 (GTUC8)

This register can be changed only in the DEFAULT\_CONFIG or CONFIG state.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved			NMS[12:8]*				
ACCESS_TYPE	R0,W0			R/W				
PROT_TYPE	-							
INITIAL_VALUE	000			00000				

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	NMS[7:0]*							
ACCESS_TYPE	R/W							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved							
ACCESS_TYPE	R0,W0							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved		MSL*					
ACCESS_TYPE	R0,W0		R/W					
PROT_TYPE	-							
INITIAL_VALUE	00		000010					

**[bit31:29] Reserved: Reserved bits**

**[bit28:16] NMS[12:0]: Number of minislots bits (gNumberOfMinislots)**

These bits set the number of minislots within the dynamic segment of 1 cycle. The number of minislots must be the same for all nodes in a cluster. Valid values are 0 to 7986.

**[bit15:6] Reserved: Reserved bits**

**[bit5:0] MSL[5:0]: Minislot length bits (gdMinislot)**

These bits set the minislot period in macroticks. The minislot length must be the same for all nodes in a cluster. Valid values are 2 to 63 MT.





#### 4.4.16. GTU Configuration Register 9 (GTUC9)

This register can be changed only in the DEFAULT\_CONFIG or CONFIG state.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,W0							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved						DSI*	
ACCESS_TYPE	R0,W0						R/W	
PROT_TYPE	-							
INITIAL_VALUE	000000						00	

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved			MAPO*				
ACCESS_TYPE	R0,W0			R/W				
PROT_TYPE	-							
INITIAL_VALUE	000			00001				

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved		APO*					
ACCESS_TYPE	R0,W0		R/W					
PROT_TYPE	-							
INITIAL_VALUE	00		000001					

**[bit31:18] Reserved: Reserved bits**

**[bit17:16] DSI[1:0]: Dynamic slot idle phase bits (gdDynamicSlotIdlePhase)**

These bits set the duration of the dynamic slot idle phase. The duration must be equal to or greater than the idle detection time. All nodes in a cluster must be the same. Valid values are 0 to 2 minislots.

**[bit15:13] Reserved: Reserved bits**

**[bit12:8] MAPO[4:0]: Minislot action point offset bits (gdMinislotActionPointOffset)**

These bits set the action point offset within the minislot of dynamic segment in macroticks. All nodes in a cluster must be the same. Valid values are 1 to 31 MT.

**[bit7:6] Reserved: Reserved bits**

**[bit5:0] APO[5:0]: Action point offset bits (gdActionPointOffset)**

These bits set the action point offset in the static slot and symbol window in macroticks. It must be the same for all nodes in a cluster. Valid values are 1 to 63 MT.

#### 4.4.17. GTU Configuration Register 10(GTUC10)

This register can be changed only in the DEFAULT\_CONFIG or in CONFIG state.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved					MRC[10:8]*		
ACCESS_TYPE	R0,W0					R/W		
PROT_TYPE								
INITIAL_VALUE	00000					000		

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	MRC[7:0]*							
ACCESS_TYPE	R/W							
PROT_TYPE	-							
INITIAL_VALUE	00000010							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved		MOC[13:8]*					
ACCESS_TYPE	R0,W0		R/W					
PROT_TYPE	-							
INITIAL_VALUE	00		000000					

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	MOC[7:0]*							
ACCESS_TYPE	R/W							
PROT_TYPE	-							
INITIAL_VALUE	00000101							

**[bit31:27] Reserved: Reserved bit**

**[bit26 to bit16] MRC[10:0]: Maximum rate correction value bit (pRateCorrectionOut)**

These bits set the maximum allowable rate correction value used by the internal clock synchronous algorithm. The sum of the internal rate correction and the external rate correction (absolute value) is compared with this value. Valid values are 2 to 1923  $\mu$ T.

**[bit15:14] Reserved: Reserved bit**

**[bit13:0] MOC[13:0]: Maximum offset correction value bit (pOffsetCorrectionOut)**

These bits set the maximum allowable offset correction value used by the internal clock synchronous algorithm (absolute value). The sum of the internal offset correction and the external offset correction is compared with this value. Valid values are 5 to 15266  $\mu$ T.



#### 4.4.18. GTU Configuration Register 11(GTUC11)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved					ERC*		
ACCESS_TYPE	R0,W0					R/W		
PROT_TYPE	-							
INITIAL_VALUE	00000					000		

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved					EOC*		
ACCESS_TYPE	R0,W0					R/W		
PROT_TYPE	-							
INITIAL_VALUE	00000					000		

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved						ERCC	
ACCESS_TYPE	R0,W0						R/W	
PROT_TYPE	-							
INITIAL_VALUE	000000						00	

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved						EOCC	
ACCESS_TYPE	R0,W0						R/W	
PROT_TYPE	-							
INITIAL_VALUE	000000						00	

**[bit31:27] Reserved : Reserved bit**

**[bit26 to bit24] ERC[2:0]: External rate correction bit (pExternRateCorrection)**

These bits set the external rate correction value used by the internal clock synchronous algorithm in microticks. The value is used to add to the calculated rate correction value or to subtract from the rate correction value. The value is applied during NIT. The value can be changed only in the DEFAULT\_CONFIG or CONFIG state. Valid values are 0 to 7  $\mu$ T.

**[bit23:19] Reserved : Reserved bit**

**[bit19 to bit16] EOC[2:0]: External offset correction bit (pExternOffsetCorrection)**

These bits set the external offset correction value used by the internal clock synchronous algorithms in microticks. The value is used to add to the calculated offset correction value or to subtract from the offset correction value. The value is applied during NIT. The value can be changed only in the DEFAULT\_CONFIG or CONFIG state. Valid values are 0 to 7  $\mu$ T.

**[bit15:10] Reserved : Reserved bit**

**[bit9:8] ERCC[1:0]: External rate correction control bit (vExternRateControl)**

These bits enable the external rate correction by writing the set value shown below to ERCC[1:0].

Change the value outside NIT.

Value	Description
00	No external rate correction value
01	
10	Subtract amount comprising external rate correction value from calculated rate correction value
11	Add amount comprising external rate correction value to calculated rate correction value

**[bit7:2] Reserved : Reserved bit**

**[bit1:0] EOCC[1:0]: External offset correction control bit (vExternOffsetControl)**

These bits enable external offset correction by writing the set value shown below to EOCC[1:0]. Change the value outside NIT.

Value	Description
00	No external offset correction value
01	
10	Subtract amount comprising external offset correction value from calculated offset correction value
11	Add amount comprising external offset correction value to calculated offset correction value



## 4.5. Communication Controller (CC) Status Registers

By on chip bus clock (HCLK) frequency, the status vector information successively changes faster than the host polls the status vector.

### 4.5.1. CC Status Vector Register(CCSV)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved		PSL					
ACCESS_TYPE	R0,W0		R,WX					
PROT_TYPE	-							
INITIAL_VALUE	00		000000					

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	RCA					WSV		
ACCESS_TYPE	R,WX					R,WX		
PROT_TYPE	-							
INITIAL_VALUE	00010					000		

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved	CSI	CSAI	CSNI	Reserved		SLM	
ACCESS_TYPE	R0,W0	R,WX	R,WX	R,WX	R0,W0		R,WX	
PROT_TYPE	-							
INITIAL_VALUE	0	1	0	0	00		00	

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	HRQ	FSI	POCS					
ACCESS_TYPE	R,WX	R,WX	R,WX					
PROT_TYPE	-							
INITIAL_VALUE	0	0	000000					

**[bit31:30] Reserved : Reserved bit**

**[bit29 to bit24] PSL[5:0]: POC status log bit**

These bits indicate the POCS[5:0] status before entering the HALT state. POC is set when entering the HALT state. These bits enter the HALT state with the FREEZE command issued in the HALT state, and FSI is not yet set. That is, the bits do not reach the HALT state with the FREEZE command. The bits are reset to "000000" when a transition from the HALT state occurs.

**[bit23:19] RCA[4:0]: Remaining cold start attempts bit (vRemainingColdstartAttempts)**

These bits indicate the remaining number of cold-start attempts. The RUN command resets this counter to the maximum number of cold-start attempts set at SUCC1:CSA[4:0]. The initial value in the CONFIG or DEFAULT\_CONFIG state is also the value of SUCC1:CSA[4:0].

**[bit18:16] WSV[2:0]: Wakeup status bit (vPOC!WakeupStatus)**

These bits indicate the current wakeup state (See Section "3.5.6. WAKEUP State"). POC is reset by CHI instruction reset, or by transition from the RESET\_STATUS\_INDICATORS or DEFAULT\_CONFIG state to the CONFIG state.

Value	Description
000	UNDEFINED Wakeup not started
001	RECEIVED_HEADER This is set when wakeup ends due to the reception of an error free frame header on any of the channels in the WAKEUP_LISTEN state
010	RECEIVED_WUP This is reset when wakeup ends due to the reception of a valid wakeup pattern on the specified wakeup channel in the WAKEUP_LISTEN state
011	A COLLISION_HEADER This is reset when wakeup stops due to the detection of a collision resulting from the reception of a valid header on either of the channels during wakeup pattern transmission
100	COLLISION_WUP This is set when wakeup stops due to the detection of a collision resulting from the reception of a valid wakeup pattern on the specified wakeup channel during wakeup pattern transmission
101	COLLISION_UNKNOWN This is set when wakeup stops due to a transition from the WAKEUP_DETECT state because the wakeup timer did not receive both the valid wakeup pattern and valid frame header within the specified time
110	TRANSMITTED This is set when the wakeup pattern transmission is normally completed
111	reserved

**[bit15] Reserved : Reserved bit**

**[bit14] CSI: Cold Start Inhibit bit (vColdStartInhibit)**

This bit indicates that the node is disabled from cold start. This flag is set to "1" whenever POC is in the READY state due to CHI instruction READY. This flag is reset by setting SUCC1:CMD[3:0]="1001"(CHI instruction ALLOW\_COLDSTART).

Value	Description
0	Cold start of node enabled
1	Cold start of node disabled

**[bit13] CSAI: Cold start abort indicator bit**

This bit indicates that the cold start was aborted. It is reset by CHI instruction reset, by transition from the RESET\_STATUS\_INDICATORS or HALT state to the DEFAULT\_CONFIG state, or by transition from the READY state to the STARTUP state.

**[bit12] CSNI: Cold start noise indicator bit (vColdStartNoise)**

This bit indicates that the cold-start procedure was executed under noisy conditions. It is reset by CHI instruction reset, by transition from the RESET\_STATUS\_INDICATORS or HALT state to the DEFAULT\_CONFIG state, or transition from the READY state to the STARTUP state.



**[bit11:10] Reserved : Reserved bit**

**[bit9:8] SLM[1:0]: Slot mode bit (vPOC!SlotMode)**

These bits indicate the current POC slot mode in the READY, STARTUP, NORMAL\_ACTIVE, and NORMAL\_PASSIVE state. The default value is SINGLE slot mode. This is changed to ALL using SUCC1:TSM. In the NORMAL\_ACTIVE or NORMAL\_PASSIVE state, when CHI command CMD[3:0]="0101"(ALL\_SLOTS) is set, the slot mode changes from SINGLE slot mode to ALL slot mode via ALL\_PENDING. In all other states, it changes to SINGLE slot mode.

Value	Description
00	SINGLE slot mode
01	reserved
10	ALL_PENDING
11	ALL slot mode

**[bit7] HRQ: Halt request bit (vPOC!CHIHaltRequest)**

This bit indicates that the host requests a state transition to the HALT state at the end of communication cycle. It is reset when transiting from the HALT state to the DEFAULT\_CONFIG state, or transiting to the READY state.

**[bit6] FSI: Freeze status indicator bit (vPOC!Freeze)**

This bit indicates a state transition to the HALT state either because CMD[3:0]="0111"(CHI command FREEZE) is set or because an error requiring the immediate transition to the HALT state occurs. It is reset when transiting from the HALT state to the DEFAULT\_CONFIG state.

**[bit5:0] POCS[5:0]: POC state (Protocol Operation Control Status)**

These bits display the current POC execution state.

Value	Description
00 0000	DEFAULT_CONFIG state
00 0001	READY state
00 0010	NORMAL_ACTIVE state
00 0011	NORMAL_PASSIVE state
00 0100	HALT state
00 0101	MONITOR_MODE state
00 0110 to 00 1110	reserved
00 1111	CONFIG state

These bits display the current POC state in the wakeup procedure(010000 to 011111).

Value	Description
01 0000	WAKEUP_STANDBY state
01 0001	WAKEUP_LISTEN state
01 0010	WAKEUP_SEND state
01 0011	WAKEUP_DETECT state
01 0100 to 01 1111	reserved

These bits display the current POC state in the startup procedure(010000 to 011111).

Value	Description
10 0000	STARTUP_PREPARE state
10 0001	COLDSTART_LISTEN state
10 0010	COLDSTART_COLLISION_RESOLUTION state
10 0011	COLDSTART_CONSISTENCY_CHECK state
10 0100	COLDSTART_GAP state
10 0101	COLDSTART_JOIN state
10 0110	INTEGRATION_COLDSTART_CHECK state
10 0111	INTEGRATION_LISTEN state
10 1000	INTEGRATION_CONSISTENCY_CHECK state
10 1001	INITIALIZE_SCHEDULE state
10 1010	ABORT_STARTUP state
10 1011	START UP_SUCCESS state
10 1011 to 11 1111	reserved





### 4.5.2. CC Error Vector Register (CCEV)

This register is reset when transiting from the HALT state to the DEFAULT\_CONFIG state, or transiting to the READY state.

BIT_OFFSET	31-16
BIT_NAME	Reserved
ACCESS_TYPE	R0,W0
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved			PTAC				
ACCESS_TYPE	R0,W0			R,WX				
PROT_TYPE	-							
INITIAL_VALUE	000			00000				

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	ERRM		Reserved		CCFC			
ACCESS_TYPE	R,WX		R0,W0		R,WX			
PROT_TYPE	-							
INITIAL_VALUE	00		00		0000			

**[bit31:13] Reserved: Reserved bit**

**[bit12:8] PTAC[4:0]: Counter for cycle pair number required for state transition from passive to active bit (vAllowPassiveToActive)**

These bits indicate the number of continuous odd/even cycle pairs that passed because the rate correction time and the offset correction time are valid while the node is transiting from the NORMAL\_PASSIVE state to the NORMAL\_ACTIVE state. The state transition is performed when PTAC[4:0] is equal to SUCC1:PTA[4:0]-1.

**[bit7:6] ERRM[1:0]: Error mode (vPOC!ErrorMode)**

These bits indicate the current error mode of POC.

Value	Description
00	ACTIVE
01	PASSIVE
10	COMM_HALT
11	reserved

**[bit5:4] Reserved: Reserved bit**

**[bit3:0] CCFC[3:0]: Clock correction failed counter bit (vClockCorrectionFailed)**

The counter increases by one at the end of an odd communication cycle when either an offset correction missing error or rate correction missing error occurs. When neither an offset correction missing error nor a rate correction missing error occurs, the counter is reset to "0" at the end of an odd communication cycle. The clock correction failed counter stops at 15.



### 4.5.3. Slot Counter Value Register (SCV)

This register is reset when transitioning from the CONFIG state or transitioning to the STARTUP state.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved					SCCB[10:8]		
ACCESS_TYPE	R0,W0					R,WX		
PROT_TYPE	-							
INITIAL_VALUE	00000					000		

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	SCCB[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved					SCCA[10:8]		
ACCESS_TYPE	R0,W0					R,WX		
PROT_TYPE	-							
INITIAL_VALUE	00000					000		

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	SCCA[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

**[bit31:27] Reserved: Reserved bits**

**[bit26 to bit16] SCCB[10:0]: Channel B slot counter bit (vSlotCounter[B])**

These bits indicate the current slot counter value of channel B. This value becomes 1 at the start of the communication cycle and increments up to the end of the cycle at the end of each static slot. Valid values are 0 to 2047.

**[bit15:11] Reserved: Reserved bits**

**[bit10:0] SCCA[10:0]: Channel A slot counter bit (vSlotCounter[A])**

These bits indicate the current slot counter value of channel A. This value becomes 1 at the start of the communication cycle and increments up to the end of the cycle at the end of each static slot. Valid values are 0 to 2047.

#### 4.5.4. Macrotick and Cycle Counter Value Register (MTCCV)

This register is reset when transitioning from the CONFIG state or transitioning to the STARTUP state.

BITS	31	30	29	28	27	26	25	24
BIT_OFFSET								
BIT_NAME	Reserved							
ACCESS_TYPE	R0,W0							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved		CCV					
ACCESS_TYPE	R0,W0		R,WX					
PROT_TYPE	-							
INITIAL_VALUE	00		000000					

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved		MTV[13:8]					
ACCESS_TYPE	R0,W0		R,WX					
PROT_TYPE	-							
INITIAL_VALUE	00		000000					

BITS	7	6	5	4	3	2	1	0
BIT_OFFSET								
BIT_NAME	MTV[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

**[bit31:22] Reserved: Reserved bit**

**[bit21:16] CCV[5:0]: Cycle counter value bit (vCycleCounter)**

These bits indicate the current cycle counter value. This value increments at the start of the communication cycle. Valid values are 0 to 63.

**[bit15:14] Reserved: Reserved bit**

**[bit13:0] MTV[13:0]: Macrotick value bit (vMacrotick)**

These bits indicate the current macrotick value. This value becomes 0 at the start of the communication cycle and increments up to the end of the cycle. Valid values are 0 to 15999.



#### 4.5.5. Rate Correction Value Register (RCV)

This register is reset when transiting from the CONFIG state or transiting to the STARTUP state.

BIT_OFFSET	31-16
BIT_NAME	Reserved
ACCESS_TYPE	R0,W0
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved				RCV[11:8]			
ACCESS_TYPE	R0,W0				R,WX			
PROT_TYPE	-							
INITIAL_VALUE	0000				0000			

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	RCV[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

**[bit31:12] Reserved : Reserved bits**

#### **[bit11:0] RCV[11:0]: Rate correction value bit (vRateCorrection)**

These bits indicate the rate correction value (complement of 2). This is the rate correction value calculated inside the controller before being restricted with the maximum rate correction value GTUC10:MRC[10:0]. When the value exceeds the maximum rate correction value, the SFS.RCLR flag is set to "1".

#### 4.5.6. Offset Correction Value Register (OCV)

This register is reset when transiting from the CONFIG state or transiting to the STARTUP state.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,W0							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved					OCV[18:16]		
ACCESS_TYPE	R0,W0					R,WX		
PROT_TYPE	-							
INITIAL_VALUE	00000					000		

BIT_OFFSET	15-0							
BIT_NAME	OCV[15:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	-							
INITIAL_VALUE	00000000_00000000							

**[bit31:19] Reserved : Reserved bits**

##### **[bit18:0] OCV[18:0]: Offset correction value bit (vOffsetCorrection)**

These bits indicate the offset correction value (complement of 2). This is the offset correction value calculated internally before being restricted with the maximum offset correction value GTUC10:MOC[10:0]. When the value exceeds the maximum offset correction value, the SFS:OCLR flag is set to "1".

**Note:**

- The external rate/offset correction value is added to the rate/offset correction value restricted with the maximum rate/offset correction value.



### 4.5.7. Sync Frame Status Register (SFS)

The maximum value of the valid synchronous frame at 1 communication cycle is 15.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,W0							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved				RCLR	MRCS	OCLR	MOCS
ACCESS_TYPE	R0,W0				R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0000				0	0	0	0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	VSBO				VSBE			
ACCESS_TYPE	R,WX				R,WX			
PROT_TYPE	-							
INITIAL_VALUE	0000				0000			

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	VSAO				VSAE			
ACCESS_TYPE	R,WX				R,WX			
PROT_TYPE	-							
INITIAL_VALUE	0000				0000			

**[bit31:20] Reserved: Reserved bits**

**[bit19] RCLR: Rate correction limit reached bit**

This bit indicates that the rate correction value exceeded the limit value defined by GTUC10:MRC[10:0]. This flag is updated when the offset correction phase starts.

Value	Description
0	Rate correction value not exceeded limit value
1	Rate correction value exceeded limit value

**[bit18] MRCS: Missing rate correction signal bit**

This bit indicates that the rate correction calculation was not performed because even/odd synchronous frame pairs were not received. This flag is updated when the offset correction phase starts.

Value	Description
0	Rate correction signal enabled
1	Rate correction signal missing

**[bit17] OCLR: Offset correction limit reached bit**

This bit indicates that the offset correction value exceeded the limit value defined by GTUC10:MOC[13:0]. This flag is updated when the offset correction phase starts.

Value	Description
0	Offset correction value not exceeded limit value
1	Offset correction value exceeded limit value

**[bit16] MOCS: Missing offset correction signal bit**

This bit indicates that the offset correction calculation was not performed because the synchronous frame was not received. This flag is updated when the offset correction phase starts.

Value	Description
0	Offset correction signal enabled
1	Offset correction signal missing

**[bit15:12] VSBO[3:0]: Channel B valid sync frames, odd communication cycle bits**

These bits indicate the number of valid synchronous frames received in the odd communication cycle on channel B. If synchronous frame transmission is permitted by SUCC1:TXSY, the value increases by one each. This value is updated during NIT period of each odd communication cycle.

**Note:**

- The above bit fields are enabled only when each channel is assigned by SUCC1:CCHA or SUCC1:CCHB.

**[bit11:8] VSBE[3:0]: Channel B valid sync frames, even communication cycle bits**

These bits indicate the number of valid synchronous frames received in the even communication cycle on channel B. If synchronous frame transmission is permitted by SUCC1:TXSY, the value increases by one each. This value is updated during the NIT period of each even communication cycle.

**[bit7:4] VSAO[3:0]: Channel A valid sync frames, odd communication cycle bits**

These bits indicate the number of valid synchronous frames received in the odd communication cycle on channel A. If synchronous frame transmission is permitted by SUCC1:TXSY, the value increases by one each. This value is updated during the NIT period of each odd communication cycle.

**[bit3:0] VSAE[3:0]: Channel A valid sync frames, even communication cycle bits**

These bits indicate the number of valid synchronous frames received in the even communication cycle on channel A. If synchronous frame transmission is permitted by SUCC1:TXSY, the value increases by one each. This value is updated during the NIT period of each even communication cycle.





### 4.5.8. Symbol Window and NIT Status Register (SWNI)

The bits that hold symbol window related status information are shown below. These bits are updated when the symbol window of each channel ends. They are not updated during startup.

This register is reset when transiting from the CONFIG state or transiting to the STARTUP state.

BIT_OFFSET	31-16
BIT_NAME	Reserved
ACCESS_TYPE	R0,W0
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved				SBNB	SENB	SBNA	SENA
ACCESS_TYPE	R0,W0				R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0000				0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	MTSB	MTSA	TCSB	SBSB	SESB	TCSA	SBSA	SESA
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:12] Reserved: Reserved bits

[bit11] SBNB: Channel B slot boundary violation during NIT bit (vSS!BViolationB)

Value	Description
0	No slot boundary violation detected during NIT on channel B
1	Slot boundary violation detected during NIT on channel B

[bit10] SENB: Channel B syntax error during NIT bit (vSS!SyntaxErrorB)

Value	Description
0	No syntax error detected during NIT on channel B
1	Syntax error detected during NIT on channel B

[bit9] SBNA: Channel A slot boundary violation during NIT bit (vSS!BViolationA)

Value	Description
0	No slot boundary violation detected during NIT on channel A
1	Slot boundary violation detected during NIT on channel A

[bit8] SENA: Channel A syntax error during NIT bit (vSS!SyntaxErrorA)

Value	Description
0	No syntax error detected during NIT on channel A
1	Syntax error detected during NIT on channel A

**[bit7] MTSB: Channel B media access test symbol detection bit (vSS!ValidMTSB)**

Media access test symbol was received on channel B of the previous symbol window.

This bit is updated by the CC of each channel at the end of the symbol window. In addition, when this bit is set to "1", interrupt flag SIR:MTSA is set to "1".

Value	Description
0	No MTS symbol detected on channel B
1	MTS symbol detected on channel B

The bits that hold NIT related status information are shown below. These bits are updated when NIT of each channel ends.

**[bit6] MTSA: Channel A media access test symbol detection bit (vSS!ValidMTSA)**

Media access test symbol received on channel A of the previous symbol window.

This bit is updated by the CC of each channel at the end of the symbol window. In addition, when this bit is set to "1", interrupt flag SIR:MTSA is set to "1".

Value	Description
0	No MTS symbol detected on channel A
1	MTS symbol detected on channel A

**[bit5] TCSB: Channel B symbol window transmission collision detection bit (vSS!TxConflictB)**

Value	Description
0	No transmission collision detected during symbol window of channel B
1	Transmission collision detected during symbol window of channel B

**[bit4] SBSB: Channel B symbol window slot boundary violation bit (vSS!BViolationB)**

Value	Description
0	No slot boundary violation detected during symbol window of channel B
1	Slot boundary violation detected during symbol window of channel B

**[bit3] SESB: Channel B symbol window syntax error bit (vSS!SyntaxErrorB)**

Value	Description
0	No syntax error detected during symbol window of channel B
1	Syntax error detected during symbol window of channel B

**[bit2] TCSA: Channel A symbol window transmission collision detection bit (vSS!TxConflictA)**

Value	Description
0	No transmission collision detected during symbol window of channel A
1	Transmission conflict detected during symbol window of channel A

**[bit1] SBSA: Channel A symbol window slot boundary violation bit (vSS!BViolationA)**

Value	Description
0	No slot boundary violation detected during symbol window of channel A
1	Slot boundary violation detected during symbol window of channel A



**[bit0] SESA: Channel A symbol window syntax error bit (vSS!SyntaxErrorA)**

Value	Description
0	No syntax error detected during symbol window of channel A
1	Syntax error detected during symbol window of channel A

#### 4.5.9. Aggregated Channel Status Register (ACS)

This register provides the status that occurs during the channel operation of communication slots regardless of whether all communication slots are assigned for transmission or reception. In addition, this register includes the status data from the symbol window and NIT. The status data is updated after each slot (the latest data at the end of the next slot).

Each flag of this register is cleared by writing "1" to the corresponding bit position. Writing "0" has no effect on the flag. The register is reset when transiting from the CONFIG state or transiting to the STARTUP state.

BIT_OFFSET	31-16
BIT_NAME	Reserved
ACCESS_TYPE	R0,W0
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved			SBVB	CIB	CEDB	SEDB	VFRB
ACCESS_TYPE	R0,W0			R/W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	000			0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved			SBVA	CIA	CEDA	SEDA	VFRA
ACCESS_TYPE	R0,W0			R/W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	000			0	0	0	0	0

**[bit31:13] Reserved: Reserved bits**

#### **[bit12] SBVB: Channel B slot boundary violation bit (vSS!BViolationB)**

This bit indicates that 1 or more slot boundary violations were observed on channel B in any of a static slot, dynamic slot, symbol window, or NIT.

Value	Description
0	No slot boundary violation observed on channel B
1	Slot boundary violation observed on channel B

**Note:**

- If there is only 1 frame in the slot and the slot boundary at the end of the slot enters the idle phase, a set of conditions of flag CIA and CIB is satisfied.  
When any of the SEDB, CIB, CEDB, or SBVB flag changes from "0" to "1", interrupt flag EIR:EDB is set to "1". When any of the SEDA, CEDA, CIA, or SBVA flag changes from "0" to "1", interrupt flag EIR:EDA is set to "1".



**[bit11] CIB: Channel B additional communication detection bit**

This bit indicates that 1 or more valid frames were received in a slot containing additional communication on channel B. This means that 1 or more slots received the valid frame(s) and that there was a combination of any of a syntax error, content error, or slot boundary violation.

Value	Description
0	No frame containing additional communication received on channel B
1	Frame containing additional communication received on channel B

**[bit10] CEDB: Channel B content error detection bit (vSS!ContentErrorB)**

This bit indicates that 1 or more frames containing a content error were received in a static slot or dynamic slot on channel B.

Value	Description
0	No frame containing content error received on channel B
1	Frame containing content error received on channel B

**[bit9] SEDB: Channel B syntax error detection bit (vSS!SyntaxErrorB)**

This bit indicates that 1 or more syntax errors were observed on channel B in any of a static slot, dynamic slot, symbol window, or NIT.

Value	Description
0	No syntax error observed on channel B
1	Syntax error observed on channel B

**[bit8] VFRB: Channel B valid frame reception bit (vSS!ValidFrameB)**

This bit indicates that 1 or more valid frames were received in a static slot or dynamic slot on channel B.

Value	Description
0	No valid frame received on channel B
1	Valid frame received on channel B

**[bit7:5] Reserved: Reserved bits**

**[bit4] SBVA: Channel A slot boundary violation bit (vSS!BViolationA)**

This bit indicates that 1 or more slot boundary violations were observed on channel A in any of a static slot, dynamic slot, symbol window or NIT.

Value	Description
0	No slot boundary violation observed on channel A
1	Slot boundary violation observed on channel A

**[bit3] CIA: Channel A additional communication detection bit**

This bit indicates that 1 or more valid frames were received in a slot containing additional communication on channel A. This means that 1 or more slots receive the valid frame(s) and that there is a combination with any of a syntax error, content error, or slot boundary violation.

Value	Description
0	No frame containing additional communication received on channel A
1	Frame containing additional communication received on channel A

**[bit2] CEDA: Channel A content error detection bit (vSS!ContentErrorA)**

This bit indicates that 1 or more frames containing a content error were received in a static slot or dynamic slot on channel A.

Value	Description
0	No frame containing content error received on channel A
1	Frame containing content error received on channel A

**[bit1] SEDA: Channel A syntax error detection bit (vSS!SyntaxErrorA)**

This bit indicates that 1 or more syntax errors were observed on channel A in any of a static slot, dynamic slot, symbol window, or NIT.

Value	Description
0	No syntax error observed on channel A
1	Syntax error observed on channel A

**[bit0] VFRA: Channel A valid frame reception bit (vSS!ValidFrameA)**

This bit indicates that 1 or more valid frames were received in a static slot or dynamic slot on channel A.

Value	Description
0	No valid frame received on channel A
1	Valid frame received on channel A



#### 4.5.10. Even Cycle Sync Frame ID Register (ESIDn)

The 15 registers from ESID1 to ESID15 store in ascending order the frame IDs of the synchronous frames received during the even communication cycles and are used for clock synchronization up to the gSyncNodeMax limit. Therefore, the smallest synchronous frame ID received is stored in the ESID1 register. When the node transmits a synchronous frame during an even communication cycle, the ESID1 register stores the transmission synchronous frame ID and RXEA and RXEB are set. The register value is updated during NIT of each even communication cycle. The register is reset when transiting from the CONFIG state or transiting to the STARTUP state.

BIT_OFFSET	31-16
BIT_NAME	Reserved
ACCESS_TYPE	R0,W0
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	RXEB	RXEA	Reserved				EID[9:8]	
ACCESS_TYPE	R,WX	R,WX	R0,W0				R,WX	
PROT_TYPE	-							
INITIAL_VALUE	0	0	0000				00	

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	EID[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

**[bit31:16] Reserved: Reserved bits**

##### **[bit15] RXEB: Even cycle sync frame channel B reception bit**

This bit indicates that the synchronous frame corresponding to the synchronous ID at even cycle was received on channel B. The node is configured as a sync node of key slot=EID[9:0](ESID1 only).

Value	Description
0	No sync frame received on channel B/Not transmission sync frame
1	Sync frame received on channel B/Transmission sync frame

##### **[bit14] RXEA: Even cycle sync frame channel A reception bit**

This bit indicates that the synchronous frame corresponding to the synchronous ID at even cycle was received on channel A. The node is configured as a sync node of key slot=EID[9:0](ESID1 only).

Value	Description
0	No sync frame received on channel A/Not transmission sync frame
1	Sync frame received on channel A/Transmission sync frame

**[bit13:10] Reserved : Reserved bits**

**[bit9:0] EID[9:0]: Even cycle sync frame ID bit (vsSyncIDListA,B even)**

These bits indicate the synchronous frame ID of the even communication cycle.





#### 4.5.11. Odd Cycle Sync Frame ID Register (OSIDn)

The 15 registers from OSID1 to OSID15 store in ascending order the frame IDs of the synchronous frames received during the odd or even communication cycles and are used for clock synchronization up to the gSyncNodeMax limit. Therefore, the smallest synchronous frame ID received is stored in the OSID1 register. When the node transmits a synchronous frame during an odd communication cycle, the OSID1 register stores the transmission synchronous frame ID and RXOA and RXOB are set. The register value is updated during NIT of each odd communication cycle. The register is reset when transiting from the CONFIG state or transiting to the STARTUP state.

BIT_OFFSET	31-16
BIT_NAME	Reserved
ACCESS_TYPE	R0,W0
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	RXOB	RXOA	Reserved				OID[9:8]	
ACCESS_TYPE	R,WX	R,WX	R0,W0				R,WX	
PROT_TYPE	-							
INITIAL_VALUE	0	0	0000				00	

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	OID[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

**[bit31:16] Reserved: Reserved bits**

##### **[bit15] RXOB: Odd cycle sync frame channel B reception bit**

This bit indicates that the synchronous frame corresponding to the synchronous ID during the odd cycle was received on channel B. The node is configured as a sync node of key slot=OID[9:0](OSID1 only).

Value	Description
0	No sync frame received on channel B/Not transmission sync frame
1	Sync frame received on channel B/Transmission sync frame

##### **[bit14] RXOA: Odd cycle sync frame channel A reception bit**

This bit indicates that the synchronous frame corresponding to the synchronous ID was received on channel A during the odd cycle. The node is configured as a sync node of key slot=OID[9:0](OSID1 only).

Value	Description
0	No sync frame received on channel A/Not transmission sync frame
1	Sync frame received on channel A/Transmission sync frame

**[bit13:10] Reserved: Reserved bits**

**[bit9:0] OID[9:0]: Odd cycle sync frame ID bit (vsSyncIDListA,B odd)**

These bits indicate the synchronous frame ID of an odd communication cycle.



#### 4.5.12. Network Management Register [1...3] (NMVn)

There are 3 network management registers that store the generated NM vector (configurable 0 to 12 bytes). The NM vector is generated by the bit-wise OR operation of each NM vector received on each channel (valid static frame where PPI="1"). As long as it is either in the NORMAL\_ACTIVE or NORMAL\_PASSIVE state, the NM vector is updated at the end of each communication cycle. The NM vector is reset when transiting from the CONFIG or to the STARTUP state.

An NMVn register exceeding the set NM vector length is not valid.

BIT_OFFSET	31-0
BIT_NAME	NM
ACCESS_TYPE	R,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000_00000000

**[bit31:0] NM[31:0]: NM vector bit**

Figure 4-2 below shows the assignment of byte data at the network management vector.

**Figure 4-2 Byte Data Assignment at Network Management Vector**

Bit	3	3	2	2	2	2	2	2	2	2	2	2	1	1	1	1	1	1	1	1	1	1	9	8	7	6	5	4	3	2	1	0
Word	1	0	9	8	7	6	5	4	3	2	1	0	9	8	7	6	5	4	3	2	1	0	9	8	7	6	5	4	3	2	1	0
NMV1	Data3								Data2								Data1								Data0							
NMV2	Data7								Data6								Data5								Data4							
NMV3	Data11								Data10								Data9								Data8							

## 4.6. Message Buffer Control Registers

### 4.6.1. Message RAM Configuration Register (MRC)

The message RAM configuration register defines the message buffers to be assigned to the static segment, dynamic segment, and FIFO. Writing to this register is permitted only in the DEFAULT\_CONFIG or CONFIG state.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved					SPLM*	SEC*	
ACCESS_TYPE	R0,W0					R/W	R/W	
PROT_TYPE	-							
INITIAL_VALUE	00000					0	01	

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	LCB*							
ACCESS_TYPE	R/W							
PROT_TYPE	-							
INITIAL_VALUE	10000000							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	FFB*							
ACCESS_TYPE	R/W							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	FDB*							
ACCESS_TYPE	R/W							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

**[bit31:27] Reserved: Reserved bits**

#### **[bit26] SPLM: Sync frame payload multiplex bit**

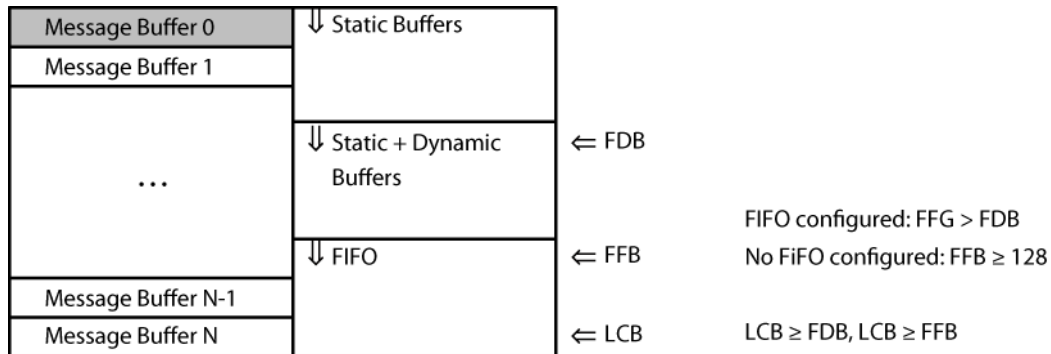
This bit is enabled when the node is set as a sync node (SUCC1:TXSY= 1) or for single-slot mode (SUCC1:TSM="1"). When this bit is set to "1", message buffers 0 and 1 are exclusively for sync frame transmission with different payloads on channel A and channel B, respectively. When this bit is set to "0", the sync frame has the same payload data on both channels and is transmitted from message buffer 0. Be sure to follow the channel filter setting of message buffer 0 and select message buffer 1.

Value	Description
0	Reset locked for both message buffers 0 and 1
1	Reset locked for message buffer 0



**Note:**

- When the node is set as a sync node (SUCC1:TXSY= 1) or in single-slot mode (SUCC1:TSM=1), message buffers 0 and 1 are prepared as sync frame or single-slot frame, respectively, and must be set with the key slot ID complying with the node specification. When the node is not set as sync node (SUCC1:TXSY= 0) or single-slot message buffer 0 or 1, message buffers 0 and 1 are treated as other message buffers, respectively.



Confirm that FDB[7:0], FFB[7:0], and LCB[7:0] are set correctly. Operation is not guaranteed if they are not set correctly. The CC does not check for incorrect locations.

**Note:**

- The maximum number of the header section is 128, which means that a maximum of 128 message buffers can be set. The maximum length of 1 data section is 254 bytes. A different data section length can be set for each message buffer.  
For details, see Section "3.12. Message RAM".  
When more than 2 message buffers are assigned to slot 1 in cycle filtering, it must be located at the beginning of the "static buffer" or "static + dynamic buffer" section.  
With the FlexRay protocol specifications, each node must send the frame to the key slot.  
Therefore, message buffer 0 is reserved for key slot transmission.  
As a result, the maximum number of 127 message buffers can be assigned to FIFO. Nevertheless, non-protocol in the configuration without a transmission slot in the static segment continues to operate.  
Through WRHS2:PLC[6:0] and WRHS3:DP[10:0], set the same data section length and the same payload for all message buffers that belong to FIFO.  
When CC is not in the DEFAULT\_CONFIG or CONFIG state, reset is locked for the message buffers that belong to FIFO.

**[bit25:24] SEC[1:0]: Secure buffer bit**

This bit has no effect in the DEFAULT\_CONFIG or CONFIG state.  
For temporary unlock, see "3.12.4. Parity Error Handling".

Value	Description
00	Reset enabled for message buffers with a buffer number equal to or less than FFB. Exception: Operation message buffer 0 (If SPLM is "1", the message buffer is also 1) is always locked for sync frame transmission or in single-slot mode
01	Reset locked for message buffers with a number equal to or less than FDB. Moreover, message buffers that have a buffer number equal to or greater than FDB and that are set for the static segment cannot be transmitted
10	Reset for all message buffers locked
11	Reset for all message buffers locked

Message buffers that have a buffer number equal to or greater than FDB and that are set for the static segment cannot be transmitted.

**[bit23:16] LCB[7:0]: Last message buffer number bit**

Value	Description
0...127	Number of message buffers is (LCB + 1)
≥ 128	No message buffers set

**[bit15:8] FFB[7:0]: First FIFO buffer number bit**

Value	Description
0	All message buffers assigned to FIFO area
Message buffers from 1-127	FFB to LCB are assigned to FIFO area
≥ 128	No message buffers assigned to FIFO area

**[bit7:0] FDB[7:0]: First dynamic buffer number bit**

Value	Description
0	No buffer group exclusively set for static segment
Message buffers from 1-127	0 to FDB-1 are assigned to the static segment
≥ 128	No buffers set for dynamic segment



#### 4.6.2. FIFO Rejection Filter Register (FRF)

For the FIFO rejection filter register, the bit column to be compared with the channel, frame ID, and cycle count of the reception frame is set. This register, by being combined with the FIFO rejection filter mask register, determines whether the message is rejected by FIFO. Writing to this register is permitted only in the DEFAULT\_CONFIG or CONFIG state.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							RNF*
ACCESS_TYPE	R0,W0							R/W
PROT_TYPE	-							
INITIAL_VALUE	0000000							1

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	RSS*	CYF*						
ACCESS_TYPE	R/W	R/W						
PROT_TYPE	-							
INITIAL_VALUE	1	0000000						

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved			FID[10:6]*				
ACCESS_TYPE	R0,W0			R/W				
PROT_TYPE	-							
INITIAL_VALUE	000			00000				

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	FID[5:0]*						CH*	
ACCESS_TYPE	R/W						R/W	
PROT_TYPE	-							
INITIAL_VALUE	000000						00	

**[bit31:25] Reserved : Reserved bit**

**[bit24] RNF: Null frame rejection bit**

When this bit is set to "1", the received null frames are not stored in FIFO.

Value	Description
0	Null frames stored in FIFO
1	No null frames stored in FIFO

**[bit23] RSS: Message rejection in static segment bit**

When this bit is set to "1", FIFO receives only messages in the dynamic segment.

Value	Description
0	Messages in static segment and dynamic segment received
1	Messages in static segment not received

**[bit22:16] CYF[6:0]: Cycle code filter bits**

A 7-bit cycle counter filter specifies the cycle set and determines the communication cycle to which the frame ID filter and channel filter are applied. Due to the cycle set specified by this register, all frames are not received during the cycle to which the frame ID filter and channel filter are not applied. For details of the cycle counter filter setting, see "3.7.2. Cycle Counter Filtering".

**[bit15:13] Reserved: Reserved bits**

**[bit12:2] FID[10:0]: Frame ID filter bits**

These bits indicate that the frame ID is rejected at FIFO. In additional configuration of register FRFM, the corresponding frame ID filter bit (which induces additional rejected frame IDs) is ignored.

When FRFM.MFID[10:0] is 0, FIFO receives all frame IDs if 0 frame ID is set to this filter value.

0...2047 = Frame ID filter value]

**[bit1:0] CH[1:0]: Channel Filter bits**

Value	Description
00	Reception on both channels
01	Reception only on channel B
10	Reception only on channel A
11	Reception not possible

**Note:**

- When reception is set for both channels, both frames in the static segment are stored in FIFO (from channel A and channel B) even if they are the same frames.





### 4.6.3. FIFO Rejection Filter Mask Register (FRFM)

The FIFO rejection filter mask register specifies the bit to be compared with FRF:FID to perform rejection filtering. When "1" is set to a bit in this register, a comparison with the corresponding FRF:FID bit is not performed. Writing to this register is permitted only in the DEFAULT\_CONFIG or CONFIG state.

BIT_OFFSET	31-16
BIT_NAME	Reserved
ACCESS_TYPE	R0,W0
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved			MFID[10:6]*				
ACCESS_TYPE	R0,W0			R/W				
PROT_TYPE	-							
INITIAL_VALUE	000			00000				

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	MFID[5:0]*						Reserved	
ACCESS_TYPE	R/W						R0,W0	
PROT_TYPE	-							
INITIAL_VALUE	000000						00	

[bit31:13] Reserved: Reserved bits

[bit12:2] MFID[10:0]: Mask frame ID filter bits

Value	Description
0	Corresponding frame ID filter bit used for rejection filtering
1	Ignore corresponding frame ID filter bit

[bit1:0] Reserved: Reserved bits

#### 4.6.4. FIFO Critical Level Register (FCL)

Writing to this register is permitted only in the DEFAULT\_CONFIG or CONFIG state.

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,W0
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	CL*							
ACCESS_TYPE	R/W							
PROT_TYPE	-							
INITIAL_VALUE	10000000							

**[bit31:8] Reserved: Reserved bit**

##### **[bit7:0] CL[7:0]: Critical level bits**

These bits set the critical level flag FSR:RFCL when the value of reception FIFO fill level FSR:RFFL[7:0] is equal to or greater than that of this register. When a value equal to or greater than 128 is set, the critical level flag FSR:RFCL is not set. In addition, the SIR:RFCL signal is also set and an interrupt signal is generated if interrupt is permitted.



## 4.7. Message Buffer Status Registers

### 4.7.1. Message Handler Status Register (MHDS)

The bits to which writing is permitted in this register are cleared when "1" is written. Writing "0" has no effect on the bits. Hard reset will reset this register.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved	MBU						
ACCESS_TYPE	R0,W0	R,WX						
PROT_TYPE	-							
INITIAL_VALUE	0	0000000						

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved	MBT						
ACCESS_TYPE	R0,W0	R,WX						
PROT_TYPE	-							
INITIAL_VALUE	0	0000000						

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved	FMB						
ACCESS_TYPE	R0,W0	R,WX						
PROT_TYPE	-							
INITIAL_VALUE	0	0000000						

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	CRAM	MFMB	FMBD	PTBF2	PTBF1	PMR	POBF	PIBF
ACCESS_TYPE	R,WX	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

**[bit31] Reserved: Reserved bit**

#### **[bit30:24] MBU[6:0]: Updated message buffer number bits**

These bits show the number of the last updated message buffer. The ND and MBS flags in the NDAT1/2/3/4 register and the MBSC1/2/3/4 register corresponding to this message buffer are updated.

**Note:**

- MBT[6:0] and MBU[6:0] are reset when transiting from the CONFIG state or transiting to the STARTUP state.

**[bit23] Reserved: Reserved bit**

**[bit22:16] MBT[6:0]: Transmission message buffer number bits**

These bits show the number of the last message buffer transmitted normally. When the message buffer is set in single-shot mode, each TXR flag of the TXRQ1/2/3/4 register is reset.

**[bit15] Reserved: Reserved bit**

**[bit14:8] FMB[6:0]: Faulty message error bits**

These bits show the message buffer number when a parity error occurs in the following cases:

- When reading message buffer
- When transferring data from input buffer or transient buffer 1 and 2 to message buffer

This value is valid only when any of PIBF, PMR, PTBF1, PTBF2, or FMBD is set. This flag is updated after flag FMBD is reset. While the FMBD flag is being set, it is not updated.

**[bit7] CRAM: Clear of all internal RAM bit**

This bit indicates whether the CHI command CLEAR\_RAM(CMD[3:0]="1100") is running (is writing "0" to all bits in all internal RAM blocks). This bit is set to "1" by hard reset or CHI command CLEAR\_RAM.

Value	Description
0	CHI command CLEAR_RAM not running
1	CHI command CLEAR_RAM running

**[bit6] MFMB: Multiple faulty message buffers detected bit**

Value	Description
0	No other faulty message buffers exist
1	While FMBD flag is set, other faulty message buffers detected

**[bit5] FMBD: Faulty message buffer detected bit**

Value	Description
0	No faulty message buffers exist
1	Message buffer referred to by FMB[6:0] has faulty error due to parity error

**[bit4] PTBF2: Parity error detection at read of transient buffer RAM B bit**

Value	Description
0	No parity error occurs
1	Parity error occurs when transient buffer RAM B is read

**Note:**

- When any of PIBF, POBF, PMR, PTBF1, or PTBF2 changes from "0" to "1", EIR:PERR is set to "1".



**[bit3] PTBF1: Parity error detection at read of transient buffer RAM A bit**

Value	Description
0	No parity error occurs
1	Parity error occurs when transient buffer RAM A is read

**[bit2] PMR: Parity error detection at read of message RAM bit**

Value	Description
0	No parity error occurs
1	Parity error occurs when message RAM is read

**[bit1] POBF: Parity error detection at read of output buffer RAM 1 and 2 bit**

Value	Description
0	No parity error occurs
1	Parity error occurs when output buffer RAM 1 and 2 are read

**[bit0] PIBF: Parity error detection at read of input buffer RAM 1 and 2 bit**

Value	Description
0	No parity error occurs
1	Parity error occurs when input buffer RAM 1 and 2 are read

### 4.7.2. Last Dynamic Transmission Slot Register (LDTs)

This register is reset when transiting from the CONFIG state or transiting to the STARTUP state, or by CHI command CLEAR\_RAMs(CMD[3:0]="1100").

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved					LDTB[10:8]		
ACCESS_TYPE	R0,W0					R,WX		
PROT_TYPE	-							
INITIAL_VALUE	00000					000		

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	LDTB[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved					LDTA[10:8]		
ACCESS_TYPE	R0,W0					R,WX		
PROT_TYPE	-							
INITIAL_VALUE	00000					000		

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	LDTA[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

**[bit31:27] Reserved : Reserved bit**

**[bit26:16] LDTB[10:0]: Last dynamic transmission channel B bit**

These bits indicate the vSlotCounter[B] value in the dynamic segment at the last frame transmission on channel B. It is updated at the end of the dynamic segment and becomes 0 if no frame is transmitted in the dynamic segment.

**[bit15:11] Reserved: Reserved bits**

**[bit10:0] LDTA[10:0]: Last dynamic transmission channel A bit**

These bits indicate the vSlotCounter[A] value in the dynamic segment at the last frame transmission on channel A. It is updated at the end of the dynamic segment and becomes 0 if no frame is transmitted in the dynamic segment.



### 4.7.3. FIFO Status Register (FSR)

The register is reset when transiting from the CONFIG state or transiting to the STARTUP state, or by CHI command CLEAR\_RAMs(CMD[3:0]="1100").

BIT_OFFSET	31-16
BIT_NAME	Reserved
ACCESS_TYPE	R0,W0
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	RFFL							
ACCESS_TYPE	R,WX							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved					RFO	RFCL	RFNE
ACCESS_TYPE	R0,W0					R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	00000					0	0	0

**[bit31:16] Reserved: Reserved bits**

**[bit15:8] RFFL[7:0]: Reception FIFO fill level bits**

These bits show the number of FIFO buffers not yet read by the host. The maximum value is 128.

**[bit7:3] Reserved: Reserved bits**

**[bit2] RFO: Reception FIFO overrun bit**

This bit is set when reception FIFO overrun is detected. In the case of an overrun, the oldest message is overwritten. Moreover, the interrupt flag EIR:RFO is set. The flag is cleared by FIFO read access.

Value	Description
0	Reception FIFO is not overrunning
1	Reception FIFO is overrunning

**[bit1] RFCL: Reception FIFO critical level bit**

This flag is set when the reception FIFO fill level RFFL[7:0] is equal to or higher than the set critical level FCL:CL[7:0]. When the level decreases, the flag is cleared immediately. When RFCL is set from "0" to "1", SIR:RFCL is set to "1". When it is valid, interrupt is generated.

Value	Description
0	Reception FIFO is lower than critical level
1	Reception FIFO is at critical level

**[bit0] RFNE: Reception FIFO not empty bit**

This bit is set when a valid frame (null frame dependent on data or rejection mask) is received and stored in FIFO. In addition, the interrupt flag SIR:RFNE is set. The bit is reset after the host reads all messages from FIFO.

Value	Description
0	Reception FIFO empty
1	Reception FIFO not empty





#### 4.7.4. Message Handler Constraints Flags (MHDF)

The bits to which writing is permitted in this register are cleared by writing "1". Writing "0" to these bits has no effect on the bits. Hard reset will clear this register.

The register is reset when transiting from the CONFIG state or transiting to the STARTUP state, or by CHI command CLEAR\_RAM (CMD[3:0]= 1100).

BIT_OFFSET	31-16
BIT_NAME	Reserved
ACCESS_TYPE	R0,W0
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved							WAHP
ACCESS_TYPE	R0,W0							R/W
PROT_TYPE	-							
INITIAL_VALUE	0000000							0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	TNSB	TNSA	TBFB	TBFA	FNFB	FNFA	SNUB	SNUA
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

**[bit31:9] Reserved: Reserved bits**

#### **[bit8] WAHP: Header partition write bit**

This bit is set when the message handler attempts to write message data to the header partition of the message RAM due to the incomplete location of a message buffer that is in a state other than the DEFAULT\_CONFIG and CONFIG state. Writing is not performed to protect the header partition from unintended write accesses.

Value	Description
0	Not written to header partition
1	Written to header partition

**Note:**

- To change the SNUA, SNUB, FNFA, FNFB, TBFA, TBFB, TNSA, TNSB, and WAHP signal from "0" to "1", set the EIR:MHF interrupt flag to "1".

#### **[bit7] TNSB: Channel B transmission not started bit**

This bit is set when the message handler is not prepared to start the scheduled transmission on channel B of the action point of the configured slot.

Value	Description
0	Transmission on channel B started
1	Transmission on channel B not started

**[bit6] TNSA: Channel A transmission not started bit**

This bit is set when the message handler is not prepared to start the scheduled transmission on channel A of the action point of the configured slot.

Value	Description
0	Transmission on channel A started
1	Transmission on channel A not started

**[bit5] TBFB: Channel B transient buffer access failure bit**

This bit is set when the read or write access to TBF B requested by PRT B does not complete within the available time.

Value	Description
0	No TBF B access failure
1	TBF B access failure

**[bit4] TBFA: Channel A transient buffer access failure bit**

This bit is set when the read or write access to TBF A requested by PRT A does not complete within the available time.

Value	Description
0	No TBF A access failure
1	TBF A access failure

**[bit3] FNFB: Channel B sequence not finished bit**

This bit is set when the message handler cannot end the find sequence (message RAM scan for message buffer match) on channel B due to overload.

Value	Description
0	No unfinished sequence detected on channel B
1	Unfinished sequence detected on channel B

**[bit2] FNFA: Channel A sequence not finished bit**

This bit is set when the message handler cannot end the find sequence (message RAM scan for message buffer match) on channel A due to overload.

Value	Description
0	No unfinished sequence detected on channel A
1	Unfinished sequence detected on channel A

**[bit1] SNUB: Channel B status not updated bit**

This bit is set when the message handler cannot update message buffer status MBS on channel B due to overload.

Value	Description
0	No overload occurred when updating MBS on channel B
1	MBS on channel B not updated



**[bit0] SNUA: Channel A status not updated bit**

This bit is set when the message handler cannot update message buffer status MBS on channel A due to overload.

Value	Description
0	No overload occurred when updating MBS on channel A
1	MBS on channel A not updated

#### 4.7.5. Transmission Request Register 1/2/3/4 (TXRQ1/2/3/4)

These 4 registers reflect the TXR flag status of all set message buffers. When the number of the set message buffer is lower than 128, the remaining TXR flags have no meaning.

##### (1) TXRQ4

BIT_OFFSET	31-0
BIT_NAME	TXR[127:96]
ACCESS_TYPE	R,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000_00000000

##### [bit31:0] TXR[127:65]: Transmission request bits

When the flag is set to "1", the corresponding message buffer is set as a transmission buffer to indicate that the transmission is in progress for such message buffer. In single-shot mode, the flag is reset after the transmission completes.

##### (2) TXRQ3

BIT_OFFSET	31-0
BIT_NAME	TXR[95:64]
ACCESS_TYPE	R,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000_00000000

##### [bit31:0] TXR[95:64]: Transmission request bits

When the flag is set to "1", the corresponding message buffer is set as a transmission buffer to indicate that the transmission is in progress for such message buffer. In single-shot mode, the flag is reset after the transmission completes.

##### (3) TXRQ2

BIT_OFFSET	31-0
BIT_NAME	TXR[63:32]
ACCESS_TYPE	R,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000_00000000

##### [bit31:0] TXR[63:32]: Transmission request bits

When the flag is set to "1", the corresponding message buffer is set as a transmission buffer to indicate that the transmission is in progress for such message buffer. In single-shot mode, the flag is reset after the transmission completes.



**(4) TXRQ1**

BIT_OFFSET	31-0
BIT_NAME	TXR[31:0]
ACCESS_TYPE	R,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000_00000000

**[bit31:0] TXR[31:0]: Transmission request bits**

When the flag is set to "1", the corresponding message buffer is set as a transmission buffer to indicate that the transmission is in progress for such message buffer. In single-shot mode, the flag is reset after the transmission completes.

#### 4.7.6. New Data Register 1/2/3/4 (NDAT1/2/3/4)

These 4 registers reflect the ND flag status of all set message buffers. If a message buffer is set as a transmission buffer, the ND flag corresponding to the message buffer has no meaning. When the number of the set message buffer is lower than 128, the remaining ND flags have no meaning.

This register is reset when transiting from the CONFIG state or transiting to the STARTUP state.

##### (1) NDAT4

BIT_OFFSET	31-0
BIT_NAME	ND[127:96]
ACCESS_TYPE	R,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000_00000000

##### [bit31:0] ND[127:96]: New Data bits

This flag is set to "1" when the data section of each message buffer is updated by the valid reception frame that passed the set message buffer filter. With the exception of message buffers for the reception FIFO, no flags are set for the reception of invalid frames. When the header section of the corresponding message buffer is reconfigured, or when the data section is transferred to the output buffer, the flag is cleared to "0".

##### (2) NDAT3

BIT_OFFSET	31-0
BIT_NAME	ND[95:64]
ACCESS_TYPE	R,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000_00000000

##### [bit31:0] ND[95:64]: New Data bits

This flag is set to "1" when the data section of each message buffer is updated by the valid reception frame that passed the set message buffer filter. With the exception of message buffers for the reception FIFO, no flags are set for the reception of invalid frames. When the header section of the corresponding message buffer is reconfigured, or when the data section is transferred to the output buffer, the flag is cleared to "0".

##### (3) NDAT2

BIT_OFFSET	31-0
BIT_NAME	ND[63:32]
ACCESS_TYPE	R,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000_00000000

##### [bit31:0] ND[63:32]: New Data bits

This flag is set to "1" when the data section of each message buffer is updated by the valid reception frame that passed the set message buffer filter. With the exception of message buffers for the reception FIFO, no flags are set for the reception of invalid frames. When the header section of the corresponding message buffer is reconfigured, or when the data section is transferred to the output buffer, the flag is cleared to "0".



**(4) NDAT1**

BIT_OFFSET	31-0
BIT_NAME	ND[31:0]
ACCESS_TYPE	R,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000_00000000

**[bit31:0] ND[127:0]: New Data bit**

This flag is set to "1" when the data section of each message buffer is updated by the valid reception frame that passed the set message buffer filter. With the exception of message buffers for the reception FIFO, no flags are set for the reception of invalid frames. When the header section of the corresponding message buffer is reconfigured, or when the data section is transferred to the output buffer, the flag is cleared to "0".

#### 4.7.7. Message Buffer Status Changed Register 1/2/3/4 (MBSC)

These 4 registers reflect the MBC flag status of all set message buffers. If the number of the set message buffer is lower than 128, the remaining MBC flags have no meaning.

This register is reset when transiting from the CONFIG state or transiting to the STARTUP state.

##### (1) MBSC4

BIT_OFFSET	31-0
BIT_NAME	MBC[127:96]
ACCESS_TYPE	R,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000_00000000

##### [bit31 to bit0] MBC[127:96]: Message buffer status changed bit

This flag is set to "1" when the status flag of each message buffer (VFRA, VFRB, SEOA, SEOB, CEOA, CEOb, SVOA, SVOB, TCIA, TCIB, ESA, ESB, MLST, FTA, FTB) is changed. The flag is cleared to "0" when the header section of the corresponding message buffer is reconfigured or when the data section is transferred to the output buffer.

##### (2) MBSC3

BIT_OFFSET	31-0
BIT_NAME	MBC[95:64]
ACCESS_TYPE	R,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000_00000000

##### [bit31:0] MBC[95:64]: Message buffer status changed

This flag is set to "1" when the status flag of each message buffer (VFRA, VFRB, SEOA, SEOB, CEOA, CEOb, SVOA, SVOB, TCIA, TCIB, ESA, ESB, MLST, FTA, FTB) is changed. The flag is cleared to "0" when the header section of the corresponding message buffer is reconfigured or when the data section is transferred to the output buffer.

##### (3) MBSC2

BIT_OFFSET	31-0
BIT_NAME	MBC[63:32]
ACCESS_TYPE	R,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000_00000000

##### [bit31:0] MBC[63:32]: Message buffer status changed

This flag is set to "1" when the status flag of each message buffer (VFRA, VFRB, SEOA, SEOB, CEOA, CEOb, SVOA, SVOB, TCIA, TCIB, ESA, ESB, MLST, FTA, FTB) is changed. The flag is cleared to "0" when the header section of the corresponding message buffer is reconfigured or when the data section is transferred to the output buffer.





**(4) MBSC1**

BIT_OFFSET	31-0
BIT_NAME	MBC[31:0]
ACCESS_TYPE	R,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000_00000000

**[bit31:0] MBC[127:0]: Message buffer status changed bit**

This flag is set to "1" when the status flag of each message buffer (VFRA, VFRB, SEOA, SEOB, CEOA, CEOB, SVOA, SVOB, TCIA, TCIB, ESA, ESB, MLST, FTA, FTB) is changed. The flag is cleared to "0" when the header section of the corresponding message buffer is reconfigured or when the data section is transferred to the output buffer.

## 4.8. Identification Registers

### 4.8.1. Core Release Register (CREL)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	REL				STEP[7:4]			
ACCESS_TYPE	R,WX				R,WX			
PROT_TYPE	-							
INITIAL_VALUE	0001				0000			

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	STEP[3:0]				YEAR			
ACCESS_TYPE	R,WX				R,WX			
PROT_TYPE	-							
INITIAL_VALUE	0011				1001			

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	MON							
ACCESS_TYPE	R,WX							
PROT_TYPE	-							
INITIAL_VALUE	00000010							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	DAY							
ACCESS_TYPE	R,WX							
PROT_TYPE	-							
INITIAL_VALUE	00000110							

**[bit31:28] REL[3:0]: Release bits**

Single digit (BCD) 0x1 is read.

**[bit27:20] STEP[7:0]: Release step bits**

Double digits (BCD) 0x3 is read.

**[bit19:16] YEAR[3:0]: Year bits**

Single digit (BCD) 0x9 is read.

**[bit15:8] MON[7:0]: Month bits**

Double digits (BCD) 0x02 is read

**[bit7:0] DAY[7:0]: Date bits**

Double digits (BCD) 0x06 is read.



## 4.8.2. Endian Register (ENDN)

BIT_OFFSET	31-0
BIT_NAME	ETV
ACCESS_TYPE	R,WX
PROT_TYPE	-
INITIAL_VALUE	10000111_01100101_01000011_00100001

**[bit31:0] ETV[31:0]: Endian test value bits**

Test value 0x87654321

## 4.9. Input Buffer

The input buffer has a double-buffer configuration consisting of the input buffer host and the input buffer shadow. It is transferred from the input buffer shadow to the message RAM while the host is permitted to write to the input buffer. The input buffer stores the header section and data section to be transferred to the selected message buffer. Moreover, the buffer is used for the message buffer configuration of the message RAM and the data section update of the transmission buffer.

When updating the header section of the message buffer in the message RAM, the message buffer status is automatically reset to 0 as is shown in Section "3.11.2. Host Access to Message RAM".

Change the header section of the message buffer that belongs to the reception FIFO only in the DEFAULT\_CONFIG or CONFIG state.

A detailed explanation on the data transfer between input buffer (IBF) and message RAM is given in "a) Data Transfer from Input Buffer to Message RAM".



#### 4.9.1. Write Data Section Register (WRDSn [1...64])

This register sets the data to be transferred to the data section of the message buffer. This data (DWn) is written in message RAM from DW1 (byte 0, byte 1) to DWPL (PL=data number in 2-byte units, which is defined by payload length) according to the order of transmission.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	MD[31:24]							
ACCESS_TYPE	R/W							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	MD[23:16]							
ACCESS_TYPE	R/W							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	MD[15:8]							
ACCESS_TYPE	R/W							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	MD[7:0]							
ACCESS_TYPE	R/W							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

##### [bit31:0] MD[31:0]: Message Data bit

ページ : 1920

MD[31:24] = DW2n, byte4n-1

MD[23:16] = DW2n, byte4n-2

MD[15:8] = DW2n-1, byte4n-3

MD[7:0] = DW2n-1, byte4n-4

##### Note:

- DW127 is located at WRDS64:MD[15:0]. In this case, WRDS64:MD[31:16] is not used (undefined data). Input buffer RAM is initialized to 0 by hard reset end or by CHI command CLEAR\_RAMs (CMD[3:0] = 1100).

The transfer order of FlexRay bus is from each msb bit of WRDSn[7:0], WRDSn[15:8], WRDSn[23:16], and WRDSn[31:24].

To check on how to adjust to the endian of host CPU, see the register ENDN.

## 4.9.2. Write Header Section Register 1 (WRHS1)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved		MBI	TXM	PPIT	CFG	CHB	CHA
ACCESS_TYPE	R0,W0		R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	00		0	0	0	0	0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved	CYC						
ACCESS_TYPE	R0,W0	R/W						
PROT_TYPE	-							
INITIAL_VALUE	0	0000000						

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved					FID[10:8]		
ACCESS_TYPE	R0,W0					R/W		
PROT_TYPE	-							
INITIAL_VALUE	00000					000		

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	FID[7:0]							
ACCESS_TYPE	R/W							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

**[bit31:30] Reserved : Reserved bits**

**[bit29] MBI: Message buffer interrupt bit**

This bit enables transmission and reception interrupts to each message buffer. After a message is received to the reception buffer, SIR:RXI or SIR:MBSI is set to "1". After the message is normally sent from the transmission buffer, the SIR:TXI flag is set to "1".

Value	Description
0	Invalid transmission and reception interrupt to the corresponding message buffer
1	Valid transmission and reception interrupt to the corresponding message buffer

**[bit28] TXM: Transmission mode bit**

This bit is for selecting the transmission mode (See Section "3.8.3. Transmission Buffer").

Value	Description
0	Continuous mode
1	Single-shot mode



**[bit27] PPIT: Payload preamble indicator transmission bit**

This bit is used to control the payload preamble indicator status in transmission frame. When this bit is set to the static message buffer, each message buffer retains network management information. When this bit is set to the dynamic message buffer, the first 2 bytes of the payload segment are used for message ID filtering. The message ID filtering of the reception frame is not supported by FlexRay controller.

Value	Description
0	Does not set payload preamble indicator
1	Set payload preamble indicator

**[bit26] CFG: Message buffer configuration bit**

This bit is used to set each buffer as the transmission or reception buffer. The bit is invalid for the message buffer that belongs to the reception FIFO.

Value	Description
0	The corresponding buffer is set as reception buffer
1	The corresponding buffer is set as transmission buffer

**[bit25:24] CHA, CHB: Channel filter control bits**

These 2-bit channel filtering fields, related to each buffer, have the function of the filter for the reception buffer and the control field for the transmission buffer.

CHA	CHB	Transmission Buffer (Transmission Frame)	Reception Buffer (Save Reception Frame)
1	1	both channels (static segment only)	channel A or B (store first semantically valid frame, static segment only)
1	0	channel A	channel A
0	1	channel B	channel B
0	0	no transmission	ignore frame

**Note:**

- When a message buffer is set for the dynamic segment and both bits of the channel filtering control (CHA, CHB) are set to "1", no frames are transmitted and reception frames are ignored. (Same function as CHA = CHB = 0)

**[bit23] Reserved: Reserved bit**

**[bit22:16] CYC[6:0]: Cycle code bits**

These 7-bit codes determine the cycle set to be used for cycle counter filtering. For details of the cycle code setting, see Section "3.7.2. Cycle Counter Filtering".

**[bit15:11] Reserved: Reserved bits**

**[bit10:0] FID[10:0]: Frame ID bits**

These bits indicate the frame ID of the selected message buffer. The frame ID defines the slot number for each transmission and reception of messages. The message buffer of frame ID="0" is invalid.

### 4.9.3. Write Header Section Register 2 (WRHS2)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,W0							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved	PLC						
ACCESS_TYPE	R0,W0	R/W						
PROT_TYPE	-							
INITIAL_VALUE	0	0000000						

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved					CRC[10:8]		
ACCESS_TYPE	R0,W0					R/W		
PROT_TYPE	-							
INITIAL_VALUE	00000					000		

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	CRC[7:0]							
ACCESS_TYPE	R/W							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

**[bit31:23] Reserved: Reserved bits**

**[bit22:16] PLC[6:0]: Configured payload length bits**

These bits indicate the length of the data section (number in 2-byte units) set by the host. The static frame payload length set at MHDC:SFDL[6:0] in the static segment defines the payload length of all static frames. When the payload length set by PLC[6:0] is shorter than that set by MHDC:SFDL[6:0], a padding byte is inserted to guarantee the payload length of the static frame. Padding byte is indicated by "0" (See Section "3.8.3. Transmission Buffer").

**[bit15:11] Reserved: Reserved bits**

**[bit10:0] CRC[10:0]: Header CRC bit (vRF!Header!HeaderCRC)**

Reception buffer: Setting is not necessary.

Transmission buffer: Header CRC is calculated by the host for setting.

For calculation of header CRC, the payload length of the frame must be transmitted to the host. The payload length of all frames in the static segment is set at MHDC:SFDL[6:0].





#### 4.9.4. Write Header Section Register 3 (WRHS3)

BIT_OFFSET	31-16
BIT_NAME	Reserved
ACCESS_TYPE	R0,W0
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved					DP[10:8]		
ACCESS_TYPE	R0,W0					R/W		
PROT_TYPE	-							
INITIAL_VALUE	00000					000		

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	DP[7:0]							
ACCESS_TYPE	R/W							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

**[bit31:11] Reserved: Reserved bits**

**[bit10:0] DP[10:0]: Data pointer bits**

These bits indicate the pointer to the first 32-bit data of the data section of message buffer.

#### 4.9.5. Input Buffer Command Mask Register (IBCM)

This register sets the update method of the message buffer selected by the IBCR register. When the IBF host and IBF shadow are exchanged, mask bit LSHS, LDSS, STXRH, and mask bit LHSS, LDSS, and STXRS are also exchanged.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,W0							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved					STXRS	LDSS	LHSS
ACCESS_TYPE	R0,W0					R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	00000					0	0	0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved							
ACCESS_TYPE	R0,W0							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved					STXRH	LDSS	LHSS
ACCESS_TYPE	R0,W0					R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	00000					0	0	0

[bit31:19] Reserved: Reserved bits

[bit18] STXRS: Transmission request flag shadow setting bit

Value	Description
0	TXR flag reset
1	Set TXR flag. Message in transmission buffer being released, or these operations complete

[bit17] LDSS: Data section shadow load bit

Value	Description
0	Data section not transferred from input buffer to message RAM
1	Data section being transferred from input buffer to message RAM, or transfer completed

[bit16] LHSS: Header section shadow load bit

Value	Description
0	Header section not transferred from input buffer to message RAM
1	Header section being transferred from input buffer to message RAM, or transfer completed



**[bit15:3] Reserved: Reserved bits**

**[bit2] STXRH: Transmission request flag host setting bit**

When this bit is set to "1", the TXR flag of the selected message buffer is set to "1" in the TXRQ1/2/3/4 register and the message in the transmission buffer is released. This flag is cleared after completion of the transmission in single-shot mode.

Value	Description
0	Reset TXR flag
1	Set TXR flag. Message in transmission buffer released

**[bit1] LDSH: Data section host load bit**

Value	Description
0	Data section not transferred
1	Data section transferred from input buffer to message RAM

**[bit0] LHSH: Header section host load bit**

Value	Description
0	Header section not transferred
1	Header section transferred from input buffer to message RAM

#### 4.9.6. Input Buffer Command Request Register (IBCR)

When writing the target message buffer number in the message RAM to IBRH[6:0], the IBF host and IBF shadow are exchanged. Moreover, the message number to be stored in IBRH[6:0] and IBRS[6:0] are exchanged, too. (See "a) Data Transfer from Input Buffer to Message RAM" in Section "3.11.2. Host Access to Message RAM".)

IBSYS bit is set to "1" in this write operation. Then, the message handler starts to transfer the content of the IBF shadow to the message buffer in the message RAM selected by IBRS[6:0].

While transferring data from the IBF shadow to the message buffer in the message RAM, the following transmission message can be written in the IBF host. IBSYS is cleared to "0" after transfer between the IBF shadow and the message RAM completes. By writing the target message buffer number of the following transmission message to IBRH[6:0], the following transfer to the message RAM starts.

When write access to IBRH[6:0] occurs during IBSYS="1", IBSYH is set to "1".

After the current data transfer from the IBF shadow to the message RAM completes, the IBF host and the IBF shadow are exchanged and, at the same time, the message buffer numbers to be stored in IBRH[6:0] and IBRS[6:0] are also exchanged. At this time, IBSYH is reset to "0". If the "1" setting remains in IBSYS, the following transfer to the message RAM starts.

When writing to this input buffer register while "1" is set to both IBSYS and IBSYH, error flag EIR:IBA is set to "1". In this case, the input buffer is not changed.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	IBSYS	Reserved						
ACCESS_TYPE	R,WX	R0,W0						
PROT_TYPE	-							
INITIAL_VALUE	0	0000000						

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved	IBRS						
ACCESS_TYPE	R0,W0	R,WX						
PROT_TYPE	-							
INITIAL_VALUE	0	0000000						

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	IBSYH	Reserved						
ACCESS_TYPE	R,WX	R0,W0						
PROT_TYPE	-							
INITIAL_VALUE	0	0000000						

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved	IBRH						
ACCESS_TYPE	R0,W0	R/W						
PROT_TYPE	-							
INITIAL_VALUE	0	0000000						



**[bit31] IBSYS: Input buffer shadow busy bit**

This flag is set to "1" when writing to IBRH[6:0]. IBSYS is cleared to "0" when transfer between the IBF shadow and the message RAM completes.

IBSYS is cleared to "0" when transfer between the RAMs completes.

Value	Description
0	Transfer between IBF shadow and message RAM completed
1	Transfer between IBF shadow and message RAM in progress

**[bit30:23] Reserved: Reserved bits**

**[bit22:16] IBRS[6:0]: Input buffer shadow transfer request bits**

These bits indicate the current updated or last updated target message buffer number. Valid values are 0x00 to 0x7F (0 to 127).

**[bit15] IBSYH: Input buffer host busy bit**

This flag is set to "1" when writing to IBRH[6:0] is performed while IBSYS is still "1". This flag is cleared to "0" after the current data transfer completes between the IBF shadow and the message RAM.

Value	Description
0	Message transfer not suspended
1	Message transfer suspended

**[bit14:7] Reserved: Reserved bits**

**[bit6:0] IBRH[6:0]: Input buffer host transfer request bits**

These bits select the target message buffer number in the message RAM for data transfer from the input buffer. Valid values are 0x00 to 0x7F (0 to 127).

## 4.10. Output Buffer

The output buffer has a double buffer configuration consisting of the output buffer host and the output buffer shadow, and is used for reading the message buffer from the message RAM. While the host is permitted to read the output buffer host, the output buffer transfers the selected message buffer from the message RAM to the output buffer shadow. For details of data transfer between the message RAM and the output buffer (OBF), see "b) Data Transmission from Message RAM to Output Buffer" in Section "3.11.2. Host Access to Message RAM".

### 4.10.1. Read Data Section Register (RDDS<sub>n</sub>)

This register sets the data read from the data section of the message buffer. This data (DW<sub>n</sub>) is read from the message RAM from DW1 (byte 0, byte 1) to DWPL (PL = data number in 2-byte units, which is defined by payload length) according to the reception order.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	MD[31:24]							
ACCESS_TYPE	R,WX							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	MD[23:16]							
ACCESS_TYPE	R,WX							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	MD[15:8]							
ACCESS_TYPE	R,WX							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	MD[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

#### [bit31:0] MD[31:0]: Message data bit

MD[31:24] = DW<sub>2n</sub>, byte<sub>4n-1</sub>

MD[23:16] = DW<sub>2n</sub>, byte<sub>4n-2</sub>

MD[15:8] = DW<sub>2n-1</sub>, byte<sub>4n-3</sub>

MD[7:0] = DW<sub>2n-1</sub>, byte<sub>4n-4</sub>

#### Note:

- DW127 is located at RDDS64:MD[15:0]. In this case, RDDS64:MD[31:16] is not used (undefined data). Output buffer RAM is initialized to 0 by hard reset end or by CHI command CLEAR\_RAM(CMD[3:0]="1100").



### 4.10.2. Read Header Section Register 1 (RDHS1)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved		MBI	TXM	PPIT	CFG	CHB	CHA
ACCESS_TYPE	R0,W0		R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	00		0	0	0	0	0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved	CYC						
ACCESS_TYPE	R0,W0	R,WX						
PROT_TYPE	-							
INITIAL_VALUE	0	0000000						

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved					FID[10:8]		
ACCESS_TYPE	R0,W0					R,WX		
PROT_TYPE	-							
INITIAL_VALUE	00000					000		

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	FID[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

The following are the values to be set through WRHS1.

**[bit31:30] Reserved: Reserved bits**

**[bit29] MBI: Message buffer interrupt bit**

**[bit28] TXM: Transmission mode bit**

**[bit27] PPIT: Payload preamble indicator transmit bit**

**[bit26] CFG: Message buffer configuration bit**

**[bit25:24] CHA,CHB: Channel filter control bits**

**[bit23] Reserved: Reserved bits**

**[bit22:16] CYC[6:0]: Cycle code bits**

**[bit15:11] Reserved: Reserved bits**

**[bit10:0] FID[10:0]: Frame ID bits**

When the message buffer read from the message RAM belongs to the reception FIFO, FID[10:0] retains the received frame ID while CYC[6:0], CHA, CHB, CFG, PPIT, TXM, MBI is reset to "0".

### 4.10.3. Read Header Section Register 2 (RDHS2)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved	PLR						
ACCESS_TYPE	R0,W0	R,WX						
PROT_TYPE	-							
INITIAL_VALUE	0	0000000						

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved	PLC						
ACCESS_TYPE	R0,W0	R,WX						
PROT_TYPE	-							
INITIAL_VALUE	0	0000000						

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved					CRC[10:8]		
ACCESS_TYPE	R0,W0					R,WX		
PROT_TYPE	-							
INITIAL_VALUE	00000					000		

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	CRC[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

**[bit31] Reserved: Reserved bit**

#### **[bit30:24] PLR[6:0]: Reception payload length bits (vRF!Header!Length)**

These bits indicate the value of the payload length updated by the reception frame.

With respect to the reception payload length and configured payload length, the following behavior takes place when a message is stored in the message buffer:

Value	Description
PLR[6:0] > PLC[6:0]	The payload data stored in message buffer is shortened to the payload length set at PLC[6:0] or PLC[6:0]+1
PLR[6:0] ≤ PLC[6:0]	The reception payload data is stored in the data section of the message buffer. The remaining data bytes of the data section set at PLC[6:0] are filled with undefined data
PLR[6:0] = 0	The data section of message buffer is filled with undefined data
PLC[6:0] = 0	The message buffer has no data section. No data is stored in the data section of the message buffer

**Note:**

- The message RAM has a 4-byte configuration. When the received data is stored in the data section of the message buffer, the number of data words in 2-byte units, which is written to the message buffer, is rounded to the next even number, PLC[6:0]. Set PLC[6:0] so that it is the same for all message buffers for the reception FIFO. Header 2 is updated from only the data frames.





**[bit23] Reserved: Reserved bit**

**[bit22:16] PLC[6:0]: Configured payload length bits**

These bits indicate the data section length (number in 2-byte units) set by the host.

**[bit15:11] Reserved: Reserved bits**

**[bit10:0] CRC[10:0]: Header CRC bit (vRF!Header!HeaderCRC)**

Reception buffer: The header CRC is updated by the reception frame.

Transmission buffer: The header CRC set by the message transfer from the input buffer is displayed.

#### 4.10.4. Read Header Section Register 3 (RDHS3)

BITS_OFFSET	31	30	29	28	27	26	25	24
BITS_NAME	Reserved		RES	PPI	NFI	SYN	SFI	RCI
ACCESS_TYPE	R0,W0		R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	00		0	0	0	0	0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved		RCC					
ACCESS_TYPE	R0,W0		R,WX					
PROT_TYPE	-							
INITIAL_VALUE	00		000000					

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved					DP[10:8]		
ACCESS_TYPE	R0,W0					R,WX		
PROT_TYPE	-							
INITIAL_VALUE	00000					000		

BITS_OFFSET	7	6	5	4	3	2	1	0
BITS_NAME	DP[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

**[bit31:30] Reserved: Reserved bits**

**[bit29] RES: Reserved bit (vRF!Header!Reserved)**

This bit shows the received reserved bit status.

**Note:**

- Header 3 is updated only from data frame.

**[bit28] PPI: Payload preamble indicator bit (vRF!Header!PPIndicator)**

This bit shows whether the network management vector or the message ID is included in the payload segment in the received frame.

Value	Description
0	Neither network management vector nor message ID included in payload segment of received frame
1	Static segment: Network management vector included in beginning of payload Dynamic segment: Message ID included in beginning of payload



**[bit27] NFI: Null frame indicator bit (vRF!Header!NFIndicator)**

When this bit is "0", there is no valid data in the payload segment of the received frame.

Value	Description
0	Received frame is null frame
1	Received frame is not null frame

**[bit26] SYN: Sync frame indicator bit (vRF!Header!SyFIndicator)**

This bit shows that the received frame is a sync frame.

Value	Description
0	Received frame is not sync frame
1	Received frame is sync frame

**[bit25] SFI: Startup frame indicator bit (vRF!Header!SuFIndicator)**

This bit shows that the received frame is a startup frame.

Value	Description
0	Received frame is not startup frame
1	Received frame is startup frame

**[bit24] RCI: Reception channel indicator bit (vSS!Channel)**

This bit shows from which channel the reception frame updating each reception buffer is received.

Value	Description
0	Frame received on channel B
1	Frame received on channel A

**[bit23:22] Reserved: Reserved bits**

**[bit21:16] RCC[5:0]: Reception cycle counter bits (vRF!Header!CycleCount)**

These bits indicate the cycle counter value updated by received frame.

**[bit15:11] Reserved: Reserved bits**

**[bit10:0] DP[10:0]: Data pointer bits**

These bits show the pointer to the first 32-bit data of the data section of message buffer.

#### 4.10.5. Message Buffer Status Register (MBS)

The message buffer status is updated with respect to the assigned channel at the end of the slot that follows the slot assigned to that message buffer. When only 1 channel (A or B) is assigned to a certain message buffer, the status flag of the other channel is cleared to "0". When both channels are assigned to 1 message buffer, the status flags of both channels are updated.

The message buffer status always indicates the status of the latest slot assigned to the message buffer. When the host updates the message buffer through the input buffer, all MBS flags are reset even though IBCM bits are set. For details of transmission and reception filtering, see Section " 3.7. Filtering and Masking", "3.8. Transmission Procedure", and "3.9. Reception Procedure". Whenever the message handler changes any of the flags VFRA, VFRA, SEOA, SEOB, CEOA, SVOA, SVOB, TCIA, TCIB, ESA, ESB, MLST, FTA, or FTB, the MBC flag of each message buffer of register MBSC1/2/3/4 is always set.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved		RESS	PPIS	NFIS	SYNS	SFIS	RCIS
ACCESS_TYPE	R0,W0		R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	00		0	0	0	0	0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved		CCS					
ACCESS_TYPE	R0,W0		R,WX					
PROT_TYPE	-							
INITIAL_VALUE	00		000000					

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	FTB	FTA	Reserved	MLST	ESB	ESA	TCIB	TCIA
ACCESS_TYPE	R,WX	R,WX	R0,W0	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	SVOB	SVOA	CEOB	CEOA	SEOB	SEOA	VFRB	VFRA
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

##### a) Status flag regarding reception buffer

**[bit31:30] Reserved: Reserved bits**

##### **[bit29] RESS: Reserved bit status bit (vRF!Header!Reserved)**

This bit indicates the received reserved bit status. The reserved bit is transmitted as "0".



**[bit28] PPIS: Payload preamble indicator status bit (vRF!Header!PPIndicator)**

The payload preamble indicator defines whether the network management vector or message ID is included in the payload segment of the reception frame.

Value	Description
0	Not included
1	Static segment: Network management included Dynamic segment: Message ID included

**[bit27] NFIS: Null frame indicator status bit (vRF!Header!NFIndicator)**

Value	Description
0	Reception frame is null frame
1	Reception frame is not null frame

**[bit26] SYNS: Sync frame indicator status bit (vRF!Header!SyFIndicator)**

Value	Description
0	No sync frame received
1	Sync frame received

**[bit25] SFIS: Startup frame indicator status bit (vRF!Header!SuFIndicator)**

Value	Description
0	No startup frame received
1	Startup frame received

**[bit24] RCIS: Channel indicator status reception bit (vSS!Channel)**

Value	Description
0	Frame received on channel B
1	Frame received on channel A

**[bit23:22] Reserved: Reserved bits**

**[bit21:16] CCS[5:0]: Cycle count status bits**

This bit performs cycle count when status is updated.

In the reception buffer (CFG="0"), the following status bits are updated from both valid and invalid frame data. When a valid frame is not received, the previous value is maintained.

The bits have no meaning for the transmission buffer and thus are ignored.

**[bit15] FTB: Channel B frame transmission bit**

This bit indicates that the data frame was transmitted to channel B.

Value	Description
0	No data frame transmitted on channel B
1	Data frame transmitted on channel B

**[bit14] FTA: Channel A frame transmission bit**

This bit indicates that the data frame was transmitted to channel A.

Value	Description
0	No data frame transmitted on channel A
1	Data frame transmitted on channel A

**Note:**

- Only the host can reset FTA and FTB. Therefore, cycle count status CCS[5:0] is valid when the bit is set to "1".

**[bit13] Reserved: Reserved bit**

**[bit12] MLST: Message lost bit**

This bit is set when the message is not read before the message buffer is overwritten by a new message. Null frame reception has no effect except for the message buffer to the reception FIFO. The flag is reset when new message is saved in the message buffer after the ND flag of the message buffer is reset by writing the message buffer to IBF or reading the message buffer from OBF.

Value	Description
0	No message lost
1	Unread message overwritten

**[bit11] ESB: Channel B empty slot bit**

An empty slot means that the bus is in the idle state. In other words, no frame transmission is detected. This state is checked in static and dynamic slots.

Value	Description
0	Bus is not in idle state in assigned slot on channel B
1	Bus is in idle state in assigned slot on channel B

**[bit10] ESA: Channel A empty slot bit**

An empty slot means that the bus is in the idle state. In other words, no frame transmission is detected. This state is checked in static and dynamic slots.

Value	Description
0	Bus is not in idle state in assigned slot on channel A
1	Bus is in idle state in assigned slot on channel A

**b) Status flag regarding transmission buffer**

**[bit9] TCIB: Channel A transmission collision indicator bit (vSS!TxConflictB)**

This bit is set to "1" when a transmission collision is detected on channel B.

Value	Description
0	No transmission collision detected on channel B
1	Transmission collision detected on channel B



**[bit8] TCIA: Channel A transmission collision indicator bit (vSS!TxConflictA)**

This bit is set to "1" when a transmission collision is detected on channel A.

Value	Description
0	No transmission collision detected on channel A
1	Transmission collision detected on channel A

**c) Status flag for reception and transmission buffer**

**[bit7] SVOB: Channel B boundary violation bit (vSS!BViolationB)**

This bit indicates that a slot boundary violation was detected in the slot assigned to channel B. It means that the channel is active at the start or end of the set slot.

Value	Description
0	No slot boundary violation detected on channel B
1	Slot boundary violation detected on channel B

**[bit6] SVOA: Channel A boundary violation bit (vSS!BViolationA)**

This bit indicates that slot boundary violation was detected in the slot assigned to channel A. It means that the channel is active at the start or end of the set slot.

Value	Description
0	No slot boundary violation detected on channel A
1	Slot boundary violation detected on channel A

**[bit5] CEOB: Channel B content error bit (vSS!ContentErrorB)**

This bit indicates that content error was detected in the slot assigned to channel B.

Value	Description
0	No content error detected on channel B
1	Content error detected on channel B

**[bit4] CEOA: Channel A content error bit (vSS!ContentErrorA)**

This bit indicates that a content error was detected in the slot assigned to channel A.

Value	Description
0	No content error detected on channel A
1	Content error detected on channel A

**[bit3] SEOB: Channel B syntax error bit (vSS!SyntaxErrorB)**

This bit indicates that a syntax error was detected in the slot assigned to channel B.

Value	Description
0	No syntax error detected on channel B
1	Syntax error detected on channel B

**[bit2] SEOA: Channel A syntax error bit (vSSISyntaxErrorA)**

This bit indicates that a syntax error was detected in the slot assigned to channel A.

Value	Description
0	No syntax error detected on channel A
1	Syntax error detected on channel A

**[bit1] VFRB: Channel B reception valid frame bit (vSSIValidFrameB)**

This bit is set to "1" when a valid frame is received on channel B.

Value	Description
0	No valid frame received on channel B
1	Valid frame received on channel B

**[bit0] VFRA: Channel A reception valid frame bit (vSSIValidFrameA)**

This bit is set to "1" when a valid frame is received on channel A.

Value	Description
0	No valid frame received on channel A
1	Valid frame received on channel A





#### 4.10.6. Output Buffer Command Mask Register (OBCM)

This register configures how the output buffer is updated by the message buffer selected by the OBCR register. If OBCR:REQ requests transfer of the message buffer, the RDSS and RHSS mask bits are copied to the internal registers. If the OBF host and OBF shadow are exchanged, the RDSH and RHSH mask bits and the RDSS and RHSS mask bits are also exchanged. "b) Data Transmission from Message RAM to Output Buffer" in Section "3.11.2. Host Access to Message RAM" describes data transfer between the output buffer (OBF) and message RAM in detail.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,W0							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved						RDSH	RHSH
ACCESS_TYPE	R0,W0						R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0000000						0	0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved							
ACCESS_TYPE	R0,W0							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved						RDSS	RHSS
ACCESS_TYPE	R0,W0						R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0000000						0	0

[bit31:18] Reserved: Reserved bits

[bit17] RDSH: Read data section host bit

Value	Description
0	Data section not read
1	Data section transferred from message RAM to output buffer

**Notes:**

- After transfer of the header section from the message RAM to the OBF shadow is completed, the message buffer status change flag, MBS, of the message buffer selected in the MBSC1/2/3/4 register is cleared to "0".
- After transfer of the data section from the message RAM to the OBF shadow is completed, the new data flag, ND, of the message buffer selected in the NDAT1/2/3/4 register is cleared to "0".

**[bit16] RSH: Read header section host bit**

Value	Description
0	Header section not read
1	Header section transferred from message RAM to output buffer

**[bit15:2] Reserved: Reserved bits**

**[bit1] RDSS: Read data section shadow bit**

Value	Description
0	Data section not read
1	Data section transferred from message RAM to output buffer

**[bit0] RHSS: Read header section shadow bit**

Value	Description
0	Header section not read
1	Header section transferred from message RAM to output buffer



#### 4.10.7. Output Buffer Command Request Register (OBCR)

If REQ is set to "1" while OBSYS is "0", OBSYS is set to "1". OBR[6:0] is copied to the internal register, and the OBCM:RDSS and OBCM:RHSS mask bits are copied to the internal register OBCM. Then, for the message buffer selected by OBR[6:0], transfer from that message buffer to the OBF shadow begins. After the transfer between the message buffer and OBF shadow is completed, the OBSYS bit is cleared to "0".

If VIEW is set to "1" while OBSYS is "0", the OBF host and OBF shadow are swapped with each other. Also, the OBCM:RDSH and OBCM:RHSH mask bits are replaced by the internal register OBCM to handle each output buffer transfer. OBRH[6:0] indicates the number of message buffers accessible to the CPU.

If both REQ and VIEW are set to "1" simultaneously while OBSYS is "0", OBSYS is set to "1".

Subsequently, the OBF host and OBF shadow are swapped with each other. Also, the OBCM:RDSH and OBCM:RHSH mask bits are swapped with the internal registers to handle each output buffer transfer. After that, OBR[6:0] is copied to the internal register. Then, transfer from the message RAM of the selected message buffer to the OBF shadow begins. While the transfer is in progress, the CPU can read the message buffer previously transferred from the OBF host. The OBSYS bit is cleared to "0" when the transfer between the message RAM and OBF shadow is completed.

If anything is written to this output buffer register while OBSYS is "1", the EIR:IOBA error flag is set to "1". In this case, the output buffer is not changed.

"b) Data Transmission from Message RAM to Output Buffer" in Section "3.11.2. Host Access to Message RAM" describes data transfer between the output buffer and message RAM in detail.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,W0							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved	OBRH						
ACCESS_TYPE	R0,W0	R,WX						
PROT_TYPE	-							
INITIAL_VALUE	0	0000000						

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	OBSYS	Reserved					REQ	VIEW
ACCESS_TYPE	R,WX	R0,W0					R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	00000					0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved	OBRs						
ACCESS_TYPE	R0,W0	R/W						
PROT_TYPE	-							
INITIAL_VALUE	0	0000000						

[bit31:23] Reserved: Reserved bits

**[bit22:16] OBRH[6:0]: Output buffer host transfer request bits**

These bits indicate the message buffer number currently accessible via RDHS[1 to 3], MBS, and RDDS[1 to 64]. Writing "1" to VIEW swaps the OBF shadow and OBF host with each other, thereby enabling access to the transferred message buffer. The range of valid values is 0x00 to 0x7F (0 to 127).

**[bit15] OBSYS: Output buffer shadow busy bit**

After the REQ bit is set to "1", this flag is set to "1". OBSYS is cleared to "0" when the transfer between the message RAM and OBF shadow is completed.

Value	Description
0	Transfer between message RAM and OBF shadow not in progress
1	Transfer between message RAM and OBF shadow in progress

**[bit14:10] Reserved: Reserved bits**

**[bit9] REQ: Request message RAM transfer bit**

This bit starts transfer of the message buffer specified by OBRS[6:0], from the message RAM to the OBF shadow. Writing is enabled only while OBSYS is "0".

Value	Description
0	Transfer from message RAM to OBF shadow not requested
1	Transfer from message RAM to OBF shadow requested

**[bit8] VIEW: Shadow buffer and host buffer swap bit**

This bit swaps the OBF shadow and OBF host. Writing is enabled only while OBSYS is "0".

Value	Description
0	OBF shadow and OBF host not swapped
1	OBF shadow and OBF host swapped

**[bit7] Reserved: Reserved bits**

**[bit6:0] OBRS[6:0]: Output buffer shadow transfer request bits**

These bits indicate the message buffer number to transfer from the message RAM to the OBF shadow. The range of valid values is 0x00 to 0x7F (0 to 127). If the first message buffer number of the reception FIFO is written in this register, the message buffer specified by GET Index (GIDX, see Section "3.10. FIFO Function") is transferred to the OBF shadow.





## CHAPTER 45: Up/Down Counter

This chapter describes the functions of a 8-bit/16-bit up/down counter.

---

1. Overview
2. Configuration
3. Operation
4. Registers
5. Precautions for Using This Device



## 1. Overview

The up/down counter consists of 3 event input pins, a 16-bit up/down counter, a 16-bit reload compare register, 16-bit counter compare registers, and their control circuits. An 8-bit counter or 16-bit counter can be selected depending on the settings.

### Functions of Up/Down Counter

#### a) Selecting 8-/16-bit Counter

Count can be performed in the range of "0x00" to "0xFF" or "0x0000" to "0xFFFF".

#### b) Selecting an Operation Mode by Selecting a Count Clock

##### – Timer mode

In timer mode, the counter counts down in synchronization with the internal clock generated by dividing the peripheral clock by 2 or 8.

- Clock obtained by dividing the peripheral clock by 2
- Clock obtained by dividing the peripheral clock by 8

##### – Up/Down count mode

In up/down count mode, the counter counts by detecting edges of input (AIN/BIN) signals from 2 external pins.

- Falling edge detection
- Rising edge detection
- Detection of both rising and falling edges
- Edge detection disabled (Count disabled)

##### – Phase difference count mode (Multiply-by-2 or 4)

In phase difference count mode, the counter counts phase differences between signals input from 2 external pins.

The phase difference count mode is appropriate for count by an encoder, such as a motor. In this mode, rotation angle, rotation speed, and other factors can easily be counted with high precision by inputting each of phases A, B, and Z output of the encoder.

The phase difference count mode includes multiply-by-2 mode and multiply-by-4 mode, each of which uses a different count method.

**Table 1-1 Operation Modes of Up/Down Counter**

Operation Mode	Count Timing	Count Direction
Timer mode	Internal clock (Peripheral clock)	Down
Up/Down Count mode	External input clock	Up/Down
Phase difference (multiply-by-2 or 4) count mode	External input signal phase	Up/Down

#### c) 2 Types of ZIN Pin Functions are Available (enabled in all operation modes)

- Counter clear function
- Gate function

#### **d) Compare and Reload Functions**

These functions can operate independently from each other or in combination. Using the combination of the functions enables up/down count with an arbitrary width.

- Compare function  
This function detects a match of (compares) the value set in the reload compare register and the counter value, clears the counter at the next up count timing, and then continues counting.
- Reload function  
This function loads the reload value to the counter when an underflow occurs, and then continues counting.
- Combination of compare and reload functions
- Compare/Reload disabled (Count disabled)

#### **e) Comparison Result Match Function and Buffer Transfer Function by Counter Compare Registers**

- Function to detect a comparison result match
- This function detects a match of (compares) the value set in a counter compare register and the counter value, and then continues counting as it is. The function can mask some of the bits of the set value and check for a comparison match of the remaining bits of the set value and the counter value.
- Buffer transfer function
- When an overflow occurs, this function can transfer the value set in a counter compare buffer transfer register to the counter compare register used for the function to detect a comparison result match. If you do not use this function, the value is transferred to the corresponding counter compare register at the same time as the write operation to the buffer transfer register.

#### **f) The Last Count Direction Can Be Identified Using the Count Direction Flag**

#### **g) Interrupt Request**

Interrupt generation can be controlled in the following cases independently from each other.

- When the count direction is inverted
- When the counter value matches (is compared to) the predefined value
- When an overflow or underflow (reload) occurs

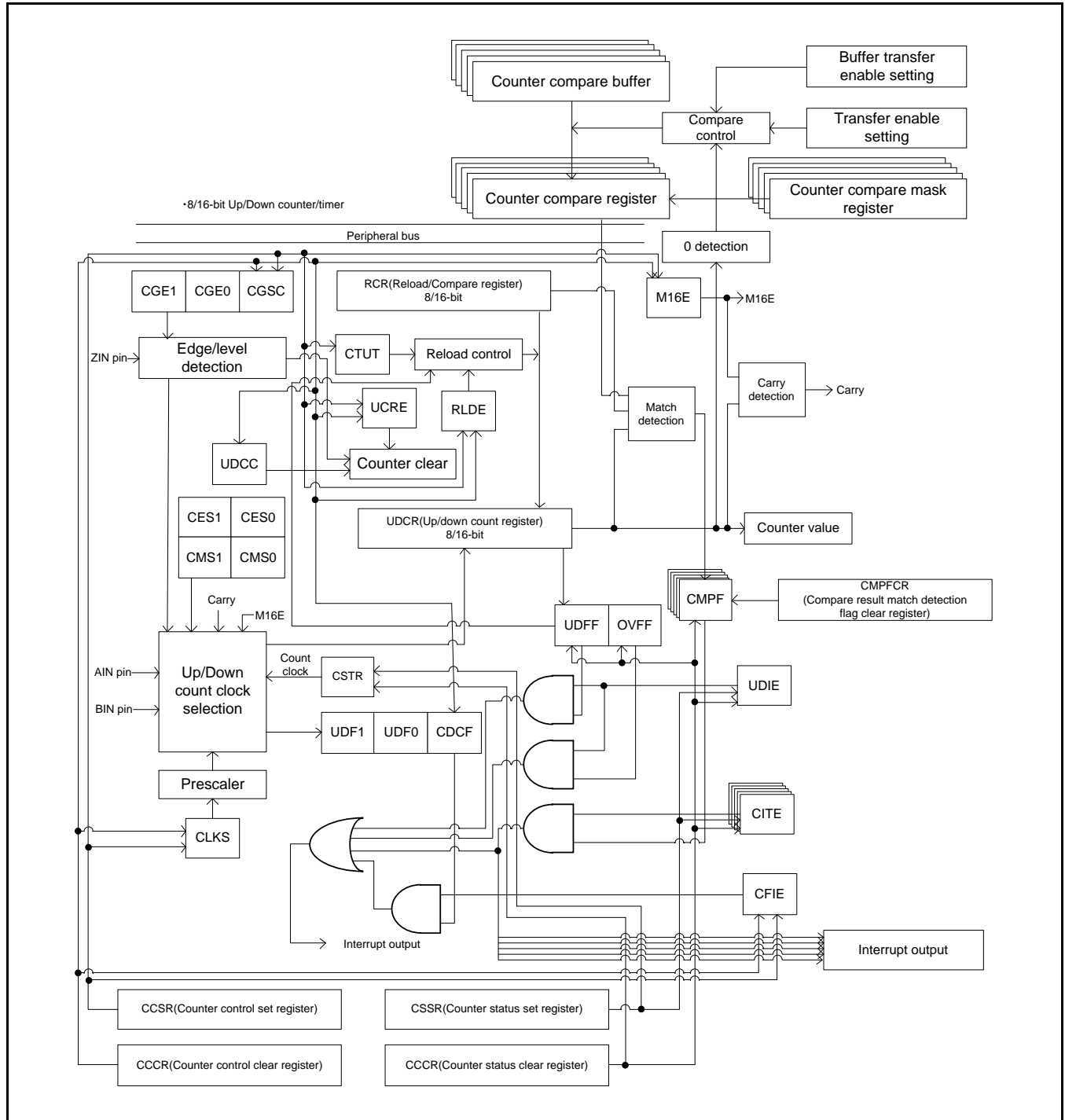


## 2. Configuration

This section shows the configuration of the up/down counter.

Figure 2-1 shows the configuration diagram of the up/down counter.

Figure 2-1 Configuration Diagram of Up/Down Counter



### 3. Operation

This section provides a summary of the operation of the up/down counter.

#### (1) Counter Operation Mode Selection

The CMS[1:0] bits in the CCR register select the operation mode of the up/down counter.

**Table 3-1 Counter Operation Modes**

CMS[1:0]		Description
0	0	Timer mode (down count) [initial value]
0	1	Up/Down count mode
1	0	Phase difference count mode, multiply-by-2
1	1	Phase difference count mode, multiply-by-4

– Timer mode [down count]

In timer mode, the counter counts down the output of the internal prescaler. Either 2 or 8 peripheral clock cycles can be selected for the internal prescaler using the CLKS bit in the CCRH register.

– Up/Down count mode

In up/down count mode, the counter counts up/down by counting the input of the external pins AIN and BIN. The input of the AIN pin controls up count, and the input of the BIN pin controls down count. Edges of AIN and BIN pin input are detected, and the CES[1:0] bits in the CCRH register can select the edge to be detected.

**Table 3-2 Count Clock Edges**

CES[1:0]		Description
0	0	Disable edge detection. [initial value]
0	1	Falling edge detection
1	0	Rising edge detection
1	1	Detection of both rising and falling edges

– Phase difference count mode (Multiply-by-2 or 4)

In phase difference count mode, the counter counts the phase difference between phase A and phase B of an encoder output signal. Thus, the input level of the BIN pin is detected when an edge of AIN pin input is detected to perform counting.

In multiply-by-2 or 4 mode, the phase difference between AIN pin input and BIN pin input is checked, and the counter counts up if AIN is earlier and counts down if BIN is earlier.

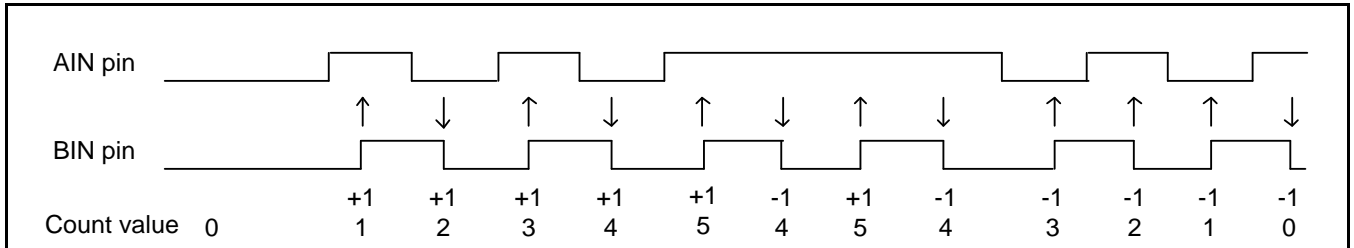
In multiply-by-2 mode, the counter counts by detecting the value of the AIN pin at the timing of both the rising and falling edges of the BIN pin. The method used in this case is as follows.



**Table 3-3 Count Method**

Edge of BIN Pin	Level of AIN Pin	Count
Rising ↑	"H" level	Up count
Rising ↑	"L" level	Down count
Falling ↓	"H" level	Down count
Falling ↓	"L" level	Up count

**Figure 3-1 Outline of Operation in Phase Difference Count Mode (Multiply-by-2)**

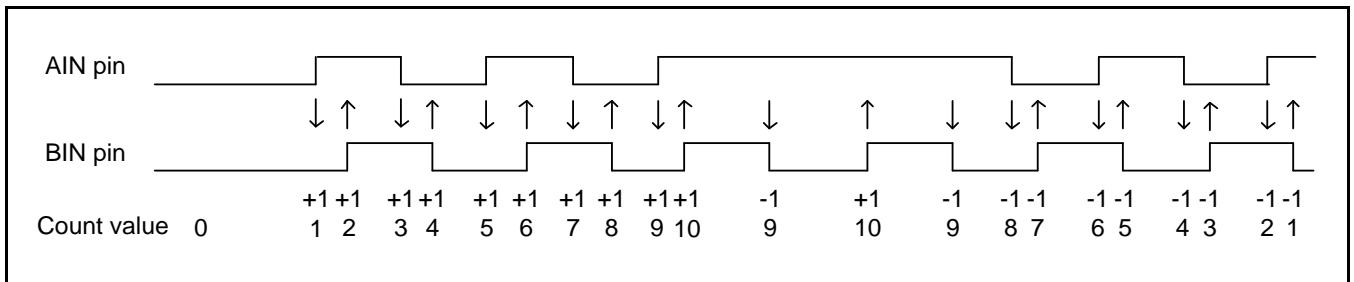


In multiply-by-4 mode, the value of the AIN pin is detected at the timing of both the rising and falling edges of the BIN pin. In addition, the value of the BIN pin is detected at the timing of both the rising and falling edges of the AIN pin to perform counting. The method used in this case is as follows.

**Table 3-4 Count Method**

Edge Detection Pin	Detected Edge	Level Check Pin	Input Level	Count Direction
BIN pin	Rising edge	AIN pin	"H" level	Count up
			"L" level	Count down
	Falling edge		"H" level	Count down
			"L" level	Count up
AIN pin	Rising edge	BIN pin	"H" level	Count down
			"L" level	Count up
	Falling edge		"H" level	Count up
			"L" level	Count down

**Figure 3-2 Outline of Operation in Phase Difference Count Mode (Multiply-by-4)**



When counting encoder output, phases A, B, and Z are input to the AIN, BIN, and ZIN pins, respectively. This makes it possible to count rotation angle and rotation speed with high precision and to detect rotation direction. Note that, when this count mode is selected, the selection of the edge to be detected that is made by the CES[1:0] bits in the CCRH register is disabled.

## (2) Reload/Compare Function

This counter has a reload function and a compare clear function. These 2 functions can operate in combination. The following provides an example of settings for the RLDE and UCRE bits in the CCRL register.

**Table 3-5 Count Clock Edges**

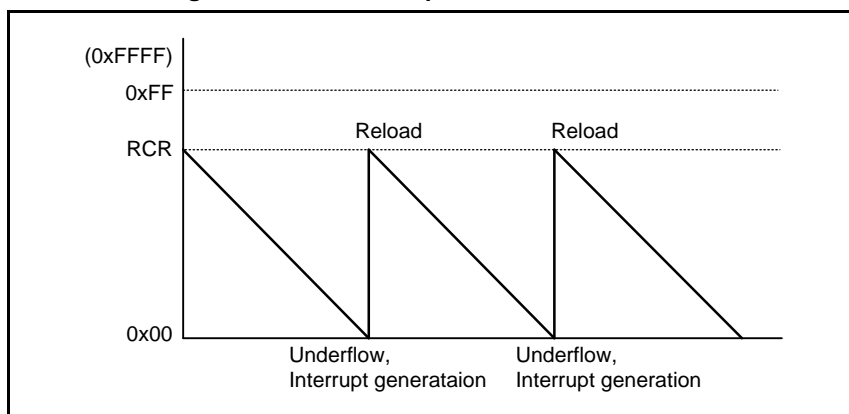
RLDE	UCRE	Description
0	0	Disable reload/compare clear [initial value].
0	1	Enable compare clear.
1	0	Enable reload.
1	1	Enable reload/compare clear.

### – Reload function

When the reload function is activated, the RCR value is transferred to the UDCR at the timing of the next down count clock after an underflow occurrence. In this case, the UDFF bit in the CSRL register is set, and an interrupt request is generated if the interrupt is enabled.

In a mode where down count is not performed, activation of this function changes from enabled to disabled.

**Figure 3-3 Outline of Operation of Reload Function**

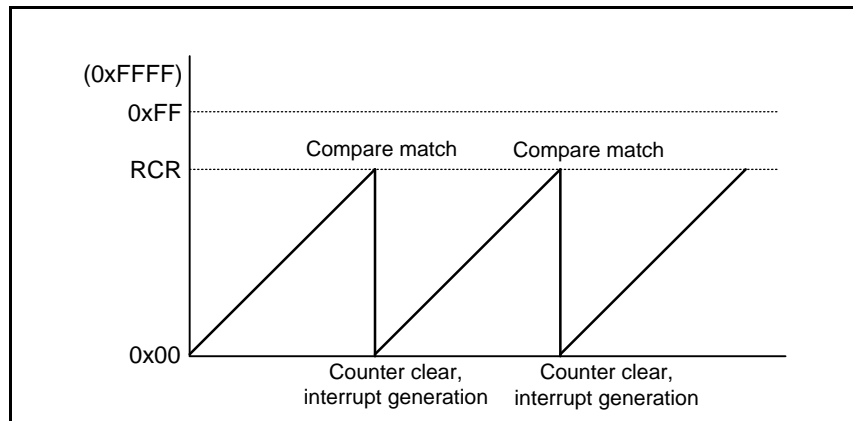


### – Compare clear function

The compare clear function is available in any mode other than timer mode. When the compare function is used, the CMPF bit is set if the RCR and UDCR values match, and an interrupt request is generated if the interrupt is enabled. In addition, the UDCR is cleared at the timing of the next up count clock (not at down count).

In a mode where up count is not performed, activation of this function changes from enabled to disabled.

Figure 3-4 Operation of Compare Clear Function



### (3) Count Direction Flag

The count direction flag (UDF[1:0]) indicates whether the last count is up count or down count when an up/down count is performed. The flag is rewritten at every count based on the count clock generated from the input of the AIN and BIN pins. If the current rotation direction needs to be found for controlling a motor, for example, check this flag.

Table 3-6 Count Direction Flag

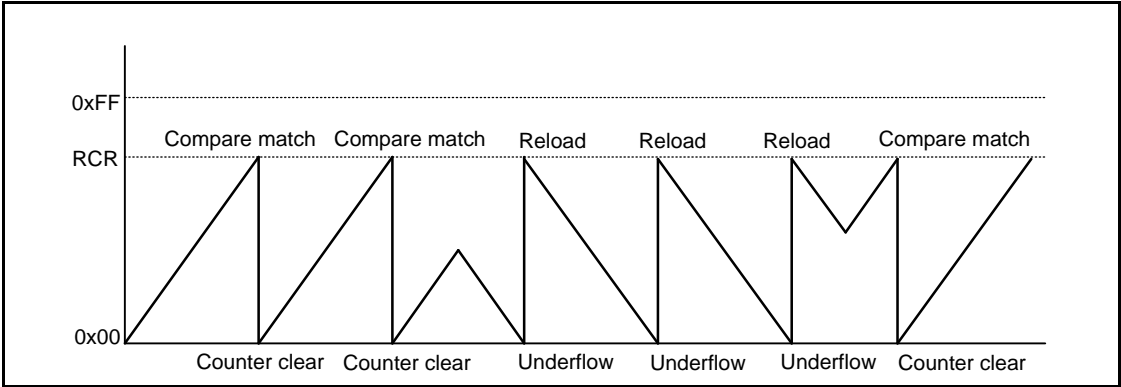
UDF[1:0]		Description
0	0	No input [initial value]
0	1	Down count
1	0	Up count
1	1	Simultaneous up/down occurrence (Count is not performed.)

### (4) Activating the Reload and Compare Functions Simultaneously

When both of the reload and compare functions are activated, up/down count can be performed with an arbitrary width.

The reload function transfers the RCR value to the UDCR at the time of underflow. The compare function clears the UDCR when the RCR and UDCR values match. Up/Down count is performed in the range of "0x0000" to RCR using these two functions.

Figure 3-5 Outline of Operation When the Reload and Compare Functions are Activated Simultaneously

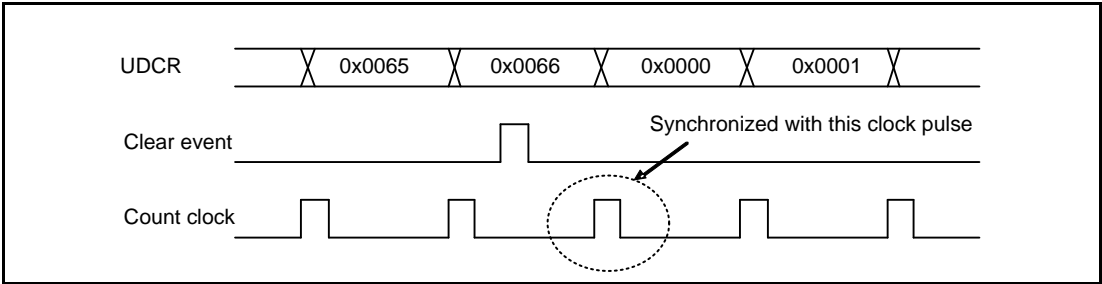


An interrupt can be generated to a CPU when a compare match or reload (underflow) occurs. The output interrupt can independently be enabled or disabled.

The timing of clearing the UDCR differs between when count is started and when it is stopped. Reload by software during count operation (writing "1" to the CTUT bit of CCR register) is disabled.

If a clear event occurs during count operation, the clear operation is performed in synchronization with every count clock.

Figure 3-6 Clear Event Occurrence Timing

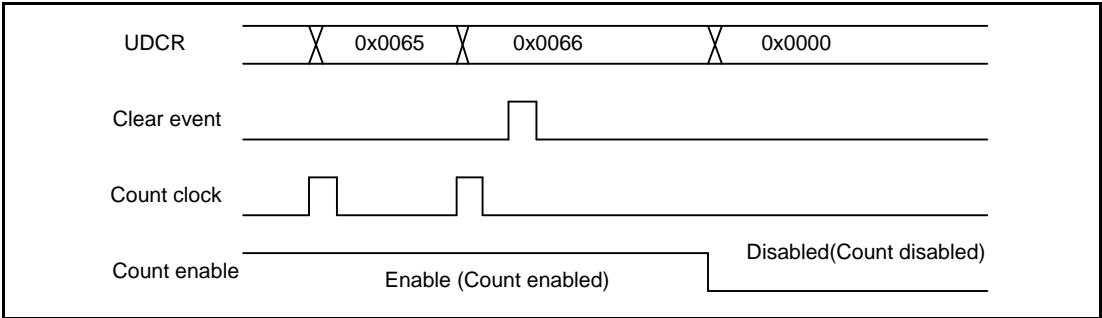


<Reference>

Reload due to an underflow during count operation is performed in synchronization with every count clock.

Suppose that a clear event occurs during count operation and the count is stopped while still waiting for count clock synchronization (a state where count input for synchronization is waited for). In this case, the clear operation is performed at the time of the stop.

Figure 3-7 Clear Event Occurrence Timing





If a reload or clear event occurs during count, the operation is performed at the time of the event occurrence.

In the case of compare clear, the clear operation is performed when the UDCR and RCR values match and up count is performed. If the count is switched to down count or stopped later, the clear operation is not performed even when the UDCR and RCR values match.

The timing of clear follows the above timing in all events except reset, and so does the timing of reload in all events.

If clear and reload events occur simultaneously, the clear event takes precedence.

#### (5) Data Writing to the UDCR

Data cannot be written directly to the UDCR. To write an arbitrary value to the UDCR, use the following procedure.

1. Write the data that you want to write to the UDCR to the RCR first. (Note that the existing data in the RCR will be lost.)
2. Write "1" to the CTUT bit in the CCR to transfer the data from the RCR to the UDCR.

Perform the above operations when count is stopped (the CSTR bit in the CSRL is "0").

#### **Note:**

- *If "1" is accidentally written to the CTUT bit of CCR register during count, the RCR value is transferred to the UDCR at the time of the writing.*

In addition to the above, the following counter clear methods are available.

- Clear using reset input
- Clear using edge input from the ZIN pin
- Clear by writing "0" to the UDCC bit in the CCR
- Compare clear

These types of writing can be performed regardless of whether count is started or stopped.

### (6) Count Clear/Gate Function

The count clear or gate function can be selected and used for the ZIN pin by using the CGSC bit in the CCR register.

When the count clear function is activated, the counter is cleared using the ZIN pin. The CGE[1:0] bits in the CCRL register can control which edge input of the ZIN pin to use for count.

When the gate function is activated, count is enabled/disabled using the ZIN pin. The CGE[1:0] bits in the CCR register can control which level input of the ZIN pin enables count. These functions are enabled in any mode.

**Table 3-7 Functions for the ZIN Pin**

CGSC	Description
0	Counter clear function [initial value]
1	Gate function

**Table 3-8 Count Clear/Gate Function**

CGE[1:0]		Counter clear function Selected (CGSC = "0")	Gate Function Selected (CGSC = "1")
0	0	Disable edge detection. [initial value]	Disable level detection. [initial value] (Count disable)
0	1	Falling edge	"L" level
1	0	Rising edge	"H" level
1	1	Setting prohibited	Setting prohibited

### (7) Count Direction Change Flag

The count direction change flag (CDCF) is set when the count direction is changed to or from up or down. At the same time as when this flag is set, an interrupt request can be generated to a CPU. Change of the count direction can be determined by checking this interrupt and the count direction flag.

However, note that in such cases as when the direction change period is short and changes occur continuously, the direction indicated by the flag after direction change may be the same, returning to the original direction.

**Table 3-9 Count Direction Change Flag**

CDCF	Description
0	No direction change has occurred. [initial value]
1	Direction change has occurred once or more.

### (8) Compare Detection Flag

The compare detection flag (CMPF) is set when the UDCR and RCR values match during count operation. Besides a count up/down match, it is set when a match due to a reload event occurrence occurs or when a match already exists when the count is started.

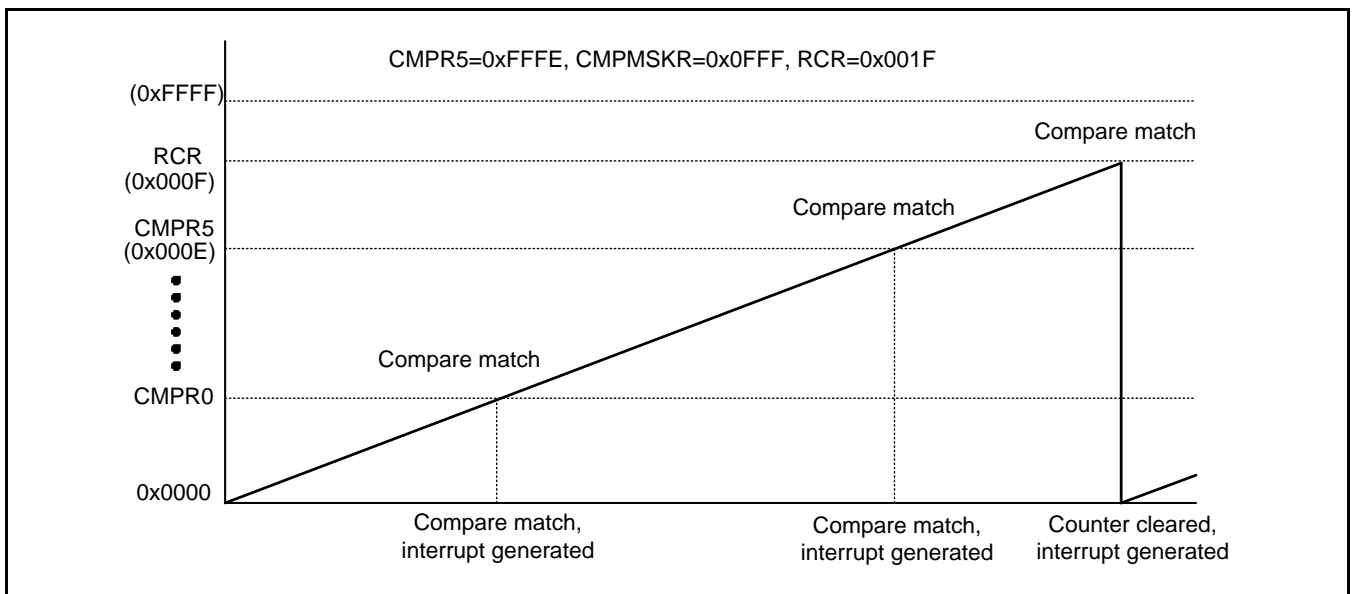




### (9) Comparison Result Match Detection Flag

The comparison result match detection flag (CMPF0 to 5) is set when the UDCR value and the value of CMPR0 to 5 match during count operation. (e.g., CMPF3 is set when CMPR3=UDCR). If CMPMSKR0 to 5 mask some of the corresponding bits CMPR0 to 5, and each of the remaining (non-masked) bits of CMPF0 to 5 matches the UDCR, CMPF0 to 5 are set. (e.g., CMPF5 is set when the count is performed up to UDCR="0x0FFF" at the time of CMPR5="0xFFFF" and CMPMSKR5="0x0FFF"). CMPF0 to 5 are not set when count operation is stopped.

**Figure 3-8 Operation of Comparison Result Match Detection Using Counter Compare Registers**



**Figure 3-9 Timing of Interrupt Using Counter Compare Registers**

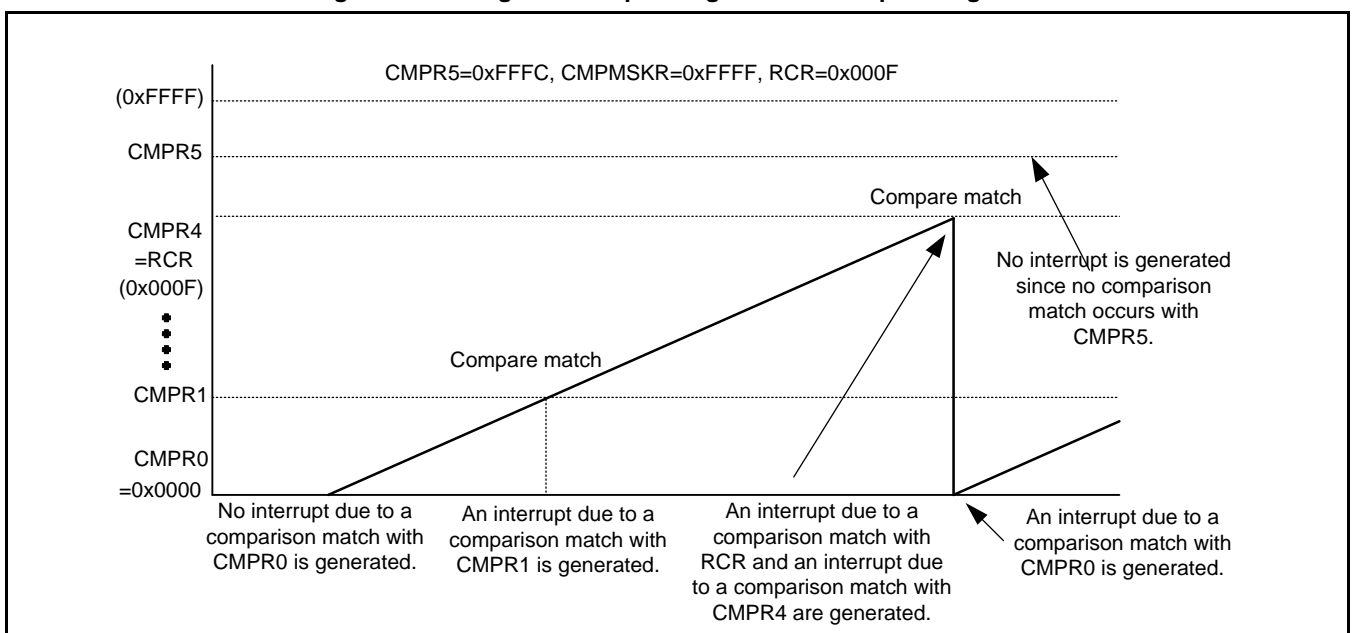
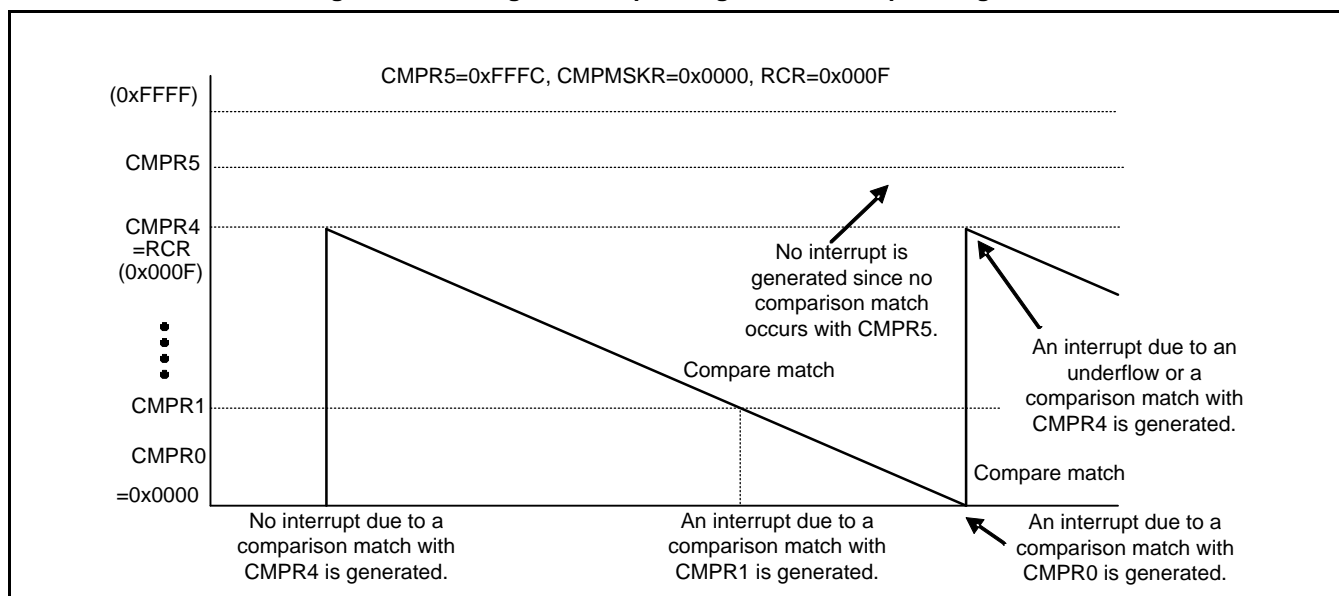


Figure 3-10 Timing of Interrupt Using Counter Compare Registers



### (10) Updating Counter Compare Registers

Counter compare registers can be updated via counter compare buffer transfer registers. The following table lists the update conditions.

Table 3-10 Conditions for Updating Counter Compare Registers

BTC[1:0]		Description
0	0	The values of CMPBR0 to 5 are not transferred to CMPR0 to 5.
0	1	The values of CMPBR0 to 5 are transferred to CMPR0 to 5 by buffer transfer simultaneously with an overflow or a count clear due to an RCR match.
1	0	When data is written to CMPBR0 to 5, it is also written to CMPR0 to 5.
1	1	Same as the case of BTC[1:0]=0b10

**Note:**

- If BTC[1:0] = 0b00 or 0b01 changes to BTC[1:0] = 0b10 or 0b11, the values of CMPBR0 to 5 are written to CMPR0 to 5 at clock rising after the change.



Figure 3-11 Update Timing in the Case of BTC[1:0]=0b00

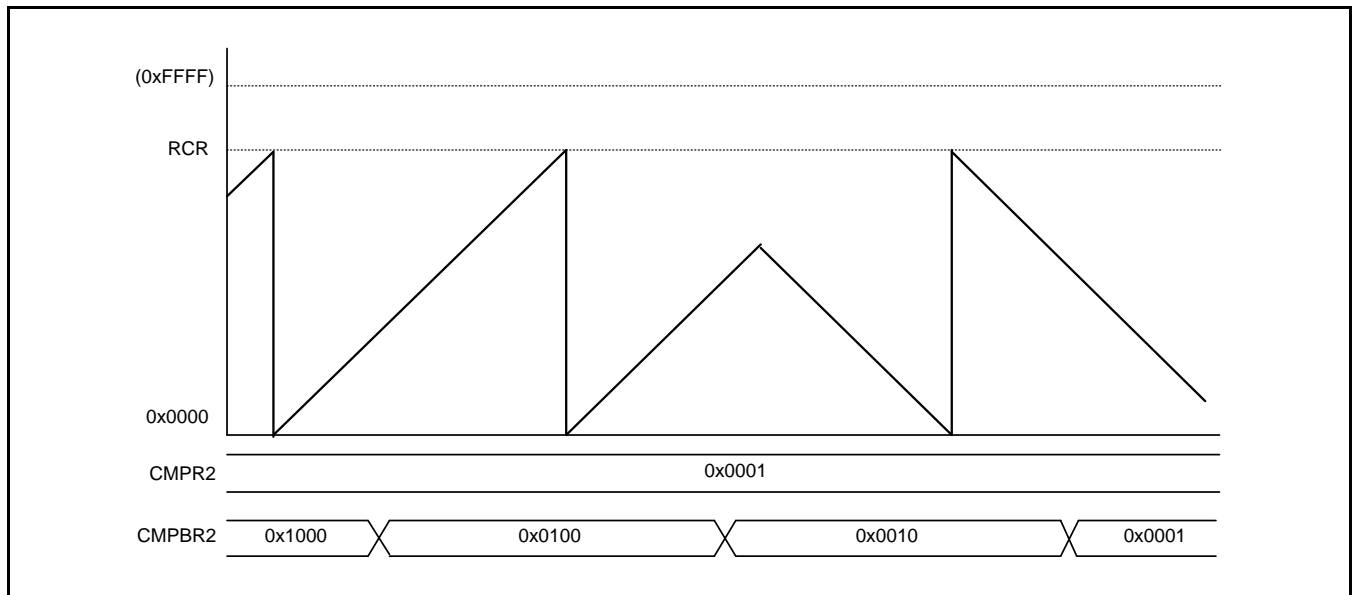


Figure 3-12 Update Timing in the Case of BTC[1:0]=0b01

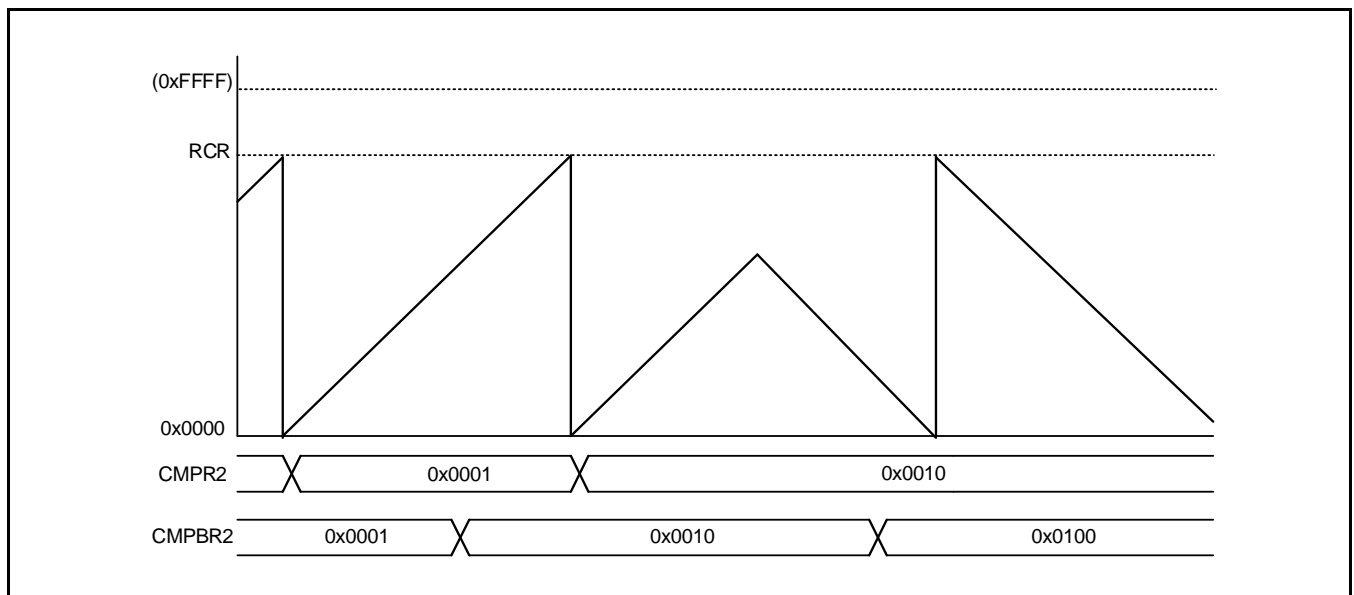


Figure 3-13 Update Timing in the Case of BTC[1:0]= 0b10 or 0b11

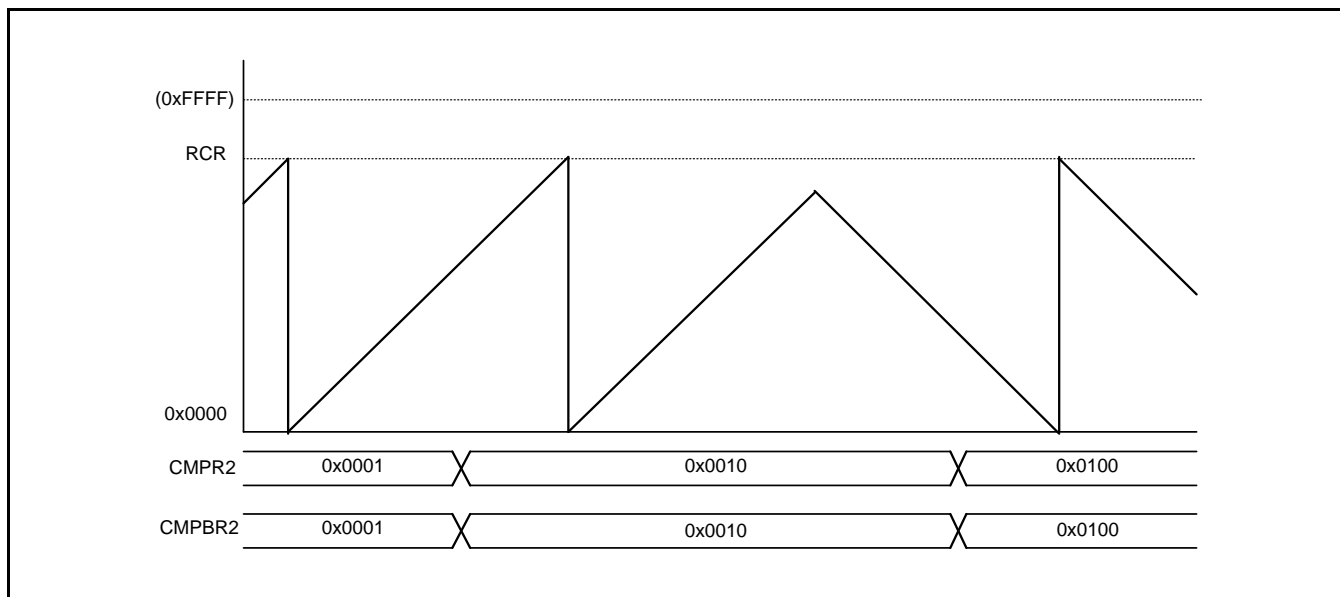
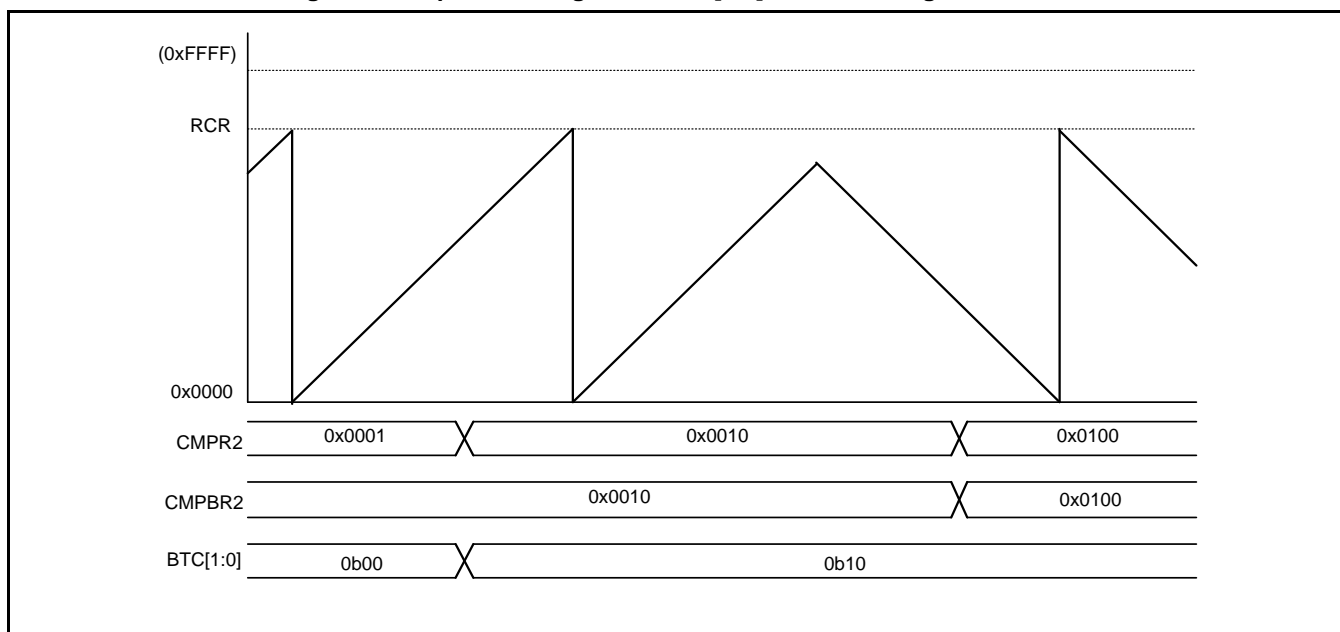


Figure 3-14 Update Timing When BTC[1:0]="0b00" Changes to "0b10"



### (11) 8-bit/16-bit Counter Operation

The up/down counter can be used as an 8-bit or 16-bit up/down counter. Write "0" to the M16E bit in the CCR register to use the counter in 8-bit mode or "1" to use it in 16-bit mode.

- 8-bit mode (M16E="0")  
Only the up/down count register, lower (UDCRL) is used. Write a reload/compare value to the reload compare register, lower (RCRL) in bytes.
- 16-bit mode (M16E="1")  
Both the upper and lower bytes of the up/down count register (UDCR) are used. Write a reload/compare value to the reload compare register (RCR) in half-words.



## (12) Interrupt Generation Timing

Table 3-11 Conditions for Updating Counter Compare Registers

Interrupt Flag	Flag Set Interrupt	Reload	Clear
CDCF (Count Direction Change Flag)	An interrupt is generated simultaneously at the flag set. Flag is set at the count clock when count direction change.		
CMPF (Compare Detection Flag)	An interrupt is generated simultaneously at the flag set. Flag is set when RCR and UDCR matches while up count, down count or reload count is activated.		The UDCR is cleared at the timing of the next up count after the match of RCR and UDCR (not at down count).
OVFF (Overflow Detection Flag)	An interrupt is generated simultaneously at the flag set. Flag is set at the up count after the count "0xFFFF".		The UDCR is cleared at the timing of the next count after the count "0xFFFF".
UDFF (Underflow Detection Flag)	An interrupt is generated simultaneously at the flag set. Flag is set at the down count after the count "0x0000".	The RCR value is transferred to the UDCR at the timing of the next count after the count "0x0000".	
CMPFn (n = 0 to 5) (Comparison Result Match Detection Flag)	An interrupt is generated simultaneously at the flag set. Flag is set when CMPRn (n = 0 to 5) matches the UDCR.		

(The corresponding count values in 8-bit mode are "0x00" and "0xFF", respectively.)

The RCR is used for both a reload value and a compare value. Therefore, if a reload is performed, the compare detection flag is always set.

If up count is performed after a compare match occurs when down count is performed with the clear function enabled, a clear occurs.

## 4. Registers

This section shows a list of up/down counter registers.

A prefix (UDC16Bxx) is added to each of the up/down counter registers. xx is the channel number .

**Table 4-1 List of Up/Down Counter Registers**

Abbreviated Register Name	Register Name	See
UDC16Bxx_UDCRH, UDC16Bxx_UDCRL	Up/Down Count Register	4.1
UDC16Bxx_RCRH, UDC16Bxx_RCRL	Reload Compare Register	4.2
UDC16Bxx_CSRL	Counter Status Register	4.3
UDC16Bxx_CCRH, UDC16Bxx_CCRL	Counter Control Register	4.4
UDC16Bxx_CMPRHn, UDC16Bxx_CMPRLn (n = 0 to 5)	Counter Compare Register	4.5
UDC16Bxx_CMPBRHn, UDC16Bxx_CMPBRLn (n = 0 to 5)	Counter Compare Buffer Transfer Register	4.6
UDC16Bxx_CMPMSKRHn, UDC16Bxx_CMPMSKRLn (n = 0 to 5)	Counter Compare Mask Register	4.7
UDC16Bxx_CMPFR	Comparison Result Match Detection Flag Register	4.8
UDC16Bxx_CITER	Comparison Result Match Interrupt Enable Register	4.9
UDC16Bxx_CBTR	Buffer Transfer Setting Register	4.10
UDC16Bxx_CCSRH, UDC16Bxx_CCSRL	Counter Control Set Register	4.11
UDC16Bxx_CCCRH, UDC16Bxx_CCCRL	Counter Control Clear Register	4.12
UDC16Bxx_CSSRL	Counter Status Set Register	4.13
UDC16Bxx_CSCRL	Counter Status Clear Register	4.14
UDC16Bxx_CMPFCR	Comparison Result Match Detection Flag Clear Register	4.15

xx: Channel number (xx = 00 to 03)



## 4.1. Up/Down Count Register (UDCR)

The up/down count register (UDCR) is an 8-/16-bit count register. Up/Down count is performed using input from an internal circuit, an internal prescaler, or input from the AIN and BIN pins. In 16-bit counter mode, the register operates as a 16-bit count register. In 8-bit counter mode, only the value of the UDCRL is valid. In 16-bit counter mode, the values of the UDCRH and the UDCRL are valid.

### (1) Up/Down Count Register, Upper (UDCRH)

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	D[15:8]							
ACCESS_TYPE	R,WX							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

### (2) Up/Down Count Register, Lower (UDCRL)

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	D[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

#### [bit15:0] D[15:0]: Up/Down count value bits

The direct write operation cannot be performed on D[15:0]. To write to this register, use the RCR. Write the value that you want to write to this register to the RCR first. Then, write "1" to the CTUT bit in the CCRL register to transfer the value from the RCR to this register (reload by software).

#### Notes:

- In 16-bit counter mode (CCR<sub>H</sub>: M16E = "1"), read the UDCR in half-words.
- In 8-bit counter mode (CCR<sub>H</sub>: M16E = "0"), only the value of the UDCRL is valid.

## 4.2. Reload Compare Register (RCR)

The reload compare register (RCR) is an 8-/16-bit reload compare register. This register sets reload and compare values. The reload value is the same as the compare value. By activating the reload function and the compare function (with clear function), up/down count can be performed in the range of "0x00" to the value of this register (16-bit counter mode: "0x0000" to the value of this register). In 8-bit counter mode, only the value of the RCRL is valid. In 16-bit counter mode, the values of the RCRH and the RCRL are valid.

### (1) Reload Compare Register, Upper (RCRH)

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	D[15:8]							
ACCESS_TYPE	RX,W							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

### (2) Reload Compare Register, Lower (RCRL)

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	D[7:0]							
ACCESS_TYPE	RX,W							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

#### [bit15:0] D[15:0]: Reload compare value bits

D[15:0] are writable but cannot be read. The value of this register can be transferred to the UDCR by writing "1" to the CTUT bit in the CCR while count is stopped. (Reload by software)

#### Notes:

- In 16-bit counter mode (CCR: M16E = "1"), write data to the RCR in half-words.
- In 8-bit counter mode (CCR: M16E = "0"), write data to the RCRL in bytes.





### 4.3. Counter Status Register (CSRL)

The counter status register (CSRL) can be used to check the state of an up/down counter or control interrupts. For details on writing to this register, see Section "5. Precautions for Using This Device".

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	CSTR	CITE	UDIE	CMPF	OVFF	UDFF	UDF	
ACCESS_TYPE	R/W	R/W	R/W	R,W	R,W	R,W	R,WX	
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	00	

#### [bit7] CSTR: Count start bit

- This bit controls the start/stop of the count operation of the UDCR.
- This bit is cleared to "0" by setting the CSTRC bit in the CSCRL register to "1".
- This bit is set to "1" by setting the CSTRS bit in the CSSRL register to "1".

Value	Description
0	Stop count operation.
1	Start count operation.

#### [bit6] CITE: Compare detection interrupt enable bit

- This bit enables/disables interrupt output to a CPU when the CMPF is set (a comparison match occurs).
- This bit is cleared to "0" by setting the CITEC bit in the CSCRL register to "1".
- This bit is set to "1" by setting the CITES bit in the CSSRL register to "1".

Value	Description
0	Disable compare detection interrupts.
1	Enable compare detection interrupts.

#### [bit5] UDIE: Overflow/Underflow interrupt enable bit

- This bit enables/disables interrupt output to a CPU when the OVFF/UDFF is set (an overflow/underflow occurs).
- This bit is cleared to "0" by setting the UDIEC bit in the CSCRL register to "1".
- This bit is set to "1" by setting the UDIES bit in the CSSRL register to "1".

Value	Description
0	Disable overflow/underflow interrupts.
1	Enable overflow/underflow interrupts.

#### [bit4] CMPF: Compare detection interrupt flag bit

- This flag indicates that the UDCR and RCR values are compared and they match.
- Writing "0" clears it.
- Writing "1" is ignored and the value of this bit does not change.
- This bit is cleared to "0" by setting the CMPFC bit in the CSCRL register to "1".

Value	Description
0	The comparison result is not a match.
1	The comparison result is a match.

**[bit3] OVFF: Overflow detection interrupt flag bit**

- This flag indicates that an overflow occurred.
- Writing "0" clears it.
- Writing "1" is ignored and the value of this bit does not change.
- This bit is cleared to "0" by setting the OVFFC bit in the CSCRL register to "1".

Value	Description
0	No overflow
1	Overflow

An overflow occurs when count up is performed when the counter value is "0xFFFF" (or "0xFF" in 8-bit mode).

**[bit2] UDFC: Underflow detection interrupt flag bit**

- This flag indicates that an underflow occurred.
- Writing "0" clears it.
- Writing "1" is ignored and the value of this bit does not change.
- This bit is cleared to "0" by setting the UDFC bit in the CSCRL register to "1".

Value	Description
0	No underflow
1	Underflow

An underflow occurs when count down is performed when the counter value is "0x0000" (or "0x00" in 8-bit mode).

**[bit1:0] UDF[1:0]: Up/Down flag bits**

- These bits indicate the last count operation (up/down).
- Write operation does not have any effect.

Value	Description
00	No input
01	Down count
10	Up count
11	Up and down occurred simultaneously.



## 4.4. Counter Control Register (CCR)

The counter control register (CCR) controls the operation mode of an up/down counter. For details on writing to this register, see Section "5. Precautions for Using This Device".

### (1) Count control register, upper (CCR<sub>H</sub>)

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	M16E	CDCF	CFIE	CLKS	CMS		CES	
ACCESS_TYPE	R/W	R,W	R/W	R/W	R/W		R/W	
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	00		00	

#### [bit15] M16E: 16-bit mode enable setting bit

- This bit selects the 8-bit or 16-bit operation mode.
- This bit is cleared to "0" by setting the M16EC bit in the CCCR<sub>H</sub> register to "1".
- This bit is set to "1" by setting the M16ES bit in the CCSR<sub>H</sub> register to "1".

Value	Description
0	8-bit operation mode
1	16-bit operation mode

#### [bit14] CDCF: Count direction change detection flag bit

- This flag is set when the count direction changes. This bit is set to "1" when the count direction changes from up to down or down to up while a count is started.
- Writing "0" clears it.
- Writing "1" is ignored and the value of this bit does not change.
- This bit is cleared to "0" by setting the CDCFC bit in the CCCR<sub>H</sub> register to "1".

Value	Description
0	No direction change has occurred.
1	Direction change has occurred once or more.

The count direction immediately after a reset is the down count direction. Therefore, the CDCF is set to "1" when up count is performed immediately after a reset.

#### [bit13] CFIE: Count direction change interrupt enable bit

- This bit controls interrupt output to a CPU when the CDCF is set. An interrupt is generated if the count direction changes even once while a count is started.
- This bit is cleared to "0" by setting the CFIEC bit in the CCCR<sub>H</sub> register to "1".
- This bit is set to "1" by setting the CFIES bit in the CCSR<sub>H</sub> register to "1".

Value	Description
0	Disable direction change interrupts.
1	Enable direction change interrupts.

**[bit12] CLKS: Internal prescaler selection bit**

- This bit selects the frequency of the internal prescaler when the timer mode is selected.
- This bit is valid only in timer mode. In this case, only down count is performed.
- This bit is cleared to "0" by setting the CLKSC bit in the CCCRH register to "1".
- This bit is set to "1" by setting the CLKSS bit in the CCSRH register to "1".

Value	Description
0	2 peripheral clock cycles
1	8 peripheral clock cycles

**[bit11:10] CMS[1:0]: Count mode selection bits**

These bits select a count mode.

Value	Description
00	Timer mode (down count)
01	Up/Down count mode
10	Phase difference count mode, multiply-by-2
11	Phase difference count mode, multiply-by-4

**[bit9:8] CES[1:0]: Count clock edge selection bits**

- These bits select the edge to be detected for internal circuit input and external pins AIN and BIN in up/down count mode.
- This setting is disabled in a mode other than up/down count mode.

Value	Description
00	Disable edge detection.
01	Falling edge detection
10	Rising edge detection
11	Detection of both rising and falling edges



## (2) Count Control Register, Lower (CCRL)

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved	CTUT	UCRE	RLDE	UDCC	CGSC	CGE	
ACCESS_TYPE	R0,W0	R0,W	R/W	R/W	R1,W	R/W	R/W	
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	1	0	00	

### [bit7] Reserved: Reserved bit

### [bit6] CTUT: Counter write bit

- This bit executes data transfer from the RCR to the UDCR.
- When "1" is written to CTUT, data is transferred from the RCR to the UDCR.
- Writing "0" is invalid and the read value is always "0".
- Do not write "1" to this bit during count operation (when the CSTR bit in the CSRL is "1").
- This bit is set to "1" by setting the CTUTS bit in the CCSRL register to "1".

### [bit5] UCRE: Compare clear enable bit

- This bit controls whether the UDCR is cleared by compare. If the UDCR matches the value of the reload compare register (RCR) when counter clear is enabled, the UDCR is cleared at the time of the next up count.
- This bit has no effect on UDCR clear functions (e.g., clear using the ZIN pin) other than compare.
- This bit is cleared to "0" by setting the UCRC bit in the CCCRL register to "1".
- This bit is set to "1" by setting the UCRES bit in the CCSRL register to "1".

Value	Description
0	Disable counter clear.
1	Enable counter clear.

### [bit4] RLDE: Reload enable bit

- This bit controls the operation of the reload function. If an underflow occurs in the UDCR when the reload function is enabled, the RCR value is transferred to the UDCR.
- This bit is cleared to "0" by setting the RLDEC bit in the CCCRL register to "1".
- This bit is set to "1" by setting the RLDES bit in the CCSRL register to "1".

Value	Description
0	Disable the reload function.
1	Enable the reload function.

### [bit3] UDCC: Counter clear bit

- This bit clears the UDCR. The UDCR is cleared to "0x0000" when "0" is written to this bit. Writing "1" is invalid and the read value is always "1".
- This bit is cleared to "0" by setting the UDCCC bit in the CCCRL register to "1".

**[bit2] CGSC: Counter clear/gate function selection bit**

- This bit selects a function of the external pin ZIN.
  - Counter clear function  
The UDCR is cleared to "0x0000" when a valid edge is input from the ZIN pin.
  - Gate function  
The counter operates while a valid level is being input from the ZIN pin.
- This bit is cleared to "0" by setting the CGSCC bit in the CCCRL register to "1".
- This bit is set to "1" by setting the CGSCS bit in the CCSRL register to "1".

Value	Description
0	Counter clear function
1	Gate function

**[bit1:0] CGE[1:0]: CGSC operation (edge/level) selection bits**

These bits select the edge/level to be detected from the external pin ZIN.

Value	Description	
	When Counter Clear Function Selected (CGSC="0")	When Gate Function Selected (CGSC="1")
00	Disable edge detection.	Disable level detection.(Count disable)
01	Falling edge	"L" level
10	Rising edge	"H" level
11	Setting prohibited	Setting prohibited



## 4.5. Counter Compare Register (CMPR0 to 5)

The counter compare register (CMPR0 to 5) is used to compare a compare value and the counter value of the up/down counter. The compare value is a value that is compared with the value counted at the time of count up.

### (1) Counter Compare Register, Upper (CMPRH0 to 5)

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	D[15:8]							
ACCESS_TYPE	R,WX							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

### (2) Counter Compare Register, Lower (CMPRL0 to 5)

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	D[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

[bit15:0] D[15:0]

**Notes:**

- In 16-bit counter mode (CCR<sub>H</sub>: M16E = "1"), read CMPR0 to 5 in half-words.
- In 8-bit counter mode (CCR<sub>H</sub>: M16E = "0"), only the value of CMPRL 0 to 5 is valid. In this case, values transferred from CMPBRH0 to 5 are read from CMPRH0 to 5.

## 4.6. Counter Compare Buffer Transfer Register (CMPBR0 to 5)

The counter compare buffer transfer register (CMPBR0 to 5) is a 16-bit register for a counter compare register (CMPR0 to CMPR5). In 8-bit counter mode (CCRH:M16E = "0"), only CMPBRL0 to 5 are valid.

### (1) Counter Compare Buffer Transfer Register, Upper (CMPBRH0 to 5)

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	D[15:8]							
ACCESS_TYPE	RX,W							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

### (2) Counter Compare Buffer Transfer Register, Lower (CMPBRL0 to 5)

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	D[7:0]							
ACCESS_TYPE	RX,W							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

[bit15:0] D[15:0]

**Notes:**

- In 16-bit counter mode (CCRH: M16E = "1"), write data to CMPBR0 to 5 in half-words.
- In 8-bit counter mode (CCRH: M16E = "0"), write data to CMPBRL0 to 5 in bytes.





## 4.7. Counter Compare Mask Register (CMPMSKR0 to 5)

This register specifies whether the compare value of a counter compare register (CMPR0 to 5) is excluded from comparison in bit unit. CMPR0 to 5 correspond to CMPMSKR0 to 5, respectively. In 8-bit counter mode (CCR0:M16E = "0"), only CMPMSKRL is valid.

### (1) Counter Compare Mask Register, Upper (CMPMSKRH0 to 5)

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	DM15	DM14	DM13	DM12	DM11	DM10	DM9	DM8
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	1	1	1	1	1	1	1	1

### (2) Counter Compare Mask Register, Lower (CMPMSKRL0 to 5)

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	DM7	DM6	DM5	DM4	DM3	DM2	DM1	DM0
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	1	1	1	1	1	1	1	1

#### [bit15:0] DM15 to DM0: Compare comparison mask bits.

These bits specify whether the compare value of the relevant compare register (CMPR0 to 5) is excluded from comparison in bit unit.

Value	Description
0	Do not compare bits. (Handle them as a match.)
1	Compare bits.

(Example of a setting)

To perform compare comparison of the lower 12 bits in CMPR0, write "0x0FFF" to CMPMSKR0.

#### Notes:

- In 16-bit counter mode (CCR0: M16E="1"), write data to CMPMSKR0 to 5 in half-words.
- In 8-bit counter mode (CCR0: M16E="0"), write data to CMPMSKRL0 to 5 in bytes.

## 4.8. Comparison Result Match Detection Flag Register (CMPFR)

This register indicates that the counter value matches the value set in a counter compare register (CMPR0 to 5). If a CITER: CITE bit (0 to 5) is set to "1" when the relevant bit is "1", a request for a comparison result match interrupt is generated. For details on writing to this register, see Section "5. Precautions for Using This Device".

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved		CMPF5	CMPF4	CMPF3	CMPF2	CMPF1	CMPF0
ACCESS_TYPE	R0,W0		R,W	R,W	R,W	R,W	R,W	R,W
PROT_TYPE	-							
INITIAL_VALUE	00		0	0	0	0	0	0

[bit7:6] Reserved: Reserved bits

[bit5:0] CMPF5 to CMPF0: Comparison result match detection flag bits

- These bits indicate whether the counter value matches the value set in a counter compare register (CMPR0 to 5).
- Writing "0" clears it.
- Writing "1" is ignored and the value of this bit does not change.
- This bit is cleared to "0" by setting the CMPFnC (n = 0 to 5) bit in the CMPFCR register to "1".

Value	Description
0	The counter value does not match the value set in the counter compare register (CMPR0 to 5).
1	The counter value matches the value set in the counter compare register (CMPR0 to 5).

(Operation example)

When CMPMSKR0 is "0xFFFF", CMPF0 is set when the counter and CMPR0 match.

### Notes:

- If the UDCR is equal to CMPRn when CMPFn (n = 0 to 5) is cleared, CMPFn is not set.
- If CMPBRn (n = 0 to 5) is set to 0 at the time of buffer transfer (CBTR:BTC[1:0]=0b10) and data is transferred from CMPBRn (n = 0 to 5) to CMPRn (n = 0 to 5), the UDCR and CMPRn (n = 0 to 5) are 0, which is a match. In this case, however, CMPFn (n = 0 to 5) is not set.



## 4.9. Comparison Result Match Interrupt Enable Flag Register (CITER)

This register specifies whether a comparison match interrupt request is generated when the counter value matches the values set in counter compare registers 0 to 5 (CMPR0 to 5) (CMPF0 to 5 = "1").

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved		CITE5	CITE4	CITE3	CITE2	CITE1	CITE0
ACCESS_TYPE	R0,W0		R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	00		0	0	0	0	0	0

**[bit7:6] Reserved: Reserved bits**

### **[bit5:0] CITE5 to CITE0: Comparison result match interrupt enable bits**

These bits enable/disable generation of a comparison match interrupt request.

Value	Description
0	Disable generation of a comparison match interrupt request.
1	Enable generation of a comparison match interrupt request.

## 4.10. Buffer Transfer Setting Register (CBTR)

This register specifies simultaneous transfer of compare values from counter compare buffer transfer registers (CMPBR0 to 5) to counter compare registers (CMPR0 to 5).

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved						BTC	
ACCESS_TYPE	R0,W0						R/W	
PROT_TYPE	-							
INITIAL_VALUE	000000						00	

**[bit7:2] Reserved: Reserved bits**

**[bit1:0] BTC[1:0]: Buffer transfer control bits**

- These bits control transfer from counter compare buffer transfer registers (CMPBR0 to 5) to counter compare registers (CMPR0 to 5).
- The following table shows the relationship of BTC[1:0]. If BTC[1:0] is changed, the buffer transfer timing is immediately changed.

Value	Description
00	The values of CMPBR0 to 5 are not transferred to CMPR0 to 5.
01	The values of CMPBR0 to 5 are transferred to CMPR0 to 5 simultaneously with an overflow or a count clear due to an RCR match.
10	When data is written to CMPBR0 to 5, it is also written to CMPR0 to 5.
11	Same as the case of BTC[1:0] = 0b10

**Note:**

- If BTC[1:0] = 0b00 or 0b01 changes to BTC[1:0] = 0b10 or 0b11, the values of CMPBR0 to 5 are written to CMPR0 to 5 at clock rising after the change.



## 4.11. Counter Control Set Register (CCSR)

This register is used to set bits in the counter control register (CCR).

### (1) Counter Control Set Register, Upper (CCSRH)

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	M16ES	Reserved	CFIES	CLKSS	Reserved			
ACCESS_TYPE	R0,W	R0,W0	R0,W	R0,W	R0,W0			
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0000			

#### [bit15] M16ES: 16-bit mode enable set bit

- Writing "1" to this bit sets the M16E bit in the CCR register.
- "0" is always read during reading.

Value	Description
0	No effect
1	Set the M16E bit in the CCR register.

#### [bit14] Reserved: Reserved bit

#### [bit13] CFIES: Count direction change interrupt enable set bit

- Writing "1" to this bit sets the CFIE bit in the CCR register.
- "0" is always read during reading.

Value	Description
0	No effect
1	Set the CFIE bit in the CCR register.

#### [bit12] CLKSS: Internal prescaler selection set bit

- Writing "1" to this bit sets the CLKS bit in the CCR register.
- "0" is always read during reading.

Value	Description
0	No effect
1	Set the CLKS bit in the CCR register.

#### [bit11:8] Reserved: Reserved bits

## (2) Counter Control Set Register, Lower (CCSRL)

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved	CTUTS	UCRES	RLDES	Reserved	CGSCS	Reserved	
ACCESS_TYPE	R0,W0	R0,W	R0,W	R0,W	R0,W0	R0,W	R0,W0	
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	00	

**[bit7] Reserved: Reserved bit**

**[bit6] CTUTS: Counter write set bit**

- Writing "1" to this bit writes "1" to the CTUT bit in the CCR register.
- "0" is always read during reading.

Value	Description
0	No effect
1	Write "1" to the CTUT bit in the CCR register.

**[bit5] UCRES: Compare clear enable set bit**

- Writing "1" to this bit sets the UCRES bit in the CCR register.
- "0" is always read during reading.

Value	Description
0	No effect
1	Set the UCRES bit in the CCR register.

**[bit4] RLDES: Reload enable set bit**

- Writing "1" to this bit sets the RLDES bit in the CCR register.
- "0" is always read during reading.

Value	Description
0	No effect
1	Set the RLDES bit in the CCR register.

**[bit3] Reserved: Reserved bit**

**[bit2] CGSCS: Counter clear/gate function selection set bit**

- Writing "1" to this bit sets the CGSCS bit in the CCR register.
- "0" is always read during reading.

Value	Description
0	No effect
1	Set the CGSCS bit in the CCR register.

**[bit1:0] Reserved: Reserved bits**



## 4.12. Counter Control Clear Register (CCCR)

This register is used to clear bits in the counter control register (CCR).

### (1) Counter Control Clear Register, Upper (CCCRH)

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	M16EC	CDCFC	CFIEC	CLKSC	Reserved			
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W	R0,W0			
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0000			

#### [bit15] M16EC: 16-bit mode enable clear bit

- Writing "1" to this bit clears the M16E bit in the CCR register.
- "0" is always read during reading.

Value	Description
0	No effect
1	Clear the M16E bit in the CCR register.

#### [bit14] CDCFC: Count direction change detection clear bit

- Writing "1" to this bit clears the CDCF bit in the CCR register.
- "0" is always read during reading.

Value	Description
0	No effect
1	Clear the CDCF bit in the CCR register.

#### [bit13] CFIEC: Count direction change interrupt enable clear bit

- Writing "1" to this bit clears the CFIE bit in the CCR register.
- "0" is always read during reading.

Value	Description
0	No effect
1	Clear the CFIE bit in the CCR register.

#### [bit12] CLKSC: Internal prescaler selection clear bit

- Writing "1" to this bit clears the CLKS bit in the CCR register.
- "0" is always read during reading.

Value	Description
0	No effect
1	Clear the CLKS bit in the CCR register.

#### [bit11:8] Reserved: Reserved bits

## (2) Counter Control Clear Register, Lower (CCCRL)

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved		UCREC	RLDEC	UDCCC	CGSCC	Reserved	
ACCESS_TYPE	R0,W0		R0,W	R0,W	R0,W	R0,W	R0,W0	
PROT_TYPE	-							
INITIAL_VALUE	00		0	0	0	0	00	

### [bit7:6] Reserved: Reserved bits

### [bit5] UCREC: Compare clear enable clear bit

- Writing "1" to this bit clears the UCRE bit in the CCR register.
- "0" is always read during reading.

Value	Description
0	No effect
1	Clear the UCRE bit in the CCR register.

### [bit4] RLDEC: Reload enable clear bit

- Writing "1" to this bit clears the RLDE bit in the CCR register.
- "0" is always read during reading.

Value	Description
0	No effect
1	Clear the RLDE bit in the CCR register.

### [bit3] UDCCC: Counter clear clear bit

- Writing "1" to this bit writes "0" to the UDCC bit in the CCR register.
- "0" is always read during reading.

Value	Description
0	No effect
1	Write "0" to the UDCC bit in the CCR register.

### [bit2] CGSCC: Counter clear/gate function selection clear bit

- Writing "1" to this bit clears the CGSC bit in the CCR register.
- "0" is always read during reading.

Value	Description
0	No effect
1	Clear the CGSC bit in the CCR register.

### [bit1:0] Reserved: Reserved bits





### 4.13. Counter Status Set Register (CSSRL)

This register is used to set bits in the counter status register (CSRL).

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	CSTRS	CITES	UDIES	Reserved				
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W0				
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	00000				

#### [bit7] CSTRS: Count start set bit

- Writing "1" to this bit sets the CSTR bit in the CSRL register.
- "0" is always read during reading.

Value	Description
0	No effect
1	Set the CSTR bit in the CSRL register.

#### [bit6] CITES: Compare detection interrupt enable set bit

- Writing "1" to this bit sets the CITE bit in the CSRL register.
- "0" is always read during reading.

Value	Description
0	No effect
1	Set the CITE bit in the CSRL register.

#### [bit5] UDIES: Overflow/Underflow interrupt enable set bit

- Writing "1" to this bit sets the UDIE bit in the CSRL register.
- "0" is always read during reading.

Value	Description
0	No effect
1	Set the UDIE bit in the CSRL register.

#### [bit4:0] Reserved: Reserved bits

#### 4.14. Counter Status Clear Register (CSCRL)

This register is used to clear bits in the counter status register (CSRL).

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	CSTRC	CITEC	UDIEC	CMPFC	OVFFC	UDFFC	Reserved	
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W0	
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	00	

##### [bit7] CSTRC: Count start clear bit

- Writing "1" to this bit clears the CSTR bit in the CSRL register.
- "0" is always read during reading.

Value	Description
0	No effect
1	Clear the CSTR bit in the CSRL register.

##### [bit6] CITEC: Compare detection interrupt enable clear bit

- Writing "1" to this bit clears the CITE bit in the CSRL register.
- "0" is always read during reading.

Value	Description
0	No effect
1	Clear the CITE bit in the CSRL register.

##### [bit5] UDIEC: Overflow/Underflow interrupt enable clear bit

- Writing "1" to this bit clears the UDIE bit in the CSRL register.
- "0" is always read during reading.

Value	Description
0	No effect
1	Clear the UDIE bit in the CSRL register.

##### [bit4] CMPFC: Comparison result match detection flag clear bit

- Writing "1" to this bit clears the CMPF bit in the CSRL register.
- "0" is always read during reading.

Value	Description
0	No effect
1	Clear the CMPF bit in the CSRL register.

##### [bit3] OVFFC: Overflow detection flag clear bit

- Writing "1" to this bit clears the OVFF bit in the CSRL register.
- "0" is always read during reading.

Value	Description
0	No effect
1	Clear the OVFF bit in the CSRL register.



**[bit2] UDFFC: Underflow detection flag clear bit**

- Writing "1" to this bit clears the UDFF bit in the CSRL register.
- "0" is always read during reading.

Value	Description
0	No effect
1	Clear the UDFF bit in the CSRL register.

**[bit1:0] Reserved: Reserved bits**

## 4.15. Comparison Result Match Detection Flag Clear Register (CMPFCR)

This register is used to clear CMPF5 to CMPF0 in the CMPFR register.

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved		CMPF5C	CMPF4C	CMPF3C	CMPF2C	CMPF1C	CMPF0C
ACCESS_TYPE	R0,W0		R0,W	R0,W	R0,W	R0,W	R0,W	R0,W
PROT_TYPE	-							
INITIAL_VALUE	00		0	0	0	0	0	0

**[bit7:6] Reserved: Reserved bits**

**[bit5:0] CMPF5C to CMPF0C: Comparison result match detection clear bits**

- Writing "1" to this bit clears bits CMPF5 to CMPF0 in the CMPFR register.
- "0" is always read during reading.

Value	Description
0	No effect
1	Clear bits CMPF5 to CMPF0 in the CMPFR register.

(Example of a setting)

To clear CMPFR: CMPF5, write "1" to CMPF5C.



## 5. Precautions for Using This Device

The following shows the notes when using the up/down counter.

### (1) Notes When Accessing a Register

#### a) When Accessing to the Counter Status Register (CSRL)

- This register supports writing from the bit-band alias area. For the bit-band alias area, see "CHAPTER: Bit-band Unit".
- To clear a specific bit in this register, write "1" to the corresponding bit in the counter status clear register (CSCRL). It is prohibited to directly clear only a specific bit in this register.
- To set a specific bit in this register, write "1" to the corresponding bit in the counter status set register (CSSRL). It is prohibited to directly set only a specific bit in this register.
- Data can be written directly to this register only when writing to all bits.

#### b) When Accessing to the Counter Control Register (CCRH and CCRL)

- This register supports writing from the bit-band alias area. For the bit-band alias area, see "CHAPTER: Bit-Band Unit".
- To clear a specified bit of this register, clear the bit by writing "1" to the applicable bit of the counter control clear register (CCCRH and CCCRL). It is prohibited to directly clear only a specific bit in this register.
- To set a specified bit of this register, set the bit by writing "1" to the applicable bit of the counter control set register (CCSRH and CCSRL). It is prohibited to directly set only a specific bit in this register.
- Data can be written directly to this register only when writing to all bits.

#### c) When Accessing to the Comparison Result Match Detection Flag Register (CMPFR)

- This register supports writing from the bit-band alias area. For the bit-band alias area, see "CHAPTER: Bit-band Unit".
- To clear a specific bit in this register, write "1" to the corresponding bit in the comparison result match detection flag clear register (CMPFCR). It is prohibited to directly clear only a specific bit in this register.
- Data can be written directly to this register only when writing to all bits.

### (2) Notes on Up/Down Counter Operation

- The count direction immediately after a reset is the down count direction. Therefore, "1" is set in the CDCF bit, which indicates that the direction has changed, in up count immediately after a reset.
- If the up/down count register (UDCR) reaches the FULL count, the count continues with no carry. The up/down count register appears to be cleared, and the count continues.
- For the minimum pulse widths for the AIN, BIN, and ZIN pins, see the Data Sheet.



## CHAPTER 46: 16-bit Free-run Timer

This chapter explains 16-bit free-run timer.

---

1. Overview
2. Configuration
3. Operation
4. Registers
5. Notes



## 1. Overview

The 16-bit free-run timer supports the 16-bit up count mode and 16-bit up/down count mode. This timer can be used with the 16-bit input capture and the 16-bit output compare. This timer can measure the input pulse width and external clock cycle.

### Functions of the 16-bit Free-run Timer

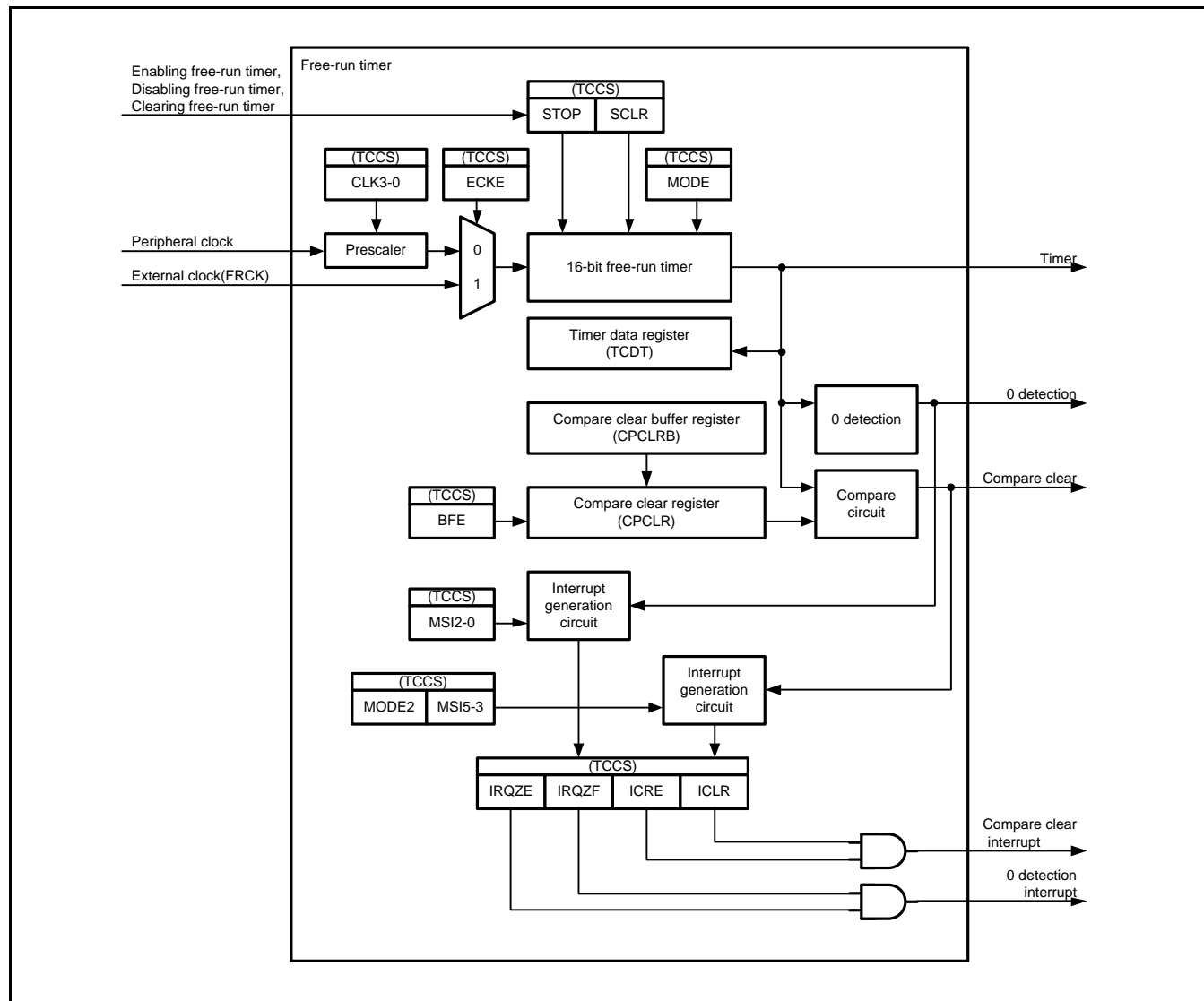
- The 16-bit free-run timer is composed of the 16-bit up/down counter, control register, 16-bit compare clear register, 16-bit compare clear buffer register, and prescaler.
- The 9 types of counter operation clocks ( $\phi$ ,  $\phi/2$ ,  $\phi/4$ ,  $\phi/8$ ,  $\phi/16$ ,  $\phi/32$ ,  $\phi/64$ ,  $\phi/128$ , and  $\phi/256$ ) ( $\phi$ : peripheral clock) can be selected.
- A compare clear interrupt is generated when the values of the compare clear register and the 16-bit counter are compared and their values match. A 0 detection interrupt is generated when the 16-bit counter detects the count value "0x0000".
- The compare clear register has a buffer register (the data written to this buffer register is transferred to the compare clear register). When the 16-bit counter is stopped, the data is transferred as soon as data is written to the buffer. When the 16-bit counter is operating, data is transferred from the buffer upon detecting the count value "0x0000".
- The counter value can be initialized to "0x0000" upon a reset, software clear or if a compare match is generated between this timer and the compare register values.
- The output value of the 16-bit counter can be used as a clock count of the output compare, the input capture and the A/D activation compare.

## 2. Configuration

This section shows a block diagram of 16-bit free-run timer.

Figure 2-1 is a configuration diagram of the 16-bit free-run timer.

Figure 2-1 Block Diagram of 16-bit Free-run Timer







### 3. Operation

This section provides a summary of the operation of the 16-bit free-run timer.

#### (1) Operation of 16-bit Free-run Timer

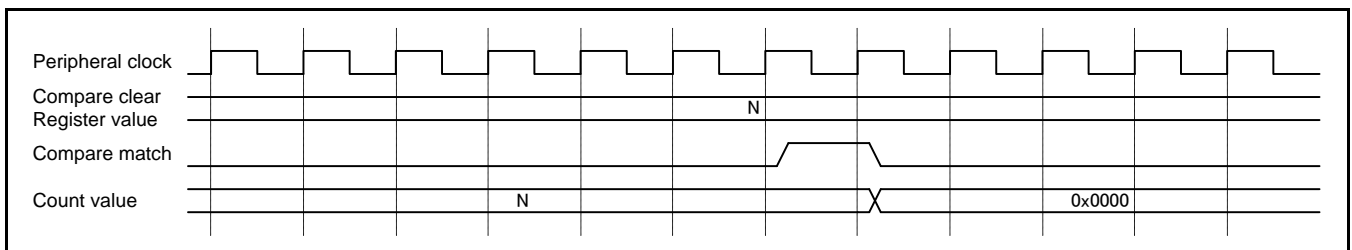
The 16-bit free-run timer starts counting from the value set in the timer data register (TCDT) after the timer is set to enabled (TCCS:STOP). When the 16-bit output compare and the 16-bit input capture are connected to the free-run timer, then the timer count value is used as base time of the 16-bit output compare and the 16-bit input capture.

#### (2) Counter Clear

The count value of the free-run timer is cleared to 0 when any of the following conditions is met.

- When the count value matches the value of the compare clear register (CPCLR) in the up count mode (MODE=0 of the timer state control register (TCCS))
- When "1" is written to the timer clear bit (SCLR) of the timer state control register (TCCS) while the free-run timer is operating (STOP=0 of the timer state control register (TCCS))
- When "0x0000" is written to the timer data register (TCDT) while the free-run timer is stopped (STOP=1 of the timer state control register (TCCS))
- When the hardware is reset. Upon reset, the counter is cleared immediately.
- When "1" is written to the timer clear bit (SCLR) of the timer state control register (TCCS) or when the count value matches the value of the compare clear register, the counter is cleared synchronously with the count timing.

Figure 3-1 Clear Timing of 16-bit Free-run Timer



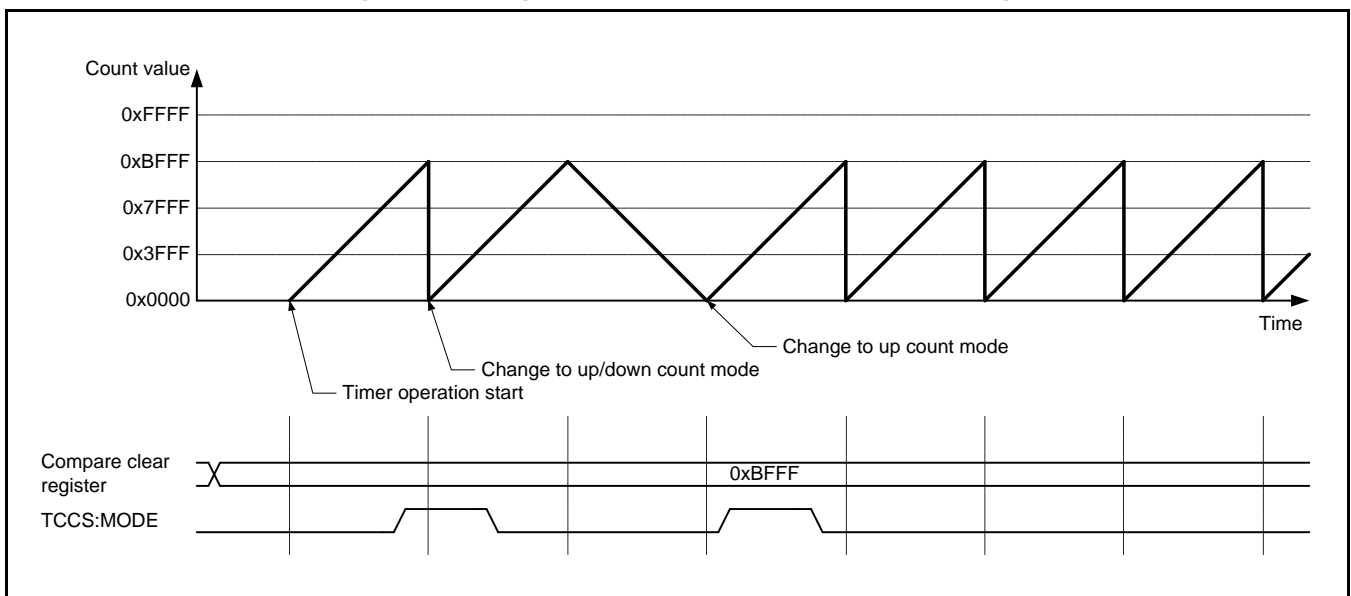
#### Notes:

- The count value of the free-run timer is not cleared if "1" is written to the timer clear bit (SCLR) of the timer state control register (TCCS) while the timer is stopped.
- If "0x0000" is written in TCDT register during the up/down counter mode (MODE=1 of timer state control register (TCCS)), an unintended counting may be performed. See Section "4.2 Timer Data Register (TCDT)" for the setting procedure of TCDT register during the up/down counter mode (MODE=1 of timer state control register (TCCS)).

### (3) Timer Mode

- Either of the following modes can be selected for the 16-bit free-run timer.
  - Up count mode (MODE=0 of the timer state control register (TCCS))
  - Up/down count mode (MODE=1 of the timer state control register (TCCS))
- In the up count mode, the counter starts counting from the timer data register (TCDT) that is set in advance. The counter continues to count up until the count value matches the value of the compare clear register (CPCLR). Then, the counter is cleared to "0x0000" and starts counting up again.
- In the up/down count mode, the counter starts counting from the timer data register (TCDT) that is set in advance. The counter continues to count up until the count value matches the value of the compare clear register (CPCLR). Then, the counter changes the counting direction from up count to down count. It continues to count down until the counter value reaches "0x0000" and starts counting up again.
- A value can be written to the timer count mode bit (MODE) of the timer state control register (TCCS) at any time, even when the timer is operating or stopped. The value written to this bit during the timer operation is stored in a buffer. The count mode changes when the count value becomes "0x0000".

Figure 3-2 Change of Timer Mode (While Timer is Operating)

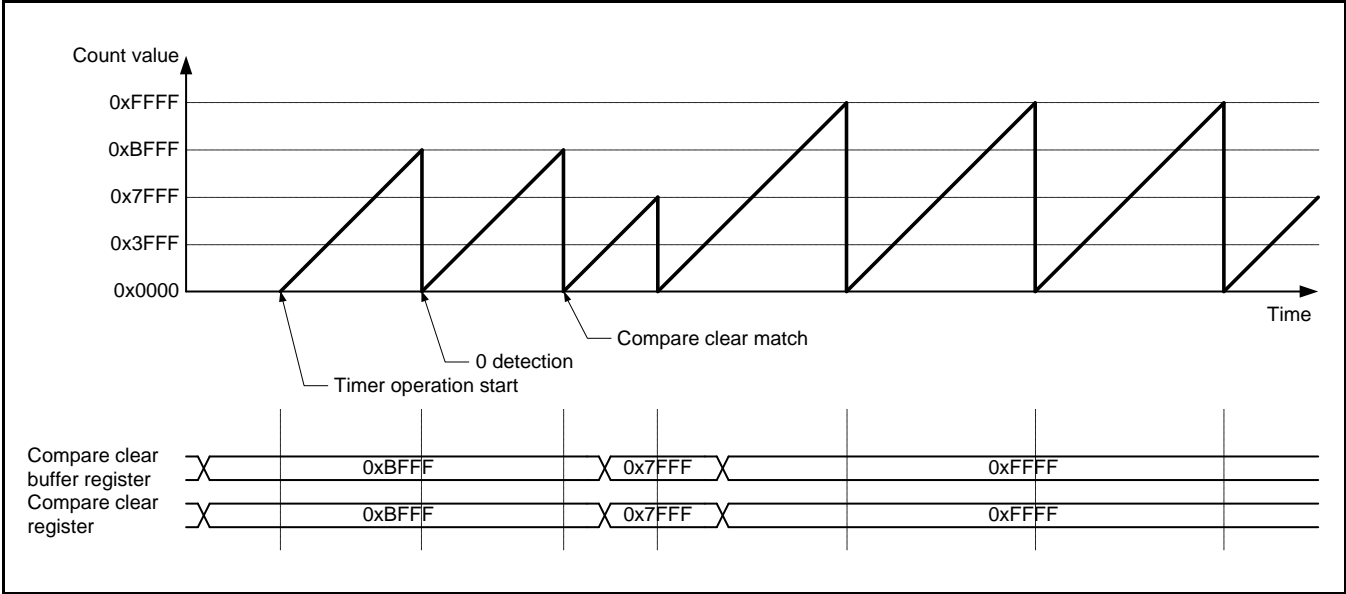


### (4) Compare Clear Buffer

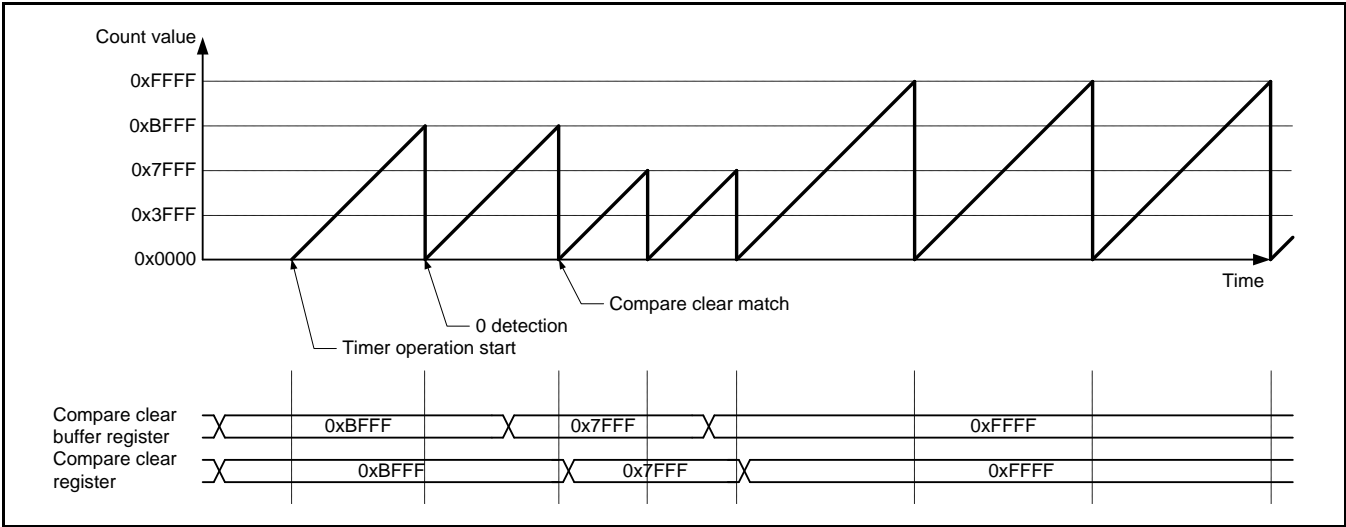
The compare clear register (CPCLR) has a buffer function that can be enabled or disabled. When the buffer function is enabled (BFE=1 of the timer state control register (TCCS)), the data written to the compare clear buffer register (CPCLRB) is transferred to the CPCLR register upon detecting the count value 0. When the buffer function is disabled (BFE=0 of the timer state control register (TCCS)), the data can be directly written to the compare clear register (CPCLR).



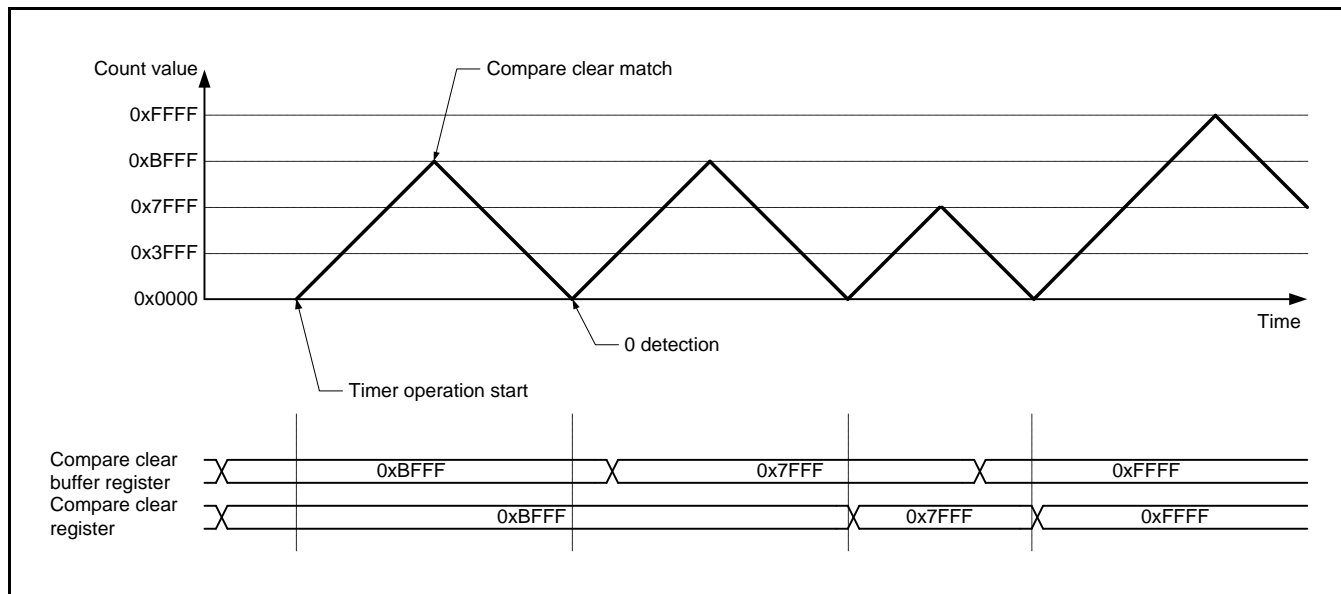
**Figure 3-3 Up Count Mode Operation When Compare Clear Buffer is Disabled  
(BFE=0 of Timer State Control Register (TCCS))**



**Figure 3-4 Up Count Mode Operation When Compare Clear Buffer is Enabled  
(BFE=1 of Timer State Control Register (TCCS))**



**Figure 3-5 Up/Down Count Mode Operation When Compare Clear Buffer is Enabled (BFE=1 of Timer State Control Register (TCCS))**



### (5) Timer Interrupt

The 16-bit free-run timer can generate the following 2 interrupts.

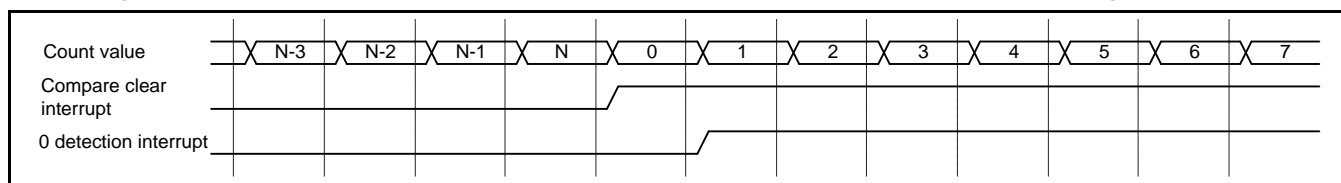
- Compare clear interrupt
- 0 detection interrupt

The compare clear interrupt is generated when the count value matches the value of the compare clear register. The 0 detection interrupt is generated when the count value reaches "0x0000".

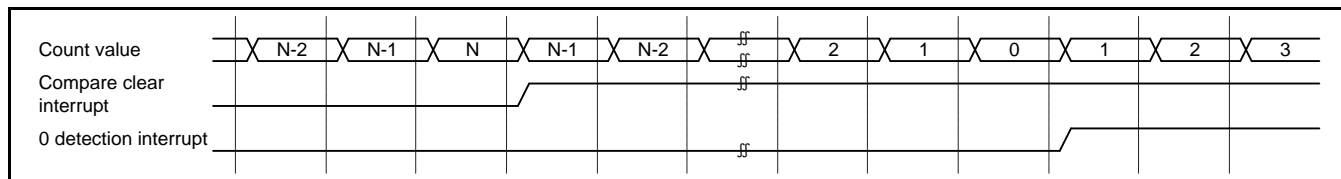
#### Note:

- The 0 detection interrupt is not generated when the timer is cleared (SCLR="1" of the timer state control register (TCCS)).

**Figure 3-6 Interrupt Generated in Up Count Mode (MODE=0 of the Timer State Control Register (TCCS))**



**Figure 3-7 Interrupt Generated in Up/Down Count Mode (MODE=1 of the Timer State Control Register (TCCS))**





#### (6) Interrupt Mask Function

Either the 0 detection interrupt or the compare clear interrupt, or both interrupts can be masked.

The following describes the case when either interrupt is masked.

- An interrupt request flag can be masked by setting the interrupt mask selection bits (MSI2 to MSI0) of the timer state control register (TCCS). The interrupt mask selection bits (MSI2 to MSI0) are the 3-bit reload down register that reloads the value when the mask count value reaches "0b000".
- The mask count value can also be loaded by directly writing data to the interrupt mask selection bits (MSI2 to MSI0). The number of mask counts is the value set in the interrupt mask selection bits (MSI2 to MSI0). An interrupt request flag is not masked when the mask count value (MSI2 to MSI0) becomes "0b000".
- The mask control of an interrupt request varies depending on the count mode (MODE of the timer state control register (TCCS)). In the up count mode (MODE=0), only the compare clear interrupt request flag can be masked and the 0 detection interrupt is generated every time a timer counter value of 0 is detected. In the up/down count mode (MODE=1), only the 0 detection interrupt request flag can be masked.

The following describes the case when both interrupt requests are masked.

- Both interrupts can be masked only when the free-run timer is in the up/down count mode (MODE=1) and when the timer extended state control register (TECCS) is set to MODE2=1 and the timer state control register (TCCS) is set to MODE=1.
- MSI2 to MSI0 bits of the timer state control register (TCCS) are used to mask the 0 detection interrupt. MSI5 to MSI3 bits of the timer extended state control register (TECCS) are used to mask the compare clear interrupt.

**Note:**

- *The 0 detection interrupt is not generated when the timer is cleared (SCLR=1 of the timer state control register (TCCS)).*



Figure 3-8 Compare Clear Interrupts to Be Masked in Up Count Mode

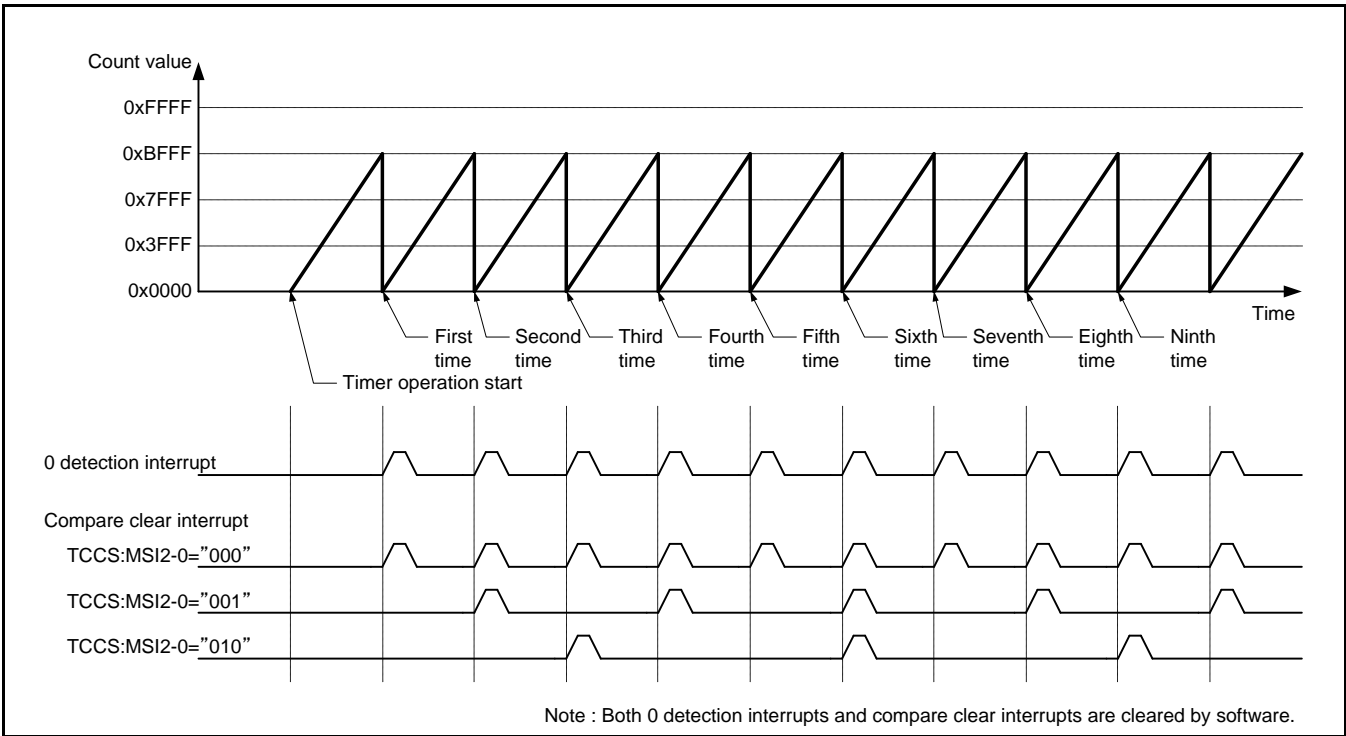


Figure 3-9 0 Detection Interrupts to Be Masked in Up/Down Count Mode

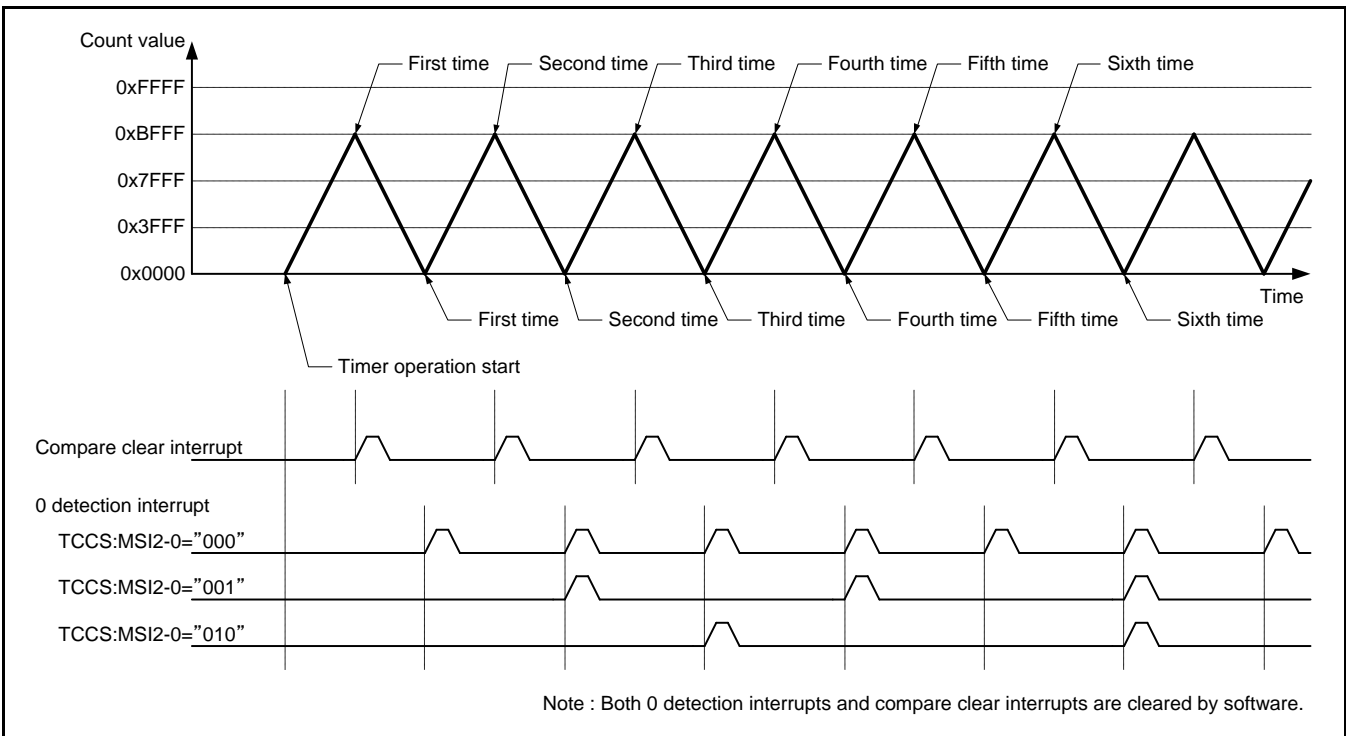
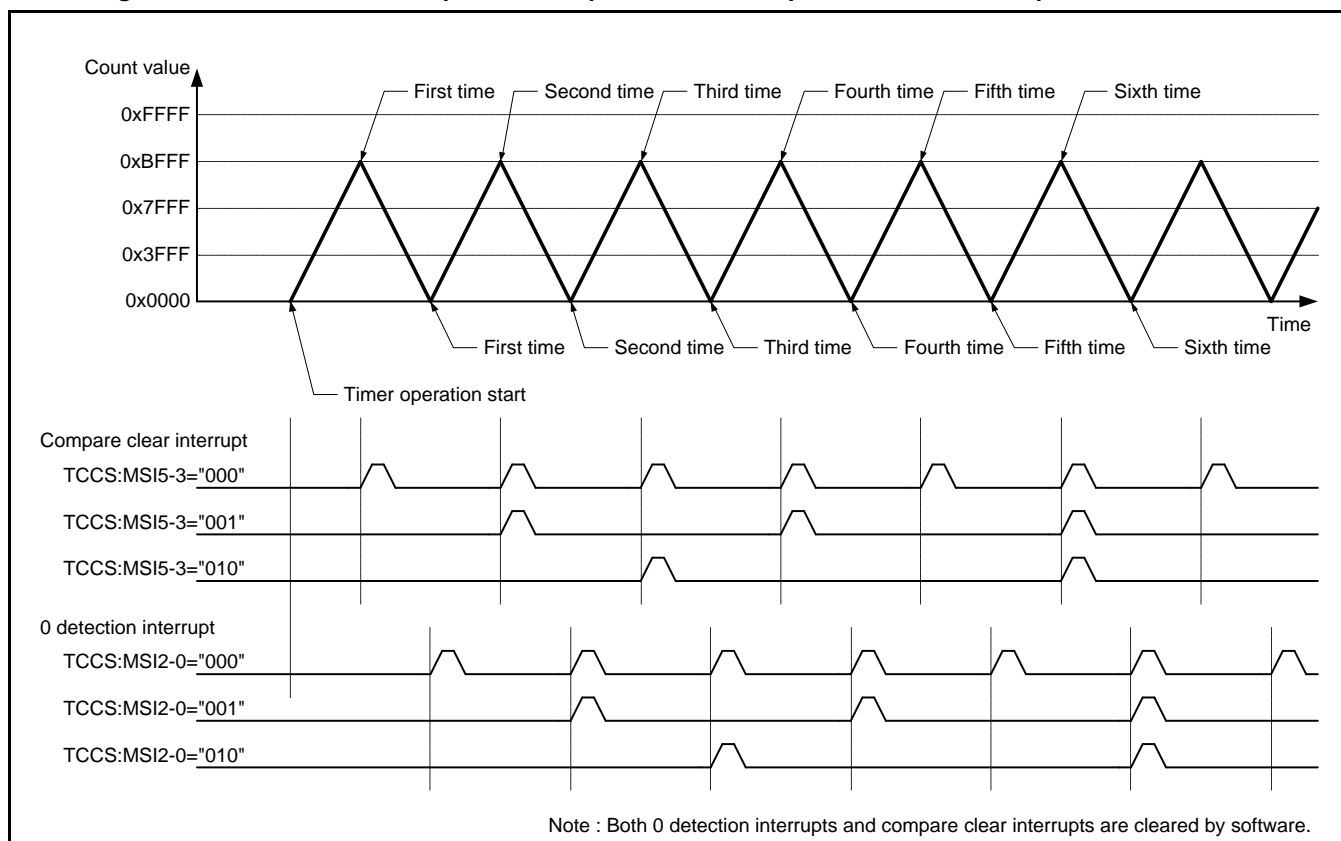


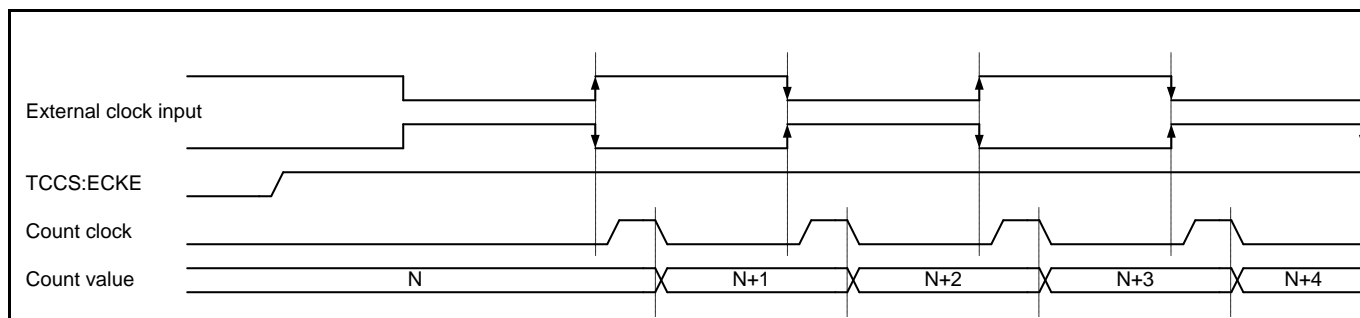
Figure 3-10 0 Detection Interrupts and Compare Clear Interrupts to be Masked in Up/Down Count Mode



### (7) Selected External Count Clock

The free-run timer counts based on an input clock (peripheral clock or external clock). If the external clock mode (ECKE=1 of the timer state control register (TCCS)) is selected, the free-run timer starts counting at a rising edge of the external input when the initial value of the external clock input (FRCK) is "H". After that, the free-run timer counts at both edges. When the initial value of the external input is "L", the free-run timer starts counting at a falling edge of the external input. After that, the free-run timer counts up at both edges.

Figure 3-11 Count Timing of Free-run Timer (In Up Count Mode)



**Note:**

- When the external clock input is used, the timer counts at both edges of the external clock.

### 3.1. Interrupts of the 16-bit Free-run Timer

There are the 2 types of interrupts as the interrupts of the 16-bit free-run timer: compare interrupt and 0 detection interrupt.

#### Free-run Timer Interrupt

Table 3-1 shows the interrupt control bits and the interrupt factor of the free-run timer.

**Table 3-1 Interrupt Control Bits and Interrupt Factor of Free-run Timer**

Control Bit and Factor	Free-run Timer	
	Compare Clear	0 Detection
Interrupt request flag bit	Compare clear interrupt flag bit (ICLR) of the timer state control register (TCCS)	0 detection interrupt flag bit (IRQZF) of the timer state control register (TCCS)
Interrupt request enable bit	Compare clear interrupt request enable bit (ICRE) of the timer state control register (TCCS)	0 detection interrupt request enable bit (IRQZE) of the timer state control register (TCCS)
Interrupt Factor	The value of the free-run timer matches the value of the compare clear register (CPCLR).	The free-run timer value becomes "0x0000".

When the value of the 16-bit free-run timer matches the compare clear register (CPCLR), ICLR of the timer state control register (TCCS) will be set. If interrupt requests are enabled (TCCS:ICRE="1") while this bit is set, an interrupt request is output to the interrupt controller.

When the timer value becomes "0x0000", IRQZF of the timer state control register (TCCS) will be set.

If interrupt requests are enabled (IRQZE of TCCS="1") while this bit is set, an interrupt request is output to the interrupt controller.





## 4. Registers

This section describes the registers of 16-bit free-run timer.

**Table 4-1 List of 16-bit Free-run Timer Registers**

Abbreviated Register Name	Register Name	See
FRT16Bxx_CPCLRB/ FRT16Bxx_CPCLR	Compare Clear Buffer Register/ Compare Clear Register	4.1
FRT16Bxx_TCDT	Timer Data Register	4.2
FRT16Bxx_TCCS	Timer State Control Register	4.3
FRT16Bxx_TCCSC	Timer State Clear Register	4.4
FRT16Bxx_TCCSS	Timer State Set Register	4.5

xx: channel number (xx=00 to 19)

## 4.1. Compare Clear Buffer Register (CPCLRB) / Compare Clear Register (CPCLR)

The compare clear buffer register (CPCLRB) is a buffer register of the compare clear register (CPCLR). The both registers are located at the same address.

### (1) Compare Clear Buffer Register (CPCLRB)

BIT_OFFSET	31-16
BIT_NAME	CL
ACCESS_TYPE	W
PROT_TYPE	-
INITIAL_VALUE	11111111_11111111

#### [bit31:16] CL[15:0]: Compare clear value buffer bits

- The compare clear buffer register (CPCLRB) is a buffer register located at the same address of the compare clear register (CPCLR).
- If the buffer function is disabled (BFE of timer state control register (TCCS) is 0) and the free-run timer stops, the value of the compare clear buffer register will be immediately transferred to the compare clear register (CPCLR).
- If the buffer function is enabled (TCCS:BFE="1"), the count value will be transferred to the compare clear register when the count value "0" of the 16-bit free-run timer is detected.

#### Notes:

- Do not set "0x0000" for the compare clear buffer register (CPCLRB).
- When accessing this register, use a half-word or word access instruction.



## (2) Compare Clear Register (CPCLR)

BIT_OFFSET	31-16
BIT_NAME	CL
ACCESS_TYPE	R
PROT_TYPE	-
INITIAL_VALUE	11111111_11111111

### [bit31:16] CL[15:0]: Compare clear value bits

- The compare clear register is used for comparison with the count value of the 16-bit free-run timer.
- In the up-count mode, if this register matches the count value of the 16-bit free-run timer, the 16-bit free-run timer will be reset to "0x0000".
- In the up/down count mode, if this register matches the count value of the 16-bit free-run timer, the 16-bit free-run timer will be converted from up count to down count or it will be converted from down count to up count when "0" is detected.

#### **Note:**

- *When accessing this register, use a half-word or word access instruction.*

## 4.2. Timer Data Register (TCDT)

The timer data register (TCDT) is used for reading the count value of the 16-bit free-run timer. It is also possible to set the count value of the 16-bit free-run timer.

BIT_OFFSET	15-0
BIT_NAME	T
ACCESS_TYPE	R/W
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000

### [bit15:0] T[15:0]: Timer data value bits

- The timer data register is used for reading the count value of the 16-bit free-run timer.
- The timer value can be set by writing a value to this register. However, a value needs to be written while the timer is inactive (STOP of timer state control register (TCCS) is 1).
- The count value of free-run timer is cleared to "0x0000" under any of the following conditions.
  - Hardware reset (after reset, the counter was cleared immediately)
  - The timer clear bit (SCLR) of the timer state control register (TCCS) is written "1" while the 16-bit free-run timer is operating (STOP=0 of the timer state control register (TCCS)).
  - The timer count value matches the compare clear register in the up-count mode (MODE of timer state control register (TCCS) is 0)
  - The timer data register (TCDT) is written "0x0000" while the 16-bit free-run timer is stopping (STOP=1 of the timer state control register (TCCS)).

When timer clear bit (SCLR) of the timer state control register (TCCS) is written "1" or the value of the compare clear register matches the timer count value, the counter was cleared synchronizing with count timing.

### Notes:

- The 16-bit free-run timer will not be cleared to "0x0000" even when the clear bit (SCLR) of the timer state control register (TCCS) is set to 1 while the 16-bit free-run timer is inactive (STOP=1 of timer state control register (TCCS)).
- When accessing the timer data register, use a half-word or word access instruction.
- If a counter value is written during the up/down counter mode (MODE=1 of timer state control register (TCCS)), an unintended counting may be performed. To write a counter value in the up/down counter mode (MODE=1 of timer state control register (TCCS)), perform the following steps.
  1. Stop the 16-bit free-run timer counter. (Writing 1 in STOP of timer state control register (TCCS))
  2. Set a counter value for the timer data register.
  3. Perform software clear. (Writing 1 in SCLR of timer state control register (TCCS))
  4. Start the 16-bit free-run timer counter.



### 4.3. Timer State Control Register (TCCS)

The timer state control register (TCCS) is a register used for controlling the operation of the 16-bit free-run timer.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	ECKE	IRQZF	IRQZE	MSI2	MSI1	MSI0	ICLR	ICRE
ACCESS_TYPE	R/W	R,W	R/W	R,W	R,W	R,W	R,W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	BFE	STOP	MODE	SCLR	CLK			
ACCESS_TYPE	R/W	R,W	R/W	R0,W	R/W			
PROT_TYPE	-							
INITIAL_VALUE	0	1	0	0	0000			

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved				MODE2	MSI5	MSI4	MSI3
ACCESS_TYPE	R0,W0				R/W	R,W	R,W	R,W
PROT_TYPE	-							
INITIAL_VALUE	0000				0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved							
ACCESS_TYPE	R1,W1							
PROT_TYPE	-							
INITIAL_VALUE	11111111							

#### [bit31] ECKE: Clock selection bit

- This bit is used for selecting the peripheral clock or external clock as a count clock for the 16-bit free-run timer.
- When this bit is set to "0":  
The peripheral clock is selected. To select the count clock frequency, you will also need to select the clock frequency selection bits (CLK3 to CLK0) of the TCCS register.
- When this bit is set to "1":  
The external clock (FRCK) is selected.
- This bit is cleared by writing "1" to the ECKEC bit of the TCCSC register.
- This bit is set by writing "1" to the ECKES bit of the TCCSS register.

Value	Description
0	Peripheral clock
1	External clock

#### Note:

- The count clock will be changed immediately as soon as this bit is set. Therefore, you need to change this bit while the output compare and input capture are inactive.

**[bit30] IRQZF: 0 detection interrupt flag bit**

- When the count value of the 16-bit free-run timer is set to "0x0000", this bit will be set to "1".
- When this bit is set to "0": This bit is cleared.
- When this bit is set to "1": This bit remains unaffected.
- This bit is cleared by writing "1" to the IRQZFC bit of the TCCSC register.

Value	Description	
	Read	Write
0	No 0 detected	This bit is cleared
1	0 detected	This bit remains unaffected

**Notes:**

- This bit will not be set by software clear (write "1" to the SCLR of the timer state control register (TCCS)) while the 16-bit free-run timer is active (STOP of timer state control register (TCCS) is 0).
- In the up/down count mode (MODE of the timer state control register (TCCS) is 1), this bit will be set to "1" when an interrupt configured by the interrupt mask selection bits (MSI2 to MSI0 of the timer state control register (TCCS) is other than "0b000") occurs. If no interrupt occurs, this bit will not be set to "1".
- In the up count mode (MODE=0), this bit will be set every time 0 detection occurs regardless of the value of MSI2 to MSI0.

**[bit29] IRQZE: 0 detection interrupt request enable bit**

- When this bit and interrupt flag bit (IRQZF) are set to "1", an interrupt request for CPU will be generated.
- This bit is cleared by writing "1" to the IRQZEC bit of the TCCSC register.
- This bit is set by writing "1" to the IRQZES bit of the TCCSS register.

Value	Description
0	Interrupt request disabled
1	Interrupt request enabled

**[bit28:26] MSI2 to MSI0: Interrupt mask selection bits**

- When MODE2 of the timer state control register (TCCS) is 0:
  - These bits are used for configuring the mask count of compare clear interrupt in the up count mode (MODE of the timer state control register (TCCS) is 0). In the up/down count mode (MODE of the timer state control register (TCCS) is 1), they are used to configure the mask count of 0 detection interrupt.
  - When this bit is set to "0b000", the interrupt factor will not be masked.
- When MODE2 of the timer state control register (TCCS) is 1:
  - In the up/down count mode (MODE of the timer state control register (TCCS) is 1), these bits are used to configure the mask count of 0 detection interrupt.
  - Settings of the up count mode (MODE of the timer state control register (TCCS) is 0) are disabled.

Value			Description
MSI2	MSI1	MSI0	
0	0	0	An interrupt will be generated when there is a match for the first time
0	0	1	An interrupt will be generated when there is a match for the second time
0	1	0	An interrupt will be generated when there is a match for the third time
0	1	1	An interrupt will be generated when there is a match for the fourth time
1	0	0	An interrupt will be generated when there is a match for the fifth time
1	0	1	An interrupt will be generated when there is a match for the sixth time

Value			Description
MSI2	MSI1	MSI0	
1	1	0	An interrupt will be generated when there is a match for the seventh time
1	1	1	An interrupt will be generated when there is a match for the eighth time

**Notes:**

- The value read is a mask counter value.
- The written data will be written to the mask register.
- The written value to the mask register while the free-run timer is active (STOP of the timer state control register (TCCS) is 0) will be reloaded to the counter only when the mask counter becomes "0".
- The written value to the mask register while the free-run timer is inactive (STOP of the timer state control register (TCCS) is 1) will be immediately reloaded to the counter.

**[bit25] ICLR: Compare clear interrupt flag bit**

- This bit will be set to "1" when the compare clear value matches the 16-bit free-run timer value.
- When this bit is set to "0": This bit is cleared.
- When this bit is set to "1": This bit remains unaffected.
- This bit is cleared by writing "1" to the ICLRC bit of the TCCSC register.

Value	Description	
	Read	Write
0	No compare clear match	This bit is cleared
1	Compare clear match	This bit remains unaffected.

**Notes:**

- In the up count mode (MODE of the timer state control register (TCCS) is 0), this bit will be set to "1" when an interrupt configured by the interrupt mask selection bits occurs. If no interrupt occurs, this bit will not be set to "1".
- In the up/down count mode (MODE of the timer state control register (TCCS) is 1), this bit will be set every time a compare clear occurs regardless of the value of the MSI2 to MSI0 bits.

**[bit24] ICRE: Compare clear interrupt request enable bit**

- When this bit and compare clear interrupt flag bit (ICLR) are set to "1", an interrupt request for CPU will be generated.
- This bit is cleared by writing "1" to the ICREC bit of the TCCSC register.
- This bit is set by writing "1" to the ICRES bit of the TCCSS register.

Value	Description
0	Interrupt request disabled
1	Interrupt request enabled

**[bit23] BFE: Compare clear buffer enable bit**

- This bit is used for validating the compare clear buffer register (CPCLRB).
- When this bit is set to "0":  
Compare clear buffer register (CPCLRB) will be invalidated. Thus, you can write to the compare clear register (CPCLR) directly.
- When this bit is set to "1":

Compare clear buffer register (CPCLRB) will be validated. Data written to and retained in the compare clear buffer register (CPCLRB) will be transferred to the compare clear register once the count value "0" from the 16-bit free-run timer has been detected.

- This bit is cleared by writing "1" to the BFEC bit of the TCCSC register.
- This bit is set by writing "1" to the BFES bit of the TCCSS register.

Value	Description
0	Invalidate the compare clear buffer
1	Validate the compare clear buffer

#### [bit22] STOP: Timer enable bit

- This bit is used to start/stop counting of the 16-bit free-run timer.
- When this bit is set to "0":
  - Start counting the 16-bit free-run timer.
- When this bit is set to "1":
  - Stop counting the 16-bit free-run timer.
- The free-run timer will not be initialized even when the SCLR of the timer state control register (TCCS) is set to "1" while the free-run timer is inactive (this bit=0).
- The value to be reflected to this bit is the one specified at the GSTOP bit of the timer synchronous activation register (TCGS) while the FRT bit of the timer synchronous activation enable register (TCGSE) is set to "1".
- This bit is cleared by writing "1" to the STOPC bit of the TCCSC register.
- This bit is set by writing "1" to the STOPS bit of the TCCSS register.

Value	Description
0	Enable counting (Start the counting)
1	Disable counting (Stop the counting)

#### [bit21] MODE: Timer count mode bit

- This bit is used to select the count mode of the 16-bit free-run timer.
- When this bit is set to "0":
  - The up count mode is selected. The timer continues to count up until the count value matches the compare clear register to be reset to "0x0000". After that, it starts counting up again.
- When this bit is set to "1":
  - The up/down count mode is selected. The timer continues to count up until the count value matches the compare clear register. After that, it will be converted to down count. Then, when the count value reaches to "0x0000", it will change to up count once again.
- You can write to this bit regardless of the timer is active or inactive. If the timer is active, the value written to this bit will be transferred to the buffer. Then, when the timer value becomes "0x0000", the count mode changes based on the buffer value.
- This bit is cleared by writing "1" to the MODEC bit of the TCCSC register.
- This bit is set by writing "1" to the MODES bit of the TCCSS register.

Value	Description
0	Up count mode
1	Up/down count mode



### [bit20] SCLR: Timer clear bit

- This bit is used to initialize the 16-bit free-run timer.
- Initialization of the 16-bit free-run timer:  
When this bit is set to "1" while the 16-bit free-run timer is active (STOP of the timer state control register (TCCS) is 0), the 16-bit free-run timer will be initialized to "0x0000" in the next count clock. The 16-bit free-run timer will not be initialized when this bit is set to "1" while the 16-bit free-run timer is inactive (STOP of the timer state control register (TCCS) is 1).
- The value read out is always "0".
- The value to be reflected to this bit is the one specified at the GSTOP bit of the timer synchronous activation register (TCGS) while the FRT bit of the timer synchronous activation enable register (TCGSE) is set to "1". See "CHAPTER: Free-run Timer Selection and Simultaneous Activation".
- This bit is set by writing "1" to the SCLRS bit of the TCCSS register.

Value	Description	
	Read	Write
0	"0" is always read out.	Counter will not be initialized.
1		Counter will be initialized to "0x0000".

### Notes:

- Writing "1" to this bit will not generate the 0 detection interrupt.
- If you write "0" to this bit prior to the next count clock after setting "1", the timer clear will not be executed.
- In the up/down count mode, when "0" is written to the SCLR bit before updating the timer count after "1" is written to the SCLR bit while the timer is counting down, the count value is not updated and the count direction is changed to up count.

### [bit19:16] CLK[3:0]: Clock frequency selection bits

- These bits are used to select the count clock frequency of the 16-bit free-run timer.
- The clock frequency is changed immediately upon setting these bits.

Value	Description					
	Count Clock	$\phi=40\text{MHz}$	$\phi=20\text{MHz}$	$\phi=10\text{MHz}$	$\phi=5\text{MHz}$	$\phi=2.5\text{MHz}$
0000	$\phi$	25ns	50ns	100ns	200ns	400ns
0001	$\phi / 2$	50ns	100ns	200ns	400ns	800ns
0110	$\phi / 4$	100ns	200ns	400ns	800ns	1.6 $\mu\text{s}$
0011	$\phi / 8$	200ns	400ns	800ns	1.6 $\mu\text{s}$	3.2 $\mu\text{s}$
0100	$\phi / 16$	400ns	800ns	1.6 $\mu\text{s}$	3.2 $\mu\text{s}$	6.4 $\mu\text{s}$
0101	$\phi / 32$	800ns	1.6 $\mu\text{s}$	3.2 $\mu\text{s}$	6.4 $\mu\text{s}$	12.8 $\mu\text{s}$
0110	$\phi / 64$	1.6 $\mu\text{s}$	3.2 $\mu\text{s}$	6.4 $\mu\text{s}$	12.8 $\mu\text{s}$	25.6 $\mu\text{s}$
0111	$\phi / 128$	3.2 $\mu\text{s}$	6.4 $\mu\text{s}$	12.8 $\mu\text{s}$	25.6 $\mu\text{s}$	51.2 $\mu\text{s}$
1000	$\phi / 256$	6.4 $\mu\text{s}$	12.8 $\mu\text{s}$	25.6 $\mu\text{s}$	51.2 $\mu\text{s}$	102.4 $\mu\text{s}$
Other settings disabled	-	-	-	-	-	-

$\phi$ : peripheral clock

### [bit15:12] Reserved: Reserved bits

**[bit11] MODE2: Interrupt mask mode bit 2**

- In the up/down count mode (MODE of the timer state control register (TCCS) is 1) of the 16-bit free-run timer, this bit will be used to mask the 0 detection interrupt and compare clear interrupt independently.
- During the MODE="1" of the timer state control register (TCCS) and if this bit is set to 1, the value configured at MSI5 to MSI3 of this register becomes valid and the compare clear interrupt is masked for the number of times specified. For the mask count of 0 detection interrupt, the value configured at MSI2 to MSI0 of the timer state control register (TCCS) becomes valid.
- This bit is cleared by writing "1" to the MODE2C bit of the TCCSC register.
- This bit is set by writing "1" to the MODE2S bit of the TCCSS register.

Value		Description
MODE2	MODE*	
0	0	Value set for MSI5 to MSI3 will be invalid
0	1	Value set for MSI5 to MSI3 will be invalid
1	0	Setting disabled (operation is not guaranteed)
1	1	Value set for MSI5 to MSI3 will be valid

\*: bit21 of the timer state control register (TCCS)

**Note:**

- During MODE="0" of the timer state control register (TCCS) and if this bit is set to "1", the operation is not guaranteed.

**[bit10:8] MSI5 to MSI3: Interrupt mask selection bits**

- These bits, which are used to configure the mask count of compare clear interrupt, are valid only when MODE of the timer state control register (TCCS) as well as MODE2 of this register are 1. Value that can be configured for the mask count of 0 detection interrupt is MSI2 to MSI0 of the timer state control register (TCCS).
- When these bits are set to "0b000", the compare clear interrupt factor will not be masked.

Value			Description
MSI5	MSI4	MSI3	
0	0	0	An interrupt will be generated when there is a match for the first time
0	0	1	An interrupt will be generated when there is a match for the second time
0	1	0	An interrupt will be generated when there is a match for the third time
0	1	1	An interrupt will be generated when there is a match for the fourth time
1	0	0	An interrupt will be generated when there is a match for the fifth time
1	0	1	An interrupt will be generated when there is a match for the sixth time
1	1	0	An interrupt will be generated when there is a match for the seventh time
1	1	1	An interrupt will be generated when there is a match for the eighth time

**Note:**

- The value read is a mask counter value.
- The written data will be written to the mask register.
- The written value to the mask register while the free-run timer is active (STOP of the timer state control register (TCCS) is 0) will be reloaded to the counter only when the mask counter becomes "0".
- The written value to the mask register while the free-run timer is inactive (STOP of the timer state control register (TCCS) is 1) will be immediately reloaded to the counter.

**[bit7:0] Reserved: Reserved bits**



## 4.4. Timer State Clear Register (TCCSC)

The timer state control clear register (TCCSC) is a register that is used to clear bits of the timer state control register (TCCS).

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	ECKEC	IRQZFC	IRQZEC	Reserved			ICLRC	ICREC
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W0			R0,W	R0,W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	000			0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	BFEC	STOPC	MODEC	Reserved				
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W0				
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	00000				

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved				MODE2C	Reserved		
ACCESS_TYPE	R0,W0				R0,W	R0,W0		
PROT_TYPE	-							
INITIAL_VALUE	0000				0	000		

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved							
ACCESS_TYPE	R0,W0							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

### [bit31] ECKEC: Clock selection clear bit

Value	Description
0	No effect
1	Clear clock selection bit

### [bit30] IRQZFC: 0 detection interrupt flag clear bit

Value	Description
0	No effect
1	Clear 0 detection interrupt flag bit

### [bit29] IRQZEC: 0 detection interrupt request enable clear bit

Value	Description
0	No effect
1	Clear 0 detection interrupt request enable bit

### [bit28:26] Reserved: Reserved bits

**[bit25] ICLRC: Compare clear interrupt flag clear bit**

Value	Description
0	No effect
1	Clear compare clear interrupt flag bit

**[bit24] ICREC: Compare clear interrupt request enable clear bit**

Value	Description
0	No effect
1	Clear compare clear interrupt request enable bit

**[bit23] BFEC: Compare clear buffer enable clear bit**

Value	Description
0	No effect
1	Clear compare clear buffer enable bit

**[bit22] STOPC: Timer enable clear bit**

Value	Description
0	No effect
1	Clear timer enable bit

**[bit21] MODEC: Timer count mode clear bit**

Value	Description
0	No effect
1	Clear timer count mode bit

**[bit20:12] Reserved: Reserved bits**

**[bit11] MODE2C: Interrupt mask mode 2 clear bit**

Value	Description
0	No effect
1	Clear interrupt mask mode 2 bit

**[bit10:0] Reserved: Reserved bits**



## 4.5. Timer State Set Register (TCCSS)

The timer state control set register (TCCSC) is a register that is used to set bits of the timer state control register (TCCS).

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	ECKES	Reserved	IRQZES	Reserved				ICRES
ACCESS_TYPE	R0,W	R0,W0	R0,W	R0,W0				R0,W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0000				0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	BFES	STOPS	MODES	SCLRS	Reserved			
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W	R0,W0			
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0000			

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved				MODE2S	Reserved		
ACCESS_TYPE	R0,W0				R0,W	R0,W0		
PROT_TYPE	-							
INITIAL_VALUE	0000				0	000		

BITS_OFFSET	7	6	5	4	3	2	1	0
BITS_NAME	Reserved							
ACCESS_TYPE	R0,W0							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

### [bit31] ECKES: Clock selection set bit

Value	Description
0	No effect
1	Set clock selection bit

### [bit30] Reserved: Reserved bit

### [bit29] IRQZES: 0 detection interrupt request enable set bit

Value	Description
0	No effect
1	Set 0 detection interrupt request enable bit

### [bit28:25] Reserved: Reserved bits

**[bit24] ICRES: Compare clear interrupt request enable set bit**

Value	Description
0	No effect
1	Set compare clear interrupt request enable bit

**[bit23] BFES: Compare clear buffer enable set bit**

Value	Description
0	No effect
1	Set compare clear buffer enable bit

**[bit22] STOPS: Timer enable set bit**

Value	Description
0	No effect
1	Set timer enable bit

**[bit21] MODES: Timer count mode set bit**

Value	Description
0	No effect
1	Set timer count mode bit

**[bit20] SCLRS: Timer clear set bit**

Value	Description
0	No effect
1	Set timer clear bit

**[bit19:12] Reserved: Reserved bits**

**[bit11] MODE2S: Interrupt mask mode 2 set bit**

Value	Description
0	No effect
1	Set interrupt mask mode 2 bit

**[bit10:0] Reserved: Reserved bits**



## 5. Notes

The following shows the notes when using the 16-bit free-run timer.

### (1) Notes when Accessing a Register

#### a) When Accessing the Compare Clear Register (CPCLR) or the Compare Clear Buffer Register (CPCLRB)

Use word access instructions to the compare clear register (CPCLR) and the compare clear buffer register (CPCLRB).

#### b) When Accessing the Timer State Control Register (TCCS)

- This register supports writing from the bit-band alias area. For the bit-band alias area, see "CHAPTER: Bit-band Unit".
- To clear a specified bit of this register, clear the bit by writing "1" to the applicable bit of the timer state control clear register (TCCSC).
- To set a specified bit of this register, set the bit by writing "1" to the applicable bit of the timer state control set register (TCCSS).
- Data can be written directly to this register only when writing to all bits.
- In the normal reading mode, the interrupt mask counter value is read from MSI2 to MSI0.

### (2) Notes when Operating the Free-run Timer

#### When Setting Using a Program

- When the hardware is reset, the count value becomes "0x0000", but the 0 detection interrupt flag is not set.
- Because the timer mode bit (MODE of the timer state control register (TCCS)) has a buffer, a specified timer mode is enabled after 0 is detected.
- The timer clear bit (SCLR=1 of the timer state control register (TCCS)) initializes the timer. However, this bit does not generate the 0 detection interrupt.
- The compare clear flag is not set if the timer starts counting when the value of the compare clear register (CPCLR) matches the count value.
- Set any values other than 0 to the compare clear register (CPCLR). Note that the following operations occur if 0 is set.
  - When the timer mode bit (MODE of the timer state control register (TCCS)) is in the up count mode (MODE=0), the count value is updated to 0 and fixed to 0. Then, the 0 detection interrupt flag and the compare clear flag are set at every count clock.
  - When the timer count mode bit (MODE of the timer state control register (TCCS)) is in the up/down count mode (MODE=1), the count value is counted up from "0x0000" to "0xFFFF", and this up count operation is repeated. The 0 detection interrupt flag and the compare clear flag are set to "1" when the count value becomes 0.



# CHAPTER 47: Free-run Timer Selection and Simultaneous Activation

This chapter describes the selection and simultaneous activation functions of the 16-bit free-run timer.

---

1. Overview
2. Configuration
3. Operation
4. Setting Procedure Example
5. Registers





## 1. Overview

This section provides a summary of the selection and simultaneous activation of the 16-bit free-run timer.

The free-run timer consists of 3 free-run timer simultaneous activation functions, 18 16-bit free-run timers (1 channel for each: 18 channels in total), and 3 free-run timer selection functions.

### (1) Free-run Timer Selection Function

- The free-run timer selection register can be used to select the 16-bit free-run timer assignment from 16-bit output compare, 16-bit input capture, A/D activation compare, and 4ch A/D activation compare.
- 3 free-run timer selection functions are mounted. They respectively correspond to 16-bit free-run timer ch.0 to ch.5, ch.6 to ch.11, and ch.12 to ch.17.

### (2) Simultaneous Activation Function for Free-run Timers

- This function can simultaneously activate/clear the specified 16-bit free-run timers.
- It simultaneously controls the timer enable bit (STOP) and timer clear bit (SCLR) of the timer state register (TCCS) for each of the 16-bit free-run timers for which free-run timer simultaneous activation is enabled.
- If simultaneous activation/clear is not performed, each of the 16-bit free-run timers can be activated/cleared individually by setting the timer enable bit (STOP) and timer clear bit (SCLR) of the timer state register (TCCS).
- 3 simultaneous activation functions for free-run timers are mounted. They can respectively control 16-bit free-run timer ch.0 to ch.5, ch.6 to ch.11, and ch.12 to ch.17.
- 16-bit free-run timer ch.6 to ch.11 can be used with a different activation source. This enables 16-bit free-run timer ch.6 to ch.17 to be activated/cleared simultaneously.

### (3) Function to Indicate Free-run Timer Direction

- The current count direction can be found by reading the free-run timer direction indication registers.
- 3 free-run timer count direction indication registers are mounted. They respectively correspond to free-run timer ch.0 to ch.5, ch.6 to ch.11, and ch.12 to ch.17.

## 2. Configuration

This section shows block diagram of the selection and simultaneous activation functions of the 16-bit free-run timer.

### (1) Block Diagram of Free-run Timer Selection Functions



Figure 2-1 Block Diagram of Free-run Timer Selection 0/1

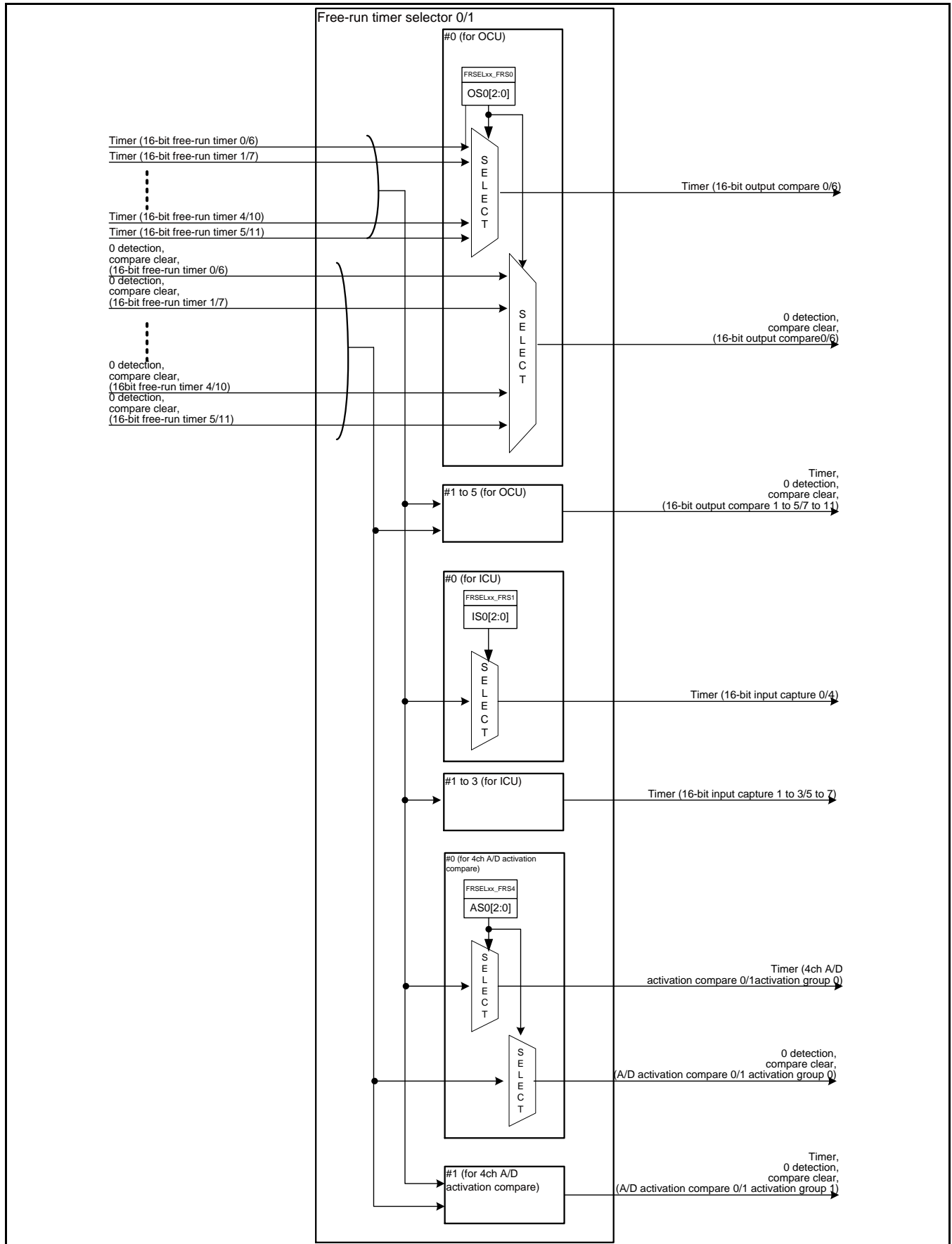
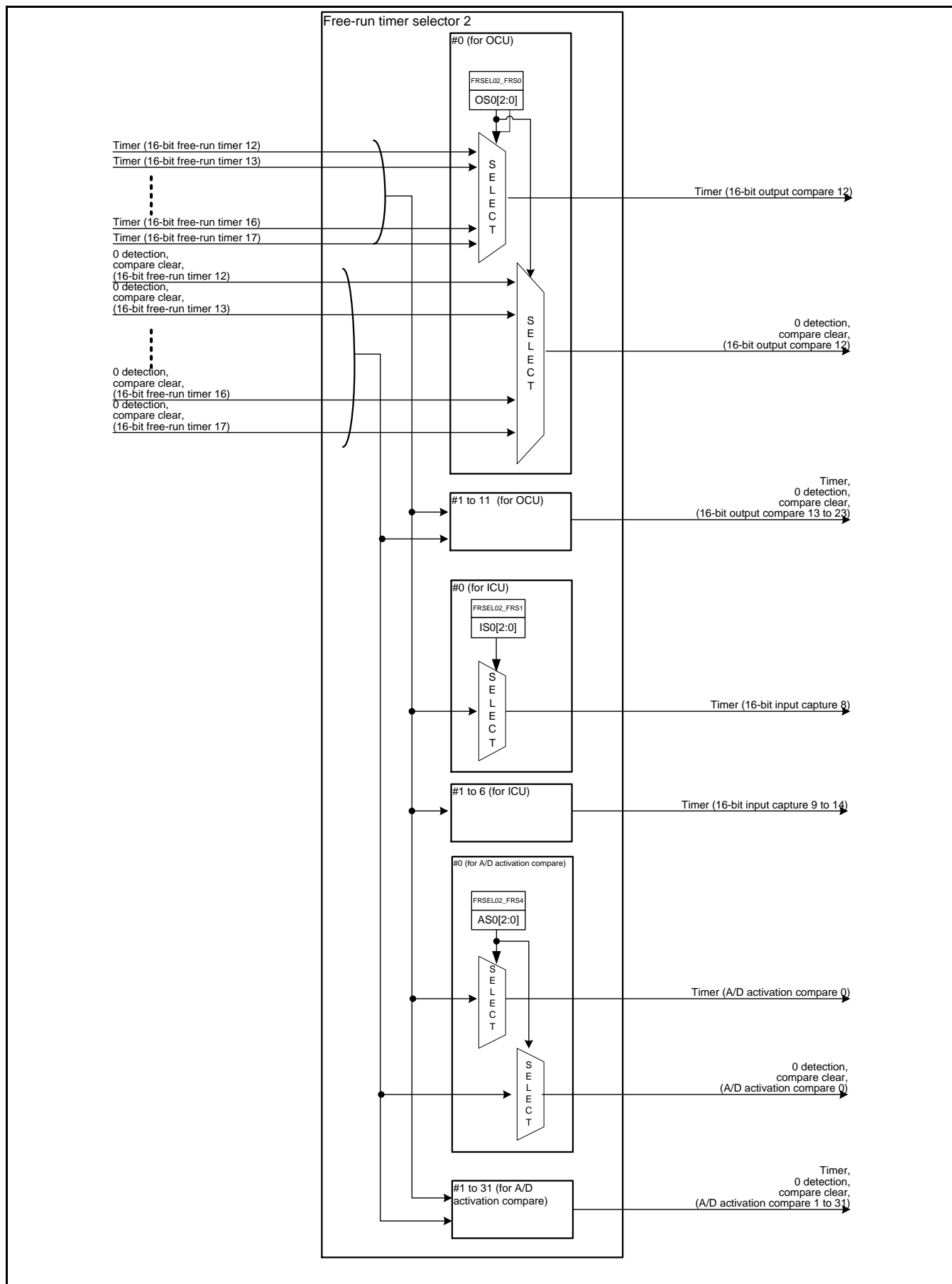


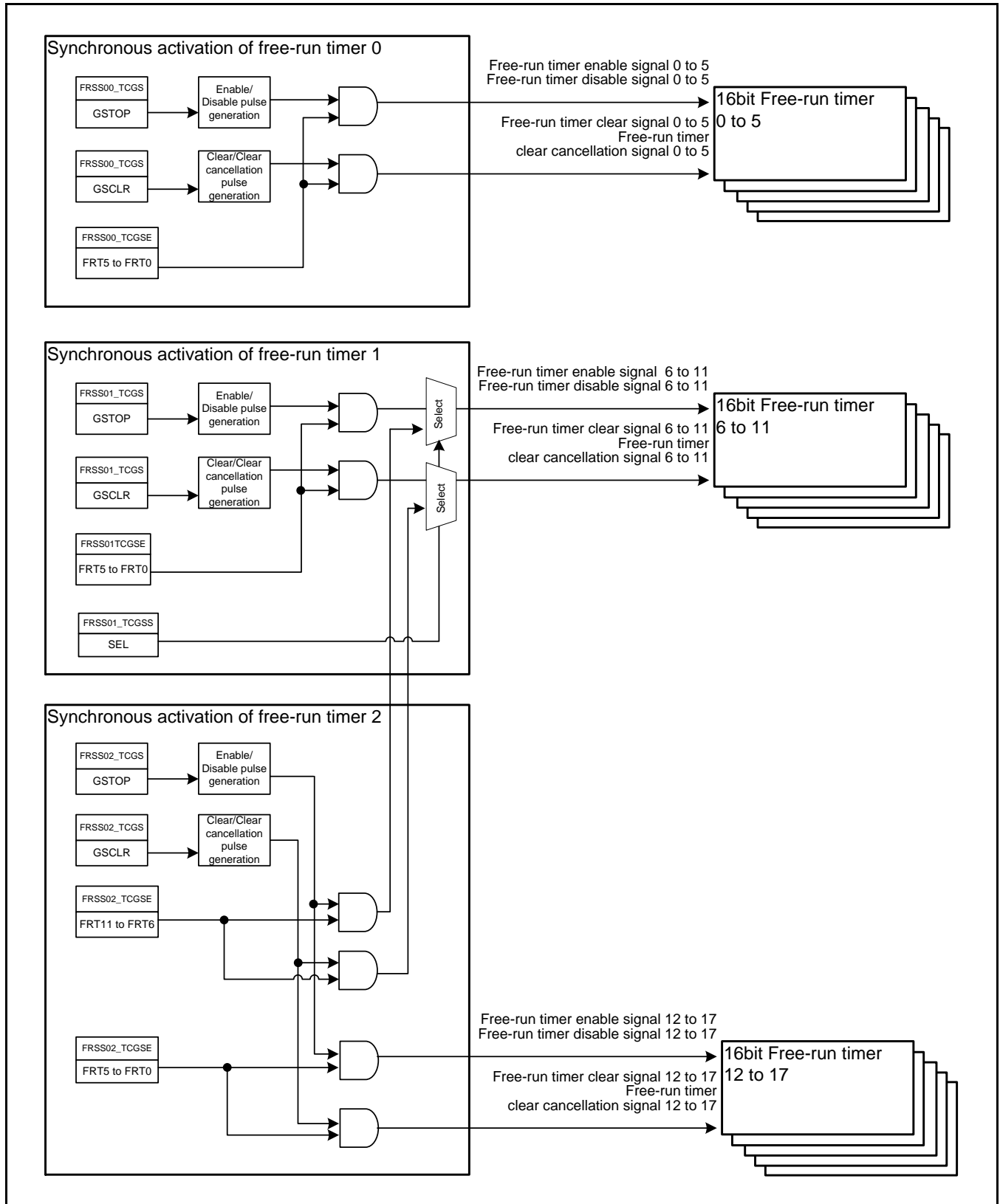
Figure 2-2 Block Diagram of Free-run Timer Selection 2





## (2) Block Diagram of Free-run Timer Simultaneous Activation

Figure 2-3 Block Diagram of Free-run Timer Simultaneous Activation



### 3. Operation

This section explains the operations of the 16-bit free-run timer selection and simultaneous activation.

#### (1) Operation of Free-run Timer Selection

- The free-run timer selection register can be used to select the free-run timer assignment from 16-bit output capture, 16-bit input capture, A/D activation compare, and 4ch A/D activation compare.
- 3 free-run timer selectors are mounted. They respectively correspond to 16-bit free-run timer ch.0 to ch.5, ch.6 to ch.11, and ch.12 to ch.17.

The relationship between connection destination resources and the corresponding registers is as follows.

**Table 3-1 Relationship between Resources and Registers of Free-run Timer Selection 0**

Free-run Timer	Resource	Register
Free-run Timer ch.0 to ch.5	16-bit output compare ch.0	FRSEL00_FRS0:OS0[2:0]
	16-bit output compare ch.1	FRSEL00_FRS0:OS1[2:0]
	16-bit output compare ch.2	FRSEL00_FRS0:OS2[2:0]
	16-bit output compare ch.3	FRSEL00_FRS0:OS3[2:0]
	16-bit output compare ch.4	FRSEL00_FRS0:OS4[2:0]
	16-bit output compare ch.5	FRSEL00_FRS0:OS5[2:0]
	16-bit input capture ch.0	FRSEL00_FRS1:IS0[2:0]
	16-bit input capture ch.1	FRSEL00_FRS1:IS1[2:0]
	16-bit input capture ch.2	FRSEL00_FRS1:IS2[2:0]
	16-bit input capture ch.3	FRSEL00_FRS1:IS3[2:0]
	4ch A/D activation compare 0, activation group 0	FRSEL00_FRS4:AS0[2:0]
	4ch A/D activation compare 0, activation group 1	FRSEL00_FRS4:AS1[2:0]

**Table 3-2 Relationship between Resources and Registers of Free-run Timer Selection 1**

Free-run Timer	Resource	Register
Free-run Timer ch.6 to ch.11	16-bit output compare ch.6	FRSEL01_FRS0:OS0[2:0]
	16-bit output compare ch.7	FRSEL01_FRS0:OS1[2:0]
	16-bit output compare ch.8	FRSEL01_FRS0:OS2[2:0]
	16-bit output compare ch.9	FRSEL01_FRS0:OS3[2:0]
	16-bit output compare ch.10	FRSEL01_FRS0:OS4[2:0]
	16-bit output compare ch.11	FRSEL01_FRS0:OS5[2:0]
	16-bit input capture ch.4	FRSEL01_FRS1:IS0[2:0]
	16-bit input capture ch.5	FRSEL01_FRS1:IS1[2:0]
	16-bit input capture ch.6	FRSEL01_FRS1:IS2[2:0]
	16-bit input capture ch.7	FRSEL01_FRS1:IS3[2:0]
	4ch A/D activation compare 1, activation group 0	FRSEL01_FRS4:AS0[2:0]
	4ch A/D activation compare 1, activation group 1	FRSEL01_FRS4:AS1[2:0]



Table 3-3 Relationship between Resources and Registers of Free-run Timer Selection 2

Free-run Timer	Resource	Register
Free-run Timer ch.12 to ch.17	16-bit output compare ch.12	FRSEL02_FRS0:OS0[2:0]
	16-bit output compare ch.13	FRSEL02_FRS0:OS1[2:0]
	16-bit output compare ch.14	FRSEL02_FRS0:OS2[2:0]
	16-bit output compare ch.15	FRSEL02_FRS0:OS3[2:0]
	16-bit output compare ch.16	FRSEL02_FRS0:OS4[2:0]
	16-bit output compare ch.17	FRSEL02_FRS0:OS5[2:0]
	16-bit input capture ch.8	FRSEL02_FRS1:IS0[2:0]
	16-bit input capture ch.9	FRSEL02_FRS1:IS0[2:0]
	16-bit input capture ch.10	FRSEL02_FRS1:IS1[2:0]
	16-bit input capture ch.11	FRSEL02_FRS1:IS2[2:0]
	16-bit output compare ch.18	FRSEL02_FRS2:OS0[2:0]
	16-bit output compare ch.19	FRSEL02_FRS2:OS1[2:0]
	16-bit output compare ch.20	FRSEL02_FRS2:OS2[2:0]
	16-bit output compare ch.21	FRSEL02_FRS2:OS3[2:0]
	16-bit output compare ch.22	FRSEL02_FRS2:OS4[2:0]
	16-bit output compare ch.23	FRSEL02_FRS2:OS5[2:0]
	16-bit input capture ch.12	FRSEL02_FRS3:IS0[2:0]
	16-bit input capture ch.13	FRSEL02_FRS3:IS1[2:0]
	16-bit input capture ch.14	FRSEL02_FRS3:IS2[2:0]
	A/D activation compare 0	FRSEL02_FRS4:AS0[2:0]
	A/D activation compare 1	FRSEL02_FRS4:AS1[2:0]
	A/D activation compare 2	FRSEL02_FRS4:AS2[2:0]
	A/D activation compare 3	FRSEL02_FRS4:AS3[2:0]
	A/D activation compare 4	FRSEL02_FRS4:AS4[2:0]
	A/D activation compare 5	FRSEL02_FRS4:AS5[2:0]
	A/D activation compare 6	FRSEL02_FRS4:AS6[2:0]
	A/D activation compare 7	FRSEL02_FRS4:AS7[2:0]
	A/D activation compare 8	FRSEL02_FRS5:AS8[2:0]
	A/D activation compare 9	FRSEL02_FRS5:AS9[2:0]
	A/D activation compare 10	FRSEL02_FRS5:AS10[2:0]
	A/D activation compare 11	FRSEL02_FRS5:AS11[2:0]
	A/D activation compare 12	FRSEL02_FRS5:AS12[2:0]
	A/D activation compare 13	FRSEL02_FRS5:AS13[2:0]
	A/D activation compare 14	FRSEL02_FRS5:AS14[2:0]
	A/D activation compare 15	FRSEL02_FRS5:AS15[2:0]
	A/D activation compare 16	FRSEL02_FRS6:AS16[2:0]
	A/D activation compare 17	FRSEL02_FRS6:AS17[2:0]
	A/D activation compare 18	FRSEL02_FRS6:AS18[2:0]
	A/D activation compare 19	FRSEL02_FRS6:AS19[2:0]
	A/D activation compare 20	FRSEL02_FRS6:AS20[2:0]
	A/D activation compare 21	FRSEL02_FRS6:AS21[2:0]
	A/D activation compare 22	FRSEL02_FRS6:AS22[2:0]
	A/D activation compare 23	FRSEL02_FRS6:AS23[2:0]
	A/D activation compare 24	FRSEL02_FRS7:AS24[2:0]
	A/D activation compare 25	FRSEL02_FRS7:AS25[2:0]
	A/D activation compare 26	FRSEL02_FRS7:AS26[2:0]
	A/D activation compare 27	FRSEL02_FRS7:AS27[2:0]
	A/D activation compare 28	FRSEL02_FRS7:AS28[2:0]
	A/D activation compare 29	FRSEL02_FRS7:AS29[2:0]

Free-run Timer	Resource	Register
Free-run Timer ch.12 to ch.17	A/D activation compare 30	FRSEL02_FRS7:AS30[2:0]
	A/D activation compare 31	FRSEL02_FRS7:AS31[2:0]

**Table 3-4 Relationship between Registers Setting**

Setting Value	Free-run Timer Selection 0	Free-run Timer Selection 1	Free-run Timer Selection 2
000	Free-run timer ch.0 (initial state)	Free-run timer ch.6 (initial state)	Free-run timer ch.12 (initial state)
001	Free-run timer ch.1	Free-run timer ch.7	Free-run timer ch.13
010	Free-run timer ch.2	Free-run timer ch.8	Free-run timer ch.14
011	Free-run timer ch.3	Free-run timer ch.9	Free-run timer ch.15
100	Free-run timer ch.4	Free-run timer ch.10	Free-run timer ch.16
101	Free-run timer ch.5	Free-run timer ch.11	Free-run timer ch.17
110	Setting disabled (operation is not guaranteed)		
111			

**Note:**

- Before configuring these bits, make sure to verify that the free-run timer is inactive.

**(2) Operation of Free-run Timer Simultaneous Activation**

- This function can simultaneously activate/clear the specified 16-bit free-run timers.
- It simultaneously controls the timer enable bit (STOP) and timer clear bit (SCLR) of the timer state register (TCCS) for each of the 16-bit free-run timers for which free-run timer simultaneous activation is enabled.
- If simultaneous activation/clear is not performed, each of the 16-bit free-run timers can be activated/cleared individually by setting the timer enable bit (STOP) and timer clear bit (SCLR) of the timer state register (TCCS).
- 3 simultaneous activation functions for free-run timers are mounted. They can respectively control 16-bit free-run timer ch.0 to ch.5, ch.6 to ch.11, and ch.12 to ch.17.
- 16-bit free-run timer ch.6 to ch.11 can be used with a different activation source. This enables 16-bit free-run timer ch.6 to ch.17 to be activated/cleared simultaneously.





### (3) Operation of Indicating Free-run Timer Count Direction

- The current count information can be found by reading the free-run timer count direction indication registers.
- 3 free-run timer count direction indication registers are mounted. They respectively correspond to free-run timer ch.0 to ch.5, ch.6 to ch.11, and ch.12 to ch.17.

The relationship between free-run timer channels and the corresponding registers is as follows.

**Table 3-5 Relationship between Resources and Registers of Free-run Timer Count Direction Indication**

Free-run Timer	Registers
ch.0	FRCD00_FRTCDD:DOWN0
ch.1	FRCD00_FRTCDD:DOWN1
ch.2	FRCD00_FRTCDD:DOWN2
ch.3	FRCD00_FRTCDD:DOWN3
ch.4	FRCD00_FRTCDD:DOWN4
ch.5	FRCD00_FRTCDD:DOWN5
ch.6	FRCD01_FRTCDD:DOWN0
ch.7	FRCD01_FRTCDD:DOWN1
ch.8	FRCD01_FRTCDD:DOWN2
ch.9	FRCD01_FRTCDD:DOWN3
ch.10	FRCD01_FRTCDD:DOWN4
ch.11	FRCD01_FRTCDD:DOWN5
ch.12	FRCD02_FRTCDD:DOWN0
ch.13	FRCD02_FRTCDD:DOWN1
ch.14	FRCD02_FRTCDD:DOWN2
ch.15	FRCD02_FRTCDD:DOWN3
ch.16	FRCD02_FRTCDD:DOWN4
ch.17	FRCD02_FRTCDD:DOWN5

## 4. Setting Procedure Example

This section provides a setting procedure example of the free-run timer simultaneous activation.

### Setting Procedure Example in the Case of (FRSS01\_TCGSS:SEL=1)

The following provides a setting procedure example to set the FRSS01\_TCGSS:SEL bit to "1" and activate 16-bit free-run timers (ch.6 to ch.11 and ch.12 to ch.17) with different buses simultaneously.

Figure 4-1 Setting Procedure in the Case of TCGSS:SEL=1

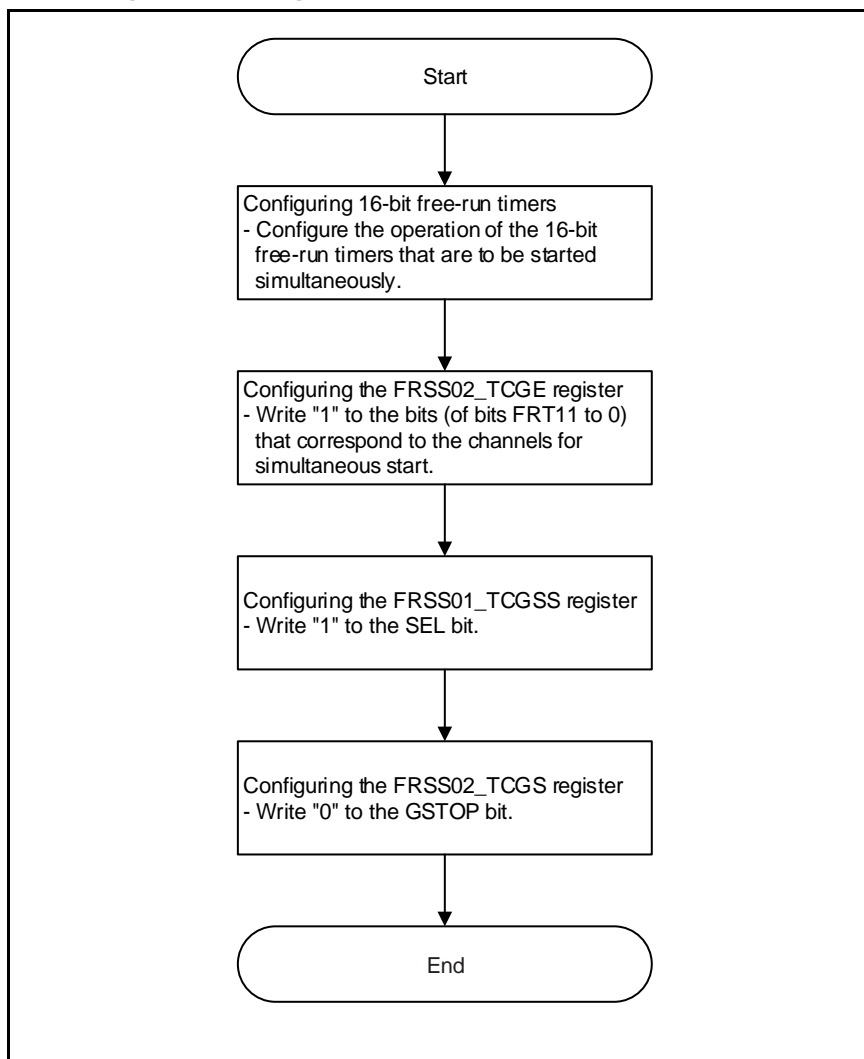
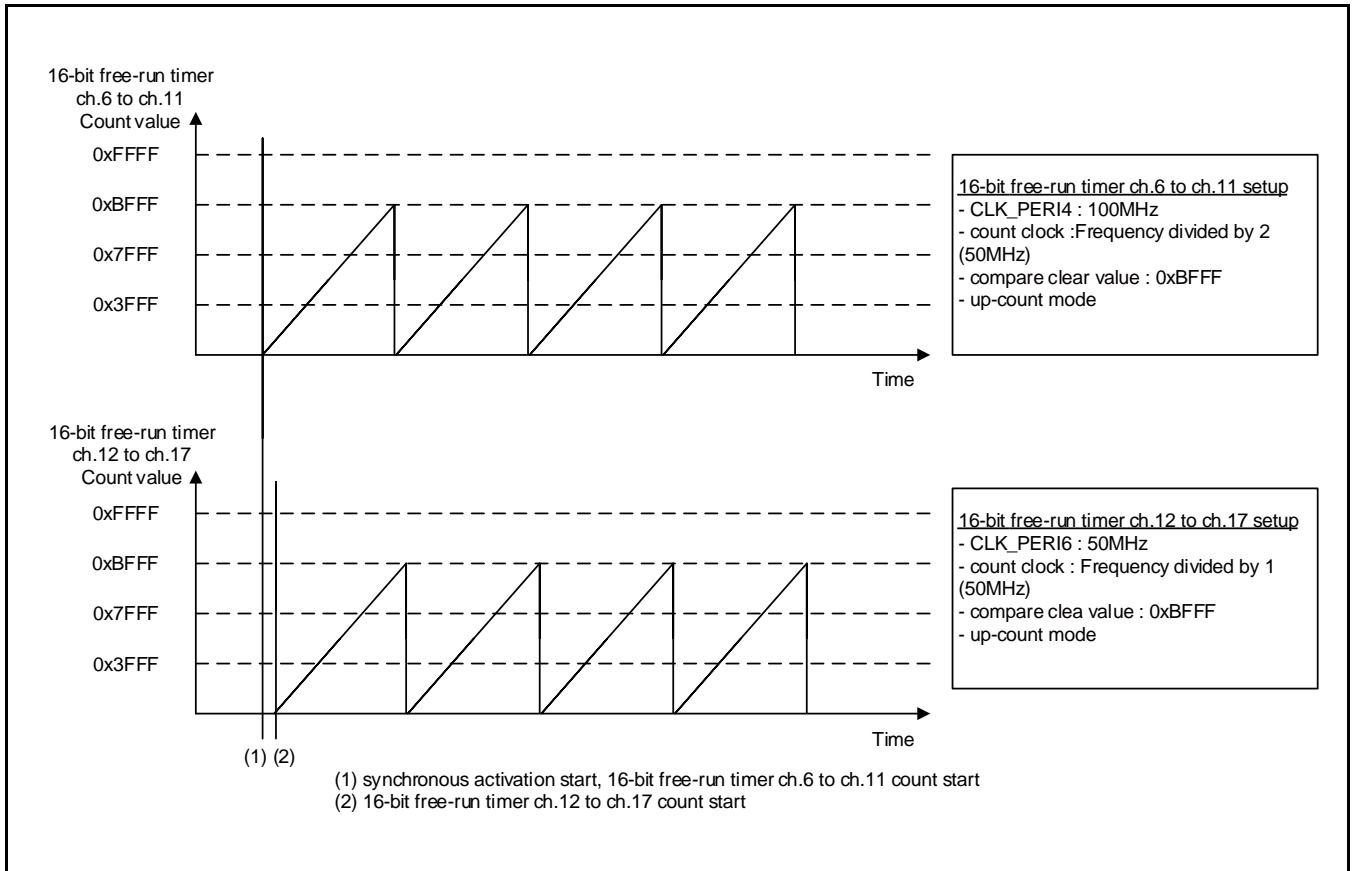




Figure 4-2 Example of Simultaneous Activation Timing



16-bit free-run timer ch.6 to ch.11 operate with CLK\_PERI4 and ch.12 to ch.17 operate with CLK\_PERI6. If CLK\_PERI4 and CLK\_PERI6 have frequencies different from each other, a gap (interval between (1) and (2) in Figure 4-2) will be present between the start of simultaneous activation and the start of count.

The gap before the start of count is determined by the division ratios of CLK\_PERI4 and CLK\_PERI6 and can be calculated using the following formula.

$$2 (\text{CLK\_PERI4 frequency} / \text{CLK\_PERI6 frequency} - 1) \times \text{CLK\_PERI4 cycle}$$

If CLK\_PERI4 and CLK\_PERI6 have the same frequency, the count starts simultaneously.

## 5. Registers

This section explains the registers for the selection and simultaneous activation of 16-bit free-run timers.

**Table 5-1 List of Free-run Timer Selection Registers**

Abbreviated Register Name	Register Name	See
FRSEL00_FRS0/ FRSEL01_FRS0/ FRSEL02_FRS0	Free-run Timer Selection Register 0	5.1.1
FRSEL00_FRS1/ FRSEL01_FRS1/ FRSEL02_FRS1	Free-run Timer Selection Register 1	5.1.1
FRSEL02_FRS2	Free-run Timer Selection Register 2	5.1.1
FRSEL02_FRS3	Free-run Timer Selection Register 3	5.1.1
FRSEL00_FRS4/ FRSEL01_FRS4/ FRSEL02_FRS4	Free-run Timer Selection Register 4	5.1.1
FRSEL02_FRS5	Free-run Timer Selection Register 5	5.1.1
FRSEL02_FRS6	Free-run Timer Selection Register 6	5.1.1
FRSEL02_FRS7	Free-run Timer Selection Register 7	5.1.1

**Table 5-2 List of Free-run Timer Simultaneous Activation Registers**

Abbreviated Register Name	Register Name	See
FRSS00_TCGS/ FRSS01_TCGS/ FRSS02_TCGS	Free-run Timer Selection Register 0	5.2.1
FRSS00_TCGSE/ FRSS01_TCGSE/ FRSS02_TCGSE	Timer Simultaneous Activation Enable Register	5.2.2
FRSS01_TCGSS	Timer Simultaneous Activation Source Selection Register	5.2.3

**Table 5-3 List of Free-run Timer Count Direction Indication Registers**

Abbreviated Register Name	Register Name	See
FRCD00_FRTCDD/ FRCD01_FRTCDD/ FRCD02_FRTCDD	Free-run Timer Count Direction Indication Register	5.3.1



## 5.1. Register for Free-run Timer Selection

The free-run timer selector has the free-run timer selection register.

### 5.1.1. Free-run Timer Selection Register (FRSELxx\_FRSn)

The free-run timer selection register (FRS) is used to set any of 6 channels in the free-run timers for the input capture, output compare, A/D activation compare respectively.

#### (1) Free-run Timer Selection Register 0 (FRSELxx\_FRS0) (xx=00, 01, 02)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R1,W1							
PROT_TYPE	-							
INITIAL_VALUE	11111111							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved	OS5			Reserved	OS4		
ACCESS_TYPE	R0,W0	R/W			R0,W0	R/W		
PROT_TYPE	-							
INITIAL_VALUE	0	000			0	000		

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved	OS3			Reserved	OS2		
ACCESS_TYPE	R0,W0	R/W			R0,W0	R/W		
PROT_TYPE	-							
INITIAL_VALUE	0	000			0	000		

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved	OS1			Reserved	OS0		
ACCESS_TYPE	R0,W0	R/W			R0,W0	R/W		
PROT_TYPE	-							
INITIAL_VALUE	0	000			0	000		

[bit31:24] Reserved: Reserved bits

[bit23, 19, 15, 11, 7, 3] Reserved: Reserved bits

[bit22:20] OS5[2:0]: 16-bit output compare ch.5/ch.11/ch.17 free-run timer selection bits

[bit18:16] OS4[2:0]: 16-bit output compare ch.4/ch.10/ch.16 free-run timer selection bits

[bit14:12] OS3[2:0]: 16-bit output compare ch.3/ch.9/ch.15 free-run timer selection bits

[bit10:8] OS2[2:0]: 16-bit output compare ch.2/ch.8/ch.14 free-run timer selection bits

[bit6:4] OS1[2:0]: 16-bit output compare ch.1/ch.7/ch.13 free-run timer selection bits

[bit2:0] OS0[2:0]: 16-bit output compare ch.0/ch.6/ch.12 free-run timer selection bits

These bits are used to configure the free-run timer assigned to the 16-bit output compare.

Value	Description
000	Free-run timer ch.0/ch.6/ch.12
001	Free-run timer ch.1/ch.7/ch.13
010	Free-run timer ch.2/ch.8/ch.14
011	Free-run timer ch.3/ch.9/ch.15
100	Free-run timer ch.4/ch.10/ch.16
101	Free-run timer ch.5/ch.11/ch.17
Other	Setting disabled (operation is not guaranteed)

**Note:**

- Before configuring these bits, make sure to verify that the free-run timer is inactive.



**(2) Free-run Timer Selection Register 2 (FRSEL02\_FRS2)**

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R1,W1							
PROT_TYPE	-							
INITIAL_VALUE	11111111							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved	OS11			Reserved	OS10		
ACCESS_TYPE	R0,W0	R/W			R0,W0	R/W		
PROT_TYPE	-							
INITIAL_VALUE	0	000			0	000		

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved	OS9			Reserved	OS8		
ACCESS_TYPE	R0,W0	R/W			R0,W0	R/W		
PROT_TYPE	-							
INITIAL_VALUE	0	000			0	000		

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved	OS7			Reserved	OS6		
ACCESS_TYPE	R0,W0	R/W			R0,W0	R/W		
PROT_TYPE	-							
INITIAL_VALUE	0	000			0	000		

**[bit31:24] Reserved: Reserved bits**

**[bit23, 19, 15, 11, 7, 3] Reserved: Reserved bits**

**[bit22:20] OS11[2:0]: 16-bit output compare ch.23 free-run timer selection bits**

**[bit18:16] OS10[2:0]: 16-bit output compare ch.22 free-run timer selection bits**

**[bit14:12] OS9[2:0]: 16-bit output compare ch.21 free-run timer selection bits**

**[bit10:8] OS8[2:0]: 16-bit output compare ch.20 free-run timer selection bits**

**[bit6:4] OS7[2:0]: 16-bit output compare ch.19 free-run timer selection bits**

**[bit2:0] OS6[2:0]: 16-bit output compare ch.18 free-run timer selection bits**

These bits are used to configure the free-run timer assigned to the 16-bit output compare.

Value	Description
000	Free-run timer ch.12
001	Free-run timer ch.13
010	Free-run timer ch.14
011	Free-run timer ch.15
100	Free-run timer ch.16
101	Free-run timer ch.17
Other	Setting disabled (operation is not guaranteed)

**Note:**

- Before configuring these bits, make sure to verify that the free-run timer is inactive.

**(3) Free-run Timer Selection Register 1 (FRSELxx\_FRS1) (xx=00, 01, 02)**

BIT_OFFSET	31-16
BIT_NAME	Reserved
ACCESS_TYPE	R1,W1
PROT_TYPE	-
INITIAL_VALUE	11111111_11111111

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved	IS3			Reserved	IS2		
ACCESS_TYPE	R0,W0	R/W			R0,W0	R/W		
PROT_TYPE	-							
INITIAL_VALUE	0	000			0	000		

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved	IS1			Reserved	IS0		
ACCESS_TYPE	R0,W0	R/W			R0,W0	R/W		
PROT_TYPE	-							
INITIAL_VALUE	0	000			0	000		

**[bit31:16] Reserved: Reserved bits**

**[bit15, 11, 7, 3] Reserved: Reserved bits**

**[bit14:12] IS3[2:0]: 16-bit input capture ch.3/ch.7/ch.11 free-run timer selection bits**

**[bit10:8] IS2[2:0]: 16-bit input capture ch.2/ch.6/ch.10 free-run timer selection bits**

**[bit6:4] IS1[2:0]: 16-bit input capture ch.1/ch.5/ch.9 free-run timer selection bits**

**[bit2:0] IS0[2:0]: 16-bit input capture ch.0/ch.4/ch.8 free-run timer selection bits**

These bits are used to configure the free-run timer assigned to the 16-bit input capture.

Value	Description
000	Free-run timer ch.0/ch.6/ch.12
001	Free-run timer ch.1/ch.7/ch.13
010	Free-run timer ch.2/ch.8/ch.14
011	Free-run timer ch.3/ch.9/ch.15
100	Free-run timer ch.4/ch.10/ch.16
101	Free-run timer ch.5/ch.11/ch.17
Other	Setting disabled (operation is not guaranteed)

**Note:**

- Before configuring these bits, make sure to verify that the free-run timer is inactive.



**(4) Free-run Timer Selection Register 3 (FRSEL02\_FRS3)**

BIT_OFFSET	31-16
BIT_NAME	Reserved
ACCESS_TYPE	R1,W1
PROT_TYPE	-
INITIAL_VALUE	11111111_11111111

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved					IS6		
ACCESS_TYPE	R0,W0					R/W		
PROT_TYPE	-							
INITIAL_VALUE	00000					000		

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved	IS5			Reserved	IS4		
ACCESS_TYPE	R0,W0	R/W			R0,W0	R/W		
PROT_TYPE	-							
INITIAL_VALUE	0	000			0	000		

**[bit31:16] Reserved: Reserved bits**

**[bit15:11, 7, 3] Reserved: Reserved bits**

**[bit10:8] IS6[2:0]: 16-bit input capture ch.14 free-run timer selection bits**

**[bit6:4] IS5[2:0]: 16-bit input capture ch.13 free-run timer selection bits**

**[bit2:0] IS4[2:0]: 16-bit input capture ch.12 free-run timer selection bits**

These bits are used to configure the free-run timer assigned to the 16-bit input capture.

Value	Description
000	Free-run timer ch.12
001	Free-run timer ch.13
010	Free-run timer ch.14
011	Free-run timer ch.15
100	Free-run timer ch.16
101	Free-run timer ch.17
Other	Setting disabled (operation is not guaranteed)

**Note:**

- Before configuring these bits, make sure to verify that the free-run timer is inactive.

**(5) Free-run Timer Selection Register 4 (FRSELxx\_FRS4) (xx=00, 01)**

BIT_OFFSET	31-8
BIT_NAME	Reserved
ACCESS_TYPE	R0,W0
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000_00000000

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved	AS1			Reserved	AS0		
ACCESS_TYPE	R0,W0	R/W			R0,W0	R/W		
PROT_TYPE	-							
INITIAL_VALUE	0	000			0	000		

**[bit31:7, 3] Reserved: Reserved bits**

**[bit6:4] AS1[2:0]: 4ch A/D activation compare activation group1 free-run timer selection bits**

**[bit2:0] AS0[2:0]: 4ch A/D activation compare activation group0 free-run timer selection bits**

These bits are used to configure the free-run timer assigned to the A/D activation compare.

Value	Description
000	Free-run timer ch.0/ch.6
001	Free-run timer ch.1/ch.7
010	Free-run timer ch.2/ch.8
011	Free-run timer ch.3/ch.9
100	Free-run timer ch.4/ch.10
101	Free-run timer ch.5/ch.11
Other	Setting disabled (operation is not guaranteed)

**Note:**

- Before configuring these bits, make sure to verify that the free-run timer is inactive.



**(6) Free-run Timer Selection Register 4 (FRSEL02\_FRS4)**

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved	AS7			Reserved	AS6		
ACCESS_TYPE	R0,W0	R/W			R0,W0	R/W		
PROT_TYPE	-							
INITIAL_VALUE	0	000			0	000		

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved	AS5			Reserved	AS4		
ACCESS_TYPE	R0,W0	R/W			R0,W0	R/W		
PROT_TYPE	-							
INITIAL_VALUE	0	000			0	000		

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved	AS3			Reserved	AS2		
ACCESS_TYPE	R0,W0	R/W			R0,W0	R/W		
PROT_TYPE	-							
INITIAL_VALUE	0	000			0	000		

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved	AS1			Reserved	AS0		
ACCESS_TYPE	R0,W0	R/W			R0,W0	R/W		
PROT_TYPE	-							
INITIAL_VALUE	0	000			0	000		

**[bit31, 27, 23, 19, 15, 11, 7, 3] Reserved: Reserved bits**

**[bit30:28] AS7[2:0]: A/D activation compare 7 free-run timer selection bits**

**[bit26:24] AS6[2:0]: A/D activation compare 6 free-run timer selection bits**

**[bit22:20] AS5[2:0]: A/D activation compare 5 free-run timer selection bits**

**[bit18:16] AS4[2:0]: A/D activation compare 4 free-run timer selection bits**

**[bit14:12] AS3[2:0]: A/D activation compare 3 free-run timer selection bits**

**[bit10:8] AS2[2:0]: A/D activation compare 2 free-run timer selection bits**

**[bit6:4] AS1[2:0]: A/D activation compare 1 free-run timer selection bits**

**[bit2:0] AS0[2:0]: A/D activation compare 0 free-run timer selection bits**

These bits are used to configure the free-run timer assigned to the A/D activation compare.

Value	Description
000	Free-run timer ch.12
001	Free-run timer ch.13
010	Free-run timer ch.14
011	Free-run timer ch.15
100	Free-run timer ch.16
101	Free-run timer ch.17
Other	Setting disabled (operation is not guaranteed)

**Note:**

- Before configuring these bits, make sure to verify that the free-run timer is inactive.

**(7) Free-run Timer Selection Register 5 (FRSEL02\_FRS5)**

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved	AS15			Reserved	AS14		
ACCESS_TYPE	R0,W0	R/W			R0,W0	R/W		
PROT_TYPE	-							
INITIAL_VALUE	0	000			0	000		

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved	AS13			Reserved	AS12		
ACCESS_TYPE	R0,W0	R/W			R0,W0	R/W		
PROT_TYPE	-							
INITIAL_VALUE	0	000			0	000		

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved	AS11			Reserved	AS10		
ACCESS_TYPE	R0,W0	R/W			R0,W0	R/W		
PROT_TYPE	-							
INITIAL_VALUE	0	000			0	000		

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved	AS9			Reserved	AS8		
ACCESS_TYPE	R0,W0	R/W			R0,W0	R/W		
PROT_TYPE	-							
INITIAL_VALUE	0	000			0	000		

**[bit31, 27, 23, 19, 15, 11, 7, 3] Reserved: Reserved bits**

**[bit30:28] AS15[2:0]: A/D activation compare 15 free-run timer selection bits**

**[bit26:24] AS14[2:0]: A/D activation compare 14 free-run timer selection bits**

**[bit22:20] AS13[2:0]: A/D activation compare 13 free-run timer selection bits**

**[bit18:16] AS12[2:0]: A/D activation compare 12 free-run timer selection bits**

**[bit14:12] AS11[2:0]: A/D activation compare 11 free-run timer selection bits**

**[bit10:8] AS10[2:0]: A/D activation compare 10 free-run timer selection bits**

**[bit6:4] AS9[2:0]: A/D activation compare 9 free-run timer selection bits**

**[bit2:0] AS8[2:0]: A/D activation compare 8 free-run timer selection bits**

These bits are used to configure the free-run timer assigned to the A/D activation compare.

Value	Description
000	Free-run timer ch.12
001	Free-run timer ch.13
010	Free-run timer ch.14
011	Free-run timer ch.15
100	Free-run timer ch.16
101	Free-run timer ch.17
Other	Setting disabled (operation is not guaranteed)

**Note:**

- Before configuring these bits, make sure to verify that the free-run timer is inactive.



**(8) Free-run Timer Selection Register 6 (FRSEL02\_FRS6)**

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved	AS23			Reserved	AS22		
ACCESS_TYPE	R0,W0	R/W			R0,W0	R/W		
PROT_TYPE	-							
INITIAL_VALUE	0	000			0	000		

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved	AS21			Reserved	AS20		
ACCESS_TYPE	R0,W0	R/W			R0,W0	R/W		
PROT_TYPE	-							
INITIAL_VALUE	0	000			0	000		

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved	AS19			Reserved	AS18		
ACCESS_TYPE	R0,W0	R/W			R0,W0	R/W		
PROT_TYPE	-							
INITIAL_VALUE	0	000			0	000		

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved	AS17			Reserved	AS16		
ACCESS_TYPE	R0,W0	R/W			R0,W0	R/W		
PROT_TYPE	-							
INITIAL_VALUE	0	000			0	000		

**[bit31, 27, 23, 19, 15, 11, 7, 3] Reserved: Reserved bits**

**[bit30:28] AS23[2:0]: A/D activation compare 23 free-run timer selection bits**

**[bit26:24] AS22[2:0]: A/D activation compare 22 free-run timer selection bits**

**[bit22:20] AS21[2:0]: A/D activation compare 21 free-run timer selection bits**

**[bit18:16] AS20[2:0]: A/D activation compare 20 free-run timer selection bits**

**[bit14:12] AS19[2:0]: A/D activation compare 19 free-run timer selection bits**

**[bit10:8] AS18[2:0]: A/D activation compare 18 free-run timer selection bits**

**[bit6:4] AS17[2:0]: A/D activation compare 17 free-run timer selection bits**

**[bit2:0] AS16[2:0]: A/D activation compare 16 free-run timer selection bits**

These bits are used to configure the free-run timer assigned to the A/D activation compare.

Value	Description
000	Free-run timer ch.12
001	Free-run timer ch.13
010	Free-run timer ch.14
011	Free-run timer ch.15
100	Free-run timer ch.16
101	Free-run timer ch.17
Other	Setting disabled (operation is not guaranteed)

**Note:**

- Before configuring these bits, make sure to verify that the free-run timer is inactive.

**(9) Free-run Timer Selection Register 7 (FRSEL02\_FRS7)**

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved	AS31			Reserved	AS30		
ACCESS_TYPE	R0,W0	R/W			R0,W0	R/W		
PROT_TYPE	-							
INITIAL_VALUE	0	000			0	000		

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved	AS29			Reserved	AS28		
ACCESS_TYPE	R0,W0	R/W			R0,W0	R/W		
PROT_TYPE	-							
INITIAL_VALUE	0	000			0	000		

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved	AS27			Reserved	AS26		
ACCESS_TYPE	R0,W0	R/W			R0,W0	R/W		
PROT_TYPE	-							
INITIAL_VALUE	0	000			0	000		

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved	AS25			Reserved	AS24		
ACCESS_TYPE	R0,W0	R/W			R0,W0	R/W		
PROT_TYPE	-							
INITIAL_VALUE	0	000			0	000		

**[bit31, 27, 23, 19, 15, 11, 7, 3] Reserved: Reserved bits**

**[bit30:28] AS31[2:0]: A/D activation compare 31 free-run timer selection bits**  
**[bit26:24] AS30[2:0]: A/D activation compare 30 free-run timer selection bits**  
**[bit22:20] AS29[2:0]: A/D activation compare 29 free-run timer selection bits**  
**[bit18:16] AS28[2:0]: A/D activation compare 28 free-run timer selection bits**  
**[bit14:12] AS27[2:0]: A/D activation compare 27 free-run timer selection bits**  
**[bit10:8] AS26[2:0]: A/D activation compare 26 free-run timer selection bits**  
**[bit6:4] AS25[2:0]: A/D activation compare 25 free-run timer selection bits**  
**[bit2:0] AS24[2:0]: A/D activation compare 24 free-run timer selection bits**

These bits are used to configure the free-run timer assigned to the A/D activation compare.

Value	Description
000	Free-run timer ch.12
001	Free-run timer ch.13
010	Free-run timer ch.14
011	Free-run timer ch.15
100	Free-run timer ch.16
101	Free-run timer ch.17
Other	Setting disabled (operation is not guaranteed)

**Note:**

- Before configuring these bits, make sure to verify that the free-run timer is inactive.



## 5.2. Register for Timer Simultaneous Activation

The free-run simultaneous activation consists of the timer synchronous activation register and the timer synchronous enable register.

### 5.2.1. (FRSSxx\_TCGS) (xx=00, 01, 02)

The timer synchronous activation register (TCGS) is used for enabling simultaneous timer and controlling simultaneous timer clear of the free-run timer. The free-run timer for enabling and clearing the simultaneous timer can be set by the timer synchronous activation enable register (TCGSE).

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved						GSTOP	GSCLR
ACCESS_TYPE	R0,W0						R0,W	R0,W
PROT_TYPE	-							
INITIAL_VALUE	000000						0	0

[bit31:26] Reserved: Reserved bits

[bit25] GSTOP: Simultaneous timer enable bit

Value	Description	
	Read	Write
0	"0" is always read out.	Enable the counting simultaneously. (Start the counting)
1		Disable the counting simultaneously. (Stop the counting)

- This bit is used to simultaneously start/stop the counting of the free-run timer specified by the timer synchronous activation enable register (TCGSE).
- When this bit is set to "0":
- Start the counting of the 16-bit free-run timer of the free-run timer specified by the timer synchronous activation enable register (TCGSE). At this time, the STOP bit of the timer state control register(TCCS) of the free-run timer specified by the timer synchronous activation enable register (TCGSE) will be cleared to "0".
- When this bit is set to "1":
- Stop the counting of the 16-bit free-run timer of the free-run timer specified by the timer synchronous activation enable register (TCGSE). At this time, the STOP bit of the timer state control register (TCCS) of the free-run timer specified by the timer synchronous activation enable register (TCGSE) will be set to "1".
- The value read out is always "0".

[bit24] GSCLR: Simultaneous timer clear bit

Value	Description	
	Read	Write
0	"0" is always read.	Counter will not be initialized
1		Counter will be initialized to "0x0000" simultaneously.

- This bit is used to initialize the free-run timer 16-bit free-run timer specified by the timer synchronous activation enable register (TCGSE) to "0x0000".
- When this bit is set to "1":

- Initialize the 16-bit free-run timer of the free-run timer specified by the timer synchronous activation enable register (TCGSE). At this time, the SCLR bit of the timer state control register (TCCS) of the free-run timer specified by the timer synchronous activation enable register (TCGSE) will be set to "1".
- When this bit is set to "0":
- Cancel the instruction of initializing the 16-bit free-run timer of the free-run timer specified by the timer synchronous activation enable register (TCGSE). At this time, the SCLR bit of the timer state control register (TCCS) of the free-run timer specified by the timer synchronous activation enable register (TCGSE) will be cleared to "0".
- The value read out is always "0".





### 5.2.2. Timer Simultaneous Activation Enable Register (FRSSxx\_TCGSE)

The timer simultaneous activation enable register (FRSSxx\_TCGSE) sets free-run timers for which simultaneous activation/clear is enabled.

#### (1) Timer Simultaneous Activation Enable Register 0 (FRSS00\_TCGSE) / Timer Simultaneous Activation Enable Register 1 (FRSS01\_TCGSE)

BITS	15	14	13	12	11	10	9	8
BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved							
ACCESS_TYPE	R0,W0							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

BITS	7	6	5	4	3	2	1	0
BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved		FRT5	FRT4	FRT3	FRT2	FRT1	FRT0
ACCESS_TYPE			R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	00		0	0	0	0	0	0

[bit15:6] Reserved: Reserved bits

#### [bit5:0] FRT5 to FRT0: Simultaneous activation/clear setting bits

Value	Description
0	Do not perform simultaneous activation/clear (free-run timer simultaneous activation 2).
1	Perform simultaneous activation/clear (free-run timer simultaneous activation 2).

- These bits set free-run timers for which simultaneous activation/clear is enabled.
- When "0" is set in these bits:  
The free-run timers are not activated or cleared when the timer simultaneous activation register (TCGS) is set.
- When "1" is set in these bits:  
The free-run timers are activated or cleared when the timer simultaneous activation register (TCGS) is set.

## (2) Timer Simultaneous Activation Enable Register 2 (FRSS02\_TCGSE)

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved				FRT11	FRT10	FRT9	FRT8
ACCESS_TYPE	R0,W0				R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0000				0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	FRT7	FRT6	FRT5	FRT4	FRT3	FRT2	FRT1	FRT0
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

**[bit15:12] Reserved: Reserved bits**

### [bit11:6] FRT11 to FRT 6: Simultaneous activation/clear setting bits

Value	Description
0	Do not perform simultaneous activation/clear (free-run timer simultaneous activation 1).
1	Perform simultaneous activation/clear (free-run timer simultaneous activation 1).

- These bits are used when controlling free-run timer simultaneous activation 1 from free-run timer simultaneous activation 2.
- To control free-run timer simultaneous activation 1 using these bits, set the SEL bit in the timer simultaneous activation selection register (TCGSS) to "1".

### [bit5:0] FRT5 to FRT0: Simultaneous activation/clear setting bits

Value	Description
0	Do not perform simultaneous activation/clear (free-run timer simultaneous activation 2).
1	Perform simultaneous activation/clear (free-run timer simultaneous activation 2).

- These bits set free-run timers for which simultaneous activation/clear is enabled.
- When "0" is set in these bits:  
The free-run timers are not activated or cleared when the timer simultaneous activation register (TCGS) is set.
- When "1" is set in these bits:  
The free-run timers are activated or cleared when the timer simultaneous activation register (TCGS) is set.



### 5.2.3. Timer Simultaneous Activation Source Selection Register (FRSS01\_TCGSS)

The timer simultaneous activation source selection register (FRSS01\_TCGSS) is used to enable/clear timers by interlocking them with simultaneous activation of another bus. This register has only free-run timer simultaneous activation 1.

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved							SEL
ACCESS_TYPE	R0,W0							R/W
PROT_TYPE	-							
INITIAL_VALUE	0000000							0

[bit7:1] Reserved: Reserved bits

[bit0] SEL: Simultaneous activation source selection

Value	Description
0	The setting of free-run timer simultaneous activation 1 is valid.
1	The setting of free-run timer simultaneous activation 2 is valid.

This bit selects the simultaneous activation source for free-run timer ch.6 to ch.11.

- When "0" is set in this bit:

The timer simultaneous activation enable register (FRSS01\_TCGSE:FRT5 to FRT0) of free-run timer simultaneous activation 1 is enabled as the simultaneous activation source. The timer simultaneous activation enable register (FRSS01\_TCGSE:FRT11 to FRT6) of free-run timer simultaneous activation 2 is disabled.

- When "1" is set in this bit:

The timer simultaneous activation enable register (FRSS01\_TCGSE:FRT11 to FRT6) of free-run timer simultaneous activation 2 is enabled as the simultaneous activation source. The timer simultaneous activation enable register (FRSS01\_TCGSE:FRT5 to FRT0) of free-run timer simultaneous activation 1 is disabled.

### 5.3. Register for Indicating Free-run Timer Count Direction

The register for indicating the free-run timer count direction is the free-run timer count direction indication register.

#### 5.3.1. Free-run Timer Count Direction Indication Register (FRCDxx\_FRTCDD) (xx = 00, 01, or 02)

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved		DOWN5	DOWN4	DOWN3	DOWN2	DOWN1	DOWN0
ACCESS_TYPE	R0,W0		R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	00		0	0	0	0	0	0

**[bit7:6] Reserved: Reserved bits**

**[bit5:0] DOWN5 to DOWN0: Count direction indication bits**

Value	Description
0	Up count (initial value)
1	Down count

- These bits indicate the current count directions of 16-bit free-run timers.
- This is disabled while count is stopped. (This is because the direction depends on the state before the count is stopped.)





# CHAPTER 48: 16-bit Input Capture

This chapter explains 16-bit input capture.

---

1. Overview
2. Configuration
3. Operation
4. Registers
5. Precautions for Using



## 1. Overview

The 16-bit input capture consists of input capture registers and input capture state control registers.

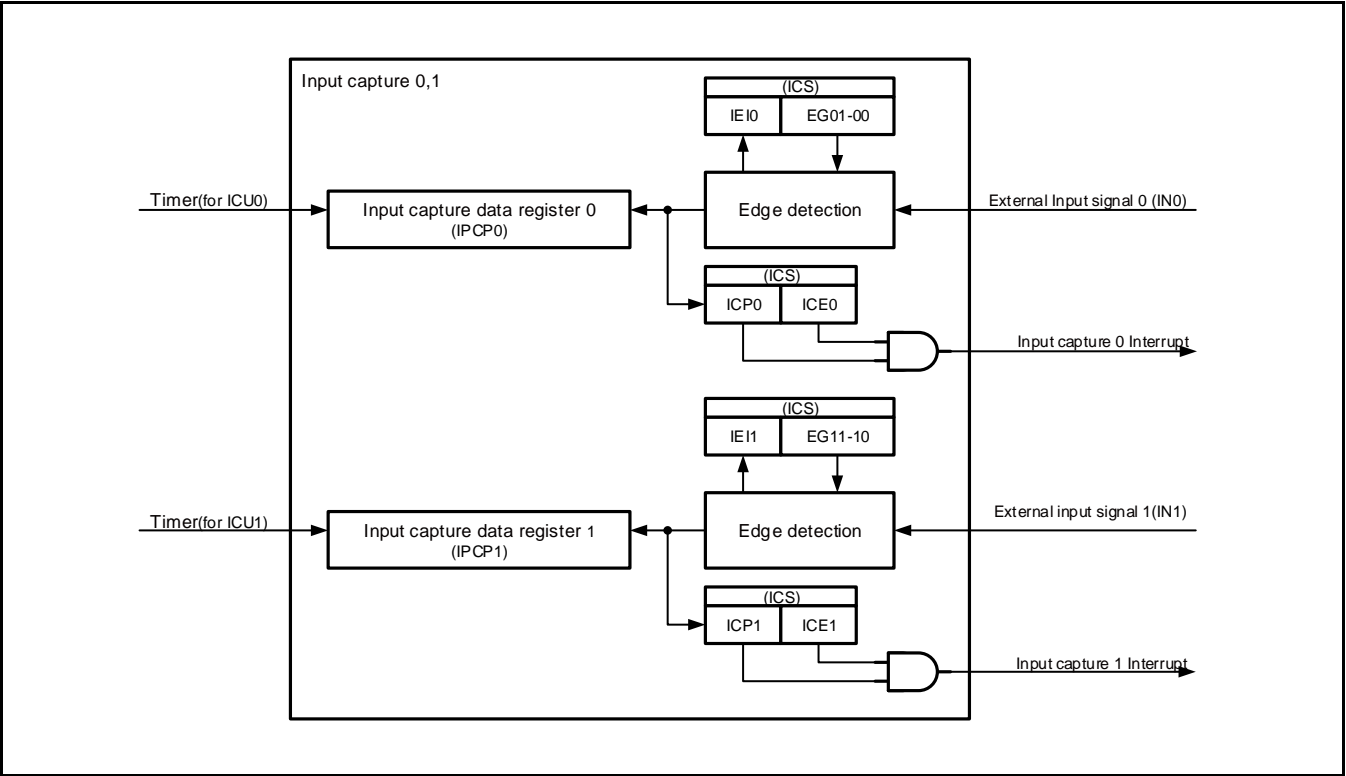
### Functions of the 16-bit Input Capture

- Input capture is composed of 2 independent external input pins, capture registers that correspond to the pins, and the capture control register. When an edge signal is detected on an external input pin, the value of the 16-bit free-run timer can be stored in the capture register. At the same time, an interrupt is generated.
- 3 types of trigger edges (rising edge, falling edge, and both edges) of an external input signal can be selected. There is a register that indicates whether a trigger edge is a rising or falling edge.
- An interrupt is generated when a valid edge is detected from an external input signal.
- Any desired free-run timer channel can be set for each compare unit.
- There are input capture channels for which any of free-run timers can be selected as the input. See Section "2. Configuration" in "CHAPTER: Function of Free-run Timer Selector and Simultaneous Activation" for details.

2. Configuration

This section shows a block diagram of 16-bit input capture.  
Figure 2-1 is a configuration diagram of the 16-bit input capture.

Figure 2-1 Block Diagram of 16-bit Input Capture (ch.0, ch.1)







### 3. Operation

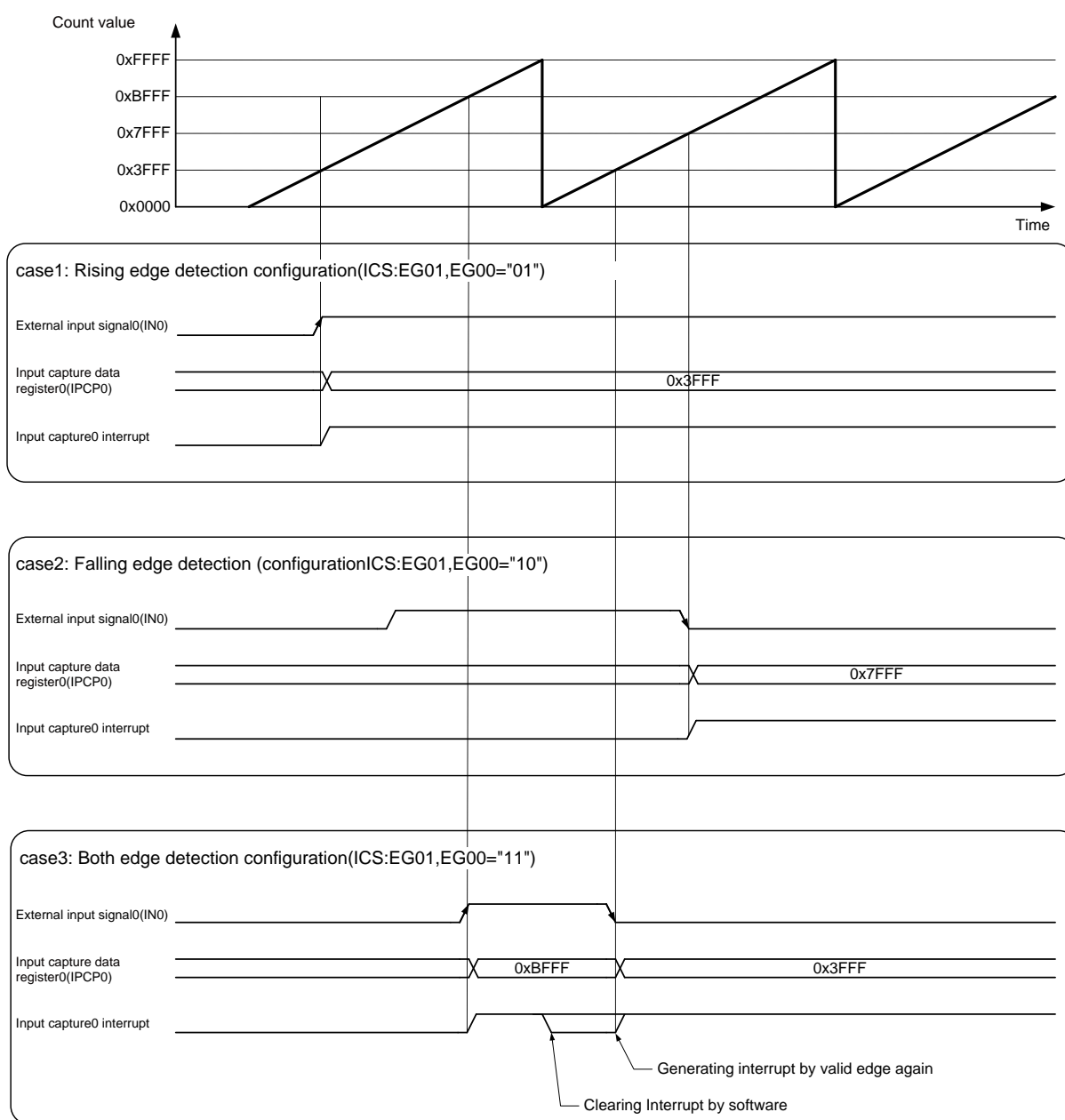
This section describes the operation of 16-bit input capture.

#### (1) Operation of the 16-bit Input Capture

The 16-bit input capture is used to detect a specified valid edge. When a valid edge is detected, an interrupt flag is set. Then the value of the 16-bit free-run timer is loaded to the input capture data register.

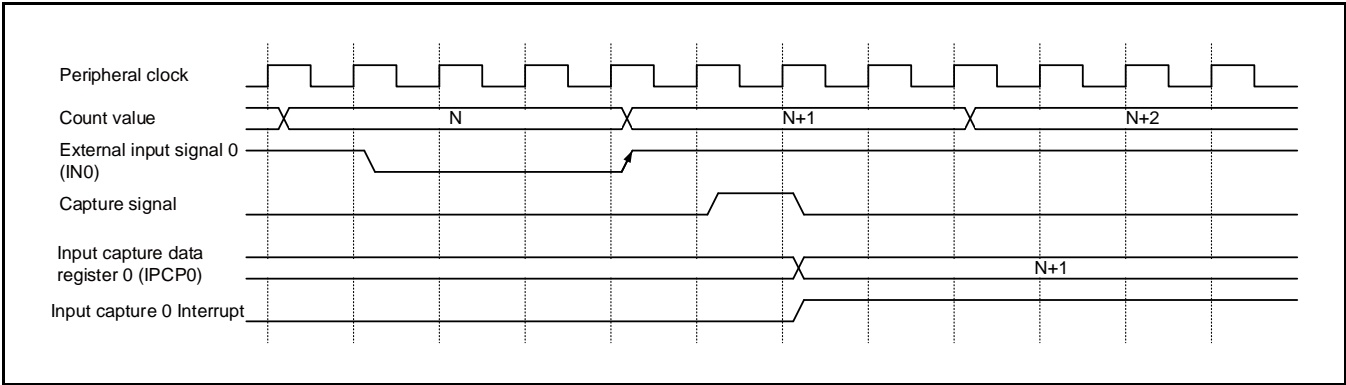
#### (2) Input Capture Operation

Figure 3-1 Example of Input Capture Timing



(3) 16-bit Input Capture Input Timing

Figure 3-2 Example of 16-bit Input Capture Timing for Input Signals





### 3.1. Interrupt of 16-bit Input Capture

As the interrupt of the 32-bit input capture, there is an input capture interrupt triggered by an external input signal. An interrupt request is generated when a valid edge is detected.

#### 16-bit Input Capture Interrupt

Table 3-1 shows the interrupt control bits and interrupt factor of the 16-bit input capture.

**Table 3-1 Interrupt Control Bits and Interrupt Factor of 16-bit Input Capture 1, 0**

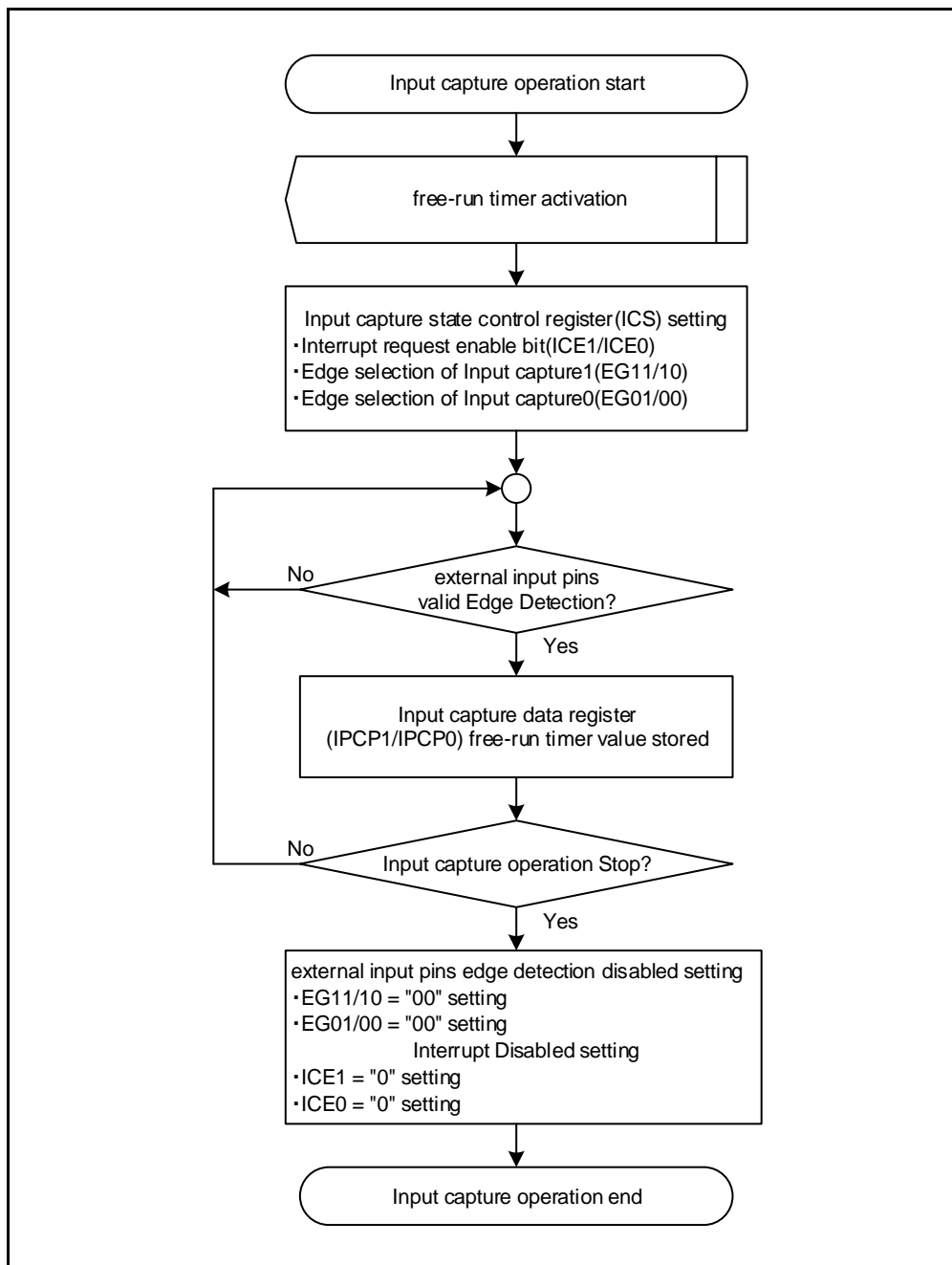
	<b>16-bit Input Capture 1, 0</b>
Interrupt Request Flag Bit	The interrupt request flag bits (ICP1, ICP0) in the input capture state control register (ICS)
Interrupt Request Enable Bit	The interrupt request enable bits (ICE1, ICE0) in the input capture state control register (ICS)
Interrupt Factor	A valid edge is detected on the external input pin (IN1, IN0).

For the 16-bit interrupt capture, when a valid edge is detected on the external input pin (IN1, IN0), the interrupt request flag (ICP1, ICP0) in the input capture state control register (ICS) is set to "1". When an interrupt request is set to enabled (ICE1, ICE0 = "0b11" in the input capture state control register (ICS)) in this state, then the interrupt request is output to the interrupt controller.

## 3.2. Setting Procedure Example

This section explains a setting procedure example of 16-bit input capture.

Figure 3-1 Setting Procedure Example of 16-bit Input Capture





## 4. Registers

This section describes the registers of 16-bit input capture.

A prefix (ICU16Bxx) is added to each of the 16-bit input capture registers. xx is the channel number.

**Table 4-1 List of 16-bit Input Capture Registers**

Abbreviated Register Name	Register Name	See
ICU16Bxx_IPCP0, ICU16Bxx_IPCP1	Input Capture Data Register 0, 1	4.1
ICU16Bxx_ICS	Input Capture State Control Register	4.2
ICU16Bxx_ICSC	Input Capture State Clear Register	4.3
ICU16Bxx_ICSS	Input Capture State Set Register	4.4

xx: channel number (xx=00, 02, 04, 06, 08, 10, 12, 14)

## 4.1. Input Capture Data Register 0, 1 (IPCP0, ICPC1)

Input capture data registers 0, 1 (IPCP0, ICPC1) are used to store the count value of the free-run timer when a valid edge of an external input signal is detected.

### (1) Input Capture Data Register 0 (IPCP0)

BIT_OFFSET	31-16
BIT_NAME	CP
ACCESS_TYPE	R,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000

#### [bit31:16] CP[15:0]: Input capture data value bits

- The input capture data register 0 (IPCP0) is used to store the value of the free-run timer when a valid edge of the external input pin IN0 signal is detected.
- The free-run timer that is mentioned here indicates the operating state of the 16-bit free-run timer that is connected to the input capture.

#### Notes:

- *For access to this register, use half-word or word access instructions.*
- *No data can be written to this register.*



## (2) Input Capture Data Register 1 (IPCP1)

BIT_OFFSET	15-0
BIT_NAME	CP
ACCESS_TYPE	R,WX
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000

### [bit15:0] CP[15:0]: Input capture data value bits

- The input capture data register 1 (IPCP1) is used to store the value of the free-run timer when a valid edge of the external input pin IN1 signal is detected.
- The free-run timer that is mentioned here indicates the operating state of the 16-bit free-run timer that is connected to the input capture.

### Notes:

- *For access to this register, use half-word or word access instructions.*
- *No data can be written to this register.*

## 4.2. Input Capture State Control Register (ICS)

The input capture state control register (ICS) is used to select an edge, enable an interrupt request, and control an interrupt request flag. This register is also used to indicate a valid edge detected in the input captures 0, 1. For details on writing to this register, see Section "5. Precautions for Using".

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved						IEI1	IEI0
ACCESS_TYPE	R0,W0						R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	000000						0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	ICP1	ICP0	ICE1	ICE0	EG1		EG0	
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W		R/W	
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	00		00	

### [bit31:26] Reserved: Reserved bits

#### [bit25] IEI1: Valid edge indication bit (input capture 1)

- This bit is a valid edge indication bit in input capture data register 1. This bit indicates that a rising or falling edge is detected.
- When a falling edge is detected, this bit is set to "0".
- When a rising edge is detected, this bit is set to "1".
- This bit is read-only.

Value	Description
0	A falling edge is detected.
1	A rising edge is detected.

#### Notes:

- The value read is meaningless if the edge selection bit (EG1[1:0]) is set to "0b00".
- If the edge selection bit (EG1[1:0]) is set to any value other than "0b00", then the value is updated when the interrupt request flag (ICP1) is set.

#### [bit24] IEI0: Valid edge indication bit (input capture 0)

- This bit is a valid edge indication bit in input capture data register 0. This bit indicates that a rising or falling edge is detected.
- When a falling edge is detected, this bit is set to "0".
- When a rising edge is detected, this bit is set to "1".
- This bit is a read-only bit.

Value	Description
0	A falling edge is detected.
1	A rising edge is detected.



**Notes:**

- The value read is meaningless if the edge selection bit (EG0[1:0]) is set to "0b00".
- If the edge selection bit (EG0[1:0]) is set to any value other than "0b00", then the value is updated when the interrupt request flag (ICP0) is set.

**[bit23] ICP1: Interrupt request flag bit (input capture 1)**

- This bit is used as an interrupt request flag of input capture 1.
- This bit is set to "1" immediately upon detecting a valid edge on the external input pin (IN1).
- An interrupt is generated as soon as this bit is set to "1" when the interrupt request enable bit (ICE1) is "1".
- When this bit is set to "0", this bit is cleared to "0".
- When this bit is set to "1", this bit is not affected.
- This bit is cleared to "0" by writing "1" to the ICP1C bit in the ICSC register.

Value	Description	
	Read	Write
0	No valid edge is detected.	This bit is cleared.
1	A valid edge is detected.	This bit is not affected.

**Note:**

- If a hardware set occur and a software clear (write "0") at the same time, the hardware set takes precedence.

**[bit22] ICP0: Interrupt request flag bit (input capture 0)**

- This bit is used as an interrupt request flag of input capture 0.
- This bit is set to "1" immediately upon detecting a valid edge on the external input pin (IN0).
- An interrupt is generated as soon as this bit is set to "1" when the interrupt request enable bit (ICE0) is "1".
- When this bit is set to "0", this bit is cleared to "0".
- When this bit is set to "1", this bit is not affected.
- This bit is cleared to "0" by writing "1" to the ICP0C bit in the ICSC register.

Value	Description	
	Read	Write
0	No valid edge is detected.	This bit is cleared.
1	A valid edge is detected.	This bit is not affected.

**Note:**

- If a hardware set occur and a software clear (write "0") at the same time, the hardware set takes precedence.

**[bit21] ICE1: Interrupt request enable bit (input capture 1)**

- This bit is used to enable an interrupt request of input capture1.
- An interrupt of input capture 1 is generated when the interrupt request flag bit (ICP1) is set while this bit is "1".
- This bit is cleared to "0" by writing "1" to the ICE1C bit in the ICSC register.
- This bit is set to "1" by writing "1" to the ICE1S bit in the ICSS register.

Value	Description
0	Disable interrupt request.
1	Enable interrupt request.

**[bit20] ICE0: Interrupt request enable bit (input capture 0)**

- This bit is used to enable an interrupt request of input capture 0.
- An interrupt of input capture 0 is generated when the interrupt request flag bit (ICP0) is set while this bit is "1".
- This bit is cleared to "0" by writing "1" to the ICE0C bit in the ICSC register.
- This bit is set to "1" by writing "1" to the ICE0S bit in the ICSS register.

Value	Description
0	Disable interrupt request.
1	Enable interrupt request.

**[bit19:18] EG1[1:0]: Edge selection bits (input capture 1)**

These bits are used to enable the operation of input capture 1. These bits specify a valid edge of the external input (IN1).

Value	Description
00	No edge is detected (stop).
01	A rising edge is detected.
10	A falling edge is detected.
11	Both edges are detected.

**[bit17:16] EG0[1:0]: Edge selection bits (input capture 0)**

These bits are used to enable the operation of input capture 0. These bits specify a valid edge of the external input (IN0).

Value	Description
00	No edge is detected (stop).
01	A rising edge is detected.
10	A falling edge is detected.
11	Both edges are detected.

**Note:**

- For access to this register, use half-word or word access instructions.



### 4.3. Input Capture State Clear Register (ICSC)

The input capture state clear register (ICSC) is a register to clear a bit in the input capture state control register (ICS).

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,W0							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	ICPC1	ICPC0	ICEC1	ICEC0	Reserved			
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W	R0,W0			
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0000			

**[bit31:24] Reserved: Reserved bits**

#### **[bit23] ICP1C: Interrupt request flag clear bit**

- This bit is always "0" when read.
- Writing "1" to this bit clears the ICP1 bit in the ICS register.

Value	Description
0	No effect
1	Clear the interrupt request flag bit

#### **[bit22] ICP0C: Interrupt request flag clear bit**

- This bit is always "0" when read.
- Writing "1" to this bit clears the ICP0 bit in the ICS register.

Value	Description
0	No effect
1	Clear the interrupt request flag bit

#### **[bit21] ICE1C: Interrupt request enable clear bit**

- This bit is always "0" when read.
- Writing "1" to this bit clears the ICE1 bit in the ICS register.

Value	Description
0	No effect
1	Clear the interrupt request enable bit

**[bit20] ICE0C: Interrupt request enable clear bit**

- This bit is always "0" when read.
- Writing "1" to this bit clears the ICE0 bit in the ICS register.

Value	Description
0	No effect
1	Clear the interrupt request enable bit

**[bit19:16] Reserved: Reserved bits**

**Note:**

- *For access to this register, use half-word or word access instructions.*



## 4.4. Input Capture State Set Register (ICSS)

The input capture state set register (ICSS) is a register that sets the bit in the input capture state control register (ICS).

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,W0							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved		ICES1	ICES0	Reserved			
ACCESS_TYPE	R0,W0		R0,W	R0,W	R0,W0			
PROT_TYPE	-							
INITIAL_VALUE	00		0	0	0000			

**[bit31:22] Reserved: Reserved bits**

**[bit21] ICES1: Interrupt request enable set bit**

- This bit is always "0" when read.
- Writing "1" to this bit sets the ICE1 bit in the ICS register to "1".

Value	Description
0	No effect
1	Set the interrupt request enable bit

**[bit20] ICES0: Interrupt request enable set bit**

- This bit is always "0" when read.
- Writing "1" to this bit sets the ICE0 bit in the ICS register to "1".

Value	Description
0	No effect
1	Set the interrupt request enable bit

**[bit19:16] Reserved: Reserved bits**

## 5. Precautions for Using

The following shows the notes when using the 16-bit input capture.

### (1) Notes When Accessing a Register

#### a) When Accessing the Input Capture Data Registers 0, 1 (IPCP0, 1)

Use half-word or word access instructions for the input capture data registers 0, 1 (IPCP0, 1).

#### b) When Accessing the Input Capture State Control Register (ICS)

- This register supports writing from the bit-band alias area. For the bit-band alias area, see "CHAPTER: Bit-band Unit".
- To clear a specified bit in this register, clear the bit by writing "1" to the applicable bit in the input capture state control clear register (ICSC). It is prohibited to directly clear only a specific bit in this register.
- To set a specified bit in this register, set the bit by writing "1" to the applicable bit in the input capture state control set register (ICSS). It is prohibited to directly set only a specific bit in this register.
- Data can be written directly to this register only when writing to all bits.

### (2) Notes on Interrupt Processing

- The valid edge indication bit (IEI1, IEI0) in the input capture state control register (ICS) indicates a detected latest edge when the level of the external input pin (IN0, IN1) switches while an interrupt routine is processing after the interrupt request flag (ICP1, ICP0) in the input capture state control register (ICS) is set to "1".



# CHAPTER 49: 16-bit Output Compare



This Chapter Explains 16-bit Output Compare.

---

1. Overview
2. Configuration
3. Operation
4. Registers
5. Precautions for Using





## 1. Overview

16-bit output compare consists of 16-bit compare registers, compare output latch, compare control registers, and compare mode control registers.

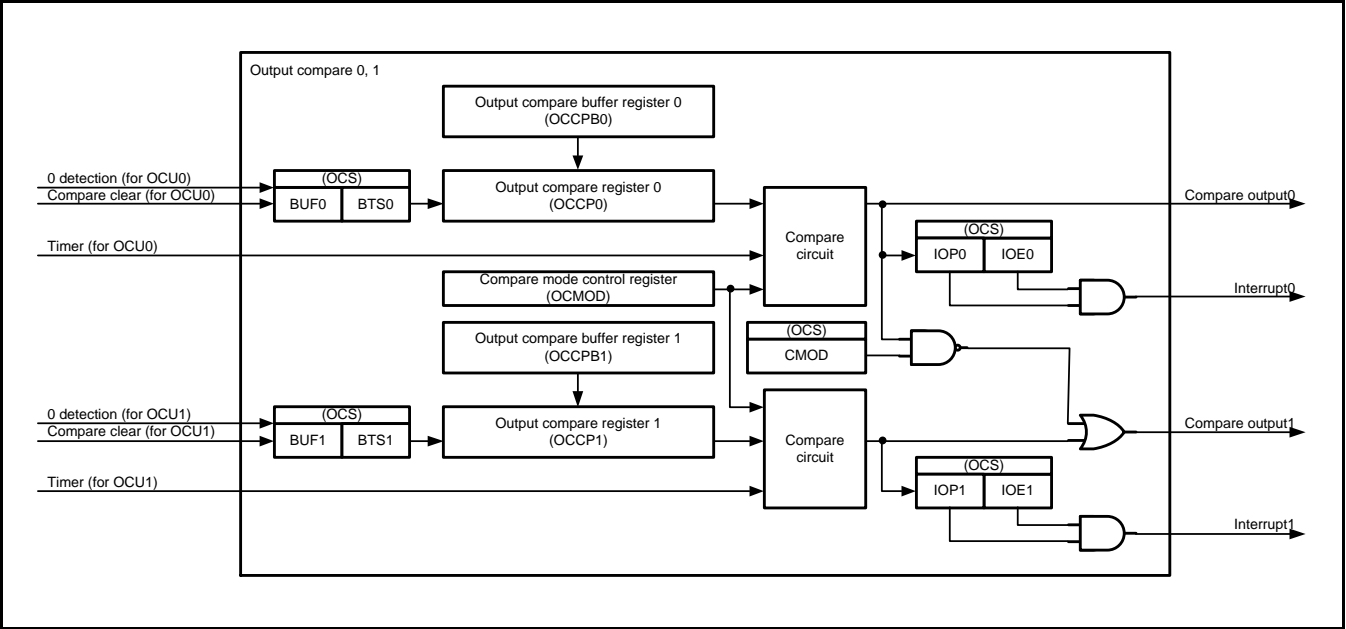
### Function of 16-bit Output Compare

- 16-bit output compare consists of 16-bit compare registers, compare output latch, compare control registers, and compare mode control registers. When the 16-bit free-run timer value matches a compare register, an interrupt is generated and the output level is inverted.
- The 12 compare registers can be operated independently. An output pin and an interrupt flag correspond to each of the compare registers.
- Two compare registers can be used as a pair to control output pins. Two compare registers combined as a pair can be used to invert the output pins.
- The initial values for output pins can be set.
- An interrupt can be generated when an output compare register matches the 16-bit free-run timer.

2. Configuration

This section shows a block diagram of 16-bit output compare.

Figure 2-1 Block Diagram of 16-bit Output Compare (ch.0, ch.1)





### 3. Operation

This section describes the operation of 16-bit output compare.

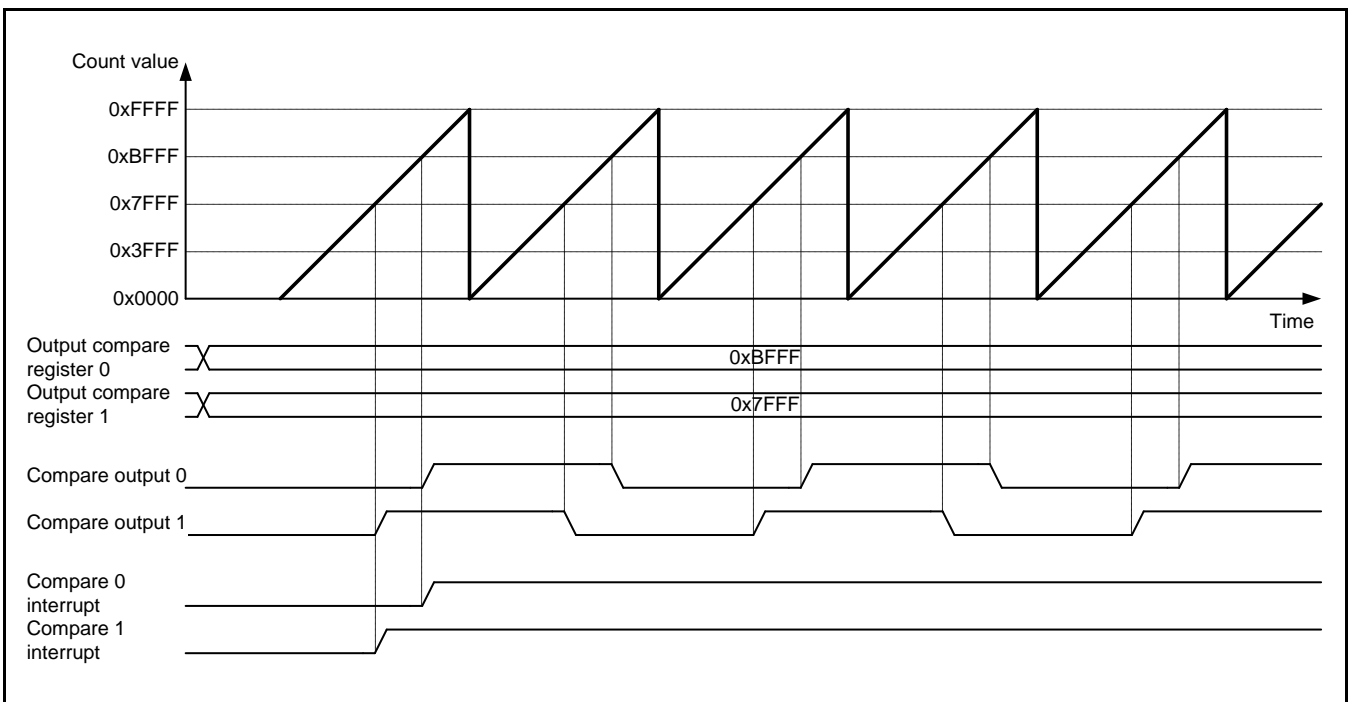
#### (1) Operation of 16-bit Output Compare

The output compare is used to compare the value set in the specified compare clear register and the value of the 16-bit free-run timer. When a match is detected, the interrupt flag is set and the output level is inverted. If there is a match between the count peak and the compare register value while the free-run timer is in up/down count mode, the match signal is ignored.

#### (2) Operation of 16-bit Output Compare (Inverted Mode, MOD0= 0 in OCMOD Register)

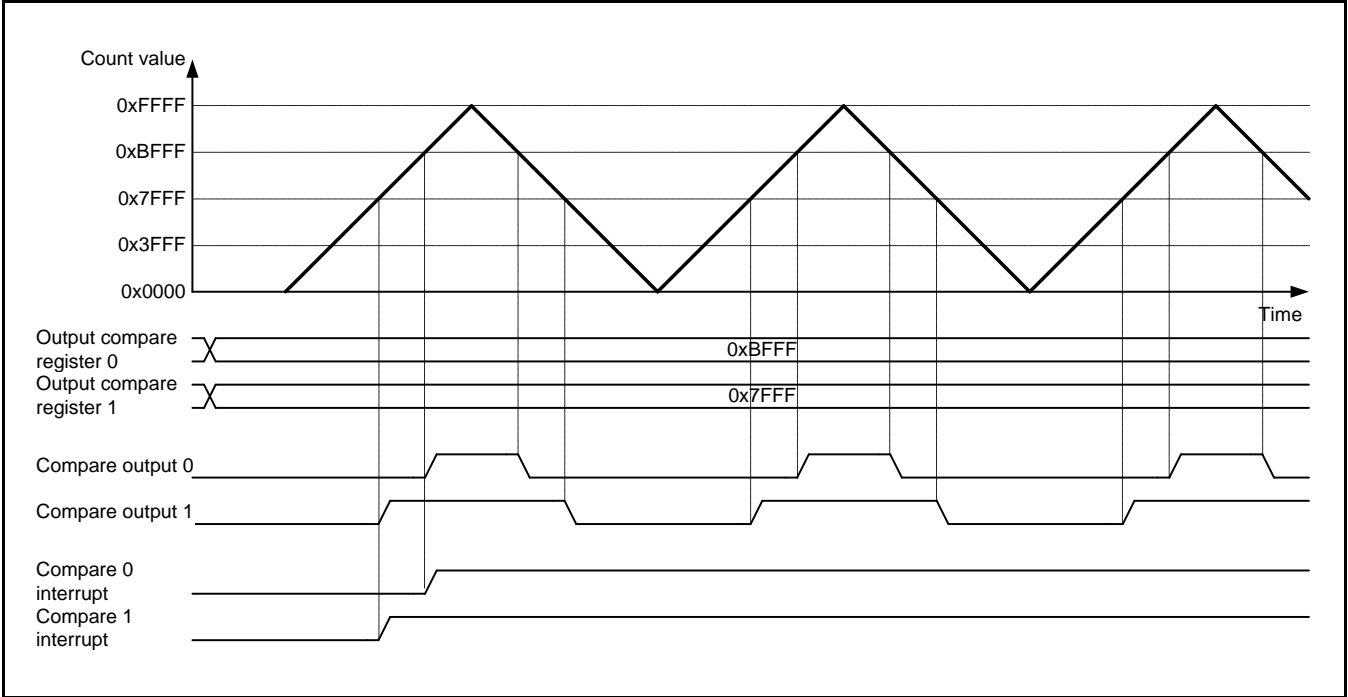
##### a) The Compare Operation can Be Performed in Each of the Channels (CMOD=0 in the Compare Control Register (OCS))

**Figure 3-1 Example of Output Waveform When the Initial Output Value is "0" and Compare Registers 0 and 1 are Used Independently (with the Free-run Timer in Up Count Mode)**





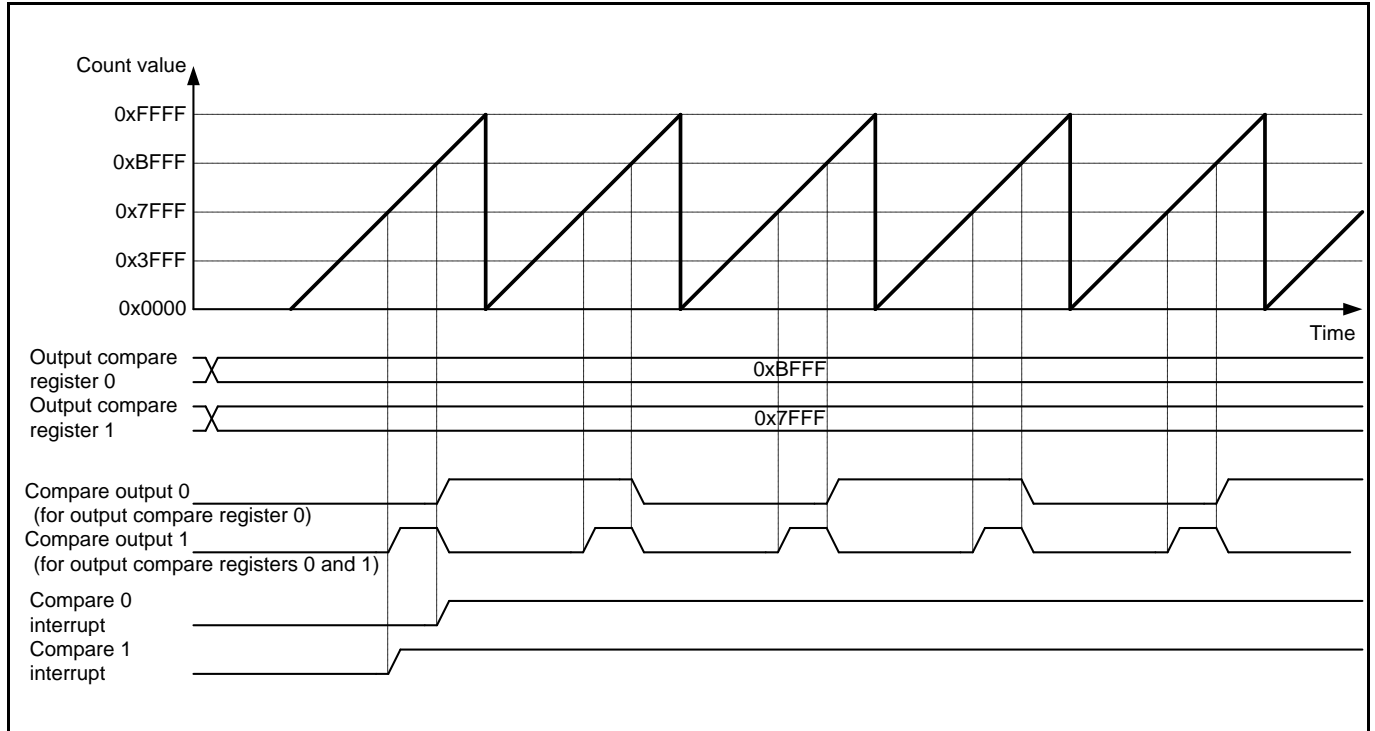
**Figure 3-2 Example of Output Waveform When the Initial Output Value is "0" and Compare Registers 0 and 1 are Used Independently (with the Free-run Timer in Up/Down Count Mode)**



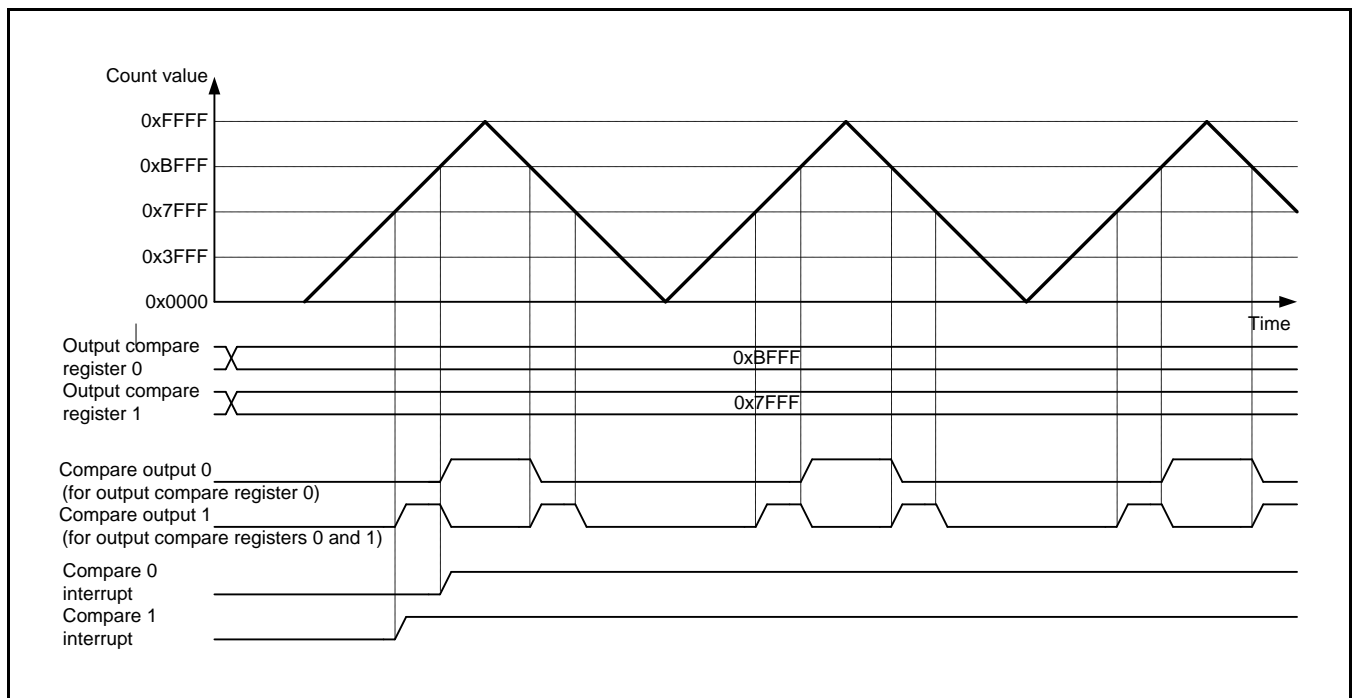


**b)The Output Level can Be Changed Using a Pair of Compare Registers (CMOD=1 in OCS).**

**Figure 3-3 Example of Output Waveform When the Initial Output Value is "0" and Compare Registers 0 and 1 are Used as a Pair (with the Free-run Timer in Up Count Mode)**

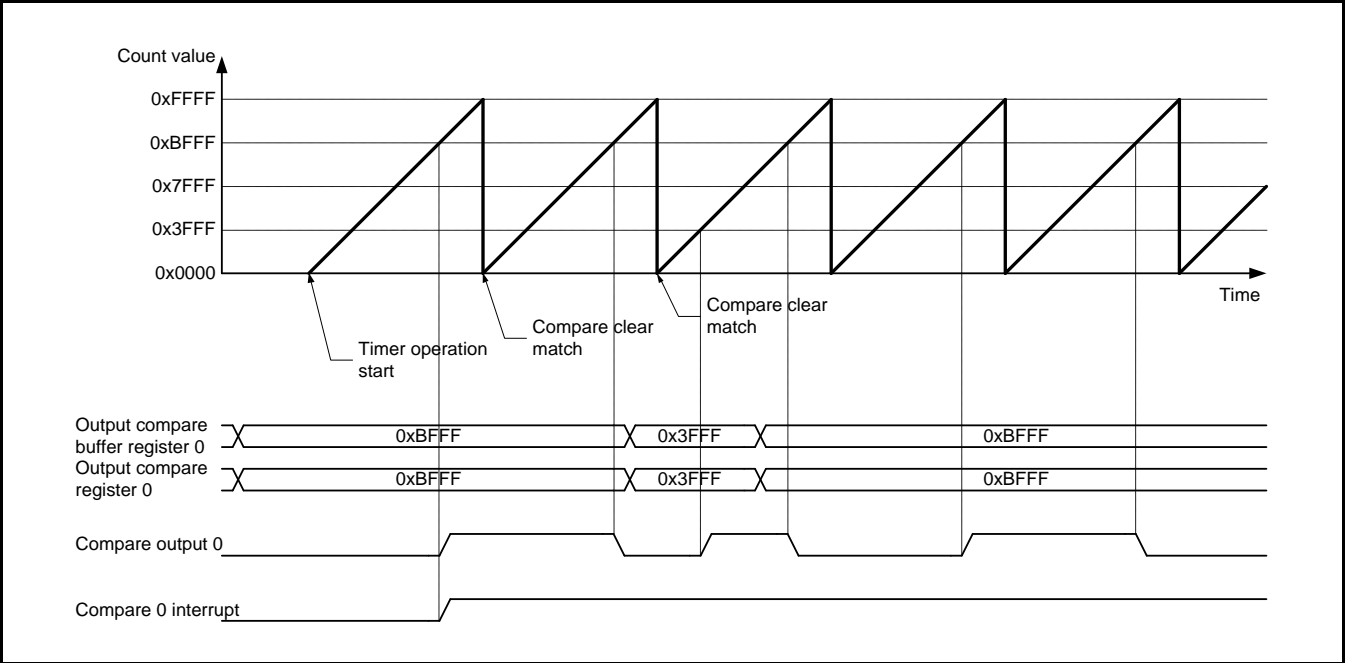


**Figure 3-4 Example of Output Waveform When the Initial Output Value is "0" and Compare Registers 0 and 1 are Used Simultaneously (with the Free-run Timer in Up/Down Count Mode)**



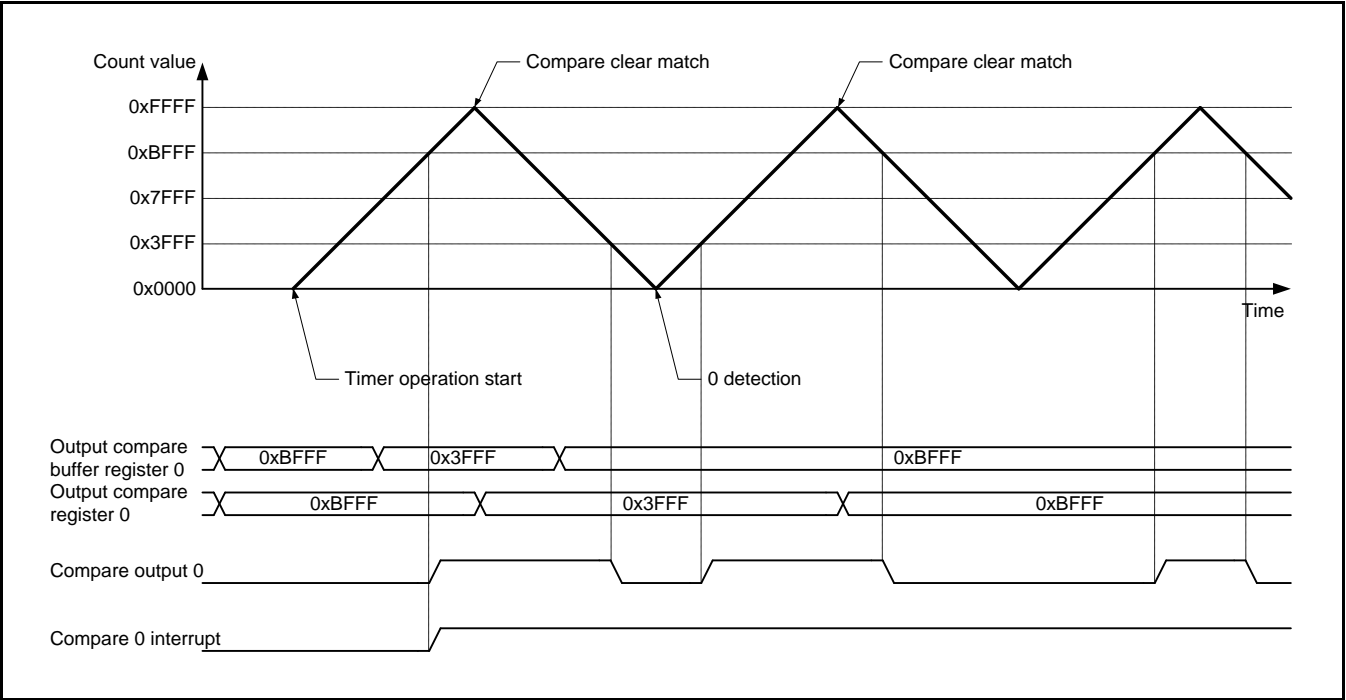
c) Output Level When the Compare Buffer is Invalid

Figure 3-5 Example of Output Waveform When the Compare Buffer is Invalid (with the Free-run Timer in Up Count Mode)



d) Output Level When the Compare Buffer is Selected Upon a Compare Clear Match

Figure 3-6 Example of Output Waveform When the Compare Buffer is Valid (with the Free-run Timer in Up/Down Count Mode)





### (3) Operation of 16-bit Output Compare (Set/Reset Mode, MODn=1 (n=0 to 1) in OCMOD Register)

Figure 3-7 Operation of 16-bit Output Compare (Set/Reset Mode) #1

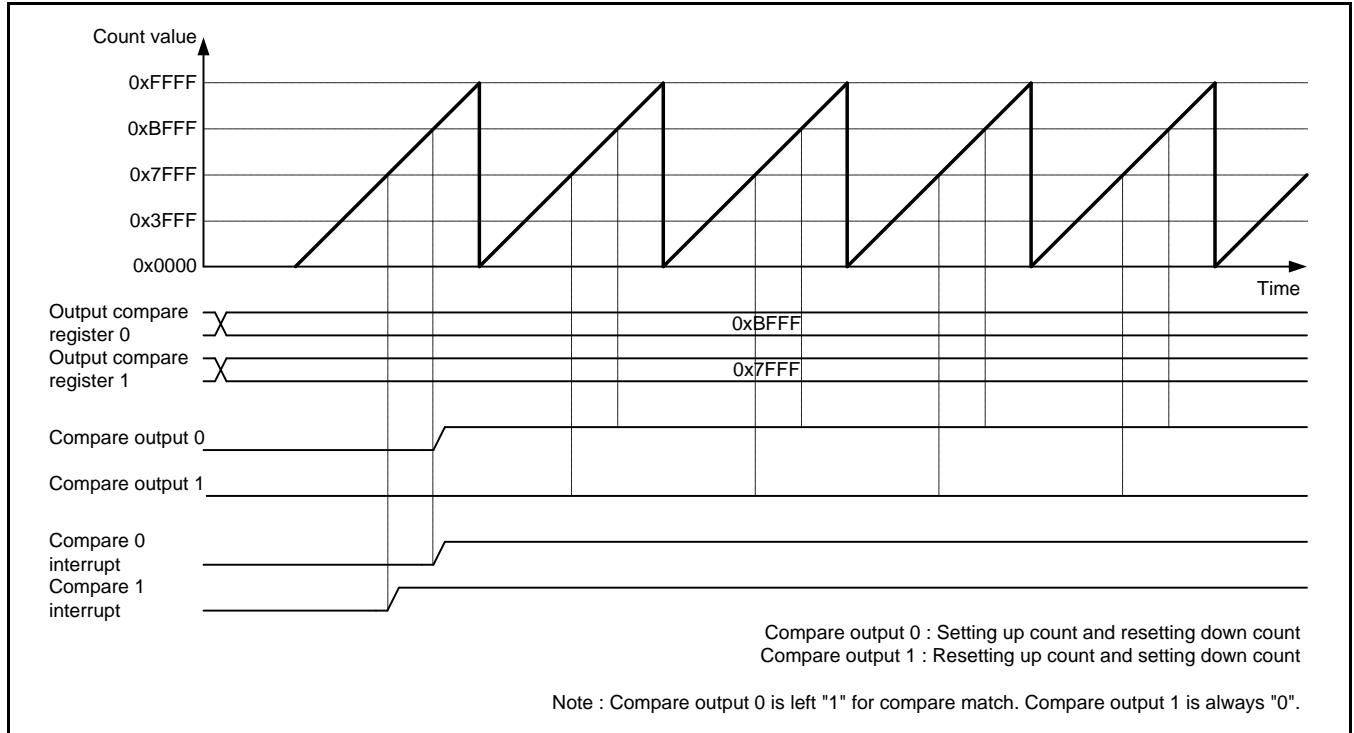
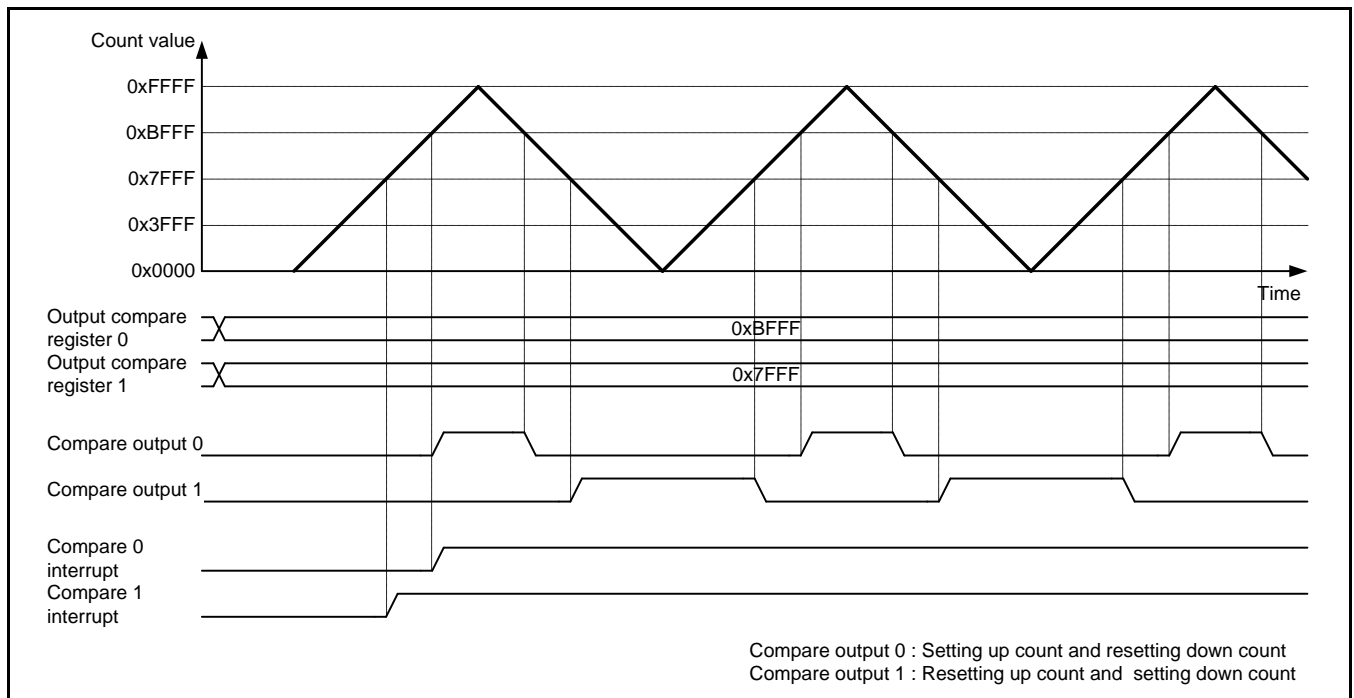


Figure 3-8 Operation of 16-bit Output Compare (Set/Reset Mode) #2



#### (4) 16-bit Output Compare Timing

When the free-run timer value matches the compare register value, the output compare generates a compare match signal and inverts the output to generate an interrupt. When a compare match occurs, the output is inverted in synchronization with the counter count timing.

Figure 3-9 Compare Register Interrupt Timing

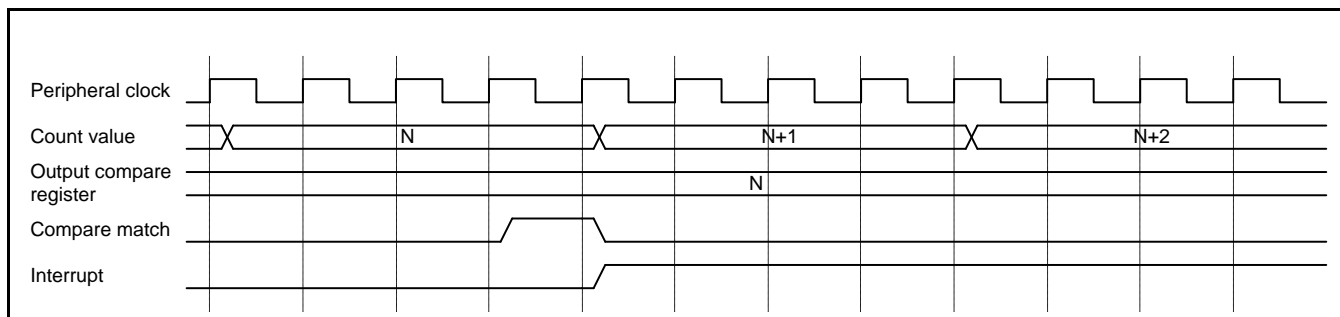
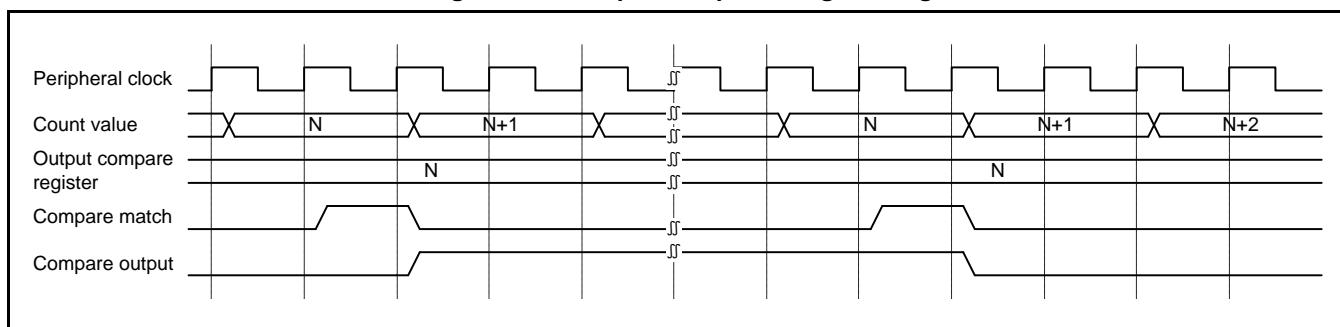


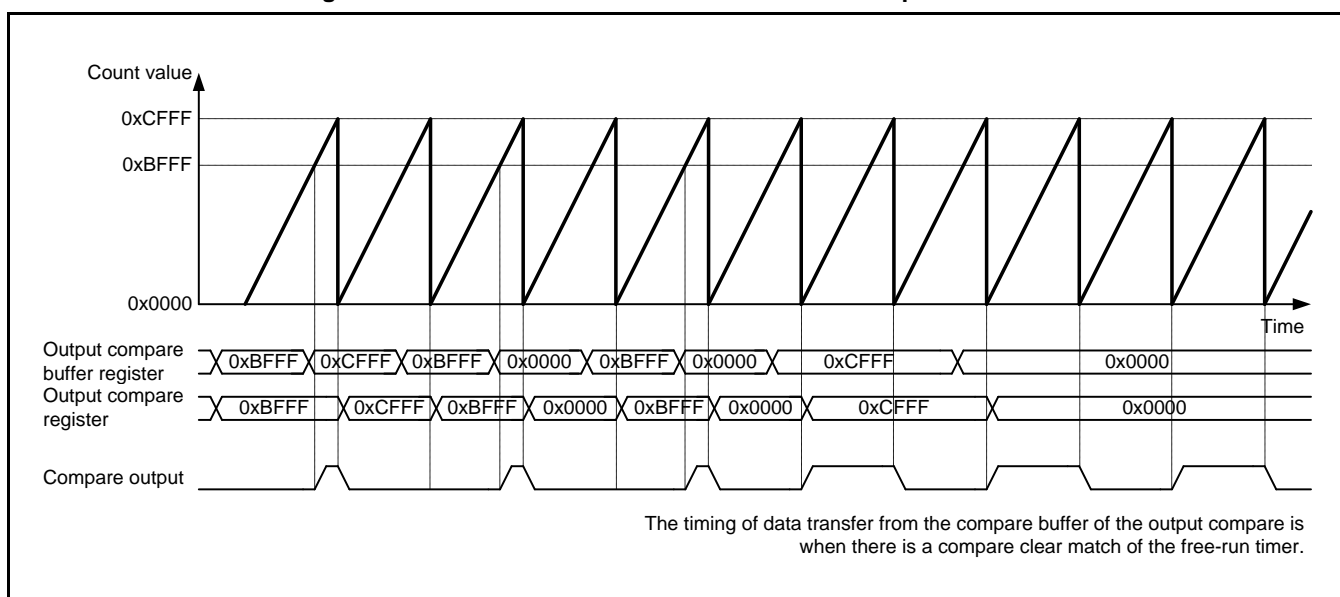
Figure 3-10 Compare Output Change Timing



#### (5) Operation of 16-bit Output Compare and Free-run Timer

##### a) Case #1 Where the Free-run Timer is in Up Count Mode

Figure 3-11 Case #1 Where the Free-run Timer is in Up Count Mode

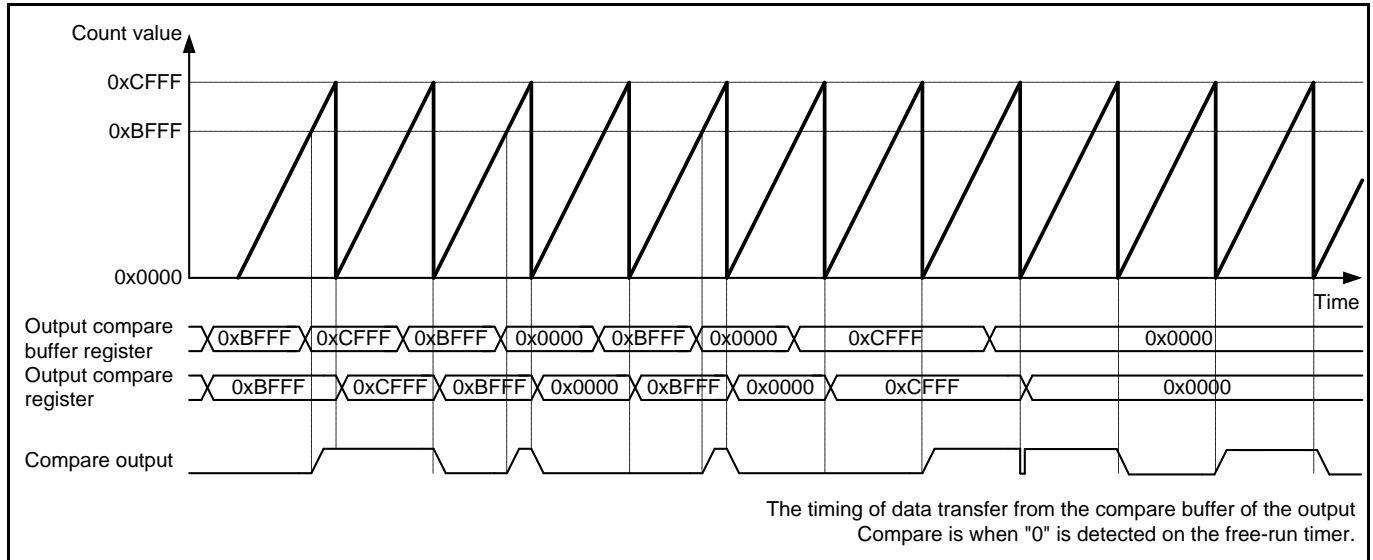






**b) Case #2 Where the Free-run Timer is in Up Count Mode**

**Figure 3-12 Case #2 Where the Free-run Timer is in Up Count Mode**



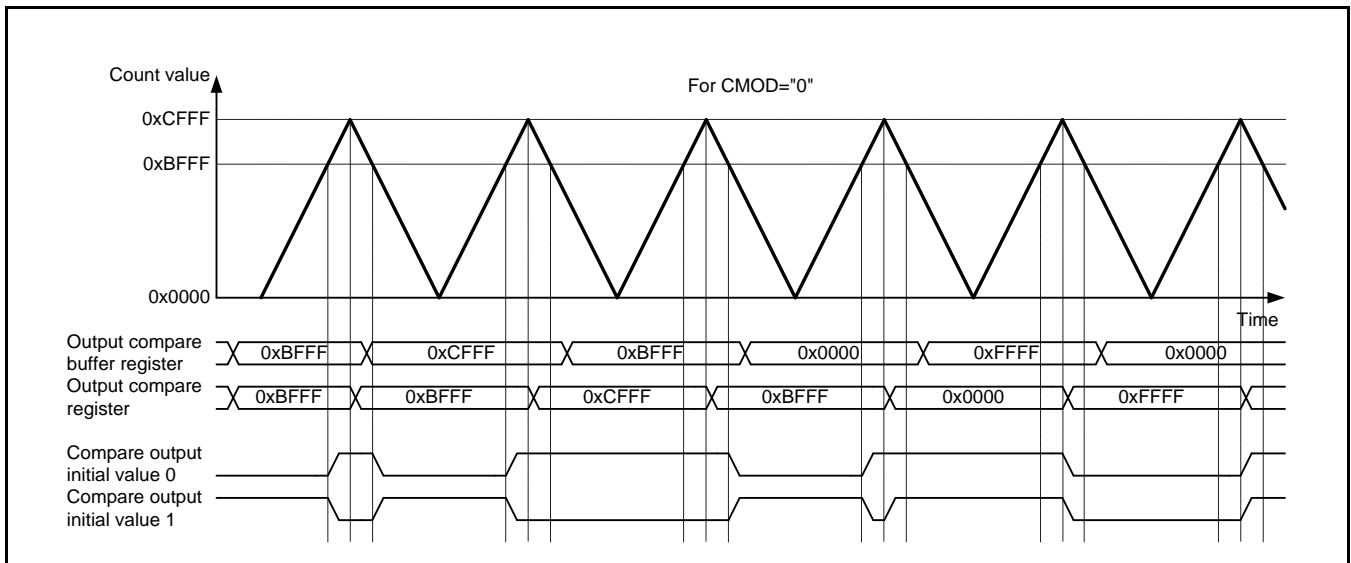
**c) Case #1 Where the Free-run Timer is in Up/Down Count Mode**

- The timing of data transfer from the compare buffer of the output compare is when there is a compare clear match of the free-run timer.
- When there is an output compare output match, the output is in inverted mode.

**Notes:**

- When the output compare register value is set to "0x0000", the compare output is set to "H" regardless of the count value of the free-run timer (or reset to "0" when CMOD=1 in OCS).
- When the output compare register value is set to "0xFFFF", the compare output is reset to "L" regardless of the count value of the free-run timer (or set to "1" when CMOD=1 in OCS).
- No comparison is made when there is a match between the compare clear register value of the free-run timer and the compare register value of the output compare. Note that a compare match occurs only once at the time of starting the free-run timer when the initial value of the free-run timer is same as the compare clear register value. If, at this time, both the compare clear register value and the compare register value are set to "0xFFFF", the compare output is reset to "L" regardless of the count value of the free-run timer.

Figure 3-13 Case #1 Where the Free-run Timer is in Up/Down Count Mode



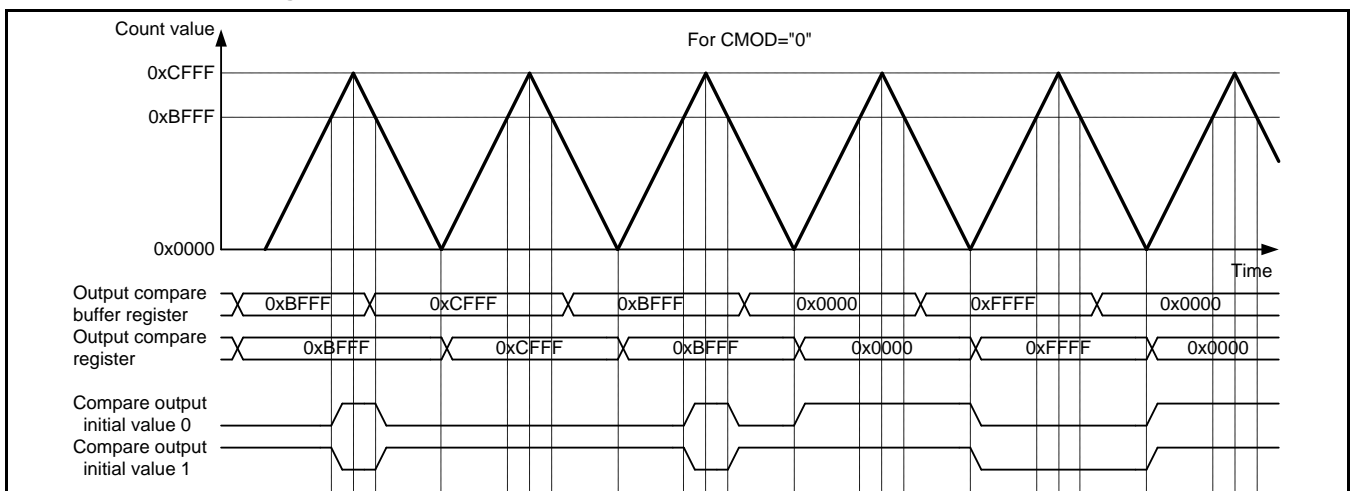
**d) Case #2 Where the Free-run Timer is in Up/Down Count Mode**

- The timing of data transfer from the compare buffer of the output compare is when "0" is detected on the free-run timer.
- When there is an output compare output match, the output is in inverted mode.

**Notes:**

- When the output compare register value is set to "0x0000", the compare output is set to "H" regardless of the count value of the free-run timer (or reset to "0" when CMOD = 1 in OCS).
- When the output compare register value is set to "0xFFFF", the compare output is reset to "L" regardless of the count value of the free-run timer (or set to "1" when CMOD = 1 in OCS).
- No comparison is made when there is a match between the compare clear register value of the free-run timer and the compare register value of the output compare. Note that a compare match occurs only once at the time of starting the free-run timer when the initial value of the free-run timer is same as the compare clear register value. If, at this time, both the compare clear register value and the compare register value are set to "0xFFFF", the compare output is reset to "L" regardless of the count value of the free-run timer.

Figure 3-14 Case #2 Where the Free-run Timer is in Up/Down Count Mode





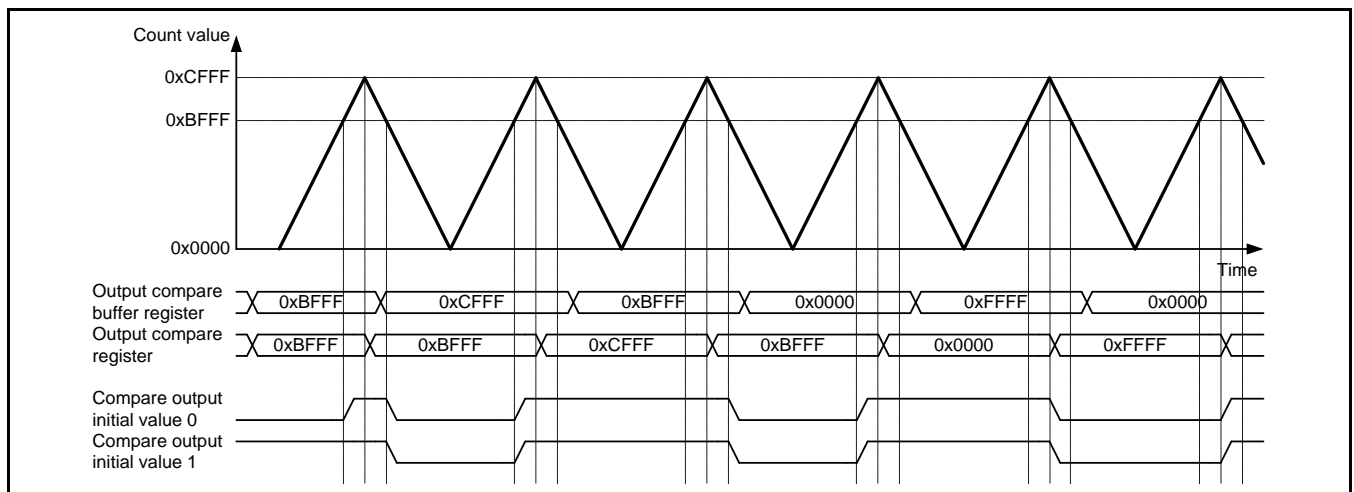
**e) Case #3 Where the Free-run Timer is in Up/Down Count Mode**

- The timing of data transfer from the compare buffer of the output compare is when there is a compare clear match of the free-run timer.
- The output compare output is set to "1" upon a match in up count mode or reset to "0" upon a match in down count mode (CMOD =0 in OCS).

**Notes:**

- When the output compare register value is set to "0x0000", the compare output is set to "H" regardless of the count value of the free-run timer.
- When the output compare register value is set to "0xFFFF", the compare output is reset to "L" regardless of the count value of the free-run timer.
- No comparison is made when there is a match between the compare clear register value of the free-run timer and the compare register value of the output compare. Note that a compare match occurs only once at the time of starting the free-run timer when the initial value of the free-run timer is same as the compare clear register value. If, at this time, both the compare clear register value and the compare register value are set to "0xFFFF", the compare output is reset to "L" regardless of the count value of the free-run timer.

**Figure 3-15 Case #3 Where the Free-run Timer is in Up/Down Count Mode**



**f) Case #4 Where the Free-run Timer is in Up/Down Count Mode**

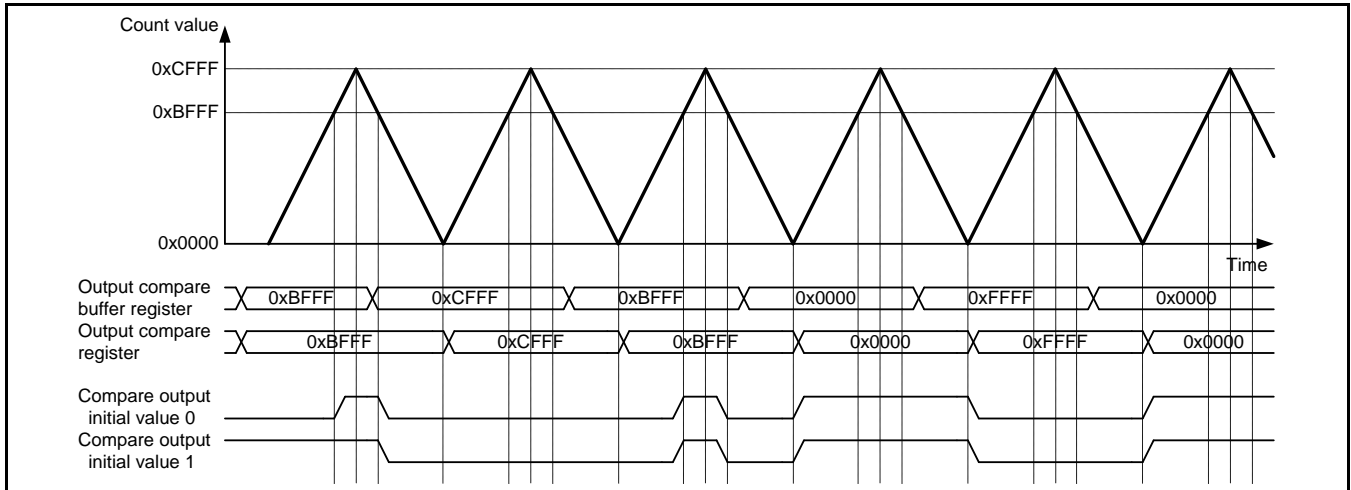
- The timing of data transfer from the compare buffer of the output compare is when "0" is detected on the free-run timer.
- The output compare output is set to "1" upon a match in up count mode or reset to "0" upon a match in down count mode (CMOD =0 in OCS).

**Notes:**

- When the output compare register value is set to "0x0000", the compare output is set to "H" regardless of the count value of the free-run timer.
- When the output compare register value is set to "0xFFFF", the compare output is reset to "L" regardless of the count value of the free-run timer.
- No comparison is made when there is a match between the compare clear register value of the free-run timer and the compare register value of the output compare. Note that a compare match occurs only once at the time of starting the free-run timer when the initial value of the free-run timer is same as the compare clear register value. If, at this time, both the compare clear register value

and the compare register value are set to "0xFFFF", the compare output is reset to "L" regardless of the count value of the free-run timer.

**Figure 3-16 Case #4 Where the Free-run Timer is in Up/Down Count Mode**



**g) Case #5 Where the Free-run Timer is in Up/Down Count Mode**

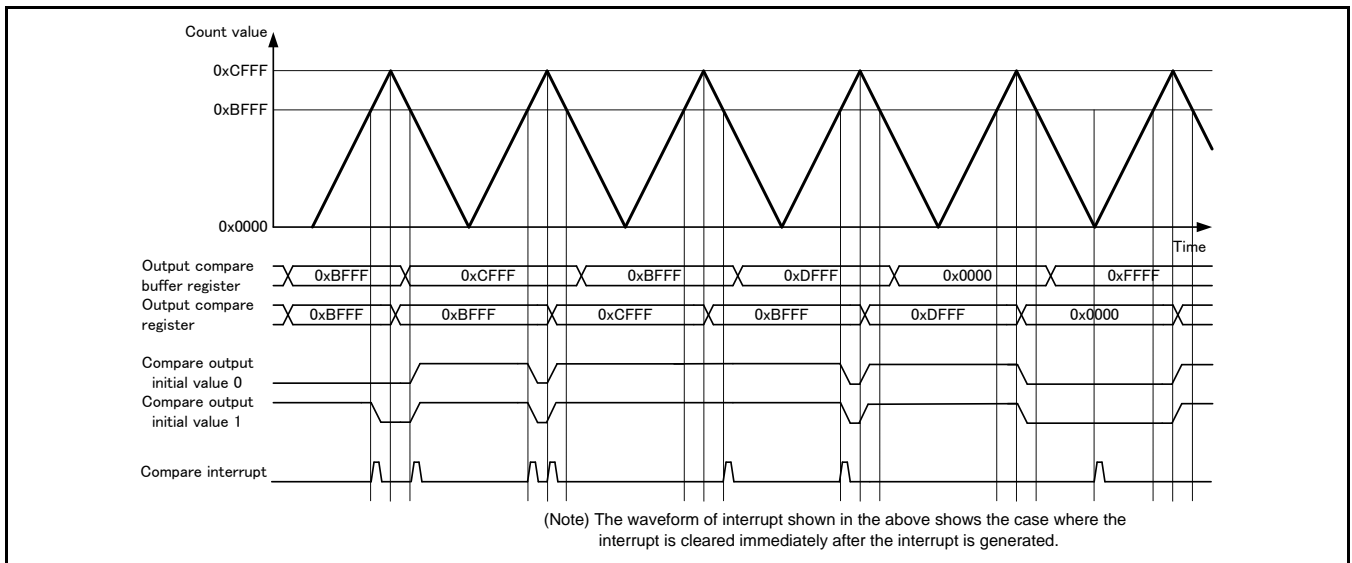
- The timing of data transfer from the compare buffer of the output compare is when there is a compare clear match of the free-run timer.
- The output compare output is set to "0" upon a match in up count mode or reset to "1" upon a match in down count mode (CMOD =1 in OCS).

**Notes:**

- When the output compare register value is set to "0x0000", the compare output is "L" when there is a compare clear match of the free-run timer.
- When the output compare register value is set to a value greater than or equal to the compare clear register value of the free-run timer, the compare output is "H" when there is a compare clear match of the free-run timer.
- A comparison will be made and an interrupt flag occurs when there is a match between the compare clear register value of the free-run timer and the compare register value of the output compare.



Figure 3-17 Case #5 Where the Free-run Timer is in Up/Down Count Mode



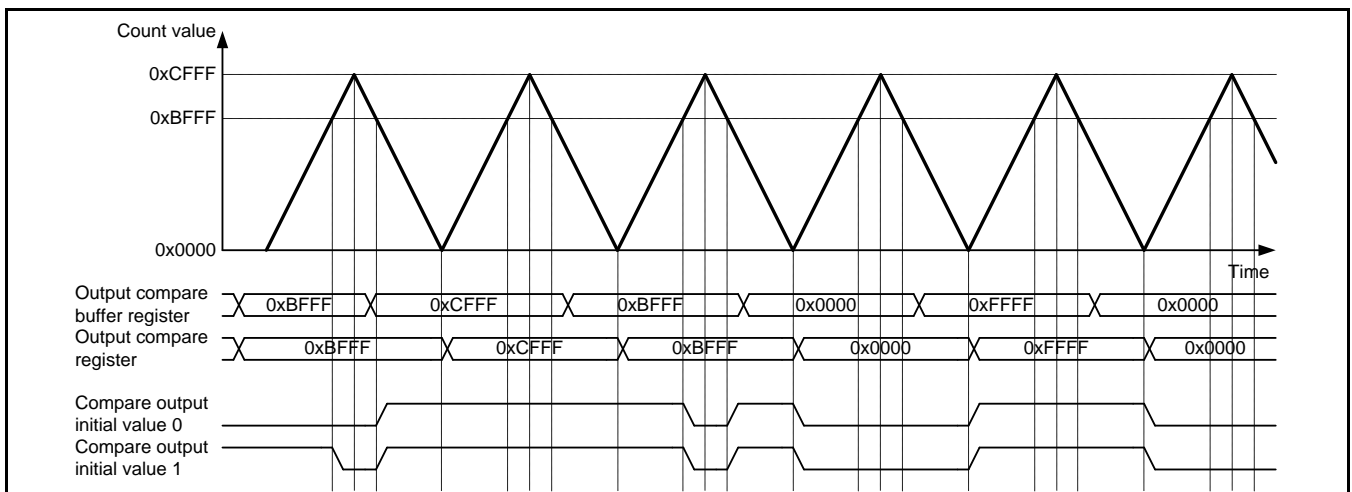
#### h) Case #6 Where the Free-run Timer is in Up/Down Count Mode

- The timing of data transfer from the compare buffer of the output compare is when "0" is detected on the free-run timer.
- The output compare output is set to "0" upon a match in up count mode or reset to "1" upon a match in down count mode (CMOD =1 in OCS).

#### Notes:

- When the output compare register value is set to "0x0000", the compare output is reset to "L" regardless of the count value of the free-run timer.
- When the output compare register value is set to "0xFFFF", the compare output is set to "H" regardless of the count value of the free-run timer.
- No comparison is made when there is a match between the compare clear register value of the free-run timer and the compare register value of the output compare. Note that a compare match occurs only once at the time of starting the free-run timer when the initial value of the free-run timer is same as the compare clear register value. If, at this time, both the compare clear register value and the compare register value are set to "0xFFFF", the compare output is reset to "L" regardless of the count value of the free-run timer.

Figure 3-18 Case #6 Where the Free-run Timer is in Up/Down Count Mode



### 3.1. Interrupt

An interrupt is generated when the 16-bit free-run timer value matches the output compare register.

#### Interrupts for 16-bit Output Compare

Table 3-1 shows the interrupt control bits and interrupt factor of the 16-bit output compare.

**Table 3-1 Interrupt Control Bits and Interrupt Factor of 16-bit Output Compare**

	Output Compare 0	Output Compare 1
Interrupt request flag bit	Compare match interrupt flag bit (IOP0) in the compare control register (OCS)	Compare match interrupt flag bit (IOP1) in the compare control register (OCS)
Interrupt request enable bit	Compare match interrupt enable bit (IOE0) in the compare control register (OCS)	Compare match interrupt enable bit (IOE1) in the compare control register (OCS)
Interrupt factor	The 16-bit free-run timer value matches the output compare register 0 (OCCP0).	The 16-bit free-run timer value matches the output compare register 1 (OCCP1).

When the 16-bit free-run timer value matches the output compare register 0 (OCCP0), IOP0 in the compare control register (OCS) are set to "1". Also, when the 16-bit free-run timer value matches the output compare register 1 (OCCP1), IOP1 in the compare control register (OCS) are set to "1". If interrupt requests are enabled (IOE1/IOE0 = 1 of OCS) in this state, an interrupt request is output to the interrupt controller.



## 4. Registers

This section describes the registers of 16-bit output compare.

A prefix (OCU16Bxx) is added to each of the 16-bit output compare registers. xx is the channel number.

**Table 4-1 List of 16-bit Output Compare Registers**

Abbreviated Register Name	Register Name	See
OCU16Bxx_OCCPB0/ OCU16Bxx_OCCP0	Output Compare Buffer Register 0 / Output Compare Register 0	4.1
OCU16Bxx_OCCPB1/ OCU16Bxx_OCCP1	Output Compare Buffer Register 1 / Output Compare Register 1	4.2
OCU16Bxx_OCS	Compare Control Register	4.3
OCU16Bxx_OCMOD	Compare Mode Control Register	4.4
OCU16Bxx_OCSC	Compare Control Clear Register	4.5
OCU16Bxx_OCSS	Compare Control Set Register	4.6

xx: channel number (xx=00, 02, 04, 06, 08, 10, 12, 14, 16, 18, 20, 22)

## 4.1. Output Compare Buffer Register 0 (OCCPB0) / Output Compare Register 0 (OCCP0)

The output compare buffer register 0 (OCCPB0) is a 16-bit buffer register for the output compare register 0 (OCCP0).

The output compare register 0 (OCCP0) is a 16-bit register to be used for comparison with the count value of the 16-bit free-run timer.

Both the output compare buffer register 0 (OCCPB0) and the output compare register 0 (OCCP0) registers are located at the same address.

### (1) Output Compare Buffer Register 0 (OCCPB0)

BIT_OFFSET	31-16
BIT_NAME	OP
ACCESS_TYPE	W
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000

#### [bit31:16] OP[15:0]: Compare value buffer bits

The output compare buffer register 0 is a buffer register for the output compare register 0 (OCCP0). If the buffer function is disabled (BUF0=1 in the compare control register (OCS)) or the free-run timer stops, the value of the output compare buffer register will be immediately transferred to the output compare register. If the buffer function is enabled (BUF0=0 in the compare control register (OCS)), the value will be transferred to the output compare register when a compare clear match or 0 is detected in accordance with the transfer selection bit (BTS0) in the compare control register (OCS).

#### **Note:**

- When accessing this register, use a half-word or word access instruction.





## (2) Output Compare Register 0 (OCCP0)

BIT_OFFSET	31-16
BIT_NAME	OP
ACCESS_TYPE	R
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000

### [bit31:16] OP[15:0]: Compare value bits

- The output compare register 0 is a 16-bit register to be used for comparison with the count value of the 16-bit free-run timer. Before enabling the operation of the 16-bit free-run timer, set a value in the output compare buffer register 0 (OCCPB0).
- When the value of the output compare register 0 matches the count value of the 16-bit free-run timer, a compare signal is generated and the output compare interrupt flag bit (IOP0 in the compare control register (OCS)) is set. When the output level is set (OTD0 in the compare control register (OCS)), the compare output level for the output compare register 0 (OCCP0) can be inverted.
- When all the conditions listed below are met and a value that exceeds the peak value of the 16-bit free-run timer is set to this register, the output compare output is "H" right after the buffer transfer. When this register is set to "0x0000", the output compare output is "L" right after the buffer transfer.
  - The free-run timer is in up/down count mode.
  - When the BUF bit in the compare control register (OCS) is "0" (buffer function enabled)
  - When the BTS bit in the compare control register (OCS) is "1" (transfer when there is a compare clear match)
  - When the CMOD bit in the compare control register (OCS) is "1"
  - When the MOD bit in the compare mode control register (OCMOD) is "1"
- When all the conditions listed above are not met, even if the value of this register matches the peak value of the 16-bit free-run timer in up/down mode, no compare signal is generated. The outcome is as follows according to the settings of the CMOD bit in the compare control register (OCS)
  - When the CMOD bit in the compare control register (OCS) is 1  
When this register is set to "0xFFFF", the output compare output is "H" regardless of the value of the 16-bit free-run timer and the inversion mode. When this register is set to "0x0000", the output compare output is "L".
  - When the CMOD bit in the compare control register (OCS) is 0  
When this register is set to "0xFFFF", the output compare output is "L" regardless of the value of the 16-bit free-run timer and the inversion mode. When this register is set to "0x0000", the output compare output is "H".

#### Note:

- When accessing this register, use a half-word or word access instruction.

## 4.2. Output Compare Buffer Register 1 (OCCPB1) / Output Compare Register 1 (OCCP1)

The output compare buffer register 1 (OCCPB1) is a 16-bit buffer register for the output compare register 1 (OCCP1).

The output compare register 1 (OCCP1) is a 16-bit register to be used for comparison with the count value of the 16-bit free-run timer.

Both the output compare buffer register 1 (OCCPB1) and the output compare register 1 (OCCP1) registers are located at the same address.

### (1) Output Compare Buffer Register 1 (OCCPB1)

BIT_OFFSET	15-0
BIT_NAME	OP
ACCESS_TYPE	W
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000

#### [bit15:0] OP[15:0]: Compare value buffer bits

The output compare buffer register 1 is a buffer register for the output compare register 1 (OCCP1). If the buffer function is disabled (BUF1=1 in the compare control register (OCS)) or the free-run timer stops, the value of the output compare buffer register will be immediately transferred to the output compare register. If the buffer function is enabled (BUF1=0 in the compare control register (OCS)), the value will be transferred to the output compare register when a compare clear match or 0 is detected in accordance with the transfer selection bit (BTS1) in the compare control register (OCS).

#### **Note:**

- When accessing this register, use a half-word or word access instruction.



## (2) Output Compare Register 0 (OCCP0)

BIT_OFFSET	15-0
BIT_NAME	OP
ACCESS_TYPE	R
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000

### [bit15:0] OP[15:0]: Compare value bits

- The output compare register 1 is a 16-bit register to be used for comparison with the count value of the 16-bit free-run timer. Before enabling the operation of the 16-bit free-run timer, set a value in the output compare buffer register 1 (OCCPB1).
- When the value of the output compare register 1 matches the count value of the 16-bit free-run timer, a compare signal is generated and the output compare interrupt flag bit (IOP0 in the compare control register (OCS)) is set. When the output level is set (OTD0 in the compare control register (OCS)), the compare output level for the output compare register 1(OCCP1) can be inverted.
- When all the conditions listed below are met and a value that exceeds the peak value of the 16-bit free-run timer is set to this register, the output compare output is "H" right after the buffer transfer. When this register is set to "0x0000", the output compare output is "L" right after the buffer transfer.
  - The free-run timer is in up/down count mode.
  - When the BUF bit in the compare control register (OCS) is "0" (buffer function enabled)
  - When the BTS bit in the compare control register (OCS) is "1" (transfer when there is a compare clear match)
  - When the CMOD bit in the compare control register (OCS) is "1"
  - When the MOD bit in the compare mode control register (OCMOD) is "1"
- When all the conditions listed above are not met, even if the value of this register matches the peak value of the 16-bit free-run timer in up/down mode, no compare signal is generated. The outcome is as follows according to the settings of the CMOD bit in the compare control register (OCS)
  - When the CMOD bit in the compare control register (OCS) is 1  
When this register is set to "0xFFFF", the output compare output is "H" regardless of the value of the 16-bit free-run timer and the inversion mode. When this register is set to "0x0000", the output compare output is "L".
  - When the CMOD bit in the compare control register (OCS) is 0  
When this register is set to "0xFFFF", the output compare output is "L" regardless of the value of the 16-bit free-run timer and the inversion mode. When this register is set to "0x0000", the output compare output is "H".

#### Note:

- When accessing this register, use a half-word or word access instruction.

### 4.3. Compare Control Register (OCS)

The compare control register (OCS) is used to control the output level, output enable, output level inversion mode, compare operation enable, compare match interrupt enable, and compare match interrupt flag in OUT0 / OUT1. For details on writing to this register, see Section "5. Precautions for Using".

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved	BTS1	BTS0	CMOD	Reserved		OTD1	OTD0
ACCESS_TYPE	R0,W0	R/W	R/W	R/W	R/W0		R,W	R,W
PROT_TYPE	-							
INITIAL_VALUE	0	1	1	0	00		0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	IOP1	IOP0	IOE1	IOE0	BUF1	BUF0	CST1	CST0
ACCESS_TYPE	R,W	R,W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	1	1	0	0

**[bit31] Reserved: Reserved bit**

#### **[bit30] BTS1: Buffer transfer selection bit**

- This bit is used to select the timing of data transfer from the output compare buffer register 1 (OCCPB1) to the output compare register 1 (OCCP1).
- When this bit is set to "0":
- Data transfer is activated when the count value of "0" is detected on the 16-bit free-run timer.
- When this bit is set to "1":
- Data transfer is activated when a compare clear match occurs on the 16-bit free-run timer.
- This bit is cleared to "0" by writing "1" to the BTSC1 bit in the OCSC register.
- This bit is set to "1" by writing "1" to the BTSS1 bit in the OCSS register.

Value	Description
0	Transfer is activated when 0 is detected (ch.1).
1	Transfer is activated when a compare clear match occurs (ch.1).

#### **[bit29] BTS0: Buffer transfer selection bit**

- This bit is used to select the timing of data transfer from the output compare buffer register 0 (OCCPB0) to the output compare register 0 (OCCP0).
- When this bit is set to "0":
- Data transfer is activated when the count value of "0" is detected on the 16-bit free-run timer.
- When this bit is set to "1":
- Data transfer is activated when a compare clear match occurs on the 16-bit free-run timer.
- This bit is cleared to "0" by writing "1" to the BTSC0 bit in the OCSC register.
- This bit is set to "1" by writing "1" to the BTSS0 bit in the OCSS register.

Value	Description
0	Transfer is activated when 0 is detected (ch.0).
1	Transfer is activated when a compare clear match occurs (ch.0).



**[bit28] CMOD: Output level inversion mode bit**

This bit is used to switch the compare output level inversion mode immediately when a match is detected.

- When this bit is set to "0":

When the compare mode control register: MODn=0

n=0

The compare output m is immediately inverted when there is a match between the 16-bit free-run timer and the output compare register (OCCPm).

n=1

The compare output 1 is immediately inverted when there is a match between the 16-bit free-run timer and the output compare register (OCCP1).

When the compare mode control register: MODn=1

Set to "1" when a match is detected in up-count mode.

Reset to "0" when a match is detected in down-count mode.

- When this bit is set to "1":

When the compare mode control register: MODn=0

n=0

The compare output m is immediately inverted when there is a match between the 16-bit free-run timer and the output compare register (OCCPm).

n=1

The compare output m+1 is immediately inverted when there is a match between the 16-bit free-run timer and the output compare register (OCCPm or OCCPm+1).

When the output compare registers (OCCP0 and OCCP1) have the same value

The operation is the same as when one compare register is used.

When the compare mode control register: MODn=1

Reset to "0" when a match is detected in up-count mode.

Set to "1" when a match is detected in down-count mode.

This bit is cleared to "0" by writing "1" to the CMODC bit in the OCSC register.

This bit is set to "1" by writing "1" to the CMODS bit in the OCSS register.

Value	Description	
	Compare Mode Control Register: MODn="0"	Compare Mode Control Register: MODn="1"
0	<p>n=0 The compare output 0 is immediately inverted when there is a match with the output compare register 0 (OCCP0).</p> <p>n=1 The compare output 1 is immediately inverted when there is a match with the output compare register 1 (OCCP1).</p>	<p>Set to "1" when a match is detected in up-count mode.</p> <p>Reset to "0" when a match is detected in down-count mode.</p>
1	<p>n=0 The compare output 0 is immediately inverted when there is a match with the output compare register 0 (OCCP0).</p> <p>n=1 The compare output 1 is immediately inverted when there is a match with the output compare register 0 or 1 (OCCP0 or OCCP1).</p>	<p>Set to "0" when a match is detected in up-count mode.</p> <p>Reset to "1" when a match is detected in down-count mode.</p>

**[bit27:26] Reserved: Reserved bits**

**[bit25] OTD1: Output level bit**

- This bit is used to change the compare output 1 level of the output compare.
- The initial value of the compare pin output is "0".
- Be sure to stop the compare operation before writing a value to this bit. The value read from this bit is the output compare value (compare output 1).
- This bit is cleared to "0" by writing "1" to the OTDC1 bit in the OCSC register.
- This bit is set to "1" by writing "1" to the OTDS1 bit in the OCSS register.

Value	Description	
	Read	Write
0	Output value of the compare output 1	The compare output 1 outputs "0".
1		The compare output 1 outputs "1".

**Note:**

- A value can be written to this bit when CST1="0" in the compare control register (OCS).

**[bit24] OTD0: Output level bit**

- This bit is used to change the compare output 0 level of the output compare.
- The initial value of the compare pin output is "0".
- Be sure to stop the compare operation before writing a value to this bit. The value read from this bit is the output compare value (compare output 0).
- This bit is cleared to "0" by writing "1" to the OTDC0 bit in the OCSC register.
- This bit is set to "1" by writing "1" to the OTDS0 bit in the OCSS register.

Value	Description	
	Read	Write
0	Output value of the compare output 0	The compare output 0 outputs "0".
1		The compare output 0 outputs "1".

**Note:**

- A value can be written to this bit when CST0="0" in the compare control register (OCS).

**[bit23] IOP1: Compare match interrupt flag bit**

- This bit is an interrupt flag that indicates whether the value of the output compare register 1 (OCCP1) matched that of the 16-bit free-run timer.
- This bit is set to "1" when the output compare register value matches the 16-bit free-run timer value.
- An output compare interrupt occurs if this bit is set while the compare match interrupt enable bit (IOE1) is enabled ("1").
- When this bit is set to "0": This bit is cleared.
- When this bit is set to "1": This bit remains unaffected.
- This bit is cleared to "0" by writing "1" to the IOPC1 bit in the OCSC register.

Value	Description	
	Read	Write
0	No compare match interrupt occurs for the output compare register 1 (OCCP1).	This bit is cleared.
1	Compare match interrupt occurs for the output compare register 1 (OCCP1).	This bit remains unaffected.

**Note:**

- If a software clear (write of "0") or a clear due to an interrupt clear signal ("H") and a hardware set occur at the same time, the hardware set takes precedence.



**[bit22] IOP0: Compare match interrupt flag bit**

- This bit is an interrupt flag that indicates whether the value of the output compare register 0 (OCCP0) matched that of the 16-bit free-run timer.
- This bit is set to "1" when the output compare register value matches the 16-bit free-run timer value.
- An output compare interrupt occurs if this bit is set while the compare match interrupt enable bit (IOE0) is enabled ("1").
- When this bit is set to "0": This bit is cleared.
- When this bit is set to "1": This bit remains unaffected.
- This bit is cleared to "0" by writing "1" to the IOPC0 bit in the OCSC register.

Value	Description	
	Read	Write
0	No compare match interrupt occurs for the output compare register 0 (OCCP0).	This bit is cleared.
1	Compare match interrupt occurs for the output compare register 0 (OCCP0).	This bit remains unaffected.

**Note:**

- If a software clear (write of "0") or a clear due to an interrupt clear signal ("H") and a hardware set occur at the same time, the hardware set takes precedence.

**[bit21] IOE1: Compare match interrupt enable bit**

- This bit is used to enable an output compare interrupt for the output compare register 1 (OCCP1).
- An output compare interrupt occurs if the compare match interrupt flag bit (IOP1) is set while this bit is set to "1".
- This bit is cleared to "0" by writing "1" to the IOEC1 bit in the OCSC register.
- This bit is set to "1" by writing "1" to the IOES1 bit in the OCSS register.

Value	Description
0	Compare match interrupt is disabled for the output compare register 1 (OCCP1).
1	Compare match interrupt is enabled for the output compare register 1 (OCCP1).

**[bit20] IOE0: Compare match interrupt enable bit**

- This bit is used to enable an output compare interrupt for the output compare register 0 (OCCP0).
- An output compare interrupt occurs if the compare match interrupt flag bit (IOP0) is set while this bit is set to "1".
- This bit is cleared to "0" by writing "1" to the IOEC0 bit in the OCSC register.
- This bit is set to "1" by writing "1" to the IOES0 bit in the OCSS register.

Value	Description
0	Compare match interrupt is disabled for the output compare register 0 (OCCP0).
1	Compare match interrupt is enabled for the output compare register 0 (OCCP0).

**[bit19] BUF1: Compare buffer invalidating bit**

- This bit is used to invalidate the buffer function of the output compare register 1 (OCCP1).
- When this bit is set to "0": This buffer function is validated.
- This bit is cleared to "0" by writing "1" to the BUFC1 bit in the OCSC register.
- This bit is set to "1" by writing "1" to the BUFS1 bit in the OCSS register.

Value	Description
0	Validates the compare buffer of the output compare register 1 (OCCP1).
1	Invalidates the compare buffer of the output compare register 1 (OCCP1).

**[bit18] BUF0: Compare buffer invalidating bit**

- This bit is used to invalidate the buffer function of the output compare register 0 (OCCP0).
- When this bit is set to "0": This buffer function is validated.
- This bit is cleared to "0" by writing "1" to the BUFC0 bit in the OCSC register.
- This bit is set to "1" by writing "1" to the BUFS0 bit in the OCSS register.

Value	Description
0	Validates the compare buffer of the output compare register 0 (OCCP0).
1	Invalidates the compare buffer of the output compare register 0 (OCCP0).

**[bit17] CST1: Compare operation enable bit**

- This bit is used to enable the compare operation between the 16-bit free-run timer and the output compare register 1 (OCCP1).
- Before enabling the compare operation, be sure to write a value to the output compare register 1 (OCCP1) and the timer data register of the free-run timer.
- This bit is cleared to "0" by writing "1" to the CSTC1 bit in the OCSC register.
- This bit is set to "1" by writing "1" to the CSTS1 bit in the OCSS register.

Value	Description
0	Disables the compare operation of the output compare register 1 (OCCP1).
1	Enables the compare operation of the output compare register 1 (OCCP1).

**[bit16] CST0: Compare operation enable bit**

- This bit is used to enable the compare operation between the 16-bit free-run timer and the output compare register 0 (OCCP0).
- Before enabling the compare operation, be sure to write a value to the output compare register 0 (OCCP0) and the timer data register of the free-run timer.
- This bit is cleared to "0" by writing "1" to the CSTC0 bit in the OCSC register.
- This bit is set to "1" by writing "1" to the CSTS0 bit in the OCSS register.

Value	Description
0	Disables the compare operation of the output compare register 0 (OCCP0).
1	Enables the compare operation of the output compare register 0 (OCCP0).





#### 4.4. Compare Mode Control Register (OCMOD)

The compare mode control register (OCMOD) controls the output level upon detection of a compare match by specifying to invert, set, or reset the output level.

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved							
ACCESS_TYPE	R1,W1							
PROT_TYPE	-							
INITIAL_VALUE	11111111							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved						MOD1	MOD0
ACCESS_TYPE	R0,W0						R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	000000						0	0

**[bit15:2] Reserved: Reserved bits**

##### **[bit1] MOD1: Compare match mode setting bit**

- This bit specifies the operation to be performed when a compare match is detected in the output compare output 1.
- When this bit is set to "0", the output value is inverted upon a compare match.
- When this bit is set to "1", the output value is set to "1" or reset to "0" upon a compare match. The switch between setting and resetting is performed according to the CMOD bit (common to ch.0 and ch.1) in the compare control register (OCS).

Value	Description
0	Invert the previous output value.
1	Set the output value to "1" or reset it to "0" according to the setting of the CMOD bit in the compare control register (OCS).

**Note:**

- Be sure to stop the compare operation before writing a value to this bit.

##### **[bit0] MOD0: Compare match mode setting bit**

- This bit specifies the operation to be performed when a compare match is detected in the output compare output 1.
- When this bit is set to "0", the output value is inverted upon a compare match.
- When this bit is set to "1", the output value is set to "1" or reset to "0" upon a compare match. The switch between setting and resetting is performed according to the CMOD bit (common to ch.0 and ch.1) in the compare control register (OCS).

Value	Description
0	Invert the previous output value.
1	Set the output value to "1" or reset it to "0" according to the setting of the CMOD bit in the compare control register (OCS).

**Note:**

- Be sure to stop the compare operation before writing a value to this bit.

## 4.5. Compare Control Clear Register (OCSC)

The compare control clear register (OCSC) is a register to clear a bit in the compare control register (OCS).

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved	BTSC1	BTSC0	CMODC	Reserved		OTDC1	OTDC0
ACCESS_TYPE	R0,W0	R0,W	R0,W	R0,W	R0,W0		R0,W	R0,W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	00		0	0

BITS_OFFSET	23	22	21	20	19	18	17	16
BITS_NAME	IOPC1	IOPC0	IOEC1	IOEC0	BUFC1	BUFC0	CSTC1	CSTC0
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

**[bit31] Reserved: Reserved bit**

**[bit30] BTSC1: Buffer transfer selection clear bit**

Value	Description
0	No effect
1	Clear the buffer transfer selection bit

- This bit is always "0" when read.
- Writing "1" to this bit clears the BTSC1 bit in the OCS register.

**[bit29] BTSC0: Buffer transfer selection clear bit**

Value	Description
0	No effect
1	Clear the buffer transfer selection bit

- This bit is always "0" when read.
- Writing "1" to this bit clears the BTSC0 bit in the OCS register.

**[bit28] CMODC: Output level inversion mode clear bit**

Value	Description
0	No effect
1	Clear the output level inversion mode bit

- This bit is always "0" when read.
- Writing "1" to this bit clears the CMOD bit in the OCS register.

**[bit27:26] Reserved: Reserved bits**



**[bit25] OTDC1: Buffer transfer selection clear bit**

Value	Description
0	No effect
1	Clear the buffer transfer selection bit

- This bit is always "0" when read.
- Writing "1" to this bit clears the OTD1 bit in the OCS register.

**[bit24] OTDC0: Buffer transfer selection clear bit**

Value	Description
0	No effect
1	Clear the buffer transfer selection bit

- This bit is always "0" when read.
- Writing "1" to this bit clears the OTD0 bit in the OCS register.

**[bit23] IOPC1: Compare match interrupt flag clear bit**

Value	Description
0	No effect
1	Clear the compare match interrupt flag bit

- This bit is always "0" when read.
- Writing "1" to this bit clears the IOP1 bit in the OCS register.

**[bit22] IOPC0: Compare match interrupt flag clear bit**

Value	Description
0	No effect
1	Clear the compare match interrupt flag bit

- This bit is always "0" when read.
- Writing "1" to this bit clears the IOP0 bit in the OCS register.

**[bit21] IOEC1: Compare match interrupt enable clear bit**

Value	Description
0	No effect
1	Clear the compare match interrupt enable bit

- This bit is always "0" when read.
- Writing "1" to this bit clears the IOE1 bit in the OCS register.

**[bit20] IOEC0: Compare match interrupt enable clear bit**

Value	Description
0	No effect
1	Clear the compare match interrupt enable bit

- This bit is always "0" when read.
- Writing "1" to this bit clears the IOE0 bit in the OCS register.

**[bit19] BUFC1: Compare buffer invalidating clear bit**

Value	Description
0	No effect
1	Clear the compare buffer invalidating bit

- This bit is always "0" when read.
- Writing "1" to this bit clears the BUF1 bit in the OCS register.

**[bit18] BUFC0: Compare buffer invalidating clear bit**

Value	Description
0	No effect
1	Clear the compare buffer invalidating bit

- This bit is always "0" when read.
- Writing "1" to this bit clears the BUF0 bit in the OCS register.

**[bit17] CSTC1: Compare operation enable clear bit**

Value	Description
0	No effect
1	Clear the compare operation enable bit

- This bit is always "0" when read.
- Writing "1" to this bit clears the CST1 bit in the OCS register.

**[bit16] CSTC0: Compare operation enable clear bit**

Value	Description
0	No effect
1	Clear the compare operation enable bit

- This bit is always "0" when read.
- Writing "1" to this bit clears the CST0 bit in the OCS register.



## 4.6. Compare Control Set Register (OCSS)

The compare control set register (OCSS) is a register to set a bit in the compare control register (OCS).

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved	BTSS1	BTSS0	CMODS	Reserved		OTDS1	OTDS0
ACCESS_TYPE	R0,W0	R0,W	R0,W	R0,W	R0,W0		R0,W	R0,W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	00		0	0

BITS_OFFSET	23	22	21	20	19	18	17	16
BITS_NAME	Reserved		IOES1	IOES0	BUFS1	BUFS0	CSTS1	CSTS0
ACCESS_TYPE	R0,W0		R0,W	R0,W	R0,W	R0,W	R0,W	R0,W
PROT_TYPE	-							
INITIAL_VALUE	00		0	0	0	0	0	0

**[bit31] Reserved: Reserved bit**

**[bit30] BTSS1: Buffer transfer selection set bit**

Value	Description
0	No effect
1	Set the buffer transfer selection bit

- This bit is always "0" when read.
- Writing "1" to this bit sets the BTS1 bit in the OCS register.

**[bit29] BTSS0: Buffer transfer selection set bit**

Value	Description
0	No effect
1	Set the buffer transfer selection bit

- This bit is always "0" when read.
- Writing "1" to this bit sets the BTS0 bit in the OCS register.

**[bit28] CMODS: Output level inversion mode set bit**

Value	Description
0	No effect
1	Set the output level inversion mode bit

- This bit is always "0" when read.
- Writing "1" to this bit sets the CMOD bit in the OCS register.

**[bit27:26] Reserved: Reserved bits**

**[bit25] OTDS1: Buffer transfer selection set bit**

Value	Description
0	No effect
1	Set the buffer transfer selection bit

- This bit is always "0" when read.
- Writing "1" to this bit sets the OTD1 bit in the OCS register.

**[bit24] OTDS0: Buffer transfer selection set bit**

Value	Description
0	No effect
1	Set the buffer transfer selection bit

- This bit is always "0" when read.
- Writing "1" to this bit sets the OTD0 bit in the OCS register.

**[bit23:22] Reserved: Reserved bits**

**[bit21] IOES1: Compare match interrupt enable set bit**

Value	Description
0	No effect
1	Set the compare match interrupt enable bit

- This bit is always "0" when read.
- Writing "1" to this bit sets the IOE1 bit in the OCS register.

**[bit20] IOES0: Compare match interrupt enable set bit**

Value	Description
0	No effect
1	Set the compare match interrupt enable bit

- This bit is always "0" when read.
- Writing "1" to this bit sets the IOE0 bit in the OCS register.

**[bit19] BUFS1: Compare buffer invalidating set bit**

Value	Description
0	No effect
1	Set the compare buffer invalidating bit

- This bit is always "0" when read.
- Writing "1" to this bit sets the BUF1 bit in the OCS register.

**[bit18] BUFS0: Compare buffer invalidating set bit**

Value	Description
0	No effect
1	Set the compare buffer invalidating bit

- This bit is always "0" when read.
- Writing "1" to this bit sets the BUF0 bit in the OCS register.



**[bit17] CSTS1: Compare operation enable set bit**

Value	Description
0	No effect
1	Set the compare operation enable bit

- This bit is always "0" when read.
- Writing "1" to this bit sets the CST1 bit in the OCS register.

**[bit16] CSTS0: Compare operation enable set bit**

Value	Description
0	No effect
1	Set the compare operation enable bit

- This bit is always "0" when read.
- Writing "1" to this bit sets the CST0 bit in the OCS register.

## 5. Precautions for Using

The following shows the notes when using the 16-bit output compare.

- If the settings are CMOD=1 and OCCP0=OCCP1, the port is inverted only once when a compare match occurs.
- Be sure to stop the compare operation before specifying the output level of the output compare output.
- An interrupt operation occurs independently for each of OCU0 to OCU11 when the compare mode bit CMOD is set to 1.

### Notes When Accessing a Register

#### When Accessing the Compare Control Register (OCS)

- This register supports writing from the bit-band alias area. For the bit-band alias area, see "CHAPTER: Bit-band Unit".
- To clear a specified bit in this register, clear the bit by writing "1" to the applicable bit in the compare control clear register (OCSC). It is prohibited to directly clear only a specific bit in this register.
- To set a specified bit in this register, set the bit by writing "1" to the applicable bit in the compare control set register (OCSS). It is prohibited to directly set only a specific bit in this register.
- Data can be written directly to this register only when writing to all bits.







## CHAPTER 50: 12-bit A/D Converter Interface

This chapter provides an overview and explains the register configuration and operation of the 12-bit A/D converter interface.

---

1. Overview
2. Configuration
3. Operation
4. Setting Procedure Example
5. Registers
6. Precautions for Using



## 1. Overview

The 12-bit A/D converter can convert analog input voltages to 12-bit digital values using the RC successive approximation conversion method. It begins A/D conversion when an A/D activation trigger is entered. When an A/D activation trigger is entered again during A/D conversion, another sequence of A/D conversion starts. The converter also supports forced stop due to an A/D conversion cancel input signal.

### Function of 12-bit A/D Converter Interface

The function of the converter is to convert the analog voltage (input voltage) input through analog input pins into digital values. It has the following features.

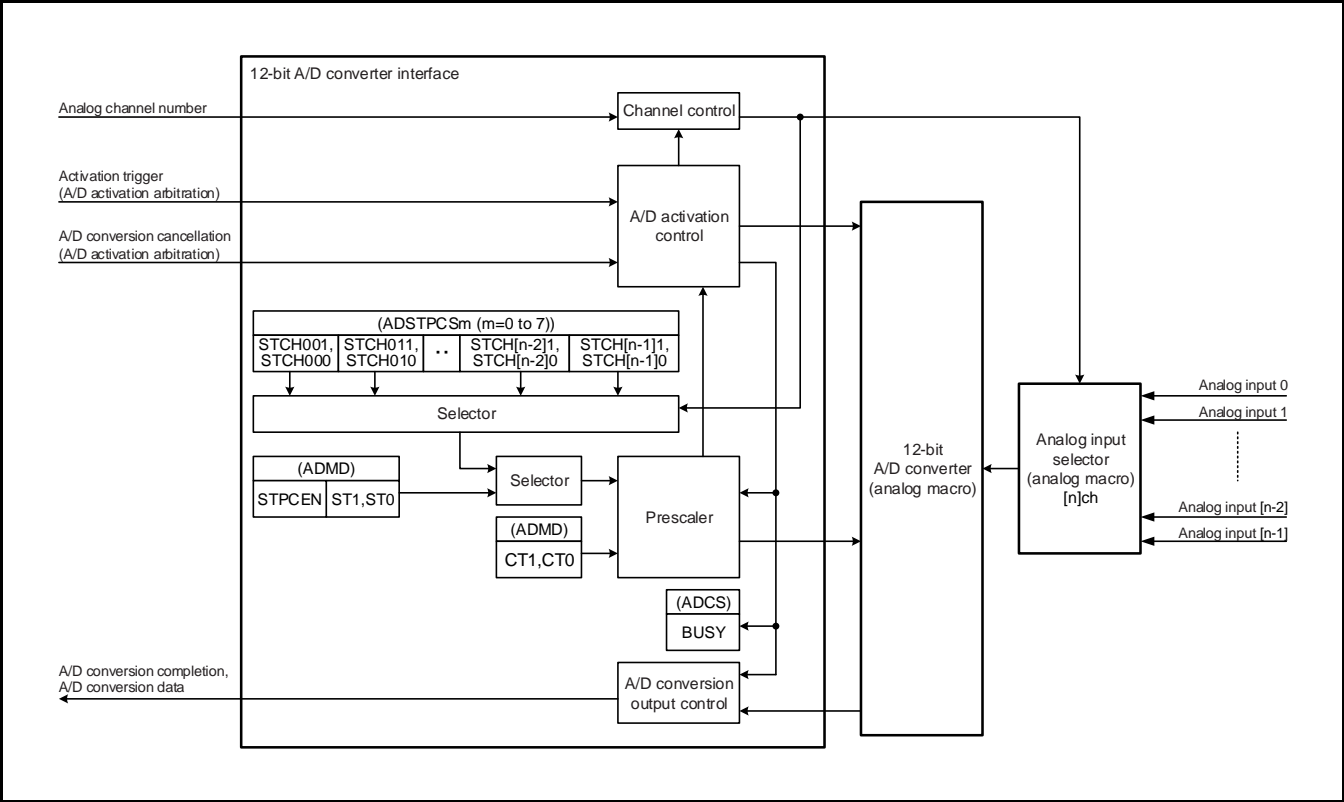
- The conversion method is the RC successive approximation method with a sample hold circuit.
- A program can select for the analog input pins. (This is configured from the A/D activation compare section.)
- Activation signals are input with pulse signals.
- The A/D conversion function converts at 1 time from the input of a 1-time activation factor.
- If an activation signal is input again during A/D conversion, the conversion is restarted. (Restart function)
- During A/D conversion, the current processing is stopped/initialized when an A/D conversion cancel signal is received. (Forced stop function)
- As for the setting of the sampling time, a common sampling time setting to all channels and the sampling time settings of each channel can be selected.



2. Configuration

This section shows a block diagram of the 12-bit A/D converter interface.

Figure 2-1 Configuration of A/D Converter Interface (n=32)





### 3. Operation

This section describes the operation of the 12-bit A/D converter interface.

#### 3.1. Operation of 12-bit A/D Converter Interface

The 12-bit A/D converter interface controls an A/D conversion.

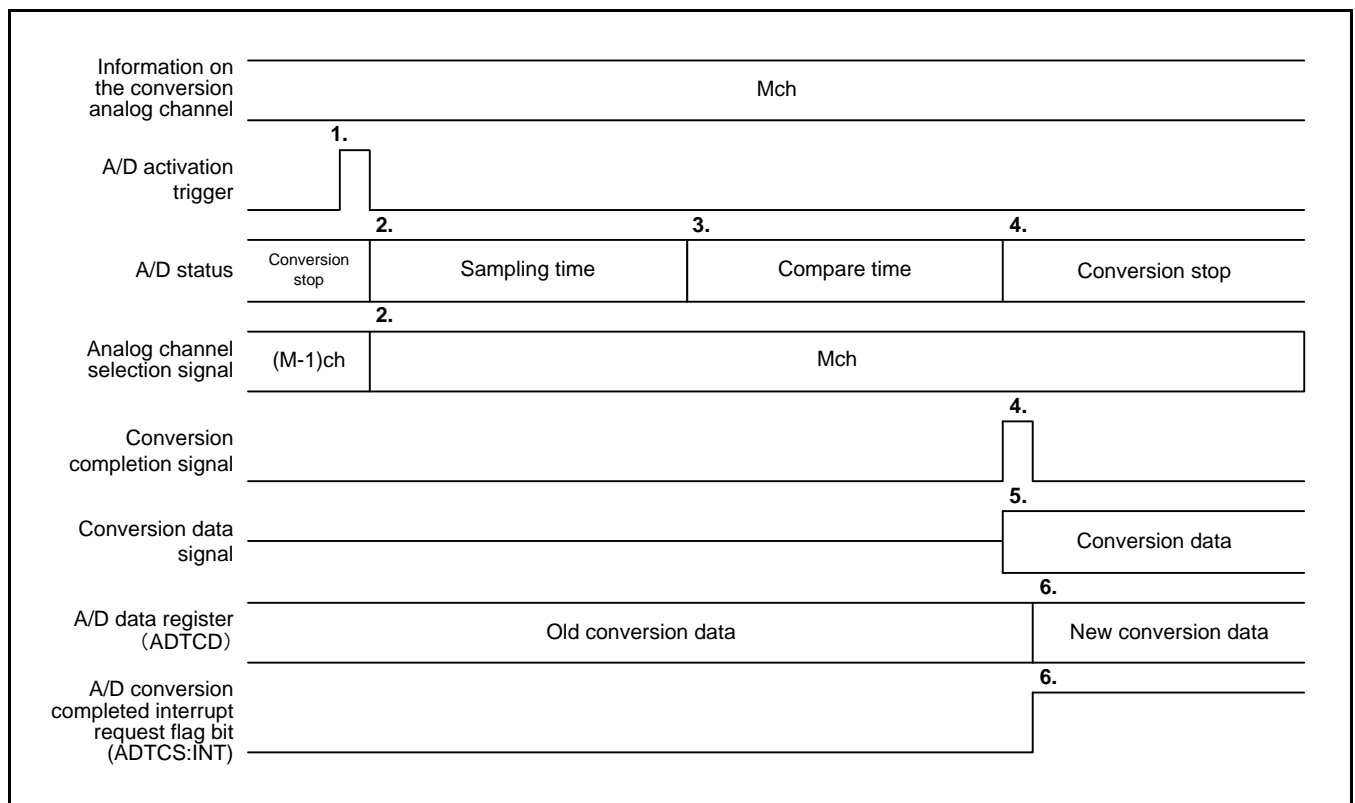
##### 3.1.1. Activation Factor

The activation factors for A/D conversion are activation trigger signals input with pulse signals.

##### 3.1.2. A/D Conversion

The A/D conversion function performs conversion 1 time from the input of a 1-time activation trigger.

**Figure 3-1 Operation Timing of A/D Converter Interface**

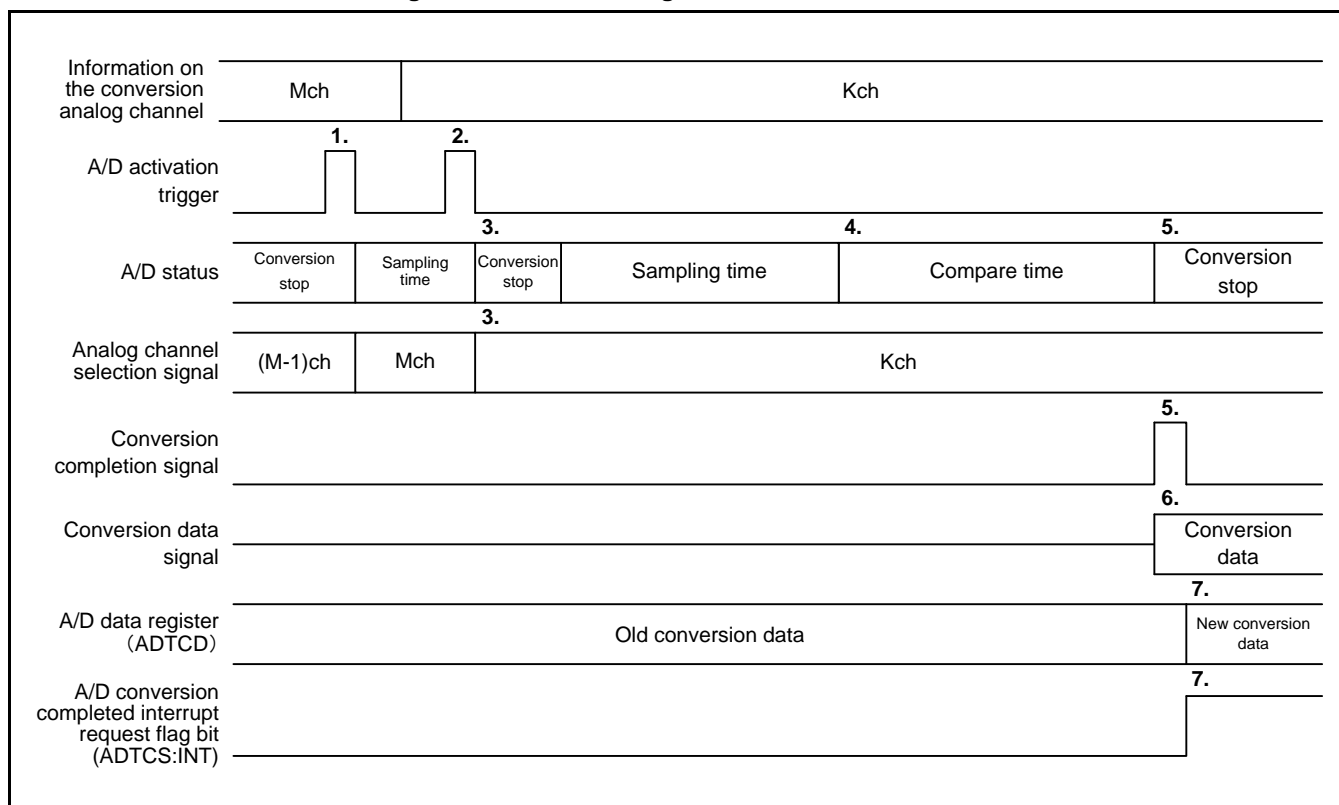


1. The pulse signal input of an A/D activation trigger signal starts A/D conversion.
2. The sampling starts by reception of the A/D activation trigger signal of 1. Then, an analog channel is held and the analog channel selection signal is outputted.
3. After the sampling time elapses, a compare operation begins.
4. After the compare time elapses, a conversion end signal is output.
5. A/D conversion data is output.
6. New conversion data is stored in an A/D data register (ADTCD) for A/D activation compare. Moreover, the A/D conversion completed interrupt request flag bit (ADTCS:INT) of A/D activation compare is set to "1".

### 3.1.3. Restart

If an activation trigger signal is input during A/D conversion, the current conversion is stopped/initialized, and the A/D conversion is restarted. Accordingly, the restart of A/D conversion begins several clocks (12-bit A/D converter clocks) later than in normal activation (A/D conversion beginning while A/D conversion is stopped).

Figure 3-2 Restart Timing of A/D Converter Interface



1. The pulse signal input of an A/D activation trigger signal starts A/D conversion.
2. Since an activation trigger signal is input during A/D conversion, the A/D conversion is restarted.
3. The A/D conversion of 1 is stopped/initialized by restart of A/D conversion of 2. Moreover, the analog channel selection signal is reacquired from the information on conversion analog channel and sampling is started newly.
4. After the sampling time elapses, a compare operation begins.
5. After the compare time elapses, a conversion end signal is output.
6. A/D conversion data is output.
7. New conversion data is stored in an A/D data register (ADTCD) of A/D activation compare. Moreover, the A/D conversion completed interrupt request flag bit (ADTCS:INT) of A/D activation compare is set to "1".

### 3.1.4. A/D Conversion Cancel

During A/D conversion, the current conversion is stopped/initialized when an A/D conversion cancel signal is received.

### 3.1.5. Analog Channel Selection

Information on the analog channel to be A/D converted is entered in addition to the activation trigger. The analog channel information, which is effective when the activation trigger is active, is used to select an analog channel.



### 3.1.6. A/D Conversion Time

The A/D conversion time is a combination of the sampling time and compare time.

#### (1) Sampling Time

The sampling time selection can be set to either each channel or to a common setting by using the sampling time setting per channel enable bit (ADMD:STPCEN).

- When ADMD:STPCEN="0", the sampling time common to all channels is enabled. The sampling time is set by the sampling time set bit (ADMD:ST[1:0]).
- When ADMD:STPCEN="1", the sampling time of each channel can be set. The sampling time of each channel is set by the sampling time setting per channel enable bit (ADSTPCS.STCHn1, STCHn0: n=00 to 31).

**Table 3-1 Sampling Time to Peripheral Clock Frequency**

ST[1:0] STCHn1, STCHn0	Function	Sampling Time(Peripheral Clock Frequency)			
		40MHz	32MHz	24MHz	16MHz
00	12 peripheral clock cycles	300ns	375ns	500ns	750ns
01	18 peripheral clock cycles	450ns	562.5ns	750ns	1125ns
10	24 peripheral clock cycles	600ns	750ns	1000ns	1500ns
11	48 peripheral clock cycles	1200ns	1500ns	2000ns	3000ns

**Notes:**

- Set a sampling time that is within the recommended values for the 12-bit A/D converter.
- A normal analog conversion value may not be obtained outside the recommended values.
- For details on the recommended values, see the data sheet.

#### (2) Compare Time

The compare time setting bits (ADMD:CT[1:0]) set the compare time.

**Table 3-2 Compare Time to Peripheral Clock Frequency**

CT[1:0]	Function	Compare Time(Peripheral Clock Frequency)			
		40MHz	32MHz	24MHz	16MHz
00	28 peripheral clock cycles	700ns	875ns	1166.7ns	1750ns
01	42 peripheral clock cycles	1050ns	1312.5ns	1750ns	2625ns
10	56 peripheral clock cycles	1400ns	1750ns	2333.4ns	3500ns
11	112 peripheral clock cycles	2800ns	3500ns	4666.7ns	7000ns

**Notes:**

- Set a compare time that is within the recommended values for the 12-bit A/D converter.
- A normal analog conversion value may not be obtained outside the recommended values.
- For details on the recommended values, see the data sheet.

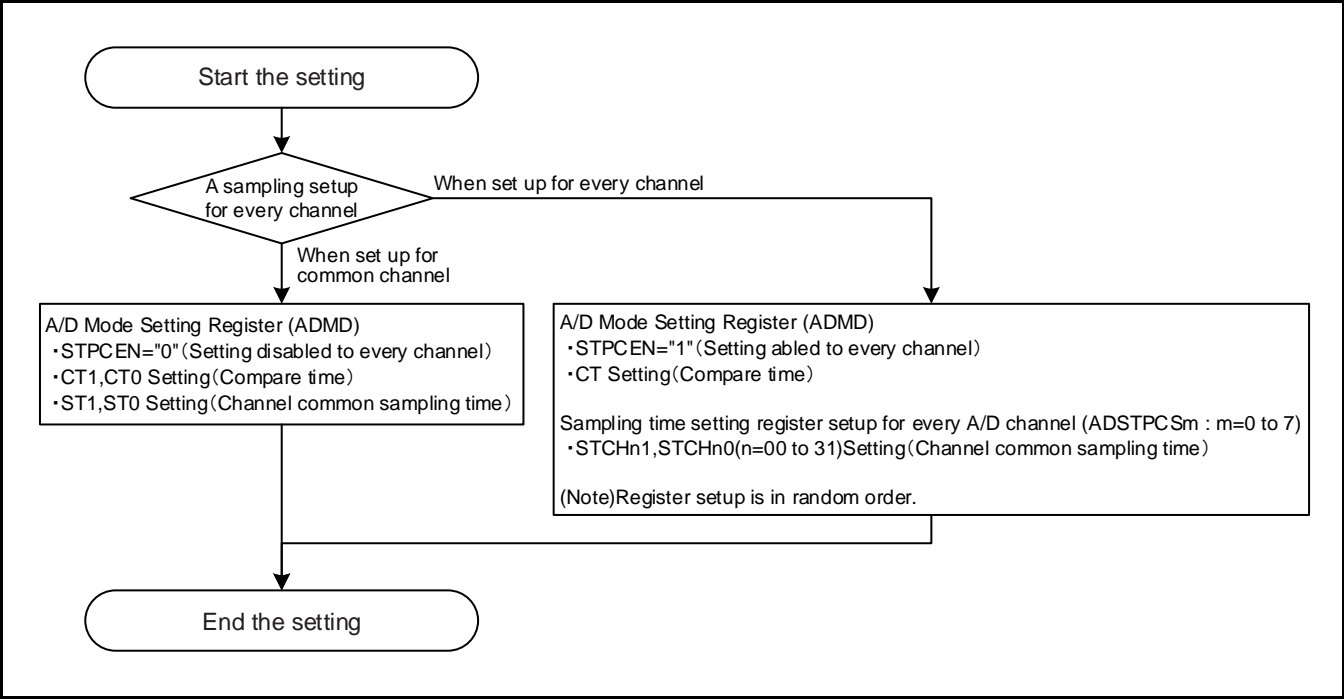
### 3.1.7. A/D Conversion End and A/D Data Fetch

Once A/D conversion ends normally without being restarted or canceled (the predetermined number of cycles has elapsed), the received conversion data is fetched and output. At this time, an A/D conversion end signal is generated.

4. Setting Procedure Example

This section shows an example of the 12-bit A/D converter interface setting procedure.

Figure 4-1 Setting Procedure Example of 12-bit A/D Converter Interface



- Note:**
- In a state in which the A/D conversion is stopped always (ADCS:BUSY="0"), set the operating mode of the 12-bit A/D converter interface.





## 5. Registers

This section describes registers of the 12-bit A/D converter interface. 12-bit A/D converter interface registers have prefix (ADC12B\_).

**Table 5-1 List of 12-bit A/D Converter Interface Registers**

Abbreviated Register Name	Register Name	See
ADCS	A/D Control Status Register	5.1
ADCH	A/D Channel Status Register	5.2
ADMD	A/D Mode Setting Register	5.3
ADSTPCSm (m=0 to 7)	A/D Sampling Time Setting Per Channel Register m	5.4

## 5.1. A/D Control Status Register (ADCS)

The A/D control status register (ADCS) indicates that A/D conversion is in operation or has stopped.

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	BUSY	Reserved						
ACCESS_TYPE	R,WX	R0,W0						
PROT_TYPE	-							
INITIAL_VALUE	0	0000000						

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved							
ACCESS_TYPE	R0,W0							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

### [bit15] BUSY: A/D conversion busy bit

Value	Description
0	A/D conversion has stopped.
1	A/D conversion is in operation.

- This bit indicates the operation of the A/D converter.
- As the value read from the A/D conversion busy bit (BUSY), "0" indicates that A/D conversion has stopped, and "1" indicates that A/D conversion is in operation.
- Writing to this bit does not change it and has no other effect.

### [bit14:0] Reserved: Reserved bits



## 5.2. A/D Channel Status Register (ADCH)

A/D channel status register (ADCH) enables, during A/D conversion, confirmation of the analog channel number for the conversion in progress.

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved			CH				
ACCESS_TYPE	R0,W0			R,WX				
PROT_TYPE	-							
INITIAL_VALUE	000			00000				

[bit7:5] Reserved: Reserved bits

[bit4:0] CH[4:0]: Analog channel bits

Value	Description
00000	ch.0
00001	ch.1
00110	ch.6
00111	ch.7
01000	ch.8
01110	ch.14
01111	ch.15
10000	ch.16
10001	ch.17
11110	ch.30
11111	ch.31

These bits can be used, during A/D conversion, to confirm the analog channel number for the conversion in progress.

### 5.3. A/D Mode Setting Register (ADMD)

The function of the A/D mode setting register (ADMD) is to set the A/D conversion compare time and sampling time.

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	STPCEN	Reserved			CT		ST	
ACCESS_TYPE	R/W	R0,W0			R/W		R/W	
PROT_TYPE	-							
INITIAL_VALUE	0	000			00		00	

#### [bit7]STPCEN: Sampling time setting per channel enable bit

Value	Description
0	Common settings of Sampling time on all channels
1	Individual settings of Sampling time in each channel

- This bit selects setting of each channel or setting common to all channels as the sampling time setting in the A/D conversion.
- When STPCEN="0", the sampling time common to all channels is enabled. The sampling time is set by the sampling time set bit (ADMD:ST[1:0]).
- When STPCEN="1", the sampling time of each channel is enabled. The sampling time of each channel is set by sampling time setting bit per A/D channel (ADSTPCS:STCHn1, STCHn0:n=00 to 31).

#### [bit6:4] Reserved: Reserved bits

#### [bit3:2] CT[1:0]: Compare time setting bits

Value	Description
00	28 peripheral clock cycles(A/D clock output: Peripheral clock/2)
01	42 peripheral clock cycles(A/D clock output: Peripheral clock/3)
10	56 peripheral clock cycles(A/D clock output: Peripheral clock/4)
11	112 peripheral clock cycles(A/D clock output: Peripheral clock/8)

- These bits select the compare time of A/D conversion.
- Analog input is captured. And when the time of the Compare time setting bits(ADMD:CT[1:0]) has elapsed and the resulting data is determined.

#### Notes:

- Set a compare time that is within the recommended values for the 12-bit A/D converter.
- A normal analog conversion value may not be obtained outside the recommended values.
- For details on the recommended values, see the data sheet.
- Be sure that the compare time setting bits (CT[1:0]) are rewritten before the conversion operation while A/D operations are stopped (ADCS:BUSY is "0").



**[bit1:0] ST[1:0]: Sampling time setting bits**

Value	Description
00	12 peripheral clock cycles(A/D clock output: Peripheral clock/2)
01	18 peripheral clock cycles(A/D clock output: Peripheral clock/3)
10	24 peripheral clock cycles(A/D clock output: Peripheral clock/4)
11	48 peripheral clock cycles(A/D clock output: Peripheral clock/8)

- These bits select the sampling time of A/D conversion.
- When the Sampling time setting per channel enable bit (ADMD:STPCEN) is set to "0", these bits are valid.
- When A/D conversion is started, during the setting time of Sampling time setting bits (ADMD:ST[1:0]), analog input is captured.

**Notes:**

- Set a sampling time that is within the recommended values for the 12-bit A/D converter.
- A normal analog conversion value may not be obtained outside the recommended values.
- For details on the recommended values, see the data sheet.
- Be sure that the sampling time setting bits (ST[1:0]) are rewritten before the conversion operation while A/D operations are stopped (ADCS:BUSY is "0").

## 5.4. A/D Sampling Time Setting Per Channel Register m(ADSTPCSm)(m=0 to 7)

The A/D sampling time setting per channel register (ADSTPCS) sets the sampling time of the A/D conversion for each channel.

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	STCH (m×4+3)1	STCH (m×4+3)0	STCH (m×4+2)1	STCH (m×4+2)0	STCH (m×4+1)1	STCH (m×4+1)0	STCH (m×4)1	STCH (m×4)0
ACCESS_TYPE	R/W		R/W		R/W		R/W	
PROT_TYPE	-							
INITIAL_VALUE	00		00		00		00	

### [bit7:0] STCHn1, STCHn0 (n=00 to 31): Sampling time setting per channel bits

STCHn1, STCHn0	Description
00	12 peripheral clock cycles(A/D clock output: Peripheral clock/2)
01	18 peripheral clock cycles(A/D clock output: Peripheral clock/3)
10	24 peripheral clock cycles(A/D clock output: Peripheral clock/4)
11	482 peripheral clock cycles(A/D clock output: Peripheral clock/8)

- These bits select the sampling time in the A/D conversion for each channel.
- If the sampling time setting permission bit (ADMD.STPCEN) is set to "1" for each channel, the sampling time setting (STCHn1, STCHn0) for each channel is enabled.
- The correspondence of a set channel and an analog channel is shown below.

**Table 5-2 Sampling Time Setting per Channel Enable and Analog Channel Number**

Sampling time setting per channel enable bit ADSTPCS0 to 7	Analog channel number
STCH001, STCH000	ch.0
STCH003, STCH002	ch.1
:	:
STCH001, STCH000	ch.6
STCH001, STCH000	ch.7
STCH001, STCH000	ch.8
:	:
STCH001, STCH000	ch.14
STCH001, STCH000	ch.15
:	:
STCH001, STCH000	ch.30
STCH001, STCH000	ch.31

#### Notes:

- Set a sampling time that is within the recommended values for the 12-bit A/D converter.
- A normal analog conversion value may not be obtained outside the recommended values.
- For details on the recommended values, see the data sheet.
- Be sure that the sampling time setting bits (ST[1:0]) are rewritten before the conversion operation while A/D operations are stopped (ADCS:BUSY is "0").



## 6. Precautions for Using

This section describes precautions on use of the 12-bit A/D converter interface.

### a) Peripheral Clock Frequency Limits

For details on peripheral clock frequency limits, see the data sheet.

### b) Sampling Time and Compare Time Settings

Configure ADMD:ST[1:0]/ADSTPCS:STCHn1, STCHn0(n=00 to 31) bits and ADMD:CT[1:0] bits such that the sampling time and compare time are within the recommended values for the 12-bit A/D converter.

### c) ADMD and ADSTPCS Setting

Rewrite the bits of A/D mode setting register (ADMD) and Sampling time setting per channel register (ADSTPCS) when the A/D operation has stopped before A/D conversion operation.



# CHAPTER 51: 12-bit A/D converter Activation Compare

This chapter explains A/D activation compare.

---

1. Overview
2. Configuration
3. Operation
4. Setting Procedure Example
5. Registers
6. Precautions for Using





## 1. Overview

The A/D activation compare controls 12-bit A/D converter with 32 channels.

### A/D Activation Compare Function

#### a) Analog Input Control

This function can enable/disable 32 channels of analog inputs.

#### b) Activation Channels

- It performs operation for A/D activation request control and A/D conversion data storage with each activation channel.
- Each activation channels is composed of the following register.
  - Compare buffer register (ADCOMPB)/Compare register (ADCOMP)
  - A/D activation trigger control status register (ADTCS)
  - A/D data register (ADTCD)
  - A/D activation trigger extended control register (ADTECS)
  - Range comparison control status register (ADRCCS)
  - Range comparison
  - Range comparison out-of-range flag register (ADRCOT)
  - Range comparison flag register (ADRCIF)
  - Activation Channel Conversion Count Setting Register (ADNCS)
  - Protected data state flag register (ADPRTF)
  - Activation Channel Conversion Completion Flag Register (ADEOCF)

#### c) A/D Activation Request

- Each activation channel issues an A/D activation request by one of the following methods. A given activation channel cannot reissue an A/D activation request during A/D conversion that was initiated by that channel.
  - Software
  - External trigger (falling edge)
  - Base timer (rising edge)
  - Compare match
- For software activation, any activation channel can be selected.
- The correspondence between the activation by base timer and by external trigger and 12-bit A/D converter units is as follows.
  - External trigger 0, Base timer channel 4: 12-bit A/D converter unit 0
- For compare match activation, the A/D activation is requested when the value in the 16 bits in the 16-bit free-run timer and the compare register of each activation channel match.
- In compare match activation, the A/D activation is requested when the free-run timer value and the compare register match during any one of the following.
  - 16-bit free-run timer up count only operation
  - 16-bit free-run timer down count only operation
  - 16-bit free-run timer up/down count operation
- A/D activation request classifies as following each activation channel.
  1. Software activation
  2. External trigger or base timer activation
  3. Compare match activation
- For each activation channel, either single mode or repeat mode can be specified as the activation request method.

- In single mode, one activation request is issued when one activation factor is encountered. One sequence of A/D conversion is performed and the activation request is reset when the A/D conversion completes.
- In repeat mode, activation requests are issued in succession as triggered by one activation factor. A/D conversion is performed repeatedly and the activation request continues its effect as long as repeat mode prevails.

#### **d) A/D Conversion Data**

- At the end of A/D conversion, the converted data is stored in an A/D data register. Each activation channel has an A/D data register.
- Each A/D data register contains an error flag bit and an error status bit, enabling checking of the status of A/D conversion data.

#### **e) Data Protection Function**

- The data protection function can be configured for each A/D data register. The data protection function runs for factors other than compare match activation.
- When enabled, the data protection function masks an A/D activation request until the data in the A/D data register is read and an interrupt flag is cleared. The reading of the data and the clearing of the interrupt flag happen in random order. Also, it is possible to select whether to include the clearing of the interrupt flag as a protection condition.
- The A/D activation request busy bit provides notification of an ongoing A/D conversion request or ongoing conversion. The current A/D conversion request or conversion can be forcibly terminated by the writing of "1" to the A/D activation request clear bit (ADTSC[n]:BUSYC) in the A/D activation request/interrupt clear register (ADTSC[n]).

#### **f) Scan Conversion with A/D Conversion Specified**

- The scan conversion that specifies the A/D conversion count of each activation channel can be executed.
- The scan conversion that specifies the A/D conversion count can set 1 type of each 12-bit A/D converter unit.
- The conversion count specification can select 1 to 4 times.
- The scan conversion that specifies the A/D conversion count can select the continuous scan conversion mode and the stop scan conversion mode.
- As for the continuous scan conversion mode, the specified activation channel is activated one by one. When the conversion of the final channel of the scan conversion is completed, the scan conversion is executed repeatedly from the top.
- As for the stop scan conversion mode, the specified activation channel is activated one by one. When the conversion of the final activation channel of the scan conversion is completed, the conversion stops.
- When the next activation factor is input, the scan conversion is executed from the top. However, the activation factor input converting the scan is disregarded.

#### **g) Range Comparison Function**

- The function can compare a range in each activation channel.
- Up to 4 types of settings are available for setting the upper- and lower-limit threshold values. An activation channel selects 1 combination from the 4 types of upper- and lower-limit threshold value settings to compare a range.
- A range comparison can select to confirm a result inside or outside the range of the upper- and lower-limit threshold values.
- Range comparison results enable noise reduction with the continuous detection function. The continuous detection function sets the range comparison flag based on a continuously detected range comparison result.
- A value of 1 to 7 times can be selected for the continuous detection frequency.
- The continuous detection count of a range comparison result can be checked.



- For confirmation of a result outside the range in a range comparison, the detection state can be checked to find out whether the result is above the upper-limit threshold value or below the lower-limit threshold value.

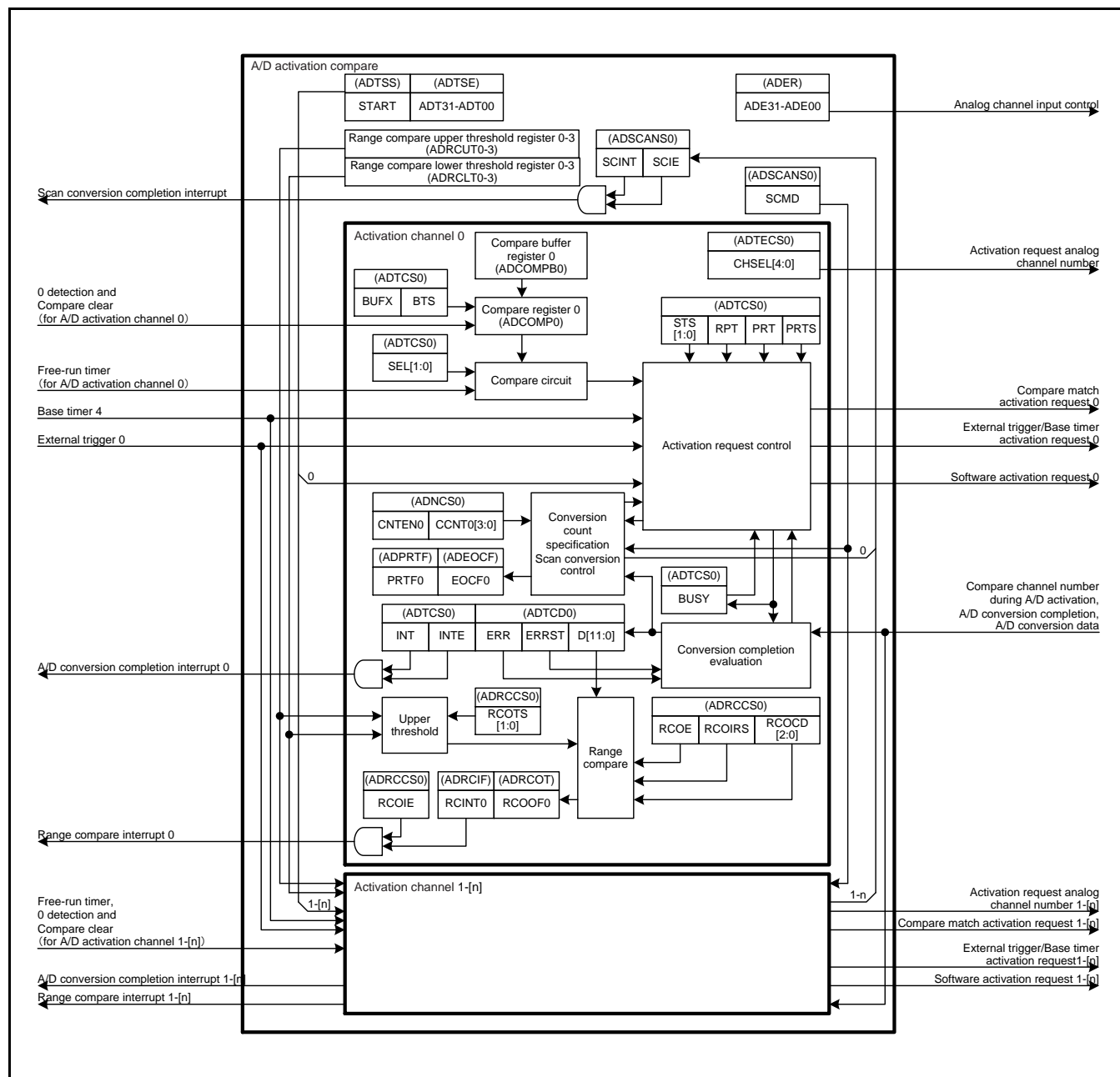
#### **h) Interrupt Request**

- Each activation channel can generate an interrupt request at the end of A/D conversion.
- Each activation channel can generate an interrupt request so that a range comparison is performed after A/D conversion ends.
- A/D activation compare can generate the interrupt request at scan conversion by the conversion count specification, when the A/D conversion of a specified count of the final activation channel is completed.

## 2. Configuration

This section shows a block diagram of A/D activation compare.

Figure 2-1 Block Diagram of A/D Activation Compare





### 3. Operation

This section describes the operation of A/D activation compare.

#### 3.1. A/D Activation Compare Interrupts

This section shows interrupt control bit and interrupt factor of A/D activation compare.

##### (1) A/D Conversion End Interrupt

**Table 3-1 Interrupt Control Bits and Interrupt Factor for A/D Conversion End Interrupt**

	A/D Conversion End Interrupt
Interrupt Request Flag Bit	INT:bit14 in A/D activation request/interrupt status register (ADTS[n])
Interrupt Request Enable Bit	INTE:bit13 in A/D activation trigger control register (ADTC[n])
Interrupt Factor	Conversion end signal for all channels

The A/D conversion completion interrupt request can be generated when the A/D conversion of compare channel that activates the A/D converter completes. Moreover, the A/D conversion completion interrupt can

be controlled in units of activation channel.

When the A/D conversion result is set in A/D data register (ADTCD), the INT bit of A/D activation trigger control status register (ADTCS) is set to "1". At this time, when the interrupt request enable bit has been enabled (ADTCS:INTE="1"), the interrupt request is output to the interrupt controller.

##### (2) Conversion Count-specified Scan Conversion Completion Interrupt

**Table 3-2 Interrupt Control Bits and Interrupt Factor When Conversion Count-specified Scan Conversion Ends**

	Range Compare Interrupt
Interrupt Request Flag Bit	RCINT[n]:bit[n] in range comparison flag register (ADRCIF)
Interrupt Request Enable Bit	RCOIE:bit3 in range comparison control status register (ADRCCS[n])
Interrupt Factor	After judgment by continuous detection function, based on executed range comparison

A/D activation compare can generate the interrupt request at scan conversion by the conversion count specification, when the A/D conversion of a specified count of the final activation channel is completed.

Moreover, the scan conversion completion interrupt by the conversion count specification is controlled in units of 12-bit A/D converter unit.

When the A/D conversion of a specified count of the final activation channel of the scan conversion by the conversion count specification is completed, scan conversion completion interrupt factor flag bit (ADSCANS:SCINT) is set to "1". At this time, when the scan conversion completion interrupt request enable bit has been enabled (ADSCANS:SCIE="1"), the interrupt request is output to the interrupt controller. Moreover, the scan conversion by the conversion count specification is controlled by conversion count specification scan conversion execution enable bit (ADNCS:CNTE) and conversion count specification bit (ADNCS:CCNT1, CCNT0).

### (3) Range Comparison Interrupt

**Table 3-3 Interrupt Control Bits and Interrupt Factor for Range Comparison Interrupt**

	Range Compare Interrupt
Interrupt Request Flag Bit	RCINT[n]:bit[n] in range comparison flag register (ADRCIF)
Interrupt Request Enable Bit	RCOIE:bit3 in range comparison control status register (ADRCCS[n])
Interrupt Factor	After judgment by continuous detection function, based on executed range comparison

(n=corresponding activation channel)

When the range comparison execution has been enabled, the A/D activation comparison performs a range comparison of the conversion result stored in the A/D data register (ADTCD) with one of the combinations within the upper bound threshold setting register 0 to 7 (ADRCUT0 to 7)/lower bound threshold setting register 0 to 7 (ADRCLT0 to 7) of the range comparison. When continuity can be confirmed as a result of comparing ranges, the range comparison interrupt request can be generated. Moreover, the range comparison interrupt can be controlled with each activation channel.

When the A/D conversion result is set in A/D data register (ADTCD) when the range comparison execution is enabled (ADRCCS:RCOE="1"), the range comparison is executed. The range comparison execution can select the range comparison condition by the inside/outside the range confirmation select bit (ADRCCS:RCOIRS).

- When confirming outside the range (ADRCCS:RCOIRS="0"), the range comparison condition is the following.
  - Lower bound threshold setting register > A/D conversion result  
or  
Upper bound threshold setting register < A/D conversion result
- When confirming inside the range (ADRCCS:RCOIRS="1"), the range comparison condition is the following.
  - Lower bound threshold setting register ≤ A/D conversion result  
and  
Upper bound threshold setting register ≥ A/D conversion result

When the range comparison condition is continuously detected, the range comparison interrupt factor flag bit (ADRCIF:RCINT) is set to "1". At this time, when the range comparison interrupt request enable bit has been enabled (ADRCCS:RCOIE="1"), the interrupt request is output to the interrupt controller. The timing of output by the range comparison interrupt request is delayed by two cycles with the peripheral clock than the A/D conversion completion interrupt. Moreover, the continuous detection count can be selected from 1 to 7 times by the continuous detection count specification bit (ADRCCS:RCOCD2 to RCOCD0).



## 3.2. A/D Activation Compare Operation

This section explains the A/D activation compare operation.

The A/D activation can be requested by either the software, the external trigger, the base timer or the compare match (When the value of 16-bit free-run timer reaches the instruction value).

### 3.2.1. A/D Activation

The activation request signal is generated to the A/D activation arbitration with either the software, the external trigger (falling), the base timer (rising), the compare match (When the free-run timer is corresponding to the compare register value). There are three A/D activation request signals, either "software activation request ", "external trigger/base timer activation request " or "compare match activation request " becomes active in each activation channel.

The activation request is cleared on completion of the A/D conversion of the corresponding channel and the conversion data is stored in the A/D data register. In that case, the interrupt can be generated.

Even if the activation factor is received in the activation request, the activation request is not reactivated in the activation channel.

### 3.2.2. A/D Activation Enable

The A/D activation factor is selected by A/D activation factor select bit (ADTCS:STS[1:0]). Either the software, the external trigger, the base timer or the compare match is selected. When the selected activation factor is generated, the A/D activation request signal is generated for the A/D activation arbitration.

For the activation channel that does not activate A/D, it is possible to disable the A/D activation request by selecting software activation (ADTCS:STS[1:0]="0b00") and disabling the software activation of a corresponding channel of A/D software activation channel select register (ADTSE).

### 3.2.3. Free-run Timer Input

The free-run timer input used for the compare match is input independently in each activation channel.

### 3.2.4. Analog Channel Selection

An analog channel that does the A/D conversion can be selected by the analog channel selection bits (ADTECS:CHSEL[4:0]).

### 3.2.5. Software Activation

The A/D activation factor selection bits are set for software activation (ADTCS:STS[1:0] is "0b00").

The channel that wants to activate software is set to the activation enable according to the A/D software activation channel select register (ADTSE). Two or more channels set to the ADTSE register can generate the activation request at the same time.

And, the software activation request signal is set by writing "1" in the START bit of the A/D software activation register (ADTSS).

### 3.2.6. External Trigger Activation

The A/D activation trigger control factor select bit is set to external trigger activation (ADTCS:STS[1:0]="0b01").

When the falling edge of the external trigger is detected, the activation request signal of external trigger/base timer is set.

### 3.2.7. Base Timer Activation

The A/D activation trigger control factor select bit is set to base timer activation (ADTCS:STS[1:0]="0b10"). When the rising edge of the base timer is detected, the activation request signal of external trigger/base timer is set.

**Note:**

- When activating with base timer, use 16/32-bit reload timer function.

### 3.2.8. Compare Match Activation

The A/D activation factor selection bits are set for compare match activation (ADTCS:STS[1:0] is "0b11").

A compare match activation request is set when a compare register (ADCOMP[n]) has a match between the 16-bit free-run timer and the compare register value (ADCOMP[n]:CMP[15:0]).

The preferred timer value for activation is set in the compare register (ADCOMP[n]). To make an A/D activation request at the same time as the 0 detection time of the 16-bit free-run timer, set "0x0000". To perform A/D activation at the same time as the compare clear time of the 16-bit free-run timer, set the same value as the compare clear value of the 16-bit free-run timer.

#### (1) Operation Mode of Compare Match Activation

The count direction selection bits (ADTCS:SEL[1:0]) control the operation mode of the compare match function.

The count direction selection bits (ADTCS:SEL[1:0]) select whether to compare the compare register value (ADCOMP:CMP[15:0]) and the 16-bit free-run timer when counting in either direction (up/down), only when counting up, or only when counting down. If the count direction selection bits (ADTCS:SEL[1:0]) are set to "0b11", no activation request signal is generated for A/D activation arbitration, even when the 16-bit free-run timer matches the compare register value (ADCOMP:CMP[15:0]).

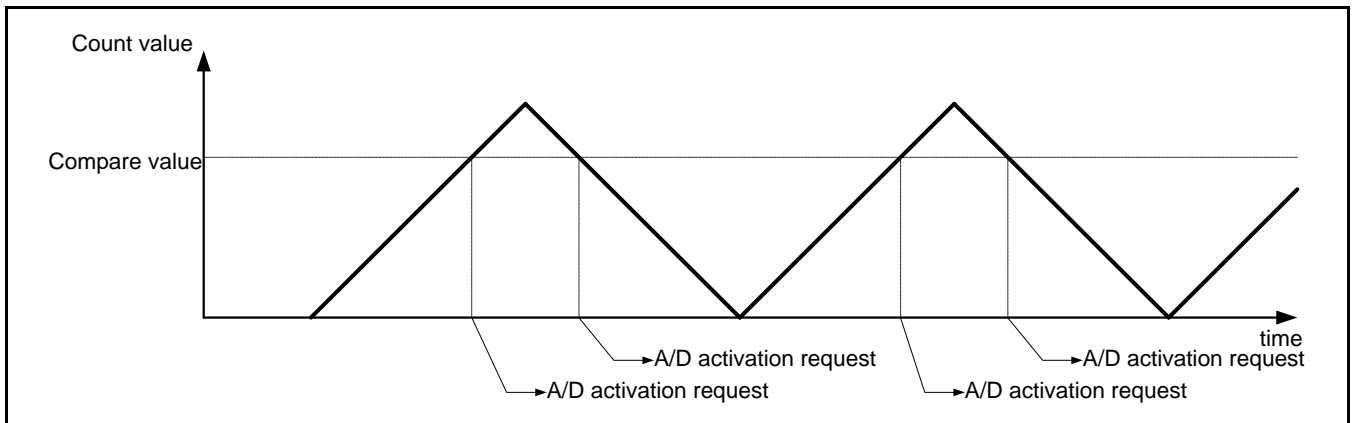
**Table 3-4 Control Details of Count Direction Selection Bits (ADTCS:SEL[1:0])**

ADTCS: SEL[1:0]	Function	Reference Drawing
Value		
00	When counting in either direction (up/down)	Figure 3-1
01	Only when counting up	Figure 3-2
10	Only when counting down	Figure 3-3
11	Compare disabled	-

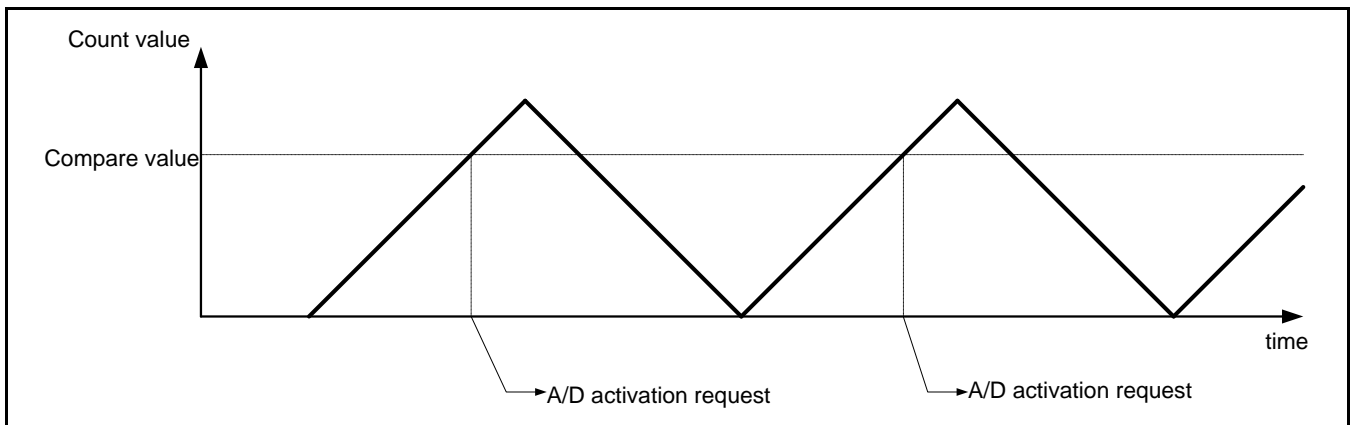




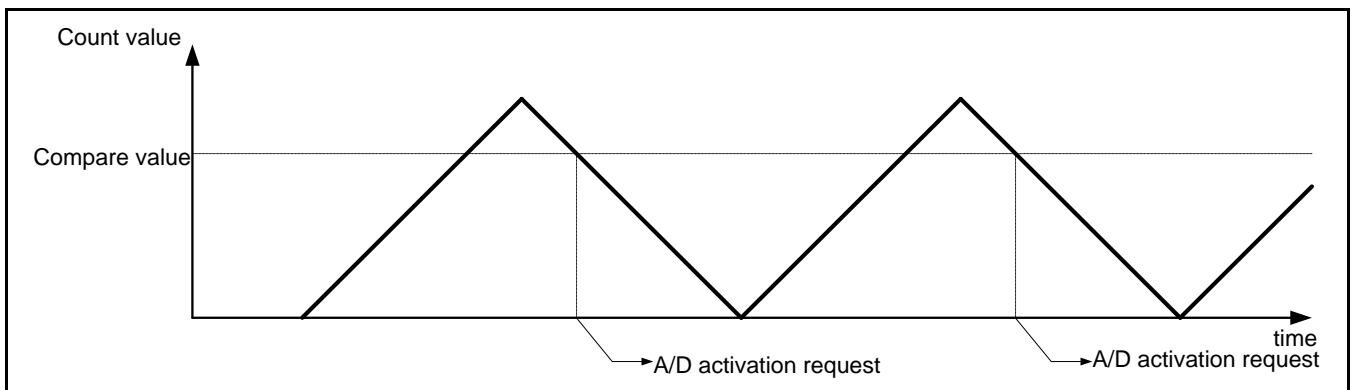
**Figure 3-1 SEL[1:0]="0b00": Compare Match Activation When both Counting Up and Down**



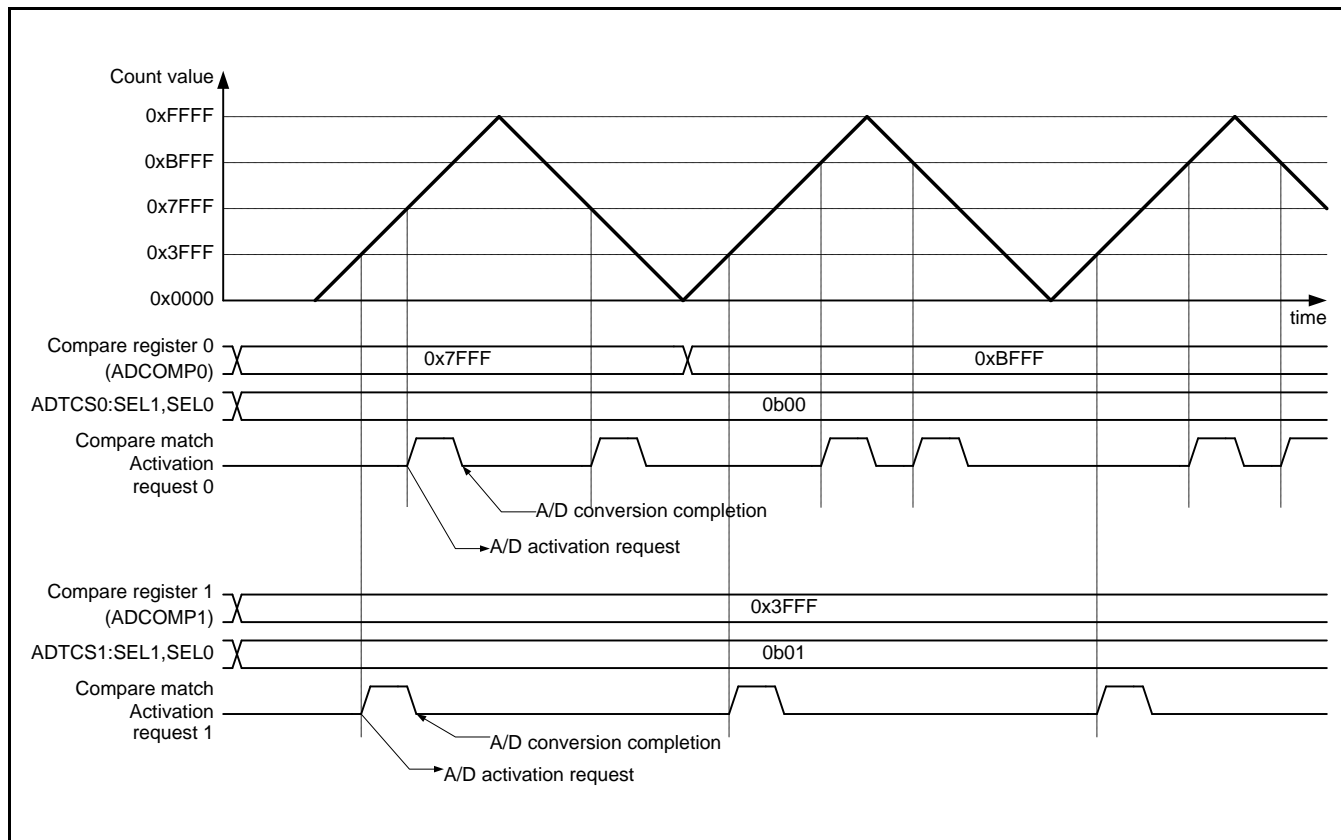
**Figure 3-2 SEL[1:0]=0b01: Compare Match Activation When only Counting Up**



**Figure 3-3 SEL[1:0]=0b01: Compare Match Activation When only Counting Up**



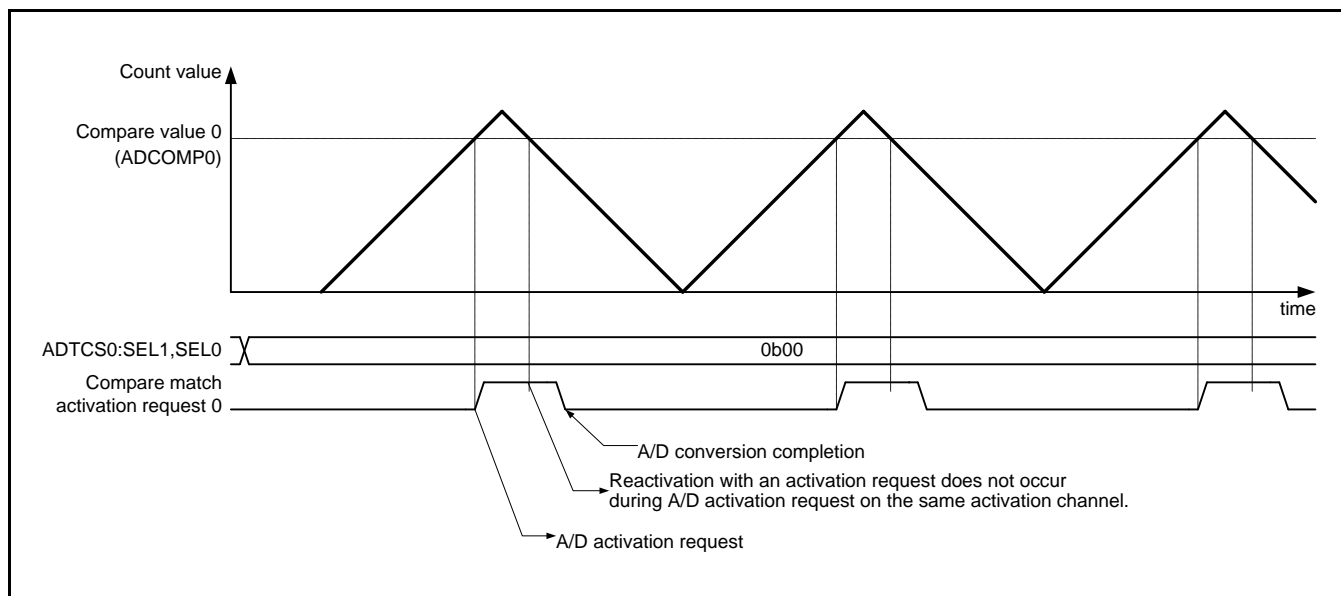
**Figure 3-4 Activation Channel 0: When both Counting Up and Down, Activation Channel 1: A/D Activation When only Counting Up**



## (2) About Compare Value Setting for Compare Match Activation

If the compare match generation interval is shorter than the A/D conversion time within the same activation channel for compare match activation, any compare match that occurs during A/D conversion is ignored.

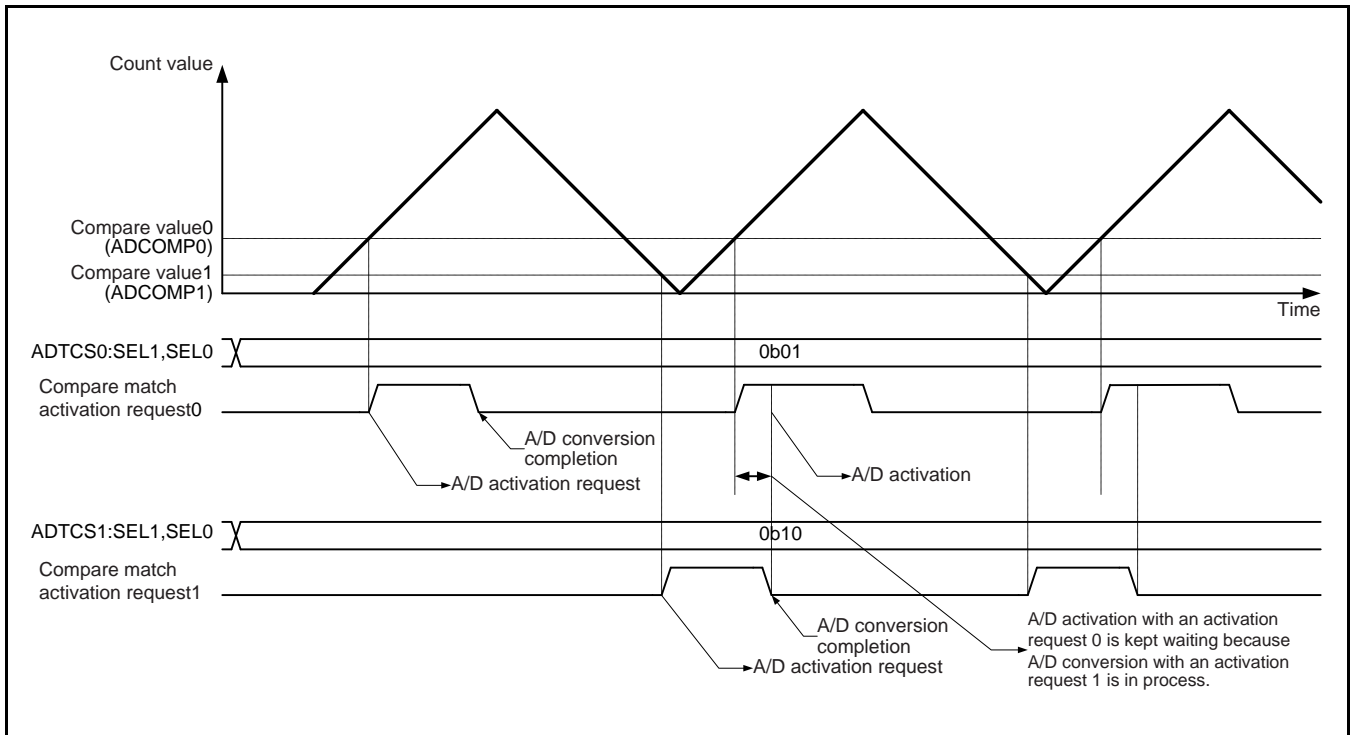
**Figure 3-5 When the Generation Interrupt the Compare Match is Shorter Than the Conversion Time of A/D in the Same Activation Channel**





Also, if the compare match generation interval is shorter than the A/D conversion time between activation channels that activate the same A/D converter unit, any activation request generated during A/D conversion delays the start of A/D conversion. A/D conversion does not begin at the intended time and the start of A/D conversion is delayed.

**Figure 3-6 When the Generation Interval of the Compare Match is the Shorter Than the Conversion Time of A/D in between the Activation Channels**



### (3) Compare Register Buffer Function for Compare Match Activation

The compare register buffer function control bit (ADTCS:BUFX) can select whether to enable use of the compare register buffer function. Writing "0" to the compare register buffer function control bit (ADTCS:BUFX) enables use of the compare register buffer function.

The timing of transfer from the buffer when the buffer function is enabled (ADTCS:BUFX is "0") can be selected by the compare register buffer transfer control bit (ADTCS:BTS). The value written in the compare buffer register (ADCOMPB) is transferred to the compare register (ADCOMP) at the following times: at the compare clear time if ADTCS:BTS is "1", and at the 0 detection time if ADTCS:BTS is "0".

**Table 3-5 Conditions for Transfer from Compare Buffer Register to Compare Register**

ADTCS		Condition for Transfer from Compare Buffer Register (ADCOMPB) to Compare Register (ADCOMP)
BUFX	BTS	
0	0	When 0 is detected in 16-bit free-run timer, or while 16-bit free-run timer is stopped
0	1	When compare clear is done on 16-bit free-run timer, or while 16-bit free-run timer is stopped
1	0 or 1	Buffer function not used (immediate transfer)



Figure 3-7 Compare Register 0: Buffer Function is Valid, Compare Register 1: Buffer Function is Invalid

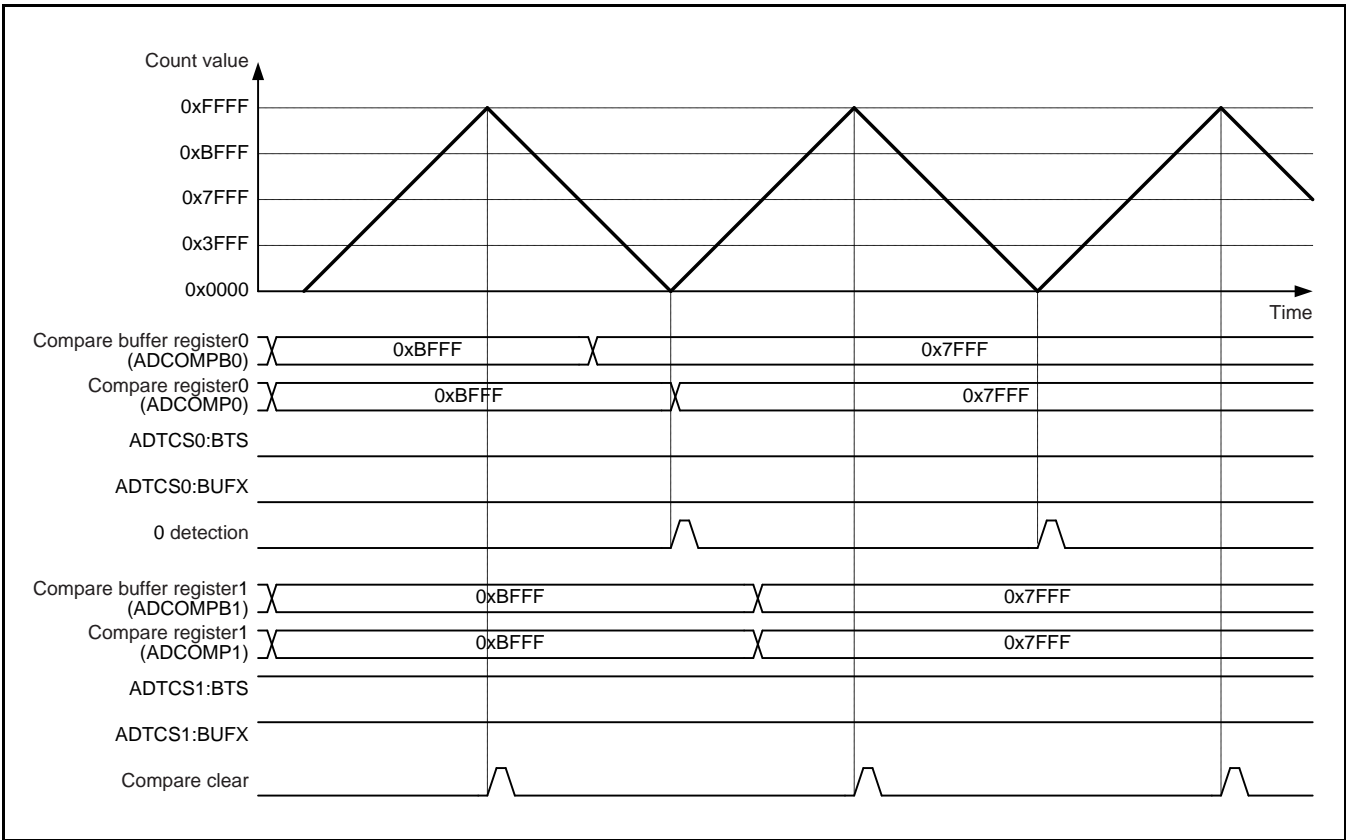


Figure 3-8 Timing of Compare Register Data Transfer as Compare Clear Register is Found to Match When 16-bit Free-run Timer is Counting Up

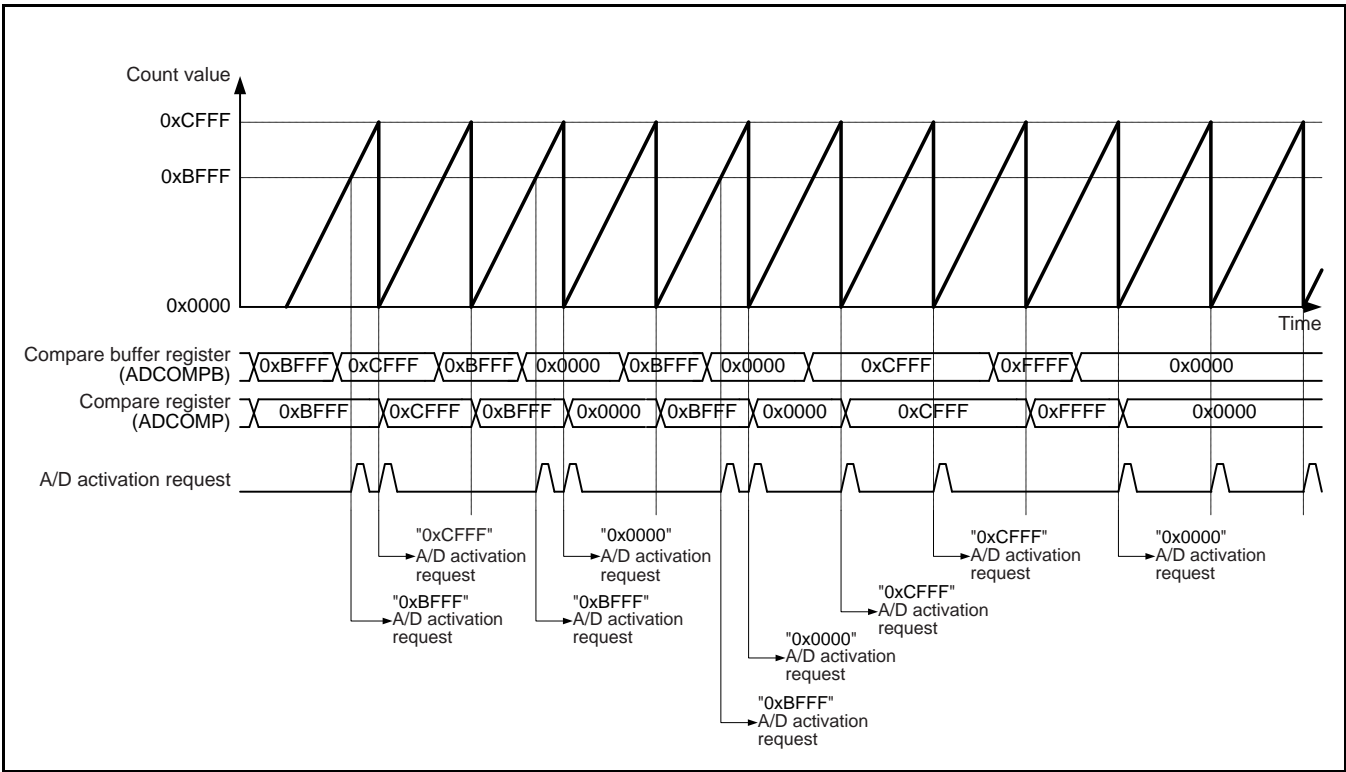




Figure 3-9 When 16-bit Free-run Timer Counting Up, the Compare Register Data Transfer Timing at 0 Detection

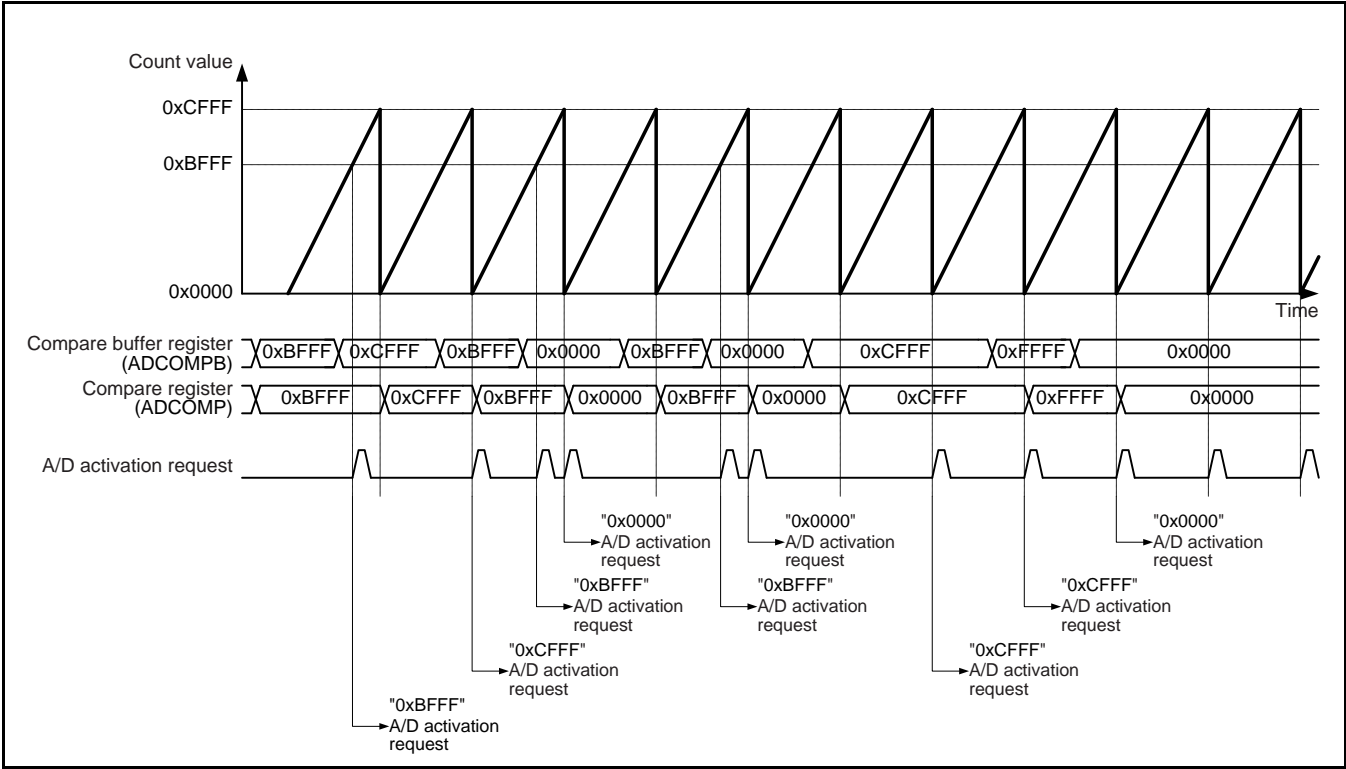
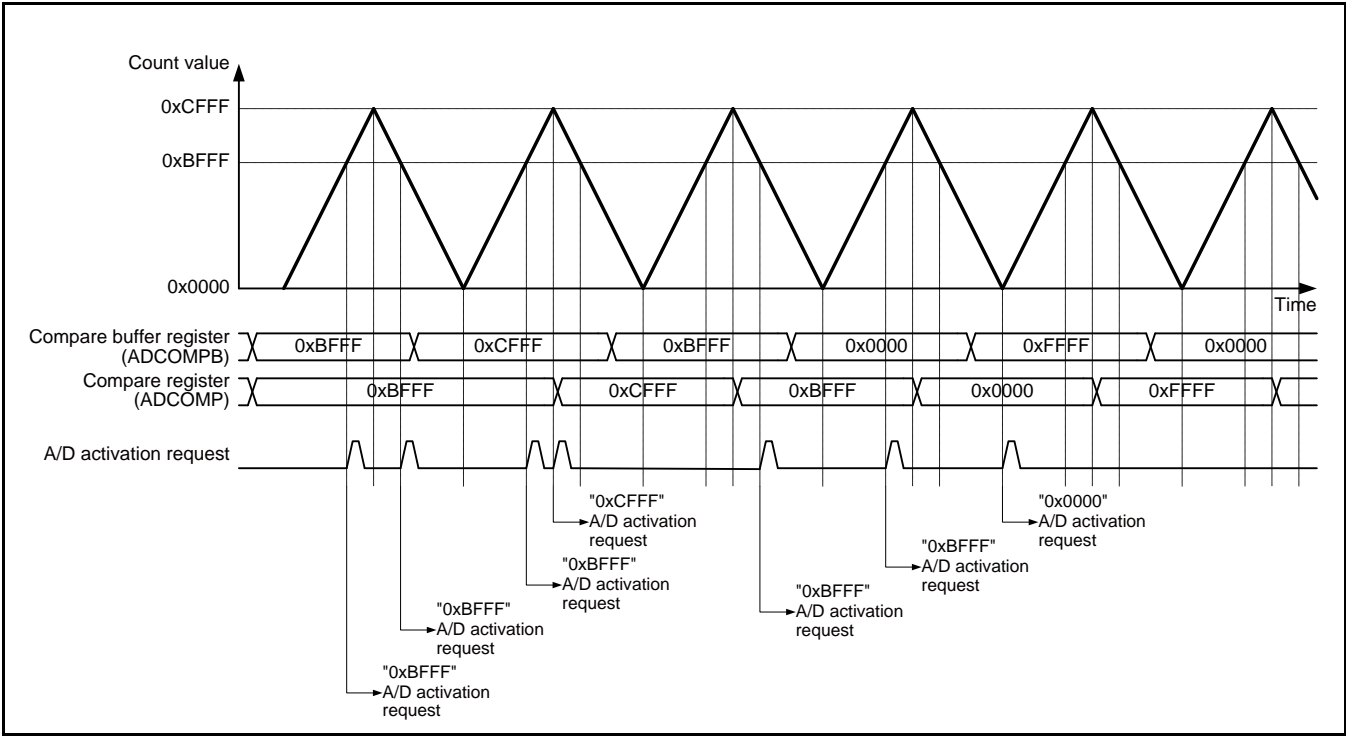
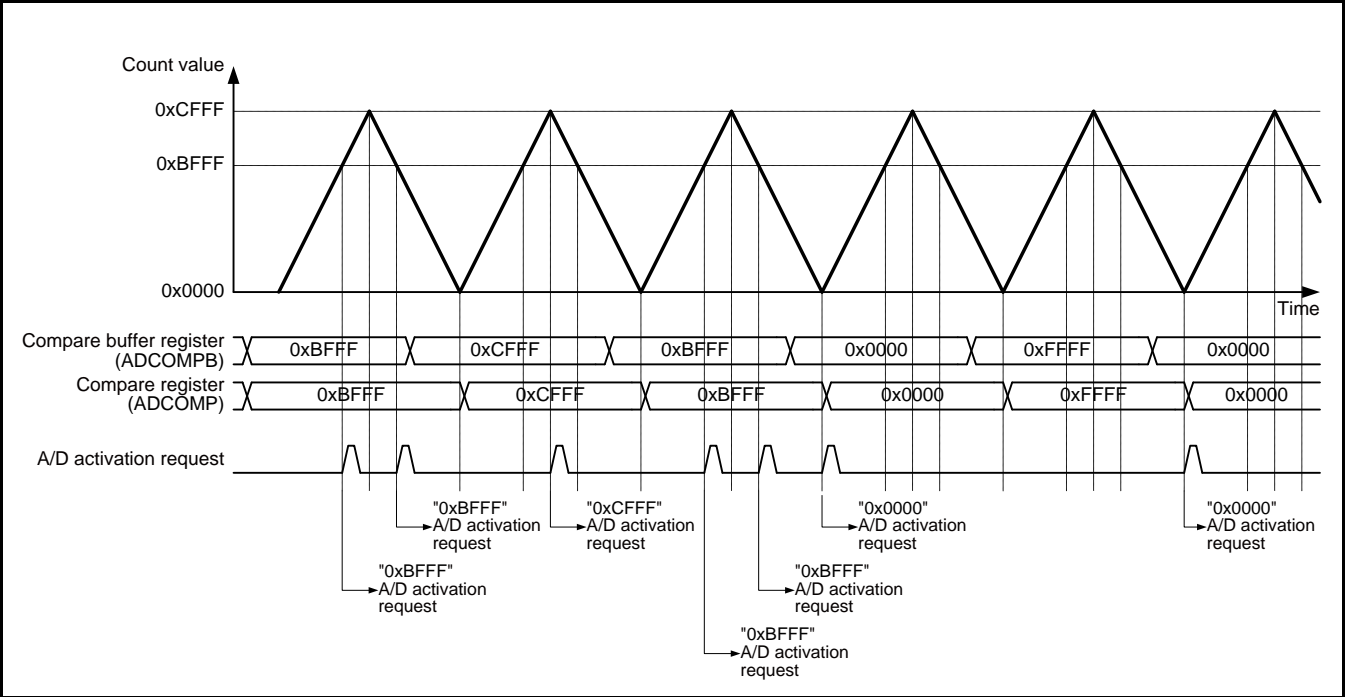


Figure 3-10 Timing of Compare Register Data Transfer as 0 is Detected When 16-bit Free-run Timer is Counting Up





**Figure 3-11 Timing of Compare Register Data Transfer as Compare Clear Register is Found to Match When 16-bit Free-run Timer is Counting Down**

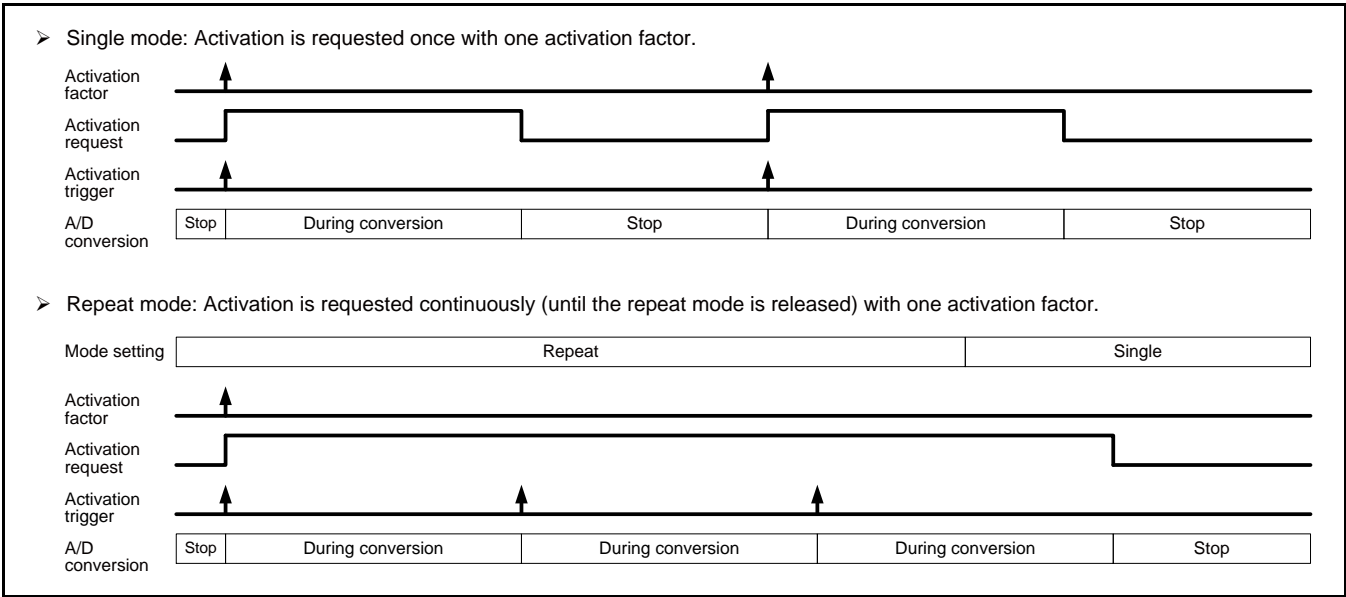




### 3.2.9. Activation Request Mode

- An activation request mode can be set for each activation channel. There are 2 activation request modes: single mode and repeat mode. The repeat conversion selection bit (ADTCS:RPT) sets the mode.
- In single mode (ADTCS:RPT is "0"), a 1-time activation factor causes a 1-time activation request. A/D conversion is performed 1 time, and the activation request is removed when the A/D conversion ends.
  - In repeat mode (ADTCS:RPT is "1"), a 1-time activation factor causes an activation request to be made continuously. A/D conversion is performed repeatedly, with the activation request being made continuously until repeat mode is canceled.

Figure 3-12 Activation Request Modes



### 3.2.10. A/D Conversion Data

The A/D conversion result data is stored in A/D data bit (ADTCD:D[11:0]) at each activation channel.

Moreover, when the data protection function is disabled (ADTCS:PRT="0"), the state of the A/D conversion data stored in A/D data bit (ADTCD:D[11:0]) can be confirmed by the conversion data error flag bit (ADTCD:ERR) and the conversion data error status bit (ADTCD:ERRST).

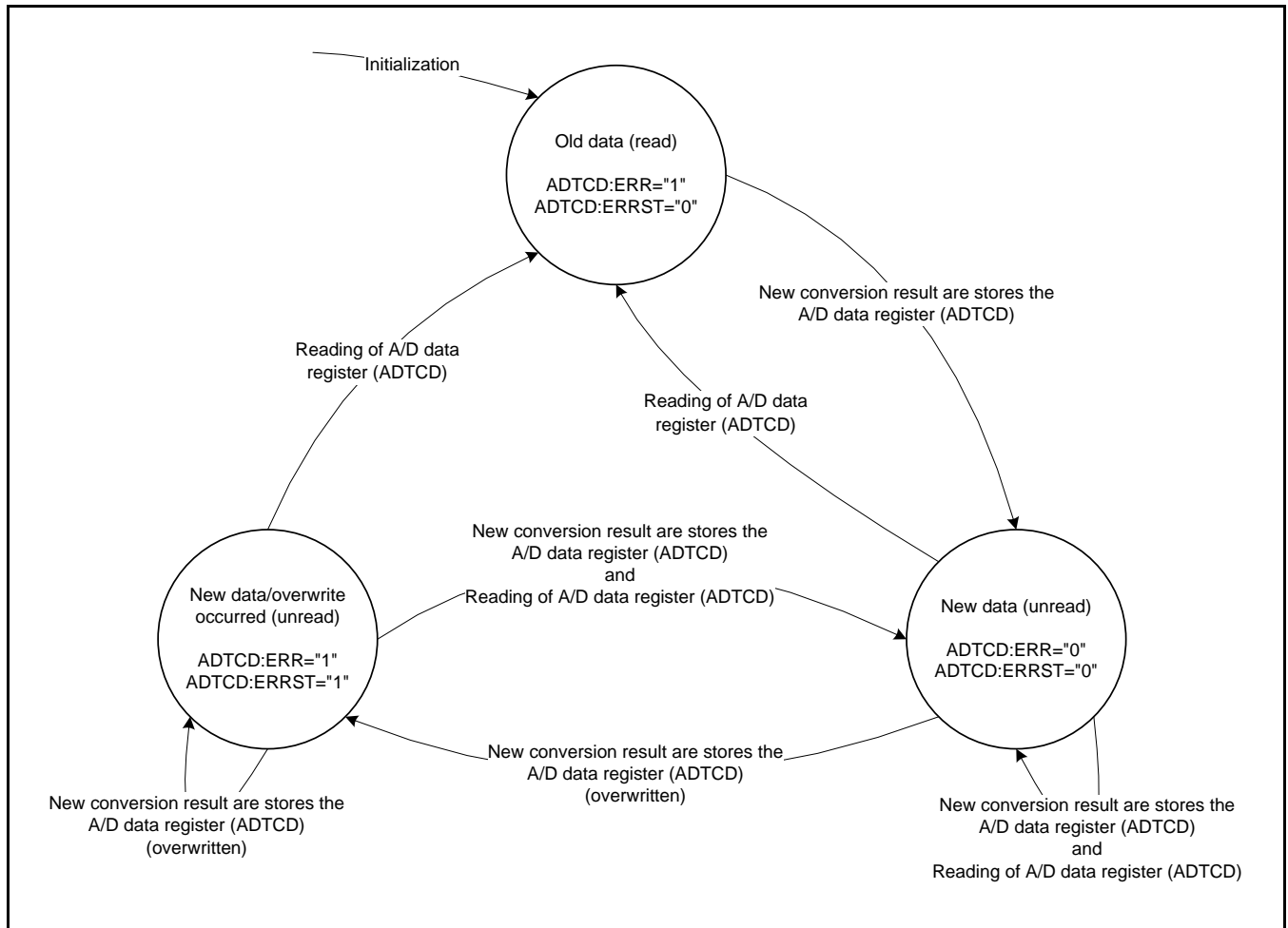
When the data protection function is enabled (ADTCS:PRT="1"), the conversion data error flag bit (ADTCD:ERR) and the conversion data error status bit (ADTCD:ERRST) are fixed to "0".

**Table 3-6 Checking of Status of A/D Conversion Data (When Data Protection Function is Disabled (ADTC[n]:PRT="0") or during Compare Match Activation (ADTCS:STS[1:0]="0b11"))**

ADTCD:ERR	ADTCD:ERRST	A/D Conversion Data Status
0	0	Latest data (unread)
0	1	- (no meaning)
1	0	Old data (read) Note: Initial value
1	1	Latest data/overwrite occurred (unread) Note: Data discarded

No combination with ADTCD:ERR and ADTCD:ERRST both being "0b01" is generated from the hardware.





### 3.2.11. Data Protection Function

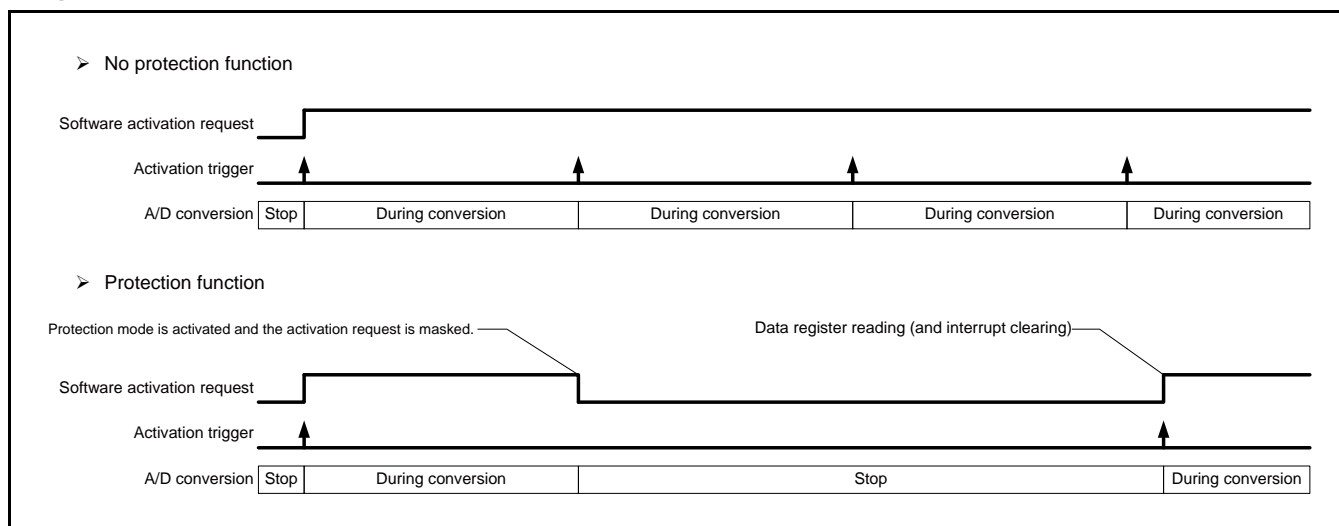
The data protection function can be configured for an A/D data register (ADTCD).

The data protection function is set from the A/D data register protection enable bit (ADTCS:PRT). The data protection function runs for factors other than compare match activation (ADTCS:STS[1:0] is "0b11").

Conversion results stored in an A/D data register (ADTCD) enter the data protected state when the data protection function is enabled (ADTCS:PRT is "1"). The A/D data register protection clear selection bit (ADTCS:PRTS) can select a condition for clearing the protected data state. While in the data protected state, unread data in the A/D data register (ADTCD[n]) is protected from being overwritten by subsequent A/D conversion data. The protection is provided by masking (making inactive) of activation request signals generated by subsequent activation factors.

- If the A/D data register protection clear selection bit (ADTCS:PRTS) is "0", activation requests are masked until the data in the A/D data register (ADTCD[n]) is read and the interrupt flag (ADTS[n]:INT) is cleared. The reading of the data and the clearing of the interrupt flag happen in random order.
- If the A/D data register protection clear selection bit (ADTCS:PRTS) is "1", activation requests are masked until the data in the A/D data register (ADTCD[n]) is read.

**Figure 3-14 Data Protection Function (Example of Software Activation Request in Repeat Mode (ADTCS:RPT="1"))**



### 3.2.12. Scan Conversion Mode

The scan conversion is operation that does the A/D conversion from the activation channel number with a small activation channel one by one. The scan conversion can set 1 type of each 12-bit A/D converter unit.

The scan conversion can be operated as follows.

- Single scan conversion
- Continuous scan conversion
- Continuous scan conversion when conversion count of each channel is specified
- Stop scan conversion when conversion count of each channel is specified

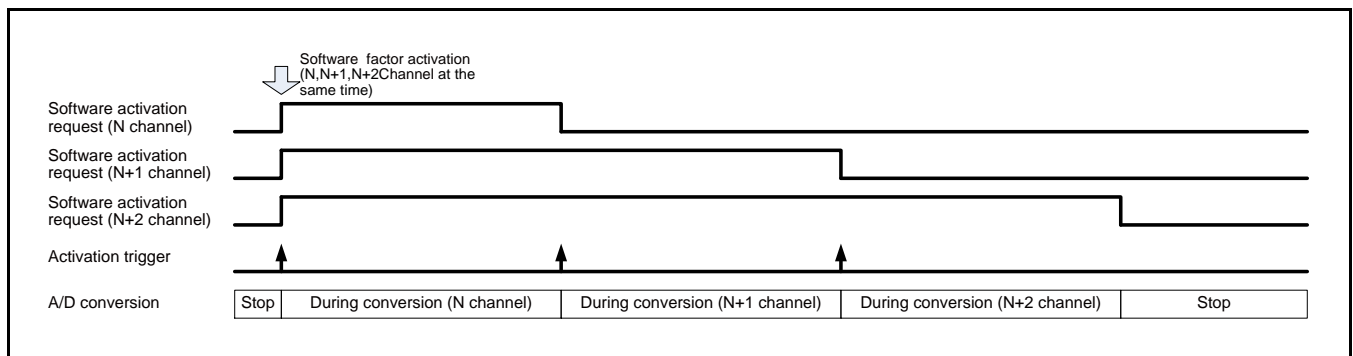
#### (1) Single Scan Conversion

Single scan conversion is executed by setting the repeat conversion selection bit to single mode

(ADTCS:RPT="0") for each activation channel of the scan conversion target, and executing A/D activation with the same activation factor at the same time for these channels.

The scan conversion starts A/D conversion from a small activation channel number (activation channel from which A/D is activated) one by one by the A/D activation arbitration of latter part. When the A/D conversion of the final activation channel is completed, the scan conversion is stopped.

Figure 3-15 Single Scan Conversion



#### (2) Continuous Scan Conversion

Continuous scan conversion is executed by setting the repeat conversion select bit to repeat mode

(ADTCS:RPT="1") and setting the data protection function to enable (ADTCS:PRT="1") for each activation channel of the scan conversion target, and executing the A/D activation with the same activation factor at the same time for these channels.

When two or more repeat modes (ADTCS:RPT="1") are set between the activation channels corresponding to the same A/D converter, only a certain channel (channel with the highest priority level) will be processed in the A/D activation arbitration of latter part.

In this case, please make the protection function effective about those activation channels.

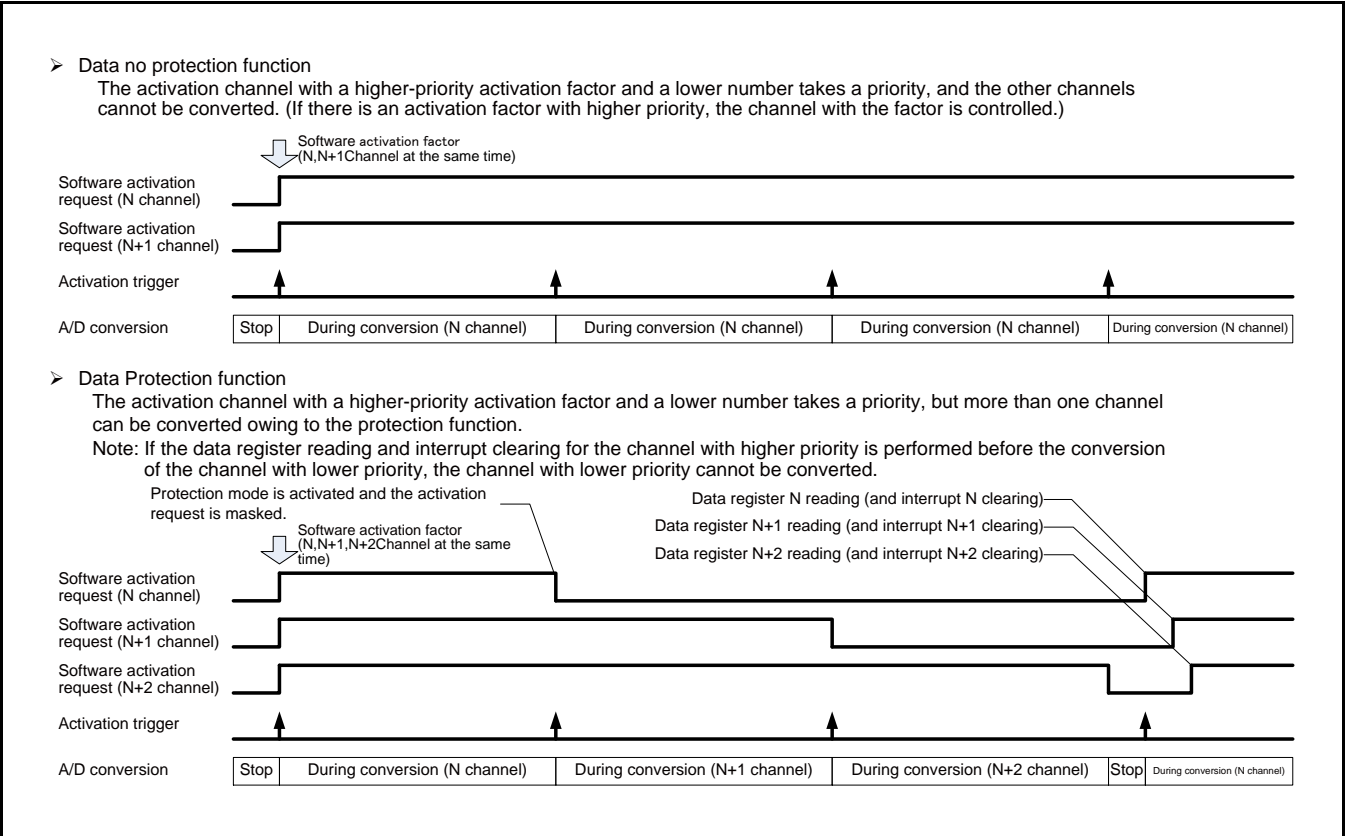
- In case of data protection function invalid (ADTCS:PRT="0")
- In A/D activation arbitration, channels with high-priority activation factors and with small activation channel numbers take priority. When the repeat mode is set (ADTCS:RPT="1"), the activation request is continuously output. Therefore, the conversion of the activation channel with low priority is not executed.
- In case of data protection function effective (ADTCS:PRT="1")
- In A/D activation arbitration, channels with high-priority activation factors and with small activation channel numbers take priority. When the repeat mode is set (ADTCS:RPT="1"), the activation request is continuously output. However, it enters the data protection state if the A/D conversion result is



stored in A/D data register (ADTCD), and the activation request is masked. As a result, the conversion of the next activation channel can be executed.

- Note:**
- When the data protection function is effective (ADTCS:PRT="1") and two or more repeat modes are set (ADTCS:RPT="1"), the conversion of the activation channel with low priority is not executed if the data protection state of the activation channel with high priority is released before converting the activation channel with low priority.

Figure 3-16 Continuous Scan Conversion





### (3) Continuous Scan Conversion When Conversion Count of Each Channel is Specified

Continuous scan conversion (when the conversion count is specified for each continuous scan conversion channel when the conversion count is specified for each channel) is executed as follows: set continuous scan conversion mode (ADSCANS:SCMD="0"); for each activation channel of the scan conversion target, set repeat conversion selection bit to repeat mode (ADTCS:RPT="1"), set conversion count specification scan conversion execution enable (ADNCS:CNTEN="1"), and set conversion count specification (ADNCS:CCNT1,CCNT0="count"); then, perform A/D activation with the same activation factor and at the same time for each channel.

The continuous scan conversion when conversion count of each channel is specified advances to the A/D conversion of the next activation channel when the A/D conversion of conversion count specification (ADNCS:CCNT1,CCNT0) is executed from activation channel (activation channel of the scan conversion target) with small number, and conversion is completed. Moreover, the activation channel that completes the A/D conversion of the conversion count specification can be confirmed by conversion count completion flag bit (ADEOCF:EOCF).

Then, when the A/D conversion of the conversion count specification of the final activation channel is completed, the scan conversion completion interrupt factor flag (ADSCANS:SCINT) is set to "1" and the scan conversion is automatically executed repeatedly from the top.

Moreover, when the scan conversion is executed repeatedly from the top, the conversion count completion flag bit (ADEOCF:EOCF) of each activation channel is automatically cleared to "0".

The conversion count specification (ADNCS:CCNT1,CCNT0) can select 1 to 4 times for each activation channel.

Disable and enable can be selected for the data protection function for each activation channel. Moreover, when the data protection function is effective (ADTCS:PRT="1"), the data protection state of each activation channel can be confirmed by data protection state flag bit (ADPRTF:PRTF).

- In case of data protection function invalid (ADTCS:PRT="0")
- The A/D conversion is continuously executed without stop. The A/D conversion data is not protected, and, the last A/D conversion data converted in the activation channel is stored.
- In case of data protection function effective (ADTCS:PRT="1")
- The A/D conversion data is protected. Therefore, the A/D conversion in the same activation channel stops the A/D conversion for the data protection period. However, when the scan conversion shifts to a different activation channel, the A/D conversion is continuously executed without stopping because A/D data register (ADTCD) is also different.

#### Notes:

- In continuous scan conversion, the conversion count completion flag (ADEOCF:EOCF) of the last activation channel of the scan conversion target is set to "1", but is soon automatically cleared to "0".
- When an activation factor that is the same as the scan conversion target is set to an activation channel with a number larger than that of the final channel of continuous scan conversion, the activation channel with the number larger than that of the final channel is evaluated once by the A/D activation arbitration of the latter part after completion of the final channel of the continuous scan conversion.
- The data protection function works at factors other than compare match activation (ADTCS:STS[1:0]="0b11").
- In the data protection function is effective (ADTCS:PRT="1"), please release the data protection state before executing the next scan conversion repeatedly. The activation channel in the data protection state does not execute the A/D conversion.

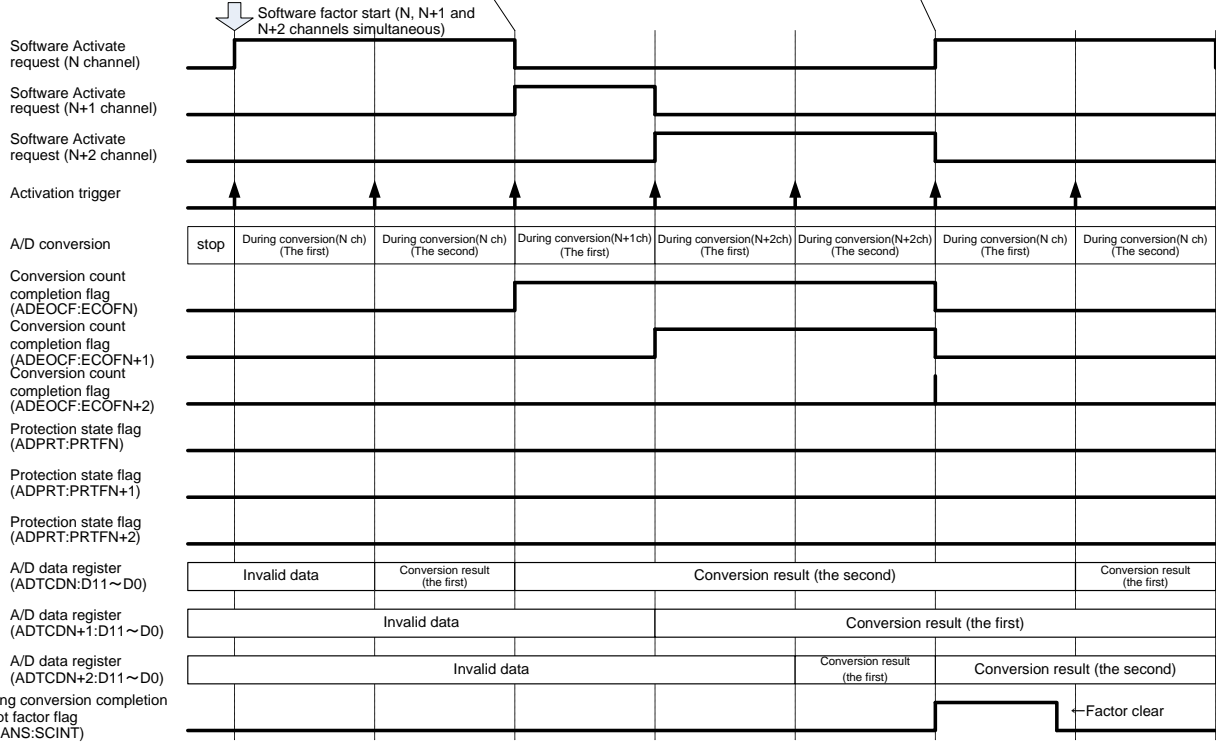
**Figure 3-17 Continuous Scan Conversion When Conversion Count of Each Channel is Specified (In Case of Data Protection Function Invalid)**

[Channel configuration information]

Activation channel N, N+1, and N+2: Repetition mode, no protection function, software activation, continuous scanning conversion mode, and conversion count specification scanning conversion execution enable continuous conversion count  
: Activation channel N+2 = twice activation channel N +1 = once activation channel N = twice

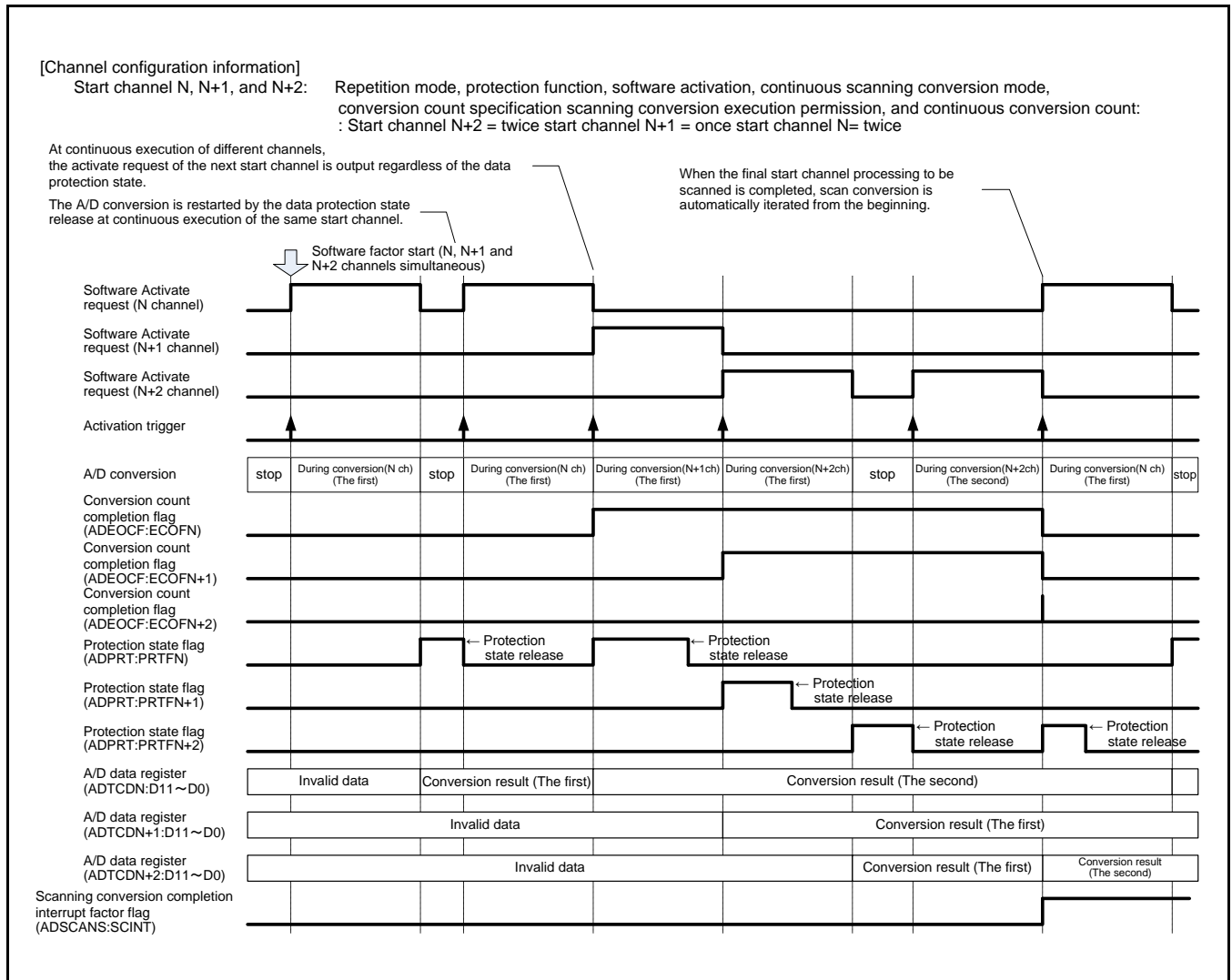
When the analog to digital conversion of a specified count is completed, the activation request is masked, and an activation request for the next activation channel is output.

When the final activation channel processing to be scanned is completed, scan conversion is automatically iterated from the beginning.





**Figure 3-18 Continuous Scan Conversion When Conversion Count of Each Channel is Specified (In Case of Data Protection Function Effective)**



#### (4) Stop Scan Conversion When Conversion Count of Each Channel is Specified

Stop scan conversion (when the conversion count is specified for each channel) is executed as follows: set stop scan conversion mode (ADSCANS:SCMD="1"); for each activation channel of the scan conversion target, set repeat conversion selection bit to repeat mode (ADTCS:RPT="1"), set conversion count specification scan conversion execution enable (ADNCS:CNTEN="1"), and set conversion count specification (ADNCS:CCNT1,CCNT0="count"); then, perform A/D activation with the same activation factor and at the same time for each channel.

The stop scan conversion when conversion count of each channel is specified advances to the A/D conversion of the next activation channel when the A/D conversion of conversion count specification (ADNCS:CCNT1,CCNT0) is executed from activation channel (activation channel of the scan conversion target) small number, and conversion is completed. Moreover, the activation channel that completes the A/D conversion of the conversion count specification can be confirmed by conversion count completion flag bit (ADEOCF:EOCF).

Then, when the A/D conversion of the conversion count specification of the final activation channel is completed, the scan conversion completion interrupt factor flag (ADSCANS:SCINT) is set in "1" and the scan conversion becomes the state of the stop. The scan conversion restarts (return from the state of the stop) when the A/D activation factor is generated even by one of the scan conversion targets.

Moreover, the conversion count completion flag bit (ADEOCF:EOCF) of each activation channel is automatically cleared to "0" by restarting (return from the state of the stop) the scan conversion.

The conversion count specification (ADNCS:CCNT1,CCNT0) can select 1 to 4 times of each activation channel.

The data protection function can select invalidity and effective of each activation channel. Moreover, when the data protection function is effective (ADTCS:PRT="1"), the data protection state of each activation channel can be confirmed by data protection state flag bit (ADPRTF:PRTF).

- In case of data protection function invalid (ADTCS:PRT="0")
- The scan conversion continuously executes A/D conversions other than the state of the stop without stopping. The data of the A/D conversion data is not protected, and, the last A/D conversion data converted in the activation channel is stored.
- In case of data protection function effective (ADTCS:PRT="1")
- The data of the A/D conversion data is protected. Therefore, the A/D conversion in the same activation channel stops the A/D conversion for the data protection period. However, when the scan conversion shifts to a different activation channel, the A/D conversion is continuously executed without stopping because A/D data register (ADTCD) is also different.

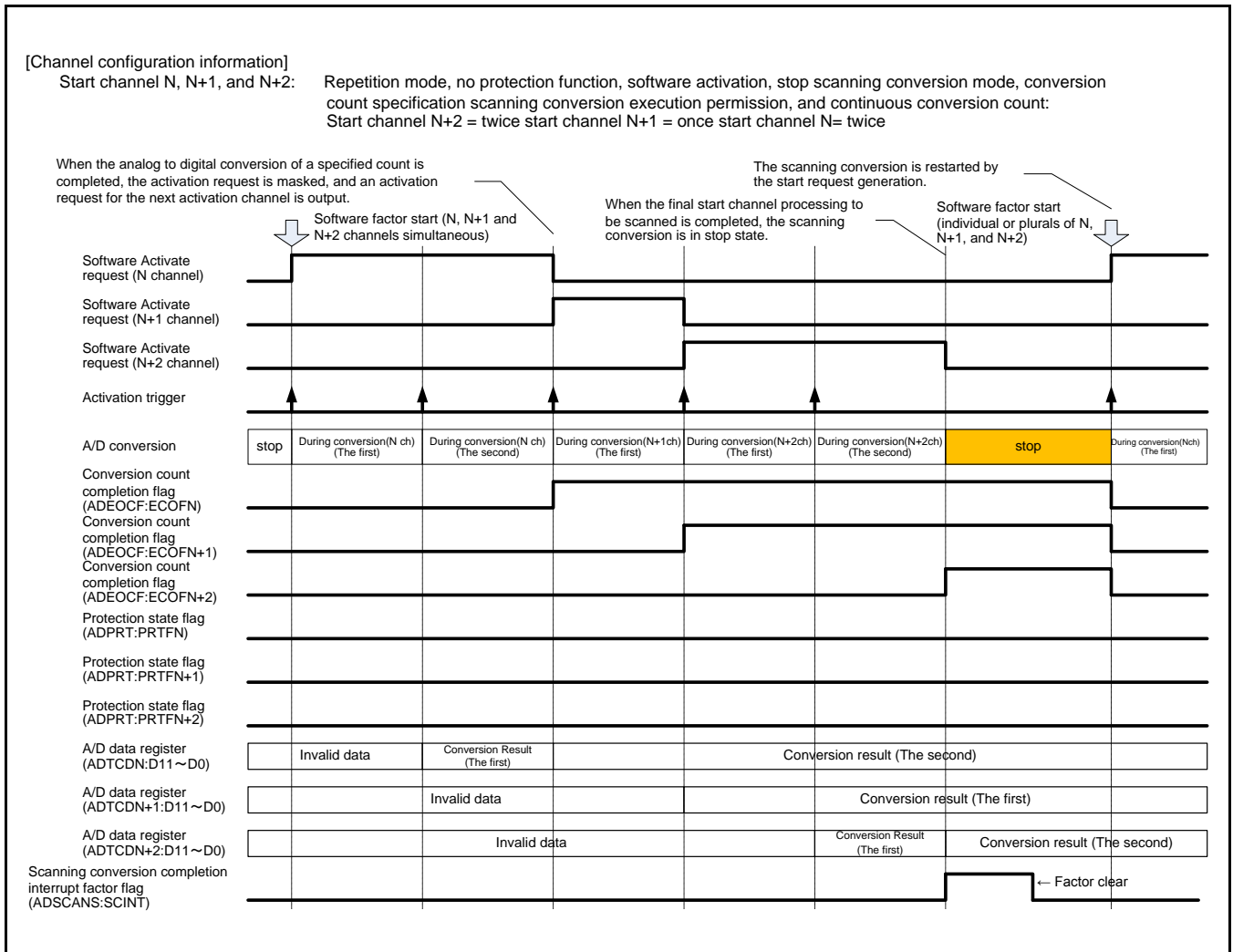
#### Notes:

- When an activation factor that is the same as the scan conversion target is set to an activation channel with a number larger than that of the final channel of stop scan conversion, A/D conversion can be executed only during the stop period, for the activation channel with the number larger than the final channel.
- The data protection function works at factors other than compare match activation (ADTCS:STS[1:0]="0b11").
- In the data protection function is effective (ADTCS:PRT="1"), please release the data protection state before starting the next scan conversion. The activation channel in the data protection state does not execute the A/D conversion.





**Figure 3-19 Stop Scan Conversion When Conversion Count of Each Channel is Specified (In Case of Data Protection Function Invalid)**



**Figure 3-20 Stop Scan Conversion When Conversion Count of Each Channel is Specified (In Case of Data Protection Function Effective)**

[Channel configuration information]

Start channel N, N+1, and N+2: Repetition mode, protection function, software activation (N to N+2 channel converted to scanning), stop scanning conversion mode, conversion count specification scanning conversion execution permission, and continuous conversion count: Start channel N+2 = twice start channel N+1 = once start channel N= twice

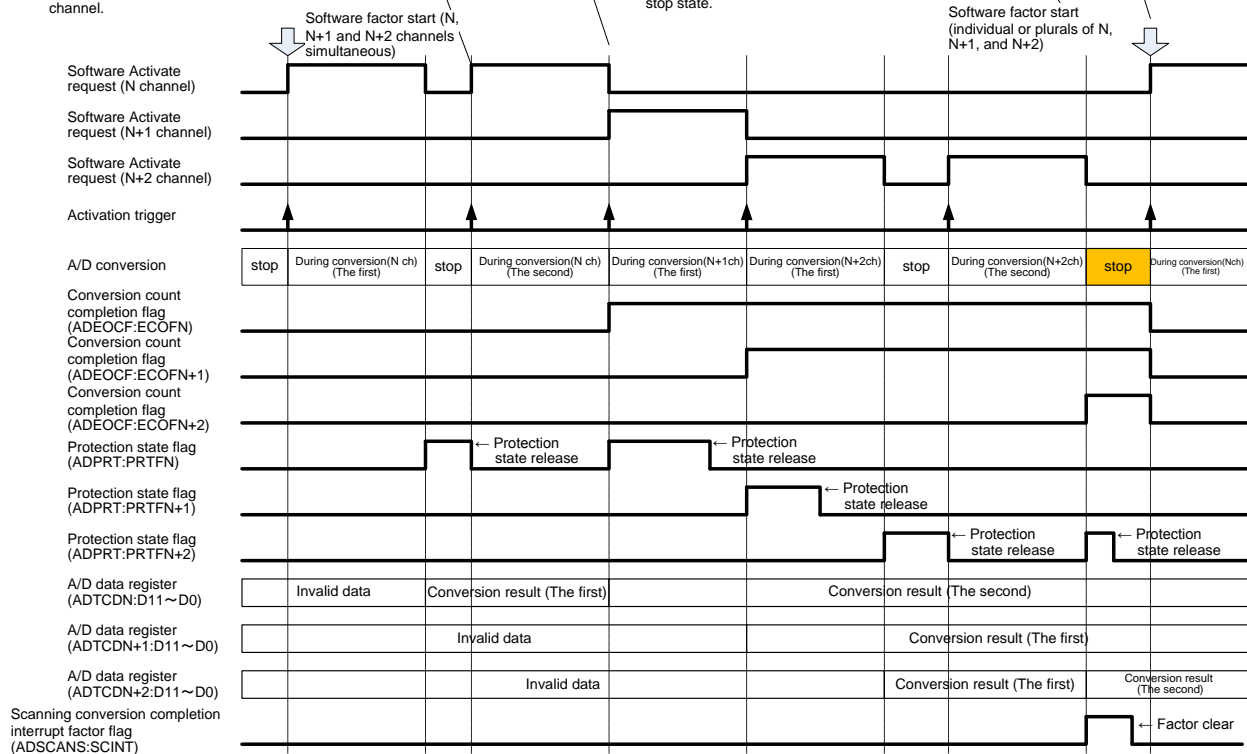
At continuous execution of different channels, the activate request of the next start channel is output regardless of the data protection state.

The A/D conversion is restarted by the data protection state release at continuous execution of the same start channel.

When the final start channel processing to be scanned is completed, the scanning conversion is in stop state.

The scanning conversion is restarted by the start request generation.

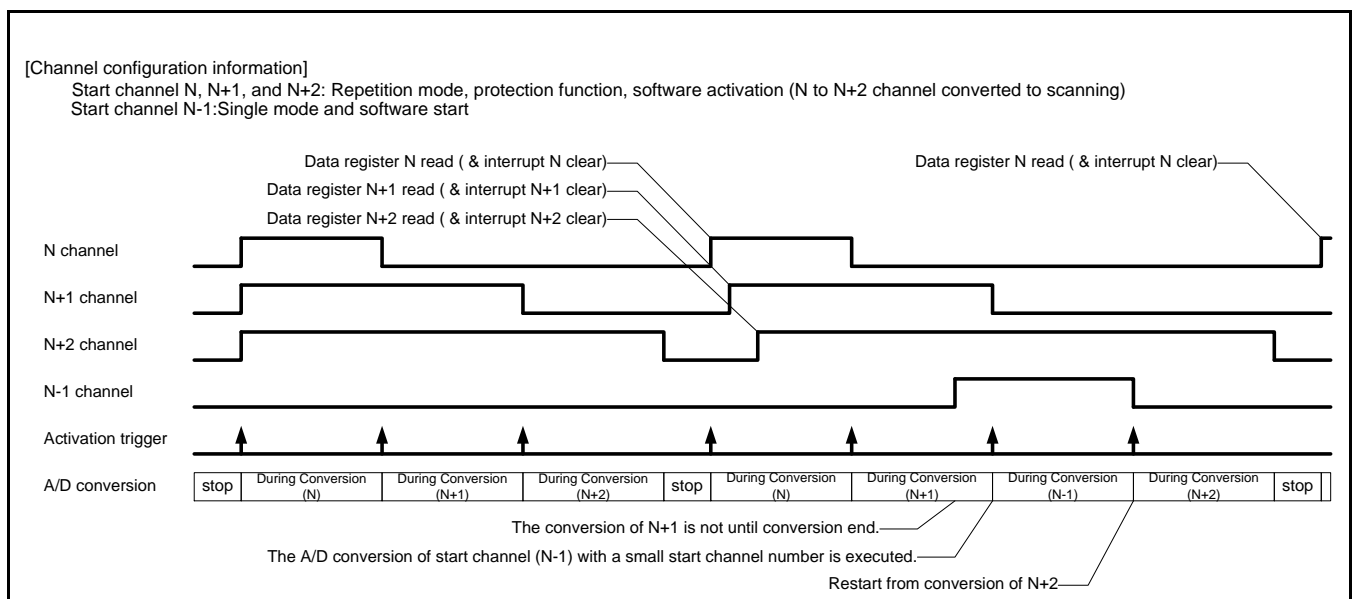
Software factor start (individual or plurals of N, N+1, and N+2)



### 3.2.13. High Priority Activation Request Operation of Other Activation Channel during the Scan Conversion

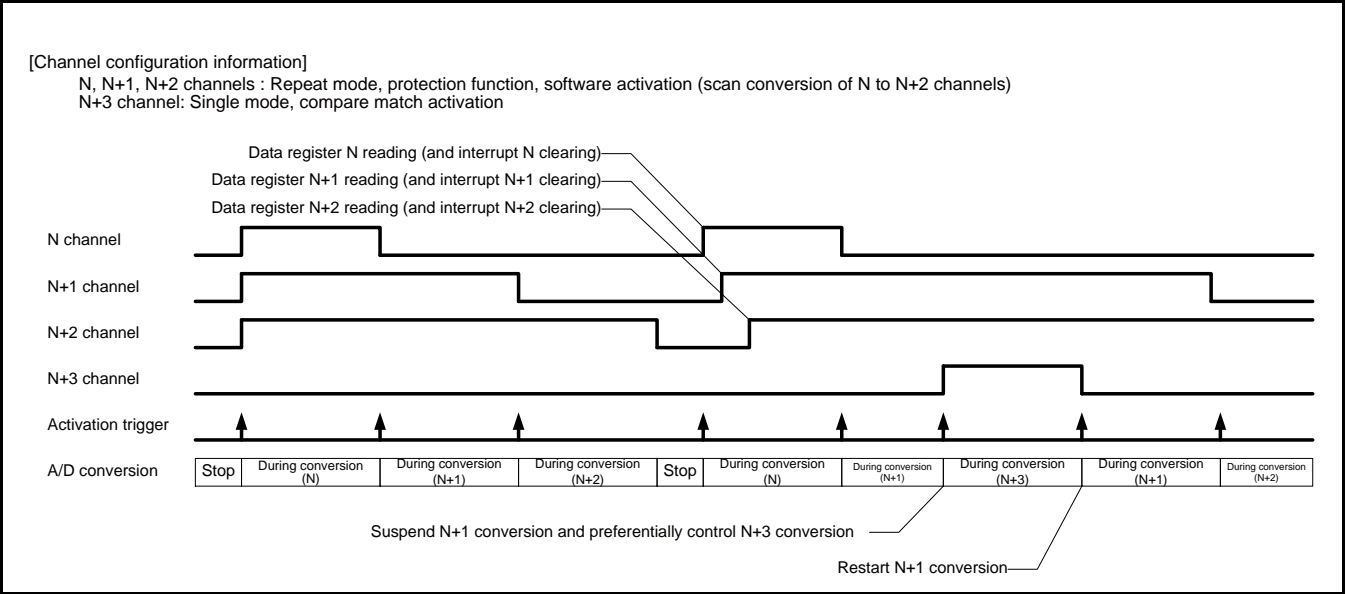
When the scan conversion is executed between the activation channels corresponding to the same A/D converter unit, the priority of the activation channels are controlled based on the activation factor and activation channel information. The priority control is done by the A/D activation arbitration of latter part. When an activation channel with the same activation factor is generated from an activation channel with an activation channel number that is smaller than that of an activation channel performing scan conversion, the scan conversion stops temporarily after completion of A/D conversion. Then, the scan conversion that stops temporarily is restarted after completing the A/D conversion of the activation channel from a small activation channel number.

**Figure 3-21 Operation of the Same Activation Factor Request from Small Activation Channel Number at Continuous Scan Conversion**



When the activation factor whose priority is higher than the activation channel that does the scan conversion is generated from other activation channels, the scan conversion is interrupted. Then, the A/D conversion is restarted from the activation channel of the interrupted scan conversion after the A/D conversion of the high priority is completed.

**Figure 3-22 Operation of High Priority Activation Factor Request by Other Activation Channel at Continuous Scan Conversion**



### 3.2.14. Forced Termination of Activation Request

Whether an A/D activation request or a conversion operation is in progress can be reported using the A/D conversion request bit. The current A/D activation request or conversion operation can be terminated forcibly by resetting the appropriate bit to "0".



### 3.2.15. Range Comparison Function

#### (1) Range Comparison Upper- and Lower-limit Threshold Value Settings

The upper-limit threshold setting register (ADRCUT) and the lower-limit threshold setting register (ADRCLT) can set 4 types of each unit. The setting selects one from the combination of the upper and lower-limit threshold setting register of 4 types by the upper and lower-limit threshold select bit (ADRCCS.RCOTS1,RCOTS0) of each activation channel.

Table 3-7 Range Comparison Upper- and Lower-limit Threshold Value Selection

Upper- and Lower-limit Threshold Selection Bits (ADRCCS[n]:RCOTS[1:0])	Selection Result
00	Upper-limit threshold setting register 0 (ADRCUT0) / lower-limit threshold setting register 0 (ADRCLT0)
01	Upper-limit threshold setting register 1 (ADRCUT1) / lower-limit threshold setting register 1 (ADRCLT1)
10	Upper-limit threshold setting register 2 (ADRCUT2) / lower-limit threshold setting register 2 (ADRCLT2)
11	Upper-limit threshold setting register 3 (ADRCUT3) / lower-limit threshold setting register 3 (ADRCLT3)

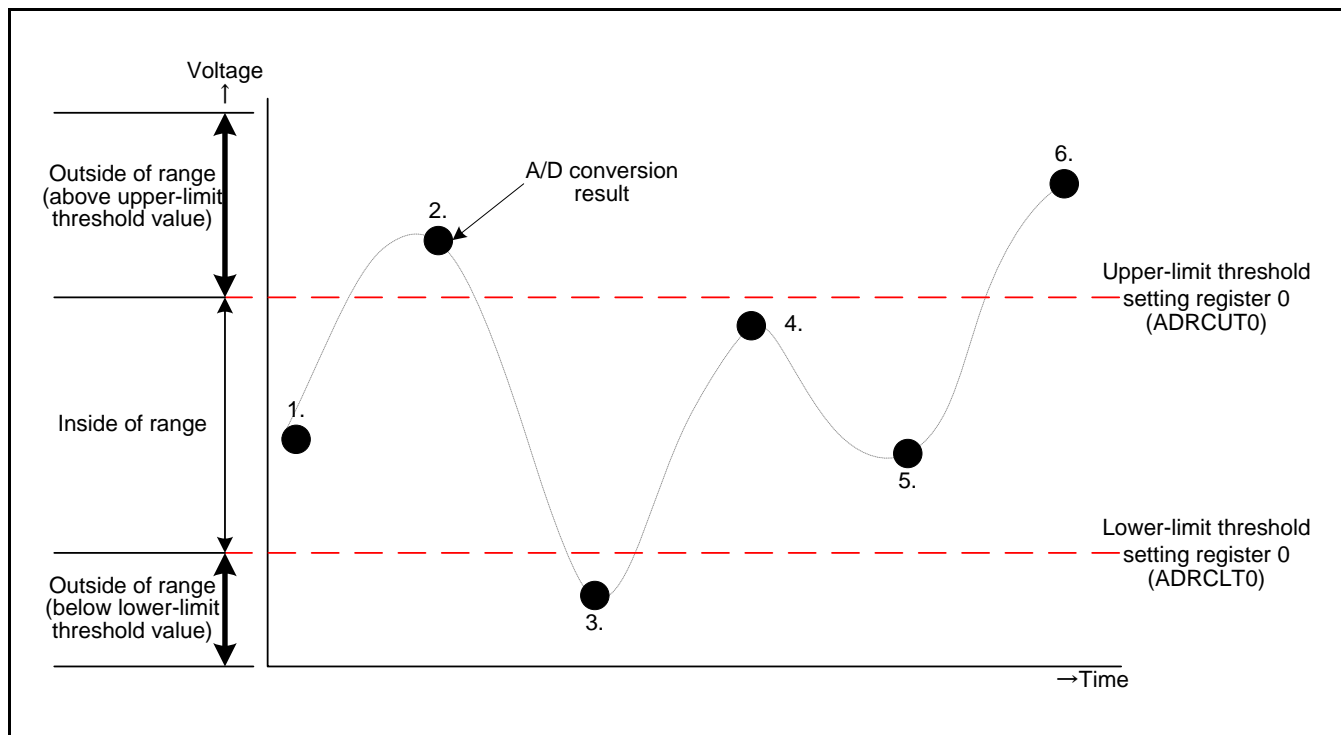
#### (2) Range Comparison Operation

When the range comparison enable setting (ADRCCS:RCOE="1"), the range comparison is executed when the A/D conversion is completed and data is stored in A/D data bit (ADTCD:D[11:0]). The range comparison compares the upper and lower bound threshold setting register (ADRCUT/ADRCLT) selected by the range comparison upper and lower bound threshold selection bit (ADRCCS:RCOTS[1:0]) with the A/D data bit (ADTCD:D[11:0]). The range comparison result is input to a continuous detecting function.

Table 3-8 Range Comparison Conditions

Range Comparison Result	Outside-of-range Confirmation (ADRCCS:RCOIRS="0")	Inside-of-range Confirmation (ADRCCS:RCOIRS="1")
Outside the range (larger than upper bound threshold) A/D data bit > Upper-limit threshold setting register Figure 3-23: 2., 6.	Detected	Undetected
Inside the range A/D data bit ≤ Upper-limit threshold setting register and A/D data bit ≥ Lower-limit threshold setting register Figure 3-23: 1., 4., 5.	Undetected	Detected
Outside the range (Smaller than lower bound threshold) A/D data bit < Lower-limit threshold setting register Figure 3-23: 3.	Detected	Undetected

Figure 3-23 Range Comparison Condition



### (3) Continuous Detection Function for Range Comparison Results

A continuous detecting function continuously detects the range comparison result, and removes the noise etc.

When the count in which the detection of the range comparison result is set by range comparison continuous detection count specification setting (ADRCSS:RCOCD[2:0]) is continuously detected, "1" is set to range comparison interrupt factor flag bit (ADRCIF:RCINT). When it becomes undetected even one time in the continuous detection by the range comparison result, the continuous detection measurement is cleared to 0 times and the measurement is restarted.

Table 3-9 Continuous Detection Function Operating Conditions

Continuous Detection Measurement Operation	<ul style="list-style-type: none"> <li>- The operation is controlled separately for each activation channel.</li> <li>- The operation is always running when range comparison execution is enabled (ADRCSS:RCOE="1").</li> </ul>
Continuous Detection Frequency	<ul style="list-style-type: none"> <li>- A value of 1 to 7 times can be selected using the continuous detection frequency specification (ADRCSS:RCOCD[2:0]).</li> <li>- The detection count can be checked using the continuous detection count display (ADRCSS:RCOCD[2:0]).</li> </ul>
Clearing Conditions	<ul style="list-style-type: none"> <li>- A condition is when range comparison execution is disabled (ADRCSS:RCOE="0").</li> <li>- A condition is when the range comparison result is not detected.</li> </ul>
Increment Conditions	<ul style="list-style-type: none"> <li>- A condition is when the range comparison result is detected. However, if the count reaches the continuous detection frequency specification (ADRCSS:RCOCD[2:0]) value, it stops at this value.</li> </ul>



**Notes:**

- When confirming outside the range, the continuous detection measurement does not clear to 0 times and continue the continuous detection even if the state of the range comparison result changes from larger than the upper bound threshold into smaller than lower bound threshold.
- When you want to initialize the continuous detection count of the range comparison result, please change to enable setting (ADRCSS:RCOE="0"→"1") after the setting of the range comparison execution disable, while the A/D is not being requested.
- After two clocks with the peripheral clock from flag set (ADTCS:INT="1") of the interrupt control bit by the A/D conversion completion interrupt, the range comparison interrupt factor flag (ADRCIF:RCINT) is set in "1" by the continuous detection.

**(4) Range Comparison Out-of-range Flag Control**

For a range comparison that confirms the result to be outside the range (ADRCSS:RCOIRS is "0"), the range comparison out-of-range flag bit (ADRCOT:RCOOF[n]) in the respective channel indicates whether the result is above the upper-limit threshold value or below the lower-limit threshold value.

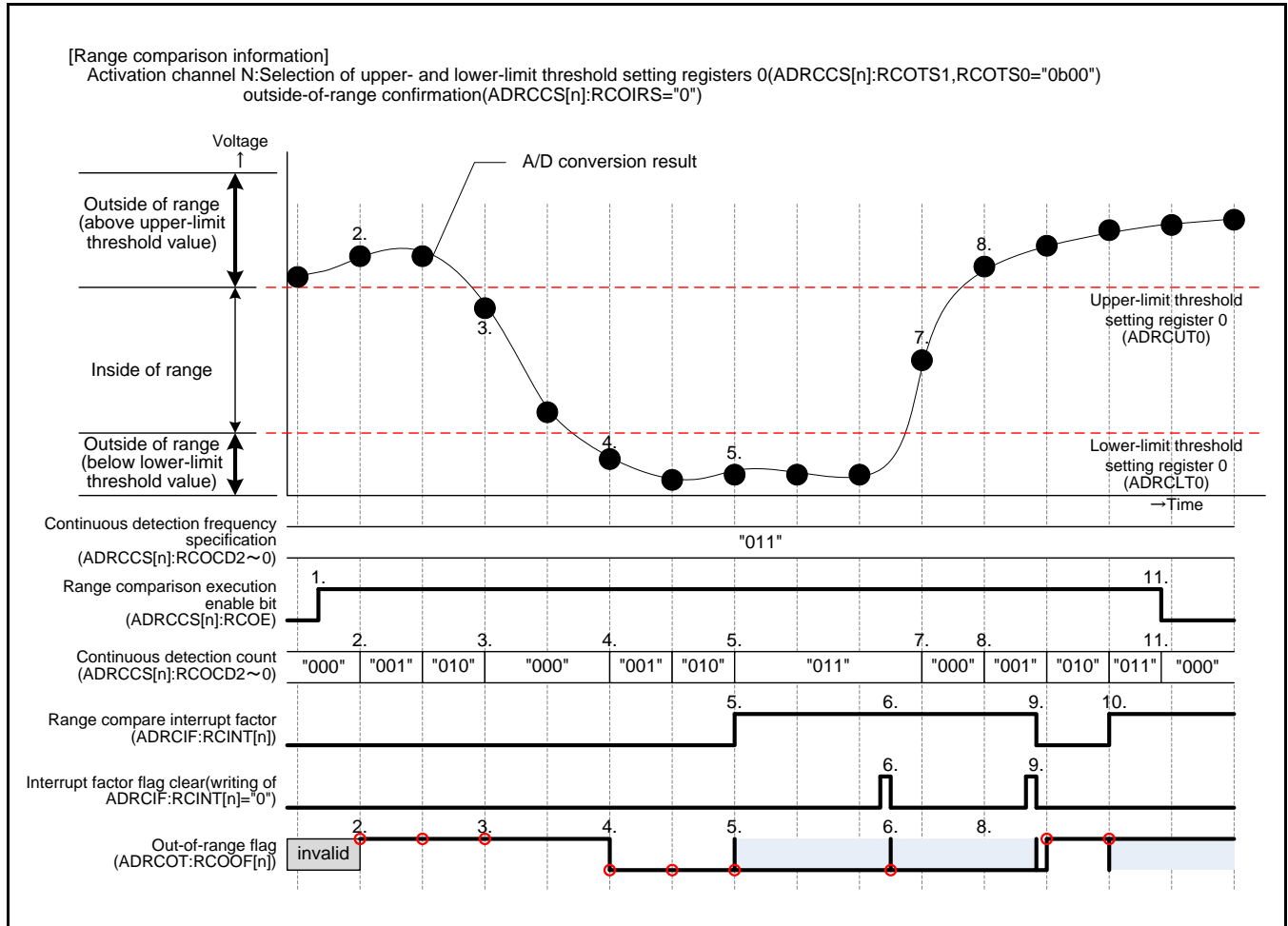
**Table 3-10 Conditions Determining Range Comparison Out-of-range Flag**

Range Comparison Result	Range Comparison Out-of-range Flag Bit (ADRCOT:RCOOF)	
	Outside-of-range Confirmation (ADRCSS:RCOIRS="0")	Inside-of-range Confirmation (ADRCSS:RCOIRS="1")
Outside of range (above upper-limit threshold value) A/D data bits > Upper-limit threshold setting register	"1"	Last value retained
Inside of range A/D data bits ≥ Lower-limit threshold setting register and A/D data bits ≤ Upper-limit threshold setting register	Last value retained	Last value retained
Outside of range (below lower-limit threshold value) A/D data bits < Lower-limit threshold setting register	"0"	Last value retained

Also, while the range compare interrupt factor flag (ADRCIF:RCINT) is set to "1", the contents of the range comparison out-of-range flag bit (ADRCOT:RCOOF) are retained.

### (5) Range Comparison Operation Example

Figure 3-24 Range Comparison Operation Example







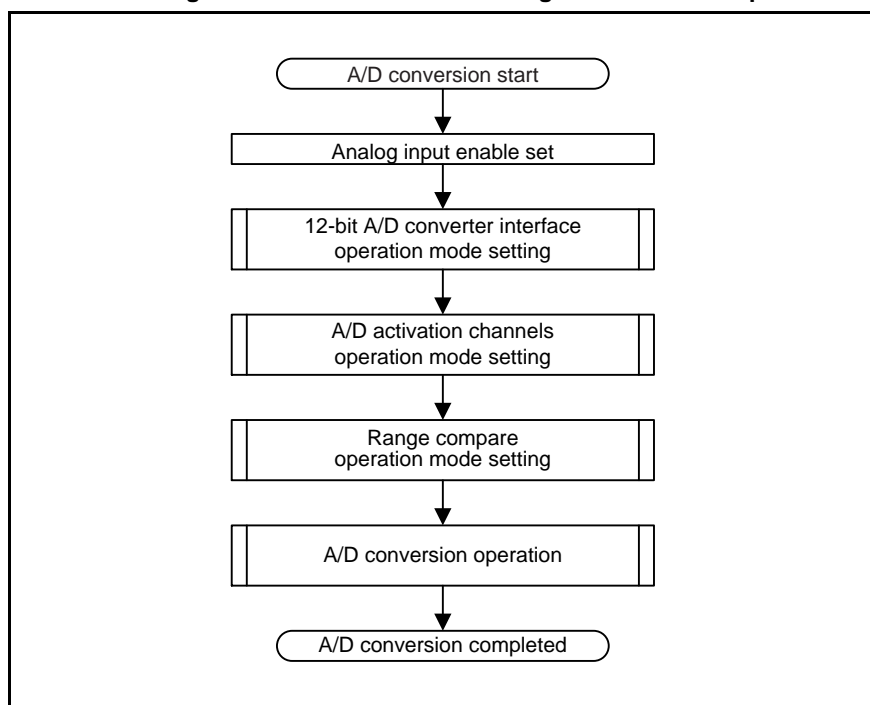
6. If the range compare interrupt factor flag clear (ADRCIF:RCINT[n] is "0") is in contention with the continuous detection state, the continuous detection state determines the priority by the set operation. With the range compare interrupt factor flag in the set state (ADRCIF:RCINT[n] is "1"), the out-of-range flag (ADRCOT:RCOOF[n]) resets the out-of-range state.
7. The continuous detection count is initialized (ADRCCS[n]:RCOCD[2:0] is "0b000") when the range comparison result is inside the range, even though the range compare interrupt factor flag is set (ADRCIF:RCINT[n] is "1").
8. The range comparison result above the upper-limit threshold value causes an increment in the continuous detection count (ADRCCS[n]:RCOCD[2:0]) even if the range compare interrupt factor flag is in the set state (ADRCIF:RCINT[n] is "1"). However, the out-of-range flag (ADRCOT:RCOOF[n]) retains its previous value because the range compare interrupt factor flag is in the set state (ADRCIF:RCINT[n] is "1").
9. The range compare interrupt factor flag is cleared (ADRCIF:RCINT[n] is "0") by the range compare interrupt factor flag clear (ADRCIF:RCINT[n] is "0"). Also, the retained state of the out-of-range flag (ADRCOT:RCOOF[n]) is released.
10. The number of consecutive range comparison results that are outside the range reaches the continuous detection frequency specification value (ADRCCS[n]:RCOCD[2:0] is "0b011"). As a result, the range compare interrupt factor flag (ADRCIF:RCINT[n]) is set to "1". Also, the out-of-range flag (ADRCOT:RCOOF[n]) is set to the out-of-range state for the time that the range compare interrupt factor flag was set (ADRCIF:RCINT[n] is "1"). This out-of-range setting is retained until the range compare interrupt factor flag is cleared (ADRCIF:RCINT[n] is "0").
11. The continuous detection count (ADRCCS[n]:RCOCD[2:0]) is initialized to "0b000" when range comparison execution is disabled (ADRCCS[n]:RCOE is "0"). Neither the range compare interrupt factor flag (ADRCIF:RCINT[n]) nor the out-of-range flag (ADRCOT:RCOOF[n]) is cleared by the disabling of range comparison execution (ADRCCS[n]:RCOE is "0").

## 4. Setting Procedure Example

This section shows examples of A/D activation compare setting procedures.

### 4.1. A/D Conversion Setting Procedure Examples

Figure 4-1 A/D Conversion Setting Procedure Example



#### (1) Analog Input Enable Settings

The analog input control register (ADER:ADE0 to ADE31) is used to enable/disable analog input. For settings for the analog input control register (ADER:ADE0 to ADE31), see Section "5.1.1. Analog Input Control Register (ADER)".

#### (2) 12-bit A/D Converter Interface Operation Mode Settings

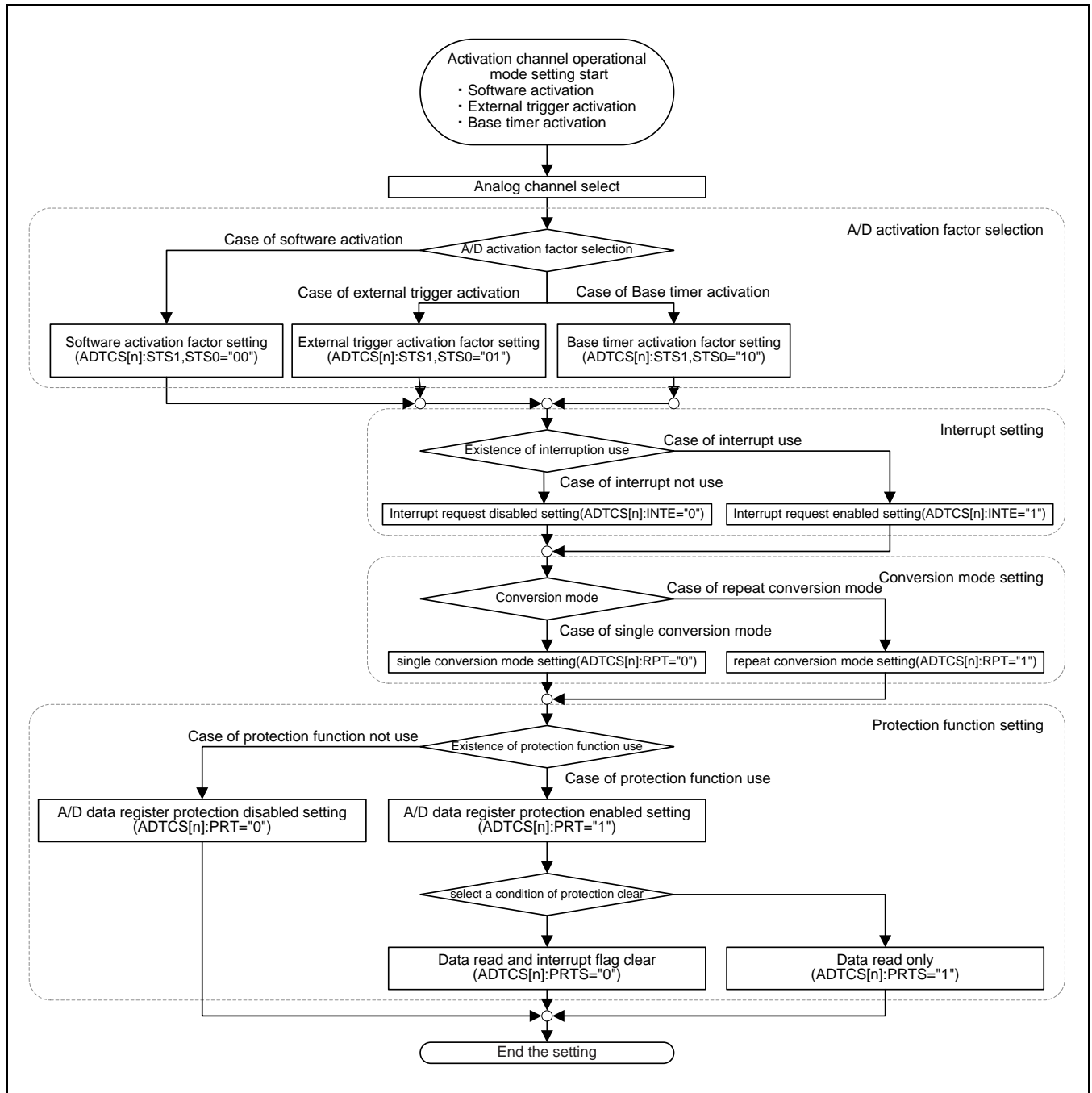
For details on the operation mode settings of the 12-bit A/D converter interface, see the setting procedure example in the following "CHAPTER: 12-bit A/D Converter Interface".



### (3) 12-bit A/D Converter Interface Operation Mode

#### a) A/D Conversion Setting Procedure Example

Figure 4-2 A/D Conversion Setting Procedure Example



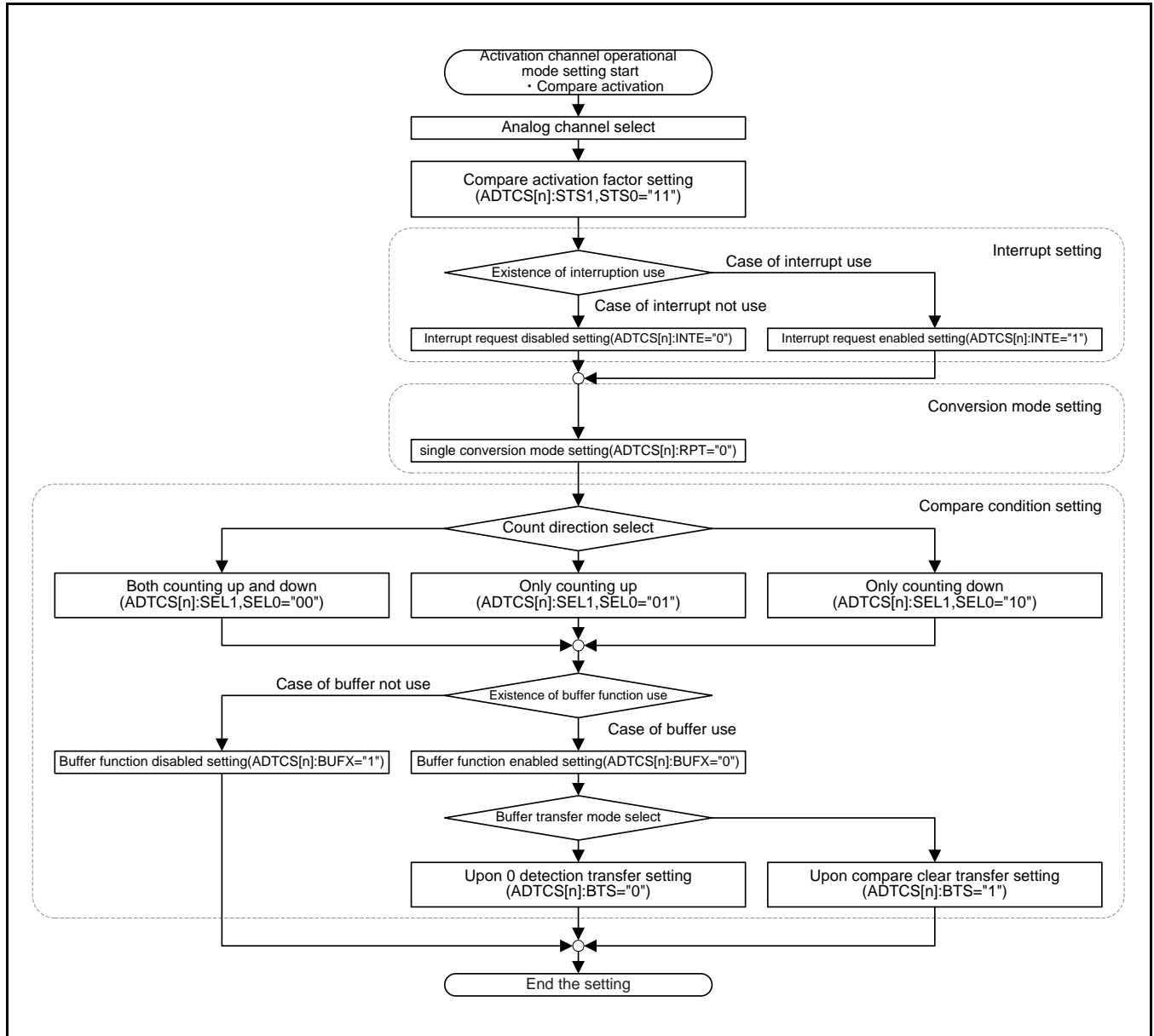
The setting contents described above can be set in random order.

- The analog channel selection bits that corresponds to the analog channel is set.
- The following control bits hold no meaning except in compare match activation mode (ADTCS:STS[1:0] is "0b11"), so they need not be configured. (They are kept set at their initial values.)
  - Compare value buffer bits (ADCOMPB[n]:CMP[15:0])
  - Compare value bits (ADCOMP[n]:CMP[15:0])

- Count direction selection bits (ADTCS[n]:SEL[1:0])
- Compare register buffer function control bit (ADTCS[n]:BUFX)
- Compare register buffer transfer control bit (ADTCS[n]:BTS)

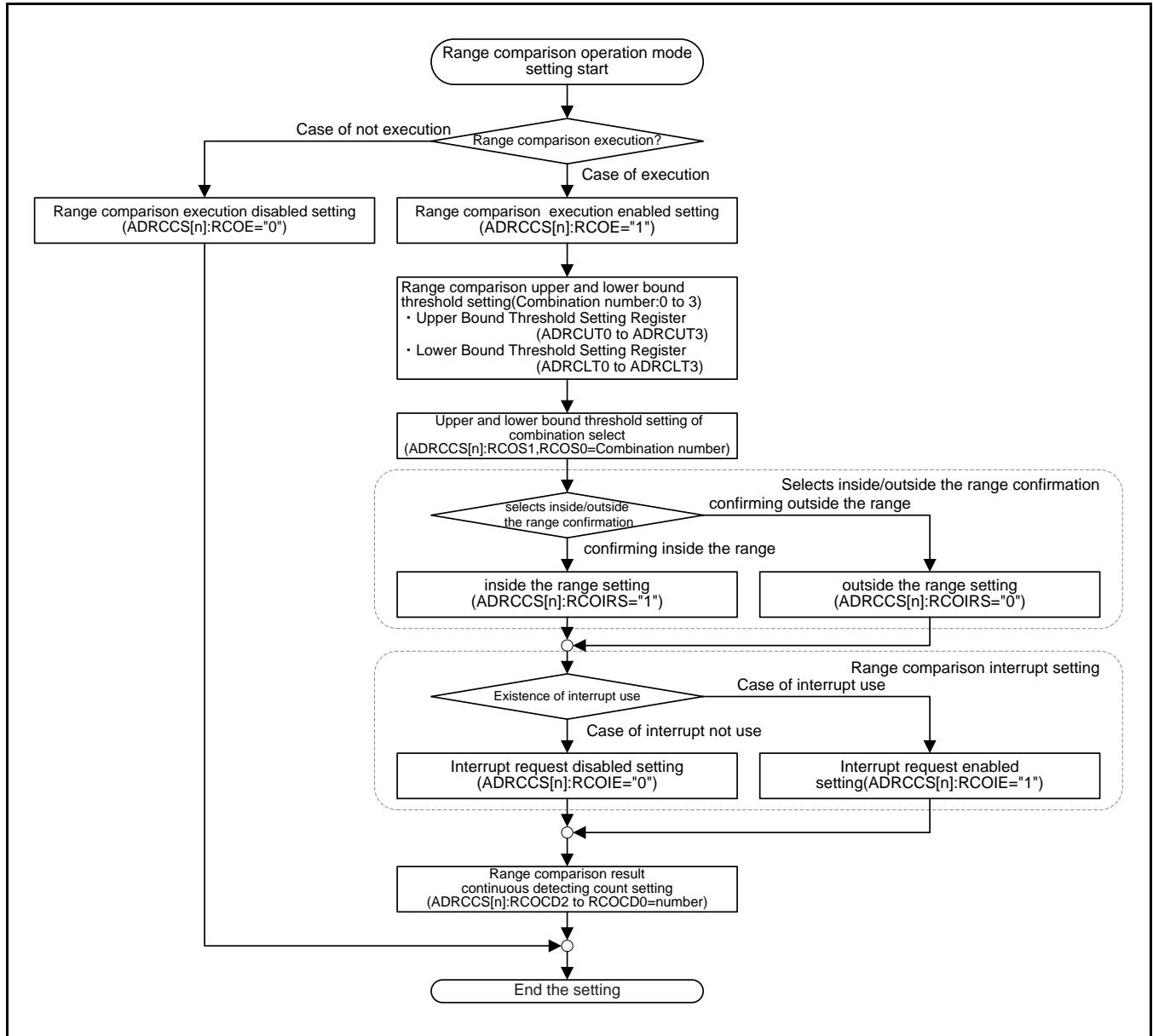
b) Activation Channel Operation Mode Settings

Figure 4-3 A/D Conversion Setting Procedure Example



(4) Setting Example for Software, External Trigger, and Base Timer Activation

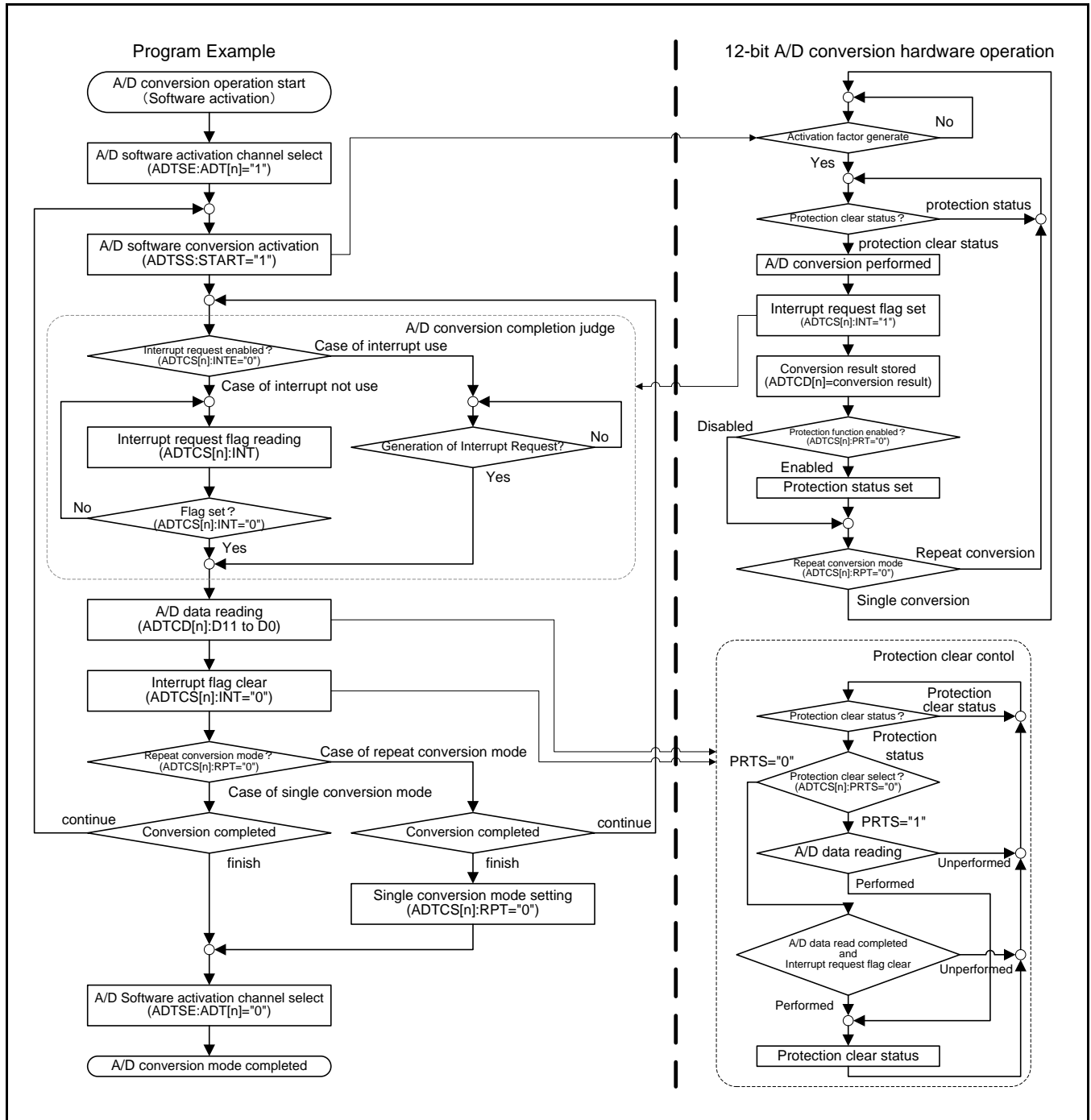
Figure 4-4 A/D Conversion Setting Procedure Example





b) Setting Example for Compare Match Activation

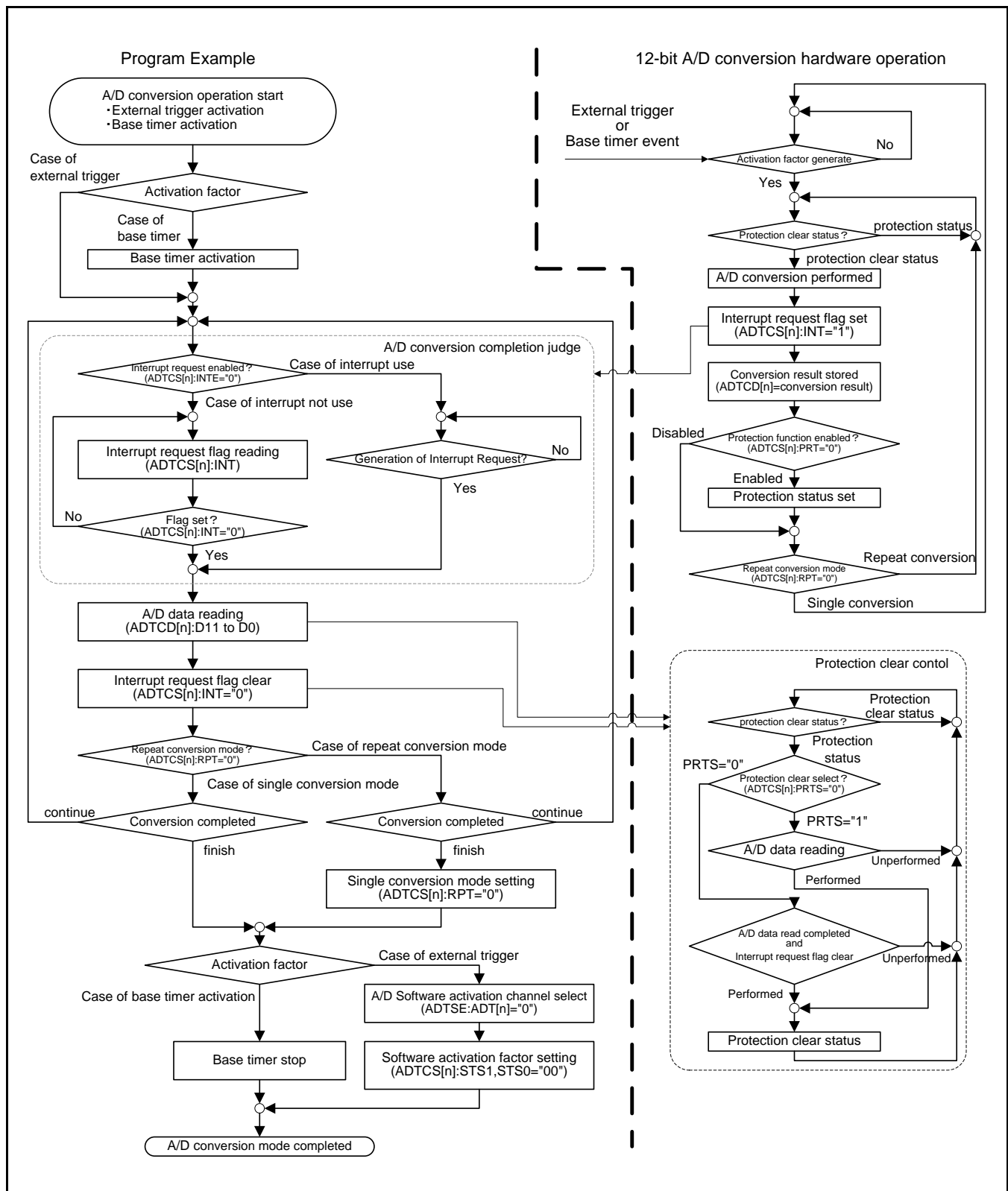
Figure 4-5 A/D Conversion Setting Procedure Example



(4) Setting Example for Range Comparison Operation Mode

b) A/D Conversion Operation Example for External Trigger and Base Timer Activation

Figure 4-6 A/D Conversion Operation Example for External Trigger and Base Timer Activation

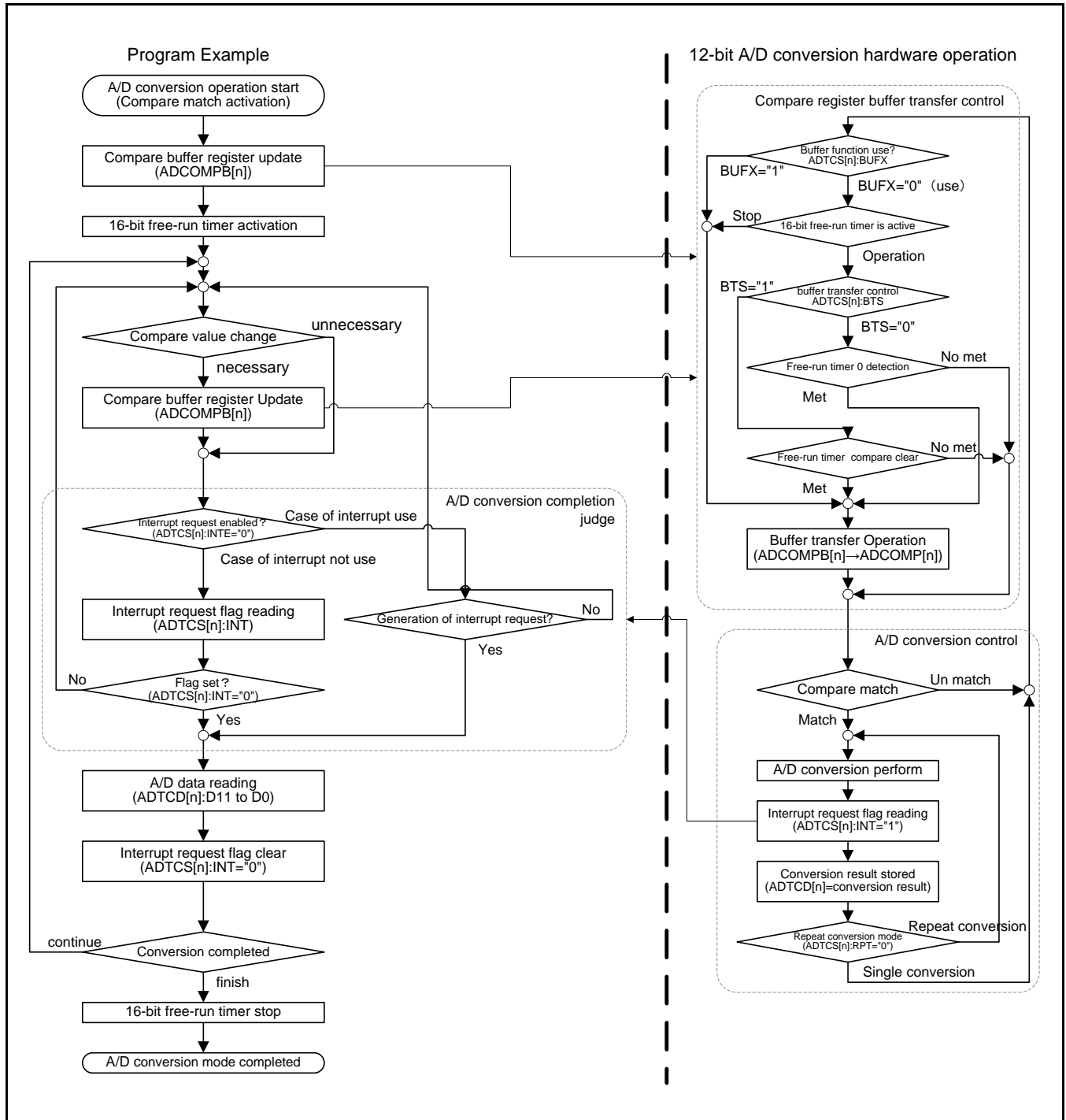






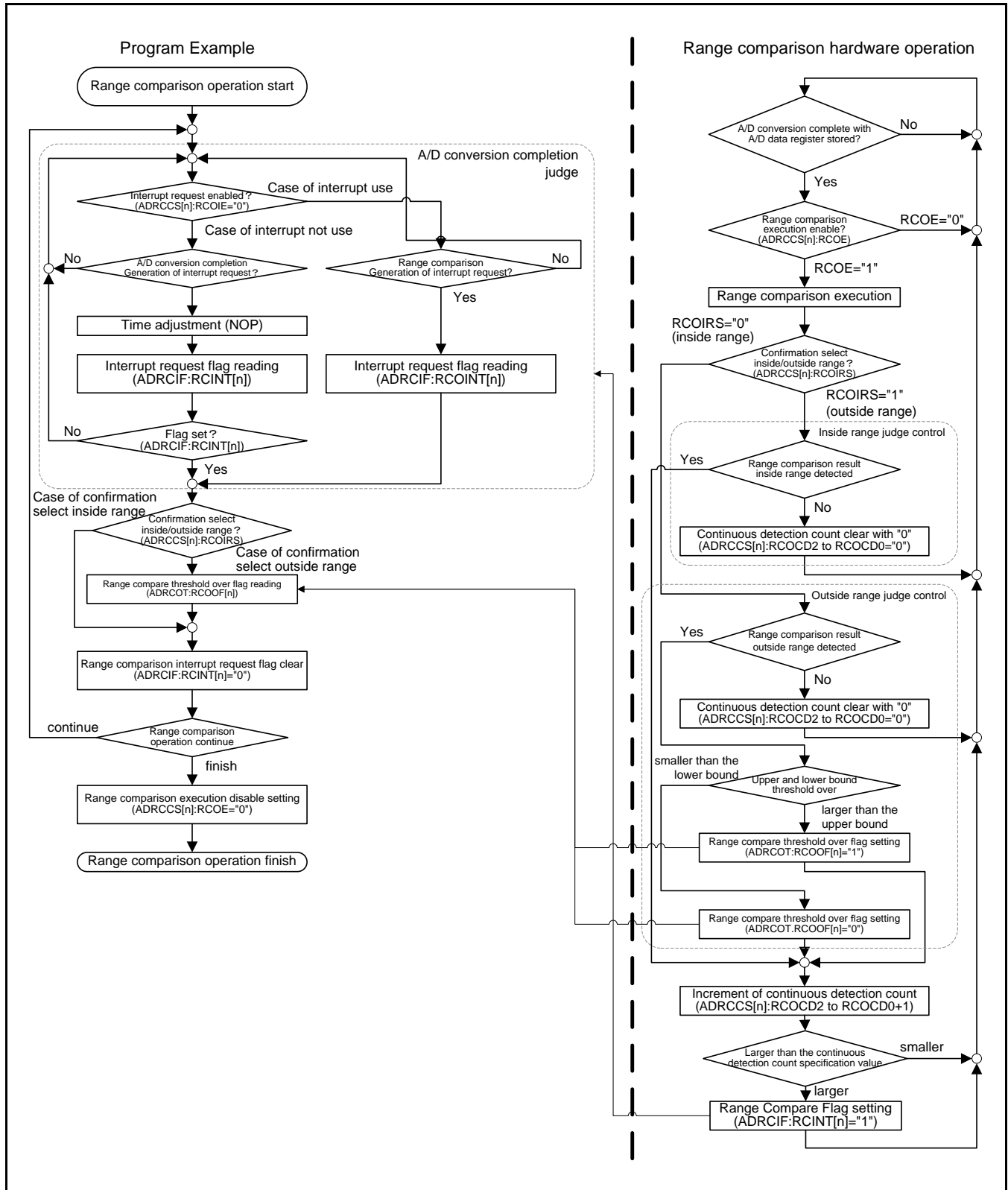
c) A/D Conversion Operation Example for Compare Match Activation

Figure 4-7 A/D Conversion Operation Example for Compare Match Activation



d) Range Comparison Operation Example

Figure 4-8 Range Comparison Operation Example

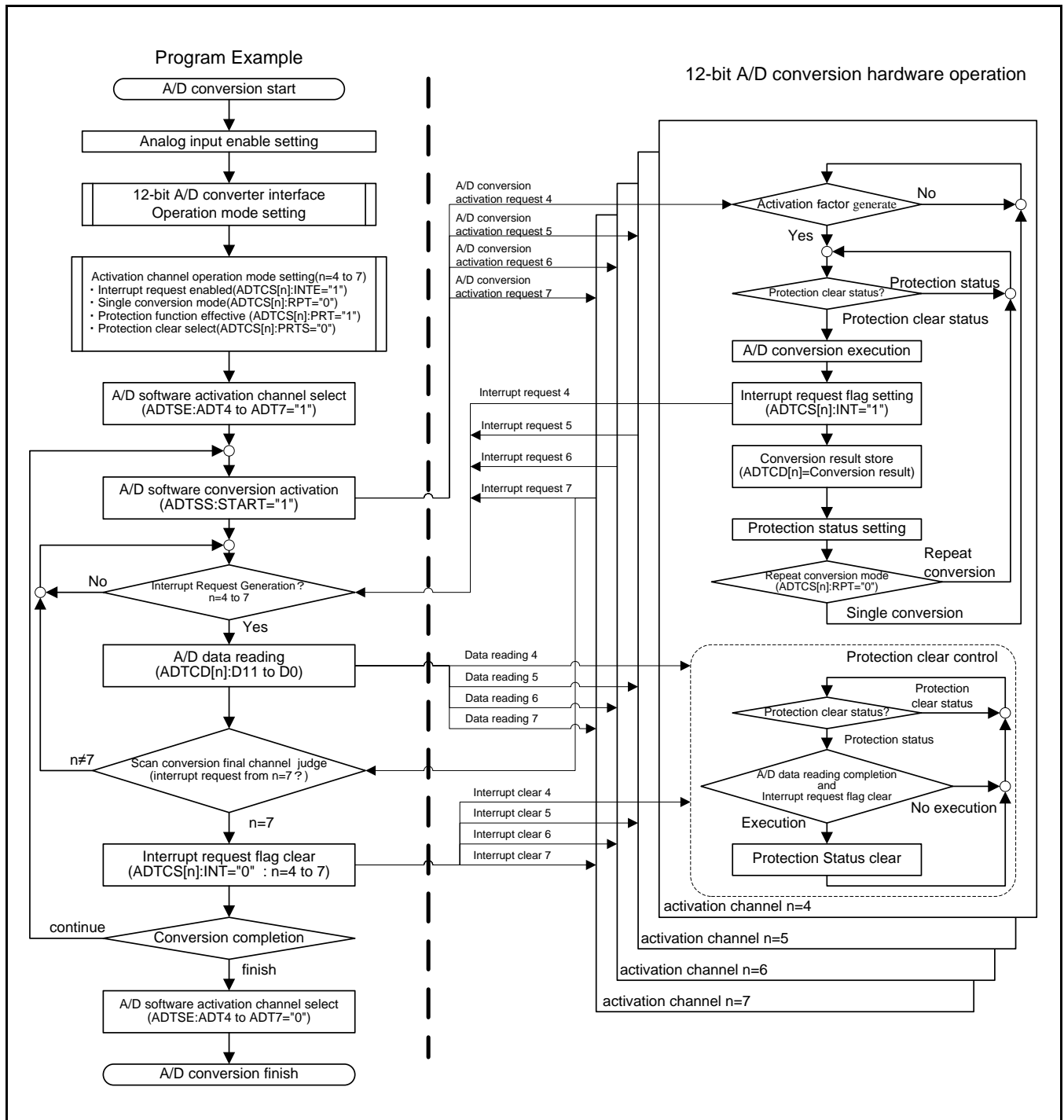


## 4.2. Setting Procedure Example for Scan Conversion

### (1) Setting Procedure Example for Single Scan Conversion

This section provides a setting procedure example for single scan conversion for the case where 4 activation channels (activation channel 4 to activation channel 7) are used and software activation is performed.

Figure 4-9 Setting Procedure Example for Single Scan Conversion



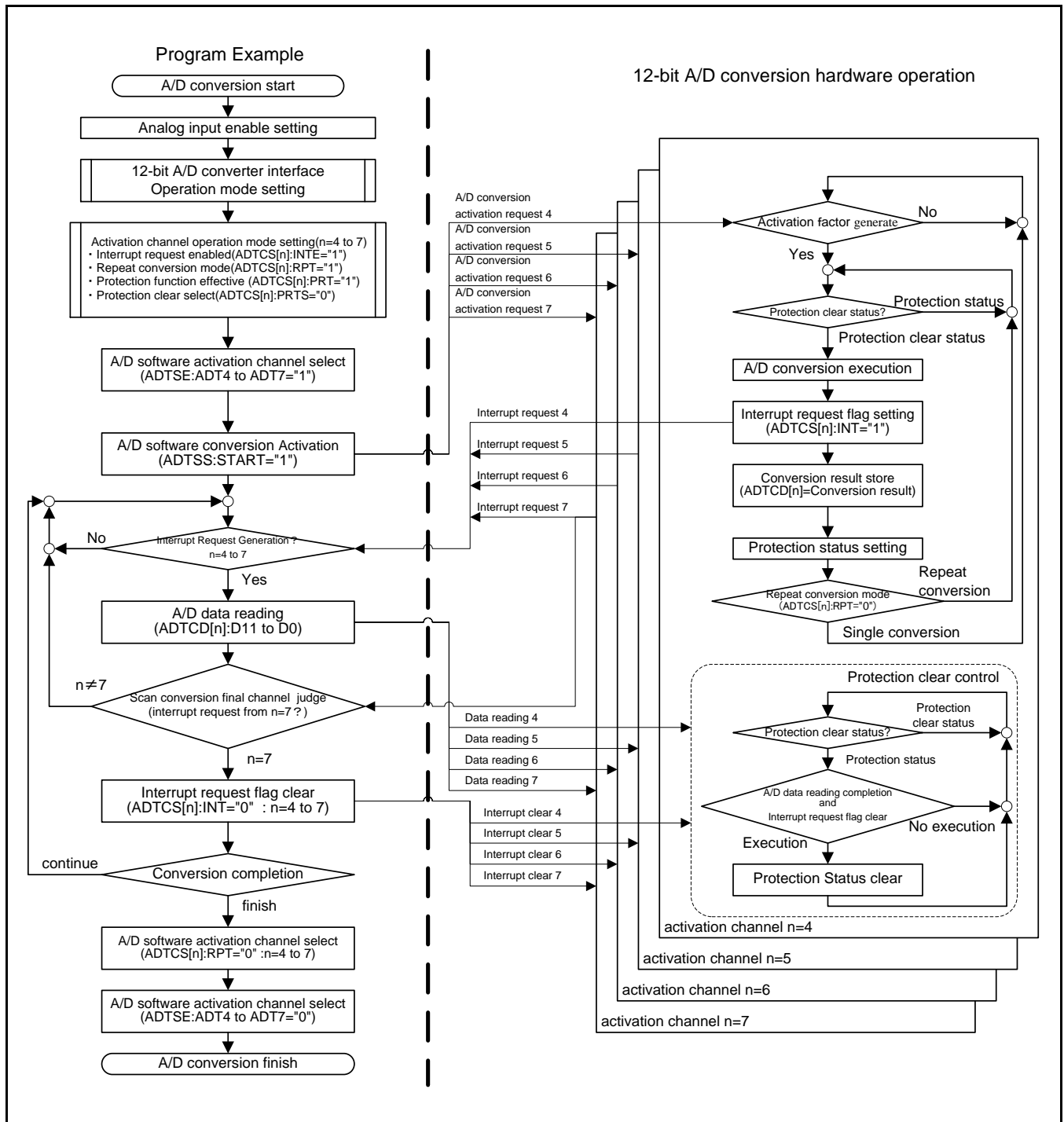
For software activation, set "1" in the A/D software activation channel selection bit (ADTSE:ADT) corresponding to an activation channel to be scanned and write "1" to the A/D software conversion start bit (ADTSS:START). Then, at the same time, the A/D conversion start request signal is generated only once, which makes the activation channel start operating.

- After scan conversion for the final activation channel ends, cancel the data protection state of each activation channel. If the data protection state is canceled in the middle of scan conversion, the scan conversion will not be performed in the correct order.
- Before software activation starts, cancel the data protection state of each activation channel. If A/D conversion is started without canceling the data protection state, scan conversion will not be performed in the correct order.
- Scan conversion is performed starting with the activation channel with the lowest activation channel number. Therefore, the order in which scan conversion is performed on analog channels can be programmed by setting the analog channel selection bits (ADTECS[n]:CHSEL[4:0]) for activation channels.

## **(2) Setting Procedure Example for Continuous Scan Conversion**

This section provides a setting procedure example for continuous scan conversion for the case where 4 activation channels (activation channel 4 to activation channel 7) are used and software activation is performed.

Figure 4-10 Setting Procedure Example for Continuous Scan Conversion



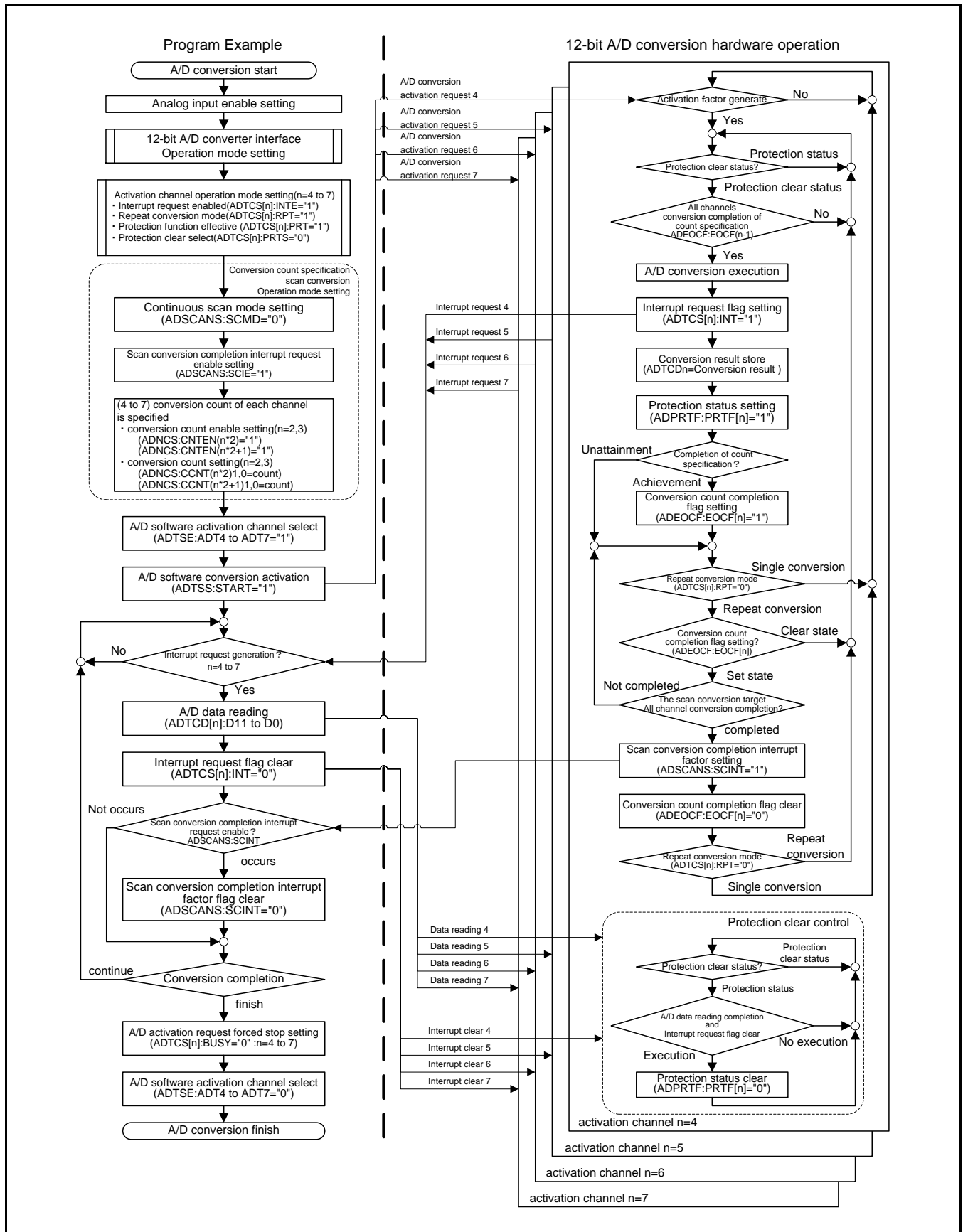
- For software activation, set "1" in the A/D software activation channel selection bit (ADTSE:ADT) corresponding to an activation channel to be scanned and write "1" to the A/D software conversion start bit (ADTSS:START). Then, at the same time, the A/D conversion start request signal is continuously generated, which makes the activation channel start operating.
- After scan conversion for the final activation channel ends, cancel the data protection state of each activation channel to start the next scan conversion. If the data protection state is canceled in the middle of scan conversion, the scan conversion will not be performed in the correct order.

- Before software activation starts, cancel the data protection state of each activation channel. If A/D conversion is started without canceling the data protection state, scan conversion will not be performed in the correct order.
- Scan conversion is performed starting with the activation channel with the lowest activation channel number. Therefore, the order in which scan conversion is performed on analog channels can be programmed by setting the analog channel selection bits (ADTECS[n]:CHSEL[4:0]) for activation channels.

### **(3) Setting Procedure Example for Continuous Scan Conversion by Specifying the Conversion Count for Each Channel**

This section provides a setting procedure example for continuous scan conversion by specifying the conversion count for each channel. This example is for the case where 4 activation channels (activation channel 4 to activation channel 7) are used and software activation is performed.

Figure 4-11 Setting Procedure Example for Continuous Scan Conversion by Specifying the Conversion Count for Each Channel



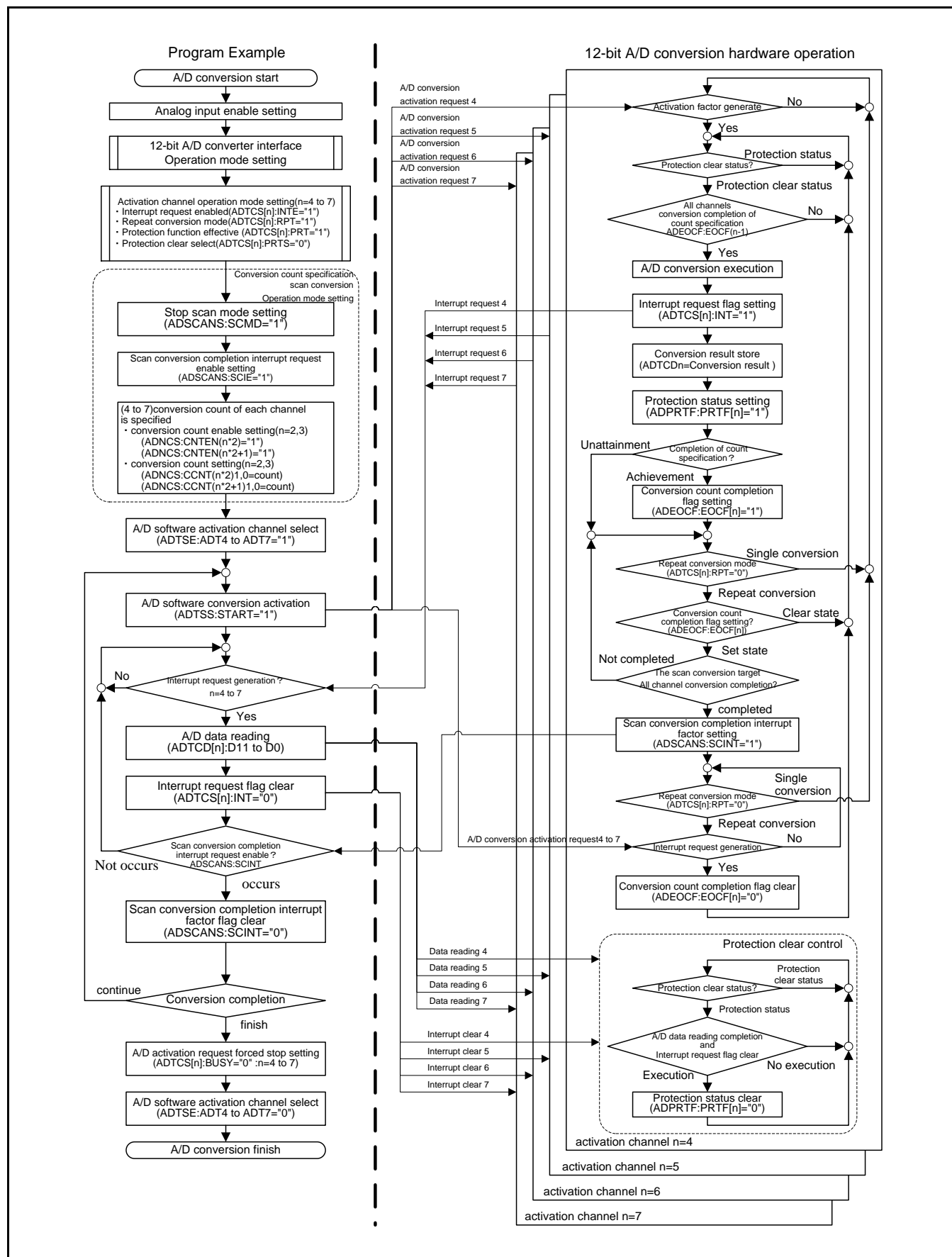
- For software activation, set "1" in all the A/D software activation channel selection bits (ADTSE:AST) corresponding to the activation channels to be scanned and write "1" to the A/D software conversion start bit (ADTSS:START). Then, the A/D conversion start request signal is continuously generated, which makes the activation channels start operating.
- The next scan conversion automatically starts when conversion is performed as many times as the count specified for the final activation channel.
- There is no restriction on when to cancel the data protection state. However, if an activation channel is in the data protection state, scan conversion is stopped until the state is canceled.
- Scan conversion is performed starting with the activation channel with the lowest activation channel number. Therefore, the order in which scan conversion is performed on analog channels can be programmed by setting the analog channel selection bits (ADTECS[n]:CHSEL[4:0]) for activation channels.
- While scan conversion by specifying the conversion count for each channel is being performed, the A/D activation request cannot be stopped for only some of the activation channels to be scanned. To change the activation channels to be scanned, stop all the activation channels to be scanned, specify the conversion count for each channel, and then reactivate the activation channels to be scanned.

#### **(4) Setting Procedure Example for Pause Scan Conversion by Specifying the Conversion Count for Each Channel**

This section provides a setting procedure example for pause scan conversion by specifying the conversion count for each channel. This example is for the case where 4 activation channels (activation channel 4 to activation channel 7) are used and software activation is performed.



**Figure 4-12 Setting Procedure Example for Pause Scan Conversion by Specifying the Conversion Count for Each Channel**



- For software activation, set "1" in all the A/D software activation channel selection bits (ADTSE:ADT) corresponding to the activation channels to be scanned and write "1" to the A/D software conversion start bit (ADTSS:START). Then, the A/D conversion start request signal is continuously generated, which makes the activation channels start operating.
- The scan conversion pauses when conversion is performed as many times as the count specified for the final activation channel.
- The scan conversion in the pause state resumes when an activation factor is generated for even 1 scan conversion target.
- There is no restriction on when to cancel the data protection state. However, if an activation channel is in the data protection state, scan conversion is stopped until the state is canceled.
- Scan conversion is performed starting with the activation channel with the lowest activation channel number. Therefore, the order in which scan conversion is performed on analog channels can be programmed by setting the analog channel selection bits (ADTECS[n]:CHSEL[4:0]) for activation channels.
- While scan conversion by specifying the conversion count for each channel is being performed, the A/D activation request cannot be stopped for only some of the activation channels to be scanned. To change the activation channels to be scanned, stop all the activation channels to be scanned, specify the conversion count for each channel, and then reactivate the activation channels to be scanned.



## 5. Registers

This section describes the registers of A/D activation compare.

**Table 5-1 List of Analog Input Control Registers**

Abbreviated Register Name	Register Name	See
ADER	Analog Input Control Register	5.1.1
KEYCDR	Key Code Register	5.1.2

A prefix (ADC12B\_) is added to each of the 12-bit A/D converter registers.

**Table 5-2 List of A/D Activation Compare Registers**

Abbreviated Register Name	Register Name	See
ADC12B_ADTSS	A/D Software Activation Register	5.2.1
ADC12B_ADTSE	A/D Software Activation Channel Selection Register	5.2.2
ADC12B_ADCOMP[n]	Compare Buffer Register	5.2.3
ADC12B_ADCOMP[n]	Compare Register	5.2.3
ADC12B_ADTCS[n]	A/D Activation Trigger Control Status Register	5.2.4
ADC12B_ADTCD[n]	A/D Data Register	5.2.5
ADC12B_ADTECS[n]	A/D Activation Trigger Extend Control Register	5.2.6
ADC12B_ADRCUT0 to 3	Upper-limit threshold setting register 0 to 3	5.2.7
ADC12B_ADRCLT0 to 3	Lower-limit threshold setting register 0 to 3	5.2.8
ADC12B_ADRCSS[n]	Range Comparison Control Status Register	5.2.9
ADC12B_ADRCOT	Range Comparison Out-of-range Flag Register	5.2.10
ADC12B_ADRCIF	Range Comparison Flag Register	5.2.11
ADC12B_ADSCANS0	Scan Conversion Control Status Register	5.2.12
ADC12B_ADNCSS[m]	Activation Channel Conversion Count Setting Register	5.2.13
ADC12B_ADPRTF	Data Protection Status Flag Register	5.2.14
ADC12B_ADEOCF	Activation Channel Conversion Completion Flag Register	5.2.15
ADC12B_ADTCS[n]	A/D Activation Trigger Control Status Clear Register	5.2.16
ADC12B_ADRCIFC	Range Comparison Flag Clear Register	5.2.17
ADC12B_ADSCANS0	Scan Conversion Control Status Clear Register	5.2.18
ADC12B_ADTCS[n]	A/D Activation Trigger Control Status Set Register	5.2.19
ADC12B_ADSCANS0	Scan Conversion Control Status Set Register	5.2.20

m=0 to 15, n=0 to 31

## 5.1. Registers of Analog Input Control

Registers of analog input control include analog input control register and key code register.

### 5.1.1. Analog Input Control Register (ADER)

Analog input control register (ADERH, ADERL) is used to control analog input.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	ADE31	ADE30	ADE29	ADE28	ADE27	ADE26	ADE25	ADE24
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	1	1	1	1	1	1	1	1

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	ADE23	ADE22	ADE21	ADE20	ADE19	ADE18	ADE17	ADE16
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	1	1	1	1	1	1	1	1

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	ADE15	ADE14	ADE13	ADE12	ADE11	ADE10	ADE09	ADE08
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	1	1	1	1	1	1	1	1

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	ADE07	ADE06	ADE05	ADE04	ADE03	ADE02	ADE01	ADE00
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	1	1	1	1	1	1	1	1

#### [bit31:0] ADE31 to ADE00: Analog input enable bit

Value	Description
0	Port I/O mode
1	Analog input mode

- These bits control analog input pin.
- When this bit is "0", analog input is disabled.
- When this bit is "1", analog input is enabled.

#### Note:

- For devices equipped with the key code function, key code setting is required for writing to this register. For whether the key code function is equipped, see "Part Number Option" in the "Data Sheet".



### 5.1.2. Key Code Register (KEYCDR)

This is the register for the register (ADER, ADER4CH\_1/ADER4CH\_0) writing settings that include the error writing protection function. If writing to this register is not executed according to the specified method, writing to the target register will become invalid. This register is only enabled for word access. In the type unequipped with the key code function, access to this register has no effect on the operation.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	KEY		SIZE		Reserved			
ACCESS_TYPE	R0,W		R0,W		R0,WX			
PROT_TYPE	-							
INITIAL_VALUE	00		00		0000			

BITS_OFFSET	23	22	21	20	19	18	17	16
BITS_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved				ADR[11:8]			
ACCESS_TYPE	R0,WX				R0,W			
PROT_TYPE	-							
INITIAL_VALUE	0000				0000			

BITS_OFFSET	7	6	5	4	3	2	1	0
BITS_NAME	ADR[7:0]							
ACCESS_TYPE	R0,W							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

#### [bit31:30] KEY[1:0]: Key code bits

These are key code setting bits. Write "0b00", "0b01", "0b10", and "0b11" continuously to these bits in this order. The key code setting becomes invalid immediately upon any writing to these bits in a different order. In such cases, set the key code again from the beginning.

Value	Description
00	1st key code
01	2nd key code
10	3rd key code
11	4th key code

**[bit29:28] SIZE[1:0]: Access size bits**

These bits set the access size for writing to the applicable key code register. Write the same data to these bits when writing key codes "0b00", "0b01", "0b10", and "0b11" in this order.

Value	Description
00	Set byte access.
01	Set half-word access.
10	Set word access.
11	Reserved

**[bit27:12] Reserved: Reserved bits**

**[bit11:0] ADR[11:0]**

These bits set the lower 12 bits of the address of the applicable key code register. Write the same data to these bits when writing key codes "0b00", "0b01", "0b10", and "0b11" in this order.

Value	Description
	Set the lower 12 bits of the address of the applicable key code register.

**Notes:**

- The key code setting becomes invalid upon any writing to the KEY[1:0] in any order than "00", "01", "10", and "11". In such case, set the key code again from the beginning.
- When different data is written to the SIZE[1:0] or the ADR[11:0] while writing the key code "00", "01", "10", and "11", the key code setting will become invalid. In such case, set it again from the beginning.
- This register is valid only for word access.
- There is no effect on the key code protection release process even if registers other than this register and the applicable key code register that is in the release processing (the register set by the ADR) are accessed while writing the key code "00", "01", "10", and "11".



## 5.2. A/D Start Compare Register

### 5.2.1. A/D Software Activation Register (ADTSS)

The A/D software activation register (ADTSS) is the register that makes A/D activation requests for the 12-bit A/D converter. The A/D software activation channel selection register (ADTSE) controls which channel to activate.

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved							START
ACCESS_TYPE	R0,W0							R0,W
PROT_TYPE	-							
INITIAL_VALUE	0000000							0

[bit7:1] Reserved: Reserved bits

[bit0] START: A/D conversion activation bit (software)

Value	Description	
	Read	Write
0	"0" is always read.	Do not activate A/D conversion.
1		Activate A/D conversion.

- This bit activates the A/D conversion operation with software.
- Writing "1" to the START bit activates A/D conversion. The A/D software activation channel selection register (ADTSE) controls which channel to activate.
- The START bit cannot restart conversion.

## 5.2.2. A/D Software Activation Channel Selection Register (ADTSE)

The A/D software activation channel selection register (ADTSE) is the register that selects the activation channel used for A/D activation requests.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	ADT31	ADT30	ADT29	ADT28	ADT27	ADT26	ADT25	ADT24
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	ADT23	ADT22	ADT21	ADT20	ADT19	ADT18	ADT17	ADT16
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	ADT15	ADT14	ADT13	ADT12	ADT11	ADT10	ADT9	ADT8
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	ADT7	ADT6	ADT5	ADT4	ADT3	ADT2	ADT1	ADT0
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

### [bit31:0] ADT31 to ADT0: Software activation channel selection bits

Value	Description
0	Disable software activation.
1	Enable software activation.

- Software activation is controlled for each activation channel. If software activation is enabled for multiple activation channels, the activation channels can be software-activated simultaneously.
- If the ADTSE:ADT bit is "0", software activation is disabled.
- If the ADTSE:ADT bit is "1", software activation is enabled.





### 5.2.3. Compare Buffer Register [n] (ADCOMPB[n]) / Compare Register (ADCOMP[n]) (n=0 to 31)

Each compare buffer register (ADCOMPB) is the 16-bit buffer register used by a compare register (ADCOMP).

The A/D converter is activated when the compare register (ADCOMP) matches the 16-bit free-run timer value.

The ADCOMPB register and the ADCOMP register have the same address.

#### (1) Compare Buffer Register (ADCOMPB[n], n=0 to 31)

BIT_OFFSET	15-0
BIT_NAME	CMP
ACCESS_TYPE	W
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000

#### [bit15:0] CMP[15:0]: Compare value buffer bits

Value	Description
	Compare value buffer

- Each compare buffer register is the buffer register used by a compare register (ADCOMP[n]).
- If the buffer function is disabled (compare register buffer function control bit (ADTCS:BUFX) is "1") or the 16-bit free-run timer is stopped, the set value of the compare value buffer is immediately transferred to the compare register (ADCOMP).
- If the buffer function is enabled (compare register buffer function control bit (ADTCS:BUFX) is "0"), the set value of the compare value buffer is transferred to the compare register (ADCOMP) when a match with the compare clear register of the 16-bit free-run timer is found or when 0 is detected.

#### Note:

- To access a compare buffer register, use half-word or word access instructions.

**(2) Compare Register (ADCOMP[n], n=0 to31)**

BIT_OFFSET	15-0
BIT_NAME	CMP
ACCESS_TYPE	R
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000

**[bit15:0] CMP[15:0]: Compare value bits**

Value	Description
	Compare value

- The compare value in a compare register (ADCOMP) is updated through the compare buffer register (ADCOMPB).
- The compare register (ADCOMP) stores the compare value used for comparison with the count value of the 16-bit free-run timer. An A/D activation request is output when the 16-bit free-run timer matches the compare value.
- The compare value stored in the compare register is immediately compared with the 16-bit free-run timer.
- The compare match activation operation is not performed when the count direction selection bits (ADTCS[n]:SEL[1:0]) are "0b11".

**Note:**

- *To access a compare buffer register, use half-word or word access instructions.*



### 5.2.4. A/D Activation Trigger Control Status Register [n] (ADTCS[n]) (n=0 to 31)

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	BUSY	INT	INTE	STS		RPT	PRT	PRTS
ACCESS_TYPE	R,W	R,W	R/W	R/W		R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	00		0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	SEL		BUFX	BTS	Reserved			
ACCESS_TYPE	R/W		R/W	R/W	R0,W0			
PROT_TYPE	-							
INITIAL_VALUE	00		1	0	0000			

#### [bit15] BUSY: A/D activation request in progress bit

Value	Description	
	Read	Write
0	A/D activation not requested	A/D activation request forced stop
1	A/D activation request or conversion in progress	No change and no influence on others

- This bit indicates an A/D activation request or conversion operation.
- In reading, if BUSY bit is "0", it indicates that no A/D conversion request has been issued; if it is "1", it indicates that A/D conversion request or A/D conversion is in progress
- In writing, the current A/D activation request or conversion operation is forced to termination by writing "0" to this bit. Writing "1" to this bit does not make any change and influence on others.
- This bit is cleared to "0" by writing "1" to BUSYC bit in the ADTCSC[n] register.

#### [bit14] INT: Interrupt request flag bit

Value	Description	
	Read	Write
0	A/D conversion not completed	Clears the bit
1	A/D conversion completed	No change and no influence on others

- When data is set in the A/D data register (ADTCD) as a result of A/D conversion, INT bit is set to "1".
- An interrupt request is generated when this bit and the interrupt request enable bit (ADTCS: INTE) are "1".
- This bit is cleared when "0" is written to it. A write of "1" does not change this bit and has no influence on others.
- INT bit is cleared when the A/D conversion completion interrupt clear signal is "H".
- This bit is cleared to "0" by writing "1" to INTC bit in the ADTCSC[n] register.

#### Note:

- If a software clear (INT of "0" is written) or clear by the interrupt clear signal ("H") occurs at the same time as a hardware set, the hardware set has priority.

**[bit13] INTE: Interrupt request enable bit**

Value	Description
0	Disable interrupt request output.
1	Enable interrupt request output.

- This bit controls the enabling/disabling of interrupt request output to the CPU.
- An interrupt request is generated when both the INTE bit and the interrupt request flag bit (ADTCS:INT) are "1".

**[bit12:11] STS[1:0]: A/D activation factor selection bits**

Value	Description
00	Software activation
01	External trigger activation (falling edge)
10	Base timer activation (rising edge)
11	Compare match activation

- STS1[1:0] bits select an activation factor for A/D conversion.

**Note:**

- The A/D activation factor selection bits (STS[1:0]) are updated at the same time that they are overwritten. Update the A/D activation factor selection bits (STS[1:0]) while the currently selected activation factor and the new activation factor are inactive and while no A/D conversion request is being processed (ADTCS:BUSY is "1").
- To not perform A/D activation requests, set the A/D activation factor selection bits (ADTCS:STS[1:0]) for software activation ("0b00"), and disable software activation (ADTSE:ADT is "0") in the corresponding bit (activation channel) in the A/D software activation channel selection register (ADTSE).
- When setting the A/D activation factor selection bits (STS[1:0]), be sure that the 16-bit free-run timer has stopped.

**[bit10] RPT: Repeat conversion selection bit**

Value	Description
0	Single conversion mode
1	Repeat conversion mode

- This bit sets the A/D conversion mode.
- If the RPT bit is set to "0", the mode is single conversion mode. A 1-time activation factor causes a 1-time A/D conversion request. The A/D conversion is done once.
- If the RPT bit is set to "1", the mode is repeat conversion mode. A 1-time activation factor causes an A/D conversion request to be made continuously. A/D conversion is performed repeatedly until single conversion mode is set.

**[bit9] PRT: A/D data register protection enable bit**

Value	Description
0	Disable protection.
1	Enable protection.

- If the PRT bit is set to "1", the A/D data register is write-protected. The protection function runs for factors other than compare match activation (ADTCS:STS[1:0] is "0b11").
- After conversion data is stored in the A/D data register, subsequent activation requests are masked and the A/D data register is write-protected until an occurrence of the factor that is set by the A/D data register protection factor selection bit (ADTCS:PRTS).



**Note:**

- Set the A/D data register protection enable bit (PRT) before the A/D conversion operation. The setting for the A/D data register protection enable bit (PRT) cannot be modified during an A/D conversion request (ADTCS:BUSY is "1") or while the A/D data register is protected.

**[bit8] PRTS: A/D data register protection clear selection bit**

Value	Description
0	Data reading and interrupt flag clearing
1	Data reading

- This bit selects the conditions for canceling the masking of activation requests when A/D data register protection function is enabled (ADTCS:PRT is "1").
- If the PRTS bit is set to "0", the conditions (in random order) for canceling the protection are the reading of an A/D data register (ADTCD) and the clearing of an interrupt request flag bit (ADTCS:INT).
- If the PRTS bit is set to "1", the condition for canceling the protection is the reading of an A/D data register (ADTCD).

**Note:**

- Set the A/D data register protection clear selection bit (PRTS) before the A/D conversion operation. The setting for the A/D data register protection clear selection bit (PRTS) cannot be modified during an A/D conversion request (ADTCS:BUSY is "1") or while the A/D data register is protected.

**[bit7:6] SEL[1:0]: Count direction selection bits**

Value	Description
00	When counting in either direction (up/down)
01	Only when counting up
10	Only when counting down
11	Compare disabled

- These bits select the compare match conditions for the 16-bit free-run timer.
- If the SEL[1:0] bits are set to "0b00", the compare match operation is performed any time that the 16-bit free-run timer is counting up/down.
- If the SEL[1:0] bits are set to "0b01", the compare match activation operation is performed only when the 16-bit free-run timer is counting up.
- If the SEL[1:0] bits are set to "0b10", the compare match activation operation is performed only when the 16-bit free-run timer is counting down.
- If the SEL[1:0] bits are set to "0b11", the compare operation is not performed.
- The compare operation is not performed while the selected 16-bit free-run timer is stopped.

**Note:**

- Setting the count direction selection bits (ADTC[n]:SEL[1:0]) to "0b10" is prohibited when the 16-bit free-run timer is in the mode of counting up.

**[bit5] BUFX: Compare register buffer function control bit**

Value	Description
0	Enable the buffer function.
1	Disable the buffer function.

- If the ADTC:BUFX bit is set to "0", the buffer function is enabled.
- If the ADTC:BUFX bit is set to "1", the buffer function is disabled.

**[bit4] BTS: Compare register buffer transfer control bit**

Value	Description
0	0 detection time
1	Compare clear time

- This bit sets the transfer condition when the compare register buffer function is enabled (ADTCS:BUFX is "0").
- The BTS setting is enabled when the compare register buffer function is enabled (ADTCS:BUFX is "0").
- If the ADTCS:BTS bit is set to "0", the compare value of a compare buffer register (ADCOMPB) is transferred to a compare register (ADCOMP) when 0 is detected in the 16-bit free-run timer.
- If the ADTCS:BTS bit is set to "1", the compare value of a compare buffer register (ADCOMPB) is transferred to a compare register (ADCOMP) when the compare clear register matches the 16-bit free-run timer.

**Note:**

- *When setting the compare register buffer transfer control bit (BTS), be sure that the 16-bit free-run timer has stopped.*

**[bit3:0] Reserved: Reserved bits**



### 5.2.5. A/D Data Register [n] (ADTCD[n]) (n=0 to 31)

Each A/D data register (ADTCD) is a register for storing A/D conversion results.

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	ERR	ERRST	Reserved			D[11:8]		
ACCESS_TYPE	R,WX	R,WX	R0,W0			R,WX		
PROT_TYPE	-							
INITIAL_VALUE	1	0	00		0000			

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	D[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

#### [bit15] ERR: Conversion data error flag bit

Value	Description
0	Conversion data is normal.
1	Conversion data is abnormal.

- This bit indicates the existence of an error regarding A/D conversion data. The details of the error can be checked using the conversion data error status bit (ADTCD:ERRST) when the ERR bit is "1".
- For details on the ERR bit operation, see Section "3.2.10. A/D Conversion Data".
- If the A/D data register protection function is enabled (ADTCS:PRT is "1") when the factor is other than compare match activation (ADTCS:STS[1:0] is "0b11"), "0" is read.

#### [bit14] ERRST: Conversion data error status bit (enabled only if ERR="1")

Value	Description
0	The conversion data is old results.
1	The conversion data has been overwritten by new data.

- This bit is a flag indicating the details of an A/D conversion data error when the ERR bit is "1".
- The ERR bit of "1" and ERRST bit of "0" indicate that the conversion results read by the CPU are old.
- The ERR bit of "1" and ERRST bit of "1" indicate that the conversion results read by the CPU was overwritten by new conversion results before the CPU finished reading the old conversion results. Consequently, the old conversion result data was lost.
- For details on the ERRST bit operation, see Section "3.2.10. A/D Conversion Data".
- If the A/D data register protection function is enabled (ADTCS:PRT is "1") when the factor is other than compare match activation (ADTCS:STS[1:0] is "0b11"), "0" is read.

#### [bit13:12] Reserved: Reserved bits

**[bit11:0] D[11:0]: A/D data bits**

Value	Description
	Conversion data

- The A/D data bits (D[11:0]) store A/D conversion results. These bits are overwritten each time that conversion ends.
- Normally, the bits store the final conversion value.

**Note:**

- *Do not write to A/D data register (ADTCD).*





## 5.2.6. A/D Activation Trigger Extend Control Register [n] (ADTECS[n]) (n=0 to 31)

The A/D activation trigger extend control register (ADTECS) selects analog input channel.

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved							Reserved
ACCESS_TYPE	R0,W0							R/W0
PROT_TYPE	-							
INITIAL_VALUE	0000000							0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved			CHSEL				
ACCESS_TYPE	R0,W0			R/W				
PROT_TYPE	-							
INITIAL_VALUE	000			00000				

[bit15:9] Reserved: Reserved bits

[bit8] Reserved: Reserved bit

[bit7:5] Reserved: Reserved bits

[bit4:0] CHSEL[4:0]: Analog channel selection bits

Value	Description
00000	Channel 0
00001	Channel 1
:	:
00110	Channel 6
00111	Channel 7
01000	Channel 8
:	:
01110	Channel 14
01111	Channel 15
10000	Channel 16
:	:
11110	Channel 30
11111	Channel 31

These bits select the analog channel with the specified value.

**Note:**

- Do not change the analog channel select bits (CHSEL[[4:0]]) when A/D conversion is being requested (ADTCS:BUSY="1").

### 5.2.7. Upper-limit Threshold Setting Register 0 to 3 (ADRCUT0 to 3)

Each upper-limit threshold setting register (ADRCUT) sets the upper-limit threshold value used in range comparisons. The register can have 4 types of settings for the upper-limit threshold value.

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved				C[11:8]			
ACCESS_TYPE	R0,W0				R/W			
PROT_TYPE	-							
INITIAL_VALUE	0000				0000			

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	C[7:0]							
ACCESS_TYPE	R/W							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

[bit15:12] Reserved: Reserved bits

[bit11:0] C[11:0]: Upper-limit threshold bits

Value	Description
	Upper-limit threshold value

- These bits set the upper-limit threshold value used in range comparisons.

**Note:**

- The upper-limit threshold bits (C[11:0]) cannot be modified during an A/D conversion request (ADTCS:BUSY is "1").



### 5.2.8. Lower-limit Threshold Setting Register 0 to 3 (ADRCLT0 to 3)

Each lower-limit threshold setting register (ADRCLT) sets the lower-limit threshold value used in range comparisons. The register can have 4 types of settings for the lower-limit threshold value.

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved				C[11:8]			
ACCESS_TYPE	R0,W0				R/W			
PROT_TYPE	-							
INITIAL_VALUE	0000				0000			

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	C[7:0]							
ACCESS_TYPE	R/W							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

[bit15:12] Reserved: Reserved bits

[bit11:0] C[11:0]: Lower-limit threshold bits

Value	Description
	Lower-limit threshold value

- These bits set the lower-limit threshold value used in range comparisons.

**Note:**

- The lower-limit threshold bits (C[11:0]) cannot be modified during an A/D conversion request (ADTCS:BUSY is "1").

### 5.2.9. Range Comparison Control Status Register [n] (ADRCSS[n]) (n=0 to 31)

Each range comparison control status register (ADRCSS) checks the continuous detection frequency specification and count, selects inside/outside-the-range confirmation, enables/disables range compare interrupt requests, enables/disables range comparison execution, and selects the upper- and lower-limit threshold values.

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	RCOCD			RCOIRS	RCOIE	RCOE	RCOTS	
ACCESS_TYPE	R,W			R/W	R/W	R/W	R/W	
PROT_TYPE	-							
INITIAL_VALUE	000			0	0	0	00	

#### [bit7:5] RCOCD[2:0]: Continuous detection frequency specification and count display bits

Value	Description	
	Read	Write
000	Continuous detection state: 0 times (initial value)	Setting prohibited
001	Continuous detection state: 1 time	Continuous detection specified 1 time (initial value)
010	Continuous detection state: 2 times	Continuous detection specified 2 times
011	Continuous detection state: 3 times	Continuous detection specified 3 times
100	Continuous detection state: 4 times	Continuous detection specified 4 times
101	Continuous detection state: 5 times	Continuous detection specified 5 times
110	Continuous detection state: 6 times	Continuous detection specified 6 times
111	Continuous detection state: 7 times	Continuous detection specified 7 times

- These bits specify the continuous detection frequency and display the continuous detection count for a range comparison result.
- The range compare interrupt factor flag bit (ADRCIF:RCINT[n]) of the corresponding activation channel is set to "1" when a range comparison result reaches the specified value for the continuous count. Also, the continuous detection state stops at the specified value for the continuous count.
- At the read time, the continuous detection state is read, rather than the continuous detection frequency specification that is set for the write time.

#### Notes:

- The continuous detection frequency specification and count display bits (RCOCD[2:0]) cannot be modified during an A/D conversion request (ADTCS:BUSY is "1").
- The continuous detection frequency specification and count display bits (RCOCD[2:0]) cannot be modified while range comparison execution is enabled (ADRCSS[n]:RCOE is "1").
- The continuous detection frequency specification and count display bits (RCOCD[2:0]) cannot be set to "0b000".



**[bit4] RCOIRS: Outside/Inside range confirmation selection bit**

Value	Description
0	Confirm that the result is outside the range.
1	Confirm that the result is inside the range.

- This bit selects outside the range or inside the range as a range comparison condition for A/D data bits (ADTCD:D[11:0]). The range corresponds to the upper-limit threshold bits (ADRCUT:C[11:0]) and lower-limit threshold bits (ADRCLT:C[11:0]) selected by the upper- and lower-limit threshold selection bits (ADRCCS[n]:RCOTS[1:0]).
- For outside-the-range confirmation (ADRCCS[n]:RCOIRS is "0"), the range comparison condition is as follows.
  - *A/D data bits (ADTCD:D[11:0]) > Upper-limit threshold bits (ADRCUT:C[11:0])*  
or  
*A/D data bits (ADTCD:D[11:0]) < Lower-limit threshold bits (ADRCLT:C[11:0])*
- For inside-the-range confirmation (ADRCCS[n]:RCOIRS is "1"), the range comparison condition is as follows.
  - *A/D data bits (ADTCD:D[11:0]) ≤ Upper-limit threshold bits (ADRCUT:C[11:0])*  
and  
*A/D data bits (ADTCD:D[11:0]) ≥ Lower-limit threshold bits (ADRCLT:C[11:0])*
- For a range comparison detection with outside-the-range confirmation (RCOIRS is "0"), the out-of-range flag bit (ADRCOT:RCOOF) enables checking of whether the result is above the upper-limit threshold value or below the lower-limit threshold value.

**Note:**

- The outside/inside range confirmation selection bit (RCOIRS) cannot be modified during an A/D conversion request (ADTCS:BUSY is "1").

**[bit3] RCOIE: Range compare interrupt request enable bit**

Value	Description
0	Disable range compare interrupts.
1	Enable range compare interrupts.

- If the range compare interrupt factor flag bit (ADRCIF:RCINT[n]) of the corresponding activation channel is set to "1" and range compare interrupt requests are enabled (RCOIE is "1"), an interrupt request is generated.

**[bit2] RCOE: Range comparison execution enable bit**

Value	Description
0	Disable range comparison execution.
1	Enable range comparison execution.

- This bit selects whether to enable or disable range comparison execution.
- Range comparison execution is disabled when the range comparison execution enable bit (RCOE) is "0". Also, the continuous detection count is initialized to "0b000".
- The range comparison execution is enabled when the range comparison execution enable bit (RCOE) is "1".

**[bit1:0] RCOTS[1:0]: Upper- and lower-limit threshold selection bits**

Value	Description
00	Upper limit threshold setting register 0/lower limit threshold setting register 0 selected
01	Upper limit threshold setting register 1/lower limit threshold setting register 1 selected
10	Upper limit threshold setting register 2/lower limit threshold setting register 2 selected
11	Upper limit threshold setting register 3/lower limit threshold setting register 3 selected

- These bits select 1 combination from the 4 types available with upper-limit threshold setting registers 0 to 3 (ADRCUT0 to 3) and lower-limit threshold setting registers 0 to 3 (ADRCLT0 to 3).

**Note:**

- *The upper- and lower-limit threshold selection bits (RCOTS[1:0]) cannot be modified during an A/D conversion request (ADTCS:BUSY is "1").*



### 5.2.10. Range Comparison Out-of-range Flag Register (ADRCOT)

The range comparison out-of-range flag register (ADRCOT) indicates whether the result of a range comparison with the setting of outside the range is over the upper-limit threshold value or below the lower-limit threshold value.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	RCOOF31	RCOOF30	RCOOF29	RCOOF28	RCOOF27	RCOOF26	RCOOF25	RCOOF24
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	RCOOF23	RCOOF22	RCOOF21	RCOOF20	RCOOF19	RCOOF18	RCOOF17	RCOOF16
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	RCOOF15	RCOOF14	RCOOF13	RCOOF12	RCOOF11	RCOOF10	RCOOF9	RCOOF8
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	RCOOF7	RCOOF6	RCOOF5	RCOOF4	RCOOF3	RCOOF2	RCOOF1	RCOOF0
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

#### [bit31:0] RCOOF31 to RCOOF0: Out-of-range flag bits

Value	Description
0	Below lower-limit threshold value (A/D data < Lower-limit threshold bits)
1	Above upper-limit threshold value (A/D data > Upper-limit threshold bits)

- For outside-the-range confirmation (ADRCSS:RCOIRS is "0"), these bit indicate whether the range comparison result is larger than the upper-limit threshold setting register (RCOOF is "1") or smaller than the lower-limit threshold setting register (RCOOF is "0").
- With outside-the-range confirmation (ADRCSS:RCOIRS is "0"), if the range comparison result is inside the range, the out-of-range flag bit retains its previous value.
- Even with outside-the-range confirmation (ADRCSS:RCOIRS is "0") where the range comparison result has been detected as outside the range, if the range compare interrupt factor flag bit (ADRCIF:RCINT[n]) of the corresponding activation channel is set to "1", the out-of-range flag bit (RCOOF) is not updated and retains its previous value.
- For inside-the-range confirmation (ADRCSS:RCOIRS is "1"), the out-of-range flag bit holds no meaning. (The previous value is retained.)

### 5.2.11. Range Compare Flag Register (ADRCIF)

The range comparison flag register (ADRCIF) indicates an interrupt factor resulting from the continuous detection of a range comparison result. For details on writing to this register, see Section "6. Precautions for Using".

BITS_OFFSET	31	30	29	28	27	26	25	24
BITS_NAME	RCINT31	RCINT30	RCINT29	RCINT28	RCINT27	RCINT26	RCINT25	RCINT24
ACCESS_TYPE	R,W	R,W	R,W	R,W	R,W	R,W	R,W	R,W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BITS_OFFSET	23	22	21	20	19	18	17	16
BITS_NAME	RCINT23	RCINT22	RCINT21	RCINT20	RCINT19	RCINT18	RCINT17	RCINT16
ACCESS_TYPE	R,W	R,W	R,W	R,W	R,W	R,W	R,W	R,W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BITS_OFFSET	15	14	13	12	11	10	9	8
BITS_NAME	RCINT15	RCINT14	RCINT13	RCINT12	RCINT11	RCINT10	RCINT9	RCINT8
ACCESS_TYPE	R,W	R,W	R,W	R,W	R,W	R,W	R,W	R,W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BITS_OFFSET	7	6	5	4	3	2	1	0
BITS_NAME	RCINT7	RCINT6	RCINT5	RCINT4	RCINT3	RCINT2	RCINT1	RCINT0
ACCESS_TYPE	R,W	R,W	R,W	R,W	R,W	R,W	R,W	R,W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

#### [bit31:0] RCINT31 to RCINT0: Conversion data error flag bits

Value	Description	
	Read	Write
0	Range compare interrupt factor clear state	Clear of bit
1	Interrupt factor resulting from the continuous detection of a range comparison result	This bit value does not change and there is no influence on an operation by writing.

- The continuous detection of a range comparison result in the corresponding activation channel sets the RCINT[n] bit to "1".
- A range compare interrupt request is generated when both the RCINT[n] bit of the corresponding activation channel and the range compare interrupt request enable bit (ADRCES[n]:RCOIE) are "1".
- The RCINT bit is cleared when "0" is written to this bit. A write of "1" does not change this bit and has no influence on others.
- This bit is cleared to "0" by writing "1" to RCINTC bit in ADRCIFC register.

#### Note:

- If a software clear (RCINT of "0" is written) occurs at the same time as a hardware set, the hardware set has priority.





### 5.2.12. Scan Conversion Control Status Register 0 (ADSCANS0)

The scan conversion control status register (ADSCANS) selects continuous or stop scan mode when the conversion count is specified, selects scan conversion completion interrupt request enable/disable, and indicates the state of the scan conversion completion interrupt request. For details on writing to this register, see Section "6. Precautions for Using".

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	SCINT	SCIE	SCMD	Reserved				
ACCESS_TYPE	R,W	R/W	R/W	R0,W0				
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	00000				

#### [bit7] SCINT: Scan conversion completion interrupt factor flag bit

Value	Description	
	Read	Write
0	Scan conversion completion interrupt factor clear state when conversion count of each channel is specified	Clear of bit
1	State of interrupt factor generation by scan conversion completion when conversion count of each channel is specified	This bit value does not change and there is no influence on an operation by writing.

- The SCINT bit is set to "1" by the scan conversion completion when the conversion count of each channel is specified.
- When SCINT bit and scan conversion completion interrupt request enable bit are "1", the scan conversion completion interrupt request when the conversion count is specified is generated.
- The SCINT bit is cleared when "0" is written to this bit. A write of "1" does not change this bit and has no influence on others.
- This bit is cleared to "0" by writing "1" to SCINTC bit in ADSCANS0 register.

#### Note:

- The hardware setting is given priority when software clearing (SCINT="0" writing) and the hardware setting are generated at the same time.

#### [bit6] SCIE: Scan conversion completion interrupt request enable bit

Value	Description
0	Scan conversion completion interrupt disabled
1	Scan conversion completion interrupt enabled

- When the scan conversion completion interrupt factor flag bit and the scan conversion completion interrupt request enable bit are set in "1", the interrupt request is generated.
- This bit is cleared to "0" by writing "1" to SCIEC bit in ADSCANS0 register.
- This bit is set to "1" by writing "1" to SCIES bit in ADSCANS0 register.

**[bit5] SCMD: Continuous and stop scan conversion mode select bit**

Value	Description
0	Continuous scan conversion mode
1	Stop scan conversion mode

- The scan conversion mode is selected when the conversion count of each channel is specified.
- When SCMD bit is "0", continuous scan conversion mode is selected.
- When SCMD bit is "1", stop scan conversion mode is selected.
- This bit is cleared to "0" by writing "1" to SCMDC bit in ADSCANSC0 register.
- This bit is set to "1" by writing "1" to SCMDS bit in ADSCANSS0 register.

**[bit4:0] Reserved: Reserved bits**



### 5.2.13. Activation Channel Conversion Count Setting Register [m] (ADNCS[m]) (m=0 to 15)

The activation channel conversion count setting register (ADNCS) sets conversion count specification scan conversion execution enable/disable and specifies the conversion count for each activation channel.

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	CNTEN (m*2+1)	Reserved	CCNT (m*2+1)1	CCNT (m*2+1)0	CNTEN (m*2)	Reserved	CCNT (m*2)1	CCNT (m*2)0
ACCESS_TYPE	R/W	R0,W0	R/W	R/W	R/W	R0,W0	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

#### [bit7, bit3] CNTEN[n]: Conversion count specification scan conversion execution enable bit (n=0 to 31)

Value	Description
0	Conversion count specification scan conversion execution disabled
1	Conversion count specification scan conversion execution enabled

(n=0 to 31)

- These bits can set conversion count specification scan conversion execution enable/disable for each activation channel.
- Only one scan conversion group can be controlled by the scan conversion by conversion count specification for each channel, in units of 12-bit A/D converter unit.
- An activation channel for which the conversion count-specified scan conversion enable bit (CNTEN) is set to "0" is exempted from conversion count-specified scan conversion.
- An activation channel for which the conversion count-specified scan conversion enable bit (CNTEN) is set to "1" is subject to conversion count-specified scan conversion.

#### Notes:

- When enabling conversion count-specified scan conversion (CNTEN="1"), set the mode to the repeat conversion mode (ADRCs:RPT="1"). If the mode is other than the repeat conversion mode, conversion count-specified scan conversion will not be performed normally.
- Do not change the setting of the conversion count-specified scan conversion enable bit (CNTEN[n]) while the corresponding activation channel is requesting A/D conversion (ADTCS[n]:BUSY="1").

#### [bit6, bit2] Reserved: Reserved bits

**[bit5, bit4, bit1, bit0] CCNT[n]1, CCNT[n]0: Conversion count specification bits (n=0 to 31)**

CCNT[n]1, CCNT[n]0	Description
00	1 time of conversion count
01	2 times of conversion count
10	3 times of conversion count
11	4 times of conversion count

(n=0 to 31)

- When the conversion count specification scan conversion execution of the activation channel is enabled (ADNCS:CNTEN="1"), the scan conversion count is controlled.

**Note:**

- Do not change the conversion count specification bit (CCNT[n]1, CCNT[n]0) while the corresponding activation channel is requesting the A/D conversion (ADTCS[n]:BUSY="1").



### 5.2.14. Data Protection Status Flag Register (ADPRTF)

The protected data state flag register (ADPRTF) indicates the protection state of each activation channel A/D data register (ADTCD).

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	PRTF31	PRTF30	PRTF29	PRTF28	PRTF27	PRTF26	PRTF25	PRTF24
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	PRTF23	PRTF22	PRTF21	PRTF20	PRTF19	PRTF18	PRTF17	PRTF16
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	PRTF15	PRTF14	PRTF13	PRTF12	PRTF11	PRTF10	PRTF9	PRTF8
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	PRTF7	PRTF6	PRTF5	PRTF4	PRTF3	PRTF2	PRTF1	PRTF0
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

#### [bit31:0] PRTF31 to PRTF0: Protected data state flag bits

Value	Description
0	Data is not protected.
1	Data is protected.

- These bits indicate the data protected state of the A/D data register (ADTCD[n]) of each activation channel.
- The writing operation doesn't influence the data protection state.

### 5.2.15. Activation Channel Conversion Completion Flag Register (ADEOCF)

The activation channel conversion completion flag register (ADEOCF) indicates the conversion count completion for each activation channel when the conversion count specification scan is converted.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	EOCF31	EOCF30	EOCF29	EOCF28	EOCF27	EOCF26	EOCF25	EOCF24
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	1	1	1	1	1	1	1	1

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	EOCF23	EOCF22	EOCF21	EOCF20	EOCF19	EOCF18	EOCF17	EOCF16
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	1	1	1	1	1	1	1	1

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	EOCF15	EOCF14	EOCF13	EOCF12	EOCF11	EOCF10	EOCF9	EOCF8
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	1	1	1	1	1	1	1	1

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	EOCF7	EOCF6	EOCF5	EOCF4	EOCF3	EOCF2	EOCF1	EOCF0
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	1	1	1	1	1	1	1	1

#### [bit31:0] EOCF31 to EOCF0: Conversion count completion flag bits

Value	Description
0	The count of the conversion count specification is not completed.
1	The count of the conversion count specification is completed.

- When the conversion count-specified scan conversion is enabled (ADNCS:CNTEN[n] = "1") for the corresponding activation channel, the indication of the conversion count completion flag bit (EOCF) is valid.
- The state where the scan conversion has been completed the specified number of times (ADNCS:CCNT[1:0]) is indicated.
- If EOCF is "0", the scan conversion has not been completed the specified number of times.
- If EOCF is "1", the scan conversion has been completed the specified number of times.
- The conversion count completion flag bit (EOCF) is cleared to "0" when A/D activation is not requested (ADTCS:BUSY = "0") or the scan conversion resumes from the beginning.
- The writing operation doesn't influence the data protection state.



### 5.2.16. A/D Activation Trigger Control Status Clear Register [n] (ADTCSC[n]) (n=0 to 31)

A/D activation trigger control status clear register (ADTCSC[n] (n=0 to 31)) is used to clear bit of A/D activation trigger control status register (ADTCS[n] (n=0 to 31)).

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	BUSYC	INTC	INTEC	Reserved		RPTC	PRTC	PRTSC
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W0		R0,W	R0,W	R0,W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	00		0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved		BUFXC	BTSC	Reserved			
ACCESS_TYPE	R0,W0		R0,W	R0,W	R0,W0			
PROT_TYPE	-							
INITIAL_VALUE	00		0	0	0000			

#### [bit15] BUSYC: A/D activation request in progress clear bit

Value	Description
0	No effect
1	Clear the BUSY bit

#### [bit14] INTC: Interrupt request flag clear bit

Value	Description
0	No effect
1	Clear the INT bit

#### [bit13] INTEC: Interrupt request enable clear bit

Value	Description
0	No effect
1	Clear the INTE bit

#### [bit12:11] Reserved: Reserved bits

#### [bit10] RPTC: Repeat conversion selection clear bit

Value	Description
0	No effect
1	Clear the PRT bit

#### [bit9] PRTC: A/D data register protection enable clear bit

Value	Description
0	No effect
1	Clear the PRT bit

**[bit8] PRTSC: A/D data register protection clear selection clear bit**

Value	Description
0	No effect
1	Clear the PRTS bit

**[bit7:6] Reserved: Reserved bits**

**[bit5] BUFXC: Compare register buffer function control clear bit**

Value	Description
0	No effect
1	Disable

**[bit4] BTSC: Compare register buffer transfer control clear bit**

Value	Description
0	No effect
1	Clear the BTS bit

**[bit3:0] Reserved: Reserved bits**





### 5.2.17. Range Compare Flag Clear Register (ADRCIFC)

Range compare flag clear register (ADRCIFC) is used to clear bit of range compare flag register (ADRCIF).

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	RCINTC31	RCINTC30	RCINTC29	RCINTC28	RCINTC27	RCINTC26	RCINTC25	RCINTC24
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	RCINTC23	RCINTC22	RCINTC21	RCINTC20	RCINTC19	RCINTC18	RCINTC17	RCINTC16
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	RCINTC15	RCINTC14	RCINTC13	RCINTC12	RCINTC11	RCINTC10	RCINTC9	RCINTC8
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	RCINTC7	RCINTC6	RCINTC5	RCINTC4	RCINTC3	RCINTC2	RCINTC1	RCINTC0
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

#### [bit31:0] RCINTC31 to RCINTC0: Conversion data error flag clear bits

Value	Description
0	No effect
1	Clear the RCINT31 to RCINTC0.

### 5.2.18. Scan Conversion Control Status Clear Register 0 (ADSCANSC0)

Scan conversion control status clear register 0 (ADSCANSC0) is used to clear bit of Scan conversion control status register 0 (ADSCANS0).

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	SCINTC	SCIEC	SCMDC	Reserved				
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W0				
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	00000				

#### [bit7] SCINTC: Scan conversion completion interrupt factor flag clear bit

Value	Description
0	No effect
1	Clear the SCINT bit.

#### [bit6] SCIEC: Scan conversion completion interrupt request enable clear bit

Value	Description
0	No effect
1	Clear the SCIE bit

#### [bit5] SCMDC: Continuous and stop scan conversion mode select clear bit

Value	Description
0	No effect
1	Clear SCMD bit

#### [bit4:0] Reserved: Reserved bits



### 5.2.19. A/D Activation Trigger Control Status Set Register (ADTCSS[n]) (n=0 to 31)

A/D activation trigger control status set register (ADTCSS[n] (n=0 to 31)) is used to set bit of A/D activation trigger control status register (ADTCS[n] (n=0 to 31)).

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved		INTES	Reserved		RPTS	PRTS	PRTSS
ACCESS_TYPE	R0,W0		R0,W	R0,W0		R0,W	R0,W	R0,W
PROT_TYPE	-							
INITIAL_VALUE	00		0	00		0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved		BUFXS	BTSS	Reserved			
ACCESS_TYPE	R0,W0		R0,W	R0,W	R0,W0			
PROT_TYPE	-							
INITIAL_VALUE	00		0	0	0000			

[bit15:14] Reserved: Reserved bits

[bit13] INTES: Interrupt request enable set bit

Value	Description
0	No effect
1	Set the INTE bit

[bit12:11] Reserved: Reserved bits

[bit10] RPTS: A/D data register protection enable set bit

Value	Description
0	No effect
1	Set the RPT bit

[bit9] PRTS: Repeat conversion selection set bit

Value	Description
0	No effect
1	Set the PRT bit

[bit8] PRTSS: A/D data register protection clear selection set bit

Value	Description
0	No effect
1	Set the PRTS bit

[bit7:6] Reserved: Reserved bits

**[bit5] BUFXS: Compare register buffer function control set bit**

Value	Description
0	No effect
1	Set the BUFX bit

**[bit4] BTSS: Compare register buffer transfer control set bit**

Value	Description
0	No effect
1	Set the BTS bit

**[bit3:0] Reserved: Reserved bits**



### 5.2.20. Scan Conversion Control Status Set Register 0 (ADSCANSS0)

Scan conversion control status set register 0 (ADSCANSS0) is used to set bit of Scan conversion control status register 0 (ADSCANS0).

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved	SCIES	SCMDS	Reserved				
ACCESS_TYPE	R0,W0	R0,W	R0,W	R0,W0				
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	00000				

**[bit7] Reserved: Reserved bit**

**[bit6] SCIES: Scan conversion completion interrupt request enable set bit**

Value	Description
0	No effect
1	Set SCIE bit

**[bit5] SCMDS: Continuous and stop scan conversion mode select set bit**

Value	Description
0	No effect
1	Set the SCMDS bit

**[bit4:0] Reserved: Reserved bits**

## 6. Precautions for Using

This section explains cautions on A/D activation compare usage.

### (1) Notes When Accessing a Register

#### a) When Accessing to the A/D Activation Trigger Control Status Register (ADTCS[n] (n=0 to 31))

- This register supports writing from the bit-band alias area. For the bit-band alias area, see "CHAPTER: Bit-band Unit".
- To clear a specified bit of this register, clear the bit by writing "1" to the applicable bit of the A/D activation trigger control status clear register (ADTCSC[n] (n=0 to 31)). It is prohibited to directly clear only a specific bit in this register.
- To set a specified bit of this register, set the bit by writing "1" to the applicable bit of the A/D activation trigger control status set register (ADTCSS[n] (n=0 to 31)). It is prohibited to directly set only a specific bit in this register.
- Data can be written directly to this register only when writing to all bits.

#### b) When Accessing to the Range Comparison Register (ADRCIF)

- This register supports writing from the bit-band alias area. For the bit-band alias area, see "CHAPTER: Bit-band Unit".
- To clear a specified bit of this register, clear the bit by writing "1" to the applicable bit of the range comparison flag clear register (ADRCIFC). It is prohibited to directly clear only a specific bit in this register.
- Data can be written directly to this register only when writing to all bits.

#### c) When Accessing to the Scan Conversion Control Status Register (ADSCANS0)

- This register supports writing from the bit-band alias area. For the bit-band alias area, see "CHAPTER: Bit-band Unit".
- To clear a specified bit of this register, clear the bit by writing "1" to the applicable bit of the scan conversion control status clear register 0 (ADSCANS0). It is prohibited to directly clear only a specific bit in this register.
- To set a specified bit of this register, set the bit by writing "1" to the applicable bit of the scan conversion control status set register 0 (ADSCANS0). It is prohibited to directly set only a specific bit in this register.
- Data can be written directly to this register only when writing to all bits.

### (2) Notes on Using A/D Activation Compare

#### a) Configuration of Select

Be sure to configure select while the free-run timer is inactive.

#### b) Configuration of Protection of A/D Data Register

It is necessary to set the PRT bit and the PRTS bit before the A/D conversion starts. It is not allowed to set this bit during the A/D conversion and the state that the A/D data register is being protected.

When cancelling the protection function of the A/D data register, execute the protection cancellation which is set in the PRTS bit after the A/D conversion stops, or disable the protection function by the PRT bit.

If the PRTS bit is changed while the A/D data register is being protected, it is necessary to execute the protection cancellation operation which is set in the PRTS bit (for example reading of the A/D data register, or clearing operation by writing "0" to an interrupt request flag bit) after changing the PRTS bit in order to cancel the A/D data register protection. For example, PRTS is set to 0 after clearing an interrupt request flag bit while PRT=1 and PRTS=1, and the A/D data register is being protected, it is needed to read the A/D data register and clear by writing "0" to the interrupt request flag bit again in order to cancel the protection.

### c) Compare Match Activation

If the compare register (ADCOMP) is set to 0x0000 or the same value as the value set in the compare clear register of the free-run timer, an A/D activation request signal will be generated when a compare match is detected, regardless of whether the setting of the counting direction select bits (SEL0 and SEL1) of the A/D activation trigger control register (ADTCS) specifies counting-up or counting-down.

### d) Setting of the Counting Direction Select Bits of the A/D Activation Trigger Control Status Register

When the free-run timer is in the up count mode, the counting direction select bits SEL0 and SEL1 of the A/D activation trigger control status register (ADTCS) must not be set respectively to "1" and "0" (SEL1, SEL0)=(1,0) (only counting down).

### e) Range Compare

The execution timing of the range comparison is after the A/D conversion result is set in A/D data register (ADTCD) when the A/D conversion is completed. Therefore, the range comparison result is reflected (RCOCD2 to RCOCD0 bit, RCOOF bit, and RCINT bit) delaying 2 cycles in the peripheral clock from the generation of the A/D conversion completion interrupt request. Also, the range comparison interrupt request is generated.

When confirming that the comparison target is outside the range comparison range, the continuous detection measurement is not cleared to 0 times and continue the continuous detection even if the state of the range comparison result changes from larger than the upper bound threshold into smaller than lower bound threshold.

To initialize the continuous detection count state of the range comparison result, switch to enable range comparison execution (ADRCCS.RCOE="0" → "1") after range comparison execution disable has been set, while the A/D has not yet been requested.

### f) About the Continuous Scan Conversion that Doesn't Use the Conversion Count Specification of Each Channel

When two or more repeat modes (ADTCS.RPT="1") are set between the activation channels corresponding to the same A/D converter unit, only a certain channel (channel with the highest priority level) will be processed in the A/D activation arbitration of latter part. Therefore, please make the protection function effective about those activation channels when the repeat mode is set with two or more activation channels. When two or more repeat modes (ADTCS.RPT="1") are set between the activation channels corresponding to the same A/D converter unit, the conversion of the activation channel with low priority is not executed if the data protection state of the activation channel with high priority is released before converting the activation channel with low priority. Therefore, the activation channel of low priority is not executed and the scan conversion is not executed normally when the data protection state is released before the final activation channel of the continuous scanning conversion is executed.

### g) About the Continuous Scan Conversion When the Conversion Count of Each Channel is Specified

Please switch the activation channel whose conversion count specification has been set to scan conversion execution enable, to repeat conversion mode.

When the continuous scan conversion mode with the conversion count of each channel specified, the EOCF bit set to the final channel of the scan conversion is cleared to "0" immediately even if set to "1" because it is restarted from the top immediately after the scan conversion is completed.

When an activation factor that is the same as the scan conversion target is set to an activation channel with a number larger than that of the final channel of continuous scan conversion (with the continuous scan conversion mode with the conversion count of each channel specified effective), the activation channel with the number larger than that of the final channel is evaluated once by the A/D activation arbitration of the latter part after completion of the final channel of the continuous scan conversion.

When an activation factor that is the same as the scan conversion target is set to an activation channel with a number larger than that of the final channel of stop scan conversion (with the stop scan conversion mode with the conversion count of each channel specified effective), the activation channel with the number larger than that of the final channel can execute A/D conversion only during the stop period.

The timing to release the data protection state does not have the restriction if the data protection function is effective. However, when the activation channel that does the scan conversion is in the data protection state, the scan conversion stops until the data protection state is released.







## CHAPTER 52: 12-bit A/D Converter Arbitration

This chapter explains 12-bit A/D converter arbitration.

---

1. Overview
2. Configuration
3. Operation



## 1. Overview

When an A/D activation trigger is entered again during A/D conversion, 12-bit A/D activation arbitration starts another sequence of A/D conversion. The converter also supports forced stop due to an A/D conversion cancel input signal.

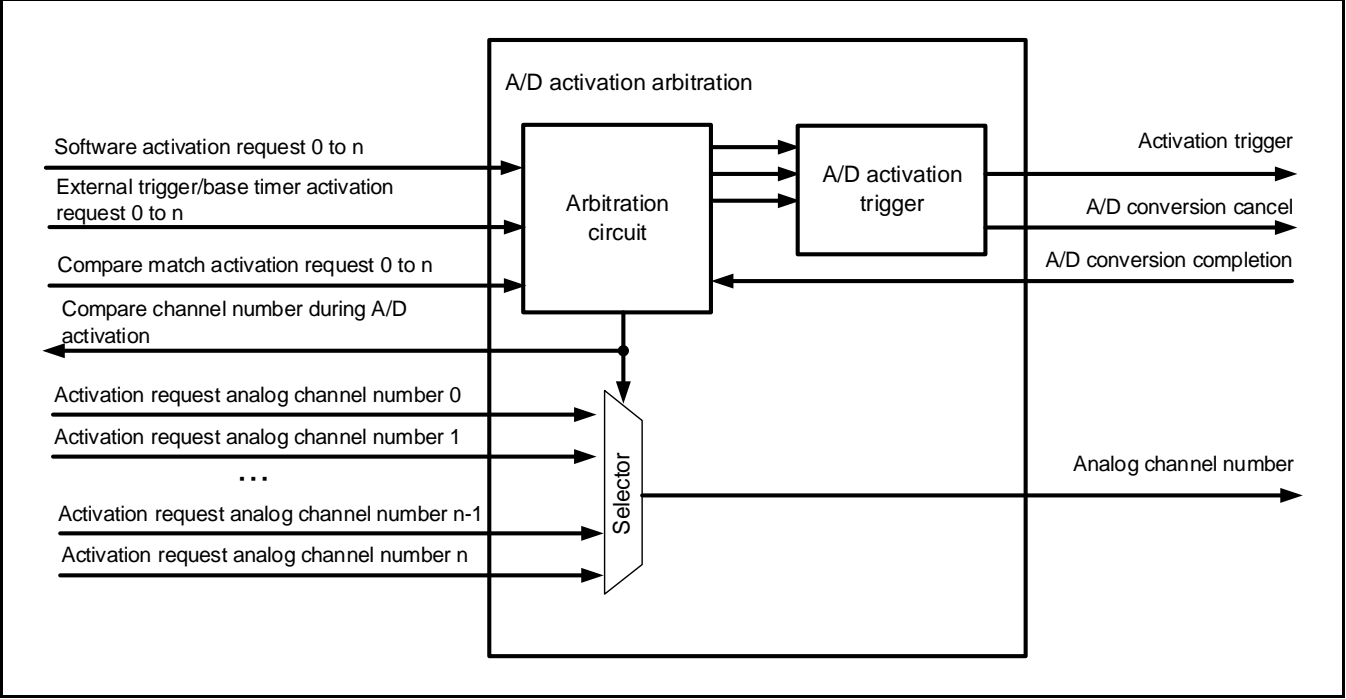
### Function of A/D Activation Arbitration

- The A/D activation arbitration consists of arbitration circuit, an A/D activation trigger generation section, and an analog channel number select section.
- Activation requests from A/D activation compare are arbitrated and the activation triggers, A/D conversion cancel signals, and analog channel numbers are generated.
- An activation trigger is generated for one selected activation request from an A/D activation compare channel. If a contention occurs between activation requests from A/D activation compare channels, A/D activation arbitration uses priority control. The priority order is as follows: "Lower activation channel numbers are assigned higher priorities (priority control based on channel numbers)" or "compare match > external trigger/base timer > software activation" (priority control based on activation factors). Activation requests that are not selected are made to wait. When the current A/D conversion completes, arbitration restarts. Priority control based on activation factors is also performed during A/D conversion. In this case, the current conversion is suspended and the activation factor assigned a higher priority is serviced. The suspended activation factor will be activated again after the higher-priority conversion is processed if the re-arbitration process does not find a lower channel number of a higher-priority activation factor.
  - If an activation factor with the same priority is encountered during A/D conversion suspension: The request from the activation channel with the lower number is processed first.
  - If an activation factor with a different priority is encountered during A/D conversion suspension: The request based on the higher-priority activation factor is processed first.
  - If an activation factor with a higher priority is encountered during A/D conversion: The current conversion is suspended, and the higher-priority activation factor is processed. After this processing completion, arbitration is performed again. The suspended activation factor is then processed.
  - If an activation factor with a lower priority is encountered during A/D conversion: After the current conversion completions, arbitration is performed again. The activation factor with the lower priority is then processed.
  - If an activation factor with the same priority is encountered during A/D conversion: After the current conversion completes, arbitration is performed again. The activation factor with the same priority is then processed.
- A conversion cancel signal is generated to force the current conversion processing to termination when the activation factor that is in process of conversion becomes inactive and there is no other active activation factor.
- For the analog channel number, the activation request analog number entered from the activation channel of the activation request arbitration result is selected.

2. Configuration

This section shows a configuration of A/D activation arbitration.

Figure 2-1 Configuration of A/D Activation Arbitration (n=31)





### 3. Operation

This section describes the operation of A/D activation arbitration.

#### 3.1. A/D Activation Arbitration Operation

The operation of A/D activation arbitration is explained.

A/D activation requests from A/D activation compare are arbitrated to generate an A/D activation trigger. In addition, the analog channel to be A/D-converted is determined.

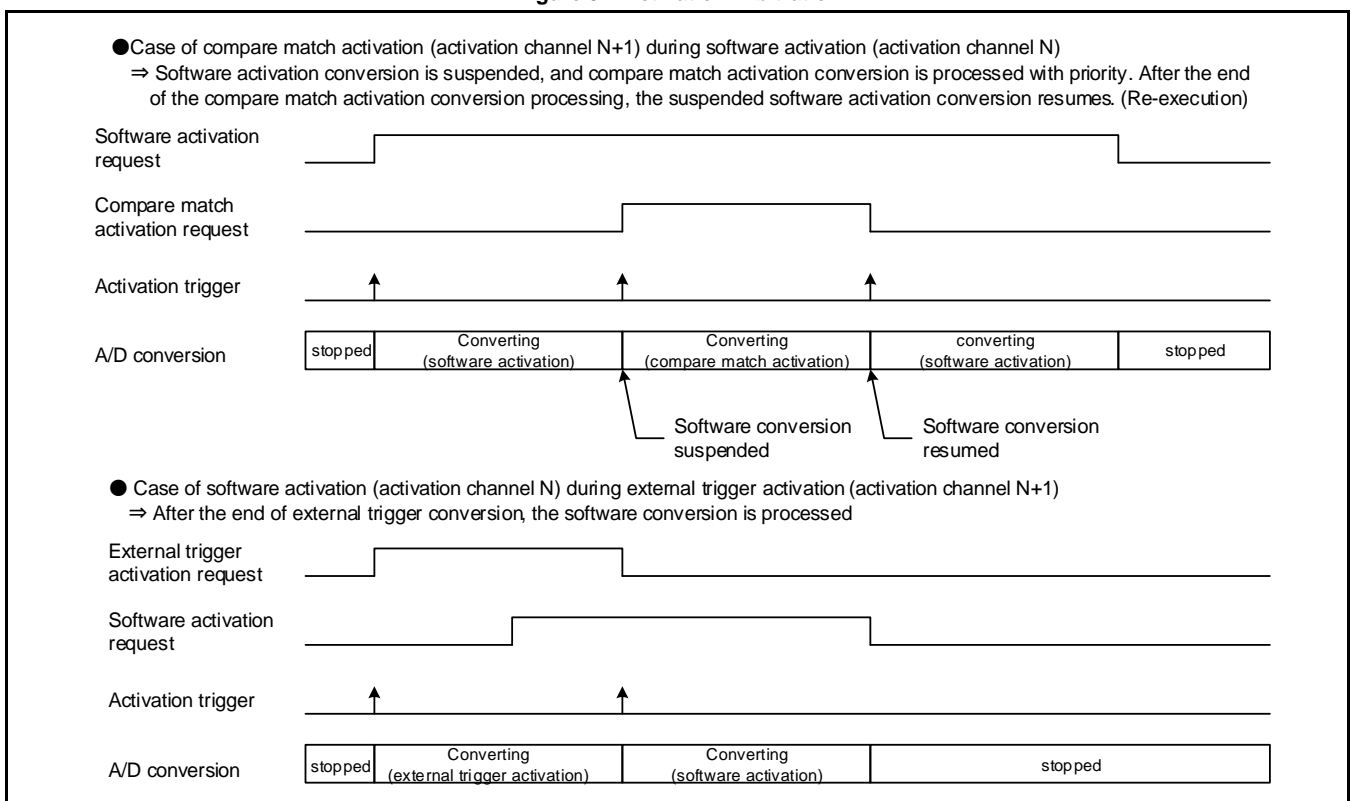
### 3.1.1. A/D Activation Trigger Arbitration

The A/D activation trigger arbitration is explained.

An A/D activation trigger is generated after one of activation requests from A/D activation compare channels is selected. Three types of activation requests from individual A/D activation compare channels are entered: software activation request, external trigger/base timer activation request, and compare match activation request. A/D activation trigger signals are thus generated. If contention occurs among multiple activation requests, the activation request with the lowest compare channel number takes precedence. Activation requests that are not selected are made to wait. When the current A/D conversion completes, arbitration restarts. The sequence of activation arbitration priorities based on the activation factor is as follows: compare match activation request > external trigger/base timer activation request > software activation request. If two activation requests are caused by activation factors assigned the same priority, the request with the lower activation channel number takes precedence.

- If an activation factor with the same priority is encountered during A/D conversion suspension:  
The request from the activation channel with the lower number is processed first.
- If an activation factor with a different priority is encountered during A/D conversion suspension:  
The request based on the higher-priority activation factor is processed first.
- If an activation factor with a higher priority is encountered during A/D conversion:  
The current conversion is suspended, and the higher-priority activation factor is processed. After this processing completes, arbitration is performed again. The suspended activation factor is then processed.
- If an activation factor with a lower priority is encountered during A/D conversion:  
After the current conversion completes, arbitration is performed again. The activation factor with the lower priority is then processed.
- If an activation factor with the same priority is encountered during A/D conversion:  
After the current conversion completes, arbitration is performed again. The activation factor with the same priority is then processed.

Figure 3-1 Activation Arbitration





### 3.1.2. Analog Channel Selection

The analog channel selection is explained.

The A/D activation request and the A/D conversion target analog channel number are entered from the A/D activation compare.

The A/D activation arbitration selects the activation request analog channel number of the selected A/D activation compare channel.

### 3.1.3. A/D Conversion Cancel Function

The A/D conversion cancel function is explained.

When the activation request from the request source becomes inactive during the A/D conversion, an A/D conversion cancel signal is generated to abort the current conversion processing. If the activation factor from another activation channel is active when the activation request from the request source becomes inactive, an A/D conversion cancel signal is not generated, but an A/D activation trigger based on the active activation factor is generated.



## CHAPTER 53: Waveform Generator

This chapter explains waveform generator.

---

1. Overview
2. Configuration
3. Operation
4. Registers
5. Precautions for Using





## 1. Overview

The waveform generator consists of three 16-bit dead timer registers, three timer state control registers, 16-bit dead timer reload interrupt register, 16-bit waveform control register and PPG output control register.

### Function of Waveform Generator

- The waveform generator consists of three 16-bit dead timer registers, three timer state control registers, 16-bit dead timer reload interrupt register, 16-bit dead timer minus setting buffer control register, 16-bit dead timer minus control register, 16-bit waveform control register and PPG output control register.
- You will be able to generate real-time output, 16-bit PPG waveform output, non-overlap 3-phase waveform output (for inverter control), and DC chopper waveform output using the waveform generator.
- You will be able to generate non-overlap waveform output based on the dead time of the 16-bit dead timer (dead time timer function).
- When a real-time output compare match is detected, GATE signal will be generated. This signal starts or stops the PPG timer operation (GATE function)\*.
- When a real-time output compare match is detected, the 16-bit dead timer becomes active. You will be able to start or stop the PPG timer easily by generating GATE signal that controls the PPG operation (GATE function)\*.
- You will be able to select DTTI input to the waveform generator using the DTTI selector.
- By using the DTTI pin, you can control the waveform generator output to forcibly stop it.
- By using the DTTI register, you can control the waveform generator output to forcibly stop it.
- Stop can also be controlled forcibly by using DTTI from the arithmetic motor vector operation accelerator.
- The forcible stop operation of the DTTI pin and the DTTI register can forcibly disable waveform generator output and make the port work as a general-purpose port. This is achieved regardless of the setting of the POF[2:0] bits in the port setting register (PPC\_PCFGRijj).
- The DTTI interrupt requests of waveform generators 00 and 01 can be output to error detection output pins ch.0 (ERDS0) and ch.1 (ERDS1), respectively.

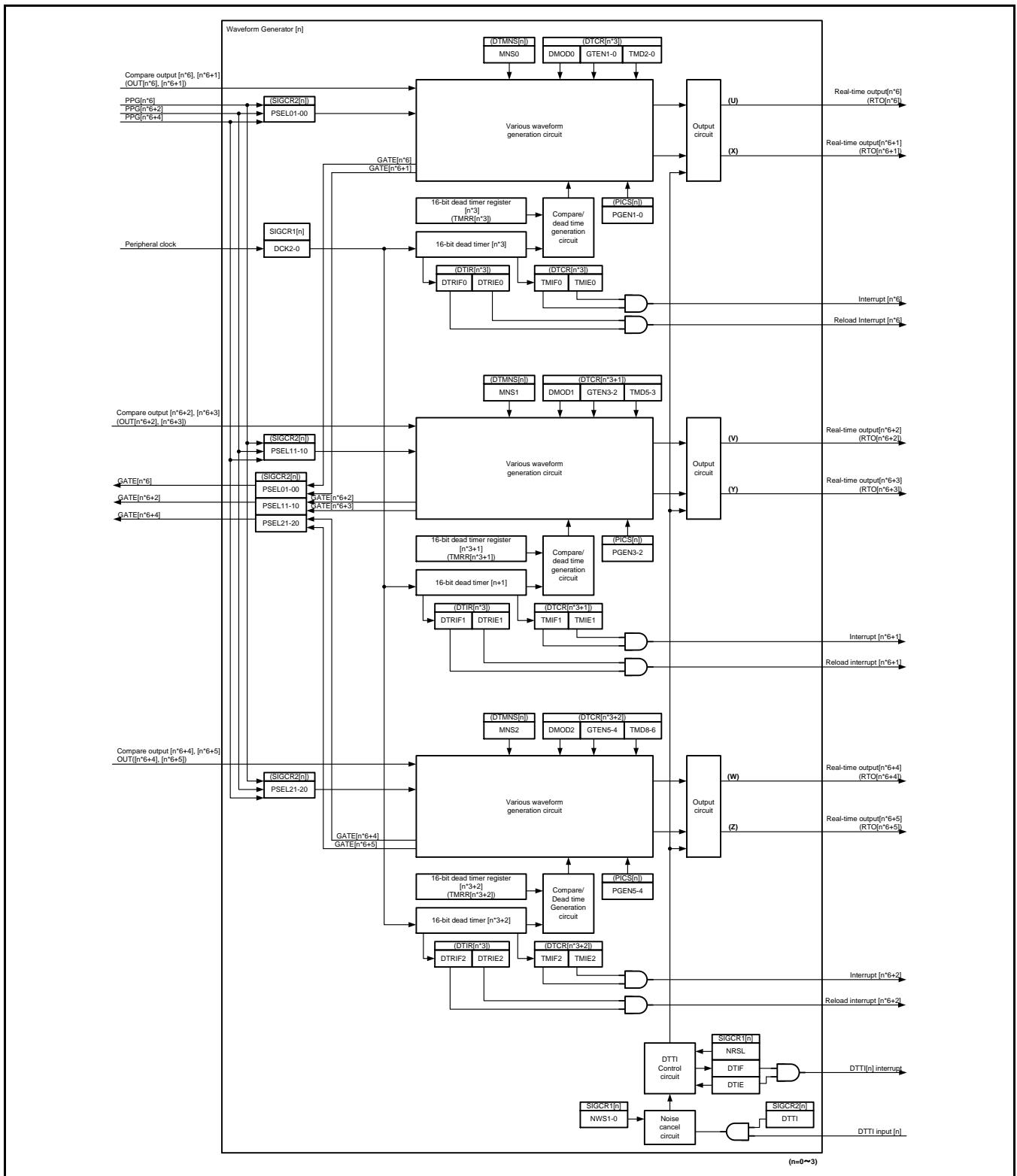
\*: This model does not support PPG timer operation stop.

## 2. Configuration

This section shows a block diagram of waveform generator.

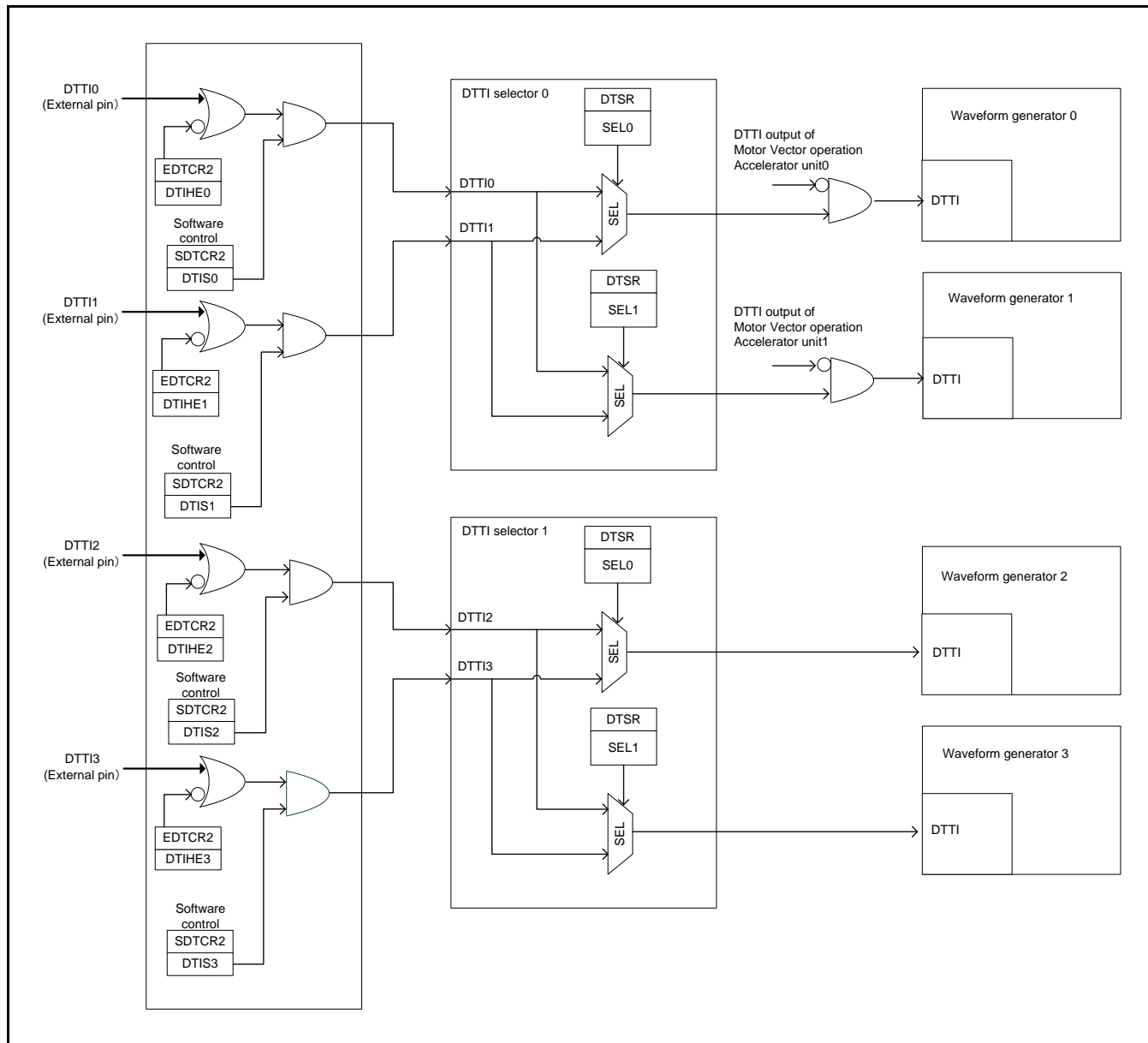
### (1) Block Diagram of Waveform Generator

Figure 2-1 Block Diagram of Waveform Generator



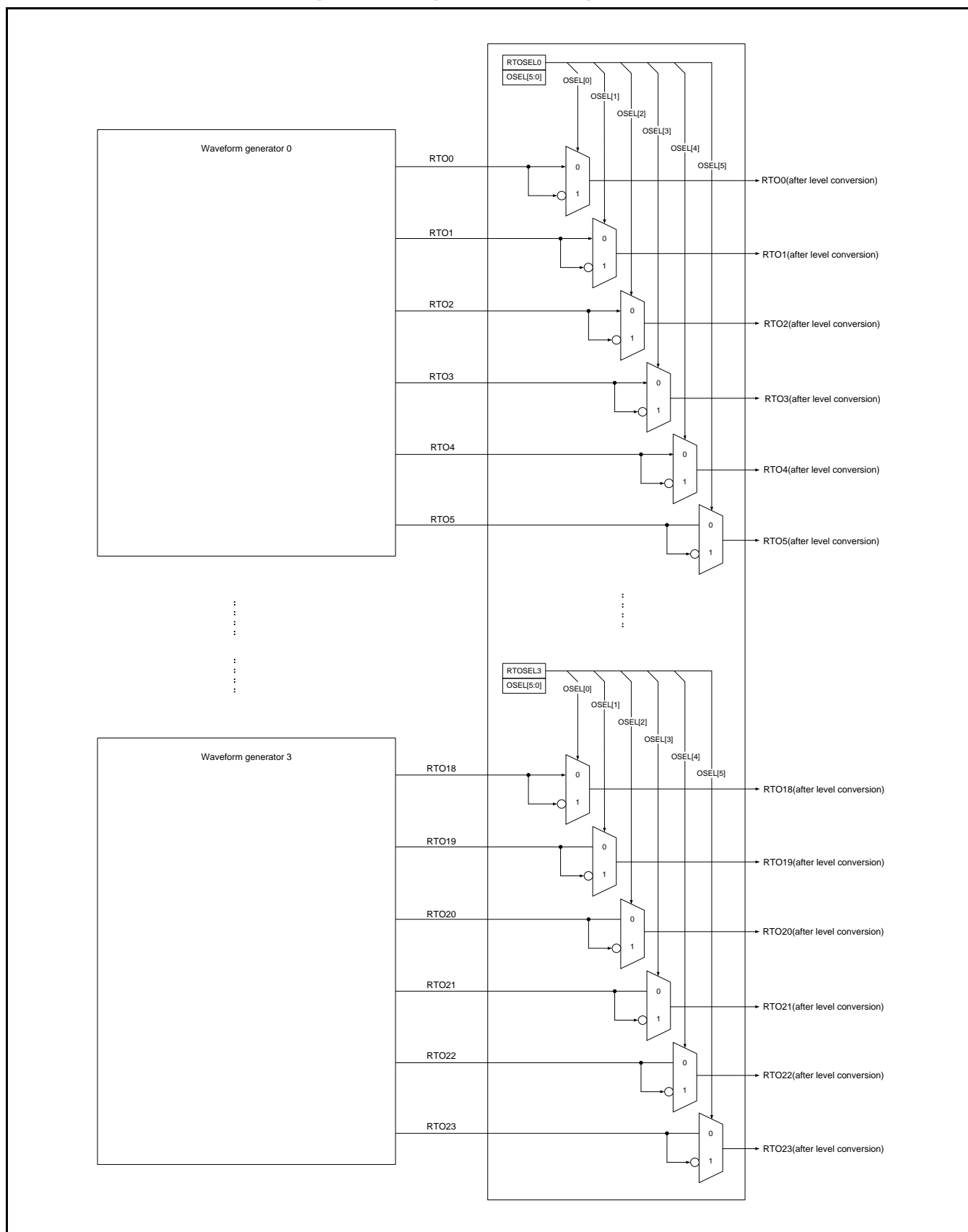
## (2) Configuration of DTTI Input Signal

Figure 2-2 Configuration of DTTI



### (3) Configuration of RTO Output Signal Control

### Figure 2-3 Configuration of RTO Signal Control





### 3. Operation

This section describes the operation of waveform generator.

#### (1) Operation of Waveform Generator

The waveform generator can generate various types of waveforms (including dead time) using the real-time output (RTO0 to RTO5), 16-bit PPG timer 0/2/4, and 16-bit dead timer 0/1/2.

#### (2) Output State of RTO0 to RTO5 and GATE

Table 3-1 RTO/GATE Output State and Bit Settings

TMD2	TMD1	TMD0	GTENn <sup>*4</sup>	PGENn <sup>*4</sup>	RTON <sup>*4</sup>	GATE
0	0	0	X	X	Compare output OUTn <sup>*4</sup> (16-bit output compare output)	Always "0"
0	0	1	0/1	0	Compare output OUTn <sup>*4</sup> (16-bit output compare output)	(OUTn <sup>*4</sup> and GTENn <sup>*4</sup> ) <sup>*2</sup>
			0	1	Output the PPG0/PPG2/PPG4 pulses while the OUTn <sup>*4</sup> is "H" <sup>*1</sup>	Always "0"
			1	1	Output the PPG0/PPG2/PPG4 pulses activated by GATE signals while the OUTn <sup>*4</sup> is "H"	(OUT0/OUT1/ OUT2/OUT3/ OUT4/OUT5)
0	1	0	0/1	0	Activate the 16-bit dead timer 0 by the rising edge of OUT0 and OUT1, and "H" is being output until the 16-bit dead timer 0 underflows.	When GTENn <sup>*4</sup> and the timer is active, "H" is being output <sup>*3</sup>
			0/1	0	Activate the 16-bit dead timer 1 by the rising edge of OUT2 and OUT3, and "H" is being output until the 16-bit dead timer 1 underflows.	
			0/1	0	Activate the 16-bit dead timer 2 by the rising edge of OUT4 and OUT5, and "H" is being output until the 16-bit dead timer 2 underflows.	
			0	1	Activate the 16-bit dead timer 0 by the rising edge of OUT0 and OUT1, and PPG0/PPG2/PPG4 pulses are being output until the 16-bit dead timer 0 underflows. <sup>*1</sup>	Always "0"
			0	1	Activate the 16-bit dead timer 1 by the rising edge of OUT2 and OUT3, and PPG0/PPG2/PPG4 pulses are being output until the 16-bit dead timer 1 underflows. <sup>*1</sup>	
			0	1	Activate the 16-bit dead timer 2 by the rising edge of OUT4 and OUT5, and PPG0/PPG2/PPG4 pulses are being output until the 16-bit dead timer 2 underflows. <sup>*1</sup>	
			1	1	Activate the 16-bit dead timer 0 by the rising edge of OUT0 and OUT1, and PPG0/PPG2/PPG4 pulses activated by GATE signals are being output until the 16-bit dead timer 0 underflows.	When the timer is active, "H" is being output <sup>*3</sup>
			1	1	Activate the 16-bit dead timer 1 by the rising edge of OUT2 and OUT3, and PPG0/PPG2/PPG4 pulses activated by GATE signals are being output until the 16-bit dead timer 1 underflows.	
			1	1	Activate the 16-bit dead timer 2 by the rising edge of OUT4 and OUT5, and PPG0/PPG2/PPG4 pulses activated by GATE signals are being output until the 16-bit dead timer 2 underflows.	
1	0	0	X	X	Generate a non-overlap signal at OUT1	Always "0"
			X	X	Generate a non-overlap signal at OUT3	
			X	X	Generate a non-overlap signal at OUT5	

TMD2	TMD1	TMD0	GTENn*4	PGENn*4	RTO n*4	GATE
1	1	1	0	X	Setting prohibited	-
			1	X	Setting prohibited	-
Others					Always "0"	Always "0"

\*1: You will need to select a channel you wish to use from the PPG0/PPG2/PPG4 and activate the PPG in advance.

\*2: GTENn\*<sup>4</sup> signal will be generated from the OUTn\*<sup>4</sup> of which GTEN bit is set to "1".

\*3: GATE signal will be generated during the operation of the timer activated by the OUTn\*<sup>4</sup> of which GTENn\*<sup>4</sup> bit is set to "1". If more than one GATEN\*<sup>4</sup> bit is set to "1", GATE signal will be the OR signal of signals in each active timer operation.

\*4: n=0 to 5

**Note:**

- The RTO0 and RTO1 are controlled by the TMD2 to TMD0 of the 16-bit dead timer state control register (DTCR0). The RTO2 and RTO3 are controlled by the TMD5 to TMD3 of the DTCR1 register. The RTO4 and RTO5 are controlled by the TMD8 to TMD6 of the DTCR2 register.

### (3) PPG Output Control

PPG output to the RTO0 to RTO5 pins can be enabled by the PGEN5 to PGEN0 of the PPG output control register (PICS).

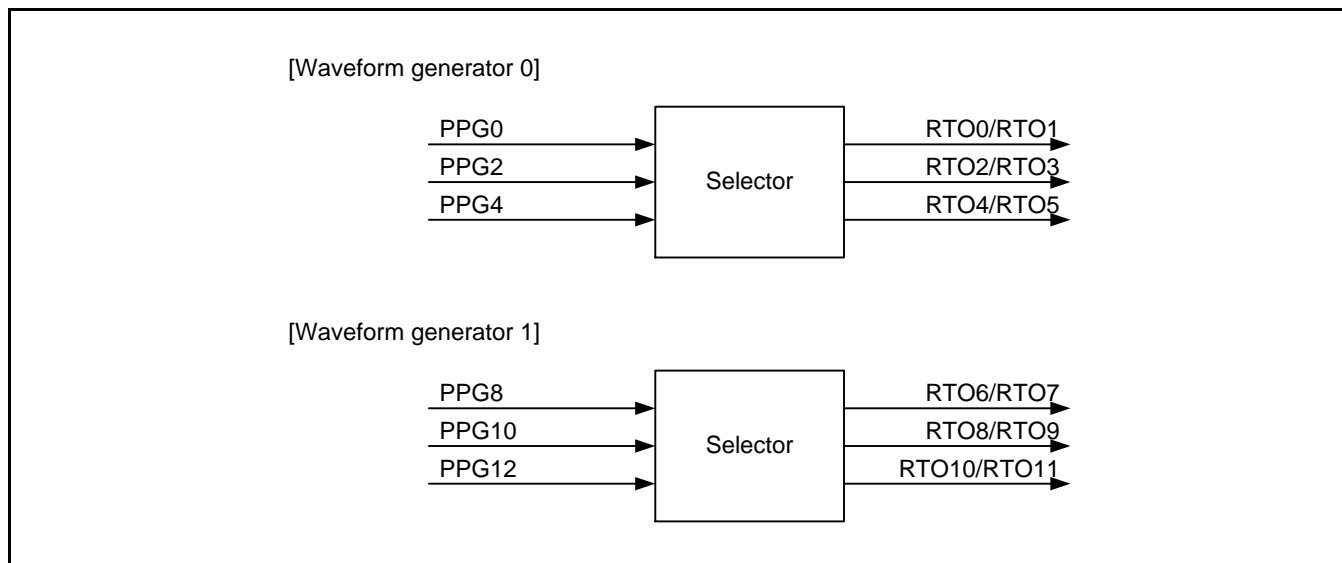
### (4) Gate-triggered PPG Output

The waveform generator can generate GATE signals by the real-time output RTO0 to RTO5. Each of the 16-bit dead timer 0, 1, and 2 operates two real-time outputs (RTO0/RTO2/RTO4, RTO1/RTO3/RTO5), which generates six distinct gate signals. These six gate signals generates OR GATE signal, which triggers the PPG count. In addition, when you use the PGEN0 to PGEN5 signals, you will be able to output six types of waveforms to the RTO0 to RTO5 pins by using the PPG only.

**Note:**

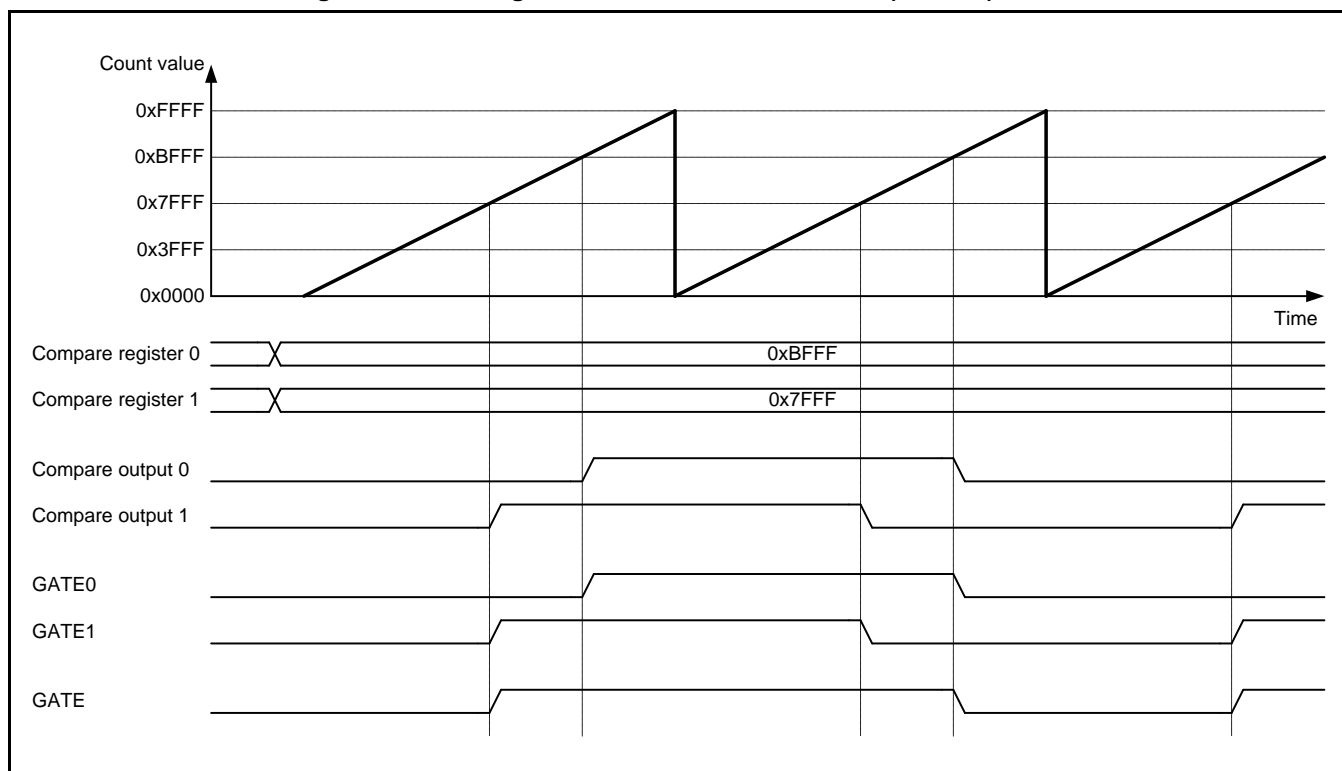
- The case of waveform generator 0 is introduced as an example. For waveform generator 1, PPG channels you will be able to select are shown in the table below.

Figure 3-1 Combination of Waveform Generator and PPG



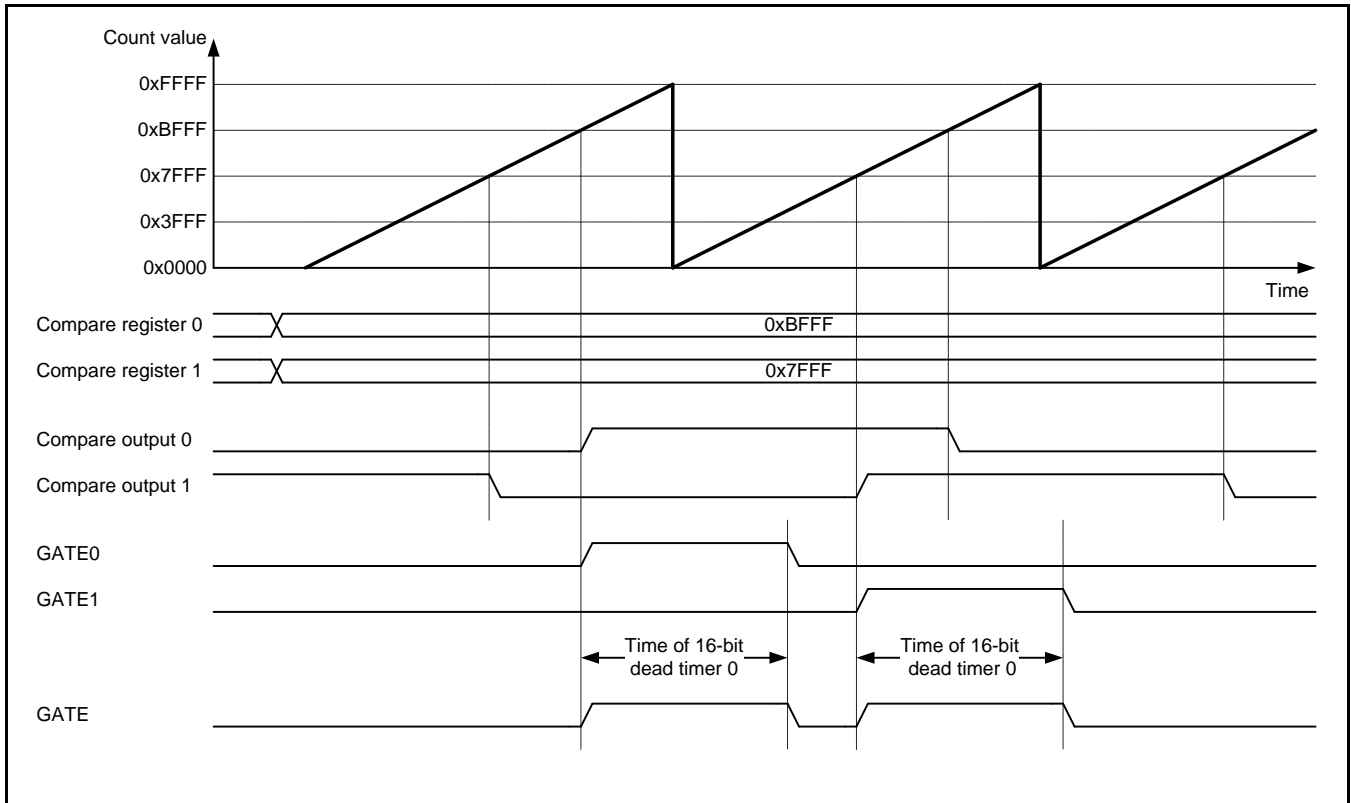
a) GATE Signal Generation When the GATEn is Active and Each OUTn is "H" (TMD8 to TMD0 of the 16-bit Dead Timer State Control (DTSCR0, DTSCR1, and DTSCR2) are "0b001")

Figure 3-2 GATE Signal Generation When the OUTn (n=0 to 5) is "H"



**b) GATE Signal Generation from the Rising Edge of the OUTn Through the Underflow of the 16-bit Dead Timer 0, 1, and 2 When the GTENn is Active (TMD8 to TMD0 of the DTSCR0, DTSCR1, and DTSCR2 Registers are "0b010")**

**Figure 3-3 GATE Signal Generation from the Rising Edge of the OUTn through the 16-bit Dead Timer Underflow**



**Note:**

- Each 16-bit dead timer will be used for two OUTs. In other words, the 16-bit dead timer 0 is used for OUT0 and OUT1, the 16-bit dead timer 1 is used for OUT2 and OUT3, and the 16-bit dead timer 2 is used for OUT4 and OUT5. Thus, do not activate the timer which is already active, using the OUT. If you activate such timer, the GATE signal output will be extended, which may cause a malfunction as a result.

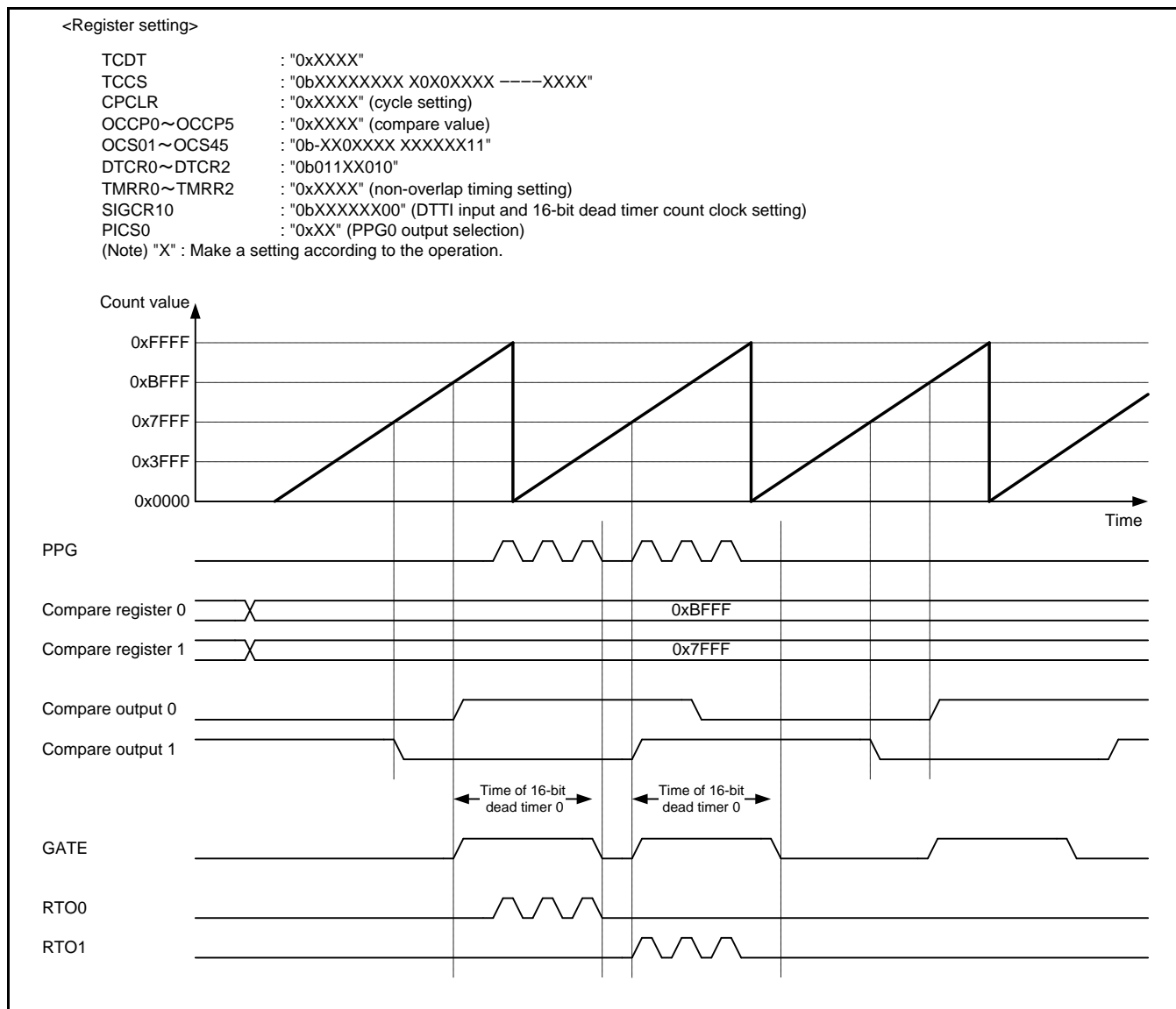
**(5) Operation in Timer Mode**

If a rising edge of the OUT0 to OUT5 pins has been detected, the value will be reloaded to the 16-bit dead timer and the timer starts counting down. The PPG timer continues to output to the RTO0 to RTO5 pins until the 16-bit dead timer underflows.

**a) PPG Output Pulse Generation from the OUT Rising Edge Through the 16-bit Dead Timer Underflow (TMD8 to TMD0 of the DTSCR0, DTSCR1, and DTSCR2 Registers are "0b010")**



Figure 3-4 Waveform Generated When the TMD2 to TMD0 are "0b010"



**Note:**

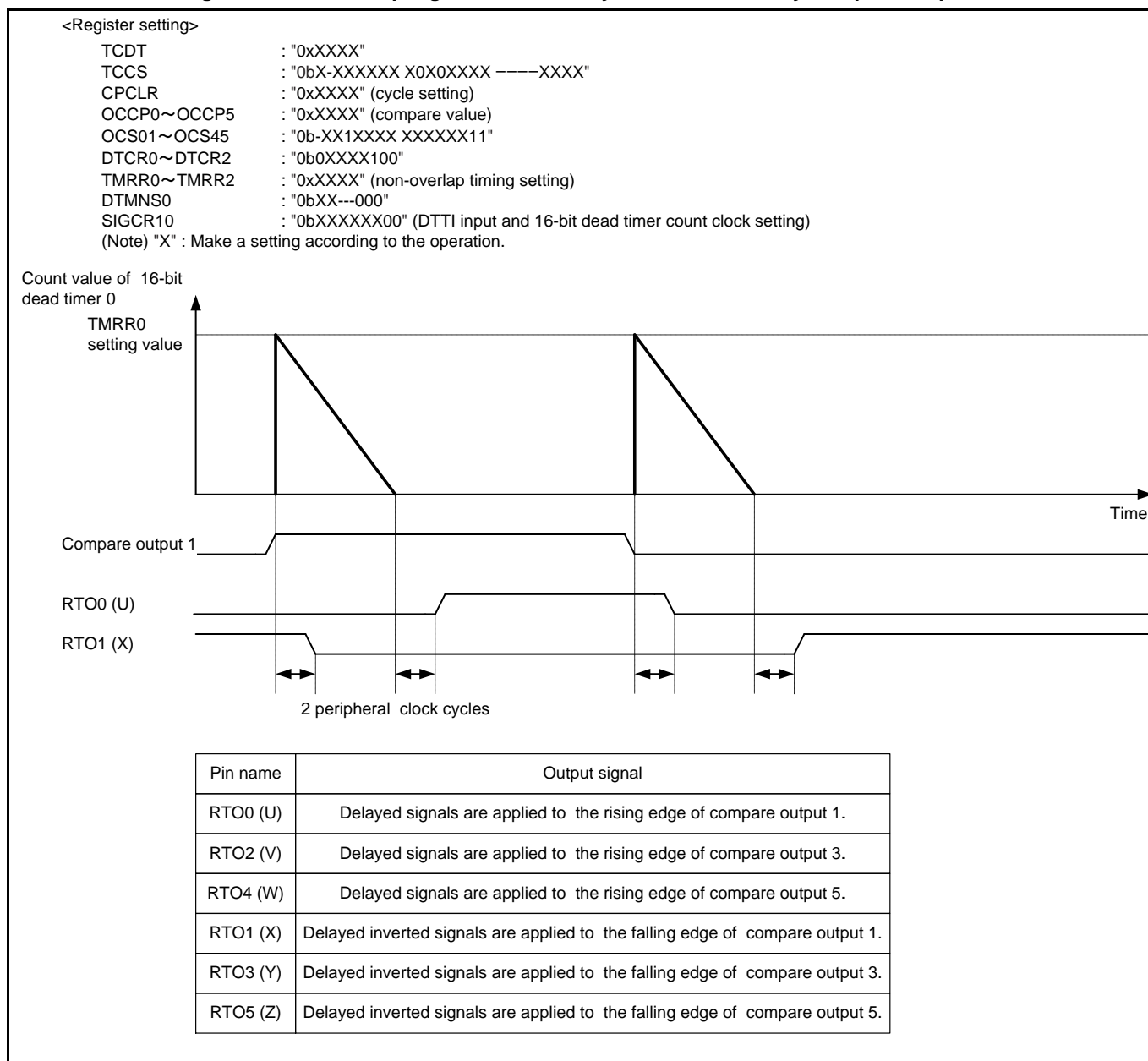
- Each 16-bit dead timer will be used for two OUTs. In other words, the 16-bit dead timer 0 is used for OUT0 and OUT1, the 16-bit dead timer 1 is used for OUT2 and OUT3, and the 16-bit dead timer 2 is used for OUT4 and OUT5. Thus, do not activate the timer which is already active, using the OUT. If you activate such timer, the GATE signal output will be extended, which may cause a malfunction as a result.

#### **(6) Operation of the Dead Time Timer Mode**

The dead time generator inputs the compare output (OUT1, OUT3, OUT5) and outputs non-overlap signal (inverted signal) to the external pins (RTO0 to RTO5).

##### **a) Non-overlap Signal Generation by the Normal Polarity OUT1, OUT3, and OUT5 (TMD8 to TMD0 of the 16-bit Dead Timer Control Registers (DTCR0, DTCR1, and DTCR2) are "0b100")**

If you select the non-overlap signal of which DMOD2 to DMOD0 of the DTCR0, DTCR1, and DTCR2 registers are "0" (normal polarity), the delay that corresponds to the non-overlap time configured at the 16-bit dead timer registers (TMRR0 to TMRR2) will be applied. This delay will be applied to the rising edge or the falling edge of the OUT1, OUT3, and OUT5 pins. If the edge transition time of the OUT1, OUT3, and OUT5 is less than the non-overlap time configured, the 16-bit dead timer restarts counting down from the value of time from the dead timer start at the next RT edge to the restart. When the dead timer is activated once again before the counting down of the restarted dead timer ends, the counting down will be restarted with the values of registers TMRR0 to TMRR2.

**Figure 3-5 Non-overlap Signal Generation by the Normal Polarity Compare Output**


**b) Non-overlap Signal Generation of Minus Control by the Inverted Polarity OUT1, OUT3, and OUT5 (TMD8 to TMD0 of the 16-bit Dead Timer Control Registers (DTCR0, DTCR1, and DTCR2) are "0b100")**

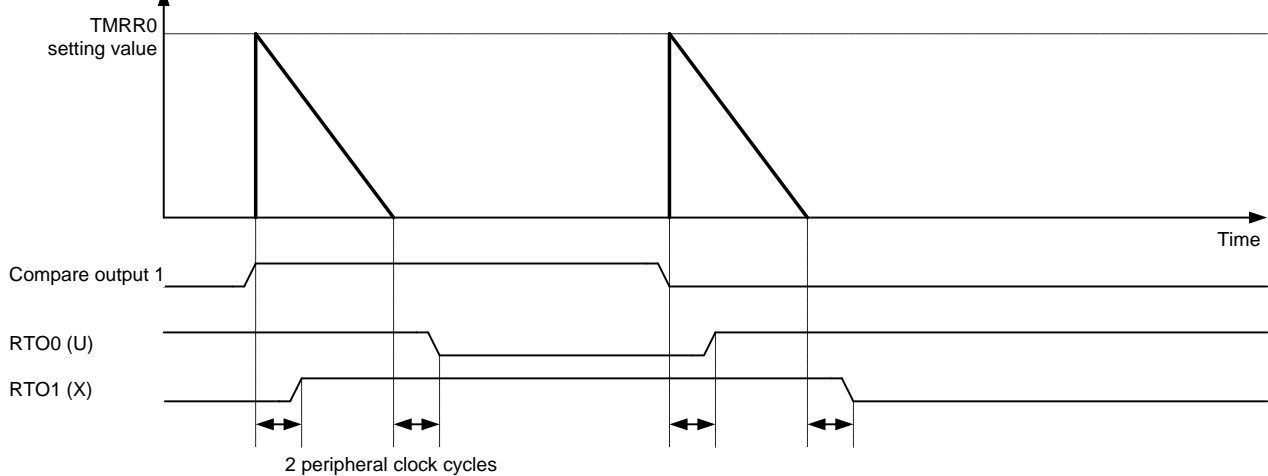
If you select the non-overlap signal of which DMOD2 to DMOD0 of the DTCR0, DTCR1, and DTCR2 registers are "1" (inverted polarity), the delay that corresponds to the non-overlap time configured at the 16-bit dead timer registers (TMRR0 to TMRR2) will be applied. This delay will be applied to the rising edge or the falling edge of the OUT1, OUT3, and OUT5 pins. If the edge transition time of the OUT1, OUT3, and OUT5 is less than the non-overlap time configured, the 16-bit dead timer restarts counting down from the value of time from the dead timer start at the next RT edge to the restart. When the dead timer is activated once again before the counting down of the restarted dead timer ends, the counting down will be restarted with the values of registers TMRR0 to TMRR2.

Figure 3-6 Non-overlap Signal Generation by the Inverted Polarity Compare Output

<Register setting>

TCDT : "0XXXXX"  
TCCS : "0bX-XXXXXX X0X0XXXX ----XXXX"  
CPCLR : "0XXXXX" (cycle setting)  
OCCP0~OCCP5 : "0XXXXX" (compare value)  
OCS01~OCS45 : "0b-XX1XXXX XXXXXX11"  
DTCR0~DTCR2 : "0b1XXXX100"  
TMRR0~TMRR2 : "0XXXXX" (non-overlap timing setting)  
DTMNS0 : "0bXX---000"  
SIGCR10 : "0bXXXXXX00" (DTTI input and 16-bit dead timer count clock setting)  
(Note) "X" : Make a setting according to the operation.

Count value of 16-bit  
dead timer



Pin name	Output signal
RTO0 (U)	Delayed inverted signals are applied to the rising edge of compare output 1.
RTO2 (V)	Delayed inverted signals are applied to the rising edge of compare output 3.
RTO4 (W)	Delayed inverted signals are applied to the rising edge of compare output 5.
RTO1 (X)	Delayed signals are applied to the falling edge of compare output 1.
RTO3 (Y)	Delayed signals are applied to the falling edge of compare output 3.
RTO5 (Z)	Delayed signals are applied to the falling edge of compare output 5.

### (7) Operation of the Dead Time Timer Mode (Minus Control)

You will be able to execute the minus control function of the non-overlap time (MNSx bit of the DTMNS register is 1) using the 16-bit dead timer minus control register (DTMNS).

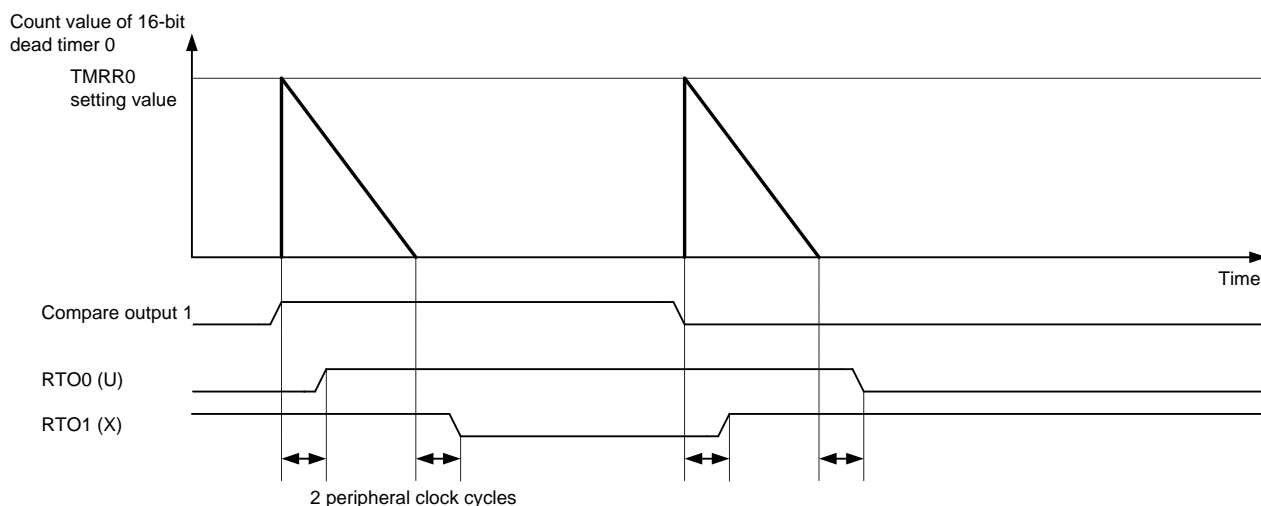
#### a) Non-overlap Signal Generation of Minus Control by the Normal Polarity OUT1, OUT3, and OUT5 (TMD8 to TMD0 of the 16-bit Dead Timer Control Registers (DTCR0, DTCR1, and DTCR2) are "0b100")

Signal generation is executed by outputting U/V/W and X/Y/Z of the inverted polarity non-overlap signal that does not operate by minus control as X/Y/Z and U/V/W.

**Figure 3-7 Non-overlap Signal Generation When MNSn Bit of the Normal Polarity DTMNS Register is 1 (Minus Setting)**

<Register setting>

TCDT : "0XXXXX"  
TCCS : "0bX-XXXXXX X0X0XXXX ----XXXX"  
CPCLR : "0XXXXX" (cycle setting)  
OCCP0~OCCP5 : "0XXXXX" (compare value)  
OCS01~OCS45 : "0b-XX1XXXX XXXXXX11"  
DTCR0~DTCR2 : "0b1XXXX100"  
TMRR0~TMRR2 : "0XXXXX" (non-overlap timing setting)  
DTMNS0 : "0bXX---111"  
SIGCR10 : "0bXXXXXX00" (DTTI input and 16-bit dead timer count clock setting)  
(Note) "X" : Make a setting according to the operation.

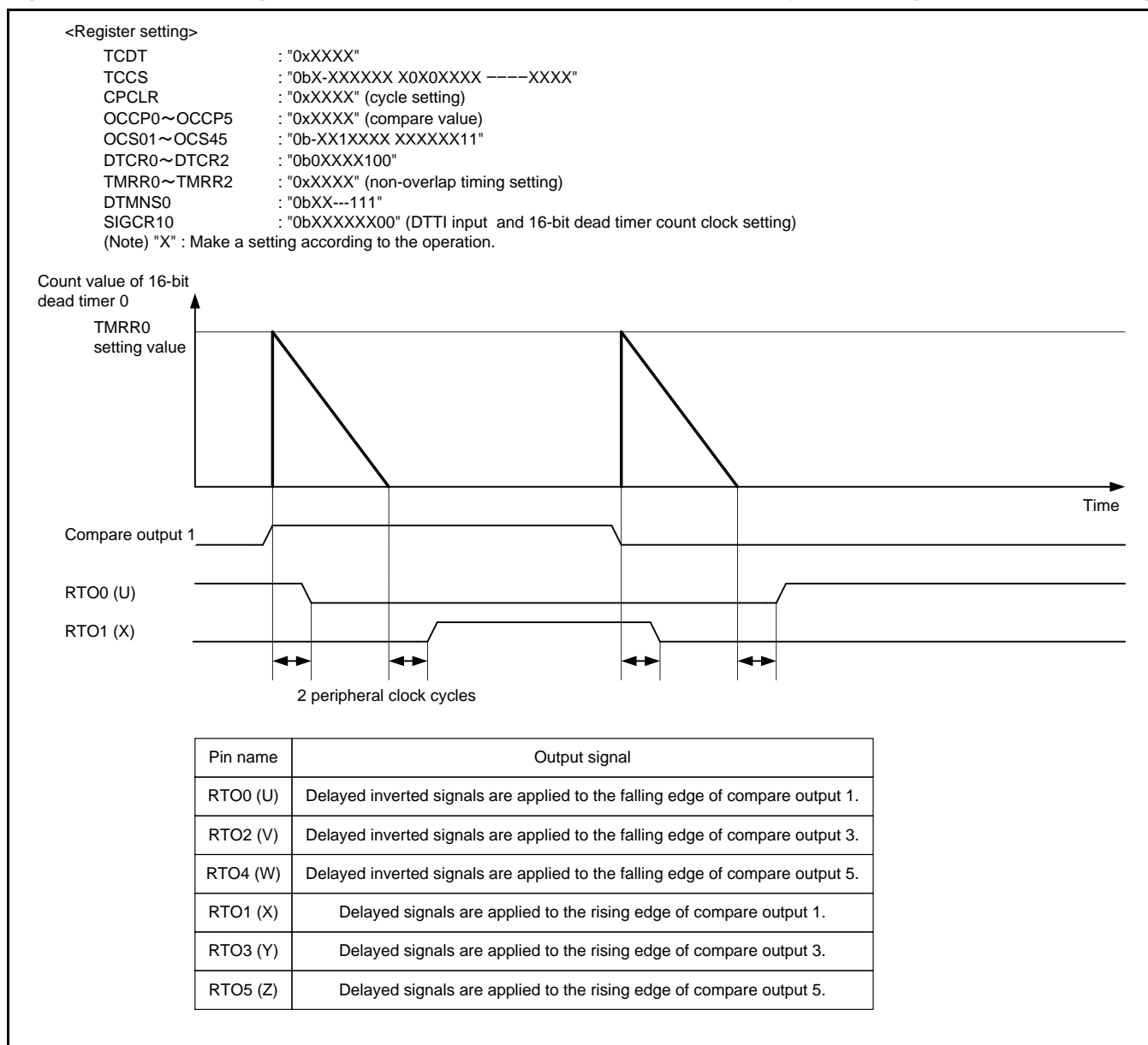


Pin name	Output signal
RTO0 (U)	Delayed signals are applied to the falling edge of compare output 1.
RTO2 (V)	Delayed signals are applied to the falling edge of compare output 3.
RTO4 (W)	Delayed signals are applied to the falling edge of compare output 5.
RTO1 (X)	Delayed inverted signals are applied to the rising edge of compare output 1.
RTO3 (Y)	Delayed inverted signals are applied to the rising edge of compare output 3.
RTO5 (Z)	Delayed inverted signals are applied to the rising edge of compare output 5.

**b) Non-overlap Signal Generation by the Inverted Polarity OUT1, OUT3, and OUT5 Control (TMD8 to TMD0 of the 16-bit Dead Timer Control Registers (DTCR0, DTCR1, and DTCR2) are "0b100")**

Signal generation is executed by outputting U/V/W and X/Y/Z of the normal polarity non-overlap signal that does not operate by minus control as X/Y/Z and U/V/W.

**Figure 3-8 Non-overlap Signal Generation When MNSn Bit of the Inverted Polarity DTMNS Register is 1 (Minus Setting)**



## (8) Operation of the Dead Time Timer Mode (Precautions)

### a) Signal Generation

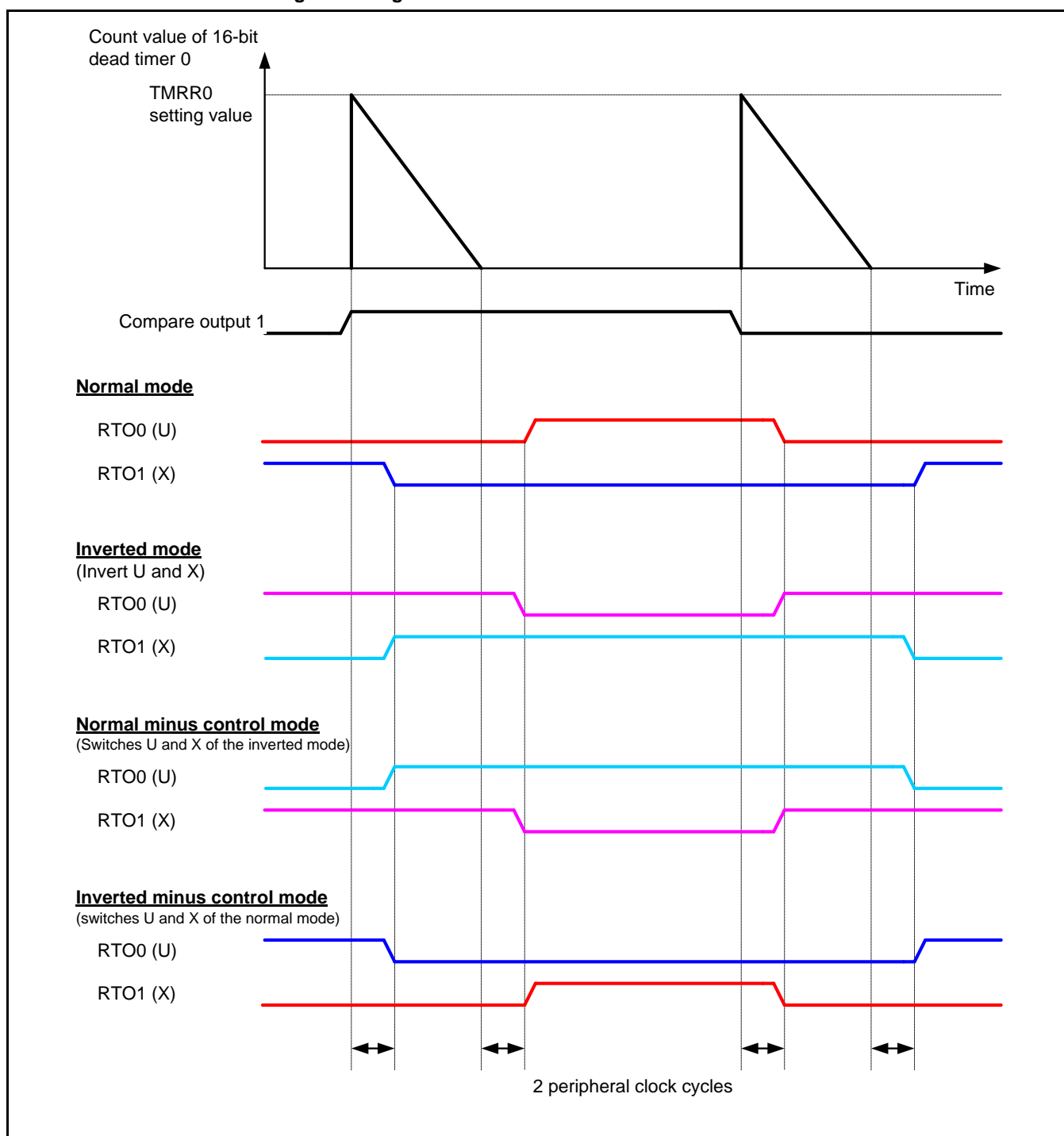
Default is the normal mode.

In the inverted mode, each U and X of the normal mode will be output after being inverted.

In the normal minus control mode, each U and X of the normal mode will be output after being inverted and replaced.

In the inverted minus control mode, each U and X of the normal mode will be output after being replaced.

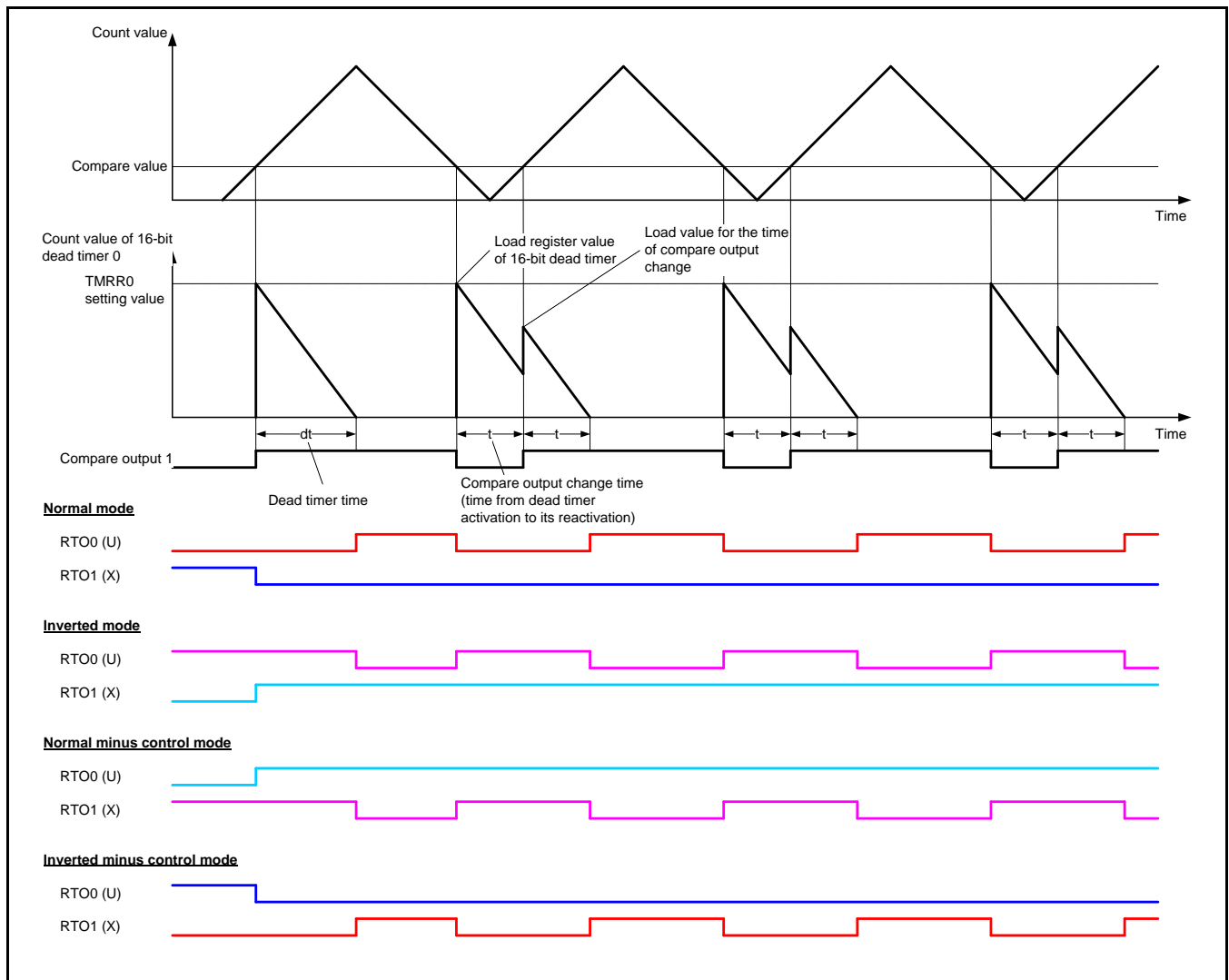
Figure 3-9 Signal Generation in the Dead Time Timer Mode



**b) When the "H" Interval of the Compare Output is Long (or Short) and a Reload Occurs Before the Dead Timer Underflows (the Reload of the Dead Timer is Once)**

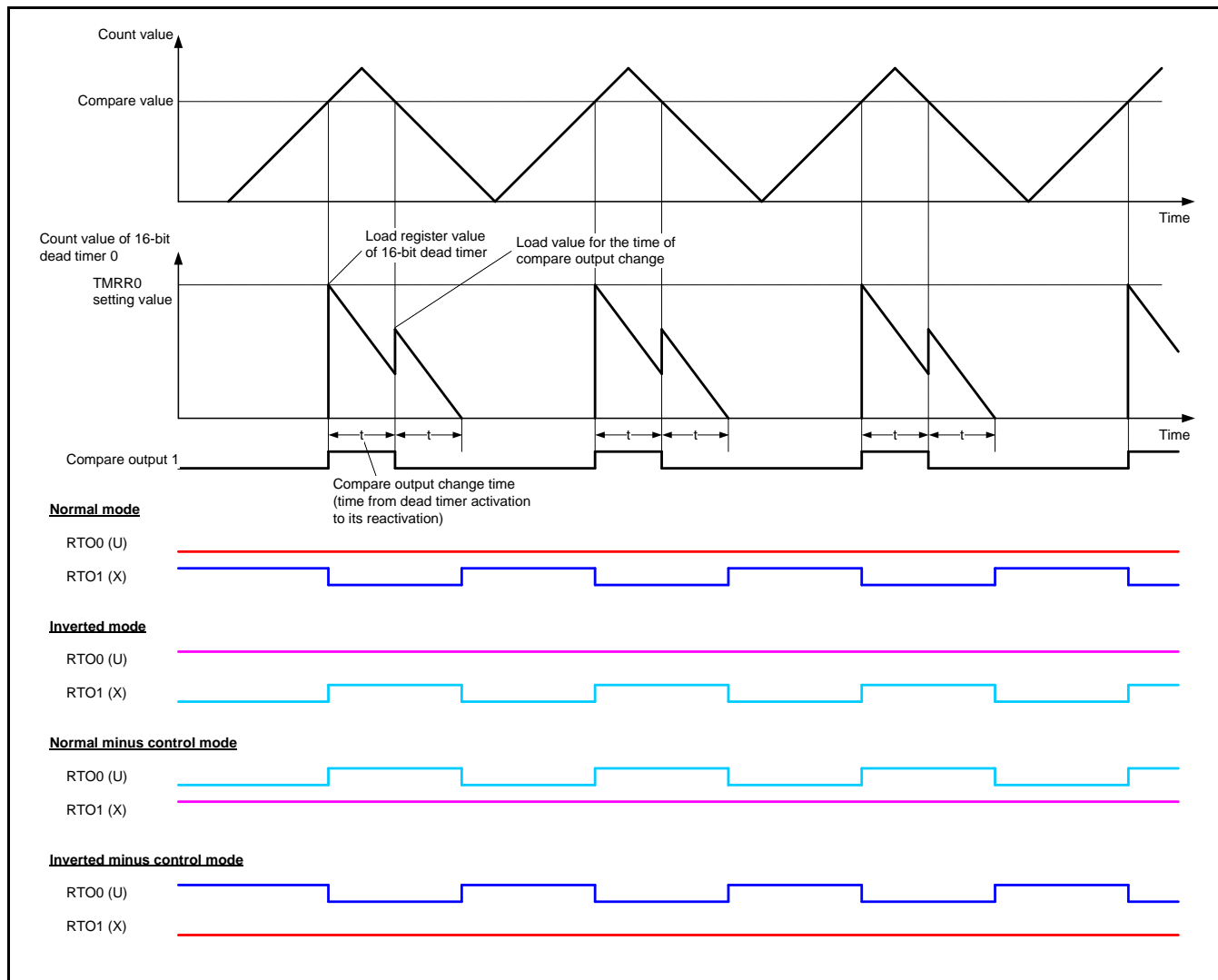
In the normal mode or inverted minus control mode, output of the X (or U) is fixed to "L". In the normal minus control mode or inverted mode, output of the U (or X) is fixed to "H". While the dead timer is counting down, both U and X are output as "H". The interval is 2 times of the time from the start to the restart of the dead timer (compare output transition time). In addition, when a reload occurs before the timer underflows while the dead timer is active, the interrupt request flag bit of the 16-bit dead timer reload interrupt register (DTIR) will be set. If interrupts are enabled, the interrupt will be notified.

**Figure 3-10 When the "H" Interval of the Compare Output is Long and a Reload Occurs Before the Dead Timer Underflows**





**Figure 3-11 When the "H" Interval of the Compare Output is Short and a Reload Occurs Before the Dead Timer Underflows**

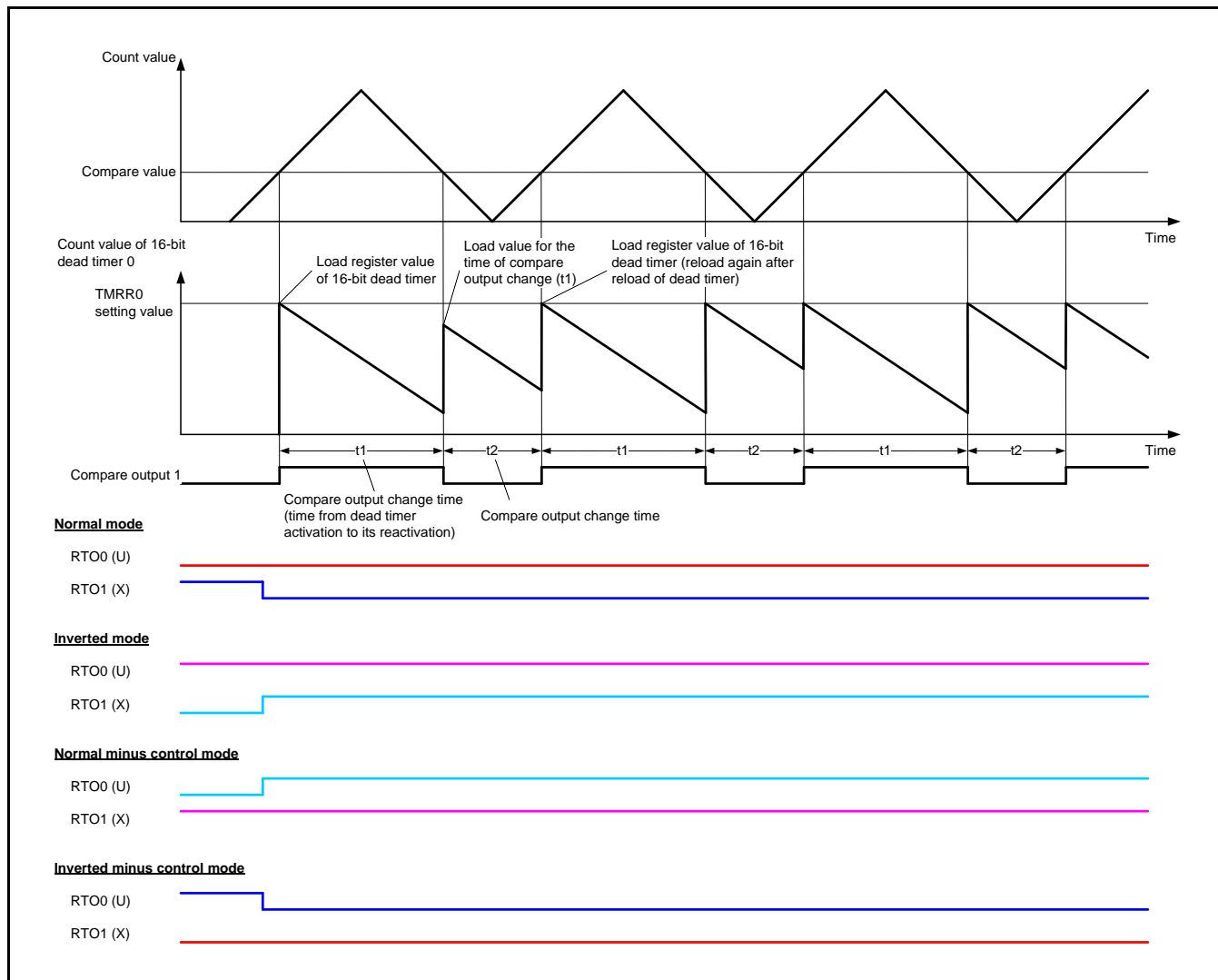


**c) When the Compare Output Transition Time is Short and Reload Continues Before the Dead Timer Underflows (the Reload of the Dead Timer Continues)**

A setting of 16-bit dead timer register (TMRR[n]) with a short compare output transition time that allows continuous reloading before the dead timer underflows is prohibited. In the case with the setting above in normal mode or inverted minus control mode, output of the X and U will be fixed to "L". In the normal minus control mode or inverted mode, output of the U and X will be fixed to "H".

In addition, when a reload occurs before the timer underflows while the dead timer is active, the interrupt request flag bit of the 16-bit dead timer reload interrupt register (DTIR) will be set. If interrupts are enabled, the interrupt will be notified.

**Figure 3-12 When the Compare Output Transition Time is Short and Reload Continues Before the Dead Timer Underflows**

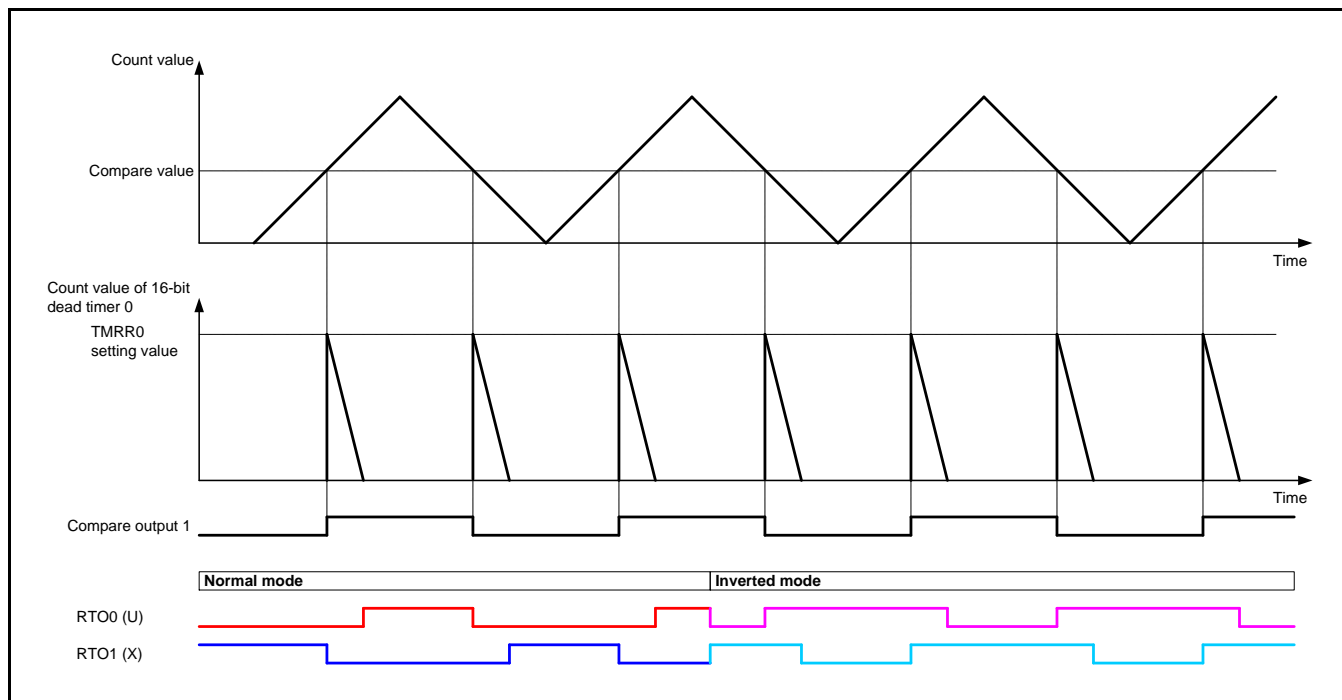


#### **d) Changing From Normal Mode to Inverted Mode while the Dead Time Timer Mode is Active**

If you change the operation mode from the normal mode to the inverted mode while the dead time timer mode is active, the point of variation of U and X overlap. Note that the operation mode will be changed from the normal mode to the inverted mode immediately.



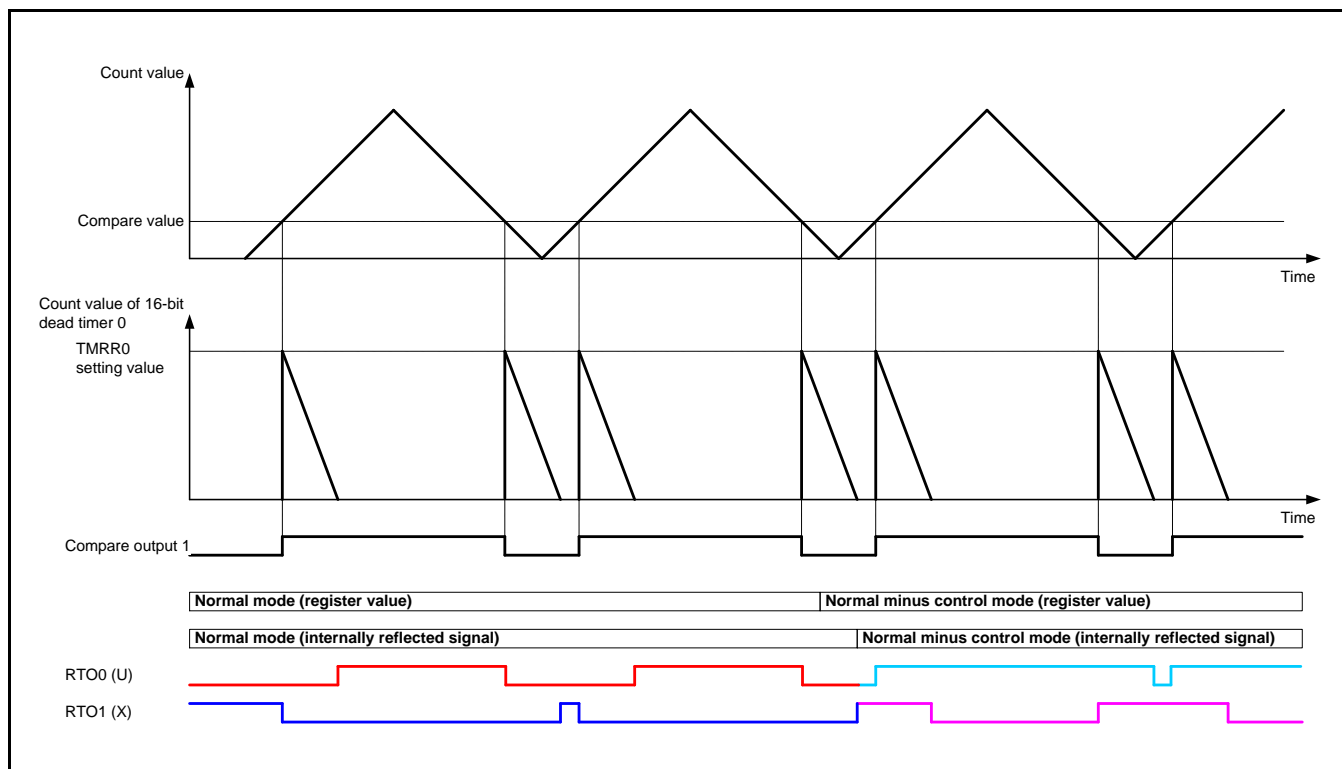
**Figure 3-13 Changing from Normal Mode to Inverted Mode while the Dead Time Timer Mode is Active**



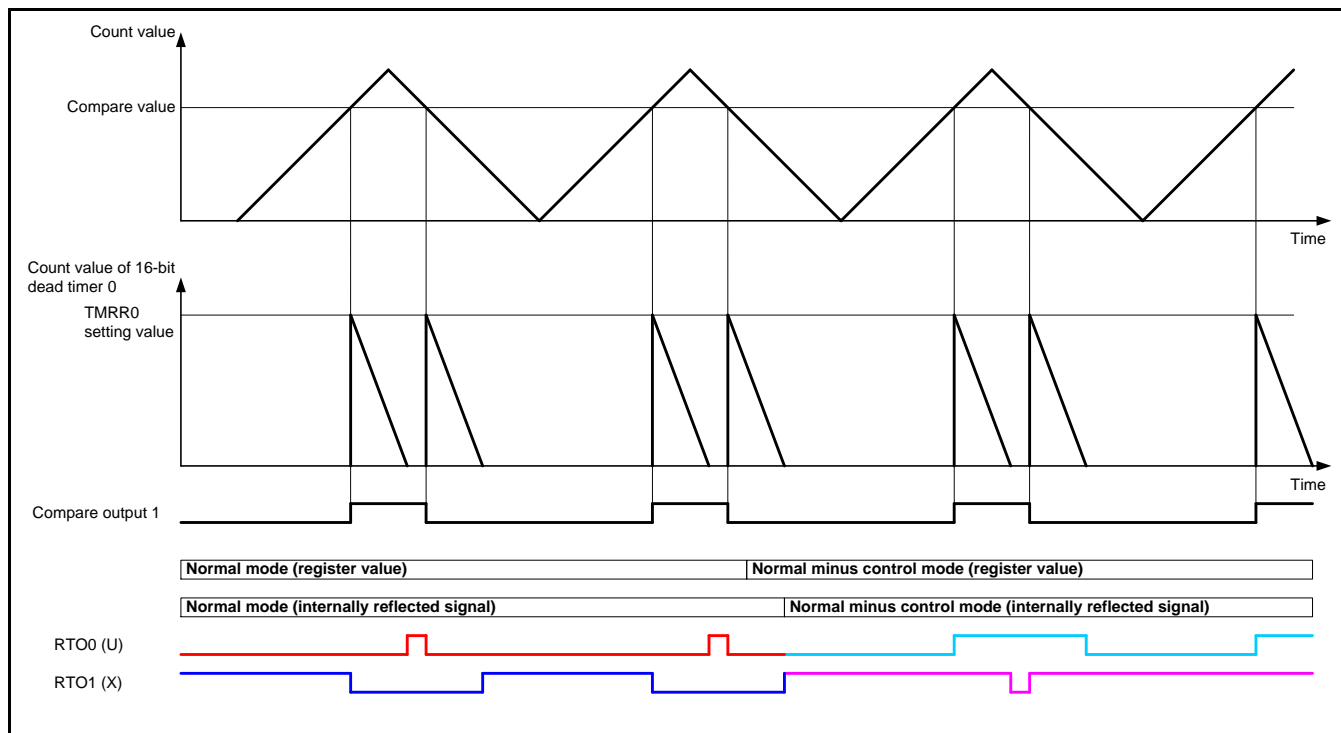
**e) Changing the Minus Control Mode while the Dead Time Timer Mode is Active**

When you change the minus control mode while the dead time timer mode is active, you will need to reflect the settings of minus control mode while the dead timer is inactive and the trigger input (compare output) is "L" in order to prevent the point of variation of U and X from being overlapped.

**Figure 3-14 Changing from Normal Mode to Normal Minus Control Mode While the Dead Time Timer Mode is Active #1**



**Figure 3-15 Changing from Normal Mode to Normal Minus Control Mode while the Dead Time Timer Mode is Active #2**



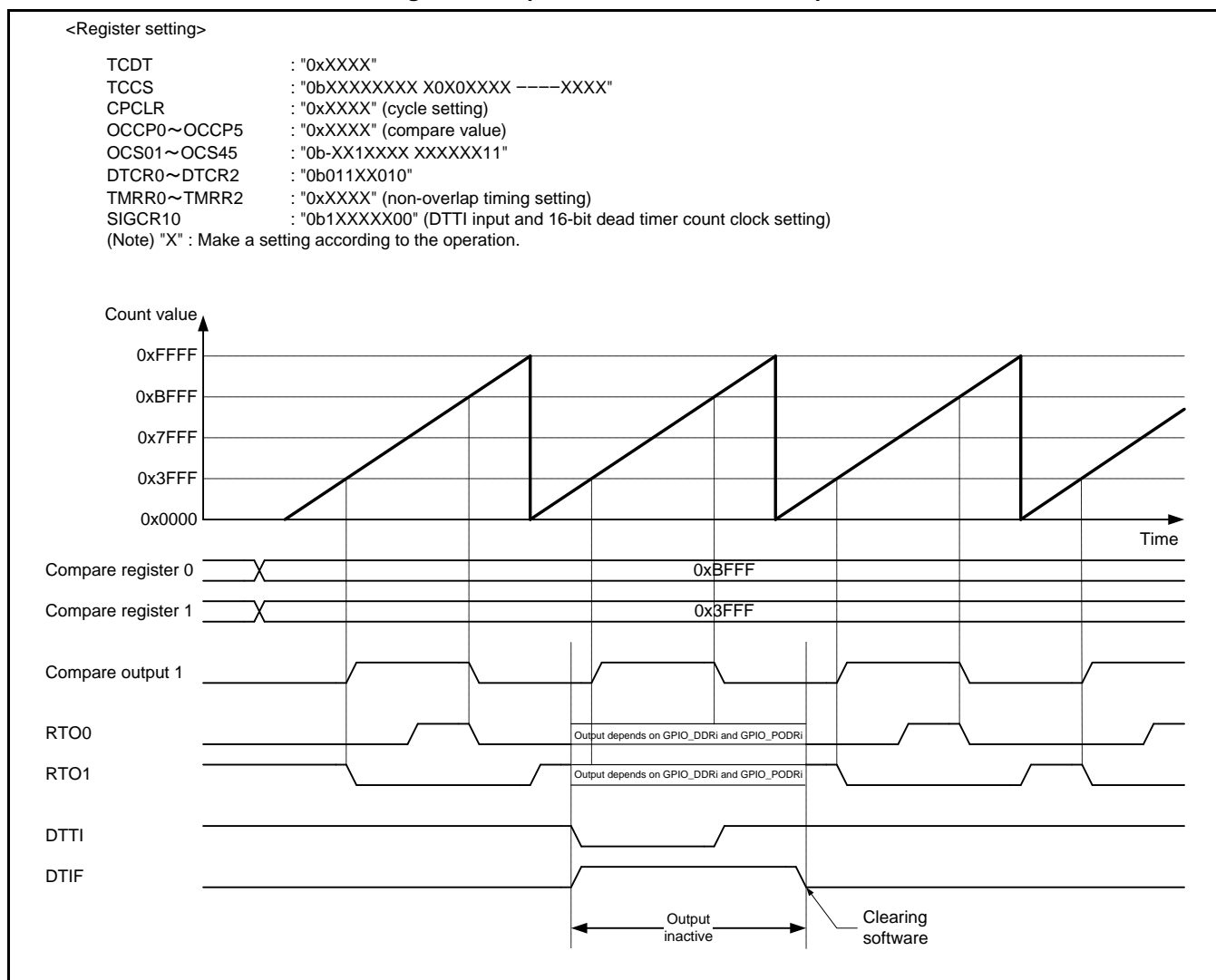
#### (9) Operation of the DTTI Pin Control

If "1" is set in DTIE in the waveform control register 1 (SIGCR1), output in the range of RTO0 to RTO5 can be controlled by using the DTTI pin. When "L" is detected from the DTTI pin, the function of pins RTO0 to RTO5 switches to the port function. This function continues until the interrupt flag (DTIF in the SIGCR1 register) is cleared. DTTI signals from an external pin is enabled/disabled and software DTTI is generated outside the waveform generator module. (See Figure 2-2)

#### (10) Operation of DTTI Pin Input

Even if "L" level of the DTTI pin input is detected, the timer continues its operation while the waveform generator is active.

Figure 3-16 Operation of the Valid DTTI Input



### (11) DTTI Operation of the Waveform Control Register 2 (SIGCR2)

DTTI output of the waveform control register 2 is designed to compute OR with the DTTI pin input and generate DTTI input. Therefore, if you set this register to "0", the operation is always in the DTTI input state and any inputs to the DTTI pin will be ignored.

When you clear this register by writing "1" to this register, the value of the DTTI input pin will be used.

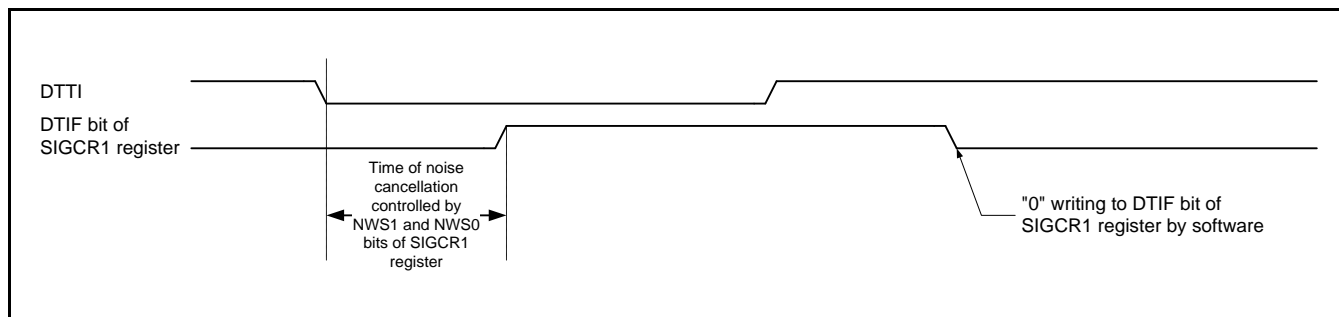
### (12) DTTI Pin Noise Cancellation Feature

When you set the NRSL of the waveform control register 1 (SIGCR1) to "1", the noise cancellation feature of the DTTI pin input will become effective. Once the noise cancellation feature becomes effective, the time required to have the output pins (RTO0 to RTO5) fixed to inactive level will be delayed by the 4, 8, 16, or 32 peripheral clock cycles (to be selected by NWS[1:0] of the SIGCR1 register). Since the noise cancellation circuit uses resources, the input will be invalid even if the DTTI input becomes effective in a mode where the oscillation stops (i.e. stop mode).

### (13) DTTI Interrupt

When "L" level of the DTTI is detected, the DTTI interrupt flag (DTIF of the SIGCR1 register) is set to "1" after the noise cancellation time has elapsed, and the interrupt request will be transmitted to the interrupt controller.

Figure 3-17 DTTI Interrupt Timing



**Notes:**

- If the value of NWS1 and NWS0 bits of the SIGCR1 register changes during the noise cancellation time, the greater noise cycle value (NWS[1:0]) will become effective.
- DTIF of the SIGCR1 register can be cleared by software only.

### (14) Output level Conversion Control of Waveform Generator

The output level of waveform generators can be set individually according to the polarity of the switch connected externally.(See Figure 2-3)



### 3.1. Interrupts for Waveform Generator

The interrupts for waveform generator include underflow interrupt and DTTI interrupt.

Table 3-2 and Table 3-3 show the interrupt control bits and interrupt factors of the waveform generator.

**Table 3-2 Interrupt Control Bits and Interrupt Factor of the Waveform Generator #1**

	16-bit Dead Timer 0/1/2			DTTI
Interrupt request flag bit	16-bit dead timer state control register 0 (DTCR0) TMIF0	16-bit dead timer state control register 1 (DTCR1) TMIF1	16-bit dead timer state control register 2 (DTCR2) TMIF2	Waveform control register 1 (SIGCR1) DTIF
Interrupt request enable bit	16-bit dead timer state control register 0 (DTCR0) TMIE0	16-bit dead timer state control register 1 (DTCR1) TMIE1	16-bit dead timer state control register 2 (DTCR2) TMIE2	-
Interrupt factor	16-bit dead timer 0 underflow	16-bit dead timer 0 underflow	16-bit dead timer 0 underflow	Detection of "L" level at DTTI

For the waveform generator, TMIF0/TMIF1/TMIF2 of the 16-bit dead timer state control register (DTCR) will be set to "1" when an underflow occurs at the 16-bit dead timer and TMD2 to TMD0/TMD5 to TMD3/TMD8 to TMD6 of the DTCR register is either "0b000" or "0b001". If interrupt requests are enabled (DTCR register; TMIE0/TMIE1/TMIE2 (bit27/bit19/ bit11)=1) in this state, an interrupt request is output to the interrupt controller.

**Table 3-3 Interrupt Control Bits and Interrupt Factor of the Waveform Generator #2**

	16-bit Dead Timer 0/1/2		
Interrupt request flag bit	16-bit dead timer reload interrupt register (DTIR) DTRIF0	16-bit dead timer reload interrupt register (DTIR) DTRIF1	16-bit dead timer reload interrupt register (DTIR) DTRIF2
Interrupt request enable bit	16-bit dead timer reload interrupt register (DTIR) DTRIE0	16-bit dead timer reload interrupt register (DTIR) DTRIE1	16-bit dead timer reload interrupt register (DTIR) DTRIE2
Interrupt factor	While 16-bit dead timer is in operation, reload occurs before the timer underflows	While 16-bit dead timer is in operation, reload occurs before the timer underflows	While 16-bit dead timer is in operation, reload occurs before the timer underflows

If a reload occurs before the timer underflows while the 16-bit dead timer is active, the interrupt request flag bit will be set. An interrupt request will be output to the interrupt controller when the corresponding interrupt request enable bit is enabled.

## 4. Registers

This section explains the registers of waveform generator registers.

A prefix (WFGxx\_) is added to each of the registers of waveform generators except some registers. xx represents a channel number (xx = 00 to 03).

**Table 4-1 List of Waveform Generator Registers**

Abbreviated Register Name	Register Name	See
WFGxx_TMRR0/ WFGxx_TMRR1/ WFGxx_TMRR2	16-bit Dead Timer Register	4.1
WFGxx_DTCR0/ WFGxx_DTCR1/ WFGxx_DTCR2	16-bit Dead Timer State Control Register	4.2
WFGxx_DTIR	16-bit Dead Timer Reload Interrupt Register	4.3
WFGxx_DTMNS	16-bit Dead Timer Minus Control Register	4.4
WFGxx_SIGCR1/ WFGxx_SIGCR2	Waveform Control Register 1/2	4.5
WFGxx_PICS	PPG Output Control Register	4.6
WFGxx_DTCRC0/ WFGxx_DTCRC1/ WFGxx_DTCRC2	16-bit Dead Timer State Control Clear Register	4.7
WFGxx_DTIRC	16-bit Dead Timer Reload Interrupt Clear Register	4.8
WFGxx_SIGCR1C	Waveform Control Clear Register 1	4.9
WFGxx_DTCRS0/ WFGxx_DTCRS1/ WFGxx_DTCRS2	16-bit Dead Timer State Control Set Register	4.10
WFGxx_DTIRS	16-bit Dead Timer Reload Interrupt Set Register	4.11
WFGxx_SIGCR1S	Waveform Control Set Register 1	4.12
WFG02_DTISR/ WFG03_DTISR	DTTI Selection Register	4.13
SDTCR2	Software DTTI Control Register	4.14
EDTCR2	External DTTI Input Control Register	4.15
RTOSEL3/ RTOSEL2/ RTOSEL1/ RTOSEL0	RTO Output Level Conversion Register	4.16

xx: unit number (xx=00, 01, 02, 03)





## 4.1. 16-bit Dead Timer Register n (WFGxx\_TMRRn) (n=0 to 2)

The 16-bit dead timer register n (WFGxx\_TMRRn) (n=0 to 2) sets the compare value of the 16-bit dead timer.

### (1) 16-bit Dead Timer Register 0, 2 (WFGxx\_TMRR0, WFGxx\_TMRR2)

BIT_OFFSET	31-16
BIT_NAME	TR
ACCESS_TYPE	R/W
PROT_TYPE	-
INITIAL_VALUE	00000000_00000001

#### [bit31:16] TR[15:0]: 16-bit dead timer compare value bits

- These bits are used to store compare value of the 16-bit dead timer.
- The value of these registers will be reloaded when the 16-bit dead timer starts operating.
- If the value is rewritten to these registers while the timer is active, this new value will become valid when the timer starts/operates next time.
- In the dead time timer mode, these registers are used to configure non-overlap time.
- Non-overlap time = (set value) × selected clock
- In the timer mode, these registers are used to configure GATE time for PPG timer operation.
- GATE time = (set value) × selected clock

#### Notes:

- When accessing these registers, use a half-word or word access instruction.
- Do not set "0x0000" to these registers.

**(2) 16-bit Dead Timer Register (WFGxx\_TMRR1)**

BIT_OFFSET	15-0
BIT_NAME	TR
ACCESS_TYPE	R/W
PROT_TYPE	-
INITIAL_VALUE	00000000_00000001

**[bit15:0] TR[15:0]: 16-bit dead timer compare value bits**

- These bits are used to store compare value of the 16-bit dead timer.
- The value of these registers will be reloaded when the 16-bit dead timer starts operating.
- If the value is rewritten to these registers while the timer is active, this new value will become valid when the timer starts/operates next time.
- In the dead time timer mode, these registers are used to configure non-overlap time.
- Non-overlap time = (set value) × selected clock
- In the timer mode, these registers are used to configure GATE time for PPG timer operation.
- GATE time = (set value) × selected clock

**Notes:**

- *When accessing this register, use a half-word or word access instruction.*
- *Do not set "0x0000" to these registers.*

## 4.2. 16-bit Dead Timer State Control Register n (WFGxx\_DTCRn) (n=0 to 2)

The 16-bit dead timer state control register n (WFGxx\_DTCRn) is used to control operation mode, permission of interrupt request, interrupt request flag, permission of GATE signal, and output level polarity of the waveform generator. For details on writing to this register, see "5. Precautions for Using".

### (1) 16-bit Dead Timer State Control Register 0 (WFGxx\_DTCR0)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	DMOD0	GTEN1	GTEN0	TMIF0	TMIE0	TMD2	TMD1	TMD0
ACCESS_TYPE	R/W	R/W	R/W	R,W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

#### [bit31] DMOD0: Output polarity control bit

- This bit is used to configure U/V/W output in the dead time timer mode.
- When this bit is set, output polarity of U/V/W will be inverted.
- This bit is cleared by writing "1" to the DMODC0 bit of the DTCRC0 register.
- This bit is set by writing "1" to the DMODS0 bit of the DTCRS0 register.

Value	Description
0	Normal polarity output
1	Inverted polarity output

#### Note:

- This bit does not mean anything if the dead time timer mode is not selected (TMD2 =0).

#### [bit30] GTEN1: GATE signal control bit 1

- This bit is used to control the PPG timer's GATE signal output for the compare output of the output compare.
- If it is set to 0, GATE signal will not be output.
- If it is set to 1, GATE signal will be output.
- This bit is cleared by writing "1" to the GTENC1 bit of the DTCRC0 register.
- This bit is set by writing "1" to the GTENS1 bit of the DTCRS0 register.

Value	Description
0	GATE signal will not be controlled by a compare output 1 of the output compare. (asynchronous mode)
1	GATE signal will be controlled by a compare output 1 of the output compare. (synchronous mode)

#### [bit29] GTEN0: GATE signal control bit 0

- This bit is used to control the PPG timer's GATE signal output for the compare output of the output compare.
- If it is set to 0, GATE signal will not be output.
- If it is set to 1, GATE signal will be output.
- This bit is cleared by writing "1" to the GTENC0 bit of the DTCRC0 register.
- This bit is set by writing "1" to the GTENS0 bit of the DTCRS0 register.

Value	Description
0	GATE signal will not be controlled by a compare output 0 of the output compare. (asynchronous mode)
1	GATE signal will be controlled by a compare output 0 of the output compare. (synchronous mode)

**[bit28] TMIF0: Interrupt request flag bit**

- This bit is used as an interrupt request flag for the 16-bit dead timer.
- This bit is set to "1" when an underflow occurs at the 16-bit dead timer.
- When this bit is set to "0", this bit is cleared. Writing "1" to this bit does not have any effects on this bit.
- This bit is cleared by writing "1" to the TMIFC0 bit of the DTCRC0 register.

Value	Description	
	Read	Write
0	No counter underflow has been detected.	This bit is cleared.
1	A counter underflow has been detected.	This bit remains unaffected.

**Notes:**

- *This bit functions only when the TMD2 to TMD0 are "0b000" or "0b001". Otherwise, the bit is always "0".*
- *If a software clear (write of "0") or a hardware set (an underflow occurs at the 16-bit dead timer 0) occur at the same time, the hardware set has a priority over the software clear.*

**[bit27] TMIE0: Interrupt request enable bit, software trigger bit**

- This bit is used as a software trigger bit and an interrupt enable bit for the 16-bit dead timer.
- When TMD2 to TMD0 are "0b000" or "0b001", this bit will be used as a software trigger for the 16-bit dead timer. When this bit is changed from "0" to "1", it will function as a trigger for the 16-bit dead timer and reload the value to start the down count.
- When this bit is set to "1" and the interrupt request flag bit (TMIF0) is "1", an interrupt request will be transmitted to the CPU.
- This bit is cleared by writing "1" to the TMIEC0 bit of the DTCRC0 register.
- This bit is set by writing "1" to the TMIES0 bit of the DTCRS0 register.

Value	Description
0	An interrupt will not be generated even when an underflow occurs at the 16-bit dead timer.
1	An interrupt will be generated when an underflow occurs at the 16-bit dead timer.

**Note:**

- *If you have the 16-bit dead timer triggered again, make sure to write "0" to this bit before writing "1" to it.*

### [bit26:24] TMD2 to TMD0: Operation mode bits

- These bits are used to select the operation mode for the waveform generator.
- If the TMD2 to TMD0 are "0b000", the compare output 0 and 1 signals of the output compare will be output from RTO0 and RTO1 respectively. The 16-bit dead timer can also be used as a reload timer.
- If the TMD2 to TMD0 are "0b001", the compare output 0 and 1 signals of the output compare will be output from RTO0 and RTO1 respectively when PPG output is disabled (PGEN0 =0, PGEN1 =0 of the PPG output control register (PICS)). When PPG output is enabled (PGEN0 =1, PGEN1 =1 of PPG output control register (PICS)), PPG pulse will be output from RTO0 and RTO1 respectively while the compare output 0 and 1 signals of the output compare are "H". The 16-bit dead timer can also be used as a reload timer.
- The output compare output0 and rising edge of output compare output 1 function as a trigger and activates the 16-bit dead timer when TMD2 to TMD0 are "0b010". If PPG output is disabled (PICS:PGEN0="0", PICS:PGEN1="0"), RTO0 and RTO1 will be output "H" from the start of the 16-bit dead timer to the stop. If PPG output is enabled (PICS:PGEN0="1", PICS:PGEN1="1"), PPG pulse will be output until the 16-bit dead timer stops. (Timer mode)
- When TMD2 to TMD0 are "0b100", non-overlap signal will be generated at OUT signal. (Dead time timer mode)

TMD2	TMD1	TMD0	Description
0	0	0	Output compare output signal will be output.
0	0	1	If PPG output is disabled, output compare output signal will be output. If PPG output is enabled, PPG pulse will be output while the output compare output signal is "H".
0	1	0	The rising edge of output compare signal functions as a trigger and activates the 16-bit dead timer. If PPG output is disabled, "H" will be output until the 16-bit dead timer stops. If PPG output is enabled, PPG pulse will be output until the 16-bit dead timer stops. (Timer mode)
1	0	0	Non-overlap signal will be generated at output compare output signal. (Dead time timer mode)
1	1	1	Disabled
Others			Disabled

## (2) 16-bit dead Timer State Control Register 1 (WFGxx\_DTCR1)

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	DMOD1	GTEN3	GTEN2	TMIF1	TMIE1	TMD5	TMD4	TMD3
ACCESS_TYPE	R/W	R/W	R/W	R,W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

### [bit23] DMOD1: Output polarity control bit

- This bit is used to configure U/V/W output in the dead time timer mode.
- When this bit is set, output polarity of U/V/W will be inverted.
- This bit is cleared by writing "1" to the DMODC1 bit of the DTCRC1 register.
- This bit is set by writing "1" to the DMODS1 bit of the DTCRS1 register.

Value	Description
0	Normal polarity output
1	Inverted polarity output

#### Note:

- This bit does not mean anything if the dead time timer mode is not selected (TMD5=0).

### [bit22] GTEN3: GATE signal control bit 3

- This bit is used to control the PPG timer's GATE signal output for the compare output of the output compare.
- If it is set to 0, GATE signal will not be output.
- If it is set to 1, GATE signal will be output.
- This bit is cleared by writing "1" to the GTENC3 bit of the DTCRC1 register.
- This bit is set by writing "1" to the GTENS3 bit of the DTCRS1 register.

Value	Description
0	GATE signal will not be controlled by a compare output of the output compare. (asynchronous mode)
1	GATE signal will be controlled by a compare output of the output compare. (synchronous mode)

### [bit21] GTEN2: GATE signal control bit 2

- This bit is used to control the PPG timer's GATE signal output for the compare output of the output compare.
- If it is set to 0, GATE signal will not be output.
- If it is set to 1, GATE signal will be output.
- This bit is cleared by writing "1" to the GTENC2 bit of the DTCRC1 register.
- This bit is set by writing "1" to the GTENS2 bit of the DTCRS1 register.

Value	Description
0	GATE signal will not be controlled by a compare output of the output compare. (asynchronous mode)
1	GATE signal will be controlled by a compare output of the output compare. (synchronous mode)

### [bit20] TMIF1: Interrupt request flag bit

- This bit is used as an interrupt request flag for the 16-bit dead timer.
- This bit is set to "1" when an underflow occurs at the 16-bit dead timer.
- When this bit is set to "0", this bit is cleared. Writing "1" to this bit does not have any effects on this bit.
- This bit is cleared by writing "1" to the TMIFC1 bit of the DTCRC1 register.

Value	Description	
	Read	Write
0	No counter underflow has been detected.	This bit is cleared.
1	A counter underflow has been detected.	This bit remains unaffected.

#### Notes:

- *This bit functions only when the TMD5 to TMD3 are "0b000" or "0b001". Otherwise, the bit is always "0".*
- *If a software clear (write of "0") or a hardware set (an underflow occurs at the 16-bit dead timer 1) occur at the same time, the hardware set has a priority over the software clear.*

### [bit19] TMIE1: Interrupt request flag bit

- This bit is used as a software trigger bit and an interrupt enable bit for the 16-bit dead timer.
- When TMD5 to TMD3 are "0b000" or "0b001", this bit will be used as a software trigger for the 16-bit dead timer. When this bit is changed from "0" to "1", it will function as a trigger for the 16-bit dead timer and reload the value to start the down count.
- When this bit is set to "1" and the interrupt request flag bit (TMIF1) is "1", an interrupt request will be transmitted to the CPU.
- This bit is cleared by writing "1" to the TMIEC1 bit of the DTCRC1 register.
- This bit is set by writing "1" to the TMIES1 bit of the DTCRS1 register.

Value	Description
0	An interrupt will not be generated even when an underflow occurs at the 16-bit dead timer.
1	An interrupt will be generated when an underflow occurs at the 16-bit dead timer.

#### Note:

- *If you have the 16-bit dead timer triggered again, make sure to write "0" to this bit before writing "1" to it.*

**[bit18:16] TMD5 to TMD3: Operation mode bits**

- These bits are used to select the operation mode for the waveform generator.
- If the TMD5 to TMD3 are "0b000", the compare output2 and 3 signals of the output compare will be output from RTO2 and RTO3 respectively. The 16-bit dead timer can also be used as a reload timer.
- If the TMD5 to TMD3 are "0b001", the compare output2 and 3 signals of the output compare will be output from RTO2 and RTO3 respectively when PPG output is disabled (PGEN2 =0, PGEN3 =0 of the PPG output control register (PICS)). When PPG output is enabled (PGEN2 =1, PGEN3 =1 of PPG output control register (PICS)), PPG pulse will be output from RTO2 and RTO3 respectively while the compare output 2 and 3 signals of the output compare are "H". The 16-bit dead timer can also be used as a reload timer.
- The rising edge of each OUT signal functions as a trigger and activates the 16-bit dead timer when TMD5 to TMD3 are "0b010". If PPG output is disabled, "H" will be output until the 16-bit dead timer stops. If PPG output is enabled, PPG pulse will be output until the 16-bit dead timer stops. (Timer mode)
- When TMD5 to TMD3 are "0b100", non-overlap signal will be generated at OUT signal. (Dead time timer mode)

TMD5	TMD4	TMD3	Description
0	0	0	OUT signal will be output.
0	0	1	If PPG output is disabled, OUT signal will be output. If PPG output is enabled, PPG pulse will be output while the OUT signal is "H".
0	1	0	The rising edge of each OUT signal functions as a trigger and activates the 16-bit dead timer. If PPG output is disabled, "H" will be output until the 16-bit dead timer stops. If PPG output is enabled, PPG pulse will be output until the 16-bit dead timer stops. (Timer mode)
1	0	0	Non-overlap signal will be generated at OUT signal. (Dead time timer mode)
1	1	1	Disabled
Others			Disabled



### (3) 16-bit Dead Timer State Control Register 2 (WFGxx\_DTCR2)

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	DMOD2	GTEN5	GTEN4	TMIF2	TMIE2	TMD8	TMD7	TMD6
ACCESS_TYPE	R/W	R/W	R/W	R,W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

#### [bit15] DMOD2: Output polarity control bit

- This bit is used to configure U/V/W output in the dead time timer mode.
- When this bit is set, output polarity of U/V/W will be inverted.
- This bit is cleared by writing "1" to the DMODC2 bit of the DTCRC2 register.
- This bit is set by writing "1" to the DMODS2 bit of the DTCRS2 register.

Value	Description
0	Normal polarity output
1	Inverted polarity output

#### Note:

- This bit does not mean anything if the dead time timer mode is not selected (TMD8=0).

#### [bit14] GTEN5: GATE signal control bit 5

- This bit is used to control the PPG timer's GATE signal output for the compare output of the output compare.
- If it is set to 0, GATE signal will not be output.
- If it is set to 1, GATE signal will be output.
- This bit is cleared by writing "1" to the GTENC5 bit of the DTCRC2 register.
- This bit is set by writing "1" to the GTENS5 bit of the DTCRS2 register.

Value	Description
0	GATE signal will not be controlled by a compare output of the output compare. (asynchronous mode)
1	GATE signal will be controlled by a compare output of the output compare. (synchronous mode)

#### [bit13] GTEN4: GATE signal control bit 4

- This bit is used to control the PPG timer's GATE signal output for the compare output of the output compare.
- If it is set to 0, GATE signal will not be output.
- If it is set to 1, GATE signal will be output.
- This bit is cleared by writing "1" to the GTENC4 bit of the DTCRC2 register.
- This bit is set by writing "1" to the GTENS4 bit of the DTCRS2 register.

Value	Description
0	GATE signal will not be controlled by a compare output of the output compare. (asynchronous mode)
1	GATE signal will be controlled by a compare output of the output compare. (synchronous mode)

**[bit12] TMIF2: Interrupt request flag bit**

- This bit is used as an interrupt request flag for the 16-bit dead timer.
- This bit is set to "1" when an underflow occurs at the 16-bit dead timer.
- When this bit is set to "0", this bit is cleared. Writing "1" to this bit does not have any effects on this bit.
- This bit is cleared by writing "1" to the TMIFC2 bit of the DTCRC2 register.

Value	Description	
	Read	Write
0	No counter underflow has been detected.	This bit is cleared.
1	A counter underflow has been detected.	This bit remains unaffected.

**Notes:**

- *This bit functions only when the TMD8 to TMD6 are "0b000" or "0b001". Otherwise, the bit is always "0".*
- *If a software clear (write of "0") or a hardware set (an underflow occurs at the 16-bit dead timer 2) occur at the same time, the hardware set has a priority over the software clear.*

**[bit11] TMIE2: Interrupt request flag bit**

- This bit is used as a software trigger bit and an interrupt enable bit for the 16-bit dead timer.
- When TMD8 to TMD6 are "0b000" or "0b001", this bit will be used as a software trigger for the 16-bit dead timer. When this bit is changed from "0" to "1", it will function as a trigger for the 16-bit dead timer and reload the value to start the down count.
- When this bit is set to "1" and the interrupt request flag bit (TMIF2) is "1", an interrupt request will be transmitted to the CPU.
- This bit is cleared by writing "1" to the TMIEC2 bit of the DTCRC2 register.
- This bit is set by writing "1" to the TMIES2 bit of the DTCRS2 register.

Value	Description
0	An interrupt will not be generated even when an underflow occurs at the 16-bit dead timer.
1	An interrupt will be generated when an underflow occurs at the 16-bit dead timer.

**Note:**

- *If you have the 16-bit dead timer triggered again, make sure to write "0" to this bit before writing "1" to it.*

**[bit10:8] TMD8 to TMD6: Operation mode bits**

- These bits are used to select the operation mode for the waveform generator.
- If the TMD8 to TMD6 are "0b000", the compare output 4 and 5 signals of the output compare will be output from RTO4 and RTO5 respectively. The 16-bit dead timer can also be used as a reload timer.
- If the TMD8 to TMD6 are "0b001", the compare output 4 and 5 signals of the output compare will be output from RTO4 and RTO5 respectively when PPG output is disabled (PGEN4 =0, PGEN5 =0 of the PPG output control register (PICS)). When PPG output is enabled (PGEN4 =1, PGEN5 =1 of PPG output control register (PICS)), PPG pulse will be output from RTO4 and RTO5 respectively while the compare output 4 and 5 signals of the output compare are "H". The 16-bit dead timer can also be used as a reload timer.
- The rising edge of each OUT signal functions as a trigger and activates the 16-bit dead timer when TMD8 to TMD6 are "0b010". If PPG output is disabled, "H" will be output until the 16-bit dead timer stops. If PPG output is enabled, PPG pulse will be output until the 16-bit dead timer stops. (Timer mode)
- When TMD8 to TMD6 are "0b100", non-overlap signal will be generated at OUT signal. (Dead time timer mode)

TMD8	TMD7	TMD6	Description
0	0	0	OUT signal will be output.
0	0	1	If PPG output is disabled, OUT signal will be output. If PPG output is enabled, PPG pulse will be output while the OUT signal is "H".
0	1	0	The rising edge of each OUT signal functions as a trigger and activates the 16-bit dead timer. If PPG output is disabled, "H" will be output until the 16-bit dead timer stops. If PPG output is enabled, PPG pulse will be output until the 16-bit dead timer stops. (Timer mode)
1	0	0	Non-overlap signal will be generated at OUT signal. (Dead time timer mode)
1	1	1	Disabled
Others			Disabled

### 4.3. 16-bit Dead Timer Reload Interrupt Register (WFGxx\_DTIR)

The 16-bit dead timer reload interrupt register (WFGxx\_DTIR) is used to control the interrupt request when the timer is reloaded before it underflows as well as to control the interrupt request permission. For details on writing to this register, see Section "5. Precautions for Using".

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	DTRIF2	DTRIE2	DTRIF1	DTRIE1	DTRIF0	DTRIE0	Reserved	
ACCESS_TYPE	R,W	R/W	R,W	R/W	R,W	R/W	R0,W0	
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	00	

#### [bit23] DTRIF2: 16-bit dead timer 2 reload interrupt flag bit

- For the 16-bit dead timer 2, if the timer is reloaded before it underflows, this bit will be set to "1".
- An interrupt request is generated when this bit and the interrupt request enable bit (DTIR: DTRIE2) are "1".
- This bit is cleared when "0" is written to it. A write of "1" does not change this bit and has no influence on others.
- This bit is cleared by writing "1" to the DTRIFC2 bit of the DTIRC register.

Value	Description	
	Read	Write
0	No reload of the dead timer has been detected.	This bit is cleared.
1	A reload of the dead timer has been detected.	This bit remains unaffected.

#### Note:

- If a software clear (write of "0") and a hardware set occur at the same time, the hardware set has a priority over the software clear. Then, this bit will be set.

#### [bit22] DTRIE2: 16-bit dead timer 2 reload interrupt enable bit

- This bit enables or disables the output of interrupts to the CPU.
- An interrupt request is generated when this bit and the interrupt request flag bit (DTIR: DTRIF2) are "1".
- This bit is cleared by writing "1" to the DTRIEC2 bit of the DTIRC register.
- This bit is set by writing "1" to the DTRIES2 bit of the DTIRS register.

Value	Description
0	An interrupt will not be generated even when a reload occurs at the 16-bit dead timer.
1	An interrupt will be generated when a reload occurs at the 16-bit dead timer.

#### [bit21] DTRIF1: 16-bit dead timer 1 reload interrupt flag bit

- For the 16-bit dead timer 1, if the timer is reloaded before it underflows, this bit will be set to "1".
- An interrupt request is generated when this bit and the interrupt request enable bit (DTIR: DTRIE1) are "1".
- This bit is cleared when "0" is written to it. A write of "1" does not change this bit and has no influence on others.
- This bit is cleared by writing "1" to the DTRIFC1 bit of the DTIRC register.

Value	Description	
	Read	Write
0	No counter underflow has been detected.	This bit is cleared.
1	A counter underflow has been detected.	This bit remains unaffected.

**Note:**

- If a software clear (write of "0") and a hardware set occur at the same time, the hardware set has a priority over the software clear. Then, this bit will be set.

**[bit20] DTRIE1: 16-bit dead timer 1 reload interrupt enable bit**

- This bit enables or disables the output of interrupts to the CPU.
- An interrupt request is generated when this bit and the interrupt request flag bit (DTIR: DTRIF1) are "1".
- This bit is cleared by writing "1" to the DTRIEC1 bit of the DTIRC register.
- This bit is set by writing "1" to the DTRIES1 bit of the DTIRS register.

Value	Description
0	An interrupt will not be generated even when a reload occurs at the 16-bit dead timer.
1	An interrupt will be generated when a reload occurs at the 16-bit dead timer.

**[bit19] DTRIF0: 16-bit dead timer 0 reload interrupt flag bit**

- For the 16-bit dead timer 0, if the timer is reloaded before it underflows, this bit will be set to "1".
- An interrupt request is generated when this bit and the interrupt request enable bit (DTIR: DTRIE0) are "1".
- This bit is cleared when "0" is written to it. A write of "1" does not change this bit and has no influence on others.
- This bit is cleared by writing "1" to the DTRIFC0 bit of the DTIRC register.

Value	Description	
	Read	Write
0	No counter underflow has been detected.	This bit is cleared.
1	A counter underflow has been detected.	This bit remains unaffected.

**Note:**

- If a software clear (write of "0") and a hardware set occur at the same time, the hardware set has a priority over the software clear. Then, this bit will be set.

**[bit18] DTRIE0: 16-bit dead timer 1 reload interrupt enable bit**

- This bit enables or disables the output of interrupts to the CPU.
- An interrupt request is generated when this bit and the interrupt request flag bit (DTIR: DTRIF0) are "1".
- This bit is cleared by writing "1" to the DTRIEC0 bit of the DTIRC register.
- This bit is set by writing "1" to the DTRIES0 bit of the DTIRS register.

Value	Description
0	Normal polarity output
1	Inverted polarity output

**[bit17:16] Reserved: Reserved bits**

#### 4.4. 16-bit Dead Timer Minus Control Register (WFGxx\_DTMNS)

The 16-bit dead timer minus control register (DTMNS) is used to configure the minus control of the dead time function.

You will need to be careful when setting the dead time function selection bits (MNS2 to MNS0) since this register has key enable bits (KEY[1:0]).

To set values to the MNS2 to MNS0 bits, write the values in the following sequence: KEY1, KEY0=00 and MNS2 to MNS0="value you wish to set", → KEY1, KEY0=01 and MNS2 to MNS0="value you wish to set (the same value as the last time)" → KEY1, KEY0=10 and MNS2 to MNS0="value you wish to set (the same value as the last time)" → KEY1, KEY0=11 and MNS2 to MNS0="value you wish to set (the same value as the last time)". For the MNS2 to MNS0, the value will be reflected when you write the value for the fourth time (when writing to KEY1, KEY0=11). Without following this process (if you write values to or read values from other registers in the middle of the writing process, the value written is incorrect, or if you read values from this register in the middle of the writing process), the value written to this register will be invalid.

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	KEY		Reserved			MNS2	MNS1	MNS0
ACCESS_TYPE	R0,W		R0,W0			R,W	R,W	R,W
PROT_TYPE	-							
INITIAL_VALUE	00		000			0	0	0

##### [bit7:6] KEY[1:0]: Key enable bits

- It is a key code register used for the settings of MNS2 to MNS0.
- To set values to the MNS2 to MNS0 bits, write the values in the following sequence: KEY1, KEY0=00 and MNS2 to MNS0="value you wish to set", → KEY1, KEY0=01 and MNS2 to MNS0="value you wish to set (the same value as the last time)" → KEY1, KEY0=10 and MNS2 to MNS0="value you wish to set (the same value as the last time)" → KEY1, KEY0=11 and MNS2 to MNS0="value you wish to set (the same value as the last time)". For the MNS2 to MNS0, the value will be reflected when you write the value for the fourth time (when writing to KEY1, KEY0=11).
- Without following this process (if you write values to or read values from other registers in the middle of the writing process, the value written is incorrect, or if you read values from this register in the middle of the writing process), the value written to this register will be invalid.
- These bits are always "0" when read.

##### [bit5:3] Reserved: Reserved bits

##### [bit2] MNS2: Dead time function selection bit

- Select the control of the dead timer function for RTO4 and RTO5.
- When you set this bit to "0": Minus control of the dead timer function will not be executed.
- When you set this bit to "1": Minus control of the dead timer function will be executed.
- When you set a value to this bit, follow the writing procedure using the KEY[1:0] bits.

Value	Description
0	Minus control of the dead timer function will not be executed.
1	Minus control of the dead timer function will be executed.



**[bit1] MNS1: Dead time function selection bit**

- Select the control of the dead timer function for RTO2 and RTO3.
- When you set this bit to "0": Minus control of the dead timer function will not be executed.
- When you set this bit to "1": Minus control of the dead timer function will be executed.
- When you set a value to this bit, follow the writing procedure using the KEY[1:0] bits.

Value	Description
0	Minus control of the dead timer function will not be executed.
1	Minus control of the dead timer function will be executed.

**[bit0] MNS0: Dead time function selection bit**

- Select the control of the dead timer function for RTO0 and RTO1.
- When you set this bit to "0": Minus control of the dead timer function will not be executed.
- When you set this bit to "1": Minus control of the dead timer function will be executed.
- When you set a value to this bit, follow the writing procedure using the KEY[1:0] bits.

Value	Description
0	Minus control of the dead timer function will not be executed.
1	Minus control of the dead timer function will be executed.

## 4.5. Waveform Control Register 1/2 (WFGxx\_SIGCR1, WFGxx\_SIGCR2)

The waveform control register 1/2 (SIGCR1, SIGCR2) is used to control the operating clock frequency, noise cancel function valid settings, DTTI input valid settings, and DTTI interrupts. For details on writing to this register, see Section "5. Precautions for Using".

### (1) Waveform Control Register 1 (WFGxx\_SIGCR1)

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	DTIE	DTIF	NRSL	DCK			NWS	
ACCESS_TYPE	R/W	R,W	R/W	R/W			R/W	
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	000			00	

#### [bit23] DTIE: DTTI input validating bit

- This bit is used to validate the output level control DTTI signals for the RTO0 to RTO5 pins.
- This bit is cleared by writing "1" to the DTIEC bit of the SIGCR1C register.
- This bit is set by writing "1" to the DTIES bit of the SIGCR1S register.

Value	Description
0	Invalidate the DTTI
1	Validate the DTTI

#### [bit22] DTIF: DTTI Interrupt flag bit

- This bit is an interrupt flag for the DTTI.
- When the DTTI input becomes valid (DTIE=1) and "L" level of the DTTI is detected, this bit will be set and an interrupt request will be generated.
- If DTTI input is enabled (DTIE="1"), the DTTI interrupt requests of waveform generators 00 and 01 can be output to error detection output pins ch.0 (ERDS0) and ch.1 (ERDS1), respectively.
- When this bit is set to "0": This bit is cleared.
- When this bit is set to "1": This bit remains unaffected.
- This bit is cleared by writing "1" to the DTIFC bit of the SIGCR1C register.

Value	Description	
	Read	Write
0	No counter underflow has been detected.	This bit is cleared.
1	A counter underflow has been detected.	This bit remains unaffected.

#### Notes:

- When the noise cancel function becomes valid (NRSL=1) and a noise pulse is generated, this bit will be set to "1".
- If a set operation by hardware (DTTI "L" level detection) and a clear operation by means of software clear (writing "0") occur simultaneously, the set operation by hardware takes precedence.
- To output a DTTI interrupt request to an external pin, a setting needs to be made using the POF[2:0] bits in the port setting register (PPC\_PCFGR<sub>ij</sub>) for the I/O port. For details, see "CHAPTER: I/O Port" and Section "Output Resource Selection of I/O Port Various Settings" in "CHAPTER: Appendix".



### [bit21] NRSL: Noise cancel function validating bit

- This bit is used for validating the noise cancel function.
- The noise cancel circuit receives DTTI input signals until an overflow occurs at the counter while it remains at the "L" level. The counter is a N-bit counter operated by the "L" level input.
- Value for N will be either 2, 3, 4, or 5 based on the settings of the NWS[1:0] bits.
- This bit is cleared by writing "1" to the NRSLC bit of the SIGCR1C register.
- This bit is set by writing "1" to the NRSLS bit of the SIGCR1S register.

Value	Description
0	Noise cancel circuit of the DTTI will be invalidated
1	Noise cancel circuit of the DTTI will be validated

#### Note:

- To cancel the noise pulse width, you will need an approximately  $2^n$  peripheral clock. If you select the noise cancel circuit, the input will be invalidated while it is in the mode that the peripheral clock is inactive (i.e. stop mode).

### [bit20:18] DCK[2:0]: Operating clock selection bits

These bits are used to select the operating clock for the 16-bit dead timer.

Value	Description
000	$\phi$
001	$\phi / 2$
010	$\phi / 4$
011	$\phi / 8$
100	$\phi / 16$
101	$\phi / 32$
110	$\phi / 64$
111	Setting disabled

$\phi$ : peripheral clock

### [bit17:16] NWS[1:0]: DTTI noise width selection bits

These bits are used to select the DTTI noise pulse width to be removed.

Value	Description
00	Cancel the 4-peripheral clock cycle noise
01	Cancel the 8-peripheral clock cycle noise
10	Cancel the 16-peripheral clock cycle noise
11	Cancel the 32-peripheral clock cycle noise

## (2) Waveform Control Register 2 (WFGxx\_SIGCR2)

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	PSEL2		PSEL1		PSEL0		Reserved	DTTI
ACCESS_TYPE	R/W		R/W		R/W		R0,W0	R/W
PROT_TYPE	-							
INITIAL_VALUE	00		00		00		0	1

### [bit7:6] PSEL2[1:0]: PPG input channel selection bits (RTO4, RTO5)

- These bits are used to select the PPG input for the RTO4 and RTO5.
- These bits are also used to select the GATE output destination for PPG.
- Settings of "0b11" is prohibited.

Value	Description
00	PPG0
01	PPG2
10	PPG4
11	Setting disabled (operation is not guaranteed)

### [bit5:4] PSEL1[1:0]: PPG input channel selection bits (RTO2, RTO3)

- These bits are used to select the PPG input for the RTO2 and RTO3.
- These bits are also used to select the GATE output destination for PPG.
- Settings of "0b11" is prohibited.

Value	Description
00	PPG0
01	PPG2
10	PPG4
11	Setting disabled (operation is not guaranteed)

### [bit3:2] PSEL0[1:0]: PPG input channel selection bits (RTO0, RTO1)

- These bits are used to select the PPG input for the RTO0 and RTO1.
- These bits are also used to select the GATE output destination for PPG.
- Settings of "0b11" is prohibited.

Value	Description
00	PPG0
01	PPG2
10	PPG4
11	Setting disabled (operation is not guaranteed)

### [bit1] Reserved: Reserved bit

### [bit0] DTTI: Software DTTI

- Writing "0" will set the DTTI.
- Writing "1" will clear the DTTI.

Value	Description
0	DTTI set
1	DTTI clear



**Note:**

- *Since it computes the logical OR with DTTI input, Software DTTI bit depends on the DTTI input level.*

## 4.6. PPG Output Control Register (WFGxx\_PICS)

The PPG output control register (WFGxx\_PICS) is used to control the PPG output.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	PGEN5	PGEN4	PGEN3	PGEN2	PGEN1	PGEN0	Reserved	
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R0,W0	
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	00	

### [bit31] PGEN5: PPG output enable bit

This bit is used to select PPG output to the RTO5.

Value	Description
0	Disable PPG output to the RTO5
1	Enable PPG output to the RTO5

### [bit30] PGEN4: PPG output enable bit

This bit is used to select PPG output to the RTO4.

Value	Description
0	Disable PPG output to the RTO4
1	Enable PPG output to the RTO4

### [bit29] PGEN3: PPG output enable bit

This bit is used to select PPG output to the RTO3.

Value	Description
0	Disable PPG output to the RTO3
1	Enable PPG output to the RTO3

### [bit28] PGEN2: PPG output enable bit

This bit is used to select PPG output to the RTO2.

Value	Description
0	Disable PPG output to the RTO2
1	Enable PPG output to the RTO2

### [bit27] PGEN1: PPG output enable bit

This bit is used to select PPG output to the RTO1.

Value	Description
0	Disable PPG output to the RTO1
1	Enable PPG output to the RTO1



**[bit26] PGEN0: PPG output enable bit**

This bit is used to select PPG output to the RTO0.

Value	Description
0	Disable PPG output to the RTO0
1	Enable PPG output to the RTO0

**[bit25:24] Reserved: Reserved bits**

## 4.7. 16-bit Dead Timer State Control Clear Register (WFGxx\_DTCRCn) (n=0 to 2)

16-bit dead timer state control clear register n (WGxx\_DTCRCn) is used to clear the bit of 16-bit dead timer state control register (WFGxx\_DTCRn).

### (1) 16-bit Dead Timer State Control Clear Register 0 (WFGxx\_DTCRC0)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	DMODC0	GTENC1	GTENC0	TMIFC0	TMIEC0	Reserved		
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W0		
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	000		

#### [bit31] DMODC0: Output polarity control clear bit

Value	Description
0	No effect
1	Clear the DMOD0 bit

#### [bit30] GTENC1: GATE signal control clear bit 1

Value	Description
0	No effect
1	Clear the GTEN1 bit

#### [bit29] GTENC0: GATE signal control clear bit 0

Value	Description
0	No effect
1	Clear the GTEN0 bit

#### [bit28] TMIFC0: Interrupt request flag clear bit

Value	Description
0	No effect
1	Clear the TMIF0 bit

#### [bit27] TMIEC0: Interrupt request enable clear bit

Value	Description
0	No effect
1	Clear the TMIE0 bit

#### [bit26:24] Reserved: Reserved bits

**(2) 16-bit Dead Timer State Control Clear Register 1 (WFGxx\_DTCRC1)**

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	DMODC1	GTENC3	GTENC2	TMIFC1	TMIEC1	Reserved		
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W0		
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	000		

**[bit23] DMODC1: Output polarity control clear bit**

Value	Description
0	No effect
1	Clear the DMOD1 bit

**[bit22] GTENC3: GATE signal control clear bit 3**

Value	Description
0	No effect
1	Clear the GTEN3 bit

**[bit21] GTENC2: GATE signal control clear bit 2**

Value	Description
0	No effect
1	Clear the GTEN2 bit

**[bit20] TMIFC1: Interrupt request flag clear bit**

Value	Description
0	No effect
1	Clear the TMIF1 bit

**[bit19] TMIEC1: Interrupt request enable clear bit**

Value	Description
0	No effect
1	Clear the TMIE1 bit

**[bit18:16] Reserved: Reserved bits**

**(3) 16-bit Dead Timer State Control Clear Register 2 (WFGxx\_DTCRC2)**

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	DMODC2	GTENC5	GTENC4	TMIFC2	TMIEC2	Reserved		
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W0		
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	000		

**[bit15] DMODC2: Output polarity control clear bit**

Value	Description
0	No effect
1	Clear the DMOD2 bit

**[bit14] GTENC5: GATE signal control clear bit 5**

Value	Description
0	No effect
1	Clear the GTEN5 bit

**[bit13] GTENC4: GATE signal control clear bit 4**

Value	Description
0	No effect
1	Clear the GTEN4 bit

**[bit12] TMIFC2: Interrupt request flag clear bit**

Value	Description
0	No effect
1	Clear the TMIF2 bit

**[bit11] TMIEC2: Interrupt request enable clear bit**

Value	Description
0	No effect
1	Clear the TMIE2 bit

**[bit10:8] Reserved: Reserved bits**





## 4.8. 16-bit Dead Timer Reload Interrupt Clear Register (WFGxx\_DTIRC)

16-bit dead timer reload interrupt clear register (WGxx\_DTIRC) is used to clear the bit of 16-bit dead timer reload interrupt register (WFGxx\_DTIR).

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	DTRIFC2	DTRIEC2	DTRIFC1	DTRIEC1	DTRIFC0	DTRIEC0	Reserved	
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W0	
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	00	

### [bit23] DTRIFC2: 16-bit dead timer 2 reload interrupt flag clear bit

Value	Description
0	No effect
1	Clear the DTRIF2 bit

### [bit22] DTRIEC2: 16-bit dead timer 2 reload interrupt enable clear bit

Value	Description
0	No effect
1	Clear the DTRIE2 bit

### [bit21] DTRIFC1: 16-bit dead timer 1 reload interrupt flag clear bit

Value	Description
0	No effect
1	Clear the DTRIF1 bit

### [bit20] DTRIEC1: 16-bit dead timer 1 reload interrupt enable clear bit

Value	Description
0	No effect
1	Clear the DTRIE1 bit

### [bit19] DTRIFC0: 16-bit dead timer 0 reload interrupt flag clear bit

Value	Description
0	No effect
1	Clear the DTRIF0 bit

### [bit18] DTRIEC0: 16-bit dead timer 0 reload interrupt enable clear bit

Value	Description
0	No effect
1	Clear the DTRIE0 bit

### [bit17:16] Reserved: Reserved bits

## 4.9. Waveform Control Clear Register 1 (WFGxx\_SIGCR1C)

Waveform control set register 1 (WGxx\_SIGCR1S) is used to set the bit of waveform control register 1 (WFGxx\_SIGCR1).

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	DTIEC	DTIFC	NRSLC	Reserved				
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W0				
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	00000				

### [bit23] DTIEC: DTTI input validating clear bit

bit	Description
0	No effect
1	Clear the DTIE bit.

### [bit22] DTIFC: DTTI Interrupt flag clear bit

bit	Description
0	No effect
1	Clear the DTIF bit

### [bit21] NRSLC: Noise cancel function validating clear bit

bit	Description
0	No effect
1	Clear the NRSL bit

### [bit20:16] Reserved: Reserved bits



#### 4.10. 16-bit Dead Timer State Control Set Register (WFGxx\_DTCRSn) (n=0 to 2)

16-bit dead timer state control set register n (WFGxx\_DTCRSn) is used to set the bit of 16-bit dead timer state control register (WFGxx\_DTCRn).

##### (1) 16-bit Dead Timer State Control Set Register 0 (WFGxx\_DTCRS0)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	DMODS0	GTENS1	GTENS0	Reserved	TMIES0	Reserved		
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W0	R0,W	R0,W0		
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	000		

##### [bit31] DMODS0: Output polarity control set bit

Value	Description
0	No effect
1	Set the DMOD0 bit.

##### [bit30] GTENS1: GATE signal control set bit 1

Value	Description
0	No effect
1	Set the GTEN1 bit.

##### [bit29] GTENS0: GATE signal control set bit 0

Value	Description
0	No effect
1	Set the GTEN0 bit.

##### [bit28] Reserved: Reserved bit

##### [bit27] TMIES0: Interrupt request enable set bit

Value	Description
0	No effect
1	Set the TMIE0 bit.

##### [bit26:24] Reserved: Reserved bits

**(2) 16-bit Dead Timer State Control Set Register 1 (WFGxx\_DTCRS1)**

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	DMODS1	GTENS3	GTENS2	Reserved	TMIES1	Reserved		
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W0	R0,W	R0,W0		
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	000		

**[bit23] DMODS1: Output polarity control set bit**

Value	Description
0	No effect
1	Set the DMOD1 bit.

**[bit22] GTENS3: GATE signal control set bit 3**

Value	Description
0	No effect
1	Set the GTEN3 bit.

**[bit21] GTENS2: GATE signal control set bit 2**

Value	Description
0	No effect
1	Set the GTEN2 bit.

**[bit20] Reserved: Reserved bit**

**[bit19] TMIES1: Interrupt request enable set bit**

Value	Description
0	No effect
1	Set the TMIE1 bit.

**[bit18:16] Reserved: Reserved bits**



**(3) 16-bit Dead Timer State Control Set Register 2 (WFGxx\_DTCRS2)**

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	DMODS2	GTENS5	GTENS4	Reserved	TMIES2	Reserved		
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W0	R0,W	R0,W0		
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	000		

**[bit15] DMODS2: Output polarity control set bit**

Value	Description
0	No effect
1	Set the DMOD2 bit.

**[bit14] GTENS5: GATE signal control set bit 5**

Value	Description
0	No effect
1	Set the GTEN5 bit.

**[bit13] GTENS4: GATE signal control set bit 4**

Value	Description
0	No effect
1	Set the GTEN4 bit.

**[bit12] Reserved: Reserved bit**

**[bit11] TMIES2: Interrupt request enable set bit**

Value	Description
0	No effect
1	Set the TMIE2 bit.

**[bit10:8] Reserved: Reserved bits**

#### 4.11. 16-bit Dead Timer Reload Interrupt Set Register (WFGxx\_DTIRS)

16-bit dead timer reload interrupt set register (WGxx\_DTIRS) is used to set the bit of 16-bit dead timer reload interrupt register (WFGxx\_DTIR).

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved	DTRIES2	Reserved	DTRIES1	Reserved	DTRIES0	Reserved	
ACCESS_TYPE	R0,W0	R0,W	R0,W0	R0,W	R0,W0	R0,W	R0,W0	
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	00	

[bit23] Reserved: Reserved bit

[bit22] DTRIES2: 16-bit dead timer 2 reload interrupt enable set bit

Value	Description
0	No effect
1	Set the DTRIE2 bit.

[bit21] Reserved: Reserved bit

[bit20] DTRIES1: 16-bit dead timer 1 reload interrupt enable set bit

Value	Description
0	No effect
1	Set the DTRIE1 bit.

[bit19] Reserved: Reserved bit

[bit18] DTRIES0: 16-bit dead timer 0 reload interrupt enable set bit

Value	Description
0	No effect
1	Set the DTRIE0 bit.

[bit17:16] Reserved: Reserved bits



## 4.12. Waveform Control Set Register 1 (WFGxx\_SIGCR1S)

Waveform control set register 1 (WFGxx\_SIGCR1S) is used to set the bit of waveform control register 1 (WFGxx\_SIGCR1).

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	DTIES	Reserved	NRSLS	Reserved				
ACCESS_TYPE	R0,W	R0,W0	R0,W	R0,W0				
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	00000				

### [bit23] DTIES: DTTI input validating set bit

Value	Description
0	No effect
1	Set the DTIE bit.

### [bit22] Reserved: Reserved bit

### [bit21] NRSLS: Noise cancel function validating set bit

Value	Description
0	No effect
1	Set the NRSL bit.

### [bit20:16] Reserved: Reserved bits

### 4.13. DTTI Selection Register (WFG02\_DTSR, WFG03\_DTSR)

The DTTI selection register (DTSR) is used to select which of the DTTI input available in 2 inputs is assigned to each waveform generator.

#### (1) DTTI Selection Register (WFG02\_DTSR)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved						SEL1	SEL0
ACCESS_TYPE	R0,W0						R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	000000						1	0

[bit31:26] Reserved: Reserved bits

#### [bit25] SEL1: DTTI input selection bit for waveform generator 1

Value	Description
0	DTTI0
1	DTTI1

**Note:**

- Before configuring this bit, make sure to verify that the waveform generator is inactive.

#### [bit24] SEL0: DTTI input selection bit for waveform generator 0

Value	Description
0	DTTI0
1	DTTI1

**Note:**

- Before configuring this bit, make sure to verify that the waveform generator is inactive.





**(2) DTTI Selection Register (WFG03\_DTSTR)**

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved						SEL1	SEL0
ACCESS_TYPE	R0,W0						R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	000000						1	0

**[bit31:26] Reserved: Reserved bits**

**[bit25] SEL1: DTTI input selection bit for waveform generator 3**

Value	Description
0	DTTI2
1	DTTI3

**Note:**

- Before configuring this bit, make sure to verify that the waveform generator is inactive.

**[bit24] SEL0: DTTI input selection bit for waveform generator 2**

Value	Description
0	DTTI2
1	DTTI3

**Note:**

- Before configuring this bit, make sure to verify that the waveform generator is inactive.

## 4.14. Software DTTI Control Register (SDTCR2)

This register is used to generate DTTI input to waveform generator by software.

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved				DTIS3	DTIS2	DTIS1	DTIS0
ACCESS_TYPE	R0,WX				R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0000				1	1	1	1

**[bit7:4] Reserved: Reserved bits**

**[bit3:0] DTIS3 to DTIS0: Waveform generator control bits**

Value	Description
0	Set software DTTI. The output of the corresponding waveform generator is disabled.
1	Clear software DTTI. The output of the corresponding waveform generator is enabled.

- Combination of DTISn bit and waveform generator is selected by DTTI selection register (WFG02\_DTISR, WFG03\_DTISR).
- You can change this register dynamically.



#### 4.15. External DTTI Input Control Register (EDTCR2)

This register is used to control external DTTI pin input.

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved				DTIHE3	DTIHE2	DTIHE1	DTIHE0
ACCESS_TYPE	R0,W0				R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0000				0	0	0	0

**[bit7:4] Reserved: Reserved bits**

**[bit3:0] DTIHE3 to DTIHE0: External DTTI pin input control bits**

Value	Description
0	Disable external DTTI input
1	Enable external DTTI input

- Combination of DTIHE<sub>n</sub> bit and waveform generator is selected by DTTI selection register (WFG02\_DTISR, WFG03\_DTISR).
- You can change this register dynamically.

## 4.16. RTO Output Level Conversion Register (RTOSELn) (n=0 to 3)

This register is used to control output polarity of waveform generator.

### (1) RTOSEL3 Output Level Conversion Register 3 (RTOSEL3)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved		OSEL5	OSEL4	OSEL3	OSEL2	OSEL1	OSEL0
ACCESS_TYPE	R0,WX		R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	00		0	0	0	0	0	0

[bit31:30] Reserved: Reserved bits

[bit29:24] OSEL5 to OSEL0: Output polarity control bits (waveform generator unit3)

Value	Description
0	Normal output
1	Inverted output

You can change this register dynamically.

### (2) RTOSEL2 Output Level Conversion Register 2 (RTOSEL2)

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved		OSEL5	OSEL4	OSEL3	OSEL2	OSEL1	OSEL0
ACCESS_TYPE	R0,WX		R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	00		0	0	0	0	0	0

[bit23:22] Reserved: Reserved bits

[bit21:16] OSEL5 to OSEL0: Output polarity control bits (waveform generator unit2)

Value	Description
0	Normal output
1	Inverted output

You can change this register dynamically.



### (3) RTOSEL1 Output Level Conversion Register 1 (RTOSEL1)

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved		OSEL5	OSEL4	OSEL3	OSEL2	OSEL1	OSEL0
ACCESS_TYPE	R0,WX		R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	00		0	0	0	0	0	0

[bit15:14] Reserved: Reserved bits

[bit13:8] OSEL5 to OSEL0: Output polarity control bits (waveform generator unit1)

Value	Description
0	Normal output
1	Inverted output

You can change this register dynamically.

### (4) RTOSEL0 Output Level Conversion Register 0 (RTOSEL0)

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved		OSEL5	OSEL4	OSEL3	OSEL2	OSEL1	OSEL0
ACCESS_TYPE	R0,WX		R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	00		0	0	0	0	0	0

[bit7:6] Reserved: Reserved bits

[bit5:0] OSEL5 to OSEL0: Output polarity control bits (waveform generator unit0)

Value	Description
0	Normal output
1	Inverted output

You can change this register dynamically.

## 5. Precautions for Using

The following shows the notes when using the waveform generator.

### Notes When Accessing a Register

#### a) When Accessing to the 16-bit Dead Timer State Control Register n (DTCRn)

- This register supports writing from the bit-band alias area. For the bit-band alias area, see "CHAPTER: Bit-band Unit".
- To clear a specified bit of this register, clear the bit by writing "1" to the applicable bit of the 16-bit dead timer state control clear register n (DTCRCn). It is prohibited to directly clear only a specific bit in this register.
- To set a specified bit of this register, set the bit by writing "1" to the applicable bit of the 16-bit dead timer state control set register n (DTCRSn). It is prohibited to directly set only a specific bit in this register.
- Data can be written directly to this register only when writing to all bits.

#### b) When Accessing to the 16-bit Dead Timer Reload Interrupt Register (DTIR)

- This register supports writing from the bit-band alias area. For the bit-band alias area, see "CHAPTER: Bit-band Unit".
- To clear a specified bit of this register, clear the bit by writing "1" to the applicable bit of the 16-bit dead timer reload interrupt clear register (DTIRC). It is prohibited to directly clear only a specific bit in this register.
- To set a specified bit of this register, set the bit by writing "1" to the applicable bit of the 16-bit dead timer reload interrupt set register (DTIRS). It is prohibited to directly set only a specific bit in this register.
- Data can be written directly to this register only when writing to all bits.

#### c) When Accessing to the Waveform Control Register 1 (SIGCR1)

- This register supports writing from the bit-band alias area. For the bit-band alias area, see "CHAPTER: Bit-band Unit".
- To clear a specified bit of this register, clear the bit by writing "1" to the applicable bit of the waveform control clear register 1 (SIGCR1C). It is prohibited to directly clear only a specific bit in this register.
- To set a specified bit of this register, set the bit by writing "1" to the applicable bit of the waveform control set register 1 (SIGCR1S). It is prohibited to directly set only a specific bit in this register.
- Data can be written directly to this register only when writing to all bits.

### (2) Notes on Setting by a Program

- When you change the value of TMD8, TMD5, TMD2, TMD7, TMD4, TMD1, TMD6, TMD3, and TMD0 bits for the 16-bit dead timer state control register (DTCR[n]) while the waveform generator is active (TMD2 to TMD0/TMD5 to TMD3/TMD8 to TMD6 of the DTCR[n] register are "0b001", "0b010" or "0b100"), make sure that counting operation of the trigger source and the 16-bit dead timer is not in progress. Without following this procedure, an unexpected waveform may be output from the RTO pin due to the output scheduled in the previous trigger. In the case of RTO output, however, it will return to the normal operation when the timer underflows or it is triggered by a new trigger source again.
- When the TMD8 to TMD0 of the DTCR[n] register are "0b001", the trigger source is "H" level of RT". When the TMD8 to TMD0 bits are "0b010", it is the "rising edge of RT". When the TMD8 to TMD0 bits are "0b100", it is the "rising or falling edge of RT". For example, when the TMD8 to TMD0 bits change from "0b100" to "0b010", you will be able to execute the following procedure:
  1. Set the 16-bit dead timer register (TMRR[n]) to an extremely small value as "0x0001".
  2. Set the output of RTO1, RTO3, and RTO5 to "L" or "H" and wait until an underflow occurs at timer0, 1, and 2.
  3. Change the mode bits (TMD8 to TMD0) and the corresponding settings.



4. The modified output waveform will be appeared at RTO pin after 1 machine cycle.
- When a value is written to the 16-bit dead timer register (TMRR[n]) while the timer is counting, this new value will become effective in the next timer trigger. When accessing the timer register, use a half-word or word transfer instruction.
  - Change the DCK[2:0] of the waveform control register 1 (SIGCR1) only when the timer is not counting.
  - Change the NWS[1:0] of the waveform control register 1 (SIGCR1) only when the noise cancellation feature is disabled.



## CHAPTER 54: R/D Converter

This chapter describes the R/D converter.

---

### 1. Overview





## 1. Overview

This section provides an overview of the R/D converter.

The R/D converter A/D-converts an analog signal input to the microcomputer from an external resolver, and apply arithmetic operations to the value obtained through the conversion to calculate an angle and angular velocity.

Also, when a physical error, such as a short circuit or line break, occurs in the interface signal with the resolver, the R/D converter notifies of the error.

Moreover, it monitors arithmetic operation results during tracking loop processing (tracking interpolation arithmetic operation processing), and issues a warning when an error is detected.

### Features

- Operating mode setting function
- Individually obtaining an angle parameter and angular velocity parameter
- Tracking loop warning
- Detecting various errors
- Sine/cosine output function

For more information about R/D converter, contact a support representative.



## CHAPTER 55: D/A Converter

This chapter explains D/A converter.

---

1. Overview
2. Configuration
3. Operation
4. Setting Procedure Example
5. Registers
6. Precautions for Using



## 1. Overview

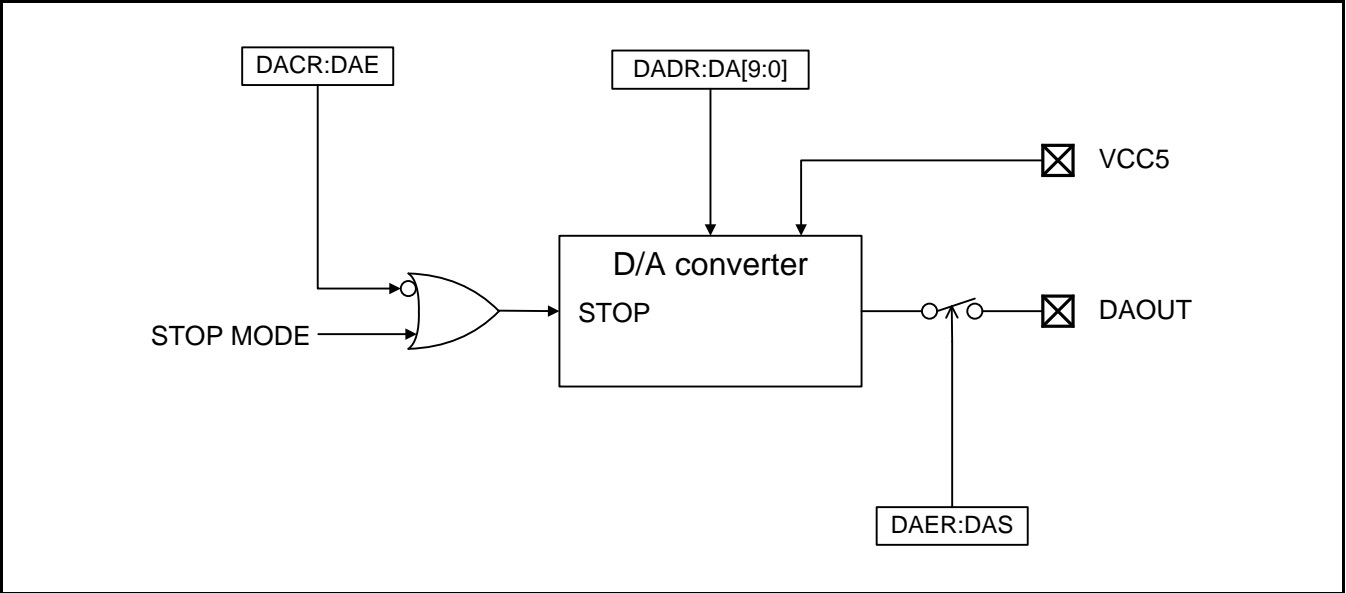
This section gives a brief overview of D/A converter.

The D/A converter is a peripheral function to convert digital signals to analog signals. The resolution of D/A converter is 10-bit.

2. Configuration

This section shows a block diagram of D/A converter.

Figure 2-1 Block Diagram of D/A Converter





### 3. Operation

This section describes the operation of D/A converter.

The D/A converter outputs analog voltage from the DAOUT pins by calculating the output voltage based on the values written in the D/A data register (DACxx\_DADR).

If values are written to the DA[9:0] bits of the D/A data register (DADR) and "1" is written to the DAE bit of the D/A control register (DACxx\_DACR), analog signals will be output from the D/A converter.

If "0" is written to the DAE bit of the D/A control register (DACxx\_DACR), the output pin DAOUT from the D/A converter becomes high impedance state. In addition, even if "1" is written to the DAE bit, the output pin DAOUT from the D/A converter also becomes high impedance state when this LSI is in stop mode.

The voltage, which can be output when the output from the D/A converter is enabled by writing "1" to the DAE bit of the D/A control register (DACxx\_DACR), is from 0.0V to  $1023/1024 \times VCC5V$ .

## 4. Setting Procedure Example

Set the D/A converter with following procedures and methods.

### (1) Output from DAOUT Pins

Set it according to the procedure from 1 to 4.

1. Set pin to port function side. (DACxx\_DAER:DAS="0")
2. Set the output voltage of the D/A converter. (DACxx\_DADR:DA[9:0])
3. Enable the output of the D/A converter. (DACxx\_DACR:DAE="1")
4. Set pin to D/A output side. (DACxx\_DAER:DAS="1")

### (2) Disabling the Output from the DAOUT Pins

Set the following either.

- Set the D/A output pin to port function side. (DACxx\_DAER:DAS="0")
- Set the D/A converter output disabled. (DACxx\_DACR:DAE="0")



## 5. Registers

This section describes the registers of D/A converter.

**Table 5-1 List of D/A Converter Registers**

Abbreviated Register Name	Register Name	See
DACxx_DACR	D/A Control Register	5.1
DACxx_DADR	D/A Data Register	5.2
DACxx_DAER	Analog Output Control Register	5.3
DACxx_KEYCDR	Key Code Register	5.4

xx: channel number (xx=00, 01)

## 5.1. D/A Control Register (DACxx\_DACR)

This register enables the output from the DAOUT pins.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							DAE
ACCESS_TYPE	R0,WX							R/W
PROT_TYPE	-							
INITIAL_VALUE	0000000							0

[bit31:25] Reserved: Reserved bits

[bit24] DAE: D/A output enable bit

Value	Description
0	Output disabled. DAOUT pins become high impedance state by writing to "0".
1	Output enabled. The voltage corresponding to the set value in DACxx_DADR:DA[9:0] is output.





## 5.2. D/A Data Register (DACxx\_DADR)

This register is used to set the output voltage from the DAOUT pins.

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved						DA[9:8]	
ACCESS_TYPE	R0,WX						R/W	
PROT_TYPE	-							
INITIAL_VALUE	000000						XX	

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	DA[7:0]							
ACCESS_TYPE	R/W							
PROT_TYPE	-							
INITIAL_VALUE	XXXXXXXX							

**[bit15:10] Reserved: Reserved bits**

**[bit9:0] DA[9:0]: D/A output value**

These bits set the voltage output to the DAOUT pins by an unsigned integer with 10-bit. The relation between the values set to DA[9:0] and the output voltage is shown in the following.

Value	Output voltage (theoretical values)
00_0000_0000	$0 / 1024 \times VCC5$
00_0000_0001	$1 / 1024 \times VCC5$
00_0000_0010	$2 / 1024 \times VCC5$
:	:
11_1111_1101	$1021 / 1024 \times VCC5$
11_1111_1110	$1022 / 1024 \times VCC5$
11_1111_1111	$1023 / 1024 \times VCC5$

### 5.3. Analog Output Control Register (DACxx\_DAER)

This register selects function of DAOUT pins.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							DAS
ACCESS_TYPE	R0,WX							R/W
PROT_TYPE	-							
INITIAL_VALUE	0000000							0

**[bit31:25] Reserved: Reserved bits**

**[bit24] DAS: Analog output control bit**

Value	Description
0	Function of DAOUT pin is port (digital).
1	Function of DAOUT pin is analog output.

**Note:**

- In part number equipped with a key code function, a key code setting is required for writing to this register. For details on which part number have the key code function, see "Part Number Option" in the Data Sheet.



## 5.4. Key Code Register (DACxx\_KEYCDR)

This register sets register writing with a function for protection against erroneous writing. If writing to this register is not done using the prescribed method, writing to the relevant register is invalid. This register is valid only for word access. In the part number unequipped with the key code function, access to this register has no effect on the operation.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	KEY		SIZE		Reserved			
ACCESS_TYPE	R0,W		R0,W		R0,WX			
PROT_TYPE	-							
INITIAL_VALUE	00		00		0000			

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved				ADR[11:8]			
ACCESS_TYPE	R0,WX				R0,W			
PROT_TYPE	-							
INITIAL_VALUE	0000				0000			

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	ADR[7:0]							
ACCESS_TYPE	R0,W							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

### [bit31:30] KEY[1:0]: Key Code bits

These are key code setting bits. Write "0b00", "0b01", "0b10", and "0b11" continuously to these bits in this order. The key code setting becomes invalid immediately upon any writing to these bits in a different order. In such cases, set the key code again from the beginning.

Value	Description
00	1st key code
01	2nd key code
10	3rd key code
11	4th key code

**[bit29:28] SIZE[1:0]: Access size bits**

These bits set the access size for writing to the applicable key code register. Write the same data to these bits when writing key codes "0b00", "0b01", "0b10", and "0b11" in this order.

Value	Description
00	Set byte access.
01	Set half-word access.
10	Set word access.
11	Reserved

**[bit27:12] Reserved: Reserved bits**

**[bit11:0] ADR[11:0]: Address bits**

These bits set the lower 12 bits of the address of the applicable key code register. Write the same data to these bits when writing key codes "0b00", "0b01", "0b10", and "0b11" in this order.

Value	Description
	Set the lower 12 bits of the address of the applicable key code register.

**Notes:**

- The key code setting becomes invalid upon any writing to the KEY[1:0] in any order than "00", "01", "10", and "11". In such case, set the key code again from the beginning.
- When different data is written to the SIZE[1:0] or the ADR[11:0] while writing the key code "00", "01", "10", and "11", the key code setting will become invalid. In such case, set it again from the beginning.
- This register is valid only for word access.
- There is no effect on the key code protection release process even if registers other than this register and the applicable key code register that is in the release processing (the register set by the ADR) are accessed while writing the key code "00", "01", "10", and "11".



## 6. Precautions for Using

This section explains precautions for using the D/A converter.

### **Wait Time when Switching Port Function**

If the pin function is GPIO (analog output is disabled), the internal amplifier is powered down. In this state, if the DA output is enabled by controlling the DAS bit of analog output control register (DACxx\_DAER), note that a certain timeframe (10 $\mu$ s) is required for power-on of the amplifier and for output stabilization.

# CHAPTER 56: 12-bit 4ch A/D Converter Interface

This chapter provides an overview and explains the register configuration and operation of the 12-bit 4ch A/D converter interface.



1. Overview
2. Configuration
3. Operation
4. Setting Procedure Example
5. Registers
6. Precautions for Using



## 1. Overview

The 12-bit 4ch A/D converter has functions for simultaneous sampling of 4ch analog input voltage and serial conversion into 12-bit serial digital values through the RC successive approximation conversion method. A/D conversion is performed from A/D activation trigger input. If an A/D activation trigger is input again during the A/D conversion, the A/D conversion is restarted. The converter also supports a forced stop function using A/D conversion cancel input signals.

### Function of 12-bit 4ch A/D Converter Interface

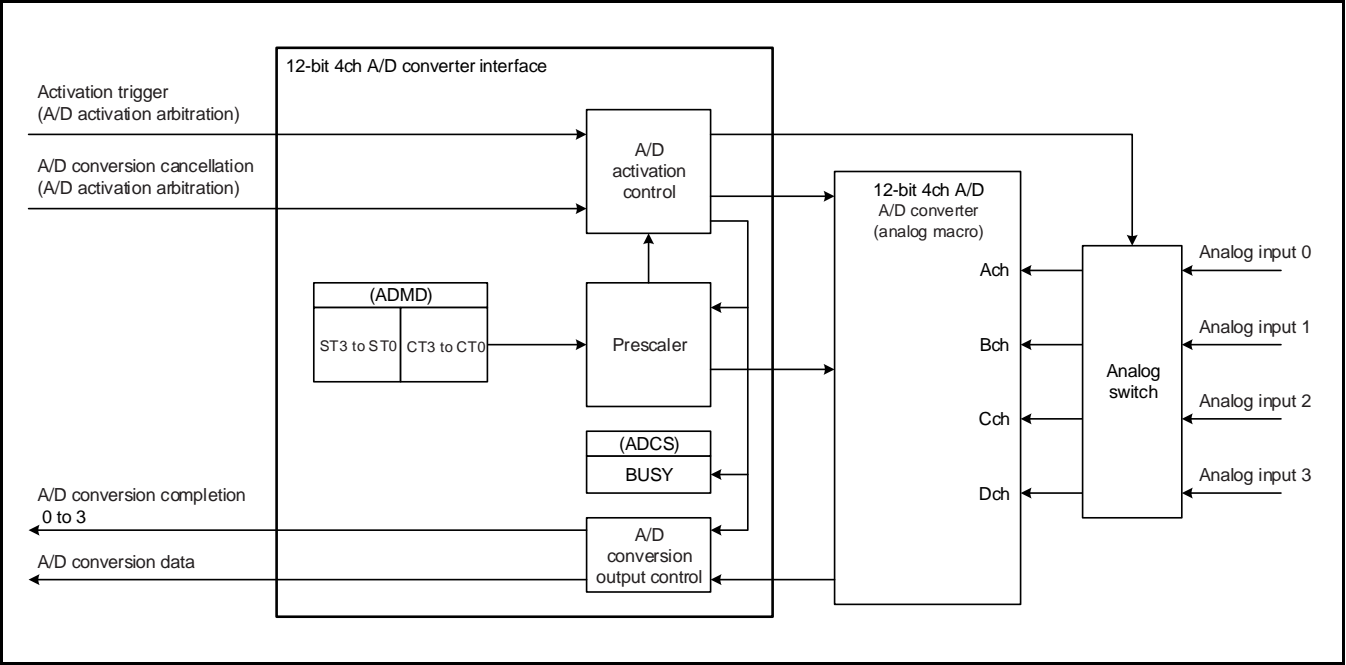
The function of the converter is to convert the analog voltage (input voltage) input through analog input pins into digital values. It has the following features.

- The conversion method is the RC successive approximation method with a sample hold circuit.
- A program can select up to 4 channels for the analog input pins. (This is configured from the A/D interface.)
- Activation signals are input with pulse signals.
- The A/D conversion function converts up to 4 channels at 1 time from the input of a 1-time activation factor.
- If an activation signal is input again during A/D conversion, the conversion is restarted. (Restart function)
- During A/D conversion, the current processing is stopped/initialized when an A/D conversion cancel signal is received. (Forced stop function)

2. Configuration

This section shows a block diagram of the 12-bit 4ch A/D converter interface.

Figure 2-1 12-bit 4ch A/D Converter Interface Configuration







### **3. Operation**

This section explains the operation of the 12-bit 4ch A/D converter interface.

#### **3.1. 12-bit 4ch A/D Converter Interface Operation**

The 12-bit 4ch A/D converter interface controls A/D conversion.



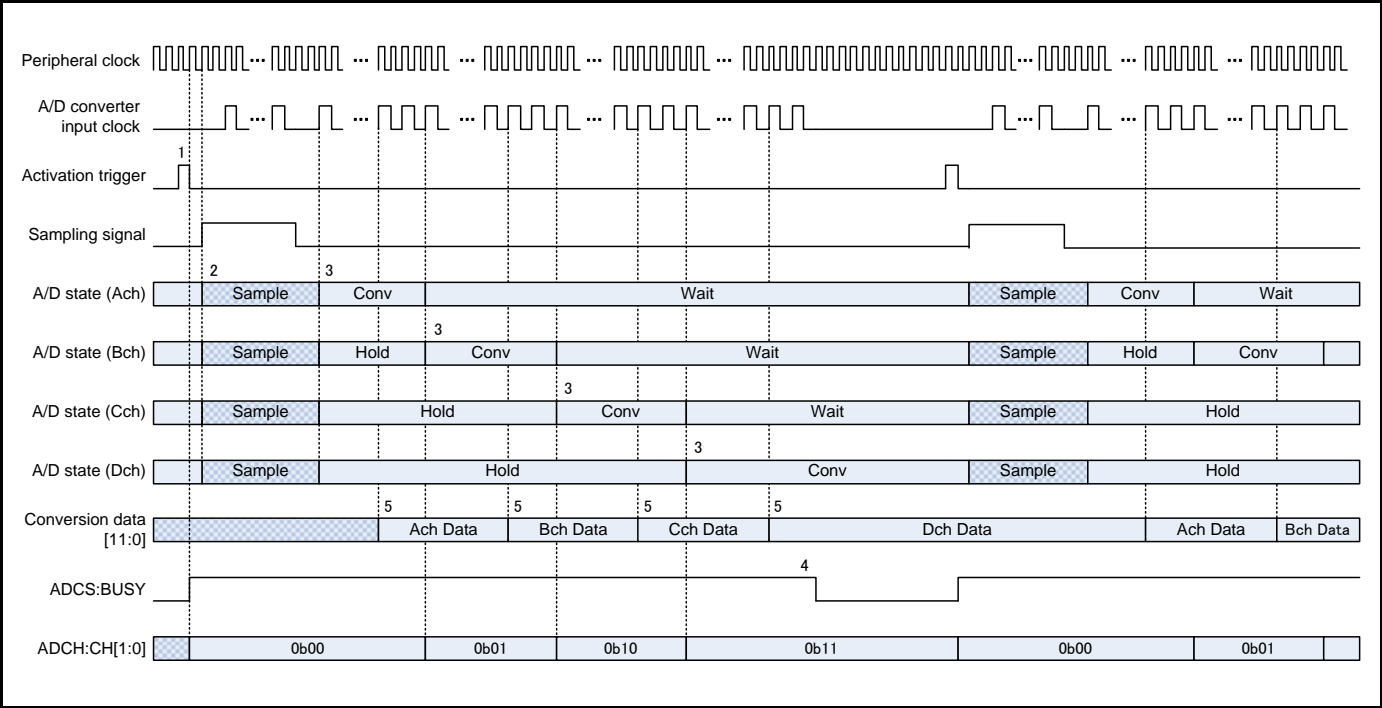
3.1.1.     **Activation Factor**

The activation factors for A/D conversion are activation trigger signals input with pulse signals.

3.1.2.     **A/D Conversion**

The A/D conversion function performs conversion 1 time from the input of a 1-time activation trigger.

**Figure 3-1 Operation Timing of 12-bit 4ch A/D Converter Interface (During 4ch Operation)**



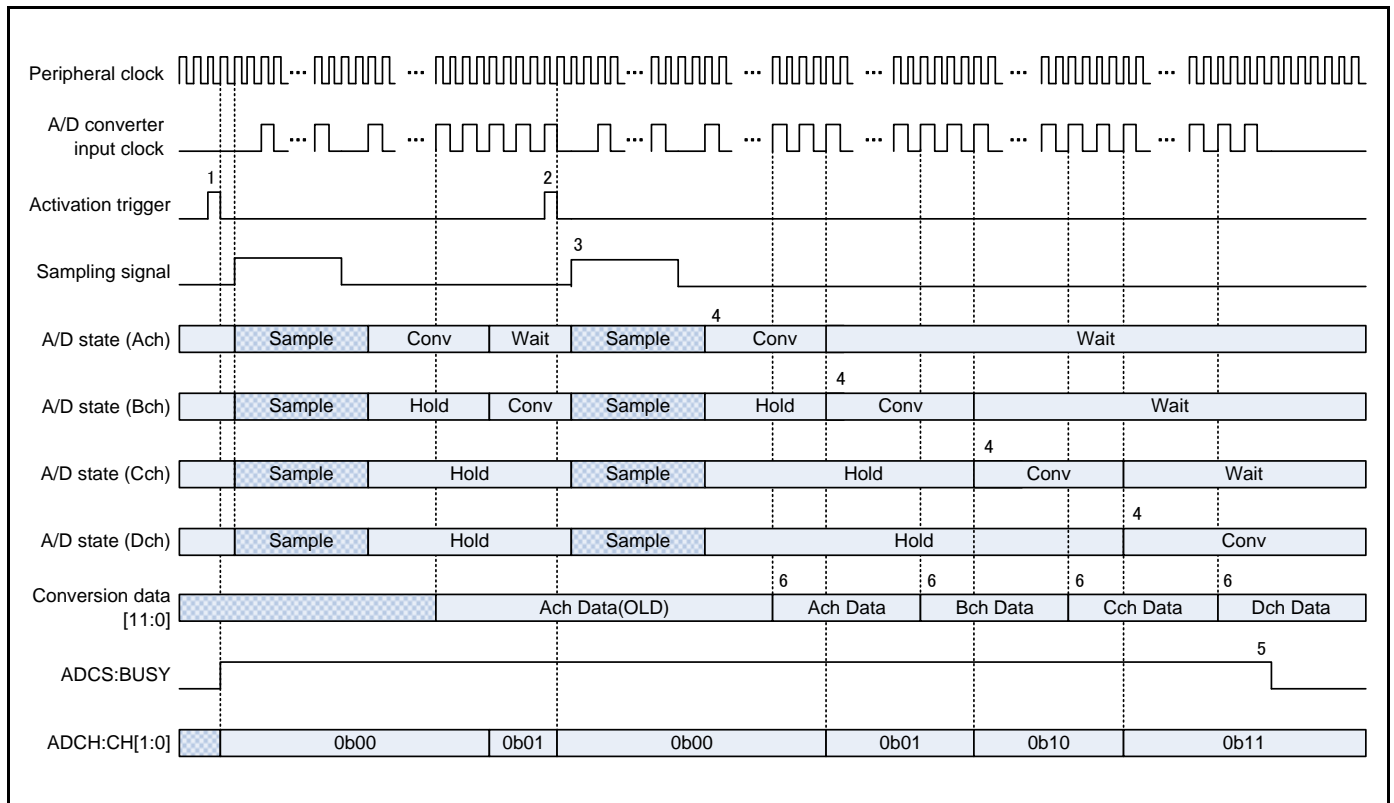
1. The input of an A/D activation trigger signal starts A/D conversion.
2. The reception of the A/D activation trigger signal at 1 starts the simultaneous sampling of 4 channels.
3. After the sampling time elapses, a compare operation begins with Ach, followed by serial compare operations until Dch.
4. After the compare time of each channel elapses, a conversion end signal for the channel is output. Once conversion in all 4 channels is complete, A/D conversion ends.
5. A/D conversion data is output serially as conversion in each channel ends.
6. New conversion data is stored in an A/D data register (ADTCD) for A/D activation compare.



### 3.1.3. Restart

If an activation trigger signal is input during A/D conversion, the current conversion is stopped/initialized, and the A/D conversion is restarted. Accordingly, the restart of A/D conversion begins several clocks (12-bit 4ch A/D converter clocks) later than in normal activation (A/D conversion beginning while A/D conversion is stopped).

Figure 3-2 Restart Operation Timing of 12-bit 4ch A/D Converter Interface



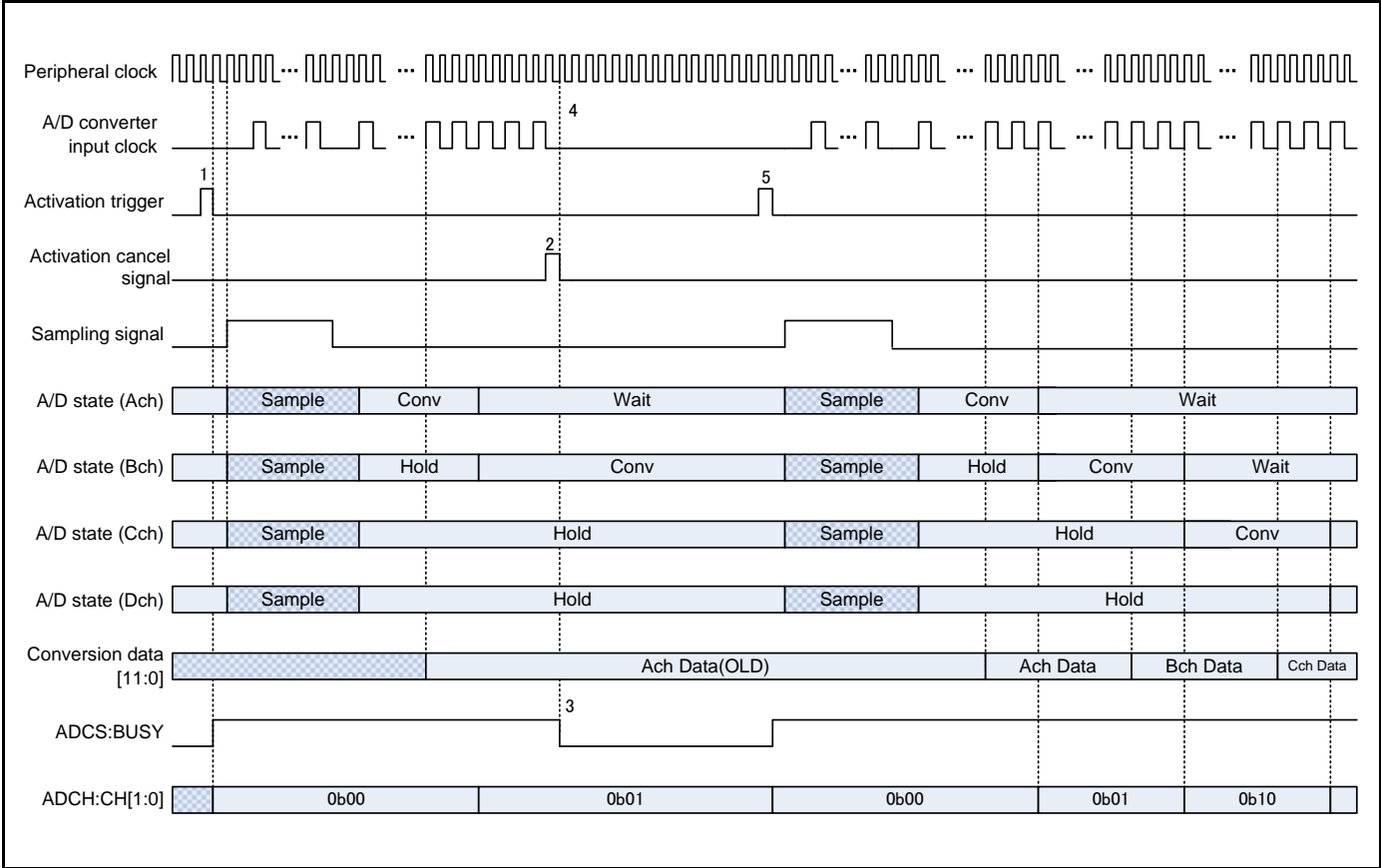
1. The input of an A/D activation trigger pulse signal starts A/D conversion.
2. An activation trigger signal is input during A/D conversion, so the A/D conversion is restarted.
3. The A/D activation restart at 2 stops/initializes the A/D conversion that has been converting since 1.
4. After the sampling time elapses, a compare operation begins with Ach, followed by serial A/D conversion until Dch.
5. After the compare time of each channel elapses, a conversion end signal is output. Conversion ends when conversion end signals for all configured channels have been output.
6. A/D conversion data is output.
7. New conversion data is stored in an A/D data register (ADTCD) for A/D activation compare.



3.1.4.     **A/D Conversion Cancel**

During A/D conversion, the current conversion is stopped/initialized when an A/D conversion cancel signal is received.

**Figure 3-3 Conversion Cancel Operation Timing of 12-bit 4ch A/D Converter Interface**



1. The input of an A/D activation trigger pulse signal starts A/D conversion.
2. The signals within the A/D conversion include an input conversion cancel signal, which cancels the conversion.
3. The A/D conversion cancellation at 2 stops/initializes the A/D conversion that has been converting since 1.
4. The A/D conversion cancellation at 2 stops the A/D converter input clock.
5. The input of an A/D activation trigger pulse signal once again starts A/D conversion.



### 3.1.5. A/D Conversion Time

The A/D conversion time is a combination of the sampling time and compare time.

#### (1) Sampling Time

The sampling time is common to all channels. The sampling time setting bits (ADMD:ST[3:0]) set the sampling time.

**Table 3-1 Sampling Time by Peripheral Clock Frequency**

ST[3:0]	Function	A/D Clock	Sampling Time (Peripheral Clock Frequency)					
			100MHz	80MHz	64MHz	40MHz	32MHz	20MHz
0000	12 peripheral clock cycles	Peripheral clock/1	120ns	150ns	188ns	300ns	375ns	600ns
0001	24 peripheral clock cycles	Peripheral clock/2	240ns	300ns	375ns	600ns	750ns	1200ns
0010	36 peripheral clock cycles	Peripheral clock/3	360ns	450ns	563ns	900ns	1125ns	1800ns
0011	48 peripheral clock cycles	Peripheral clock/4	480ns	600ns	750ns	1200ns	1500ns	2400ns
0100	60 peripheral clock cycles	Peripheral clock/5	600ns	750ns	938ns	1500ns	1875ns	3000ns
0101	72 peripheral clock cycles	Peripheral clock/6	720ns	900ns	1125ns	1800ns	2250ns	3600ns
0110	84 peripheral clock cycles	Peripheral clock/7	840ns	1050ns	1313ns	2100ns	2625ns	4200ns
0111	96 peripheral clock cycles	Peripheral clock/8	960ns	1200ns	1500ns	2400ns	3000ns	4800ns
1000	108 peripheral clock cycles	Peripheral clock/9	1080ns	1350ns	1688ns	2700ns	3375ns	5400ns
1001	120 peripheral clock cycles	Peripheral clock/10	1200ns	1500ns	1875ns	3000ns	3750ns	6000ns
1010	132 peripheral clock cycles	Peripheral clock/11	1320ns	1650ns	2063ns	3300ns	4125ns	6600ns
1011	144 peripheral clock cycles	Peripheral clock/12	1440ns	1800ns	2250ns	3600ns	4500ns	7200ns
1100	156 peripheral clock cycles	Peripheral clock/13	1560ns	1950ns	2438ns	3900ns	4875ns	7800ns
1101	168 peripheral clock cycles	Peripheral clock/14	1680ns	2100ns	2625ns	4200ns	5250ns	8400ns
1110	180 peripheral clock cycles	Peripheral clock/15	1800ns	2250ns	2813ns	4500ns	5625ns	9000ns
1111	192 peripheral clock cycles	Peripheral clock/16	1920ns	2400ns	3000ns	4800ns	6000ns	9600ns

- The settings in the shaded area are prohibited because they are outside the recommended values for the 12-bit 4ch A/D converter.

#### Notes:

- Set a sampling time that is within the recommended values for the A/D converter.
- A normal analog conversion value may not be obtained outside the recommended values.
- For details on the recommended values, see the data sheet.
- "0b0000" in ST[3:0] is a prohibited setting. During a sampling period with this setting, no clock is supplied to the 12-bit 4ch A/D converter.

## (2) Compare time

The compare time setting bits (ADMD:CT[3:0]) set the compare time.

**Table 3-2 Compare Time by Peripheral Clock Frequency (4ch Total)**

CT[3:0]	Function	A/D Clock	Compare Time (Peripheral Clock Frequency)					
			100MHz	80MHz	64MHz	40MHz	32MHz	20MHz
0000	56 peripheral clock cycles	Peripheral clock/1	560ns	700ns	875ns	1400ns	1750ns	2800ns
0001	112 peripheral clock cycles	Peripheral clock/2	1120ns	1400ns	1750ns	2800ns	3500ns	5600ns
0010	168 peripheral clock cycles	Peripheral clock/3	1680ns	2100ns	2625ns	4200ns	5250ns	8400ns
0011	224 peripheral clock cycles	Peripheral clock/4	2240ns	2800ns	3500ns	5600ns	7000ns	11200ns
0100	280 peripheral clock cycles	Peripheral clock/5	2800ns	3500ns	4375ns	7000ns	8750ns	14000ns
0101	336 peripheral clock cycles	Peripheral clock/6	3360ns	4200ns	5250ns	8400ns	10500ns	16800ns
0110	392 peripheral clock cycles	Peripheral clock/7	3920ns	4900ns	6125ns	9800ns	12250ns	19600ns
0111	448 peripheral clock cycles	Peripheral clock/8	4480ns	5600ns	7000ns	11200ns	14000ns	22400ns
1000	504 peripheral clock cycles	Peripheral clock/9	5040ns	6300ns	7875ns	12600ns	15750ns	25200ns
1001	560 peripheral clock cycles	Peripheral clock/10	5600ns	7000ns	8750ns	14000ns	17500ns	28000ns
1010	616 peripheral clock cycles	Peripheral clock/11	6160ns	7700ns	9625ns	15400ns	19250ns	30800ns
1011	672 peripheral clock cycles	Peripheral clock/12	6720ns	8400ns	10500ns	16800ns	21000ns	33600ns
1100	728 peripheral clock cycles	Peripheral clock/13	7280ns	9100ns	11375ns	18200ns	22750ns	36400ns
1101	784 peripheral clock cycles	Peripheral clock/14	7840ns	9800ns	12250ns	19600ns	24500ns	39200ns
1110	840 peripheral clock cycles	Peripheral clock/15	8400ns	10500ns	13125ns	21000ns	26250ns	42000ns
1111	896 peripheral clock cycles	Peripheral clock/16	8960ns	11200ns	14000ns	22400ns	28000ns	44800ns

- The settings in the shaded area are prohibited because they are outside the recommended values for the 12-bit 4ch A/D converter.
- Number of compare-time peripheral clock cycles = 14 x number of conversion channels x division ratio of A/D clock

Example) Suppose CT[3:0] is "0b0011" and the number of conversion channels is 3.

Number of compare-time peripheral clock cycles = 14 x 3 x 4 = 168 (peripheral clock cycles)

### Notes:

- Set a compare time that is within the recommended values for the A/D converter.
- A normal analog conversion value may not be obtained outside the recommended values.
- For details on the recommended values, see the data sheet.
- "0b0000" in CT[3:0] is a prohibited setting. During a compare period with this setting, no clock is supplied to the 12-bit 4ch A/D converter.



### 3.1.6. A/D Conversion End and A/D Data Fetch

Once A/D conversion ends normally without being restarted or canceled (the predetermined number of cycles has elapsed), the received conversion data is fetched and output. At this time, an A/D conversion end signal is generated.

### 3.1.7. Conversion Ch Selection Mode

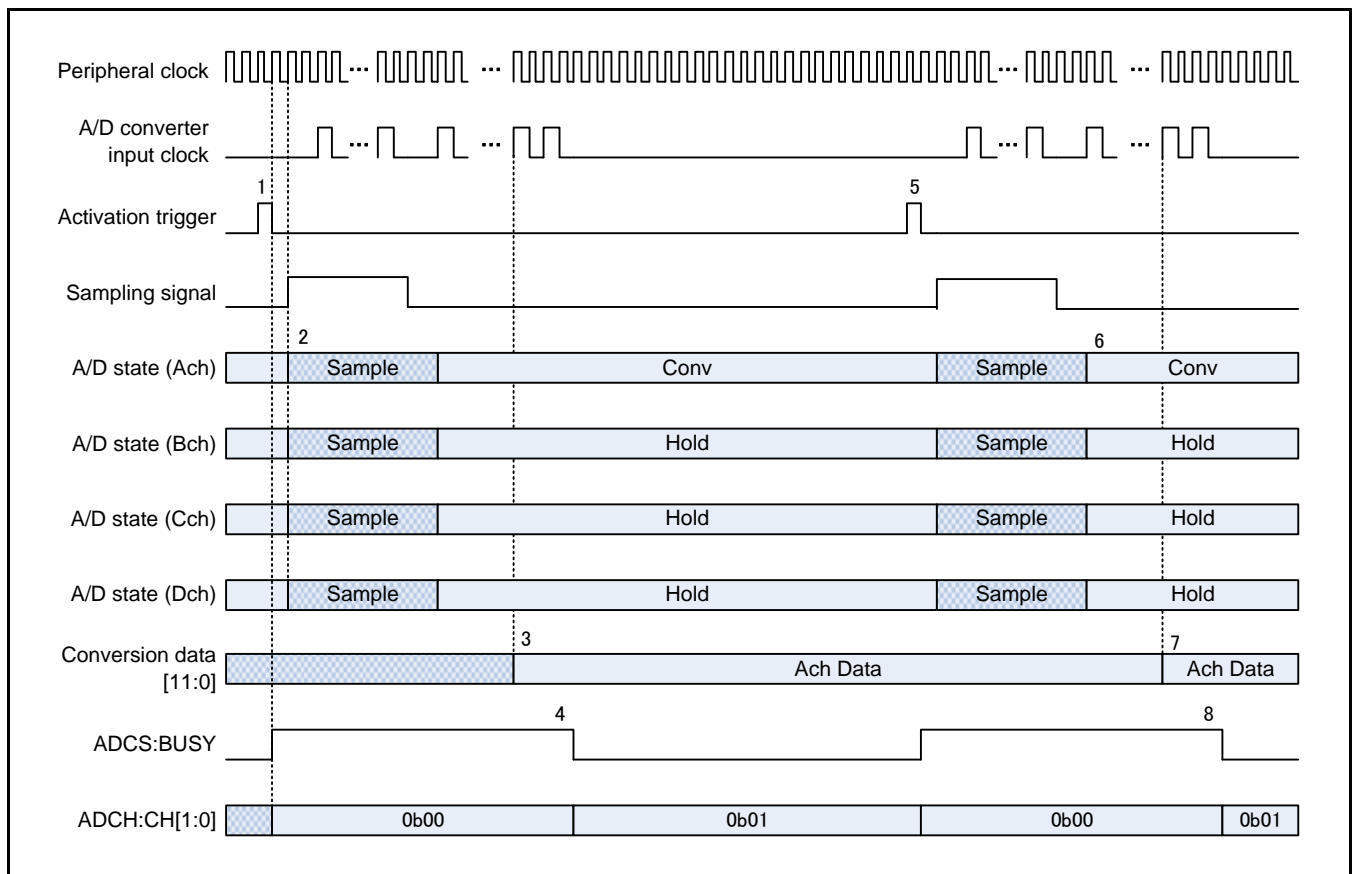
As the number of channels used for conversion, 1 to 4 channels can be selected. The A/D conversion ch selection bits (ADCHSEL:ADCSL[1:0]) set the number of conversion channels.

If 1 channel is selected, conversion is performed in only Ach.

If 2 channels are selected, conversion is performed in Ach and Bch, in this order.

If 3 channels are selected, conversion is performed in Ach, Bch, and Cch, in this order.

Figure 3-4 Operation Timing of 12-bit 4ch A/D Converter Interface (During 1ch Operation)



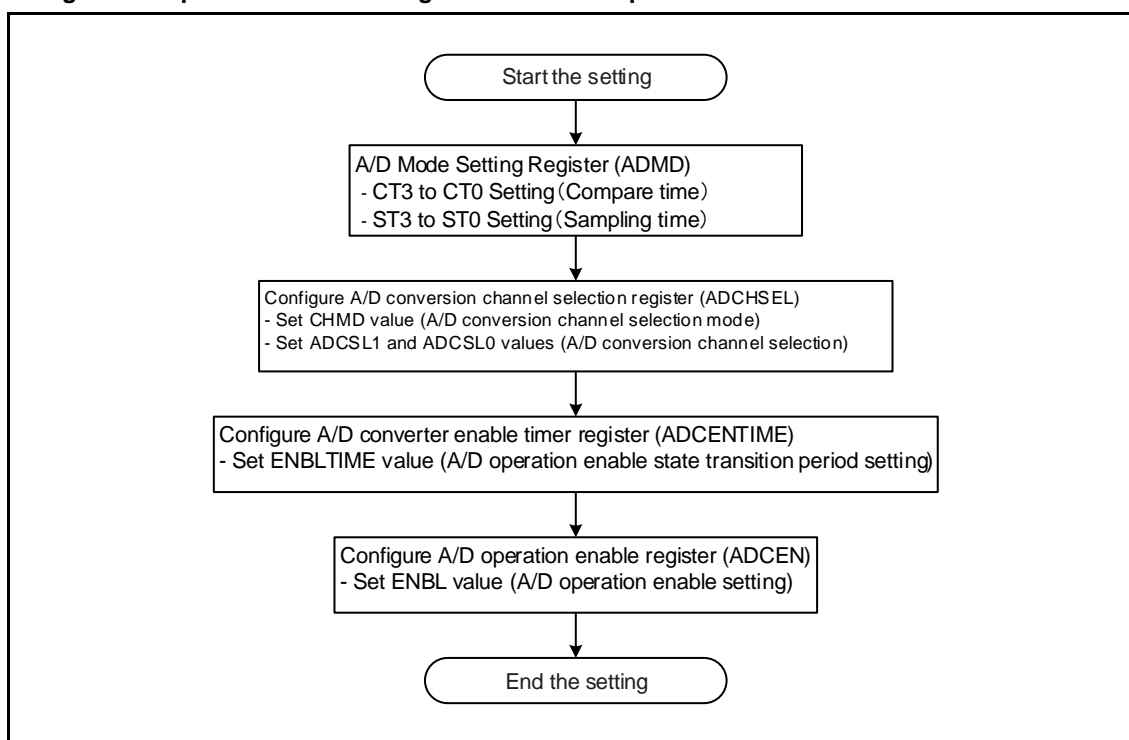
1. The input of an A/D activation trigger pulse signal starts A/D conversion.
2. A/D conversion is performed in Ach.
3. With the completion of the A/D conversion at 2, the A/D conversion data is output.
4. At this time, ADCS:BUSY becomes "L", and the A/D conversion ends without conversion in Bch to Dch.
5. The input of a new A/D activation trigger pulse signal starts A/D conversion.
6. Similar to 2, A/D conversion is performed in Ach.
7. After the A/D conversion is complete, the A/D conversion data is output.
8. The ADCS:BUSY signal becomes "L", and the A/D conversion ends without conversion in Bch to Dch.

## 4. Setting Procedure Example

This section shows an example of the 12-bit 4ch A/D converter interface setting procedure.

### 12-bit 4ch A/D Converter Interface Operation Mode Settings

Figure 4-1 Operation Mode Setting Procedure Example for 12-bit 4ch A/D Converter Interface



**Note:**

- Be sure that the A/D mode setting register (ADMD), A/D conversion ch selection register (ADCHSEL), A/D converter enable timer register (ADCENTIME), and A/D operation enable setting register (ADCEN) bits are configured before the conversion operation while A/D operations are stopped (ADCS:BUSY is "0").





## 5. Registers

This section lists the 12-bit 4ch A/D converter interface registers.

With the exception of the 4chADC analog input enable register, the 12-bit 4ch A/D converter interface register names begin with the prefix "ADC4SHxx\_". xx corresponds to the unit number.

**Table 5-1 List of 12-bit 4ch A/D Converter Interface Registers**

Abbreviated Register Name	Register Name	See
ADC4SHxx_ADCS	A/D Control Status Register	5.1.1
ADC4SHxx_ADCH	A/D Channel Status Register	5.1.2
ADC4SHxx_ADMD	A/D Mode Setting Register	5.1.3
ADC4SHxx_ADCHSEL	A/D Conversion Ch Selection Register	5.1.4
ADC4SHxx_ADCEN	A/D Operation Enable Setting Register	5.1.5
ADC4SHxx_ADCENTIME	A/D Converter Enable timer Register	5.1.6
ADER4CH_1/ADER4CH_0	4ch ADC Analog Input Enable Register	5.1.7

xx: Unit number (xx=00, 01)

## 5.1. 12-bit 4ch A/D Converter Interface Registers

The 12-bit 4ch A/D converter interface has the following registers: A/D control status register, A/D channel status register, A/D mode setting register, A/D conversion ch selection register, and A/D operation enable setting register.

### 5.1.1. A/D Control Status Register (ADCS)

The A/D control status register (ADCS) indicates that A/D conversion is in operation or has stopped.

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	BUSY	READY	Reserved					
ACCESS_TYPE	R,WX	R,WX	R0,W0					
PROT_TYPE	-							
INITIAL_VALUE	0	0	000000					

#### [bit7] BUSY: A/D conversion busy bit

Value	Description
0	A/D conversion has stopped.
1	A/D conversion is in operation (from activation until the end of conversion for the set number of channels).

- This bit indicates the operation of the A/D converter.
- As the value read from the A/D conversion busy bit (BUSY), "0" indicates that A/D conversion has stopped, and "1" indicates that A/D conversion is in operation.
- Writing to this bit does not change it and has no other effect.

#### [bit6] READY: Operation enable status bit

Value	Description
0	Operation stopped
1	Operation enabled

- This bit indicates whether 12-bit 4ch A/D converter operation is enabled.
- A/D conversion is performed only while operation is enabled.
- If A/D conversion is requested when operation is disabled, A/D conversion was executed and the result of A/D conversion was not guaranteed.
- If operation is disabled during A/D conversion, A/D conversion was executed without interruption and the result of A/D conversion was not guaranteed.

#### [bit5:0] Reserved: Reserved bits



### 5.1.2. A/D Channel Status Register (ADCH)

The A/D channel status register (ADCH) enables, during A/D conversion, confirmation of the analog channel number for the conversion in progress.

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved						CH	
ACCESS_TYPE	R0,W0						R,WX	
PROT_TYPE	-							
INITIAL_VALUE	000000						00	

**[bit7:2] Reserved: Reserved bits**

**[bit1:0] CH[1:0]: Analog channel bits**

Value	Description
00	Channel 0 (Ach) This is from activation start until the end of conversion in channel 0 (Ach).
01	Channel 1 (Bch) This is from the end of conversion in channel 0 (Ach) until the end of conversion in channel 1 (Bch).
10	Channel 2 (Cch) This is from the end of conversion in channel 1 (Bch) until the end of conversion in channel 2 (Cch).
11	Channel 3 (Dch) This is from the end of conversion in channel 2 (Cch) until the end of conversion in channel 3 (Dch).

- These bits can be used, during A/D conversion, to confirm the analog channel number for the conversion in progress.
- The value is enabled when ADCS.BUSY is "1". It holds no meaning when ADCS.BUSY is "0".

### 5.1.3. A/D Mode Setting Register (ADMD)

The function of the A/D mode setting register (ADMD) is to set the A/D conversion compare time and sampling time.

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved				CT			
ACCESS_TYPE	R0,W0				R/W			
PROT_TYPE	-							
INITIAL_VALUE	0000				0001			

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved				ST			
ACCESS_TYPE	R0,W0				R/W			
PROT_TYPE	-							
INITIAL_VALUE	0000				0001			

[bit15:12] Reserved: Reserved bits

[bit11:8] CT[3:0]: Compare time setting bits

Value	Description
0000	14 peripheral clock cycles (setting prohibited) (A/D clock output: Peripheral clock/1)
0001	28 peripheral clock cycles (A/D clock output: Peripheral clock/2)
0010	42 peripheral clock cycles (A/D clock output: Peripheral clock/3)
0011	56 peripheral clock cycles (A/D clock output: Peripheral clock/4)
0100	70 peripheral clock cycles (A/D clock output: Peripheral clock/5)
0101	84 peripheral clock cycles (A/D clock output: Peripheral clock/6)
0110	98 peripheral clock cycles (A/D clock output: Peripheral clock/7)
0111	112 peripheral clock cycles (A/D clock output: Peripheral clock/8)
1000	126 peripheral clock cycles (A/D clock output: Peripheral clock/9)
1001	140 peripheral clock cycles (A/D clock output: Peripheral clock/10)
1010	154 peripheral clock cycles (A/D clock output: Peripheral clock/11)
1011	168 peripheral clock cycles (A/D clock output: Peripheral clock/12)
1100	182 peripheral clock cycles (A/D clock output: Peripheral clock/13)
1101	196 peripheral clock cycles (A/D clock output: Peripheral clock/14)
1110	210 peripheral clock cycles (A/D clock output: Peripheral clock/15)
1111	224 peripheral clock cycles (A/D clock output: Peripheral clock/16)

- These bits select the compare time at the A/D conversion time.
- After analog input is fetched (the sampling time has elapsed), and after the set time in the compare time setting bits (CT[3:0]) has passed, the conversion result data is determined.
- The table shows the number of cycles required for the compare time per channel. To use 4 channels, 4 times as many cycles are required.

**Notes:**

- Set a compare time that is within the recommended values for the 12-bit 4ch A/D converter.
- A normal analog conversion value may not be obtained outside the recommended values.
- For details on the recommended values, see the data sheet.
- Be sure that the compare time setting bits (CT[3:0]) are rewritten before the conversion operation while A/D operations are stopped (ADCS:BUSY is "0").



- The setting of "0b0000" in the CT[3:0] bits in the A/D mode setting register (ADMD) is prohibited. During a 12-bit 4ch A/D converter compare period with this setting, no clock is supplied to the 12-bit 4ch A/D converter.

#### [bit7:4] Reserved: Reserved bits

#### [bit3:0] ST[3:0]: Sampling time setting bits

Value	Description
0000	12 peripheral clock cycles (setting prohibited) (A/D clock output: Peripheral clock/1)
0001	24 peripheral clock cycles (A/D clock output: Peripheral clock/2)
0010	36 peripheral clock cycles (A/D clock output: Peripheral clock/3)
0011	48 peripheral clock cycles (A/D clock output: Peripheral clock/4)
0100	60 peripheral clock cycles (A/D clock output: Peripheral clock/5)
0101	72 peripheral clock cycles (A/D clock output: Peripheral clock/6)
0110	84 peripheral clock cycles (A/D clock output: Peripheral clock/7)
0111	96 peripheral clock cycles (A/D clock output: Peripheral clock/8)
1000	108 peripheral clock cycles (A/D clock output: Peripheral clock/9)
1001	120 peripheral clock cycles (A/D clock output: Peripheral clock/10)
1010	132 peripheral clock cycles (A/D clock output: Peripheral clock/11)
1011	144 peripheral clock cycles (A/D clock output: Peripheral clock/12)
1100	156 peripheral clock cycles (A/D clock output: Peripheral clock/13)
1101	168 peripheral clock cycles (A/D clock output: Peripheral clock/14)
1110	180 peripheral clock cycles (A/D clock output: Peripheral clock/15)
1111	192 peripheral clock cycles (A/D clock output: Peripheral clock/16)

- These bits select the sampling time at the A/D conversion time.
- Once A/D is activated, analog input is fetched at the set time of the sampling time setting bits (ST[3:0]).

#### Notes:

- Set a sampling time that is within the recommended values for the 12-bit 4ch A/D converter.
- A normal analog conversion value may not be obtained outside the recommended values.
- For details on the recommended values, see the data sheet.
- Be sure that the sampling time setting bits (ST[3:0]) are rewritten before the conversion operation while A/D operations are stopped (ADCS:BUSY is "0").
- The setting of "0b0000" in the ST[3:0] bits in the A/D mode setting register (ADMD) is prohibited. During a 12-bit 4ch A/D converter sampling period with this setting, no clock is supplied to the 12-bit 4ch A/D converter.

### 5.1.4. A/D Conversion Ch Selection Register (ADCHSEL)

The A/D conversion ch selection register (ADCHSEL) sets the number of A/D conversion channels.

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved				ADCSL		Reserved	CHMD
ACCESS_TYPE	R0,W0				R/W		R0,W0	R/W
PROT_TYPE	-							
INITIAL_VALUE	0000				00		0	0

**[bit7:4] Reserved: Reserved bits**

**[bit3:2] ADCSL[1:0]: A/D conversion ch selection bits**

Value	Description
00	Perform A/D conversion in all the channels of Ach to Dch.
01	Perform A/D conversion in the 1 channel of Ach only.
10	Perform A/D conversion in the 2 channels of Ach and Bch.
11	Perform A/D conversion in the 3 channels of Ach, Bch, and Cch.

- These bits select the channels used for conversion in A/D conversion ch selection mode.
- This setting is disabled when CHMD is "0".

**[bit1] Reserved: Reserved bit**

**[bit0] CHMD: A/D conversion ch selection mode bit**

Value	Description
0	A/D 4ch conversion mode (initial value)
1	A/D conversion ch selection mode

- This bit selects A/D 4ch conversion mode or A/D conversion ch selection mode.

**Note:**

- Be sure that the A/D conversion ch selection mode bit (CHMD) and A/D conversion ch selection bits (ADCSL1, ADCSL0) are rewritten before the conversion operation while A/D operations are stopped (ADCS:BUSY is "0").



### 5.1.5. A/D Operation Enable Register (ADCEN)

The A/D operation enable register (ADCEN) enables 12-bit 4ch A/D converter operation.

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved							ENBL
ACCESS_TYPE	R0,W0							R/W
PROT_TYPE	-							
INITIAL_VALUE	0000000							0

**[bit7:1] Reserved: Reserved bits**

**[bit0] ENBL: 12-bit 4ch A/D operation enable bit**

Value	Description
0	Stop operation.
1	Enable operation.

- This bit enables 12-bit 4ch A/D converter operation.
- Writing "1" to the ENBL bit enables 12-bit 4ch A/D converter operation following the operation enable state transition period. Writing "0" to this bit stops 12-bit 4ch A/D converter operation. (The ADCS:READY bit becomes "0".)

**Notes:**

- Be sure that the 12-bit 4ch A/D operation enable bit (ENBL) is rewritten before the conversion operation while A/D operations are stopped (ADCS:BUSY is "0").
- When setting the CPU to watch mode or stop mode, set ENBL to "0" so that the A/D converter enters the operation stop state.

### 5.1.6. A/D Converter Enable Timer Register (ADCENTIME)

This register sets the number of cycles in the operation enable state transition period of the 12-bit 4ch A/D converter.

BIT_OFFSET	31-0
BIT_NAME	ENBLTIME
ACCESS_TYPE	R/W
PROT_TYPE	-
INITIAL_VALUE	11111111_11111111_11111111_11111111

#### [bit31:0] ENBLTIME[31:0]: Operation enable state transition cycle selection bits

Value	Description
0x0000_0000	These bits select the number of cycles in the operation enable state transition period.
...	
0xFFFF_FFFF	

- Operation enable state transition time = Peripheral clock x (ENBLTIME+1)  
Example) Suppose ENBLTIME[31:0] is "0x000F\_423F" and the peripheral clock is 100 MHz (10 ns).  
Operation enable transition time = 10 ns x (999999+1) = 10000000 ns = 10 ms
- Writing "1" to the ADCEN:ENBL bit starts the clock counting down from the number of cycles that is set in ADCENTIME:ENBLTIME as the initial value for operation enable state transitions.

#### Notes:

- Rewriting of the ENBLTIME[31:0] bits is prohibited in the period from the writing of "1" to the ADCEN:ENBL bit until the change of the ADCS:READY bit to "1".
- The A/D converter enable timer register is a 32-bit register. The register supports only word access. Byte access and halfword access are not supported.
- Be sure that the operation enable state transition cycle selection bits (ENBLTIME[31:0]) are rewritten before the conversion operation while A/D operations are stopped (ADCS:BUSY is "0").





### 5.1.7. 4ch ADC Analog Input Enable Registers (ADER4CH\_1, ADER4CH\_0)

The 4ch ADC analog input enable registers (ADER4CH\_1, ADER4CH\_0) are registers that control 4ch ADC analog input.

#### (1) 4ch ADC analog input enable register (ADER4CH\_1)

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved				ADE4C1U3	ADE4C1U2	ADE4C1U1	ADE4C1U0
ACCESS_TYPE	R0,W0				R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0000				1	1	1	1

[bit15:12] Reserved: Reserved bits

#### [bit11:8] ADE4C1U3 to ADE4C1U0: 4ch ADC analog input enable bits

Value	Description
0	Port input/output mode
1	Analog input mode

#### Notes:

- In models equipped with a key code function, a key code setting is required for writing to this register. For details on which models have the key code function, see "Part Number Option" in the Data Sheet.
- For the setting method, see the key code register (KEYCDR) descriptions in "CHAPTER: 12-bit A/D Converter Activation Compare".

**(2) 4ch ADC analog input enable register (ADER4CH\_0)**

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved				ADE4C0U3	ADE4C0U2	ADE4C0U1	ADE4C0U0
ACCESS_TYPE	R0,W0				R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0000				1	1	1	1

**[bit7:4] Reserved: Reserved bits**

**[bit3:0] ADE4C0U3 to ADE4C0U0: 4ch ADC analog input enable bits**

Value	Description
0	Port input/output mode
1	Analog input mode

**Notes:**

- In models equipped with a key code function, a key code setting is required for writing to this register. For details on which models have the key code function, see "Part Number Option" in the Data Sheet.
- For the setting method, see the key code register (KEYCDR) descriptions in "CHAPTER: 12-bit A/D Converter Activation Compare".



## 6. Precautions for Using

This section explains precautions on use of the 12-bit 4ch A/D converter interface.

### Notes to Observe on Using 12-bit 4ch A/D Converter Interface

#### a) Peripheral Clock Frequency Limits

For details on peripheral clock frequency limits, see the data sheet.

#### b) Sampling Time and Compare Time Settings

Configure the ADMD:ST[3:0] and ADMD:CT[3:0] bits such that the sampling time and compare time are within the recommended values for the 12-bit 4ch A/D converter.

#### c) ADMD Register, ADCHSEL Register, ADCENTIME Register, and ADCEN Register Settings

Be sure that the A/D mode setting register (ADMD), A/D conversion ch selection register (ADCHSEL), A/D converter enable timer register (ADCENTIME), and A/D operation enable setting register (ADCEN) bits are rewritten before the conversion operation while A/D conversion operations are stopped (ADCS:BUSY is "0").

#### d) A/D Converter Enable Timer Setting

Rewriting of the ENBLTIME[31:0] bits is prohibited in the period from the writing of "1" to the ADCEN:ENBL bit until the change of the ADCS:READY bit to "1".

When setting the CPU to watch mode or stop mode, set ENBL to "0" so that the A/D converter enters the operation stop state.

#### e) ADCENTIME Register Access Size

The A/D converter enable timer register (ADCENTIME) is a 32-bit register. The register supports only word access. Byte access and half-word access are not supported.

#### f) ADMDR Register Setting

The setting of "0b0000" in the ST[3:0] bits in the A/D mode setting register (ADMD) is prohibited. During a 12-bit 4ch A/D converter sampling period with this setting, no clock is supplied to the 12-bit 4ch A/D converter.

The setting of "0b0000" in the CT[3:0] bits in the A/D mode setting register (ADMD) is prohibited. During a 12-bit 4ch A/D converter compare period with this setting, no clock is supplied to the 12-bit 4ch A/D converter.

# CHAPTER 57: 12-bit 4ch A/D Converter A/D Activation Compare



This chapter explains the A/D activation compare function and operation of the 12-bit 4ch A/D converter.

---

1. Overview
2. Configuration
3. Operation
4. Setting Procedure Examples
5. Registers
6. Precautions for Using



## 1. Overview

The A/D activation compare block controls the 12-bit 4ch A/D converter.

### A/D Activation Compare Function

#### a) Activation Channel

- An activation channel is a unit for A/D activation request control and A/D conversion data storage.
- A/D activation requests are made in units of activation groups (activation group 0: activation channels 0 to 3; activation group 1: activation channels 4 to 7).
- An activation channel corresponds to 1 unit of the 12-bit 4ch A/D converter.
  - Activation channels 0 to 7: 12-bit 4ch A/D converter units
- Each activation channel consists of the following registers. (n=0 to 7)
  - Compare buffer register (ADCOMPB[n])/Compare register (ADCOMP[n])
  - A/D activation trigger control register (ADTC[n])
  - A/D activation request/interrupt status register (ADTS[n])
  - A/D activation request/interrupt clear register (ADTSC[n])
  - A/D data register (ADTCD[n])
  - Range comparison control status register (ADRCSS[n])
  - Range comparison out-of-range flag register (ADRCOT)
  - Range comparison flag register (ADRCIF)
  - Range comparison flag clear register (ADRCIFC)
  - Data protected state flag register (ADPRTF)

#### b) A/D Activation Request

- Each activation channel can select from the following A/D activation request factors. No A/D conversion (activation request) can be restarted inside an activation channel.
  - Software
  - External trigger (falling edge)
  - Base timer (rising edge)
  - Compare match
- The 12-bit 4ch A/D converter supports external trigger and base timer activation.
  - External trigger 0 (4ADTG0): 12-bit 4ch A/D converter unit 0
  - External trigger 1 (4ADTG1): 12-bit 4ch A/D converter unit 1
  - Base timer ch.0: 12-bit 4ch A/D converter unit 0
  - Base timer ch.1: 12-bit 4ch A/D converter unit 1
- In compare match activation, A/D activation is requested when the 16-bit free-run timer value matches the compare register of the respective activation channel.
- A/D activation is requested at any of the following times in compare match activation when the 16-bit free-run timer value matches the compare register.
  - Only when the 16-bit free-run timer is counting up
  - Only when the 16-bit free-run timer is counting down
  - When the 16-bit free-run timer is counting in either direction (up/down)
- A/D activation requests are output according to the following classifications for each activation channel.
  1. A/D activation request due to software activation
  2. A/D activation request due to an external trigger or the base timer
  3. A/D activation request due to compare match activation

- Single mode or repeat mode can be selected for the activation requests for each activation channel.
  - In single mode, a 1-time activation factor causes a 1-time activation request. A/D conversion is performed 1 time, and the activation request is released when the A/D conversion ends.
  - In repeat mode, a 1-time activation factor causes an activation request to be made continuously. A/D conversion is performed repeatedly, with the activation request being made continuously until repeat mode is canceled.

#### c) A/D Conversion Data

- At the end of A/D conversion, the converted data is stored in an A/D data register. Each activation channel has an A/D data register.
- Each A/D data register contains an error flag bit and an error status bit, enabling checking of the status of A/D conversion data.

#### d) Data Protection Function

- The data protection function can be configured for each A/D data register. The data protection function runs for factors other than compare match activation.
- When enabled, the data protection function masks an A/D activation request until the data in the A/D data register is read and an interrupt flag is cleared. The reading of the data and the clearing of the interrupt flag happen in random order. Also, it is possible to select whether to include the clearing of the interrupt flag as a protection condition.
- The A/D activation request busy bit provides notification of an ongoing A/D conversion request or ongoing conversion. The current A/D conversion request or conversion can be forcibly terminated by the writing of "1" to the A/D activation request clear bit (ADTSC[n]:BUSYC) in the A/D activation request/interrupt clear register (ADTSC[n]).
- The data protection function is enabled after conversion ends in all channels that had ongoing conversions.

#### e) Range Comparison Function

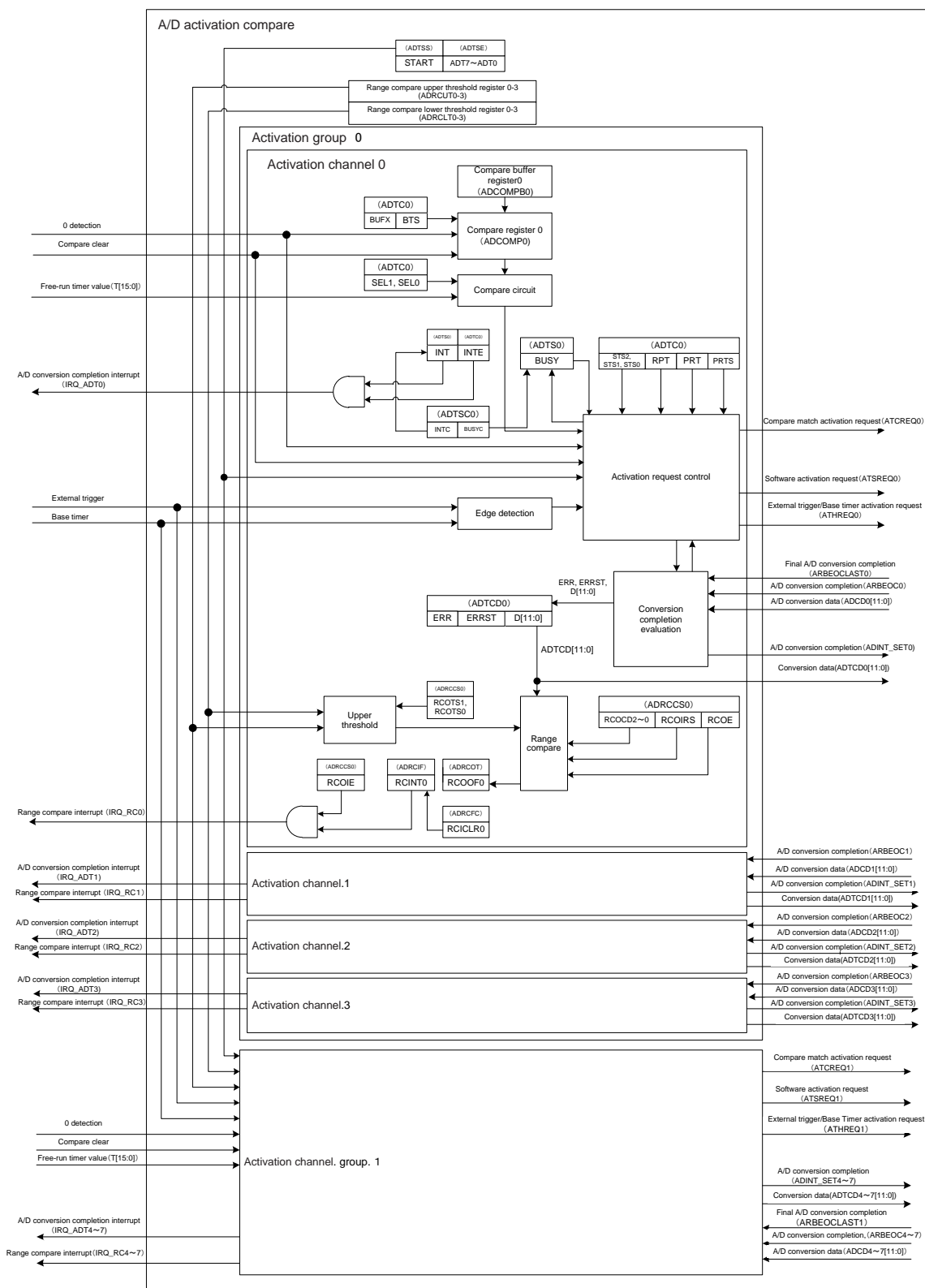
- The function can compare a range in each activation channel.
- Up to 4 types of settings are available for setting the upper- and lower-limit threshold values. An activation channel selects 1 combination from the 4 types of upper- and lower-limit threshold value settings to compare a range.
- A range comparison can select to confirm a result inside or outside the range of the upper- and lower-limit threshold values.
- Range comparison results enable noise reduction with the continuous detection function. The continuous detection function sets the range comparison flag based on a continuously detected range comparison result.
- A value of 1 to 7 times can be selected for the continuous detection frequency.
- The continuous detection count of a range comparison result can be checked.
- For confirmation of a result outside the range in a range comparison, the detection state can be checked to find out whether the result is above the upper-limit threshold value or below the lower-limit threshold value.

#### f) Interrupt Request

- Each activation channel can generate an interrupt request at the end of A/D conversion.
- Each activation channel can generate an interrupt request so that a range comparison is performed after A/D conversion ends.
- A range compare interrupt request can also be used as a DTTI input factor for the motor calculation accelerator.

This section shows an A/D activation compare block diagram.

### Figure 2-1 A/D Activation Compare Configuration



### 3. Operation

This section explains the A/D activation compare operation.

#### 3.1. A/D Activation Compare Interrupts

This section shows the A/D activation compare interrupt control bits and interrupt factors.

##### (1) A/D Conversion End Interrupt

**Table 3-1 Interrupt Control Bit and Interrupt Factor by A/D Conversion End Interrupt**

	<b>A/D Conversion End Interrupt</b>
Interrupt Request Flag Bit	INT:bit14 in A/D activation request/interrupt status register (ADTS[n])
Interrupt Request Enable Bit	INTE:bit13 in A/D activation trigger control register (ADTC[n])
Interrupt Factor	Conversion end signal for all channels

(n = Corresponding activation channel 0 to 7)

The A/D activation compare block can generate an A/D conversion end interrupt request when A/D conversion ends for all channels. A/D conversion end interrupts can be controlled in units of activation channels.

The interrupt request flag bit (ADTS[n]:INT) is set to "1" at the end time of A/D conversion for all channels. At this time, if the interrupt request enable bit is enabled (ADTC[n]:INTE is "1"), an interrupt request is output to the interrupt controller.

##### (2) Range Compare Interrupt

**Table 3-2 Interrupt Bit and Interrupt Factor by Range Compare Interrupt**

	<b>Range Compare Interrupt</b>
Interrupt Request Flag Bit	RCINT[n]:bit[n] in range comparison flag register (ADRCIF)
Interrupt Request Enable Bit	RCOIE:bit3 in range comparison control status register (ADRCSS[n])
Interrupt Factor	After judgment by continuous detection function, based on executed range comparison

(n = Corresponding activation channel 0 to 7)

When range comparison execution is enabled, the A/D activation compare block compares the range for conversion results stored in an A/D data register (ADTCD[n]). This comparison uses 1 combination from range comparison upper-limit threshold setting registers 0 to 3 (ADRCUT0 to 3)/lower-limit threshold setting registers 0 to 3 (ADRCUT0 to 3). To be able to confirm the continuity of range compared results, a range compare interrupt request can be generated. Also, range compare interrupts can be controlled in units of activation channels.

Once A/D conversion results are set in an A/D data register (ADTCD[n]) when range comparison execution is enabled (ADRCSS[n]:RCOE is "1"), a range comparison is executed. The outside/inside range confirmation selection bit (ADRCSS[n]:RCOIRS) can select a range comparison condition for the execution of the range comparison.

- Range comparison condition when outside-the-range confirmation is selected (ADRCSS[n]:RCOIRS="0")
  - Lower-limit threshold setting register > A/D conversion results or
  - Upper-limit threshold setting register < A/D conversion results
- Range comparison condition when inside-the-range confirmation is selected (ADRCSS[n]:RCOIRS="1")





Lower-limit threshold setting register  $\leq$  A/D conversion results and  
Upper-limit threshold setting register  $\geq$  A/D conversion results

The continuous detection of a range comparison condition sets the range compare interrupt factor flag bit (ADRCIF:RCINT[n]) to "1". At this time, if the range compare interrupt request enable bit is enabled (ADRCCS[n]:RCOIE is "1"), an interrupt request is output to the interrupt controller. The output timing of the range compare interrupt request for only the channel of the final conversion is later than the A/D conversion end interrupt. The output timing for other channels is earlier than the interrupts. Also, a continuous detection frequency of 1 to 7 times can be selected with the continuous detection frequency specification bits (ADRCCS[n]:RCOCD[2:0]).

## 3.2. A/D Activation Compare Operation (n=0 to 7)

An A/D activation request can come from any of the following: software, an external trigger, the base timer, and a compare match (when the 16-bit free-run timer value matches the compare register value).

### 3.2.1. A/D Activation (n=0 to 7)

Since each activation channel has registers for A/D activation requests, each has a structure that can be individually configured and controlled. However, configure activation channels 0 and 4 as described below because activation requests can be issued only by activation groups (activation group 0: activation channels 0 to 3; activation group 1: activation channels 4 to 7). Set software activation and repeat conversion mode for the remaining activation channels 1 to 3 and 5 to 7.

Activation channels generate A/D activation request signals for A/D activation arbitration. The signals are due to any of the following: software, an external trigger (falling), the base timer (rising), or a compare match (when the 16-bit free-run timer value matches the compare register value). The A/D activation request signals make the following 3 requests exclusively in each activation channel: "software activation request," "external trigger/base timer," and "compare match request".

When A/D conversion ends, the A/D activation request is cleared, and the converted data is stored in an A/D data register. An A/D conversion end interrupt can be generated at the time of completed conversion for all channels, simultaneously for all converted channels.

No activation request is restarted even if an activation factor occurs partway through an A/D activation request (ADTS[n]:BUSY is "1") in the activation channels.

**Note:**

- Before starting A/D conversion, configure activation channel 0 or 4 as described below.
- Be sure to set software activation and repeat conversion mode for activation channels 1 to 3 and 5 to 7 by using the A/D activation trigger control registers (ADTC[1] to ADTC[3] and ADTC[5] to ADTC[7]).
- For activation channels 1 to 3 and 5 to 7, enable software activation with the A/D software activation channel selection register (ADTSE), and set the activation of A/D conversion with the A/D software activation register (ADTSS).  
(Settings when using activation channels 1 to 3)  
To select 2 channels: Set the ADTSE:ADT1 bit to "1".  
To select 3 channels: Set the ADTSE:ADT1 and ADT2 bits to "1".  
To select 4 channels: Set the ADTSE:ADT1, ADT2, and ADT3 bits to "1".



### 3.2.2. A/D Activation Enable (n=0 to 7)

An A/D activation factor is selected according to the A/D activation factor selection bits (ADTC[n]:STS[2:0]). The selected factor is any of software, an external trigger, the base timer, or a compare match. When the selected activation factor occurs, an A/D activation request signal is generated for A/D activation arbitration.

Software activation is selected (ADTC[n]:STS[2:0] is "0b000") for activation channels that do not perform A/D activation. Furthermore, A/D activation requests in these channels can be disabled by the software activation disable setting (ADTSE:ADT[n] is "0") for the corresponding channels of the software activation channel selection register (ADTSE).

### 3.2.3. Free-run Timer Input

A free-run timer can be used as compare match input for the 12-bit A/D converter.

### 3.2.4. Software Activation (n=0 to 7)

The A/D activation factor selection bits are set for software activation (ADTC[n]:STS[2:0] is "0b000").

A software activation request signal is set when "1" is written to the A/D conversion activation (software) bit (ADTSS:START).

### 3.2.5. External Trigger Activation (n=0 to 7)

The A/D activation factor selection bits are set for external trigger activation (ADTC[n]:STS[2:0] is "0b001").

Each 12-bit A/D converter unit supports external triggers.

External trigger: Activation channels 0 to 7 (12-bit 4ch A/D converter units)

External trigger input becomes shared input for activation groups 0 and 1, and the input timing cannot be individually changed for each activation group.

An external trigger/base timer activation request signal is set when a falling edge of an external trigger is detected.

### 3.2.6. Base Timer Activation (n=0 to 7)

The A/D activation factor selection bits are set for base timer activation (ADTC[n]:STS[2:0] is "0b010").

The 12-bit 4ch A/D converter supports the base timer.

- Base timer: Activation channels 0 to 7 (12-bit 4ch A/D converter units)

Base timer input becomes shared input for activation groups 0 and 1, and the input timing cannot be individually changed for each activation group.

An external trigger/base timer activation request signal is set when rising is detected at the base timer.

### 3.2.7. Compare Match Activation (n=0 to 7)

The A/D activation factor selection bits are set for compare match activation (ADTC[n]:STS[2:0] is "0b011").

A compare match activation request is set when a compare register (ADCOMP[n]) has a match between the 16-bit free-run timer and the compare register value (ADCOMP[n]:CMP[15:0]).

The preferred timer value for activation is set in the compare register (ADCOMP[n]). To make an A/D activation request at the same time as the 0 detection time of the 16-bit free-run timer, set "0x0000". To perform A/D activation at the same time as the compare clear time of the 16-bit free-run timer, set the same value as the compare clear value of the 16-bit free-run timer.

#### (1) Operation Mode of Compare Match Activation

The count direction selection bits (ADTC[n]:SEL[1:0]) control the operation mode of the compare match function.

The count direction selection bits (ADTC[n]:SEL[1:0]) select whether to compare the compare register value (ADCOMP[n]:CMP[15:0]) and the 16-bit free-run timer when counting in either direction (up/down), only when counting up, or only when counting down. If the count direction selection bits (ADTC[n]:SEL[1:0]) are set to "0b11", no activation request signal is generated for A/D activation arbitration, even when the 16-bit free-run timer matches the compare register value (ADCOMP[n]:CMP[15:0]).

**Table 3-3 Control Details of Count Direction Selection Bits (ADTC[n]:SEL[1:0])**

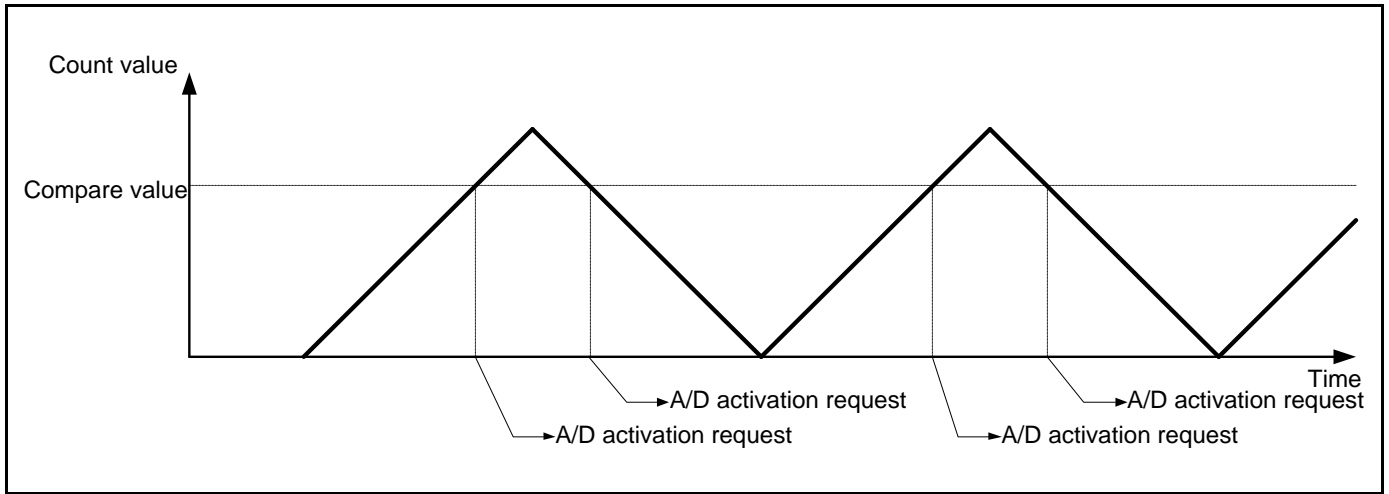
ADTC[n] SEL[1:0]		Function	Reference Drawing
0	0	When counting in either direction (up/down)	Figure 3-1
0	1	Only when counting up	Figure 3-2
1	0	Only when counting down	Figure 3-3
1	1	Compare disabled	-

**Note:**

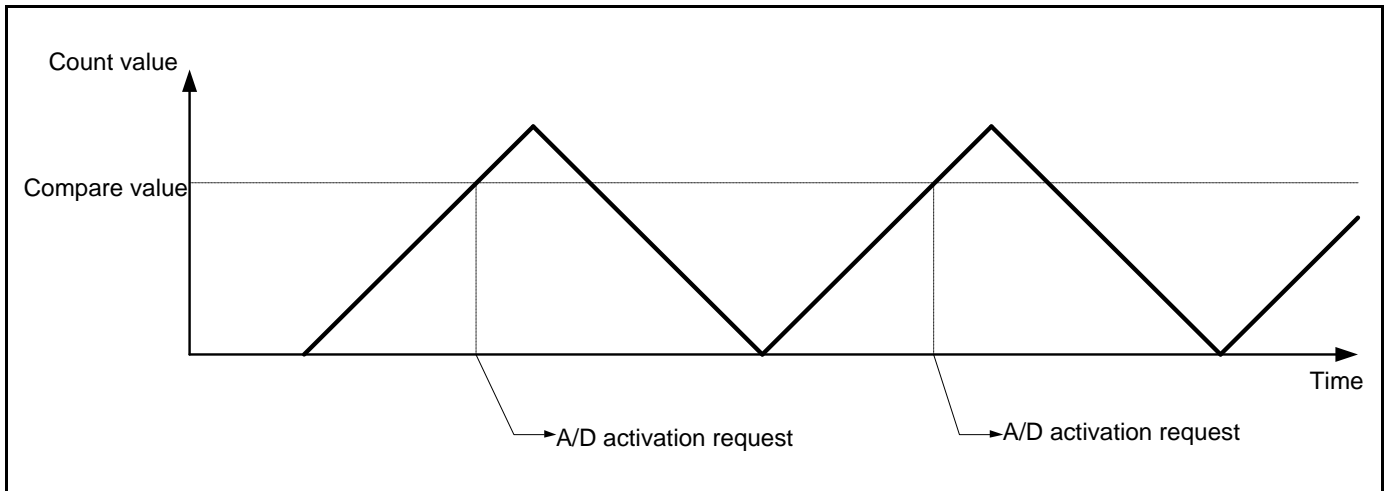
- If "0x0000" is the set value in the compare register (ADCOMP[n]) and the compare clear register of the 16-bit free-run timer has the same set value, an A/D activation request signal is generated at the compare match time. That would occur regardless of whether the count direction in the count direction selection bit setting (ADTC[n]:SEL[1:0]) is up or down.



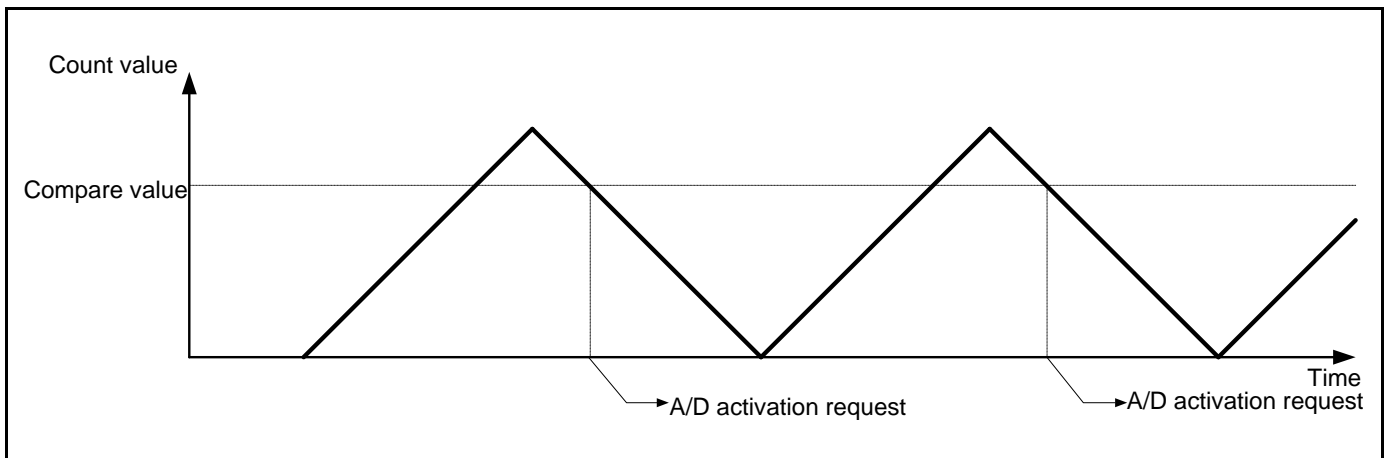
**Figure 3-1 ADTC[n]:SEL:[1:0]="0b00": Activation at Compare Match Time when Counting in Either Direction (Up/Down)**



**Figure 3-2 ADTC[n]:SEL:[1:0]="0b01": Activation at Compare Match Time Only when Counting Up**



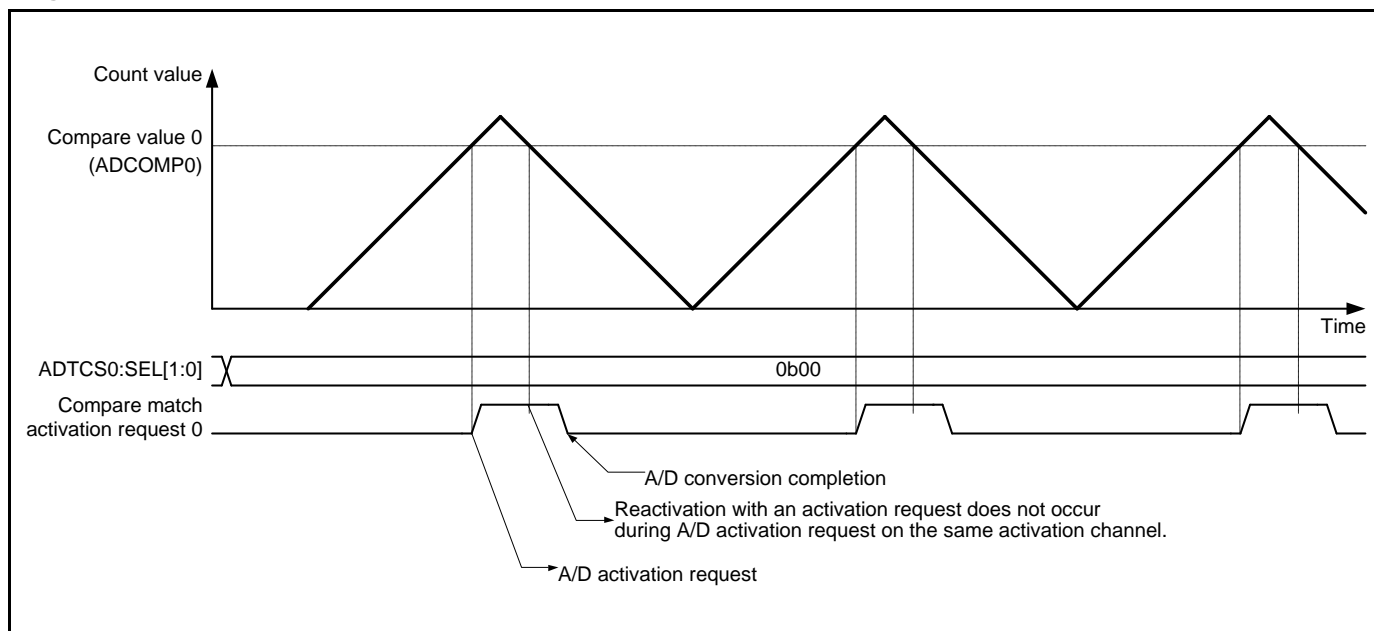
**Figure 3-3 ADTC[n]:SEL:[1:0]="0b10": Activation at Compare Match Time Only when Counting Down**



### (2) About Compare Value Setting for Compare Match Activation

If the compare match generation interval is shorter than the A/D conversion time within the same activation channel for compare match activation, any compare match that occurs during A/D conversion is ignored.

**Figure 3-4 When Compare Match Occurrence Interval is Shorter than A/D Conversion Time in Same Activation Channel**



Also, if the compare match generation interval is shorter than the A/D conversion time between activation channels that activate the same A/D converter unit, any activation request generated during A/D conversion delays the start of A/D conversion. A/D conversion does not begin at the intended time and the start of A/D conversion is delayed.

### (3) Compare Register Buffer Function for Compare Match Activation

The compare register buffer function control bit (ADTC[n]:BUFEX) can select whether to enable use of the compare register buffer function. Writing "0" to the compare register buffer function control bit (ADTC[n]:BUFEX) enables use of the compare register buffer function.

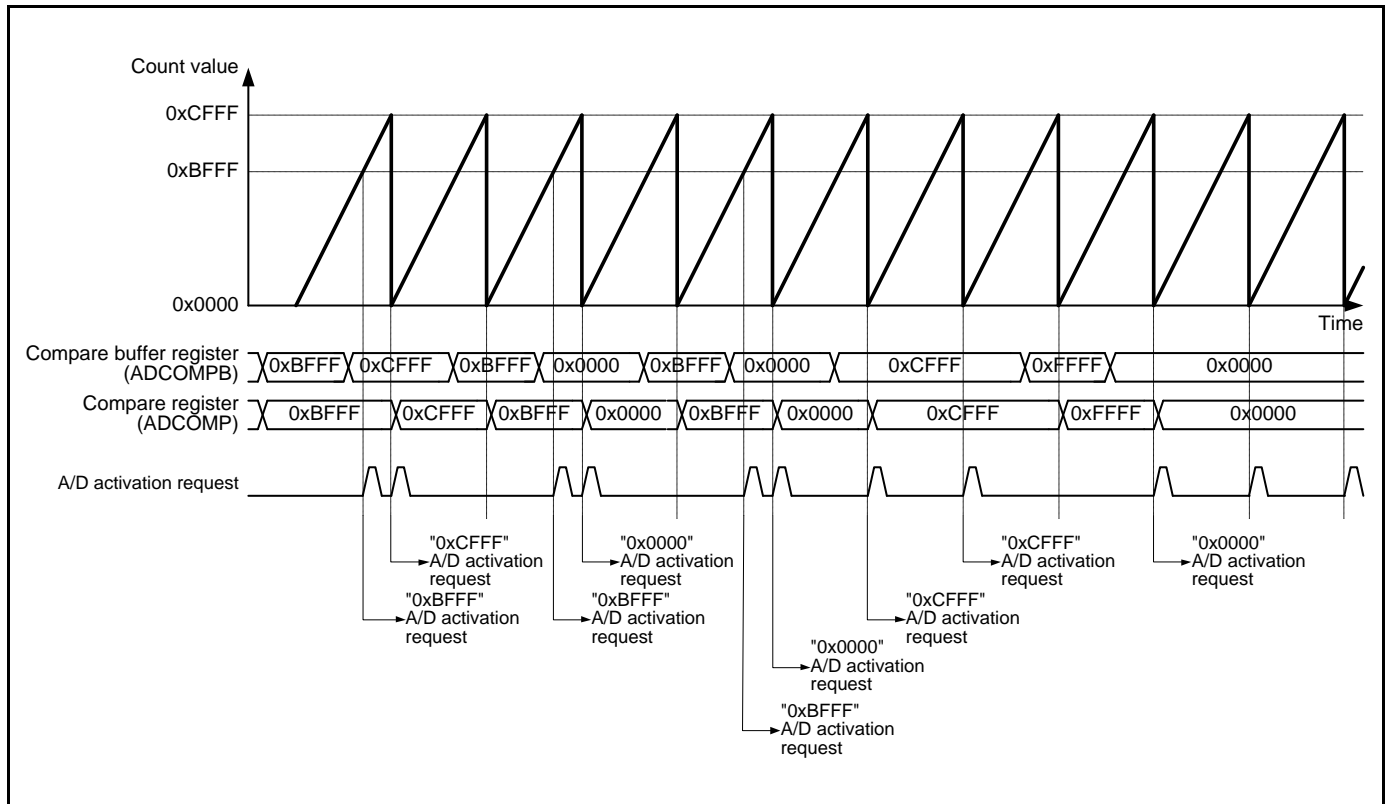
The timing of transfer from the buffer when the buffer function is enabled (ADTC[n]:BUFEX is "0") can be selected by the compare register buffer transfer control bit (ADTC[n]:BTS). The value written in the compare buffer register (ADCOMPB[n]) is transferred to the compare register (ADCOMP[n]) at the following times: at the compare clear time if ADTC[n]:BTS is "1", and at the 0 detection time if ADTC:BTS is "0".

**Table 3-4 Conditions for Transfer from Compare Buffer Register to Compare Register**

ADTC[n]		Condition for Transfer from Compare Buffer Register (ADCOMPB[n]) to Compare Register (ADCOMP[n])
BUFEX	BTS	
0	0	When 0 is detected in 16-bit free-run timer, or while 16-bit free-run timer is stopped
0	1	When compare clear is done on 16-bit free-run timer, or while 16-bit free-run timer is stopped
1	0 or 1	Buffer function not used (immediate transfer)



**Figure 3-5 Timing of Compare Register Data Transfer as Compare Clear Register is Found to Match when 16-bit Free-run Timer is Counting Up**



**Figure 3-6 Timing of Compare Register Data Transfer as 0 is Detected when 16-bit Free-run Timer is Counting Up**

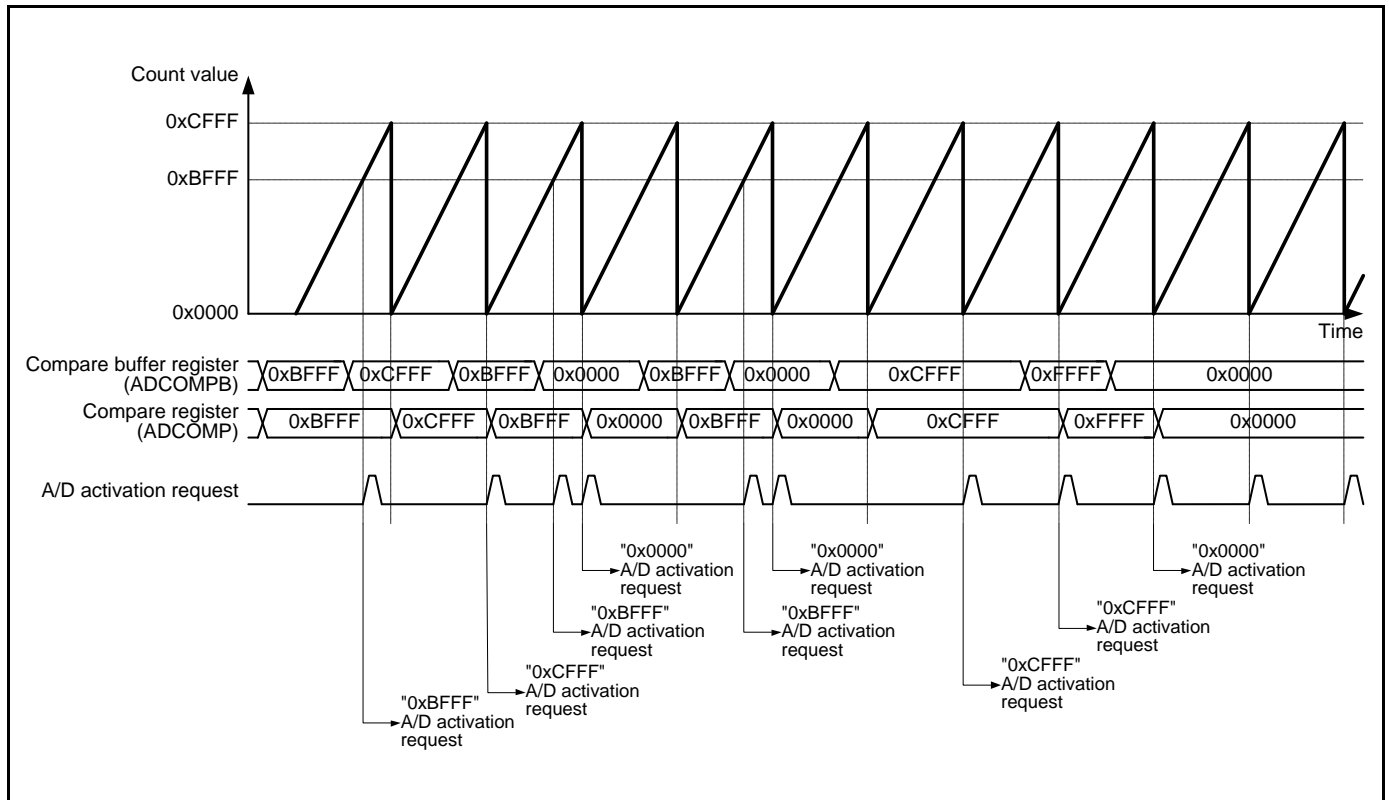


Figure 3-7 Timing of Compare Register Data Transfer as Compare Clear Register is Found to Match when 16-bit Free-run Timer is Counting Down

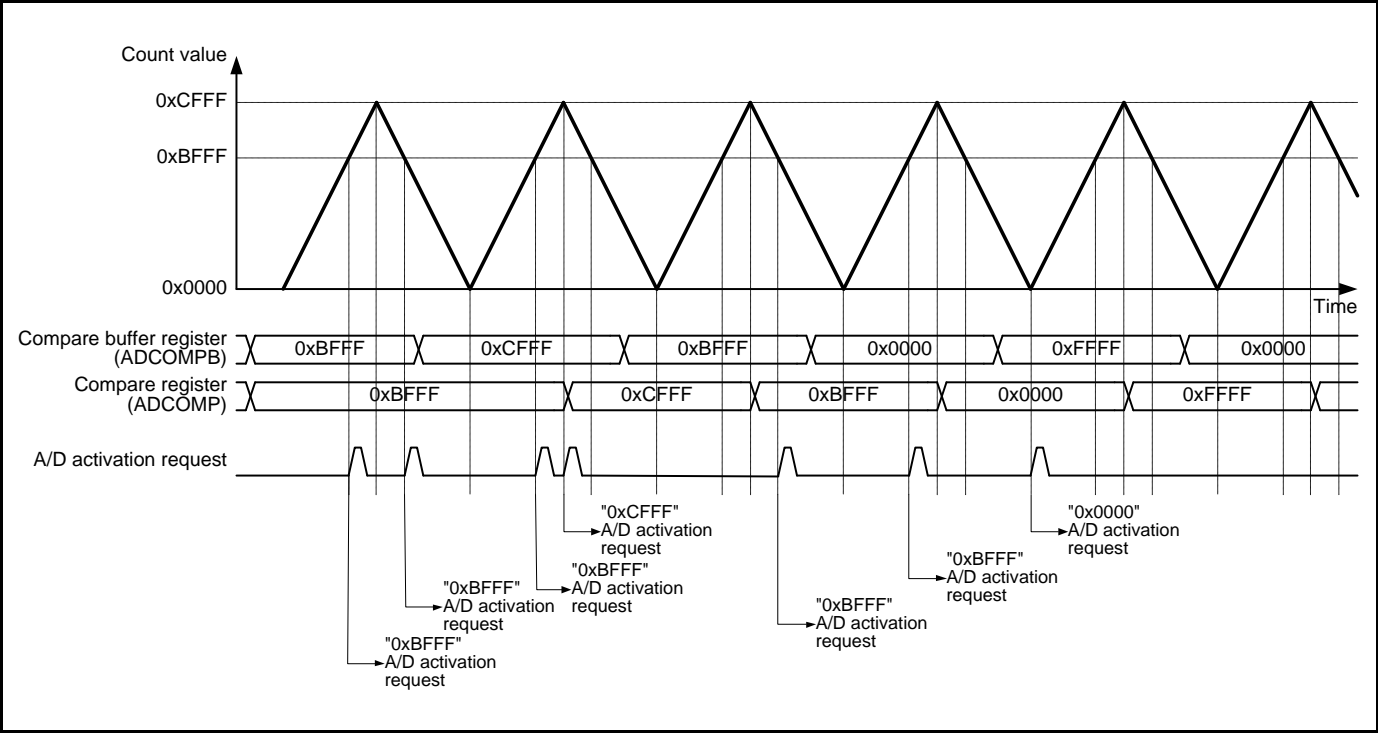
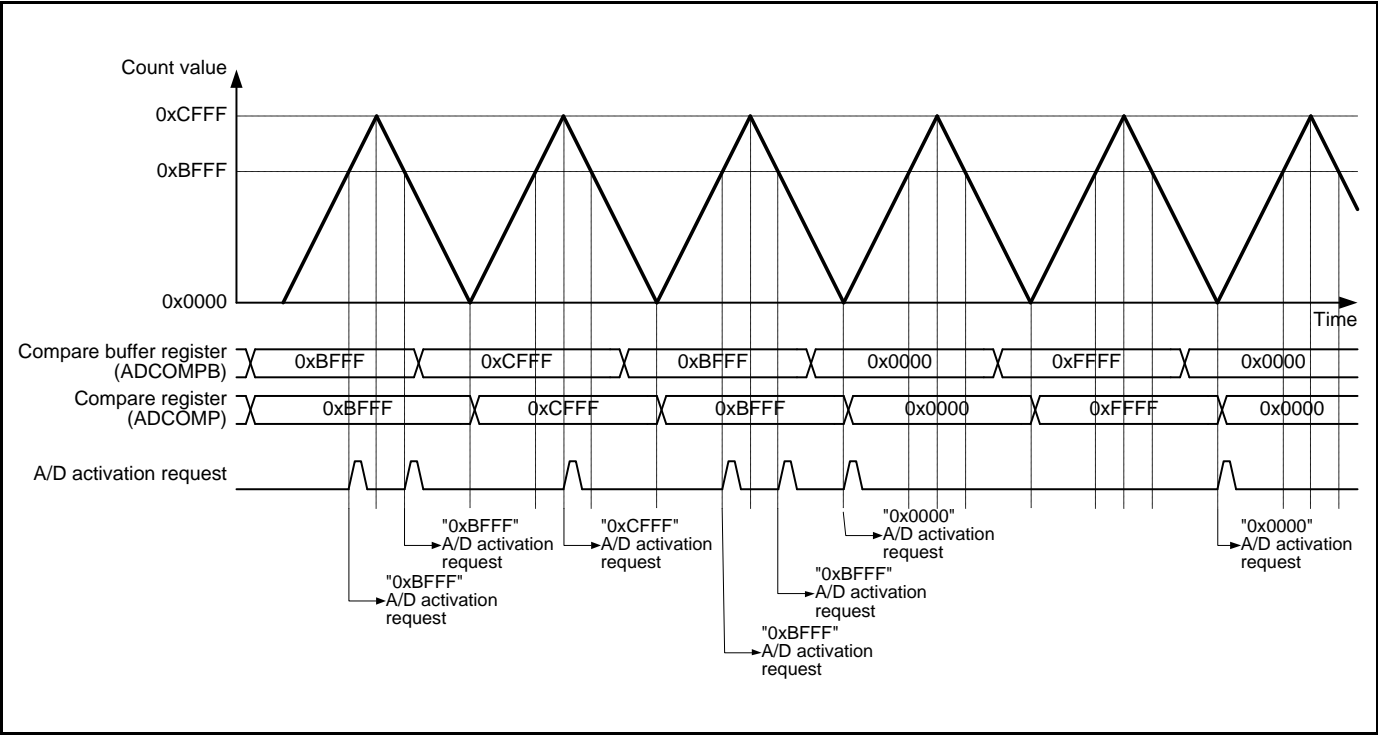


Figure 3-8 Timing of Compare Register Data Transfer as 0 is Detected when 16-bit Free-run Timer is Counting Up or Down





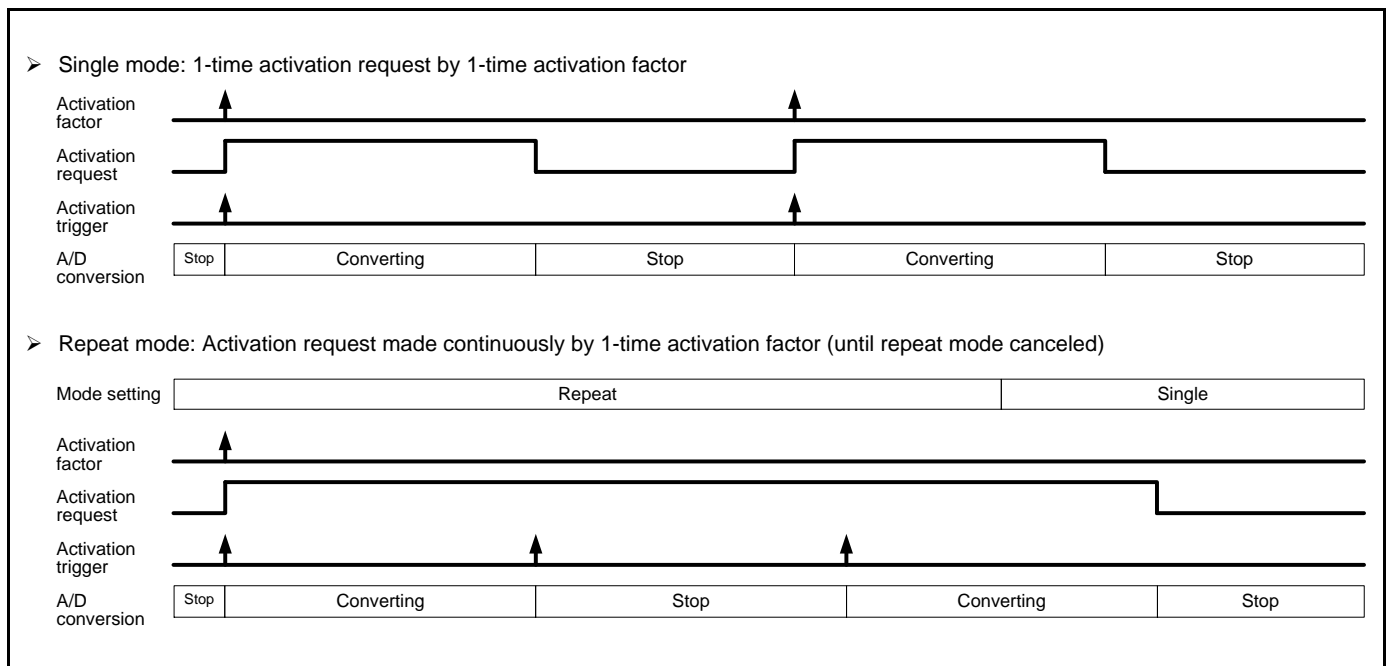


### 3.2.8. Activation Request Mode (n=0 to 7)

An activation request mode can be set for each activation channel. There are 2 activation request modes: single mode and repeat mode. The repeat conversion selection bit (ADTC[n]:RPT) sets the mode.

- In single mode (ADTC[n]:RPT is "0"), a 1-time activation factor causes a 1-time activation request. A/D conversion is performed 1 time (channel 0 to channel 3), and the activation request is removed when the A/D conversion ends.
- In repeat mode (ADTC[n]:RPT is "1"), a 1-time activation factor causes an activation request to be made continuously. A/D conversion is performed repeatedly, with the activation request being made continuously until repeat mode is canceled.
- After an A/D conversion end interrupt, if an activation request mode is set, the setting will be enabled at the time that the next A/D conversion ends.

Figure 3-9 Activation Request Modes



### 3.2.9. A/D Conversion Data (n=0 to 7)

The A/D data bits (ADTCD[n]:D[11:0]) store A/D conversion result data.

The conversion data error flag bit (ADTCD[n]:ERR) and the conversion data error status bit (ADTCD[n]:ERRST) enable checking of the status of A/D conversion data stored in the A/D data bits (ADTCD[n]:D[11:0]) when the data protection function is disabled (ADTC[n]:PRT is "0") or during compare match activation (ADTC[n]:STS[2:0] is "0b011"). For details on the operation of the conversion data error flag bit (ADTCD:ERR) and conversion data error status bit (ADTCD:ERRST), see Figure 3-10.

The conversion data error flag bit (ADTCD[n]:ERR) and the conversion data error status bit (ADTCD[n]:ERRST) are fixed at "0" when the data protection function is enabled (ADTC[n]:PRT is "1").

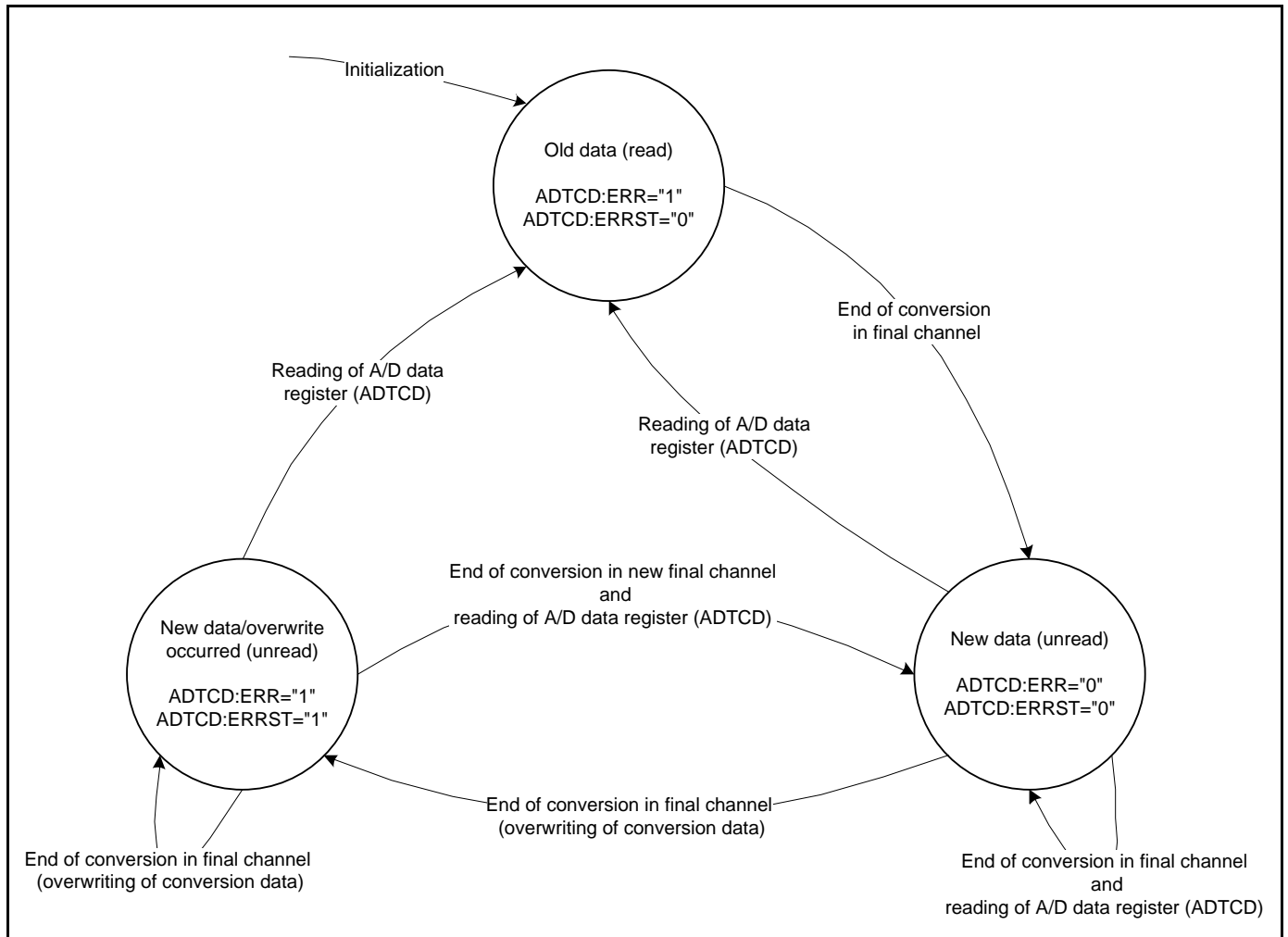
**Table 3-5 Checking of Status of A/D Conversion Data (When Data Protection Function is Disabled (ADTC[n]:PRT="0") or during Compare Match Activation (ADTC[n]:STS[2:0]="0b011"))**

ADTCD[n]:ERR	ADTCD[n]:ERRST	A/D Conversion Data Status
0	0	Latest data (unread)
0	1	- (no meaning)
1	0	Old data (read) Note: Initial value
1	1	Latest data/overwrite occurred (unread) Note: Data discarded

No combination with ADTCD[n]:ERR and ADTCD[n]:ERRST both being "0b01" is generated from the hardware.



**Figure 3-10 A/D Conversion Data State Control (When Data Protection Function is Disabled (ADTCS:PRT="0") or during Compare Match Activation (ADTC[n]:STS[2:0]="0b011"))**



**Note:**

- The conversion data error flag bit (ADTCD[n]:ERR) and the conversion data error status bit (ADTCD[n]:ERRST) are updated at the time that the conversion data for the set final channel is stored in an A/D data register (ADTCD[n]).

**Table 3-6 Update Timing Example of ADTCD[n]:ERR and ADTCD[n]:ERRST (n=0 to 7)**

Activation Request Mode	Data Protection Function	Activation Factor	ADTCD[n]:ERR, ADTCD[n]:ERRST Update Timing
Single mode (ADTC[n]:RPT="0") (n=0 to 7)	Disabled (ADTC[n]:PRT="0") (n=0 to 7)	Compare match activation (ADTC[n]:STS[2:0]="0b011") (n=0 to 7)	Timing at which A/D activation request busy bit (ADTS[n]:BUSY) changes from "1" to "0" (n=0, 4)
		Other than compare match activation	
	Enabled (ADTC[n]:PRT="1") (n=0 to 7)	Compare match activation (ADTC[n]:STS[2:0]="0b011") (n=0 to 7)	ADTCD[n]:ERR and ADTCD[n]:ERRST fixed at "0"
		Other than compare match activation	

Activation Request Mode	Data Protection Function	Activation Factor	ADTCD[n]:ERR, ADTCD[n]:ERRST Update Timing
Repeat mode (ADTC[n]:RPT="1") (n=0 to 7)	Disabled (ADTC[n]:PRT="0") (n=0 to 7)	Compare match activation (ADTC[n]:STS[2:0]="0b011") (n=0 to 7)	Timing of interrupt request flag bit (ADTS[n]:INT) change from "0" to "1" (n=0 to 7)
		Other than compare match activation	
	Enabled (ADTC[n]:PRT="1") (n=0 to 7)	Compare match activation (ADTC[n]:STS[2:0]="0b011") (n=0 to 7)	
		Other than compare match activation	ADTCD[n]:ERR and ADTCD[n]:ERRST fixed at "0"



### 3.2.10. Data Protection Function

The data protection function can be configured for an A/D data register (ADTCD[n]).

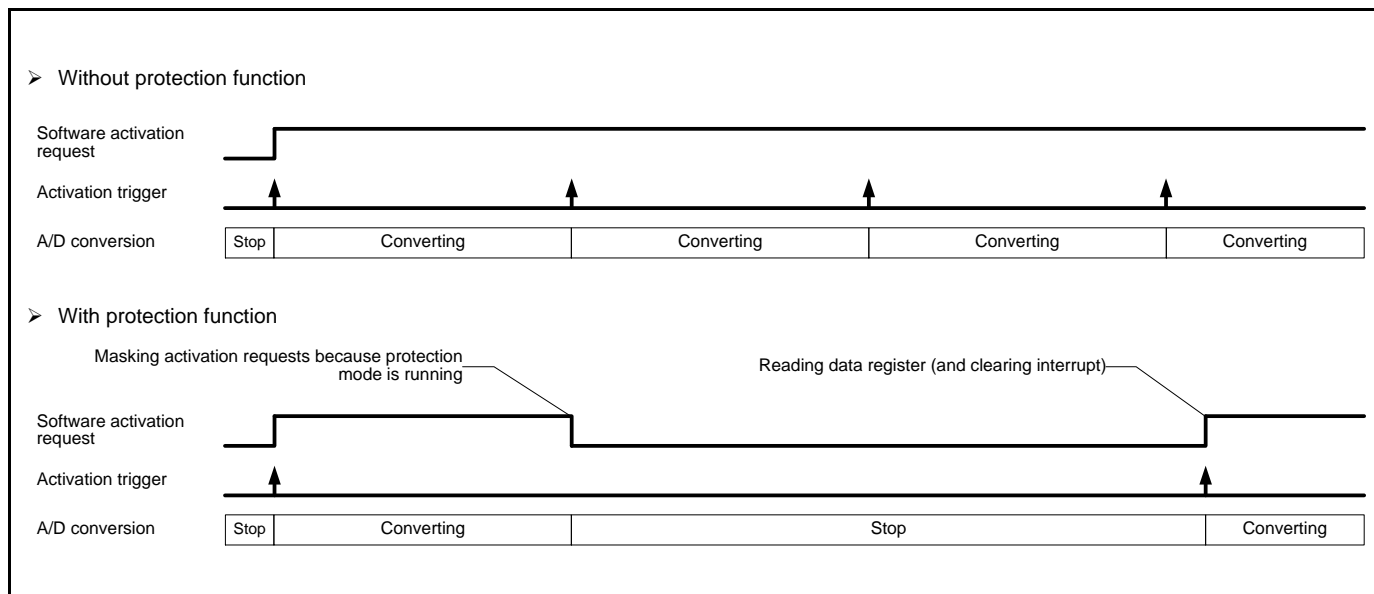
The data protection function is set from the A/D data register protection enable bit (ADTC[n]:PRT). The data protection function runs for factors other than compare match activation (ADTC[n]:STS[2:0] is "0b011").

Conversion results stored in an A/D data register (ADTCD[n]) enter the data protected state when the data protection function is enabled (ADTC[n]:PRT is "1"). The A/D data register protection clear selection bit (ADTC[n]:PRTS) can select a condition for clearing the protected data state. While in the data protected state, unread data in the A/D data register (ADTCD[n]) is protected from being overwritten by subsequent A/D conversion data. The protection is provided by masking (making inactive) of activation request signals generated by subsequent activation factors.

- If the A/D data register protection clear selection bit (ADTC[n]:PRTS) is "0", activation requests are masked until the data in the A/D data register (ADTCD[n]) is read and the interrupt flag (ADTS[n]:INT) is cleared. The reading of the data and the clearing of the interrupt flag happen in random order.
- If the A/D data register protection clear selection bit (ADTC[n]:PRTS) is "1", activation requests are masked until the data in the A/D data register (ADTCD[n]) is read.

The data protection function is enabled after conversion ends in all conversion channels.

**Figure 3-11 Data Protection Function (Example of Software Activation Request in Repeat Mode (ADTC[n]:RPT="1"))**



**Notes:**

- In a state where the A/D data register protection clear selection bit of activation channel 0 or 4 is "1" (ADTC[0]:PRTS is "1" or ADTC[4]:PRTS is "1") with the data protection function enabled for that activation channel (ADTC[0]:PRT is "1" or ADTC[4]:PRT is "1"), the data protection setting is released when the A/D data register of the activation channel (ADTCD[0] or ADTCD[4]) is read.
- In a state where the A/D data register protection clear selection bit of activation channel 0 or 4 is "0" (ADTC[0]:PRTS is "0" or ADTC[4]:PRTS is "0"), the data protection setting is released when the A/D data register of the activation channel (ADTCD[0] or ADTCD[4]) is read and the A/D conversion end interrupt in the activation channel is cleared.
- The release of the data protection setting leads to a delay in the reading of the A/D data registers of activation channels 1 to 3 or activation channels 5 to 7 (ADTCD[1] to ADTCD[3] or ADTCD[5] to ADTCD[7]). If they are not read by the time that the next A/D conversion results are to be written to these A/D data registers, the data in the registers will be overwritten. To prevent this overwriting, read the A/D data registers of activation channels 1 to 3 or activation channels 5 to 7 before reading the A/D data register of activation channel 0 or 4, respectively.

### 3.2.11. Forced Termination of Activation Request

The A/D activation request busy bit (ADTS[n]:BUSY) provides notification of an ongoing A/D activation request or ongoing conversion. To forcibly terminate the current A/D activation request or conversion, write "1" to the A/D activation request clear bit (ADTSC[n]:BUSYC).



### 3.2.12. Range Comparison Function

#### (1) Range Comparison Upper- and Lower-limit Threshold Value Settings

The upper-limit threshold setting register (ADRCUT) and the lower-limit threshold setting register (ADRCLT) can have 4 types of settings. From the combinations of the 4 types in the upper- and lower-limit threshold setting registers, 1 combination is selected in the upper- and lower-limit threshold selection bits (ADRCSS[n]:RCOTS[1:0]) of each activation channel.

**Table 3-7 Range Comparison Upper- and Lower-limit Threshold Value Selection**

Upper- and Lower-limit Threshold Selection Bits (ADRCSS[n]:RCOTS[1:0])		Selection Result
0	0	Upper-limit threshold setting register 0 (ADRCUT0) / lower-limit threshold setting register 0 (ADRCLT0)
0	1	Upper-limit threshold setting register 1 (ADRCUT1) / lower-limit threshold setting register 1 (ADRCLT1)
1	0	Upper-limit threshold setting register 2 (ADRCUT2) / lower-limit threshold setting register 2 (ADRCLT2)
1	1	Upper-limit threshold setting register 3 (ADRCUT3) / lower-limit threshold setting register 3 (ADRCLT3)

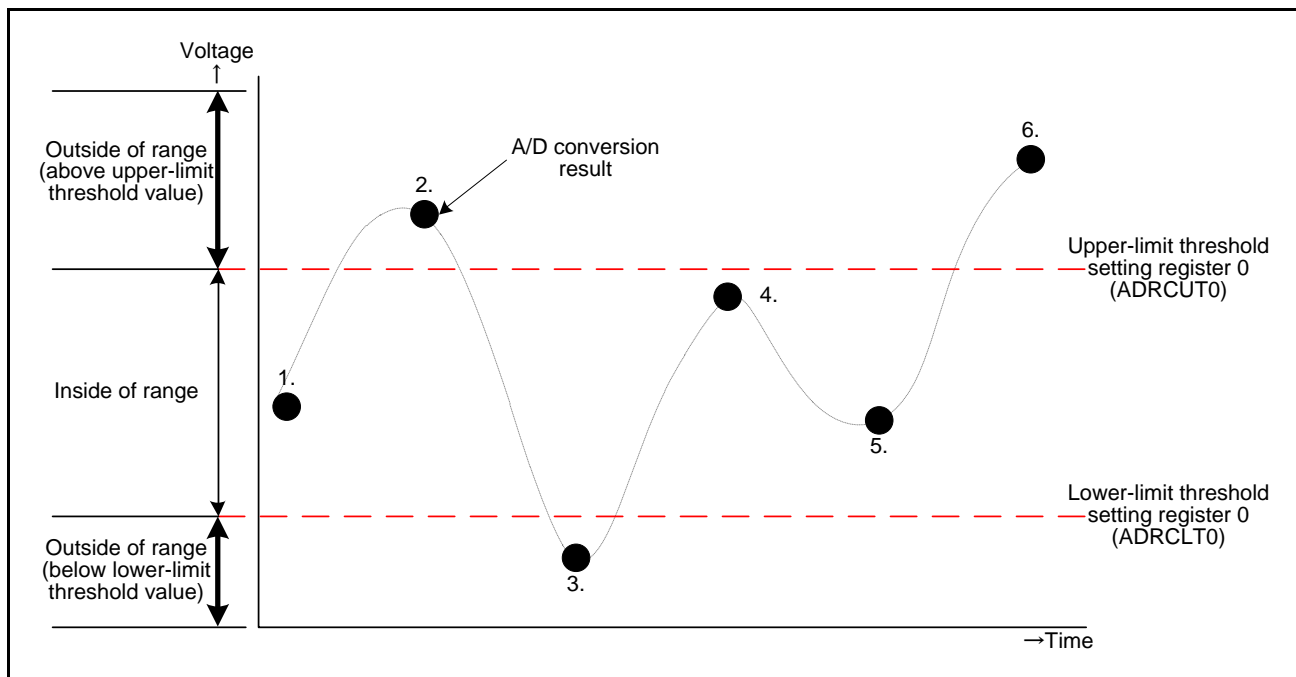
#### (2) Range Comparison Operation

When enabled (ADRCSS[n]:RCOE is "1"), a range comparison is executed after A/D conversion ends and data is stored in A/D data bits (ADTCD[n]:D[11:0]). The range comparison compares the A/D data bits (ADTCD[n]:D[11:0]) with the upper- and lower-limit threshold setting registers (ADRCUT/ADRCLT) selected by the range comparison upper- and lower-limit threshold selection bits (ADRCSS[n]:RCOTS[1:0]). The range comparison result is input to the continuous detection function.

**Table 3-8 Range Comparison Conditions**

Range Comparison Result	Outside-of-range Confirmation (ADRCSS[n]:RCOIRS="0")	Inside-of-range Confirmation (ADRCSS[n]:RCOIRS="1")
Outside of range (above upper-limit threshold value) A/D data bits > Upper-limit threshold setting register (Figure 3-12: 2., 6.)	Detected	Undetected
Inside of range A/D data bits ≥ Lower-limit threshold setting register and A/D data bits ≤ Upper-limit threshold setting register (Figure 3-12: 1., 4., 5.)	Undetected	Detected
Outside of range (below lower-limit threshold value) A/D data bits < Lower-limit threshold setting register (Figure 3-12: 3.)	Detected	Undetected

Figure 3-12 Range Comparison Conditions



### (3) Continuous Detection Function for Range Comparison Results

The continuous detection function reduces noise, etc. through the continuous detection of a range comparison result.

The continuous detection frequency specification for range comparisons (ADRCSS[n]:RCOCD[2:0]) sets the detection state for a range comparison result. After the set number of times of continuous detection according to that specification, the range compare interrupt factor flag bit (ADRCIF:RCINT) is set to "1". The continuous detection is ongoing until the first time that the range comparison result fails to be detected. After that, continuous detection measurement is cleared to 0 times and the measurement begins again.

Table 3-9 Continuous Detection Function Operating Conditions

Continuous Detection Measurement Operation	<ul style="list-style-type: none"> <li>- The operation is controlled separately for each activation channel.</li> <li>- The operation is always running when range comparison execution is enabled (ADRCSS[n]:RCOE="1").</li> </ul>
Continuous Detection Frequency	<ul style="list-style-type: none"> <li>- A value of 1 to 7 times can be selected using the continuous detection frequency specification (ADRCSS[n]:RCOCD[2:0]).</li> <li>- The detection count can be checked using the continuous detection count display (ADRCSS[n]:RCOCD[2:0]).</li> </ul>
Clearing Conditions	<ul style="list-style-type: none"> <li>- A condition is when range comparison execution is disabled (ADRCSS[n]:RCOE="0").</li> <li>- A condition is when the range comparison result is not detected.</li> </ul>
Increment Conditions	<ul style="list-style-type: none"> <li>- A condition is when the range comparison result is detected.</li> </ul> <p>However, if the count reaches the continuous detection frequency specification (ADRCSS[n]:RCOCD[2:0]) value, it stops at this value.</p>





**Notes:**

- Even if a range comparison result changes from being above the upper-limit threshold value to below the lower-limit threshold value when confirmed as outside the range (ADRCSS[n]:RCOIRS is "0"), the continuous detection of the result continues without continuous detection measurement being cleared to 0 times.
- To initialize the continuous detection count for a range comparison result, change the range comparison execution setting from disabled to enabled (from "0" to "1" in ADRCSS[n]:RCOE) while no A/D conversion is requested (ADTS[n]:BUSY is "0").
- Except in the final conversion channel, a continuous detection sets the range compare interrupt factor flag (ADRCIF:RCINT[n]) to "1" before an A/D conversion end interrupt sets the interrupt control bit flag (ADTS[n]:INT is "1"). In the final conversion channel, that is done after the A/D conversion end interrupt sets the interrupt control bit flag (ADTS[n]:INT is "1").

**(4) Range Comparison Out-of-range Flag Control**

For a range comparison that confirms the result to be outside the range (ADRCSS[n]:RCOIRS is "0"), the range comparison out-of-range flag bit (ADRCOT:RCOOF[n]) in the respective channel indicates whether the result is above the upper-limit threshold value or below the lower-limit threshold value.

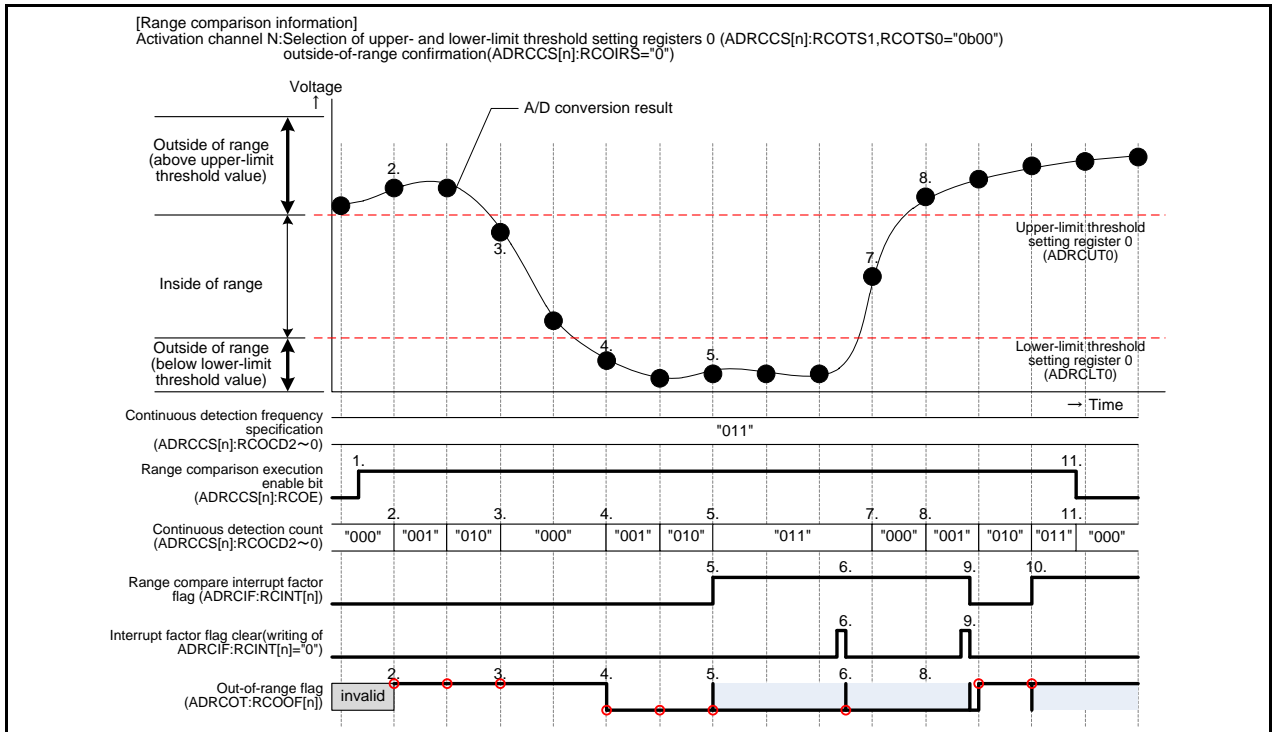
**Table 3-10 Conditions Determining Range Comparison Out-of-range Flag**

Range Comparison Result	Range Comparison Out-of-range Flag Bit (ADRCOT:RCOOF[n])	
	Outside-of-range Confirmation (ADRCSS[n]:RCOIRS="0")	Inside-of-range Confirmation (ADRCSS[n]:RCOIRS="1")
Outside of range (above upper-limit threshold value) A/D data bits > Upper-limit threshold setting register	"1"	Last value retained
Inside of range A/D data bits ≥ Lower-limit threshold setting register and A/D data bits ≤ Upper-limit threshold setting register	Last value retained	Last value retained
Outside of range (below lower-limit threshold value) A/D data bits < Lower-limit threshold setting register	"0"	Last value retained

Also, while the range compare interrupt factor flag (ADRCIF:RCINT[n]) is set to "1", the contents of the range comparison out-of-range flag bit (ADRCOT:RCOOF[n]) are retained.

### (5) Range Comparison Operation Example

Figure 3-13 Range Comparison Operation Example



1. The continuous detection count (ADRCSS[n]:RCOCD[2:0]) is initialized to "0b000" when range comparison execution is disabled (ADRCSS[n]:RCOE is "0").  
The range comparison operation begins according to the range comparison execution enable setting (ADRCSS[n]:RCOE is "1").
2. The range comparison result above the upper-limit threshold value causes an increment in the continuous detection count (ADRCSS[n]:RCOCD[2:0]).  
Also, the out-of-range flag provides notification that the result exceeded the upper-limit threshold value (ADRCOT:RCOOF[n] is "1").
3. The continuous detection count is initialized (ADRCSS[n]:RCOCD[2:0] is "0b000") because the range comparison result is detected inside the range before the continuous detection frequency specification value is reached (ADRCSS[n]:RCOCD[2:0] is "0b011").  
Also, the out-of-range flag (ADRCOT:RCOOF[n]) retains its previous value.
4. The range comparison result below the lower-limit threshold value causes an increment in the continuous detection count (ADRCSS[n]:RCOCD[2:0]).  
Also, the out-of-range flag provides notification that the result fell below the lower-limit threshold value (ADRCOT:RCOOF[n] is "0").
5. The number of consecutive range comparison results that are outside the range reaches the continuous detection frequency specification value (ADRCSS[n]:RCOCD[2:0] is "0b011"). As a result, the range compare interrupt factor flag (ADRCIF:RCINT[n]) is set to "1".  
Also, the out-of-range flag (ADRCOT:RCOOF[n]) is set to the out-of-range state for the time that the range compare interrupt factor flag was set (ADRCIF:RCINT[n] is "1"). This out-of-range setting is retained until the range compare interrupt factor flag is cleared (ADRCIF:RCINT[n]C is "1").
6. If the range compare interrupt factor flag clear (ADRCIF:RCINT[n]C is "1") is in contention with the continuous detection state, the continuous detection state determines the priority by the set operation.  
With the range compare interrupt factor flag in the set state (ADRCIF:RCINT[n] is "1"), the out-of-range flag (ADRCOT:RCOOF[n]) resets the out-of-range state.

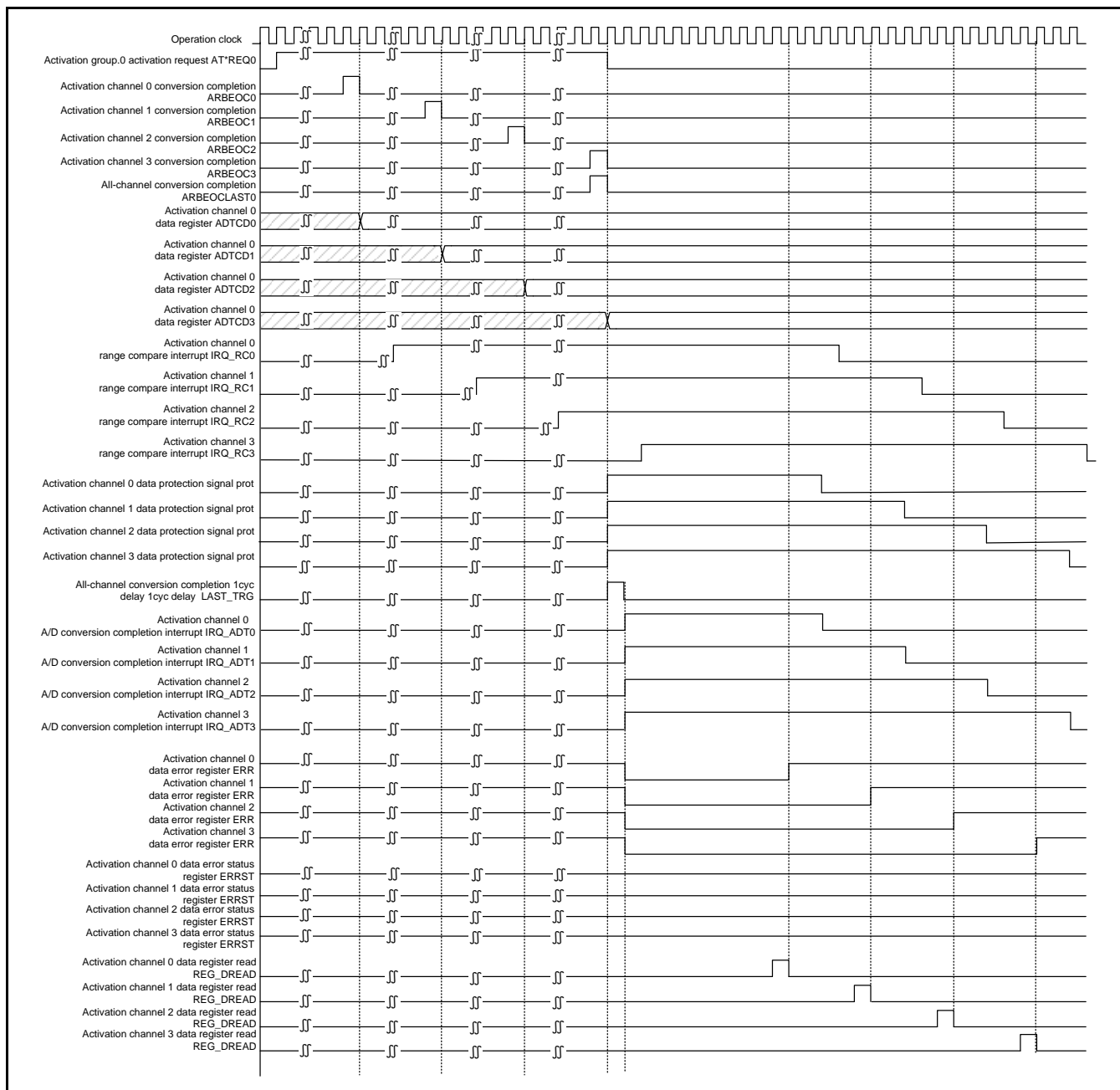


7. The continuous detection count is initialized (ADRCCS[n]:RCOCD[2:0] is "0b000") when the range comparison result is inside the range, even though the range compare interrupt factor flag is set (ADRCIF:RCINT[n] is "1").
8. The range comparison result above the upper-limit threshold value causes an increment in the continuous detection count (ADRCCS[n]:RCOCD[2:0]) even if the range compare interrupt factor flag is in the set state (ADRCIF:RCINT[n] is "1").  
However, the out-of-range flag (ADRCOT:RCOOF[n]) retains its previous value because the range compare interrupt factor flag is in the set state (ADRCIF:RCINT[n] is "1").
9. The range compare interrupt factor flag is cleared (ADRCIF:RCINT[n] is "0") by the range compare interrupt factor flag clear (ADRCIFC:RCINT[n]C is "1").  
Also, the retained state of the out-of-range flag (ADRCOT:RCOOF[n]) is released.
10. The number of consecutive range comparison results that are outside the range reaches the continuous detection frequency specification value (ADRCCS[n]:RCOCD[2:0] is "0b011"). As a result, the range compare interrupt factor flag (ADRCIF:RCINT[n]) is set to "1".  
Also, the out-of-range flag (ADRCOT:RCOOF[n]) is set to the out-of-range state for the time that the range compare interrupt factor flag was set (ADRCIF:RCINT[n] is "1"). This out-of-range setting is retained until the range compare interrupt factor flag is cleared (ADRCIFC:RCINT[n]C is "1").
11. The continuous detection count (ADRCCS[n]:RCOCD[2:0]) is initialized to "0b000" when range comparison execution is disabled (ADRCCS[n]:RCOE is "0").  
Neither the range compare interrupt factor flag (ADRCIF:RCINT[n]) nor the out-of-range flag (ADRCOT:RCOOF[n]) is cleared by the disabling of range comparison execution (ADRCCS[n]:RCOE is "0").

### 3.3. Operation Timing Example

This section explains an operation timing example. (n = Corresponding activation channel 0 to 7)

Figure 3-14 Operation Timing Example



The A/D activation compare block outputs an activation request by using the selected activation factor from the A/D activation factor selection bits (ADTC[n]:STS[2:0]) as a trigger.

A conversion end signal is input at the end time of A/D conversion in units of activation channels.

The A/D activation compare block writes the conversion data to the respective data registers of the activation channel at the conversion end signal "1" time. At this time, the data error flag bit is not modified. If the range comparison function is enabled, each channel compares a range at its own separate time. Furthermore, if interrupt notification conditions are satisfied, it generates a range compare interrupt.

At the end time of conversion in all channels, an all-channel conversion end signal is input simultaneously with the final conversion end signal.



The A/D activation compare block receives the all-channel conversion end signal and turns off activation requests. If A/D conversion end interrupts are enabled in all channels, every channel generates an A/D conversion end interrupt at the same time. The range compare interrupt of the final channel is generated later than the A/D conversion end interrupts. If the data protection function has been configured, the data protection function is enabled. A/D data register (ADTCD[n]) reading is a condition for canceling this protection when the setting of the A/D data register protection clear selection bit (ADTC[n]:PRTS) is "1". Once the register is read, the protection is canceled. The data error flag bit becomes "0" simultaneously for all channels to indicate that the latest data is stored. Once the A/D data register (ADTCD[n]) is read, the bit becomes "1".

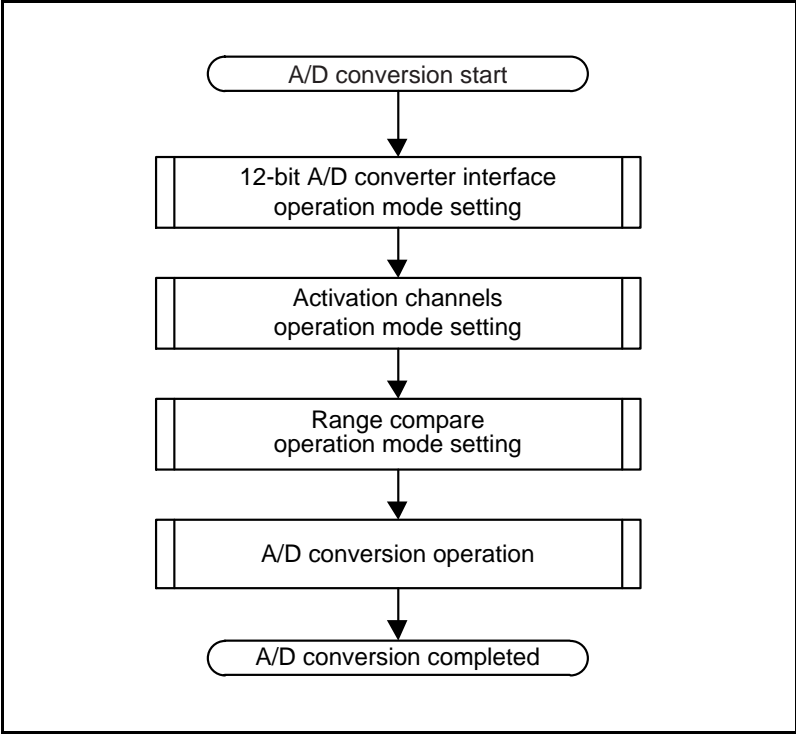
4. Setting Procedure Examples

This section shows examples of A/D activation compare setting procedures.

4.1. A/D Conversion Setting Procedure Examples

Figure 4-1 shows an example of the A/D conversion setting procedure.

Figure 4-1 A/D Conversion Setting Procedure Example



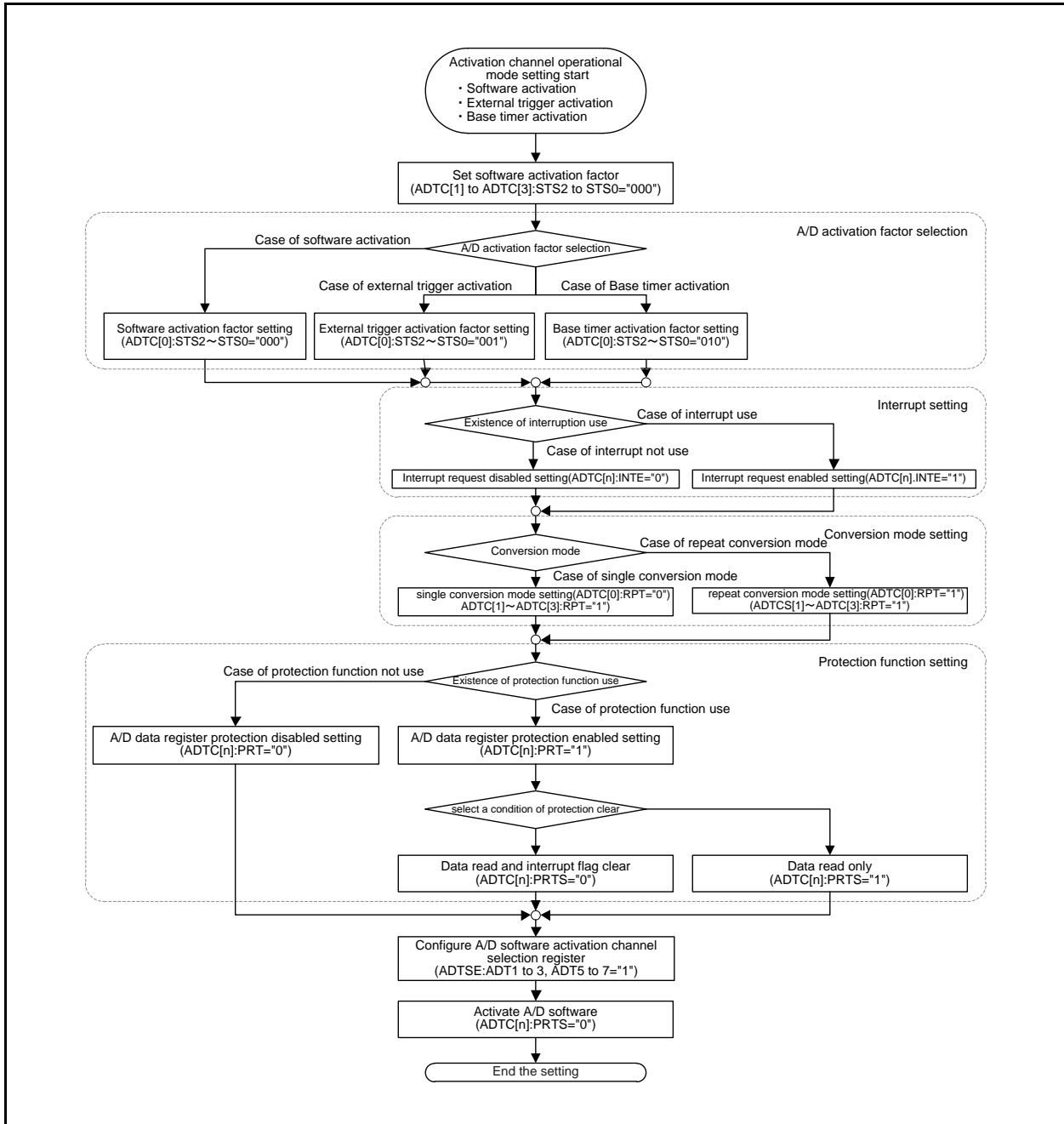
(1) 12-bit A/D Converter Interface Operation Mode Settings

For details on the operation mode settings of the 12-bit A/D converter interface, see the setting procedure example in the following "CHAPTER: 12-bit 4ch A/D Converter Interface".

## (2) Activation Channel Operation Mode Settings

### a) Setting Example for Software, External Trigger, and Base Timer Activation

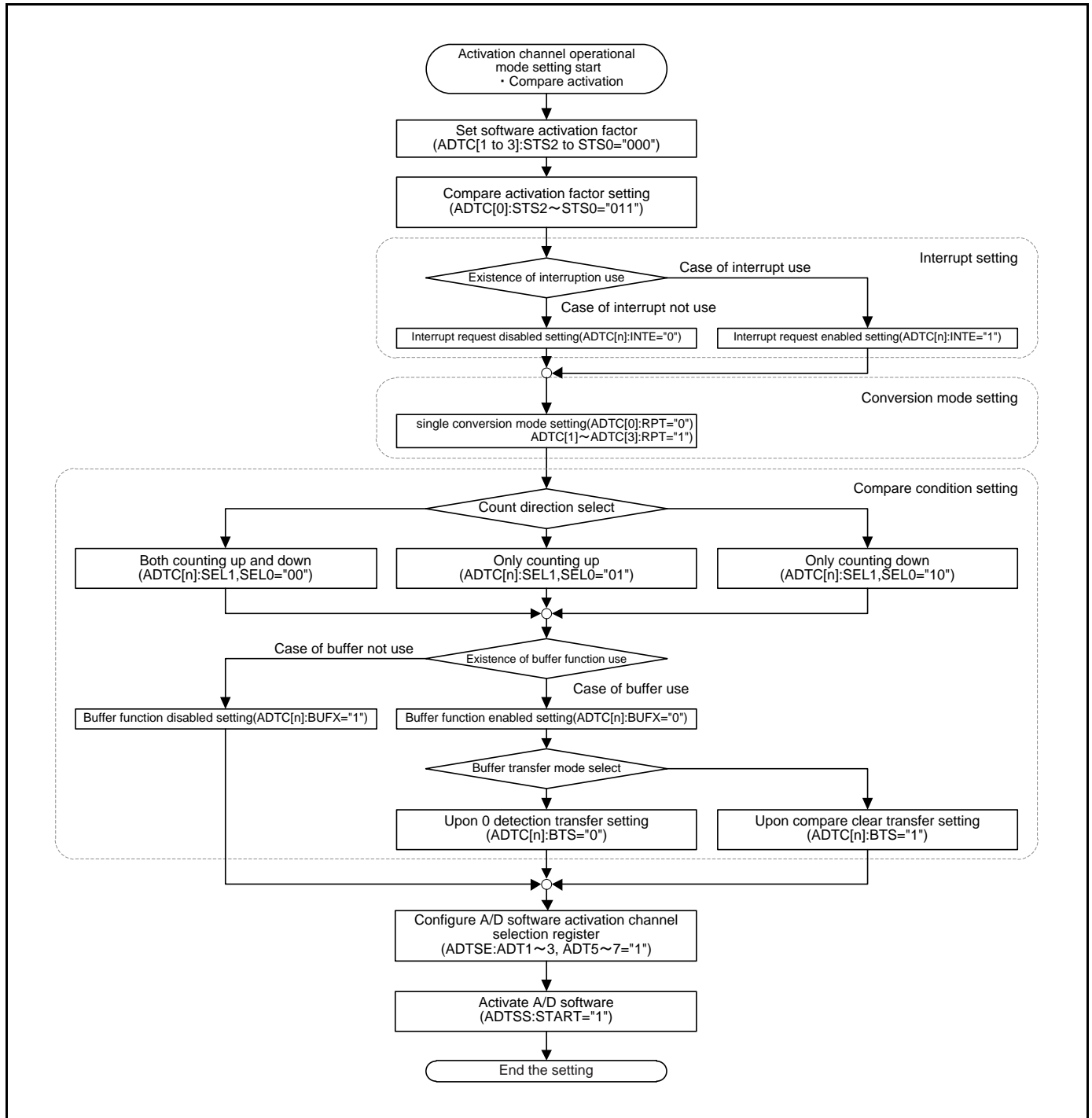
Figure 4-2 Setting Example for Software, External Trigger, and Base Timer Activation



- The setting contents described above can be set in random order.
- The following control bits hold no meaning except in compare match activation mode (ADTC[n]:STS[2:0] is "0b011"), so they need not be configured. (They are kept set at their initial values.)
  - Compare value buffer bits (ADCOMPB[n]:CMP[15:0])
  - Compare value bits (ADCOMP[n]:CMP[15:0])
  - Count direction selection bits (ADTC[n]:SEL[1:0])
  - Compare register buffer function control bit (ADTC[n]:BUFX)
  - Compare register buffer transfer control bit (ADTC[n]:BTS)

b) Setting Example for Compare Match Activation

Figure 4-3 Setting Example for Compare Match Activation

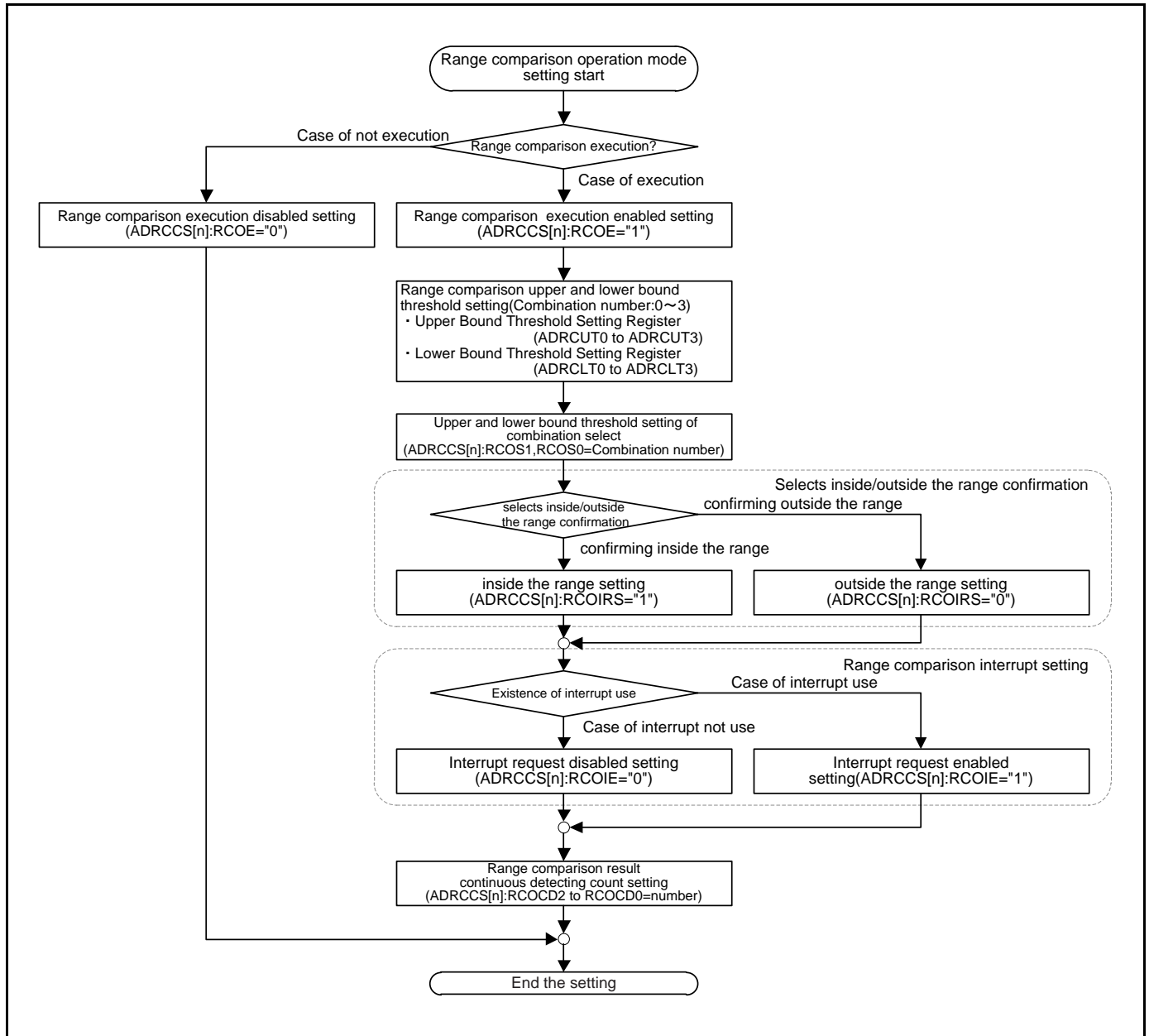


- The setting contents described above can be set in random order.
- The following control bits have the "don't care" setting in compare match activation mode (ADTC[n]:STS[2:0] is "0b011"), so they need not be configured. (They are kept set at their initial values.)
  - A/D data register protection enable bit (ADTC[n]:PRT)
  - A/D data register protection clear selection bit (ADTC[n]:PRTS)



### (3) Setting Example for Range Comparison Operation Mode

Figure 4-4 Setting Example for Range Comparison Operation Mode

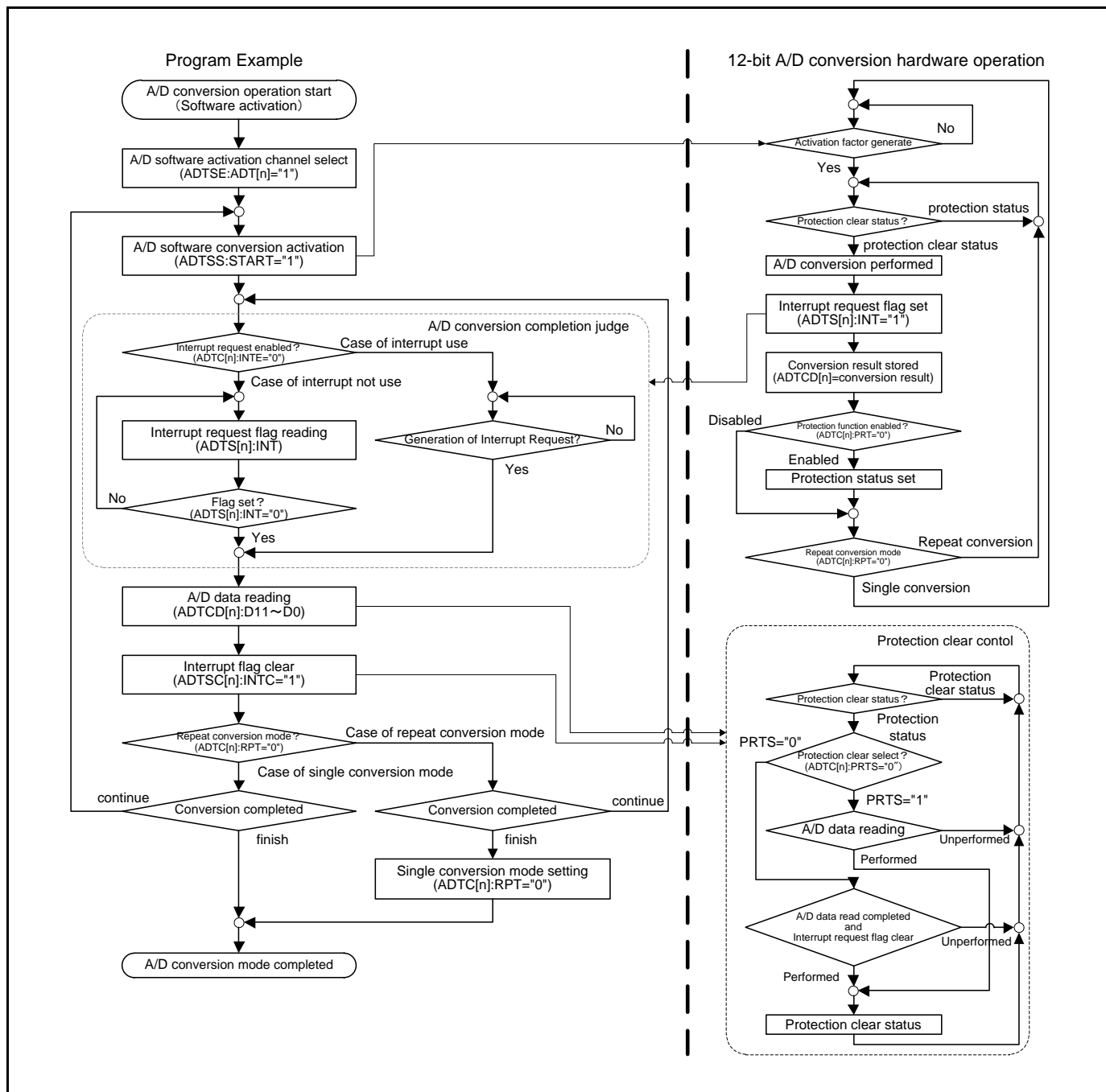


- The setting contents described above can be set in random order.
- There are 4 types of possible settings for the range comparison upper- and lower-limit threshold values.
- Each type from these 4 types of upper- and lower-limit threshold settings for range comparisons is a selected pair of upper- and lower-limit threshold settings.

#### (4) A/D Conversion Operation

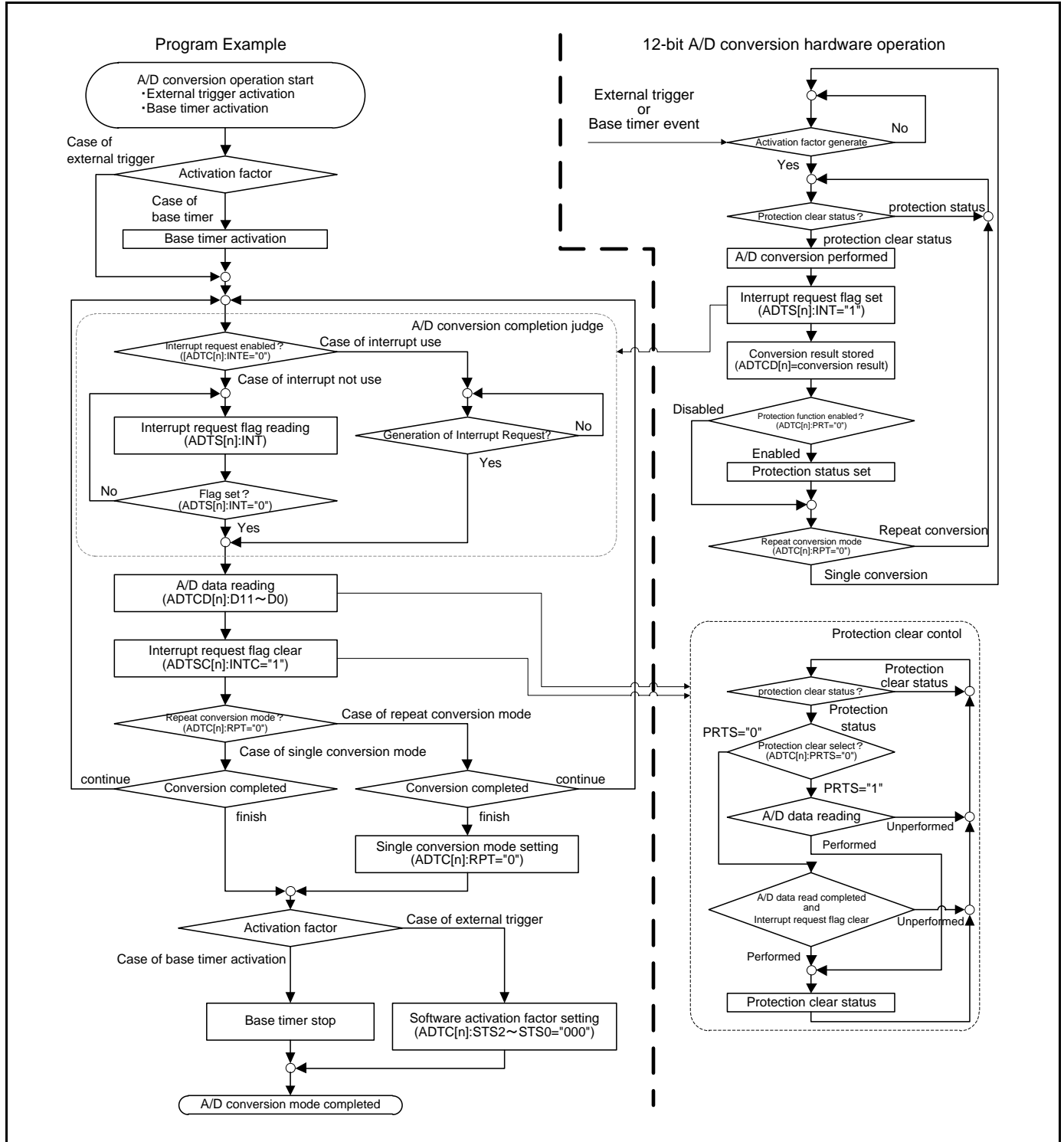
##### a) A/D Conversion Operation Example for Software Activation

Figure 4-5 A/D Conversion Operation Example for Software Activation



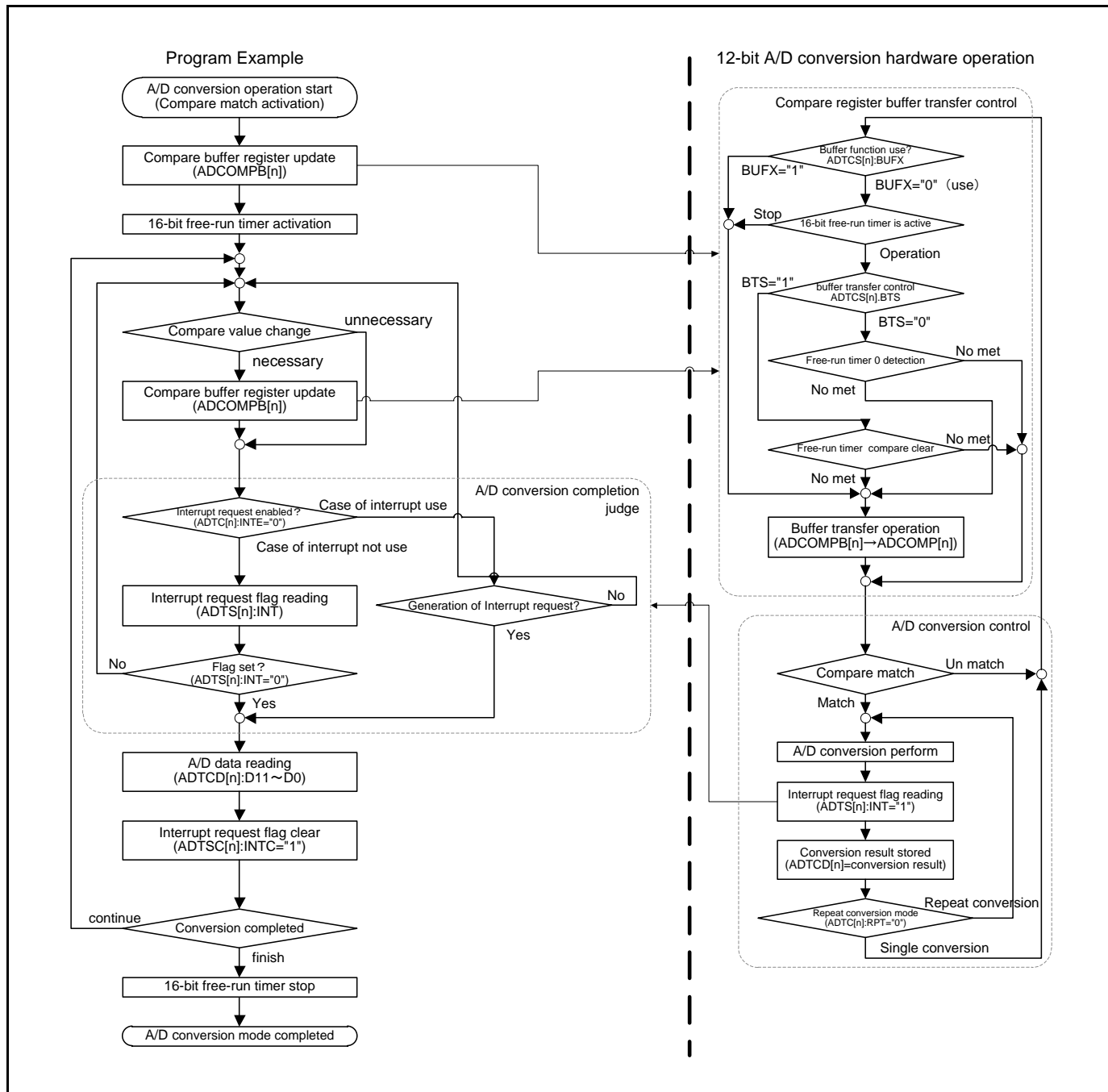
b) A/D Conversion Operation Example for External Trigger and Base Timer Activation

Figure 4-6 A/D Conversion Operation Example for External Trigger and Base Timer Activation



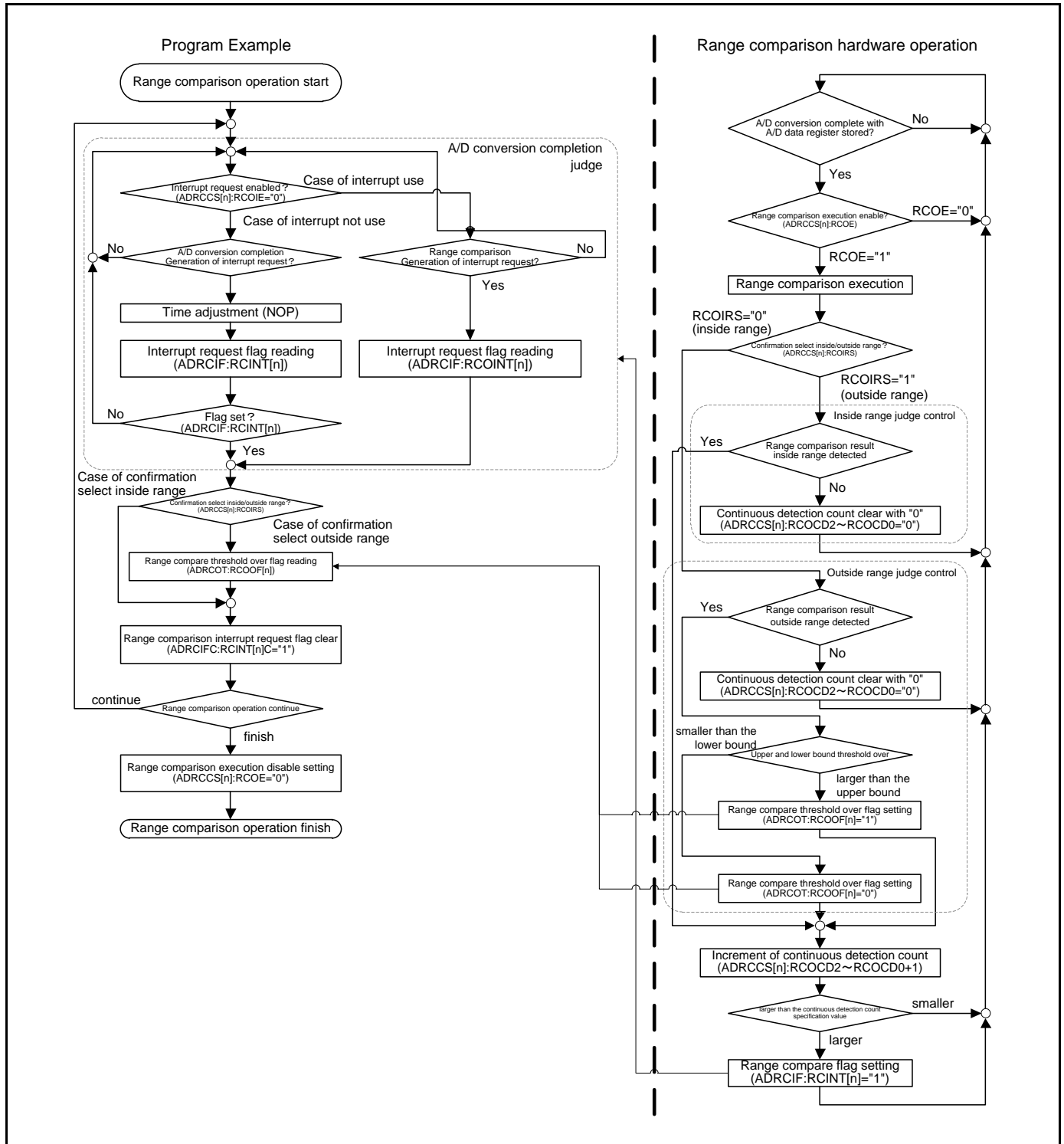
c) A/D Conversion Operation Example for Compare Match Activation

Figure 4-7 A/D Conversion Operation Example for Compare Match Activation



d) Range Comparison Operation Example

Figure 4-8 Range Comparison Operation Example



- The range comparison operation performed from a register comparison hardware operation begins at the time that conversion results are stored in the A/D data register from the end of A/D conversion.
- A range is compared in units of activation channels

## 5. Registers

This section lists the A/D activation compare registers.

The 4ch A/D activation compare register names begin with the prefix "ADC4SHxx\_". xx corresponds to the unit number.

**Table 5-1 List of A/D Activation Compare Registers**

Abbreviated Register Name	Register Name	See
ADTSS	A/D software activation register	5.1.1
ADTSE	A/D software activation channel selection register	5.1.2
ADCOMPB[n]	Compare buffer register[n]	5.1.3
ADCOMP[n]	Compare register[n]	5.1.3
ADTC[n]	A/D activation trigger control register[n]	5.1.4
ADTS[n]	A/D activation request/interrupt status register[n]	5.1.5
ADTSC[n]	A/D activation request/interrupt clear register[n]	5.1.6
ADTCD[n]	A/D data register[n]	5.1.7
ADRCUT0 to 3	Upper-limit threshold setting register 0 to 3	5.1.8
ADRCLT0 to 3	Lower-limit threshold setting register 0 to 3	5.1.9
ADRCES[n]	Range comparison control status register[n]	5.1.10
ADRCOT	Range comparison out-of-range flag register	5.1.11
ADRCIF	Range comparison flag register	5.1.12
ADRCIFC	Range comparison flag clear register	5.1.13
ADPRTF	Protected data state flag register	5.1.14

xx: Unit number (xx=00, 01)

n: Channel number (n=0 to 7)



## 5.1. A/D Activation Compare Registers

The A/D activation compare block has the following registers: A/D software activation register, A/D software activation channel selection register, compare buffer registers, compare registers, A/D activation trigger control registers, A/D activation request/interrupt status registers, A/D activation request/interrupt clear registers, A/D data registers, A/D activation trigger extended control registers, upper-limit threshold setting registers, lower-limit threshold setting registers, range comparison control status registers, range comparison flag clear register, range comparison out-of-range flag register, range comparison flag register, and protected data state flag register.

### 5.1.1. A/D Software Activation Register (ADTSS)

The A/D software activation register (ADTSS) is the register that makes A/D activation requests for the 12-bit A/D converter. The A/D software activation channel selection register (ADTSE) controls which channel to activate.

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved							START
ACCESS_TYPE	R0,W0							R0,W
PROT_TYPE	-							
INITIAL_VALUE	0000000							0

[bit7:1] Reserved: Reserved bits

[bit0] START: A/D conversion activation bit (software)

Value	Description
0	Do not activate A/D conversion.
1	Activate A/D conversion.

- This bit activates the A/D conversion operation with software.
- Writing "1" to the START bit activates A/D conversion. The A/D software activation channel selection register (ADTSE) controls which channel to activate.
- The START bit cannot restart conversion.
- If the value is read, it differs from the written value, and "0" is always read.

### 5.1.2. A/D Software Activation Channel Selection Register (ADTSE)

The A/D software activation channel selection register (ADTSE) is the register that selects the activation channel used for A/D activation requests.

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	ADT7	ADT6	ADT5	ADT4	ADT3	ADT2	ADT1	ADT0
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

#### [bit7:0] ADT7 to ADT0: Software activation channel selection bits

Value	Description
0	Disable software activation.
1	Enable software activation.

- If the ADTSE:ADT bit is "0", software activation is disabled.
- If the ADTSE:ADT bit is "1", software activation is enabled.





### 5.1.3. Compare Buffer Register[n] (ADCOMPB[n])/Compare Register[n] (ADCOMP[n]) (n=0 to 7)

Each compare buffer register (ADCOMPB[n]) is the 16-bit buffer register used by a compare register (ADCOMP[n]).

The A/D converter is activated when the compare register (ADCOMP[n]) matches the 16-bit free-run timer value.

The ADCOMPB[n] register and the ADCOMP[n] register have the same address.

#### (1) Compare Buffer Register (ADCOMPB[n], n=0 to 7)

BIT_OFFSET	15-0
BIT_NAME	CMP
ACCESS_TYPE	W
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000

#### [bit15:0] CMP[15:0]: Compare value buffer bits

Value	Description
	Compare value buffer

ADCOMPB[0], ADCOMPB[4]

- Each compare buffer register (ADCOMPB[n]) is the buffer register used by a compare register (ADCOMP[n]).
- If the buffer function is disabled (compare register buffer function control bit (ADTC[n]:BUFX) is "1") or the 16-bit free-run timer is stopped, the set value of the compare value buffer is immediately transferred to the compare register (ADCOMP[n]).
- If the buffer function is enabled (compare register buffer function control bit (ADTC[n]:BUFX) is "0"), the set value of the compare value buffer is transferred to the compare register (ADCOMP[n]) when a match with the compare clear register of the 16-bit free-run timer is found or when 0 is detected.

#### Notes:

- To access a compare buffer register, use halfword or word access instructions.
- Configure only ADCOMPB[0] and ADCOMPB[4] among these registers. Since the set values in the ADCOMPB[1] to ADCOMPB[3] and ADCOMPB[5] to ADCOMPB[7] registers are ignored, do not configure these registers.

**(2) Compare Register (ADCOMP[n], n=0 to 7)**

BIT_OFFSET	15-0
BIT_NAME	CMP
ACCESS_TYPE	R
PROT_TYPE	-
INITIAL_VALUE	00000000_00000000

ADCOMP[0], ADCOMP[4]

**[bit15:0] CMP[15:0]: Compare value bits**

Value	Description
	Compare Value

- The compare value in a compare register (ADCOMP[n]) is updated through the compare buffer register (ADCOMPB[n]).
- The compare register (ADCOMP) stores the compare value used for comparison with the count value of the 16-bit free-run timer. An A/D activation request is output when the 16-bit free-run timer matches the compare value.
- The compare value stored in the compare register is immediately compared with the 16-bit free-run timer.
- The compare match activation operation is not performed when the count direction selection bits (ADTC[n]:SEL[1:0]) are "0b11".

**Notes:**

- Use halfword or word access instructions for reading from a compare register.
- Since the set values for the ADCOMPB[1] to ADCOMPB[3] and ADCOMPB[5] to ADCOMPB[7] registers are ignored, only the compare values of ADCOMP[0] and ADCOMP[4] are updated.



#### 5.1.4. A/D Activation Trigger Control Register[n] (ADTC[n]) (n=0 to 7)

Each A/D activation trigger control register (ADTC[n]) enables/disables interrupt requests, selects an activation factor, selects a conversion mode, controls protection function, selects a compare value for use in compare operations, and controls a compare value buffer.

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved		INTE	Reserved		RPT	PRT	PRTS
ACCESS_TYPE	R0,W0		R/W	R0,W0		R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	00		0	00		0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	SEL		BUFX	BTS	Reserved	STS		
ACCESS_TYPE	R/W		R/W	R/W	R0,W0	R/W		
PROT_TYPE	-							
INITIAL_VALUE	00		1	0	0	000		

[bit15:14] Reserved: Reserved bits

[bit13] INTE: Interrupt request enable bit

Value	Description
0	Disable interrupt request output.
1	Enable interrupt request output.

- This bit controls the enabling/disabling of interrupt request output to the CPU.
- An interrupt request is generated when both the ADTC[n]:INTE bit and the interrupt request flag bit (ADTS[n]:INT) are "1".

[bit12:11] Reserved: Reserved bits

[bit10] RPT: Repeat conversion selection bit

Value	Description
0	Single conversion mode
1	Repeat conversion mode

- This bit sets the A/D conversion mode.
- If the ADTC[n]:RPT bit is set to "0", the mode is single conversion mode. A 1-time activation factor causes a 1-time A/D conversion request. The A/D conversion is done once.
- If the ADTC[n]:RPT bit is set to "1", the mode is repeat conversion mode. A 1-time activation factor causes an A/D conversion request to be made continuously. A/D conversion is performed repeatedly until single conversion mode is set.

**Note:**

- Always set repeat conversion mode ("1") in the RPT bit in ADTC[1] to ADTC[3] and ADTC[5] to ADTC[7].

**[bit9] PRT: A/D data register protection enable bit**

Value	Description
0	Disable protection.
1	Enable protection.

- If the ADTC[n]:PRT bit is set to "1", the A/D data register is write-protected. The protection function runs for factors other than compare match activation (ADTC[n]:STS[2:0] is "0b011").
- After conversion data is stored in the A/D data register, subsequent activation requests are masked and the A/D data register is write-protected until an occurrence of the factor that is set by the A/D data register protection factor selection bit (ADTC[n]:PRTS).

**Note:**

- Set the A/D data register protection enable bit (ADTC[n]:PRT) before the A/D conversion operation. The setting for the A/D data register protection enable bit (ADTC[n]:PRT) cannot be modified during an A/D conversion request (ADTS[n]:BUSY is "1") or while the A/D data register is protected.

**[bit8] PRTS: A/D data register protection clear selection bit**

Value	Description
0	Data reading and interrupt flag clearing
1	Data reading

- This bit selects the conditions for canceling the masking of activation requests when A/D data register protection function is enabled (ADTC[n]:PRT is "1").
- If the PRTS bit is set to "0", the conditions (in random order) for canceling the protection are the reading of an A/D data register (ADTCD[n]) and the clearing of an interrupt request flag bit (ADTS[n]:INT).
- If the PRTS bit is set to "1", the condition for canceling the protection is the reading of an A/D data register (ADTCD[n]).

**Note:**

- Set the A/D data register protection clear selection bit (ADTC[n]:PRTS) before the A/D conversion operation. The setting for the A/D data register protection clear selection bit (ADTC[n]:PRTS) cannot be modified during an A/D conversion request (ADTS[n]:BUSY is "1") or while the A/D data register is protected.

**[bit7:6] SEL[1:0]: Count direction selection bits**

Value	Description
00	When counting in either direction (up/down)
01	Only when counting up
10	Only when counting down
11	Compare disabled

- These bits select the compare match conditions for the 16-bit free-run timer.
- If the ADTC[n]:SEL[1:0] bits are set to "0b00", the compare match operation is performed any time that the 16-bit free-run timer is counting up/down.
- If the ADTC[n]:SEL[1:0] bits are set to "0b01", the compare match activation operation is performed only when the 16-bit free-run timer is counting up.
- If the ADTC[n]:SEL[1:0] bits are set to "0b10", the compare match activation operation is performed only when the 16-bit free-run timer is counting down.
- If the ADTC[n]:SEL[1:0] bits are set to "0b11", the compare operation is not performed.
- The compare operation is not performed while the selected 16-bit free-run timer is stopped.

**Note:**

- Setting the count direction selection bits (ADTC[n]:SEL[1:0]) to "0b10" is prohibited when the 16-bit free-run timer is in the mode of counting up.

**[bit5] BUFX: Compare register buffer function control bit**

Value	Description
0	Enable the buffer function.
1	Disable the buffer function.

- If the ADTC[n]:BUFX bit is set to "0", the buffer function is enabled.
- If the ADTC[n]:BUFX bit is set to "1", the buffer function is disabled.

**[bit4] BTS: Compare register buffer transfer control bit**

Value	Description
0	0 detection time
1	Compare clear time

- This bit sets the transfer condition when the compare register buffer function is enabled (ADTC[n]:BUFX is "0").
- The ADTC[n]:BTS setting is enabled when the compare register buffer function is enabled (ADTC[n]:BUFX is "0").
- If the ADTC[n]:BTS bit is set to "0", the compare value of a compare buffer register (ADCOMPB[n]) is transferred to a compare register (ADCOMP[n]) when 0 is detected in the 16-bit free-run timer.
- If the ADTC[n]:BTS bit is set to "1", the compare value of a compare buffer register (ADCOMPB[n]) is transferred to a compare register (ADCOMP[n]) when the compare clear register matches the 16-bit free-run timer.

**Note:**

- When setting the compare register buffer transfer control bit (ADTC[n]:BTS), be sure that the 16-bit free-run timer has stopped.

**[bit3] Reserved: Reserved bit**

**[bit2:0] STS[2:0]: A/D activation factor selection bits**

Value	Description
000	Software activation
001	External trigger activation (falling edge)
010	Base timer activation (rising edge)
011	Compare match activation
100	Setting prohibited
101	
110	
111	

The ADTC[n]:STS[2:0] bits select an activation factor for A/D conversion.

**Notes:**

- The A/D activation factor selection bits (ADTC[n]:STS[2:0]) are updated at the same time that they are overwritten. Update the A/D activation factor selection bits (ADTC[n]:STS[2:0]) while the currently selected activation factor and the new activation factor are inactive and while no A/D conversion request is being processed (ADTS[n]:BUSY is "1").
- To not perform A/D activation requests, set the A/D activation factor selection bits (ADTC[n]:STS[2:0]) for software activation ("0b000"), and disable software activation (ADTSE:ADT[n] is "0") in the corresponding bit (activation channel) in the A/D software activation channel selection register (ADTSE).
- When setting the A/D activation factor selection bits (ADTC[n]:STS[2:0]), be sure that the 16-bit free-run timer has stopped.
- Always set software activation ("0b000") in the STS[2:0] bits in ADTC[1] to ADTC[3] and ADTC[5] to ADTC[7].



### 5.1.5. A/D Activation Request/Interrupt Status Register[n] (ADTS[n]) (n=0 to 7)

Each A/D activation request/interrupt status register (ADTS[n]) checks A/D activation requests and interrupt request status.

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	BUSY	INT	Reserved					
ACCESS_TYPE	R,WX	R,WX	R0,W0					
PROT_TYPE	-							
INITIAL_VALUE	0	0	000000					

BITS	7	6	5	4	3	2	1	0
BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved							
ACCESS_TYPE	R0,W0							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

#### [bit15] BUSY: A/D activation request busy bit

Value	Description
0	A/D activation not yet requested
1	Ongoing A/D activation request or ongoing conversion for the set number of channels

(ADTS[0], ADTS[4])

- This bit indicates the operation of an A/D activation request or conversion.
- In the ADTS[n]:BUSY bit, "0" indicates that A/D conversion has not yet been requested, and "1" indicates an ongoing A/D conversion request or ongoing conversion for the set number of channels.
- Writing "1" to the A/D activation request forced termination bit (ADTSC[n]:BUSYC) when the ADTS[n]:BUSY bit is "1" clears the ADTS[n]:BUSY bit to "0".
- Writing the final conversion data for the set number of channels to the A/D data register (ADTCD[0]/ADTCD[4]) clears the ADTS[n]:BUSY bit to "0".

Value	Description
0	A/D activation not yet requested
1	Ongoing A/D activation request or ongoing conversion

(ADTS[1] to ADTS[3], ADTS[5] to ADTS[7])

- This bit indicates the operation of an A/D activation request or conversion.
- In the ADTS[n]:BUSY bit, "0" indicates that A/D conversion has not yet been requested, and "1" indicates an ongoing A/D conversion request or ongoing conversion.
- Writing "1" to the A/D activation request forced termination bit (ADTSC[n]:BUSYC) when the ADTS[n]:BUSY bit is "1" clears the ADTS[n]:BUSY bit to "0".

#### Note:

- The specification when the ADTS[n]:BUSY bit is "1" for ADTS[0] and ADTS[4] is different from that for ADTS[1] to ADTS[3] and ADTS[5] to ADTS[7].

**[bit14] INT: Interrupt request flag bit**

Value	Description
0	Do not terminate A/D conversion.
1	Terminate A/D conversion.

- The ADTS[n]:INT bit is set to "1" when the final conversion data for the number of channels set by AD conversion is set in the A/D data register (ADTCD[n]).
- An A/D conversion end interrupt request is generated when both the ADTS[n]:INT bit and the interrupt request enable bit (ADTC[n]:INTE) are "1".
- Writing "1" to the A/D conversion end interrupt software clear bit (ADTSC[n]:INTC) when the ADTS[n]:INT bit is "1" clears the ADTS[n]:INT bit to "0".
- The ADTS[n]:INT bit is cleared when the externally input A/D conversion end interrupt clear signal is "H".

**Note:**

- *If a software clear (ADTSC[n]:INTC of "1" is written) or clear by the externally input A/D conversion end interrupt clear signal ("H") occurs at the same time as a hardware set, the hardware set has priority.*

**[bit13:0] Reserved: Reserved bits**





### 5.1.6. A/D Activation Request/Interrupt Clear Register[n] (ADTSC[n]) (n=0 to 7)

Each A/D activation request/interrupt clear register (ADTSC[n]) is used to forcibly terminate A/D activation requests and to clear the A/D conversion end interrupt request flag.

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	BUSYC	INTC	Reserved					
ACCESS_TYPE	R0,W	R0,W	R0,W0					
PROT_TYPE	-							
INITIAL_VALUE	0	0	000000					

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved							
ACCESS_TYPE	R0,W0							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

#### [bit15] BUSYC: A/D activation request clear bit

Value	Description
0	Do not change or have any other effect.
1	Forcibly terminate an A/D activation request.

- This bit forcibly terminates an A/D activation request or conversion.
- Writing "1" to the ADTSC[n]:BUSYC bit forcibly terminates the A/D activation request or conversion.
- If the value is read, it differs from the written value, and "0" is always read.

#### [bit14] INTC: A/D conversion end interrupt software clear bit

Value	Description
0	Do not change or have any other effect.
1	Perform a software clear of an A/D conversion end interrupt.

- This bit is the software clear bit for A/D conversion end interrupts.
- Writing "1" to the ADTSC[n]:INTC bit clears the interrupt request flag (ADTS:INT).
- If the value is read, it differs from the written value, and "0" is always read.

#### [bit13:0] Reserved: Reserved bits

### 5.1.7. A/D Data Register[n] (ADTCD[n]) (n=0 to 7)

Each A/D data register (ADTCD[n]) is a register for storing A/D conversion results.

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	ERR	ERRST	Reserved		D[11:8]			
ACCESS_TYPE	R,WX	R,WX	R0,W0		R,WX			
PROT_TYPE	-							
INITIAL_VALUE	1	0	00		0000			

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	D[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

#### [bit15] ERR: Conversion data error flag bit

Value	Description
0	Conversion data is normal.
1	Conversion data is abnormal.

- This bit indicates the existence of an error regarding A/D conversion data. The details of the error can be checked using the conversion data error status bit (ADTCD[n]:ERRST) when the ADTCD[n]:ERR bit is "1".
- For details on the ADTCD[n]:ERR bit operation, see Section "3.2.9. A/D Conversion Data (n=0 to 7)".
- If the A/D data register protection function is enabled (ADTC[n]:PRT is "1") when the factor is other than compare match activation (ADTC[n]:STS[2:0] is "0b011"), "0" is read.

#### [bit14] ERRST: Conversion data error status bit (enabled only if ERR="1")

Value	Description
0	The conversion data is old results.
1	The conversion data has been overwritten by new data.

- This bit is a flag indicating the details of an A/D conversion data error when the ADTCD[n]:ERR bit is "1".
- The ADTCD[n]:ERR bit of "1" and ADTCD[n]:ERRST bit of "0" indicate that the conversion results read by the CPU are old.
- The ADTCD[n]:ERR bit of "1" and ADTCD[n]:ERRST bit of "1" indicate that the conversion results read by the CPU was overwritten by new conversion results before the CPU finished reading the old conversion results. Consequently, the old conversion result data was lost.
- For details on the ADTCD[n]:ERRST bit operation, see Section "3.2.9. A/D Conversion Data (n=0 to 7)".
- If the A/D data register protection function is enabled (ADTC[n]:PRT is "1") when the factor is other than compare match activation (ADTC[n]:STS[2:0] is "0b011"), "0" is read.

#### [bit13:12] Reserved: Reserved bits



**[bit11:0] D[11:0]: A/D data bits**

Value	Description
	Conversion data

- The A/D data bits (ADTCD[n]:D[11:0]) store A/D conversion results. These bits are overwritten each time that conversion ends.
- Normally, the bits store the final conversion value.

### 5.1.8. Upper-limit Threshold Setting Registers 0 to 3 (ADRCUT0 to 3)

Each upper-limit threshold setting register (ADRCUT) sets the upper-limit threshold value used in range comparisons. The register can have 4 types of settings for the upper-limit threshold value.

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved				C[11:8]			
ACCESS_TYPE	R0,W0				R/W			
PROT_TYPE	-							
INITIAL_VALUE	0000				0000			

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	C[7:0]							
ACCESS_TYPE	R/W							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

[bit15:12] Reserved: Reserved bits

[bit11:0] C[11:0]: Upper-limit threshold bits

Value	Description
	Upper-limit threshold value

These bits set the upper-limit threshold value used in range comparisons.

**Note:**

- The upper-limit threshold bits (ADRCUT[m]:C[11:0]) cannot be modified during an A/D conversion request (ADTS[n]:BUSY is "1").



### 5.1.9. Lower-limit Threshold Setting Registers 0 to 3 (ADRCLT0 to 3)

Each lower-limit threshold setting register (ADRCLT) sets the lower-limit threshold value used in range comparisons. The register can have 4 types of settings for the lower-limit threshold value.

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved				C[11:8]			
ACCESS_TYPE	R0,W0				R/W			
PROT_TYPE	-							
INITIAL_VALUE	0000				0000			

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	C[7:0]							
ACCESS_TYPE	R/W							
PROT_TYPE	-							
INITIAL_VALUE	00000000							

[bit15:12] Reserved: Reserved bits

[bit11:0] C[11:0]: Lower-limit threshold bits

Value	Description
	Lower-limit threshold value

These bits set the lower-limit threshold value used in range comparisons.

**Note:**

- The lower-limit threshold bits (ADRCLT[m]:C[11:0]) cannot be modified during an A/D conversion request (ADTS[n]:BUSY is "1").

### 5.1.10. Range Comparison Control Status Register[n] (ADRCSS[n]) (n=0 to 7)

Each range comparison control status register (ADRCSS[n]) checks the continuous detection frequency specification and count, selects inside/outside-the-range confirmation, enables/disables range compare interrupt requests, enables/disables range comparison execution, and selects the upper- and lower-limit threshold values.

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	RCOCD			RCOIRS	RCOIE	RCOE	RCOTS	
ACCESS_TYPE	R,W			R/W	R/W	R/W	R/W	
PROT_TYPE	-							
INITIAL_VALUE	000			0	0	0	00	

#### [bit7:5] RCOCD[2:0]: Continuous detection frequency specification and count display bits

Value			Description	
			Read	Write
0	0	0	Continuous detection state: 0 times (initial value)	Setting prohibited
0	0	1	Continuous detection state: 1 time	Continuous detection specified 1 time (initial value)
0	1	0	Continuous detection state: 2 times	Continuous detection specified 2 times
0	1	1	Continuous detection state: 3 times	Continuous detection specified 3 times
1	0	0	Continuous detection state: 4 times	Continuous detection specified 4 times
1	0	1	Continuous detection state: 5 times	Continuous detection specified 5 times
1	1	0	Continuous detection state: 6 times	Continuous detection specified 6 times
1	1	1	Continuous detection state: 7 times	Continuous detection specified 7 times

- These bits specify the continuous detection frequency and display the continuous detection count for a range comparison result.
- The range compare interrupt factor flag bit (ADRCIF:RCINT[n]) of the corresponding activation channel is set to "1" when a range comparison result reaches the specified value for the continuous count. Also, the continuous detection state stops at the specified value for the continuous count.
- At the read time, the continuous detection state is read, rather than the continuous detection frequency specification that is set for the write time.

#### Notes:

- The continuous detection frequency specification and count display bits (ADRCSS[n]:RCOCD[2:0]) cannot be modified during an A/D conversion request (ADTS[n]:BUSY is "1").
- The continuous detection frequency specification and count display bits (ADRCSS[n]:RCOCD[2:0]) cannot be modified while range comparison execution is enabled (ADRCSS[n]:RCOE is "1").
- The continuous detection frequency specification and count display bits (ADRCSS[n]:RCOCD[2:0]) cannot be set to "0b000".



**[bit4] RCOIRS: Outside/Inside range confirmation selection bit**

Value	Description
0	Confirm that the result is outside the range.
1	Confirm that the result is inside the range.

- This bit selects outside the range or inside the range as a range comparison condition for A/D data bits (ADTCD[n]:D[11:0]). The range corresponds to the upper-limit threshold bits (ADRCUT:C[11:0]) and lower-limit threshold bits (ADRCLT:C[11:0]) selected by the upper- and lower-limit threshold selection bits (ADRCCS[n]:RCOTS[1:0]).
- For outside-the-range confirmation (ADRCCS[n]:RCOIRS is "0"), the range comparison condition is as follows.  
A/D data bits (ADTCD[n]:D[11:0]) > Upper-limit threshold bits (ADRCUT:C[11:0])  
or  
A/D data bits (ADTCD[n]:D[11:0]) < Lower-limit threshold bits (ADRCLT:C[11:0])
- For inside-the-range confirmation (ADRCCS[n]:RCOIRS is "1"), the range comparison condition is as follows.  
A/D data bits (ADTCD[n]:D[11:0]) ≤ Upper-limit threshold bits (ADRCUT:C[11:0])  
and  
A/D data bits (ADTCD[n]:D[11:0]) ≥ Lower-limit threshold bits (ADRCLT:C[11:0])
- For a range comparison detection with outside-the-range confirmation (ADRCCS[n]:RCOIRS is "0"), the out-of-range flag bit (ADRCOT:RCOOF) enables checking of whether the result is above the upper-limit threshold value or below the lower-limit threshold value.

**Note:**

- The outside/inside range confirmation selection bit (ADRCCS[n]:RCOIRS) cannot be modified during an A/D conversion request (ADTS[n]:BUSY is "1").

**[bit3] RCOIE: Range compare interrupt request enable bit**

Value	Description
0	Disable range compare interrupts.
1	Enable range compare interrupts.

If the range compare interrupt factor flag bit (ADRCIF:RCINT[n]) of the corresponding activation channel is set to "1" and range compare interrupt requests are enabled (ADRCCS[n]:RCOIE is "1"), an interrupt request is generated.

**[bit2] RCOE: Range comparison execution enable bit**

Value	Description
0	Disable range comparison execution.
1	Enable range comparison execution.

- This bit selects whether to enable or disable range comparison execution.
- Range comparison execution is disabled when the range comparison execution enable bit (ADRCCS[n]:RCOE) is "0". Also, the continuous detection count is initialized to "0b000".
- The range comparison execution is enabled when the range comparison execution enable bit (ADRCCS[n]:RCOE) is "1".

**[bit1:0] RCOTS[1:0]: Upper- and lower-limit threshold selection bits**

These bits select 1 combination from the 4 types available with upper-limit threshold setting registers 0 to 3 (ADRCUT0 to 3) and lower-limit threshold setting registers 0 to 3 (ADRCLT0 to 3).

Value		Description
0	0	Upper limit threshold setting register 0/lower limit threshold setting register 0 selected
0	1	Upper limit threshold setting register 1/lower limit threshold setting register 1 selected
1	0	Upper limit threshold setting register 2/lower limit threshold setting register 2 selected
1	1	Upper limit threshold setting register 3/lower limit threshold setting register 3 selected

**Note:**

- The upper- and lower-limit threshold selection bits (ADRCSS[n]:RCOTS[1:0]) cannot be modified during an A/D conversion request (ADTS[n]:BUSY is "1").





### 5.1.11. Range Comparison Out-of-range Flag Register (ADRCOT)

The range comparison out-of-range flag register (ADRCOT) indicates whether the result of a range comparison with the setting of outside the range is over the upper-limit threshold value or below the lower-limit threshold value.

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	RCOOF7	RCOOF6	RCOOF5	RCOOF4	RCOOF3	RCOOF2	RCOOF1	RCOOF0
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

#### [bit7:0] RCOOF7 to RCOOF0: Out-of-range flag bits

Value	Description
0	Below lower-limit threshold value (A/D data < Lower-limit threshold bits)
1	Above upper-limit threshold value (A/D data > Upper-limit threshold bits)

- For outside-the-range confirmation (ADRCSS[n]:RCOIRS is "0"), these bit indicate whether the range comparison result is larger than the upper-limit threshold setting register (ADRCOT:RCOOF is "1") or smaller than the lower-limit threshold setting register (ADRCOT:RCOOF is "0").
- With outside-the-range confirmation (ADRCSS[n]:RCOIRS is "0"), if the range comparison result is inside the range, the out-of-range flag bit retains its previous value.
- Even with outside-the-range confirmation (ADRCSS[n]:RCOIRS is "0") where the range comparison result has been detected as outside the range, if the range compare interrupt factor flag bit (ADRCIF:RCINT[n]) of the corresponding activation channel is set to "1", the out-of-range flag bit (ADRCOT:RCOOF) is not updated and retains its previous value.
- For inside-the-range confirmation (ADRCSS[n]:RCOIRS is "1"), the out-of-range flag bit holds no meaning. (The previous value is retained.)

### 5.1.12. Range Comparison Flag Register (ADRCIF)

The range comparison flag register (ADRCIF) indicates an interrupt factor resulting from the continuous detection of a range comparison result.

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	RCINT7	RCINT6	RCINT5	RCINT4	RCINT3	RCINT2	RCINT1	RCINT0
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

#### [bit7:0] RCINT7 to RCINT0: Range compare interrupt factor flag bits

Value	Description
0	Range compare interrupt factor clear state
1	Interrupt factor resulting from the continuous detection of a range comparison result

- The continuous detection of a range comparison result in the corresponding activation channel sets the ADRCIF:RCINT[n] bit to "1".
- A range compare interrupt request is generated when both the ADRCIF:RCINT[n] bit of the corresponding activation channel and the range compare interrupt request enable bit (ADRCSS[n]:RCOIE) are "1".

**Note:**

- If a software clear (ADRCIFC:RCINT[n]C of "1" is written) occurs at the same time as a hardware set, the hardware set has priority.



### 5.1.13. Range Comparison Flag Clear Register (ADRCIFC)

The range comparison flag clear register (ADRCIFC) clears the interrupt factor resulting from the continuous detection of a range comparison result.

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	RCINT7C	RCINT6C	RCINT5C	RCINT4C	RCINT3C	RCINT2C	RCINT1C	RCINT0C
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

#### [bit7:0] RCINT7C to RCINT0C: Range compare interrupt factor flag clear bits

Value	Description
0	Do not change or have any other effect.
1	Clear the range compare interrupt factor flag.

- These bits are the clear bits of the range compare interrupt factor flag.
- Writing "1" to the ADRCIFC:RCINT[n]C bit clears (ADRCIF:RCINT[n]).
- If the value is read, it differs from the written value, and "0" is always read.

#### 5.1.14. Protected Data State Flag Register (ADPRTF)

The protected data state flag register (ADPRTF) indicates the protection state of each activation channel A/D data register (ADTCD[n]).

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	PRTF7	PRTF6	PRTF5	PRTF4	PRTF3	PRTF2	PRTF1	PRTF0
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

##### [bit7:0] PRTF7 to PRTF0: Protected data state flag bits

These bits indicate the data protected state of the A/D data register (ADTCD[n]) of each activation channel.

Value	Description
0	Data is not protected.
1	Data is protected.



## 6. Precautions for Using

This section explains cautions on A/D activation compare usage.

### Notes on A/D Activation Compare Usage (n=0 to 7)

#### a) A/D Activation

Before starting A/D conversion, configure activation channel 0 or 4 as described below.

- Be sure to set software activation and repeat conversion mode for activation channels 1 to 3 and 5 to 7 by using the A/D activation trigger control registers (ADTC[1] to ADTC[3] and ADTC[5] to ADTC[7]).
- For activation channels 1 to 3 and 5 to 7, enable software activation with the A/D software activation channel selection register (ADTSE), and set the activation of A/D conversion with the A/D software activation register (ADTSS).
- (Settings when using activation channels 1 to 3)
- To select 2 channels: Set the ADTSE:ADT1 bit to "1".
- To select 3 channels: Set the ADTSE:ADT1 and ADT2 bits to "1".
- To select 4 channels: Set the ADTSE:ADT1, ADT2, and ADT3 bits to "1".

#### b) A/D Data Register Protection Settings (n=0 to 7)

Set the A/D data register protection enable bit (ADTC[n]:PRT) and the A/D data register protection clear selection bit (ADTC[n]:PRTS) before the A/D conversion operation. Neither the ADTC[n]:PRT nor ADTC[n]:PRTS bit setting can be modified during an A/D conversion request (ADTS[n]:BUSY is "1") or while the A/D data register is protected.

To release the protection function of an A/D data register, perform the following operation after A/D conversion has stopped (ADTS[n]:BUSY is "0"): cancel the set protection in the A/D data register protection clear selection bit (ADTC[n]:PRTS). Alternatively, disable the protection function from the A/D data register protection enable bit (ADTC[n]:PRT is "0").

Suppose the ADTC[n]:PRTS bit was modified while the A/D data register was in the protected state. To release the protection on the A/D data register after modifying the ADTC[n]:PRTS bit, perform the operation that cancels the set protection in the ADTC[n]:PRTS bit (clear by reading the A/D data register and writing "1" to the A/D conversion end interrupt software clear bit). For example, suppose ADTC[n]:PRT is "1" and ADTC[n]:PRTS is "1" with the A/D data register in the protected state. Then, if the A/D conversion end interrupt request flag bit is cleared and ADTC[n]:PRTS is changed to "0", canceling the protection requires that the A/D data register be read and "1" be written to the A/D conversion end interrupt software clear bit to clear it a second time.

#### c) A/D Activation Request Busy Bit (ADTS[n]:BUSY) (n=0 to 7)

For the A/D activation request busy bit (ADTS[n]:BUSY), the specification when the ADTS[n]:BUSY bit is "1" for ADTS[0] and ADTS[4] is different from that for ADTS[1] to ADTS[3] and ADTS[5] to ADTS[7] as follows.

- ADTS[0], ADTS[4]: Ongoing A/D activation request or ongoing conversion for the set number of channels
- ADTS[1] to ADTS[3], ADTS[5] to ADTS[7]: Ongoing A/D activation request or ongoing conversion in a single channel

#### d) Interrupt Request Flag Bit (ADTS[n]:INT) (n=0 to 7)

If a software clear (ADTSC[n]:INTC of "1" is written) or clear by the externally input A/D conversion end interrupt clear signal ("H") occurs at the same time as a hardware set, the hardware set has priority.

#### e) Repeat Conversion Selection Bit (ADTC[n]:RPT) (n=0 to 7)

Always set repeat conversion mode ("1") in the ADTC[n]:RPT bit in ADTC[1] to ADTC[3] and ADTC[5] to ADTC[7].

**f) Count Direction Selection Bit Setting (ADTC[n]:SEL[1:0]) (n=0 to 7)**

Setting the count direction selection bit for counting down only (ADTC[n]:SEL[1:0] is "0b10") is prohibited when the 16-bit free-run timer is in the mode of counting up.

**g) Compare Register Buffer Transfer Control Bit (ADTC[n]:BTS) (n=0 to 7)**

When setting the compare register buffer transfer control bit (ADTC[n]:BTS), be sure that the 16-bit free-run timer has stopped.

**h) A/D Activation Factor Selection Bits (ADTC[n]:STS[2:0]) (n=0 to 7)**

The A/D activation factor selection bits (ADTC[n]:STS[2:0]) are updated at the same time that they are overwritten. Update the A/D activation factor selection bits (ADTC[n]:STS[2:0]) while the currently selected activation factor and the new activation factor are inactive and while no A/D conversion request is being processed (ADTS[n]:BUSY is "1").

To not perform A/D activation requests, set the A/D activation factor selection bits (ADTC[n]:STS[2:0]) for software activation ("0b000"), and disable software activation (ADTSE:ADT[n] is "0") in the corresponding bit (activation channel) in the A/D software activation channel selection register (ADTSE).

When setting the A/D activation factor selection bits (ADTC[n]:STS[2:0]), be sure that the 16-bit free-run timer has stopped.

Always set software activation ("0b000") in the STS[2:0] bits in ADTC[1] to ADTC[3] and ADTC[5] to ADTC[7].

**i) Upper-limit Threshold Bits (ADRCUT[m]:C[11:0]) (m=0 to 3)**

The upper-limit threshold bits (ADRCUT[m]:C[11:0]) cannot be modified during an A/D conversion request (ADTS[n]:BUSY is "1").

**j) Lower-limit Threshold Bits (ADRCLT[m]:C[11:0]) (m=0 to 3)**

The lower-limit threshold bits (ADRCLT[m]:C[11:0]) cannot be modified during an A/D conversion request (ADTS[n]:BUSY is "1").

**k) Continuous Detection Frequency Specification and Count Display Bits (ADRCCS[n]:RCOCD[2:0]) (n=0 to 7)**

The continuous detection frequency specification and count display bits (ADRCCS[n]:RCOCD[2:0]) cannot be modified during an A/D conversion request (ADTS[n]:BUSY is "1").

The continuous detection frequency specification and count display bits (ADRCCS[n]:RCOCD[2:0]) cannot be modified while range comparison execution is enabled (ADRCCS[n]:RCOE is "1").

The continuous detection frequency specification and count display bits (ADRCCS[n]:RCOCD[2:0]) cannot be set to "0b000".

**l) Outside/Inside Range Confirmation Selection Bit (ADRCCS[n]:RCOIRS) (n=0 to 7)**

The outside/inside range confirmation selection bit (ADRCCS[n]:RCOIRS) cannot be modified during an A/D conversion request (ADTS[n]:BUSY is "1").

**m) Upper- and Lower-limit Threshold Selection Bits (ADRCCS[n]:RCOTS[1:0]) (n=0 to 7)**

The upper- and lower-limit threshold selection bits (ADRCCS[n]:RCOTS[1:0]) cannot be modified during an A/D conversion request (ADTS[n]:BUSY is "1").

**n) Range Compare Interrupt Factor Flag Bits (ADRCIF:RCINT7 to RCINT0)**

If a software clear (ADRCIFC:RCINT[n]C of "1" is written) occurs at the same time as a hardware set, the hardware set has priority. (n=0 to 7)



#### **o) Compare Match Activation**

If "0x0000" is the set value in the compare register (ADCOMP[n]) and the compare clear register of the 16-bit free-run timer has the same set value, an A/D activation request signal is generated at the compare match time. That would occur regardless of whether the count direction in the count direction selection bit setting (ADTC[n]:SEL[1:0]) is up or down.

#### **p) Compare Buffer Register/Compare Register (ADCOMPB[n]/ADCOMP[n]) (n=0 to 7)**

To access a compare buffer register/compare register, use halfword or word access instructions.

Although all 8 channels have a compare buffer register/compare register (ADCOMPB[n]/ADCOMP[n]) (n=0 to 7), any set values in the ADCOMPB[1] to ADCOMPB[3] and ADCOMPB[5] to ADCOMPB[7] registers are ignored. Therefore, configure only ADCOMPB[0] and ADCOMPB[4].

Since the set values for the ADCOMPB[1] to ADCOMPB[3] and ADCOMPB[5] to ADCOMPB[7] registers are ignored, as mentioned above, only the compare values of ADCOMP[0] and ADCOMP[4] are updated.

#### **q) Range Comparison (n=0 to 7)**

The execution timing of range comparison comes after A/D conversion in each channel has completed and the A/D conversion results have been set in an A/D data register (ADTCD[n]). Accordingly, a range is compared before the generation of an A/D conversion end interrupt request, except in the final conversion channel. In the final conversion channel, the range comparison follows the generation of the A/D conversion end interrupt request. After the range comparison results are reflected in the continuous detection count display bit (ADRCSS[n]:RCOCD[2:0]), the out-of-range flag bit (ADRCOT:RCOOF[n]), and the range compare interrupt factor flag bit (ADRCIF:RCINT[n]), another range compare interrupt request is generated.

Even if the range comparison result changes from being above the upper-limit threshold value to below the lower-limit threshold value when confirmed as outside the range in a range comparison (ADRCSS[n]:RCOIRS is "0"), the continuous detection of the range continues without continuous detection measurement being cleared to 0 times.

To initialize the continuous detection count for a range comparison result, change the range comparison execution setting from disabled to enabled (from "0" to "1" in ADRCSS[n]:RCOE) while no A/D conversion is requested (ADTS[n]:BUSY is "0").

#### **r) Preventing Overwriting of Conversion Data when Configuring Data Protection Function**

In a state where the A/D data register protection clear selection bit of activation channel 0 or 4 is "1" (ADTC[0]:PRTS is "1" or ADTC[4]:PRTS is "1") with the data protection function enabled for that activation channel (ADTC[0]:PRT is "1" or ADTC[4]:PRT is "1"), the data protection setting is released when the A/D data register of the activation channel (ADTCD[0] or ADTCD[4]) is read.

In a state where the A/D data register protection clear selection bit of activation channel 0 or 4 is "0" (ADTC[0]:PRTS is "0" or ADTC[4]:PRTS is "0"), the data protection setting is released when the A/D data register of the activation channel (ADTCD[0] or ADTCD[4]) is read and the A/D conversion end interrupt in the activation channel is cleared.

The release of the data protection setting leads to a delay in the reading of the A/D data registers of activation channels 1 to 3 or activation channels 5 to 7 (ADTCD[1] to ADTCD[3] or ADTCD[5] to ADTCD[7]). If they are not read by the time that the next A/D conversion results are to be written to these A/D data registers, the data in the registers will be overwritten. To prevent this overwriting, read the A/D data registers of activation channels 1 to 3 or activation channels 5 to 7 before reading the A/D data register of activation channel 0 or 4, respectively.

### s) ADTCD[n]:ERR and ADTCD[n]:ERRST Update Timing

The conversion data error flag bit (ADTCD[n]:ERR) and the conversion data error status bit (ADTCD[n]:ERRST) are updated at the time that the conversion data for the set final channel is stored in an A/D data register (ADTCD[n]).

**Table 6-1 ADTCD[n]:ERR and ADTCD[n]:ERRST Update Timing Example (n=0 to 7)**

Activation Request Mode	Data Protection Function	Activation Factor	ADTCD[n]:ERR, ADTCD[n]:ERRST Update Timing
Single mode (ADTC[n]:RPT="0") (n=0 to 7)	Disabled (ADTC[n]:PRT="0") (n=0 to 7)	Compare match activation (ADTC[n]:STS[2:0]="0b011") (n=0 to 7)	Time that A/D activation request busy bit (ADTS[n]:BUSY) changes from "1" to "0" (n=0, 4)
		Other than compare match activation	
	Enabled (ADTC[n]:PRT="1") (n=0 to 7)	Compare match activation (ADTC[n]:STS[2:0]="0b011") (n=0 to 7)	ADTCD[n]:ERR and ADTCD[n]:ERRST fixed at "0"
		Other than compare match activation	
Repeat mode (ADTC[n]:RPT="1") (n=0 to 7)	Disabled (ADTC[n]:PRT="0") (n=0 to 7)	Compare match activation (ADTC[n]:STS[2:0]="0b011") (n=0 to 7)	Time that interrupt request flag bit (ADTS[n]:INT) changes from "0" to "1" (n=0 to 7)
		Other than compare match activation	
	Enabled (ADTC[n]:PRT="1") (n=0 to 7)	Compare match activation (ADTC[n]:STS[2:0]="0b011") (n=0 to 7)	ADTCD[n]:ERR and ADTCD[n]:ERRST fixed at "0"
		Other than compare match activation	







## CHAPTER 58: 12-bit 4ch A/D Converter Arbitration

This chapter provides an overview and explains the operation of 12-bit 4ch A/D converter arbitration.

---

1. Overview
2. Configuration
3. Operation



## 1. Overview

Each 12-bit 4ch A/D converter is equipped with a single A/D converter arbitration block.

### Function of 12-bit 4ch A/D Converter Arbitration

- Each 12-bit 4ch A/D converter unit has 1 A/D activation arbitration block.
- The A/D activation arbitration configuration contains an arbitration circuit function block and A/D activation trigger generation function block.
- This arbitration function arbitrates activation requests from A/D activation compare, and it generates activation triggers and A/D conversion cancel signals.
- An activation trigger is generated from 1 selected activation request from A/D activation compare.

If contention arises between activation requests from A/D activation compare, A/D activation arbitration controls their priority.

The order of priority is "the lowest activation group number is first" (priority control by group number) and as follows (priority control by activation factor): compare match > external trigger/base timer > software activation. The activation factor not selected has to wait. Once the A/D conversion being processed ends, arbitration is performed again. Priority is controlled by activation factor during A/D conversion too. At that time, the current conversion is suspended to process a higher-priority activation factor. After the higher-priority conversion ends, arbitration is performed again. If nothing else has a higher priority according to the activation group number or activation factor, processing resumes for the suspended activation factor.

**Table 1-1 A/D Activation Trigger Arbitration**

A/D Operation	Priority	Operation
Conversion stopped	Low	Activation factors with higher priority are processed first.
	Same	Activation factors of activation group 0 are processed.
	High	Activation factors with higher priority are processed first.
Conversion in progress	Low	After the current conversion ends, arbitration is processed again.
	Same	After the current conversion ends, arbitration is processed again.
	High	The current conversion is suspended, and higher-priority activation factors are processed.

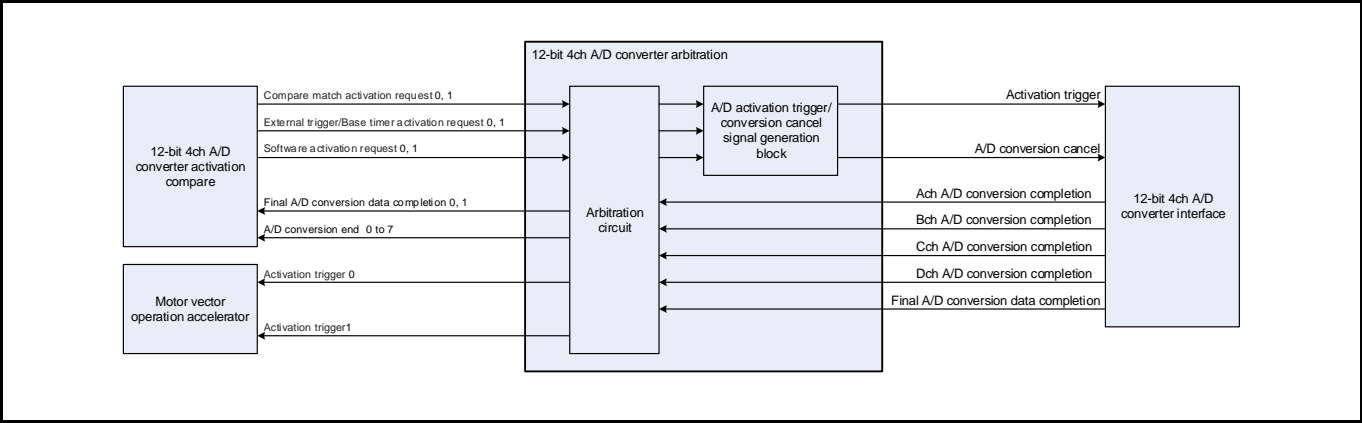
- An activation cancel signal is generated to forcibly terminate the current conversion processing when the activation factor of the conversion in progress becomes inactive and no other activation factors are active.



2. Configuration

This section shows a 12-bit 4ch A/D converter arbitration block diagram.

Figure 2-1 12-bit 4ch A/D Converter Arbitration Configuration





### 3. Operation

This section explains the operation of 12-bit 4ch A/D converter activation arbitration.

#### 3.1. Operation of 12-bit 4ch A/D Arbitration

This arbitration function arbitrates activation requests from A/D activation compare and generates A/D activation triggers. It also generates A/D conversion cancel signals.

### 3.1.1. A/D Activation Trigger Arbitration

An A/D activation trigger is generated from 1 selected activation request from activation groups 0 and 1 of A/D activation compare. Activation requests from activation groups 0 and 1 of A/D activation compare are input as any of the following 3 types: software activation requests, external trigger/base timer activation requests, and compare match activation requests. A/D activation trigger signals are then generated.

If contention arises between activation factors, the activation factor with the higher priority has priority. The activation factor not selected has to wait. Once the A/D conversion being processed ends, arbitration is performed again.

The order of priority according to the activation factor in activation arbitration is as follows: compare match activation request > external trigger/base timer activation request > software activation request. For activation factors with the same priority, the one for activation group 0 for which conversion has stopped has priority.

Table 3-1 A/D Activation Trigger Arbitration

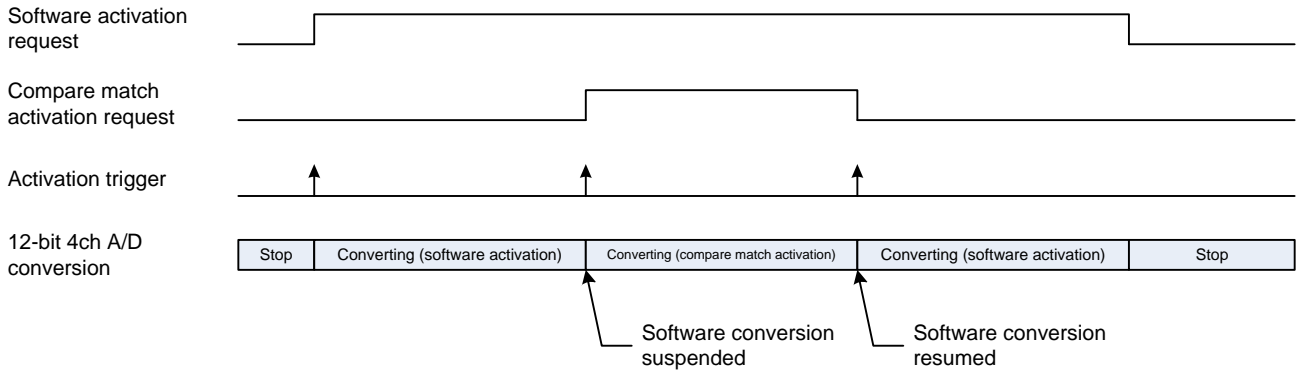
A/D Operation	Priority	Operation
Conversion stopped	Low	Activation factors with higher priority are processed first.
	Same	Activation factors of activation group 0 are processed.
	High	Activation factors with higher priority are processed first.
Conversion in progress	Low	After the current conversion ends, arbitration is processed again.
	Same	After the current conversion ends, arbitration is processed again.
	High	The current conversion is suspended, and higher-priority activation factors are processed.



Figure 3-1 Illustration of ADC Arbitration Operation

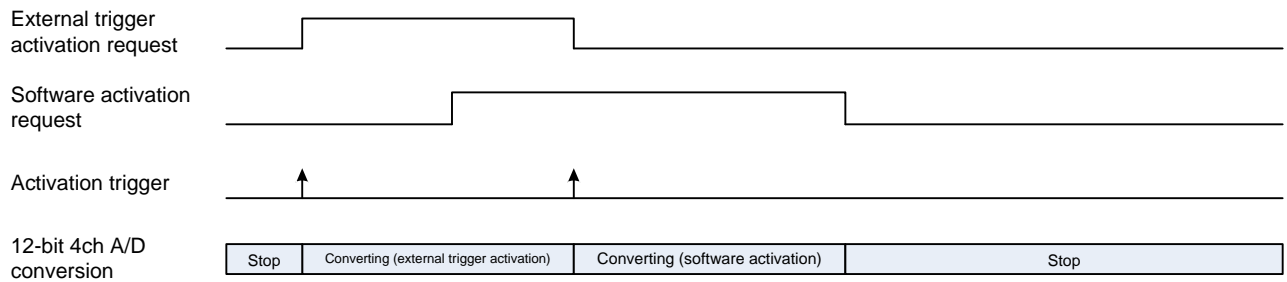
- Case of compare match activation (activation group 1) during software activation (activation group 0)

→Software activation conversion is suspended, and compare match activation conversion is processed with priority. After the end of the compare match activation conversion processing, the suspended software activation conversion resumes. (Re-execution)



- Case of software activation (activation group 0) during external trigger activation (activation group 1)

→After the end of external trigger conversion, the software conversion is processed



### **3.1.2. A/D Conversion Cancel Function**

An A/D conversion cancel signal is generated to forcibly terminate the current conversion processing when the activation request of a requester becomes inactive during A/D conversion. If an activation factor of another activation group is active when the activation request of the requester becomes inactive, the active activation factor generates an A/D activation trigger without an A/D conversion cancel signal being generated.





### 3.1.3. Operation Timing

Figure 3-2 ADC Arbitration Operation Timing (Software Activation, External Trigger/Base Timer Activation)

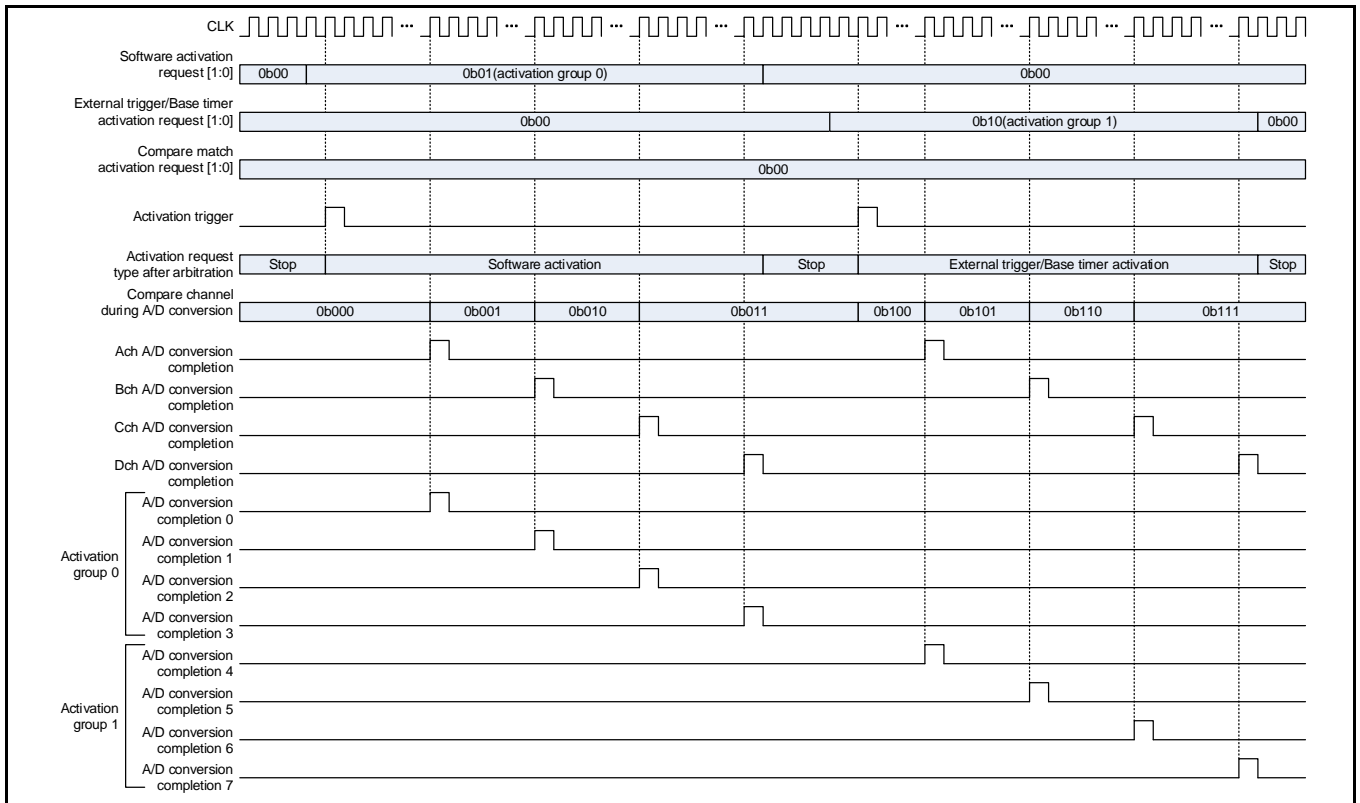


Figure 3-3 ADC Arbitration Operation Timing (Compare Match Activation)

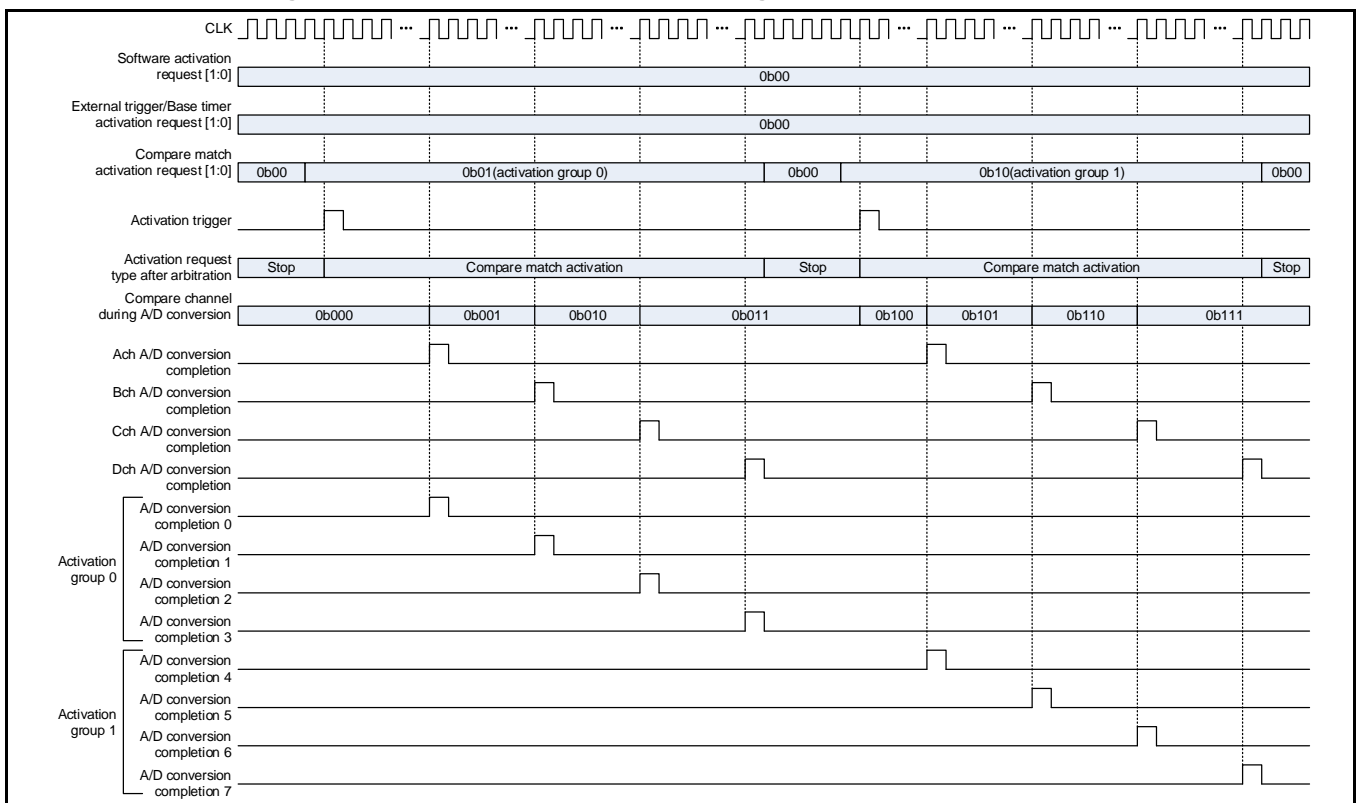
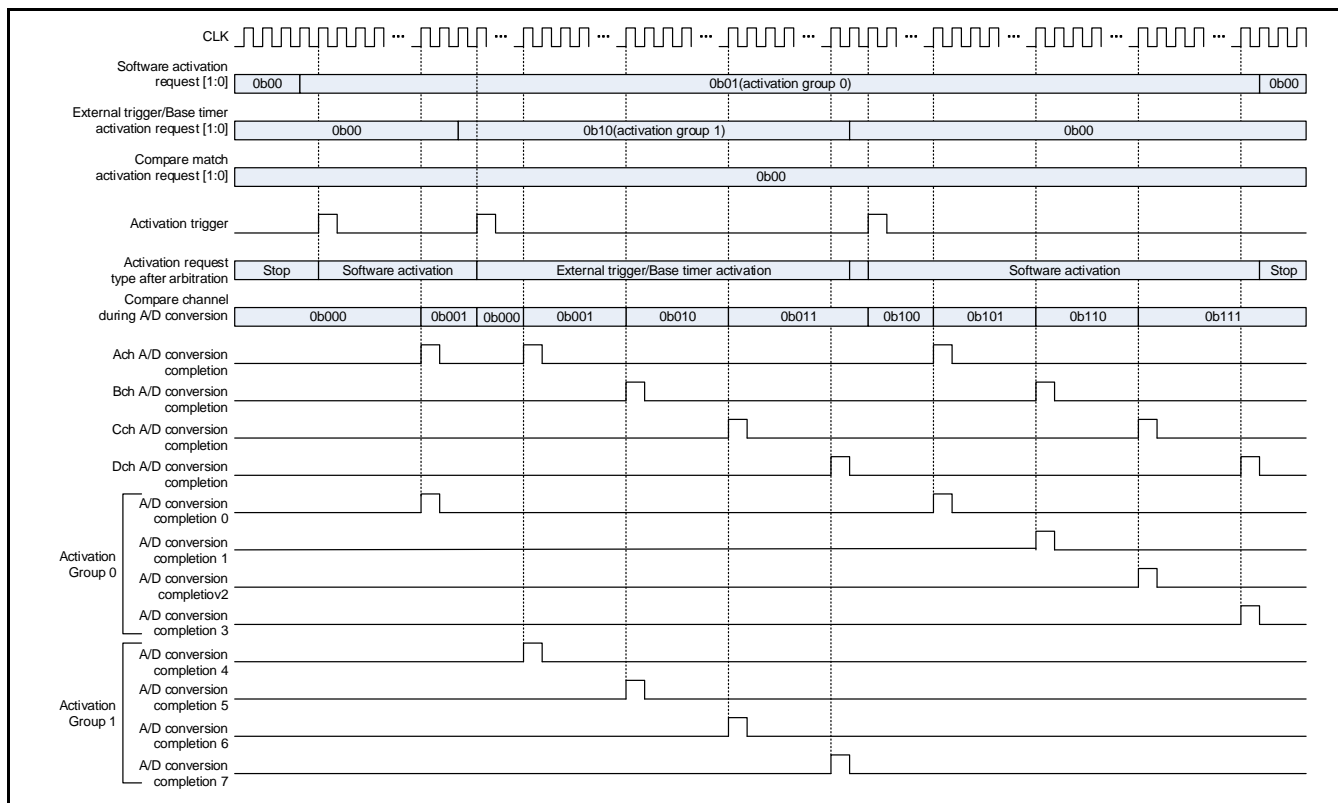


Figure 3-4 ADC Arbitration Operation Timing (Software Activation, External Trigger/Base Timer Activation)



If an activation factor occurs with a higher priority than the activation factor of the current activation in progress, the higher-priority activation factor generates an activation trigger.

(Priority control by activation factor: compare match activation > external trigger/base timer activation > software activation)





# CHAPTER 59: Motor Vector Operation Accelerator

This chapter explains the functions and operations of the motor vector operation accelerator.

---

## 1. Overview



## 1. Overview

The motor vector operation accelerator is hardware that assists with the torque control for the brushless DC motor. The assist functions include Three-phase current normalization, basic processing of vector calculation (three-phase to two-phase DC conversion/two-phase to three-phase AC conversion, angular calculation), and PID control calculation.

The execution of each calculation can be switched between hardware or a program.

In addition, the R/D converter and three-phase current used for calculations can be diagnosed.

### (1) Functions of the Motor Vector Operation Accelerator

#### a) Error Detection Function

This function detects an overflow/underflow/non-normalized error for floating-point calculation.

In addition, for the parameters used for calculation, ECC code values are used to detect errors for up to 2 bits. No errors are corrected.

#### b) Calculation Schedule

Continuous execution of all calculations is possible whereby 6 calculations from angular calculation to two-phase to three-phase AC conversion are all executed in sequence by hardware. Alternatively, programming execution is possible whereby partial calculations are processed by a program.

#### c) Calculation Activation Mode

For motor control activation, motor control calculation is periodically performed with activation control of the A/D converter synchronized with the motor control free-run timer. For motor control activation, activation with either an A/D converter activation factor or with a program can be selected.

For diagnosis activation, the R/D converter and three-phase current used for calculation are diagnosed several times within a motor control period with activation control of the A/D converter synchronized with the diagnosis base timer.

#### d) Buffer Control

With buffer control, the values of the parameters whose settings are to be changed during calculation are protected from being changed during calculation.

#### e) Setting Data Protection Function

Constant parameters used for motor control are protected with key codes.

In addition, for I/O parameters for angular calculation to two-phase to three-phase AC conversion, data is protected from being updated by any unexpected hardware/program.

#### f) Angular Calculation to Two-phase to Three-phase AC Conversion Calculation

Angular/angular velocity data is obtained from the angular information interface and the current values for three-phase are obtained from the A/D converter. Then, angular calculation to two-phase to three-phase AC conversion calculation are performed. Calculations up to the three-phase voltage ( $V_u/V_v/V_w$ ) calculation are supported.

#### g) Diagnosis Function

Using 2 response signals for the excitation signals sent from the R/D converter to the resolver, this function diagnoses the amplitudes/angles of the R/D converter.

Moreover, an abnormal current is diagnosed based on the obtained three-phase current.

## **(2) Calculation Overview**

The following calculations are performed for angular calculation to two-phase to three-phase AC conversion.

### **a) Angular Calculation**

- Obtaining angular calculation information
- Sine cosine conversion
- Sine cosine conversion after lead angle correction
- R/D converter diagnosis

### **b) Three-phase Current Normalization**

- Three-phase current selection
- Three-phase current information acquisition
- Calculation for three-phase current normalization
- Normalization calculation in two-phase current use mode
- Detection of cumulative three-phase current abnormalities

### **c) Three-phase to Two-phase DC Conversion**

- Three-phase to two-phase DC conversion calculation
- Detection of three-phase to two-phase DC value abnormality
- Filtering

### **d) PID Control**

- PID control calculation

### **e) Current to Voltage Conversion**

- Current to voltage conversion calculation

### **f) Two-phase to Three-phase AC Conversion**

- Two-phase to three-phase AC conversion calculation



### (3) Main Terms

The following explains the main terms related to the motor vector operation accelerator.

Term	Meaning
Motor control free-run timer	This free-run timer is selected to generate a motor control period.
R/D converter diagnosis free-run timer	This free-run timer generates the maximum phase value for an excitation waveform used for R/D converter diagnosis.
Synchronous angle data (angle $\phi$ )	This data retains the angle data to be input, at the timing of motor control A/D conversion start or that of diagnosis A/D conversion start.
Synchronous angular velocity data	This data retains the angular velocity data to be input, at the timing of motor control A/D conversion start or that of diagnosis A/D conversion start.
Present current	This two-phase current value is calculated by performing three-phase to two-phase DC conversion for acquired three-phase current.
Target current	This target value is used for PID control.
Control current	This current value is calculated by applying PID control to a two-phase present current value.
Deviation current	Deviation between the target and present current values used for PID control
Previous deviation current	This deviation current value is calculated with PID control in the previous motor control period.
Next-to-last deviation current	This deviation current value is calculated with PID control in the next-to-last motor control period.
Input 1/input 2	Indicates 2 items of input data in the arithmetic unit. Example) When a calculation of $A+B=C$ is performed, the following is assumed. A: Input 1, B: Input 2, and C: Output

For more information about MVA, contact a support representative.

## CHAPTER 60: Appendix

---

1. I/O Port Settings
2. Registers Map
3. List of Interrupt Factors and DMA Activation Factors
4. Master Access
5. Low-power Mode
6. Major Changes





## 1. I/O Port Settings

This section shows I/O port settings.

### 1.1. Input Level Setting

This section shows input level setting of I/O ports.

#### (1) P325 to P330

PIL[1:0]	Description
00	Automotive input (0.8Vcc/0.5Vcc)
01	FlexRay input (0.7Vcc/0.3Vcc)
10	Setting prohibited
11	Setting prohibited

#### (2) Pins Other than Above

PIL[1:0]	Description
00	Automotive input (0.8Vcc/0.5Vcc)
01	CMOS hysteresis input (0.7Vcc/0.3Vcc)
10	Setting prohibited
11	Setting prohibited

### 1.2. Output Drive Setting

This section shows output drive setting of I/O ports.

#### (1) P325 to P330

ODR[1:0]	Description	
	I <sub>OL</sub>	I <sub>OH</sub>
00	1mA	-1mA
01	2mA	-2mA
10	4mA	-4mA
11	Setting prohibited	Setting prohibited

#### (2) Pins Other than Above

ODR[1:0]	Description	
	I <sub>OL</sub>	I <sub>OH</sub>
00	1mA	-1mA
01	2mA	-2mA
10	Setting prohibited	Setting prohibited
11	Setting prohibited	Setting prohibited

### 1.3. Output Resource Selection

This section shows output resource of each pin and their settings.

Table 1-1 List of Output Resource

Register Name	Port	Output Resource Selection (POF[2:0])			
		000	001	010	011 to 111
PPC_PCFGR000	P000	GPIO_P000	-	-	-
PPC_PCFGR001	P001	GPIO_P001	RTO0	-	-
PPC_PCFGR002	P002	GPIO_P002	RTO1	-	-
PPC_PCFGR003	P003	GPIO_P003	RTO2	-	-
PPC_PCFGR004	P004	GPIO_P004	RTO3	-	-
PPC_PCFGR005	P005	GPIO_P005	RTO4	-	-
PPC_PCFGR006	P006	GPIO_P006	RTO5	-	-
PPC_PCFGR007	P007	GPIO_P007	-	-	-
PPC_PCFGR008	P008	GPIO_P008	-	-	-
PPC_PCFGR009	P009	GPIO_P009	-	-	-
PPC_PCFGR010	P010	GPIO_P010	-	-	-
PPC_PCFGR011	P011	GPIO_P011	RDC_W0	-	-
PPC_PCFGR012	P012	GPIO_P012	RDC_V0	-	-
PPC_PCFGR013	P013	GPIO_P013	RDC_U0	-	-
PPC_PCFGR014	P014	GPIO_P014	RDC_Z0	-	-
PPC_PCFGR015	P015	GPIO_P015	RDC_B0	-	-
PPC_PCFGR016	P016	GPIO_P016	RDC_A0	-	-
PPC_PCFGR017	P017	AREF20	-	-	-
PPC_PCFGR018	P018	SIN_IN0	-	-	-
PPC_PCFGR019	P019	COS_IN0	-	-	-
PPC_PCFGR020	P020	SIN_OUT0	-	-	-
PPC_PCFGR021	P021	SIN_MINUS0	-	-	-
PPC_PCFGR022	P022	SIN_PLUS0	-	-	-
PPC_PCFGR023	P023	COS_PLUS0	-	-	-
PPC_PCFGR024	P024	COS_MINUS0	-	-	-
PPC_PCFGR025	P025	COS_OUT0	-	-	-
PPC_PCFGR026	P026	RDC_ACT0	GPIO_P026	-	-
PPC_PCFGR027	P027	MAG_MINUS0	-	-	-
PPC_PCFGR028	P028	MAG_PLUS0	-	-	-
PPC_PCFGR029	P029	MAG_OUT0	-	-	-
PPC_PCFGR030	P030	GPIO_P030	-	-	-
PPC_PCFGR031	P031	GPIO_P031	RTO12	-	-
PPC_PCFGR100	P100	GPIO_P100	-	-	-
PPC_PCFGR101	P101	GPIO_P101	RTO6	-	-
PPC_PCFGR102	P102	GPIO_P102	RTO7	-	-
PPC_PCFGR103	P103	GPIO_P103	RTO8	-	-
PPC_PCFGR104	P104	GPIO_P104	RTO9	-	-
PPC_PCFGR105	P105	GPIO_P105	RTO10	-	-
PPC_PCFGR106	P106	GPIO_P106	RTO11	-	-
PPC_PCFGR107	P107	GPIO_P107	-	-	-
PPC_PCFGR108	P108	GPIO_P108	-	-	-
PPC_PCFGR109	P109	GPIO_P109	-	-	-
PPC_PCFGR110	P110	GPIO_P110	-	-	-
PPC_PCFGR111	P111	GPIO_P111	RDC_W1	-	-



Register Name	Port	Output Resource Selection (POF[2:0])			
		000	001	010	011 to 111
PPC_PCFGR112	P112	GPIO_P112	RDC_V1	-	-
PPC_PCFGR113	P113	GPIO_P113	RDC_U1	-	-
PPC_PCFGR114	P114	GPIO_P114	RDC_Z1	-	-
PPC_PCFGR115	P115	GPIO_P115	RDC_B1	-	-
PPC_PCFGR116	P116	GPIO_P116	RDC_A1	-	-
PPC_PCFGR117	P117	AREF21	-	-	-
PPC_PCFGR118	P118	SIN_IN1	-	-	-
PPC_PCFGR119	P119	COS_IN1	-	-	-
PPC_PCFGR120	P120	SIN_OUT1	-	-	-
PPC_PCFGR121	P121	SIN_MINUS1	-	-	-
PPC_PCFGR122	P122	SIN_PLUS1	-	-	-
PPC_PCFGR123	P123	COS_PLUS1	-	-	-
PPC_PCFGR124	P124	COS_MINUS1	-	-	-
PPC_PCFGR125	P125	COS_OUT1	-	-	-
PPC_PCFGR126	P126	RDC_ACT1	GPIO_P126	-	-
PPC_PCFGR127	P127	MAG_MINUS1	-	-	-
PPC_PCFGR128	P128	MAG_PLUS1	-	-	-
PPC_PCFGR129	P129	MAG_OUT1	-	-	-
PPC_PCFGR131	P131	GPIO_P131	SCK2	-	-
PPC_PCFGR200	P200	GPIO_P200	-	-	-
PPC_PCFGR201	P201	GPIO_P201	-	-	-
PPC_PCFGR202	P202	GPIO_P202	-	-	-
PPC_PCFGR203	P203	GPIO_P203	-	-	-
PPC_PCFGR204	P204	GPIO_P204	-	-	-
PPC_PCFGR205	P205	GPIO_P205	-	-	-
PPC_PCFGR206	P206	GPIO_P206	-	-	-
PPC_PCFGR207	P207	GPIO_P207	-	-	-
PPC_PCFGR208	P208	GPIO_P208	-	-	-
PPC_PCFGR209	P209	GPIO_P209	-	-	-
PPC_PCFGR210	P210	GPIO_P210	-	-	-
PPC_PCFGR211	P211	GPIO_P211	-	-	-
PPC_PCFGR212	P212	GPIO_P212	-	-	-
PPC_PCFGR213	P213	GPIO_P213	-	-	-
PPC_PCFGR214	P214	GPIO_P214	-	-	-
PPC_PCFGR215	P215	GPIO_P215	-	-	-
PPC_PCFGR216	P216	GPIO_P216	-	-	-
PPC_PCFGR217	P217	GPIO_P217	-	-	-
PPC_PCFGR218	P218	GPIO_P218	-	-	-
PPC_PCFGR219	P219	GPIO_P219	-	-	-
PPC_PCFGR220	P220	GPIO_P220	TIOA6	-	-
PPC_PCFGR221	P221	GPIO_P221	-	-	-
PPC_PCFGR222	P222	GPIO_P222	TIOA7	-	-
PPC_PCFGR223	P223	GPIO_P223	-	-	-
PPC_PCFGR224	P224	GPIO_P224	-	-	-
PPC_PCFGR225	P225	GPIO_P225	-	-	-
PPC_PCFGR226	P226	GPIO_P226	SOT4	-	-
PPC_PCFGR227	P227	GPIO_P227	SCK4	-	-
PPC_PCFGR228	P228	GPIO_P228	SCS40	-	-
PPC_PCFGR229	P229	GPIO_P229	SCS41	-	-

Register Name	Port	Output Resource Selection (POF[2:0])			
		000	001	010	011 to 111
PPC_PCFGR230	P230	GPIO_P230	SCS42	-	-
PPC_PCFGR231	P231	GPIO_P231	SCS43	-	-
PPC_PCFGR300	P300	GPIO_P300	-	-	-
PPC_PCFGR301	P301	GPIO_P301	TIOA10	-	-
PPC_PCFGR302	P302	GPIO_P302	-	-	-
PPC_PCFGR303	P303	GPIO_P303	TIOA11	-	-
PPC_PCFGR304	P304	GPIO_P304	-	-	-
PPC_PCFGR305	P305	GPIO_P305	-	-	-
PPC_PCFGR306	P306	GPIO_P306	-	-	-
PPC_PCFGR309	P309	GPIO_P309	RTO13	-	-
PPC_PCFGR310	P310	GPIO_P310	RTO14	-	-
PPC_PCFGR311	P311	GPIO_P311	RTO15	-	-
PPC_PCFGR312	P312	GPIO_P312	RTO16	-	-
PPC_PCFGR313	P313	GPIO_P313	RTO17	-	-
PPC_PCFGR314	P314	GPIO_P314	TIOA0	-	-
PPC_PCFGR315	P315	GPIO_P315	RTO18	-	-
PPC_PCFGR316	P316	GPIO_P316	RTO19	TIOA1	-
PPC_PCFGR317	P317	GPIO_P317	RTO20	-	-
PPC_PCFGR318	P318	GPIO_P318	RTO21	TIOA2	-
PPC_PCFGR319	P319	GPIO_P319	RTO22	-	-
PPC_PCFGR320	P320	GPIO_P320	RTO23	TIOA3	-
PPC_PCFGR321	P321	GPIO_P321	-	-	-
PPC_PCFGR322	P322	GPIO_P322	SOT0	-	-
PPC_PCFGR323	P323	GPIO_P323	SCK0	-	-
PPC_PCFGR324	P324	GPIO_P324	-	-	-
PPC_PCFGR325	P325	GPIO_P325	-	-	-
PPC_PCFGR326	P326	GPIO_P326	TXDA	-	-
PPC_PCFGR327	P327	GPIO_P327	TXENA	-	-
PPC_PCFGR328	P328	GPIO_P328	-	-	-
PPC_PCFGR329	P329	GPIO_P329	TXDB	-	-
PPC_PCFGR330	P330	GPIO_P330	TXENB	-	-
PPC_PCFGR406	P406	GPIO_P406	-	-	-
PPC_PCFGR407	P407	GPIO_P407	SOT1	-	-
PPC_PCFGR408	P408	GPIO_P408	SCK1	-	-
PPC_PCFGR409	P409	GPIO_P409	-	-	-
PPC_PCFGR410	P410	GPIO_P410	TX0	-	-
PPC_PCFGR411	P411	GPIO_P411	-	-	-
PPC_PCFGR412	P412	GPIO_P412	TX1	-	-
PPC_PCFGR413	P413	GPIO_P413	-	-	-
PPC_PCFGR414	P414	GPIO_P414	TX2	-	-
PPC_PCFGR415	P415	GPIO_P415	TIOA4	-	-
PPC_PCFGR416	P416	GPIO_P416	-	-	-
PPC_PCFGR417	P417	GPIO_P417	TIOA5	-	-
PPC_PCFGR418	P418	GPIO_P418	-	-	-
PPC_PCFGR419	P419	GPIO_P419	-	-	-
PPC_PCFGR420	P420	GPIO_P420	SOT2	-	-
PPC_PCFGR421	P421	GPIO_P421	-	-	-
PPC_PCFGR422	P422	GPIO_P422	SOT3	-	-
PPC_PCFGR423	P423	GPIO_P423	SCK3	-	-



Register Name	Port	Output Resource Selection (POF[2:0])			
		000	001	010	011 to 111
PPC_PCFGR425	P425	GPIO_P425	TIOA8	-	-
PPC_PCFGR426	P426	GPIO_P426	-	-	-
PPC_PCFGR427	P427	GPIO_P427	TIOA9	-	-
PPC_PCFGR428	P428	GPIO_P428	-	-	-
PPC_PCFGR429	P429	GPIO_P429	MONCLK	MM	-
PPC_PCFGR430	P430	GPIO_P430	ERDS0	-	-
PPC_PCFGR431	P431	GPIO_P431	ERDS1	-	-

## 1.4. Resource Input Selection

This section shows resource input settings and assignment.

**Table 1-2 List of Resource Input**

Register Name	Resource	Resource	
		RESSEL[3:0]	Function
RIC_RESIN0	32-bit input capture ch.0 timer value input	0000	32-bit free-run timer ch.0 32-bit timer counter output
		0001	32-bit free-run timer ch.1 32-bit timer counter output
		0010	32-bit free-run timer ch.2 32-bit timer counter output
		0011	32-bit free-run timer ch.3 32-bit timer counter output
		0100 to 1111	32-bit free-run timer ch.4 32-bit timer counter output
RIC_RESIN1	32-bit input capture ch.1 timer value input	0000	32-bit free-run timer ch.0 32-bit timer counter output
		0001	32-bit free-run timer ch.1 32-bit timer counter output
		0010	32-bit free-run timer ch.2 32-bit timer counter output
		0011	32-bit free-run timer ch.3 32-bit timer counter output
		0100 to 1111	32-bit free-run timer ch.4 32-bit timer counter output
RIC_RESIN2	32-bit input capture ch.0 external input signal	0000	External pin IN16
		0001	Multi-function serial interface ch.0 LIN Sync detection signal
		0010 to 1111	External pin IN16
RIC_RESIN3	32-bit input capture ch.1 external input signal	0000	External pin IN17
		0001	Multi-function serial interface ch.1 LIN Sync detection signal
		0010 to 1111	External pin IN17
RIC_RESIN4	32-bit input capture ch.2 timer value input	0000	32-bit free-run timer ch.0 32-bit timer counter output
		0001	32-bit free-run timer ch.1 32-bit timer counter output
		0010	32-bit free-run timer ch.2 32-bit timer counter output
		0011	32-bit free-run timer ch.3 32-bit timer counter output
		0100 to 1111	32-bit free-run timer ch.4 32-bit timer counter output
RIC_RESIN5	32-bit input capture ch.3 timer value input	0000	32-bit free-run timer ch.0 32-bit timer counter output
		0001	32-bit free-run timer ch.1 32-bit timer counter output
		0010	32-bit free-run timer ch.2 32-bit timer counter output
		0011	32-bit free-run timer ch.3 32-bit timer counter output
		0100 to 1111	32-bit free-run timer ch.4 32-bit timer counter output
RIC_RESIN6	32-bit input capture ch.2 external input signal	0000	External pin IN18
		0001	Multi-function serial interface ch.2 LIN Sync detection signal
		0010 to 1111	External pin IN18
RIC_RESIN7	32-bit input capture ch.3 external input signal	0000	External pin IN19
		0001	Multi-function serial interface ch.3 LIN Sync detection signal
		0010 to 1111	External pin IN19
RIC_RESIN8	32-bit input capture ch.4 timer value input	0000	32-bit free-run timer ch.0 32-bit timer counter output
		0001	32-bit free-run timer ch.1 32-bit timer counter output
		0010	32-bit free-run timer ch.2 32-bit timer counter output
		0011	32-bit free-run timer ch.3 32-bit timer counter output
		0100 to 1111	32-bit free-run timer ch.4 32-bit timer counter output

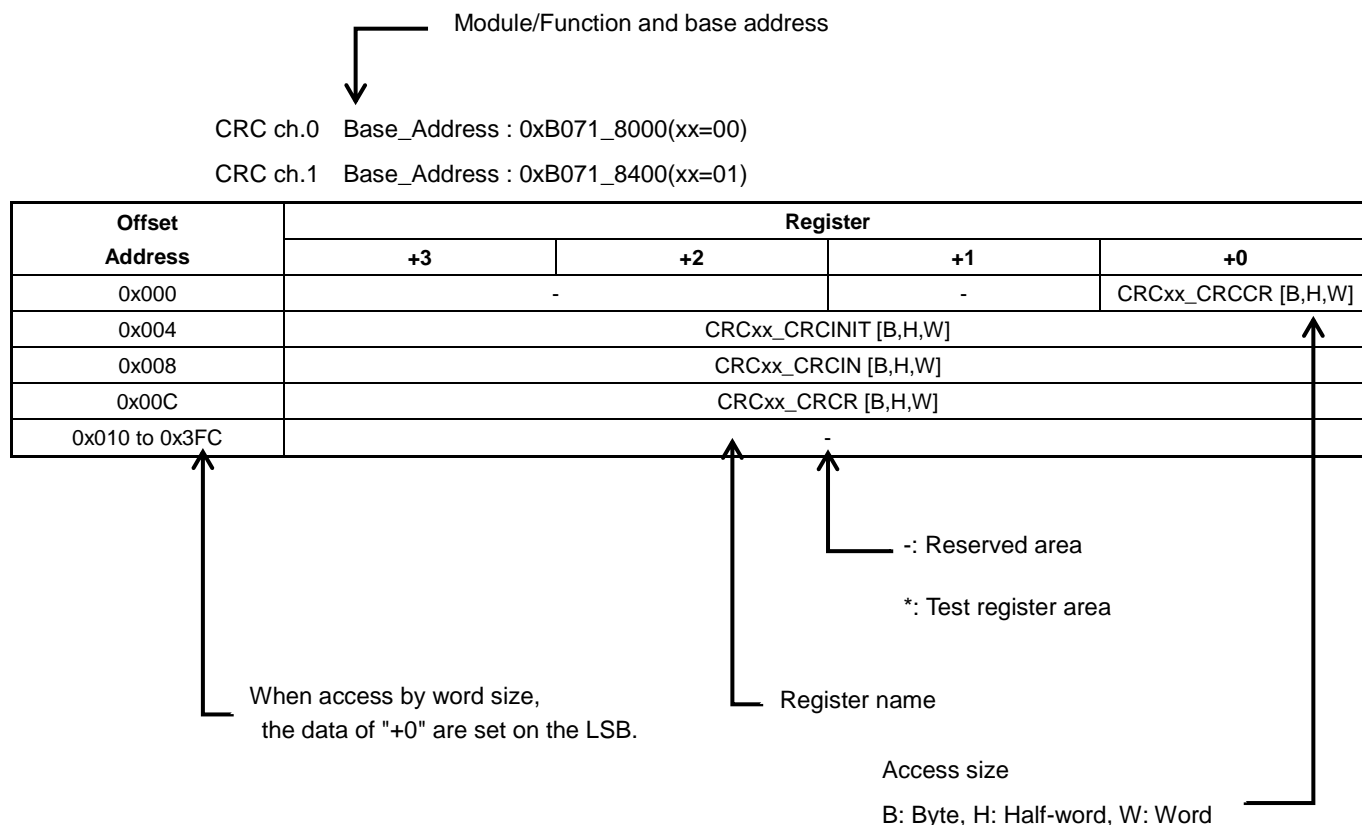


Register Name	Resource	Resource	
		RESSEL[3:0]	Function
RIC_RESIN9	32-bit input capture ch.5 timer value input	0000	32-bit free-run timer ch.0 32-bit timer counter output
		0001	32-bit free-run timer ch.1 32-bit timer counter output
		0010	32-bit free-run timer ch.2 32-bit timer counter output
		0011	32-bit free-run timer ch.3 32-bit timer counter output
		0100 to 1111	32-bit free-run timer ch.4 32-bit timer counter output
RIC_RESIN10	32-bit input capture ch.4 external input signal	0000	External pin IN20
		0001	Multi-function serial interface ch.4 LIN Sync detection signal
		0010 to 1111	External pin IN20
RIC_RESIN11	32-bit input capture ch.5 external input signal	0000 to 1111	External pin IN21

## 2. Registers Map

This section shows register map.

[How to view]



### Note:

- Register table shows in little-endian.
- When accessing data, set address as below.
  - Word access : address is multiple of 4 (lowest 2-bit are "0b00")
  - Half-word access : address is multiple of 2 (lowest bit is "0")
  - Byte access : -
- Do not access reserved area. However, you can access to register including reserved area with allowed access size in the register table.
- Do not access test register area.
- Do not access to area which not described in register table.





### (1) System Controller

System controller protection register Base\_Address : 0xB060\_0000

Offset Address	Register			
	+3	+2	+1	+0
0x000	SYSC_PROTKEYR [W]			
0x004 to 0x07C	-			

System controller RUN profile register Base\_Address : 0xB060\_0000

Offset Address	Register			
	+3	+2	+1	+0
0x080	-			
0x084	SYSC_RUNCKSRER [B,H,W]			
0x088	SYSC_RUNCKSELR0 [B,H,W]			
0x08C	-			
0x090	SYSC_RUNCKER [B,H,W]			
0x094	SYSC_RUNCKDIVR0 [B,H,W]			
0x098	SYSC_RUNCKDIVR1 [B,H,W]			
0x09C	SYSC_RUNCKDIVR2 [B,H,W]			
0x0A0 to 0x0A4	-			
0x0A8	SYSC_RUNPLLCNTR [B,H,W]			
0x0AC to 0x0B0	-			
0x0B4	SYSC_RUNLVDCFGR [B,H,W]			
0x0B8	-			
0x0BC	SYSC_RUNCSVCFGR [B,H,W]			
0x0C0 to 0x0F8	-			
0x0FC	SYSC_TRGRUNCNTR [B,H,W]			

System controller PSS profile register Base\_Address : 0xB060\_0000

Offset Address	Register			
	+3	+2	+1	+0
0x100	-			
0x104	SYSC_PSSCKSRER [B,H,W]			
0x108	SYSC_PSSCKSELR0 [B,H,W]			
0x10C	-			
0x110	SYSC_PSSCKER [B,H,W]			
0x114	SYSC_PSSCKDIVR0 [B,H,W]			
0x118	SYSC_PSSCKDIVR1 [B,H,W]			
0x11C	SYSC_PSSCKDIVR2 [B,H,W]			
0x120 to 0x124	-			
0x128	SYSC_PSSPLLCNTR [B,H,W]			
0x12C to 0x130	-			
0x134	SYSC_PSSLVDCFGR [B,H,W]			
0x138	-			
0x13C	SYSC_PSSCSVCFGR [B,H,W]			
0x140 to 0x178	-			
0x17C	SYSC_PSSSEN [B]			

System controller APPLIED profile register Base\_Address : 0xB060\_0000

Offset Address	Register			
	+3	+2	+1	+0
0x180	-			
0x184	SYSC_APPCKSRER [B,H,W]			
0x188	SYSC_APPCKSELR0 [B,H,W]			
0x18C	-			
0x190	SYSC_APPCKER [B,H,W]			
0x194	SYSC_APPCKDIVR0 [B,H,W]			
0x198	SYSC_APPCKDIVR1 [B,H,W]			
0x19C	SYSC_APPCKDIVR2 [B,H,W]			
0x1A0 to 0x1A4	-			
0x1A8	SYSC_APPPLLCNTR [B,H,W]			
0x1AC to 0x1B0	-			
0x1B4	SYSC_APPLVDCFGR [B,H,W]			
0x1B8	-			
0x1BC	SYSC_APPCSVCFGR [B,H,W]			
0x1C0 to 0x1FC	-			

System controller status profile register Base\_Address : 0xB060\_0000

Offset Address	Register			
	+3	+2	+1	+0
0x200	-			
0x204	SYSC_STSCKSRER [B,H,W]			
0x208	SYSC_STSCKSELR0 [B,H,W]			
0x20C	-			
0x210	SYSC_STSCKER [B,H,W]			
0x214	SYSC_STSCKDIVR0 [B,H,W]			
0x218	SYSC_STSCKDIVR1 [B,H,W]			
0x21C	SYSC_STSCKDIVR2 [B,H,W]			
0x220 to 0x224	-			
0x228	SYSC_STSPLLCNTR [B,H,W]			
0x22C to 0x230	-			
0x234	SYSC_STSLVDCFGR [B,H,W]			
0x238	-			
0x23C	SYSC_STSCSVCFGR [B,H,W]			
0x240 to 0x27C	-			



H A R D W A R E M A N U A L

System controller system registers Base\_Address : 0xB060\_0000

Offset Address	Register			
	+3	+2	+1	+0
0x280	-			
0x284	-			
0x288	SYSC_SYSSTSR [B,H,W]			
0x28C	SYSC_SYSINTER [B,H,W]			
0x290	SYSC_SYSICLR [B,H,W]			
0x294	SYSC_SYSERRIR0 [B,H,W]			
0x298	SYSC_SYSERRIR1 [B,H,W]			
0x29C	SYSC_SYSERRICLR0 [B,H,W]			
0x2A0	SYSC_SYSERRICLR1 [B,H,W]			
0x2A4	SYSC_SYSPROSTSR [B,H,W]			
0x2A8	SYSC_SYSRUNPEFR [B,H,W]			
0x2AC	SYSC_SYSPSSPEFR [B,H,W]			
0x2B0 to 0x2FC	-			

Clock supervisor registers Base\_Address : 0xB060\_0300

Offset Address	Register			
	+3	+2	+1	+0
0x000	SYSC_CSVMOCFGR0 [B,H,W]			
0x004	SYSC_CSVMOCFGR01 [B,H,W]			
0x008	SYSC_CSVMOCFGR10 [B,H,W]			
0x00C	SYSC_CSVMOCFGR11 [B,H,W]			
0x010 to 0x014	-			
0x018	SYSC_CSVPLLCFGR0 [B,H,W]			
0x01C	SYSC_CSVPLLCFGR1 [B,H,W]			
0x020 to 0x024	-			
0x028	SYSC_CSVSSCFGR0 [B,H,W]			
0x02C	SYSC_CSVSSCFGR1 [B,H,W]			
0x030 to 0x05C	-			
0x060	SYSC_CSVOUTER [B,H,W]			
0x064	SYSC_CSVTESTR [B,H,W]			
0x068 to 0x07C	-			

Reset registers Base\_Address : 0xB060\_0380

Offset Address	Register			
	+3	+2	+1	+0
0x000	SYSC_RSTCNTR0 [B,H,W]			
0x004	SYSC_RSTCNTR1 [B,H,W]			
0x008 to 0x00C	-			
0x010	SYSC_RSTCAUSEUR [B,H,W]			
0x014	SYSC_EXCSVRSTCAUSEUR [B,H,W]			
0x018 to 0x01C	-			
0x020	SYSC_RSTCAUSEBT [B,H,W]			
0x024	SYSC_EXCSVRSTCAUSEBT [B,H,W]			
0x028 to 0x02C	-			
0x030	SYSC_WRBOOTCPUSEL [B,H,W]			
0x034 to 0x07C	-			



H A R D W A R E M A N U A L

SCT(Fast-CR) registers Base\_Address : 0xB060\_0400

Offset Address	Register			
	+3	+2	+1	+0
0x000	SYSC_FCRCTTRGR [B,H,W]			
0x004	SYSC_FCRCTCNTR [B,H,W]			
0x008	SYSC_FCRCTCPR [B,H,W]			
0x00C	SYSC_FCRCTSTR [B,H,W]			
0x010	SYSC_FCRCTINTER [B,H,W]			
0x014	SYSC_FCRCTICLR [B,H,W]			
0x018 to 0x07C	-			

SCT(Slow-CR) registers Base\_Address : 0xB060\_0480

Offset Address	Register			
	+3	+2	+1	+0
0x000	SYSC_SCRCTTRGR [B,H,W]			
0x004	SYSC_SCRCTCNTR [B,H,W]			
0x008	SYSC_SCRCTCPR [B,H,W]			
0x00C	SYSC_SCRCTSTR [B,H,W]			
0x010	SYSC_SCRCTINTER [B,H,W]			
0x014	SYSC_SCRCTICLR [B,H,W]			
0x018 to 0x07C	-			

SCT(main oscillation) registers Base\_Address : 0xB060\_0500

Offset Address	Register			
	+3	+2	+1	+0
0x000	SYSC_MOCTTRGR [B,H,W]			
0x004	SYSC_MOCTCNTR [B,H,W]			
0x008	SYSC_MOCTCPR [B,H,W]			
0x00C	SYSC_MOCTSTR [B,H,W]			
0x010	SYSC_MOCTINTER [B,H,W]			
0x014	SYSC_MOCTICLR [B,H,W]			
0x018 to 0x07C	-			
0x080 to 0x0FC	-			

Clock system registers Base\_Address : 0xB060\_0600

Offset Address	Register			
	+3	+2	+1	+0
0x000	SYSC_CRCNTR [B,H,W]			
0x004	SYSC_MOSCCNTR [B,H,W]			
0x008	SYSC_PLLSTCNTR [B,H,W]			
0x00C	SYSC_PLLCGCNTR [B,H,W]			
0x010 to 0x01C	-			
0x020	SYSC_CKOTCNTR [B,H,W]			
0x024 to 0x07C	-			

Special setting registers Base\_Address : 0xB060\_0680

Offset Address	Register			
	+3	+2	+1	+0
0x000	SYSC_SPECFGR [B,H,W]			
0x004	SYSC_SPECPUFCFGR [B,H,W]			
0x008 to 0x07C	-			

Debugging registers Base\_Address : 0xB060\_0700

Offset Address	Register			
	+3	+2	+1	+0
0x000	SYSC_JTAGDETECT [B,H,W]			
0x004	SYSC_JTAGCNFG [B,H,W]			
0x008	SYSC_JTAGWAKEUP [B,H,W]			
0x00C to 0x0FC	-			



## (2) Mode Control

Mode control Base\_Address : 0xB060\_0800

Offset Address	Register			
	+3	+2	+1	+0
0x000	MODEC_MODER [B,H,W]			
0x004 to 0x7FC	-			

## (3) Software Watchdog Timer

Software watchdog timer Unit0 Base\_Address : 0xB060\_8000 (n=0)

Software watchdog timer Unit1 Base\_Address : 0xB060\_9000 (n=1)

Offset Address	Register			
	+3	+2	+1	+0
0x000	SWDGn_PROT [W]			
0x004	-			
0x008	SWDGn_CNT [B,H,W]			
0x00C	SWDGn_RSTCAUSE [B,H,W]			
0x010	SWDGn_TRG0 [B,H,W]			
0x014	-			
0x018	SWDGn_TRG1 [B,H,W]			
0x01C	-			
0x020	SWDGn_INT [B,H,W]			
0x024	SWDGn_INTCLR [B,H,W]			
0x028	-			
0x02C	SWDGn_TRG0CFG [B,H,W]			
0x030	SWDGn_TRG1CFG [B,H,W]			
0x034	SWDGn_RUNLLS [B,H,W]			
0x038	SWDGn_RUNULS [B,H,W]			
0x03C	SWDGn_PSSLLS [B,H,W]			
0x040	SWDGn_PSSULS [B,H,W]			
0x044	SWDGn_RSTDLY [B,H,W]			
0x048	SWDGn_CFG [B,H,W]			
0x04C	SWDGn_RUNLLC [B,H,W]			
0x050	SWDGn_RUNULC [B,H,W]			
0x054	SWDGn_PSSLLC [B,H,W]			
0x058	SWDGn_PSSULC [B,H,W]			
0x05C to 0x3FC	-			

#### (4) Hardware Watchdog Timer

Hardware watchdog timer Base\_Address : 0xB060\_C000

Offset Address	Register			
	+3	+2	+1	+0
0x000	HWDG_PROT [W]			
0x004	-			
0x008	HWDG_CNT [B,H,W]			
0x00C	HWDG_RSTCAUSE [B,H,W]			
0x010	HWDG_TRG0 [B,H,W]			
0x014	-			
0x018	HWDG_TRG1 [B,H,W]			
0x01C	-			
0x020	HWDG_INT [B,H,W]			
0x024	HWDG_INTCLR [B,H,W]			
0x028	-			
0x02C	HWDG_TRG0CFG [B,H,W]			
0x030	HWDG_TRG1CFG [B,H,W]			
0x034	HWDG_RUNLLS [B,H,W]			
0x038	HWDG_RUNULS [B,H,W]			
0x03C	HWDG_PSSLLS [B,H,W]			
0x040	HWDG_PSSULS [B,H,W]			
0x044	HWDG_RSTDLY [B,H,W]			
0x048	HWDG_CFG [B,H,W]			
0x04C	HWDG_RUNLLC [B,H,W]			
0x050	HWDG_RUNULC [B,H,W]			
0x054	HWDG_PSSLLC [B,H,W]			
0x058	HWDG_PSSULC [B,H,W]			
0x05C to 0x3FC	-			





### (5) External Interrupt

External interrupt Base\_Address : 0xB062\_0000 (xx=00)

Offset Address	Register			
	+3	+2	+1	+0
0x000	EICxx_ENIR [B,H,W]			
0x004	EICxx_ENISR [B,H,W]			
0x008	EICxx_ENICR [B,H,W]			
0x00C	EICxx_EIRR [B,H,W]			
0x010	EICxx_EIRCR [B,H,W]			
0x014	EICxx_NFER [B,H,W]			
0x018	EICxx_NFESR [B,H,W]			
0x01C	EICxx_NFECR [B,H,W]			
0x020	EICxx_ELVR0 [B,H,W]			
0x024 to 0x02C	-			
0x030	EICxx_NMIR [B,H,W]			
0x034	EICxx_DRER [B,H,W]			
0x038	EICxx_DRESR [B,H,W]			
0x03C	EICxx_DRECR [B,H,W]			
0x040	EICxx_DRFR [B,H,W]			
0x044	-			

## (6) Interrupt Controller(IRC)

### a)Memory & Config Group Area

Interrupt controller(IRC) Unit0 Base\_Address : 0xB040\_0000 (n=0)

Interrupt controller(IRC) Unit1 Base\_Address : 0xB040\_1000 (n=1)

Offset Address	Register			
	+3	+2	+1	+0
0x000	IRCn_NMIVAS [B,H,W]			
0x004	IRCn_NMIST [B,H,W]			
0x008	IRCn_IRQVAS [B,H,W]			
0x00C	IRCn_IRQST [B,H,W]			
0x010 to 0x08C	IRCn_NMIVA0 to IRCn_NMIVA31 [B,H,W]			
0x090 to 0x88C	IRCn_IRQVA0 to IRCn_IRQVA511 [B,H,W]			
0x890	IRCn_NMIPL0 [B,H,W] <sup>†1</sup>			
0x894 to 0x8AC	IRCn_NMIPL1 to IRCn_NMIPL7 [B,H,W]			
0x8B0 to 0xAAC	IRCn_IRQPL0 to IRCn_IRQPL127 [B,H,W]			
0xAB0	IRCn_NMIS [B,H,W]			
0xAB4	IRCn_NMIR [B,H,W]			
0xAB8	IRCn_NMISIS [B,H,W]			
0xABC	-			
0xAC0 to 0xAFC	IRCn_IRQS0 to IRCn_IRQS15 [B,H,W]			
0xB00 to 0xB3C	IRCn_IRQR0 to IRCn_IRQR15 [B,H,W]			
0xB40 to 0xB7C	IRCn_IRQSI0 to IRCn_IRQSI15 [B,H,W]			
0xB80 to 0xBBC	IRCn_IRQCE0 to IRCn_IRQCE15 [B,H,W]			
0xBC0 to 0xBFC	IRCn_IRQCE0 to IRCn_IRQCE15 [B,H,W]			
0xC00 to 0xC3C	IRCn_IRQCE0 to IRCn_IRQCE15 [B,H,W]			
0xC40	IRCn_NMIHC [B,H,W]			
0xC44	IRCn_NMIHS [B,H,W]			
0xC48	IRCn_IRQHC [H,W]			
0xC4C	-			
0xC50 to 0xC8C	IRCn_IRQHS0 to IRCn_IRQHS15 [B,H,W]			
0xC90	IRCn_IRQPLM [B,H,W]			
0xC94	-			
0xC98	IRCn_CSR [B,H,W]			
0xC9C	-			
0xCA0	-			
0xCA4	-			
0xCA8	IRCn_NMIRS [B,H,W]			
0xCAC	IRCn_NMIPS [B,H,W]			
0xCB0 to 0xCEC	IRCn_IRQRS0 to IRCn_IRQRS15 [B,H,W]			
0xCF0 to 0xD2C	IRCn_IRQPS0 to IRCn_IRQPS15 [B,H,W]			
0xD30	IRCn_UNLOCK [W]			
0xD34	-			
0xD38	-			
0xD3C	IRCn_IRQEEVA [B,H,W]			
0xD40	IRCn_EEI [B,H,W]			
0xD44	IRCn_EAN [B,H,W]			
0xD48	IRCn_ET [B,H,W]			
0xD4C	IRCn_EEB0 [B,H,W]			
0xD50	IRCn_EEB1 [B,H,W]			



Offset Address	Register			
	+3	+2	+1	+0
0xD54	IRCN_EEB2 [B,H,W]			
0xD58	-			
0xD5C	-			
0xD60	*			
0xD64	*			
0xD68	*			
0xD6C	*			
0xD70	*			
0xD74	*			
0xD78 to 0xFFC	-			

\*1: Access to lower 8-bit (bit[7:0]) with byte size is prohibited.

### b) Error Config Area

Interrupt controller(IRC) Unit0 Base\_Address : 0xFFFFE\_E000 (n=0)

Interrupt controller(IRC) Unit1 Base\_Address : 0xFFFFE\_E400 (n=1)

Offset Address	Register			
	+3	+2	+1	+0
0x000 to 0x3F8	-			
0x3FC	IRCN_NMIVASBR [B,H,W]			

Interrupt controller(IRC) Mirror Base\_Address : 0xFFFFE\_F800

Offset Address	Register			
	+3	+2	+1	+0
0x000 to 0x3F8	-			
0x3FC	IRC_NMIVASBR [B,H,W]			

### (7) NMI Distribution

NMI distribution Base\_Address : 0xB040\_7000

Offset Address	Register			
	+3	+2	+1	+0
0x000	NMID_UNLOCK [W]			
0x004	NMID_LST [B,H,W]			
0x008 to 0x00C	-			
0x010	NMID_DIST3 [B,H,W]	NMID_DIST2 [B,H,W]	NMID_DIST1 [B,H,W]	NMID_DIST0 [B,H,W]
0x014	NMID_DIST7 [B,H,W]	NMID_DIST6 [B,H,W]	NMID_DIST5 [B,H,W]	NMID_DIST4 [B,H,W]
0x018	NMID_DIST11 [B,H,W]	NMID_DIST10 [B,H,W]	NMID_DIST9 [B,H,W]	NMID_DIST8 [B,H,W]
0x01C	NMID_DIST15 [B,H,W]	NMID_DIST14 [B,H,W]	NMID_DIST13 [B,H,W]	NMID_DIST12 [B,H,W]
0x020	NMID_DIST19 [B,H,W]	NMID_DIST18 [B,H,W]	NMID_DIST17 [B,H,W]	NMID_DIST16 [B,H,W]
0x024	NMID_DIST23 [B,H,W]	NMID_DIST22 [B,H,W]	NMID_DIST21 [B,H,W]	NMID_DIST20 [B,H,W]
0x028	NMID_DIST27 [B,H,W]	NMID_DIST26 [B,H,W]	NMID_DIST25 [B,H,W]	NMID_DIST24 [B,H,W]
0x02C	NMID_DIST31 [B,H,W]	NMID_DIST30 [B,H,W]	NMID_DIST29 [B,H,W]	NMID_DIST28 [B,H,W]
0x030 to 0x3FC	-			



### (8) Timing Protection Unit (TPU)

Timing protection unit (TPU) Unit0 Base\_Address : 0xB040\_8000(n=0)

Timing protection unit (TPU) Unit1 Base\_Address : 0xB040\_9000(n=1)

Offset Address	Register			
	+3	+2	+1	+0
0x000	TPUn_UNLOCK [W]			
0x004	TPUn_LST [B,H,W]			
0x008	TPUn_CFG [B,H,W]			
0x00C	TPUn_TIR [B,H,W]			
0x010	TPUn_TST [B,H,W]			
0x014	TPUn_TIE [B,H,W]			
0x018	-			
0x01C to 0x02C	-			
0x030	TPUn_TCN00 [B,H,W]			
0x034	TPUn_TCN01 [B,H,W]			
0x038	TPUn_TCN02 [B,H,W]			
0x03C	TPUn_TCN03 [B,H,W]			
0x040	TPUn_TCN04 [B,H,W]			
0x044	TPUn_TCN05 [B,H,W]			
0x048	TPUn_TCN06 [B,H,W]			
0x04C	TPUn_TCN07 [B,H,W]			
0x050	TPUn_TCN10 [B,H,W]			
0x054	TPUn_TCN11 [B,H,W]			
0x058	TPUn_TCN12 [B,H,W]			
0x05C	TPUn_TCN13 [B,H,W]			
0x060	TPUn_TCN14 [B,H,W]			
0x064	TPUn_TCN15 [B,H,W]			
0x068	TPUn_TCN16 [B,H,W]			
0x06C	TPUn_TCN17 [B,H,W]			
0x070	TPUn_TCC0 [B,H,W]			
0x074	TPUn_TCC1 [B,H,W]			
0x078	TPUn_TCC2 [B,H,W]			
0x07C	TPUn_TCC3 [B,H,W]			
0x080	TPUn_TCC4 [B,H,W]			
0x084	TPUn_TCC5 [B,H,W]			
0x088	TPUn_TCC6 [B,H,W]			
0x08C	TPUn_TCC7 [B,H,W]			
0x090 to 0x3FC	-			

**(9) TCRAM IF**

TCRAM IF Unit0 Base\_Address : 0xB041\_0000(n=0)

TCRAM IF Unit1 Base\_Address : 0xB041\_0400(n=1)

Offset Address	Register			
	+3	+2	+1	+0
0x000	TRCFGn_TCMCFG0 [B,H,W]			
0x004	TRCFGn_TCMCFG1 [B,H,W]			
0x008	TRCFGn_TCMUNLOCK [W]			
0x00C	-			
0x010 to 0x01C	*			
0x020	-			
0x024	*			
0x028 to 0x02C	-			
0x030	TRCFGn_TEAR0 [B,H,W]			
0x034	TRCFGn_TEAR1 [B,H,W]			
0x038	TRCFGn_TEAR2 [B,H,W]			
0x03C	TRCFGn_TASAR [B,H,W]		TRCFGn_TAEAR [B,H,W]	
0x040	TRCFGn_TTCR [B,H,W]		TRCFGn_TICR [B,H,W]	TRCFGn_TFECD [B,H,W]
0x044	TRCFGn_TKCCR [B,H,W]	-	-	TRCFGn_TSRCD [B,H,W]
0x048 to 0x3FC	-			



**(10) TCFLASH IF**

TCFLASH IF Unit0 Base\_Address : 0xB041\_1000(n=0)

TCFLASH IF Unit1 Base\_Address : 0xB041\_1400(n=1)

Offset Address	Register			
	+3	+2	+1	+0
0x000	TCFCFGn_FCPROTKEY [W]			
0x004	-			
0x008	TCFCFGn_FCFGR [B,H,W]			
0x00C	-			
0x010	TCFCFGn_FECCCTRL [B,H,W]			
0x014	-			
0x018	TCFCFGn_FDATEIR [B,H,W]			
0x01C	TCFCFGn_FECCEIR [B,H,W]			
0x020	TCFCFGn_FICTRL0 [B,H,W]			
0x024	TCFCFGn_FICTRL1 [B,H,W]			
0x028 to 0x034	-			
0x038	TCFCFGn_FSTAT0 [B,H,W]			
0x03C	TCFCFGn_FSTAT1 [B,H,W]			
0x040 to 0x04C	-			
0x050	TCFCFGn_FSECIR [B,H,W]			
0x054	TCFCFGn_FECCEAR [B,H,W]			
0x058	TCFCFGn_FMIDR [B,H,W]			
0x05C to 0x07C	-			
0x080	TCFCFGn_FUCEDIR [B,H,W]			
0x084	TCFCFGn_FUCEAR [B,H,W]			
0x088 to 0x3FC	-			

**(11) WorkFLASH IF**

WorkFLASH IF Unit00 Base\_Address : 0xB041\_2000(xx=00)

WorkFLASH IF Unit01 Base\_Address : 0xB041\_2400(xx=01)

Offset Address	Register			
	+3	+2	+1	+0
0x000	WFCFGxx_CPR [W]			
0x004	-			
0x008	WFCFGxx_CR [B,H,W]			
0x00C	WFCFGxx_ECR [B,H,W]			
0x010	WFCFGxx_WCR [B,H,W]			
0x014	WFCFGxx_WSR [B,H,W]			
0x018	WFCFGxx_DBEIR [B,H,W]			
0x01C	WFCFGxx_EEIR [B,H,W]			
0x020	-			
0x024	WFCFGxx_ICR [B,H,W]			
0x028	WFCFGxx_SR [B,H,W]			
0x02C	WFCFGxx_SECIR [B,H,W]			
0x030	WFCFGxx_EEAR [B,H,W]			
0x034	WFCFGxx_MIR [B,H,W]			
0x038 to 0x050	-			
0x054	WFCFGxx_SEQCM [B,H,W]			
0x058 to 0x05C	-			
0x060	WFCFGxx_BERR [B,H,W]			
0x064	WFCFGxx_BERRCLR [B,H,W]			
0x068	-			
0x06C	WFCFGxx_UCESR [B,H,W]			
0x070	WFCFGxx_UCEAR [B,H,W]			
0x074 to 0x3FC	-			





## (12) Inter-processor Communication (IPCU)

Inter-processor communication(IPCU) Base\_Address : 0xB041\_5000

Offset Address	Register			
	+3	+2	+1	+0
0x000	IPCU_ISTR0 [B,H,W]			
0x004	IPCU_ISTR1 [B,H,W]			
0x008 to 0x07C	-			
0x080	IPCU_MAR0 [B,H,W]			
0x084	IPCU_MAR1 [B,H,W]			
0x088 to 0x0FC	-			
0x100	IPCU_MB0SRCR [B,H,W]			
0x104	IPCU_MB0MR [B,H,W]			
0x108	IPCU_MB0SR [B,H,W]			
0x10C	-			
0x110	IPCU_MB0DSR [B,H,W]			
0x114	IPCU_MB0DCR [B,H,W]			
0x118	IPCU_MB0DSTR [B,H,W]			
0x11C	-			
0x120	IPCU_MB0MSR [B,H,W]			
0x124	IPCU_MB0MCR [B,H,W]			
0x128	IPCU_MB0MSTR [B,H,W]			
0x12C	-			
0x130	IPCU_MB0ASR [B,H,W]			
0x134	IPCU_MB0ACR [B,H,W]			
0x138	IPCU_MB0ASTR [B,H,W]			
0x13C	IPCU_MB0ASRCR [B,H,W]			
0x140 to 0x160	IPCU_MB0DR0 to IPCU_MB0DR8 [B,H,W]			
0x164 to 0x17C	-			
0x180 to 0x1FC	MailBox1			
0x200 to 0x27C	MailBox2			
0x280 to 0x2FC	MailBox3			
0x300 to 0x37C	MailBox4			
0x380 to 0x3FC	MailBox5			
0x400 to 0x47C	MailBox6			
0x480 to 0x4FC	MailBox7			
0x500 to 0x8FC	-			
0x900	IPCU_MBSTR [B,H,W]			
0x904 to 0xFFC	-			

**Note:**

- MailBox0 is assigned address 0x100 to 0x17C and MailBox1 to MailBox7 are assigned address the same way. Register name is IPCU\_MBmXXX. (m=1 to 7).

### (13) BootROM Hardware Interface

BootROM hardware interface Base\_Address : 0xFFFE\_FC00

Offset Address	Register			
	+3	+2	+1	+0
0x000 to 0x354	-			
0x358	EXCFG_UNLOCK [W]			
0x35C	-			
0x360	EXCFG_CNFG [B,H,W]			
0x364 to 0x380	-			
0x384	EXCFG_UNDEFINACT [B,H,W]			
0x388	EXCFG_SVCINACT [B,H,W]			
0x38C	EXCFG_PABORTINACT [B,H,W]			
0x390	EXCFG_DABORTINACT [B,H,W]			
0x394 to 0x3C0	-			
0x3C4	EXCFG_UNDEFACT [B,H,W]			
0x3C8	EXCFG_SVCACT [B,H,W]			
0x3CC	EXCFG_PABORTACT [B,H,W]			
0x3D0	EXCFG_DABORTACT [B,H,W]			
0x3D4 to 0x3FC	-			

### (14) CRC

CRC ch.0 Base\_Address : 0xB071\_8000(xx=00)

CRC ch.1 Base\_Address : 0xB071\_8400(xx=01)

Offset Address	Register			
	+3	+2	+1	+0
0x000	-		-	CRCxx_CRCCR [B,H,W]
0x004	CRCxx_CRCINIT [B,H,W]			
0x008	CRCxx_CRCIN [B,H,W]			
0x00C	CRCxx_CRCR [B,H,W]			
0x010 to 0x3FC	-			



**(15) CAN(CAN/CAN Controller/CAN\_ECC)**

CAN ch.0 Base\_Address : 0xB072\_0000 (xx=00)

CAN ch.1 Base\_Address : 0xB072\_0400 (xx=01)

CAN ch.2 Base\_Address : 0xB072\_0800 (xx=02)

Offset Address	Register			
	+3	+2	+1	+0
0x000	CANxx_STATR [B,H,W]		CANxx_CTRLR [B,H,W]	
0x004	CANxx_BTR [B,H,W]		CANxx_ERRCNT [B,H,W]	
0x008	CANxx_TESTR [B,H,W]		CANxx_INTR [H,W]	
0x00C	-		CANxx_BRPER [B,H,W]	
0x010	CANxx_IF1CMSK [B,H,W]		CANxx_IF1CREQ [B,H,W]	
0x014	CANxx_IF1MSK2 [B,H,W]		CANxx_IF1MSK1 [B,H,W]	
0x018	CANxx_IF1ARB2 [B,H,W]		CANxx_IF1ARB1 [B,H,W]	
0x01C	-		CANxx_IF1MCTR [B,H,W]	
0x020	CANxx_IF1DTA2 [B,H,W]		CANxx_IF1DTA1 [B,H,W]	
0x024	CANxx_IF1DTB2 [B,H,W]		CANxx_IF1DTB1 [B,H,W]	
0x028 to 0x02C	-			
0x030 to 0x034	Reserved(IF1 data mirror)			
0x038 to 0x03C	-			
0x040	CANxx_IF2CMSK [B,H,W]		CANxx_IF2CREQ [B,H,W]	
0x044	CANxx_IF2MSK2 [B,H,W]		CANxx_IF2MSK1 [B,H,W]	
0x048	CANxx_IF2ARB2 [B,H,W]		CANxx_IF2ARB1 [B,H,W]	
0x04C	-		CANxx_IF2MCTR [B,H,W]	
0x050	CANxx_IF2DTA2 [B,H,W]		CANxx_IF2DTA1 [B,H,W]	
0x054	CANxx_IF2DTB2 [B,H,W]		CANxx_IF2DTB1 [B,H,W]	
0x058 to 0x05C	-			
0x060 to 0x064	Reserved(IF2 data mirror)			
0x068 to 0x07C	-			
0x080	CANxx_TREQ2 [B,H,W]		CANxx_TREQ1 [B,H,W]	
0x084	CANxx_TREQ4 [B,H,W]		CANxx_TREQ3 [B,H,W]	
0x088 to 0x08C	-			
0x090	CANxx_NEWDT2 [B,H,W]		CANxx_NEWDT1 [B,H,W]	
0x094	CANxx_NEWDT4 [B,H,W]		CANxx_NEWDT3 [B,H,W]	
0x098 to 0x09C	-			
0x0A0	CANxx_INTPND2 [B,H,W]		CANxx_INTPND1 [B,H,W]	
0x0A4	CANxx_INTPND4 [B,H,W]		CANxx_INTPND3 [B,H,W]	
0x0A8 to 0x0AC	-			
0x0B0	CANxx_MSGVAL2 [B,H,W]		CANxx_MSGVAL1 [B,H,W]	
0x0B4	CANxx_MSGVAL4 [B,H,W]		CANxx_MSGVAL3 [B,H,W]	
0x0B8 to 0x0BC	-			
0x0C0	CANxx_CANSEEAR [B,H,W]		CANxx_CANEESR [B,H,W]	CANxx_CANEECR [B,H,W]
0x0C4	CANxx_CANDEEAR [B,H,W]		CANxx_CANEESCR [B,H,W]	-
0x0C8	CANxx_CANEFECD [B,H,W]			
0x0CC	-			
0x0D0	-		-	CANxx_CIRRR [B,H,W]
0x0D4 to 0x3FC	-			

### (16) CAN Prescaler

CAN prescaler Base\_Address : 0xB072\_8000

Offset Address	Register			
	+3	+2	+1	+0
0x000	CANP_CANPRE [B,H,W]			
0x004	CANP_CANPCK [B,H,W]			
0x008 to 0x3FC	-			

### (17) CR Calibration

CR calibration Base\_Address : 0xB073\_0000

Offset Address	Register			
	+3	+2	+1	+0
0x000	CU_CUTD1[B,H,W]		CU_CUCR1[B,H,W]	
0x004	CU_CUTR1[B,H,W]			
0x008	CU_CUCRC1[B,H,W]			
0x00C to 0x3FC	-			



**(18) GPIO**

GPIO Base\_Address : 0xB073\_8000

Offset Address	Register			
	+3	+2	+1	+0
0x000				GPIO_POSR0 [B,H,W]
0x004				GPIO_POCR0 [B,H,W]
0x008				GPIO_DDSR0 [B,H,W]
0x00C				GPIO_DDCR0 [B,H,W]
0x010				GPIO_POSR1 [B,H,W]
0x014				GPIO_POCR1 [B,H,W]
0x018				GPIO_DDSR1 [B,H,W]
0x01C				GPIO_DDCR1 [B,H,W]
0x020				GPIO_POSR2 [B,H,W]
0x024				GPIO_POCR2 [B,H,W]
0x028				GPIO_DDSR2 [B,H,W]
0x02C				GPIO_DDCR2 [B,H,W]
0x030				GPIO_POSR3 [B,H,W]
0x034				GPIO_POCR3 [B,H,W]
0x038				GPIO_DDSR3 [B,H,W]
0x03C				GPIO_DDCR3 [B,H,W]
0x040				GPIO_POSR4 [B,H,W]
0x044				GPIO_POCR4 [B,H,W]
0x048				GPIO_DDSR4 [B,H,W]
0x04C				GPIO_DDCR4 [B,H,W]
0x050 to 0x1FC				-
0x200				GPIO_PODR0 [B,H,W]
0x204				GPIO_DDR0 [B,H,W]
0x208				GPIO_PODR1 [B,H,W]
0x20C				GPIO_DDR1 [B,H,W]
0x210				GPIO_PODR2 [B,H,W]
0x214				GPIO_DDR2 [B,H,W]
0x218				GPIO_PODR3 [B,H,W]
0x21C				GPIO_DDR3 [B,H,W]
0x220				GPIO_PODR4 [B,H,W]
0x224				GPIO_DDR4 [B,H,W]
0x228 to 0x2FC				-
0x300				GPIO_PIDR0 [B,H,W]
0x304				GPIO_PIDR1 [B,H,W]
0x308				GPIO_PIDR2 [B,H,W]
0x30C				GPIO_PIDR3 [B,H,W]
0x310				GPIO_PIDR4 [B,H,W]
0x314 to 0x3FC				-
0x400				GPIO_PORTEN [B,H,W]
0x404				GPIO_KEYCDR [W]
0x408 to 0xFFC				-

**(19) PPC**

PPC Base\_Address : 0xB074\_0000

Offset Address	Register			
	+3	+2	+1	+0
0x000	PPC_PCFGR001 [B,H,W]		PPC_PCFGR000 [B,H,W]	
0x004	PPC_PCFGR003 [B,H,W]		PPC_PCFGR002 [B,H,W]	
0x008	PPC_PCFGR005 [B,H,W]		PPC_PCFGR004 [B,H,W]	
0x00C	PPC_PCFGR007 [B,H,W]		PPC_PCFGR006 [B,H,W]	
0x010	PPC_PCFGR009 [B,H,W]		PPC_PCFGR008 [B,H,W]	
0x014	PPC_PCFGR011 [B,H,W]		PPC_PCFGR010 [B,H,W]	
0x018	PPC_PCFGR013 [B,H,W]		PPC_PCFGR012 [B,H,W]	
0x01C	PPC_PCFGR015 [B,H,W]		PPC_PCFGR014 [B,H,W]	
0x020	PPC_PCFGR017 [B,H,W]		PPC_PCFGR016 [B,H,W]	
0x024	PPC_PCFGR019 [B,H,W]		PPC_PCFGR018 [B,H,W]	
0x028	PPC_PCFGR021 [B,H,W]		PPC_PCFGR020 [B,H,W]	
0x02C	PPC_PCFGR023 [B,H,W]		PPC_PCFGR022 [B,H,W]	
0x030	PPC_PCFGR025 [B,H,W]		PPC_PCFGR024 [B,H,W]	
0x034	PPC_PCFGR027 [B,H,W]		PPC_PCFGR026 [B,H,W]	
0x038	PPC_PCFGR029 [B,H,W]		PPC_PCFGR028 [B,H,W]	
0x03C	PPC_PCFGR031 [B,H,W]		PPC_PCFGR030 [B,H,W]	
0x040	PPC_PCFGR101 [B,H,W]		PPC_PCFGR100 [B,H,W]	
0x044	PPC_PCFGR103 [B,H,W]		PPC_PCFGR102 [B,H,W]	
0x048	PPC_PCFGR105 [B,H,W]		PPC_PCFGR104 [B,H,W]	
0x04C	PPC_PCFGR107 [B,H,W]		PPC_PCFGR106 [B,H,W]	
0x050	PPC_PCFGR109 [B,H,W]		PPC_PCFGR108 [B,H,W]	
0x054	PPC_PCFGR111 [B,H,W]		PPC_PCFGR110 [B,H,W]	
0x058	PPC_PCFGR113 [B,H,W]		PPC_PCFGR112 [B,H,W]	
0x05C	PPC_PCFGR115 [B,H,W]		PPC_PCFGR114 [B,H,W]	
0x060	PPC_PCFGR117 [B,H,W]		PPC_PCFGR116 [B,H,W]	
0x064	PPC_PCFGR119 [B,H,W]		PPC_PCFGR118 [B,H,W]	
0x068	PPC_PCFGR121 [B,H,W]		PPC_PCFGR120 [B,H,W]	
0x06C	PPC_PCFGR123 [B,H,W]		PPC_PCFGR122 [B,H,W]	
0x070	PPC_PCFGR125 [B,H,W]		PPC_PCFGR124 [B,H,W]	
0x074	PPC_PCFGR127 [B,H,W]		PPC_PCFGR126 [B,H,W]	
0x078	PPC_PCFGR129 [B,H,W]		PPC_PCFGR128 [B,H,W]	
0x07C	PPC_PCFGR131 [B,H,W]		-	
0x080	PPC_PCFGR201 [B,H,W]		PPC_PCFGR200 [B,H,W]	
0x084	PPC_PCFGR203 [B,H,W]		PPC_PCFGR202 [B,H,W]	
0x088	PPC_PCFGR205 [B,H,W]		PPC_PCFGR204 [B,H,W]	
0x08C	PPC_PCFGR207 [B,H,W]		PPC_PCFGR206 [B,H,W]	
0x090	PPC_PCFGR209 [B,H,W]		PPC_PCFGR208 [B,H,W]	
0x094	PPC_PCFGR211 [B,H,W]		PPC_PCFGR210 [B,H,W]	
0x098	PPC_PCFGR213 [B,H,W]		PPC_PCFGR212 [B,H,W]	
0x09C	PPC_PCFGR215 [B,H,W]		PPC_PCFGR214 [B,H,W]	
0x0A0	PPC_PCFGR217 [B,H,W]		PPC_PCFGR216 [B,H,W]	
0x0A4	PPC_PCFGR219 [B,H,W]		PPC_PCFGR218 [B,H,W]	
0x0A8	PPC_PCFGR221 [B,H,W]		PPC_PCFGR220 [B,H,W]	
0x0AC	PPC_PCFGR223 [B,H,W]		PPC_PCFGR222 [B,H,W]	
0x0B0	PPC_PCFGR225 [B,H,W]		PPC_PCFGR224 [B,H,W]	
0x0B4	PPC_PCFGR227 [B,H,W]		PPC_PCFGR226 [B,H,W]	



H A R D W A R E M A N U A L

Offset Address	Register			
	+3	+2	+1	+0
0x0B8	PPC_PCFGR229 [B,H,W]		PPC_PCFGR228 [B,H,W]	
0x0BC	PPC_PCFGR231 [B,H,W]		PPC_PCFGR230 [B,H,W]	
0x0C0	PPC_PCFGR301 [B,H,W]		PPC_PCFGR300 [B,H,W]	
0x0C4	PPC_PCFGR303 [B,H,W]		PPC_PCFGR302 [B,H,W]	
0x0C8	PPC_PCFGR305 [B,H,W]		PPC_PCFGR304 [B,H,W]	
0x0CC	-		PPC_PCFGR306 [B,H,W]	
0x0D0	PPC_PCFGR309 [B,H,W]		-	
0x0D4	PPC_PCFGR311 [B,H,W]		PPC_PCFGR310 [B,H,W]	
0x0D8	PPC_PCFGR313 [B,H,W]		PPC_PCFGR312 [B,H,W]	
0x0DC	PPC_PCFGR315 [B,H,W]		PPC_PCFGR314 [B,H,W]	
0x0E0	PPC_PCFGR317 [B,H,W]		PPC_PCFGR316 [B,H,W]	
0x0E4	PPC_PCFGR319 [B,H,W]		PPC_PCFGR318 [B,H,W]	
0x0E8	PPC_PCFGR321 [B,H,W]		PPC_PCFGR320 [B,H,W]	
0x0EC	PPC_PCFGR323 [B,H,W]		PPC_PCFGR322 [B,H,W]	
0x0F0	PPC_PCFGR325 [B,H,W]		PPC_PCFGR324 [B,H,W]	
0x0F4	PPC_PCFGR327 [B,H,W]		PPC_PCFGR326 [B,H,W]	
0x0F8	PPC_PCFGR329 [B,H,W]		PPC_PCFGR328 [B,H,W]	
0x0FC	-		PPC_PCFGR330 [B,H,W]	
0x100	-			
0x104	-			
0x108	-			
0x10C	PPC_PCFGR407 [B,H,W]		PPC_PCFGR406 [B,H,W]	
0x110	PPC_PCFGR409 [B,H,W]		PPC_PCFGR408 [B,H,W]	
0x114	PPC_PCFGR411 [B,H,W]		PPC_PCFGR410 [B,H,W]	
0x118	PPC_PCFGR413 [B,H,W]		PPC_PCFGR412 [B,H,W]	
0x11C	PPC_PCFGR415 [B,H,W]		PPC_PCFGR414 [B,H,W]	
0x120	PPC_PCFGR417 [B,H,W]		PPC_PCFGR416 [B,H,W]	
0x124	PPC_PCFGR419 [B,H,W]		PPC_PCFGR418 [B,H,W]	
0x128	PPC_PCFGR421 [B,H,W]		PPC_PCFGR420 [B,H,W]	
0x12C	PPC_PCFGR423 [B,H,W]		PPC_PCFGR422 [B,H,W]	
0x130	PPC_PCFGR425 [B,H,W]		-	
0x134	PPC_PCFGR427 [B,H,W]		PPC_PCFGR426 [B,H,W]	
0x138	PPC_PCFGR429 [B,H,W]		PPC_PCFGR428 [B,H,W]	
0x13C	PPC_PCFGR431 [B,H,W]		PPC_PCFGR430 [B,H,W]	
0x140 to 0x3FC	-			
0x400	PPC_KEYCDR [W]			



(20) RIC

RIC   Base\_Address : 0xB074\_8000

Offset Address	Register			
	+3	+2	+1	+0
0x000	RIC_RESIN1 [B,H,W]		RIC_RESIN0 [B,H,W]	
0x004	RIC_RESIN3 [B,H,W]		RIC_RESIN2 [B,H,W]	
0x008	RIC_RESIN5 [B,H,W]		RIC_RESIN4 [B,H,W]	
0x00C	RIC_RESIN7 [B,H,W]		RIC_RESIN6 [B,H,W]	
0x010	RIC_RESIN 9 [B,H,W]		RIC_RESIN8 [B,H,W]	
0x014	RIC_RESIN 11 [B,H,W]		RIC_RESIN10 [B,H,W]	
0x018 to 0xFFC	-			





### (21) Multi-function Serial Interface

Multi-function serial interface ch.0 Base\_Address : 0xB080\_0000(xx=00)  
Multi-function serial interface ch.1 Base\_Address : 0xB080\_0400(xx=01)  
Multi-function serial interface ch.2 Base\_Address : 0xB080\_0800(xx=02)  
Multi-function serial interface ch.3 Base\_Address : 0xB080\_0C00(xx=03)  
Multi-function serial interface ch.4 Base\_Address : 0xB080\_1000(xx=04)

The notation [\*\*/\*/\*/\*/\*] in below table corresponds to operation mode UART/CSIO/LIN.

Offset Address	Register			
	+3	+2	+1	+0
0x000	MFSxx_SSR [B,H,W]	MFSxx_ESCR [B,H,W]	MFSxx_SCR [B,H,W]	MFSxx_SMR [B,H,W]
0x004	- / MFSxx_RDR3 (MFSxx_TDR3)/ - [B,H,W] <sup>*1</sup>	- / MFSxx_RDR2 (MFSxx_TDR2)/ - [B,H,W] <sup>*1</sup>	MFSxx_RDR1 (MFSxx_TDR1)/ MFSxx_RDR1 (MFSxx_TDR1)/ - [B,H,W] <sup>*1</sup>	MFSxx_RDR0 (MFSxx_TDR0) [B,H,W] <sup>*1</sup>
0x008	MFSxx_STMR1 [B,H,W]	MFSxx_STMR0 [B,H,W]	MFSxx_SACSR1 [B,H,W]	MFSxx_SACSR0 [B,H,W]
0x00C	- / MFSxx_SCSCR1/ MFSxx_SFUR1 [B,H,W]	- / MFSxx_SCSCR0/ MFSxx_SFUR0 [B,H,W]	MFSxx_STMCR1 [B,H,W]	MFSxx_STMCR0 [B,H,W]
0x010	- / MFSxx_SCSTR3/ MFSxx_LAMSR [B,H,W]	- / MFSxx_SCSTR2/ MFSxx_LAMCR [B,H,W]	- / MFSxx_SCSTR1/ MFSxx_SFLR1 [B,H,W]	- / MFSxx_SCSTR0/ MFSxx_SFLR0 [B,H,W]
0x014	-	- / MFSxx_SCSFR2/ - [B,H,W]	- / MFSxx_SCSFR1/ - [B,H,W]	- / MFSxx_SCSFR0/ - [B,H,W]
0x018	- / MFSxx_TBYTE3/ MFSxx_LAMESR [B,H,W] <sup>*2</sup>	- / MFSxx_TBYTE2/ MFSxx_LAMERT [B,H,W] <sup>*2</sup>	- / MFSxx_TBYTE1/ MFSxx_LAMIER [B,H,W] <sup>*3</sup>	MFSxx_TBYTE0/ MFSxx_TBYTE0/ MFSxx_LAMRID (MFSxx_LAMTID) [B,H,W]
0x01C	-	-	MFSxx_BGR1 [H,W]	MFSxx_BGR0 [H,W]
0x020	MFSxx_FBYTE2 [B,H,W]	MFSxx_FBYTE1 [B,H,W]	MFSxx_FCR1 [B,H,W]	MFSxx_FCR0 [B,H,W]
0x024	-	-	MFSxx_FTICR2 [B,H,W]	MFSxx_FTICR1 [B,H,W]
0x028	- / - / MFSxx_SSRC [B,H,W]	- / - / MFSxx_ESCRC [B,H,W]	- / - / MFSxx_SCRC [B,H,W]	- / - / MFSxx_SMRC [B,H,W]
0x02C	-	-	MFSxx_SACSR1C [B,H,W]	MFSxx_SACSR0C [B,H,W]
0x030	- / - / MFSxx_LAMSRC [B,H,W]	- / - / MFSxx_LAMCRC [B,H,W]	-	

Offset Address	Register			
	+3	+2	+1	+0
0x034	- / - / MFSxx_LAMESRC [B,H,W]	-	- / - / MFSxx_LAMIERC [B,H,W]	-
0x038	-	-	MFSxx_FCR1C [B,H,W]	MFSxx_FCR0C [B,H,W]
0x03C	- / - / MFSxx_SSRS [B,H,W]	- / - / MFSxx_ESCRS [B,H,W]	- / - / MFSxx_SCRS [B,H,W]	- / - / MFSxx_SMRS [B,H,W]
0x040	-		MFSxx_SACSR1S [B,H,W]	MFSxx_SACSR0S [B,H,W]
0x044	-	- / - / MFSxx_LAMCRS [B,H,W]	-	
0x048	-		- / - / MFSxx_LAMIERC [B,H,W]	-
0x04C	-		MFSxx_FCR1S [B,H,W]	MFSxx_FCR0S [B,H,W]
0x050 to 0x3FC	-			

\*1: In CSIO mode, access size is [H,W].

\*2: In LIN mode, access size is [B,H].

\*3: In LIN mode, access size is [B].



## (22) Base Timer

Base timer ch.0	Base_Address : 0xB080_8000(xx=00)
Base timer ch.1	Base_Address : 0xB080_8400(xx=01)
Base timer ch.2	Base_Address : 0xB080_8800(xx=02)
Base timer ch.3	Base_Address : 0xB080_8C00(xx=03)
Base timer ch.4	Base_Address : 0xB080_9000(xx=04)
Base timer ch.5	Base_Address : 0xB080_9400(xx=05)
Base timer ch.6	Base_Address : 0xB080_9800(xx=06)
Base timer ch.7	Base_Address : 0xB080_9C00(xx=07)
Base timer ch.8	Base_Address : 0xB080_A000(xx=08)
Base timer ch.9	Base_Address : 0xB080_A400(xx=09)
Base timer ch.10	Base_Address : 0xB080_A800(xx=10)
Base timer ch.11	Base_Address : 0xB080_AC00(xx=11)

The notation [\*\*/\*\*/\*\*/\*\*] in below table corresponds to timer mode reload/PWM/PPG/PWC.

Offset Address	Register			
	+3	+2	+1	+0
0x000	-		BTxx_PCSR / BTxx_PCSR/ BTxx_PRL / - [H]	
0x004	-		- / BTxx_PDUT/ BTxx_PRLH / BTxx_DTBF [H]	
0x008	-		BTxx_TMR / BTxx_TMR/ BTxx_TMR / - [H]	
0x00C	-		BTxx_TMCR [B,H,W]	
0x010	-		BTxx_TMCR2 [B,H,W]	BTxx_STC [B,H,W]
0x014	-		-	BTxx_STCC [B,H,W]
0x018	-		-	BTxx_STCS [B,H,W]
0x01C to 0x02C	-			
0x030 <sup>*1</sup>	-		-	BT_BTSELnm [B,H,W]
0x034 <sup>*2</sup>	-		BT_BTSSSR [B,H,W]	
0x038 to 0x3FC	-			

\*1: There are only even channel number (xx=00, 02, 04, 06, 08, 10), "nm" indicates 01, 23, 45, 67, 89, 1011 in turn.

\*2: There is only channel number 0 (xx=00).

### (23) 32-bit Free-run Timer

32-bit free-run timer ch.0 Base\_Address : 0xB082\_0000(xx=00)  
 32-bit free-run timer ch.1 Base\_Address : 0xB082\_0400(xx=01)  
 32-bit free-run timer ch.2 Base\_Address : 0xB082\_0800(xx=02)  
 32-bit free-run timer ch.3 Base\_Address : 0xB082\_0C00(xx=03)  
 32-bit free-run timer ch.4 Base\_Address : 0xB082\_1000(xx=04)

Offset Address	Register			
	+3	+2	+1	+0
0x000	FRTxx_CPCLRB/FRTxx_CPCLR [W]			
0x004	FRTxx_TCDT [W]			
0x008	FRTxx_TCCS [B,H,W]			
0x00C	FRTxx_TECCS [B,H,W]			
0x010	FRTxx_TCCSC [B,H,W]			
0x014	FRTxx_TCCSS [B,H,W]			
0x018 to 0x3FC	-			

### (24) 32-bit Input Capture

32-bit input capture ch.0 Base\_Address : 0xB082\_8000(xx=00)  
 32-bit input capture ch.1 Base\_Address : 0xB082\_8000(xx=00)  
 32-bit input capture ch.2 Base\_Address : 0xB082\_8400(xx=02)  
 32-bit input capture ch.3 Base\_Address : 0xB082\_8400(xx=02)  
 32-bit input capture ch.4 Base\_Address : 0xB082\_8800(xx=04)  
 32-bit input capture ch.5 Base\_Address : 0xB082\_8800(xx=04)

Offset Address	Register			
	+3	+2	+1	+0
0x000	ICUxx_IPCP0 [W]			
0x004	ICUxx_IPCP1 [W]			
0x008	ICUxx_ICS [B,H,W]			
0x00C	ICUxx_ICSC [B,H,W]			
0x010	ICUxx_ICSS [B,H,W]			
0x014 to 0x3FC	-			



**(25) DMAC**

DMAC Base\_Address : 0xB070\_0000

Offset Address	Register			
	+3	+2	+1	+0
0x0000+(n×0x40)	DMA0_An [B,H,W]			
0x0004+(n×0x40)	DMA0_Bn [B,H,W]			
0x0008+(n×0x40)	DMA0_SAn [B,H,W]			
0x000C+(n×0x40)	DMA0_DAn [B,H,W]			
0x0010+(n×0x40)	DMA0_Cn [B,H,W]			
0x0014+(n×0x40)	DMA0_Dn [B]			
0x0018+(n×0x40)	DMA0_SASHDWn [B,H,W]			
0x001C+(n×0x40)	DMA0_DASHDWn [B,H,W]			
0x0020+(n×0x40)	DMA0_En [B,H,W]			
0x0024+(n×0x40) to 0x003C+(n×0x40)	-			
0x0400 to 0x0FFC	-			
0x1000	DMA0_R [B,H,W]			
0x1004	DMA0_DIRQ1 [B,H,W]			
0x1008	DMA0_DIRQ2 [B,H,W]			
0x100C	DMA0_EDIRQ1 [B,H,W]			
0x1010	DMA0_EDIRQ2 [B,H,W]			
0x1014	DMA0_ID [B,H,W]			
0x1018 to 0x201C	-			
0x2020+((m-8)×0x04)	DMA0_CMICICm [B,H,W]			
0x223C to 0x27FC	-			
0x2800+(n×0x04)	DMA0_CMCHICn [B,H,W]			
0x2840 to 0x3FFC	-			

n=0 to 15 (DMA channel)

m=8 to 142 (client interface channel)

**(26) Memory Protection Unit (MPU)**

Memory protection unit (MPU) Unit0 Base\_Address : 0xB071\_0000

Offset Address	Register			
	+3	+2	+1	+0
0x000				MPUH0_CTRL0 [B,H,W]
0x004				MPUH0_NMIEN [B,H,W]
0x008				MPUH0_MERRC [B,H,W]
0x00C				MPUH0_MERRA [B,H,W]
0x010				MPUH0_CTRL1 [B,H,W]
0x014				MPUH0_SADDR1 [B,H,W]
0x018				MPUH0_EADDR1 [B,H,W]
0x01C				MPUH0_CTRL2 [B,H,W]
0x020				MPUH0_SADDR2 [B,H,W]
0x024				MPUH0_EADDR2 [B,H,W]
0x028				MPUH0_CTRL3 [B,H,W]
0x02C				MPUH0_SADDR3 [B,H,W]
0x030				MPUH0_EADDR3 [B,H,W]
0x034				MPUH0_CTRL4 [B,H,W]
0x038				MPUH0_SADDR4 [B,H,W]
0x03C				MPUH0_EADDR4 [B,H,W]
0x040				MPUH0_CTRL5 [B,H,W]
0x044				MPUH0_SADDR5 [B,H,W]
0x048				MPUH0_EADDR5 [B,H,W]
0x04C				MPUH0_CTRL6 [B,H,W]
0x050				MPUH0_SADDR6 [B,H,W]
0x054				MPUH0_EADDR6 [B,H,W]
0x058				MPUH0_CTRL7 [B,H,W]
0x05C				MPUH0_SADDR7 [B,H,W]
0x060				MPUH0_EADDR7 [B,H,W]
0x064				MPUH0_CTRL8 [B,H,W]
0x068				MPUH0_SADDR8 [B,H,W]
0x06C				MPUH0_EADDR8 [B,H,W]
0x070				MPUH0_UNLOCK [W]
0x074				MPUH0_MID [B,H,W]
0x078 to 0xFFC				-



### (27) BootROM Marker

Security Description Record (SDR) Base\_Address :

- Offset in TCM area 0x007F\_0000
- Offset in AXI area 0x00FF\_0000

In below table, SA0 indicates sector 0 in TCFLASH Flash memory A and SA1 indicates sector 1 in TCFLASH Flash memory A.

Offset Address	Register							
	SA1				SA0			
	+7	+6	+5	+4	+3	+2	+1	+0
0x000	-				SDR_FSECM [B,H,W]			
0x008	-				SDR_DSM [B,H,W]			
0x010	-				SDR_DSKM0 [B,H,W]			
0x018	-				SDR_DSKM1 [B,H,W]			
0x020	-				SDR_DSKM2 [B,H,W]			
0x028	-				SDR_DSKM3 [B,H,W]			
0x030 to 0x038	-							

Boot Description Record (BDR) Base\_Address :

- Offset in TCM area 0x007F\_0040
- Offset in AXI area 0x00FF\_0040

In below table, SA0 indicates sector 0 in TCFLASH Flash memory A and SA1 indicates sector 1 in TCFLASH Flash memory A.

Offset Address	Register							
	SA1				SA0			
	+7	+6	+5	+4	+3	+2	+1	+0
0x000	-				BDR_DWEM [B,H,W]			
0x008	-				BDR_ABVM [B,H,W]			
0x010	-				BDR_ABVEM [B,H,W]			
0x018	-							

Watchdog Description Record (WDR) Base\_Address :

- Offset in TCM area 0x007F\_0060
- Offset in AXI area 0x00FF\_0060

In below table, SA0 indicates sector 0 in TCFLASH Flash memory A and SA1 indicates sector 1 in TCFLASH Flash memory A.

Offset Address	Register							
	SA1				SA0			
	+7	+6	+5	+4	+3	+2	+1	+0
0x000	-				WDR_INTM [B,H,W]			
0x008	-				WDR_TRG0CFGM [B,H,W]			
0x010	-				WDR_TRG1CFGM [B,H,W]			
0x018	-				WDR_RUNLLM [B,H,W]			
0x020	-				WDR_RUNULM [B,H,W]			
0x028	-				WDR_PSSLLM [B,H,W]			
0x030	-				WDR_PSSULM [B,H,W]			
0x038	-				WDR_RSTDLYM [B,H,W]			
0x040	-				WDR_CFGM [B,H,W]			
0x048	-				WDR_CEM [B,H,W]			
0x050 to 0x098	-							





### (28) 16-bit Free-run Timer

16-bit free-run timer ch.0	Base_Address : 0xB200_0000 (xx=00)
16-bit free-run timer ch.1	Base_Address : 0xB200_0010 (xx=01)
16-bit free-run timer ch.2	Base_Address : 0xB200_0020 (xx=02)
16-bit free-run timer ch.3	Base_Address : 0xB200_0030 (xx=03)
16-bit free-run timer ch.4	Base_Address : 0xB200_0040 (xx=04)
16-bit free-run timer ch.5	Base_Address : 0xB200_0050 (xx=05)
16-bit free-run timer ch.6	Base_Address : 0xB100_0000 (xx=06)
16-bit free-run timer ch.7	Base_Address : 0xB100_0010 (xx=07)
16-bit free-run timer ch.8	Base_Address : 0xB100_0020 (xx=08)
16-bit free-run timer ch.9	Base_Address : 0xB100_0030 (xx=09)
16-bit free-run timer ch.10	Base_Address : 0xB100_0040 (xx=10)
16-bit free-run timer ch.11	Base_Address : 0xB100_0050 (xx=11)
16-bit free-run timer ch.12	Base_Address : 0xB101_0000 (xx=12)
16-bit free-run timer ch.13	Base_Address : 0xB101_0010 (xx=13)
16-bit free-run timer ch.14	Base_Address : 0xB101_0020 (xx=14)
16-bit free-run timer ch.15	Base_Address : 0xB101_0030 (xx=15)
16-bit free-run timer ch.16	Base_Address : 0xB101_0040 (xx=16)
16-bit free-run timer ch.17	Base_Address : 0xB101_0050 (xx=17)
16-bit free-run timer ch.18	Base_Address : 0xB200_0060 (xx=18)
16-bit free-run timer ch.19	Base_Address : 0xB100_0060 (xx=19)

Offset Address	Register			
	+3	+2	+1	+0
0x000	FRT16Bxx_CPCLRB/FRT16Bxx_CPCLR [H,W]		FRT16Bxx_TCDT [H,W]	
0x004	FRT16Bxx_TCCS [B,H,W]			
0x008	FRT16Bxx_TCCSC [B,H,W]			
0x00C	FRT16Bxx_TCCSS [B,H,W]			

### (29) Free-run Timer Simultaneous Activation

Free-run timer simultaneous activation 0 Base\_Address: 0xB200\_0070

Offset Address	Register			
	+3	+2	+1	+0
0x000	FRSS00_TCGS [B,H,W]	-	FRSS00_TCGSE [B,H,W]	
0x004 to 0x00C	-			

Free-run timer simultaneous activation 1 Base\_Address: 0xB100\_0070

Offset Address	Register			
	+3	+2	+1	+0
0x000	FRSS01_TCGS [B,H,W]	-	FRSS01_TCGSE [B,H,W]	
0x004	-		-	FRSS01_TCGSS [B,H,W]
0x008 to 0x00C	-			

Free-run timer simultaneous activation 2 Base\_Address: 0xB101\_0070

Offset Address	Register			
	+3	+2	+1	+0
0x000	FRSS02_TCGS [B,H,W]	-	FRSS02_TCGSE [B,H,W]	
0x004 to 0x00C	-			

### (30) Free-run Timer Selection

Free-run timer selection 0 Base\_Address : 0xB200\_0080(xx=00)

Free-run timer selection 1 Base\_Address : 0xB100\_0080(xx=01)

Offset Address	Register			
	+3	+2	+1	+0
0x000	FRSELxx_FRS0 [B,H,W]			
0x004	FRSELxx_FRS1 [B,H,W]			
0x008	-			
0x00C	-			
0x010	FRSELxx_FRS4 [B,H,W]			
0x014	-			
0x018	-			
0x01C	-			

Free-run timer selection 2 Base\_Address : 0xB101\_0080

Offset Address	Register			
	+3	+2	+1	+0
0x000	FRSEL02_FRS0 [B,H,W]			
0x004	FRSEL02_FRS1 [B,H,W]			
0x008	FRSEL02_FRS2 [B,H,W]			
0x00C	FRSEL02_FRS3 [B,H,W]			
0x010	FRSEL02_FRS4 [B,H,W]			
0x014	FRSEL02_FRS5 [B,H,W]			
0x018	FRSEL02_FRS6 [B,H,W]			
0x01C	FRSEL02_FRS7 [B,H,W]			

### (31) Indication Free-run Timer Count Direction

Indication free-run timer count direction 0 Base\_Address : 0xB200\_00A0 (xx=00)

Indication free-run timer count direction 1 Base\_Address : 0xB100\_00A0 (xx=01)

Indication free-run timer count direction 2 Base\_Address : 0xB101\_00A0 (xx=02)

Offset Address	Register			
	+3	+2	+1	+0
0x000	-		-	FRCDxx_FRTCDD [B,H,W]



### (32) 16-bit Output Compare

16-bit output compare ch.0 Base\_Address : 0xB200\_0100(xx=00)  
 16-bit output compare ch.1 Base\_Address : 0xB200\_0100(xx=00)  
 16-bit output compare ch.2 Base\_Address : 0xB200\_0110(xx=02)  
 16-bit output compare ch.3 Base\_Address : 0xB200\_0110(xx=02)  
 16-bit output compare ch.4 Base\_Address : 0xB200\_0120(xx=04)  
 16-bit output compare ch.5 Base\_Address : 0xB200\_0120(xx=04)  
 16-bit output compare ch.6 Base\_Address : 0xB100\_0100(xx=06)  
 16-bit output compare ch.7 Base\_Address : 0xB100\_0100(xx=06)  
 16-bit output compare ch.8 Base\_Address : 0xB100\_0110(xx=08)  
 16-bit output compare ch.9 Base\_Address : 0xB100\_0110(xx=08)  
 16-bit output compare ch.10 Base\_Address : 0xB100\_0120(xx=10)  
 16-bit output compare ch.11 Base\_Address : 0xB100\_0120(xx=10)  
 16-bit output compare ch.12 Base\_Address : 0xB101\_0100(xx=12)  
 16-bit output compare ch.13 Base\_Address : 0xB101\_0100(xx=12)  
 16-bit output compare ch.14 Base\_Address : 0xB101\_0110(xx=14)  
 16-bit output compare ch.15 Base\_Address : 0xB101\_0110(xx=14)  
 16-bit output compare ch.16 Base\_Address : 0xB101\_0120(xx=16)  
 16-bit output compare ch.17 Base\_Address : 0xB101\_0120(xx=16)  
 16-bit output compare ch.18 Base\_Address : 0xB101\_0130(xx=18)  
 16-bit output compare ch.19 Base\_Address : 0xB101\_0130(xx=18)  
 16-bit output compare ch.20 Base\_Address : 0xB101\_0140(xx=20)  
 16-bit output compare ch.21 Base\_Address : 0xB101\_0140(xx=20)  
 16-bit output compare ch.22 Base\_Address : 0xB101\_0150(xx=22)  
 16-bit output compare ch.23 Base\_Address : 0xB101\_0150(xx=22)

Offset Address	Register			
	+3	+2	+1	+0
0x000	OCU16Bxx_OCCPB0/OCU16Bxx_OCCP0 [H,W]		OCU16Bxx_OCCPB1/OCU16Bxx_OCCP1 [H,W]	
0x004	OCU16Bxx_OCS [B,H,W]		OCU16Bxx_OCMOD [B,H,W]	
0x008	OCU16Bxx_OCSC [B,H,W]		-	
0x00C	OCU16Bxx_OCSS [B,H,W]		-	

### (33) 16-bit Input Capture

16-bit input capture ch.0 Base\_Address : 0xB200\_0200(xx=00)  
 16-bit input capture ch.1 Base\_Address : 0xB200\_0200(xx=00)  
 16-bit input capture ch.2 Base\_Address : 0xB200\_0210(xx=02)  
 16-bit input capture ch.3 Base\_Address : 0xB200\_0210(xx=02)  
 16-bit input capture ch.4 Base\_Address : 0xB100\_0200(xx=04)  
 16-bit input capture ch.5 Base\_Address : 0xB100\_0200(xx=04)  
 16-bit input capture ch.6 Base\_Address : 0xB100\_0210(xx=06)  
 16-bit input capture ch.7 Base\_Address : 0xB100\_0210(xx=06)  
 16-bit input capture ch.8 Base\_Address : 0xB101\_0200(xx=08)  
 16-bit input capture ch.9 Base\_Address : 0xB101\_0200(xx=08)  
 16-bit input capture ch.10 Base\_Address : 0xB101\_0210(xx=10)  
 16-bit input capture ch.11 Base\_Address : 0xB101\_0210(xx=10)  
 16-bit input capture ch.12 Base\_Address : 0xB101\_0220(xx=12)  
 16-bit input capture ch.13 Base\_Address : 0xB101\_0220(xx=12)  
 16-bit input capture ch.14 Base\_Address : 0xB101\_0230(xx=14)

Offset Address	Register			
	+3	+2	+1	+0
0x000	ICU16Bxx_IPCP0 [H,W]		ICU16Bxx_IPCP1 [H,W]	
0x004	ICU16Bxx_ICS [H,W]		-	
0x008	ICU16Bxx_ICSC [H,W]		-	
0x00C	ICU16Bxx_ICSS [H,W]		-	



**(34) 4ch Sample Hold 12-bit A/D Converter**

4ch sample hold 12-bit A/D converter unit00 Base\_Address : 0xB200\_0300 (xx=00)

4ch sample hold 12-bit A/D converter unit01 Base\_Address : 0xB100\_0300 (xx=01)

Offset Address	Register			
	+3	+2	+1	+0
0x000	-		-	ADC4SHxx_ADTSS [B,H,W]
0x004	-		-	ADC4SHxx_ADTSE [B,H,W]
0x008	ADC4SHxx_ADCOMP0/ADC4SHxx_ADCOMP0 [H,W]		ADC4SHxx_ADCOMP1/ADC4SHxx_ADCOMP1 [H,W]	
0x00C	ADC4SHxx_ADCOMP2/ADC4SHxx_ADCOMP2 [H,W]		ADC4SHxx_ADCOMP3/ADC4SHxx_ADCOMP3 [H,W]	
0x010	ADC4SHxx_ADCOMP4/ADC4SHxx_ADCOMP4 [H,W]		ADC4SHxx_ADCOMP5/ADC4SHxx_ADCOMP5 [H,W]	
0x014	ADC4SHxx_ADCOMP6/ADC4SHxx_ADCOMP6 [H,W]		ADC4SHxx_ADCOMP7/ADC4SHxx_ADCOMP7 [H,W]	
0x018	ADC4SHxx_ADTC0 [B,H,W]		ADC4SHxx_ADTC1 [B,H,W]	
0x01C	ADC4SHxx_ADTC2 [B,H,W]		ADC4SHxx_ADTC3 [B,H,W]	
0x020	ADC4SHxx_ADTC4 [B,H,W]		ADC4SHxx_ADTC5 [B,H,W]	
0x024	ADC4SHxx_ADTC6 [B,H,W]		ADC4SHxx_ADTC7 [B,H,W]	
0x028	ADC4SHxx_ADTC0 [B,H,W]		ADC4SHxx_ADTC1 [B,H,W]	
0x02C	ADC4SHxx_ADTC2 [B,H,W]		ADC4SHxx_ADTC3 [B,H,W]	
0x030	ADC4SHxx_ADTC4 [B,H,W]		ADC4SHxx_ADTC5 [B,H,W]	
0x034	ADC4SHxx_ADTC6 [B,H,W]		ADC4SHxx_ADTC7 [B,H,W]	
0x038	ADC4SHxx_ADRCUT0 [B,H,W]		ADC4SHxx_ADRCUT1 [B,H,W]	
0x03C	ADC4SHxx_ADRCUT2 [B,H,W]		ADC4SHxx_ADRCUT3 [B,H,W]	
0x040	ADC4SHxx_ADRCUT4 [B,H,W]		ADC4SHxx_ADRCUT5 [B,H,W]	
0x044	ADC4SHxx_ADRCUT6 [B,H,W]		ADC4SHxx_ADRCUT7 [B,H,W]	
0x048	ADC4SHxx_ADRCSS0 [B,H,W]	ADC4SHxx_ADRCSS1 [B,H,W]	ADC4SHxx_ADRCSS2 [B,H,W]	ADC4SHxx_ADRCSS3 [B,H,W]
0x04C	ADC4SHxx_ADRCSS4 [B,H,W]	ADC4SHxx_ADRCSS5 [B,H,W]	ADC4SHxx_ADRCSS6 [B,H,W]	ADC4SHxx_ADRCSS7 [B,H,W]
0x050	-		-	ADC4SHxx_ADRCOT [B,H,W]
0x054	-		-	ADC4SHxx_ADRCIF [B,H,W]
0x058	-		-	ADC4SHxx_ADPRTF [B,H,W]
0x05C	ADC4SHxx_ADTS0 [B,H,W]		ADC4SHxx_ADTS1 [B,H,W]	
0x060	ADC4SHxx_ADTS2 [B,H,W]		ADC4SHxx_ADTS3 [B,H,W]	
0x064	ADC4SHxx_ADTS4 [B,H,W]		ADC4SHxx_ADTS5 [B,H,W]	
0x068	ADC4SHxx_ADTS6 [B,H,W]		ADC4SHxx_ADTS7 [B,H,W]	
0x06C	-		-	ADC4SHxx_ADRCIFC [B,H,W]
0x070	ADC4SHxx_ADTS0 [B,H,W]		ADC4SHxx_ADTS1 [B,H,W]	
0x074	ADC4SHxx_ADTS2 [B,H,W]		ADC4SHxx_ADTS3 [B,H,W]	
0x078	ADC4SHxx_ADTS4 [B,H,W]		ADC4SHxx_ADTS5 [B,H,W]	
0x07C	ADC4SHxx_ADTS6 [B,H,W]		ADC4SHxx_ADTS7 [B,H,W]	
0x080 to 0x08C	-			

Offset Address	Register			
	+3	+2	+1	+0
0x090	ADC4SHxx_ADCS [B,H,W]	ADC4SHxx_ADCH [B,H,W]	ADC4SHxx_ADMD [B,H,W]	
0x094	-		-	ADC4SHxx_ADCHSEL [B,H,W]
0x098 to 0x0A0	-			
0x0A4	-		-	ADC4SHxx_ADCEN [B,H,W]
0x0A8	ADC4SHxx_ADCEMTE [W]			
0x0AC to 0x0FC	-			



**(35) 12-bit A/D Converter**

12-bit A/D converter (ch.0 to ch.31) Base\_Address : 0xB101\_0400

Offset Address	Register			
	+3	+2	+1	+0
0x000	-			
0x004	ADC12B_ADTSS [B,H,W]	-	-	
0x008	ADC12B_ADTSE [B,H,W]			
0x00C	ADC12B_ADCOMP0/ADC12B_ADCOMP0 [H,W]		ADC12B_ADCOMP1/ADC12B_ADCOMP1 [H,W]	
0x010	ADC12B_ADCOMP2/ADC12B_ADCOMP2 [H,W]		ADC12B_ADCOMP3/ADC12B_ADCOMP3 [H,W]	
0x014	ADC12B_ADCOMP4/ADC12B_ADCOMP4 [H,W]		ADC12B_ADCOMP5/ADC12B_ADCOMP5 [H,W]	
0x018	ADC12B_ADCOMP6/ADC12B_ADCOMP6 [H,W]		ADC12B_ADCOMP7/ADC12B_ADCOMP7 [H,W]	
0x01C	ADC12B_ADCOMP8/ADC12B_ADCOMP8 [H,W]		ADC12B_ADCOMP9/ADC12B_ADCOMP9 [H,W]	
0x020	ADC12B_ADCOMP10/ADC12B_ADCOMP10 [H,W]		ADC12B_ADCOMP11/ADC12B_ADCOMP11 [H,W]	
0x024	ADC12B_ADCOMP12/ADC12B_ADCOMP12 [H,W]		ADC12B_ADCOMP13/ADC12B_ADCOMP13 [H,W]	
0x028	ADC12B_ADCOMP14/ADC12B_ADCOMP14 [H,W]		ADC12B_ADCOMP15/ADC12B_ADCOMP15 [H,W]	
0x02C	ADC12B_ADCOMP16/ADC12B_ADCOMP16 [H,W]		ADC12B_ADCOMP17/ADC12B_ADCOMP17 [H,W]	
0x030	ADC12B_ADCOMP18/ADC12B_ADCOMP18 [H,W]		ADC12B_ADCOMP19/ADC12B_ADCOMP19 [H,W]	
0x034	ADC12B_ADCOMP20/ADC12B_ADCOMP20 [H,W]		ADC12B_ADCOMP21/ADC12B_ADCOMP21 [H,W]	
0x038	ADC12B_ADCOMP22/ADC12B_ADCOMP22 [H,W]		ADC12B_ADCOMP23/ADC12B_ADCOMP23 [H,W]	
0x03C	ADC12B_ADCOMP24/ADC12B_ADCOMP24 [H,W]		ADC12B_ADCOMP25/ADC12B_ADCOMP25 [H,W]	
0x040	ADC12B_ADCOMP26/ADC12B_ADCOMP26 [H,W]		ADC12B_ADCOMP27/ADC12B_ADCOMP27 [H,W]	
0x044	ADC12B_ADCOMP28/ADC12B_ADCOMP28 [H,W]		ADC12B_ADCOMP29/ADC12B_ADCOMP29 [H,W]	
0x048	ADC12B_ADCOMP30/ADC12B_ADCOMP30 [H,W]		ADC12B_ADCOMP31/ADC12B_ADCOMP31 [H,W]	
0x04C	ADC12B_ADTCS0 [B,H,W]		ADC12B_ADTCS1 [B,H,W]	
0x050	ADC12B_ADTCS2 [B,H,W]		ADC12B_ADTCS3 [B,H,W]	
0x054	ADC12B_ADTCS4 [B,H,W]		ADC12B_ADTCS5 [B,H,W]	
0x058	ADC12B_ADTCS6 [B,H,W]		ADC12B_ADTCS7 [B,H,W]	
0x05C	ADC12B_ADTCS8 [B,H,W]		ADC12B_ADTCS9 [B,H,W]	
0x060	ADC12B_ADTCS10 [B,H,W]		ADC12B_ADTCS11 [B,H,W]	
0x064	ADC12B_ADTCS12 [B,H,W]		ADC12B_ADTCS13 [B,H,W]	
0x068	ADC12B_ADTCS14 [B,H,W]		ADC12B_ADTCS15 [B,H,W]	
0x06C	ADC12B_ADTCS16 [B,H,W]		ADC12B_ADTCS17 [B,H,W]	
0x070	ADC12B_ADTCS18 [B,H,W]		ADC12B_ADTCS19 [B,H,W]	
0x074	ADC12B_ADTCS20 [B,H,W]		ADC12B_ADTCS21 [B,H,W]	
0x078	ADC12B_ADTCS22 [B,H,W]		ADC12B_ADTCS23 [B,H,W]	
0x07C	ADC12B_ADTCS24 [B,H,W]		ADC12B_ADTCS25 [B,H,W]	
0x080	ADC12B_ADTCS26 [B,H,W]		ADC12B_ADTCS27 [B,H,W]	
0x084	ADC12B_ADTCS28 [B,H,W]		ADC12B_ADTCS29 [B,H,W]	
0x088	ADC12B_ADTCS30 [B,H,W]		ADC12B_ADTCS31 [B,H,W]	
0x08C	ADC12B_ADTCD0 [B,H,W]		ADC12B_ADTCD1 [B,H,W]	
0x090	ADC12B_ADTCD2 [B,H,W]		ADC12B_ADTCD3 [B,H,W]	
0x094	ADC12B_ADTCD4 [B,H,W]		ADC12B_ADTCD5 [B,H,W]	
0x098	ADC12B_ADTCD6 [B,H,W]		ADC12B_ADTCD7 [B,H,W]	
0x09C	ADC12B_ADTCD8 [B,H,W]		ADC12B_ADTCD9 [B,H,W]	
0x0A0	ADC12B_ADTCD10 [B,H,W]		ADC12B_ADTCD11 [B,H,W]	
0x0A4	ADC12B_ADTCD12 [B,H,W]		ADC12B_ADTCD13 [B,H,W]	
0x0A8	ADC12B_ADTCD14 [B,H,W]		ADC12B_ADTCD15 [B,H,W]	
0x0AC	ADC12B_ADTCD16 [B,H,W]		ADC12B_ADTCD17 [B,H,W]	
0x0B0	ADC12B_ADTCD18 [B,H,W]		ADC12B_ADTCD19 [B,H,W]	
0x0B4	ADC12B_ADTCD20 [B,H,W]		ADC12B_ADTCD21 [B,H,W]	

Offset Address	Register			
	+3	+2	+1	+0
0x0B8	ADC12B_ADTCD22 [B,H,W]		ADC12B_ADTCD23 [B,H,W]	
0x0BC	ADC12B_ADTCD24 [B,H,W]		ADC12B_ADTCD25 [B,H,W]	
0x0C0	ADC12B_ADTCD26 [B,H,W]		ADC12B_ADTCD27 [B,H,W]	
0x0C4	ADC12B_ADTCD28 [B,H,W]		ADC12B_ADTCD29 [B,H,W]	
0x0C8	ADC12B_ADTCD30 [B,H,W]		ADC12B_ADTCD31 [B,H,W]	
0x0CC	ADC12B_ADTECS0 [B,H,W]		ADC12B_ADTECS1 [B,H,W]	
0x0D0	ADC12B_ADTECS2 [B,H,W]		ADC12B_ADTECS3 [B,H,W]	
0x0D4	ADC12B_ADTECS4 [B,H,W]		ADC12B_ADTECS5 [B,H,W]	
0x0D8	ADC12B_ADTECS6 [B,H,W]		ADC12B_ADTECS7 [B,H,W]	
0x0DC	ADC12B_ADTECS8 [B,H,W]		ADC12B_ADTECS9 [B,H,W]	
0x0E0	ADC12B_ADTECS10 [B,H,W]		ADC12B_ADTECS11 [B,H,W]	
0x0E4	ADC12B_ADTECS12 [B,H,W]		ADC12B_ADTECS13 [B,H,W]	
0x0E8	ADC12B_ADTECS14 [B,H,W]		ADC12B_ADTECS15 [B,H,W]	
0x0EC	ADC12B_ADTECS16 [B,H,W]		ADC12B_ADTECS17 [B,H,W]	
0x0F0	ADC12B_ADTECS18 [B,H,W]		ADC12B_ADTECS19 [B,H,W]	
0x0F4	ADC12B_ADTECS20 [B,H,W]		ADC12B_ADTECS21 [B,H,W]	
0x0F8	ADC12B_ADTECS22 [B,H,W]		ADC12B_ADTECS23 [B,H,W]	
0x0FC	ADC12B_ADTECS24 [B,H,W]		ADC12B_ADTECS25 [B,H,W]	
0x100	ADC12B_ADTECS26 [B,H,W]		ADC12B_ADTECS27 [B,H,W]	
0x104	ADC12B_ADTECS28 [B,H,W]		ADC12B_ADTECS29 [B,H,W]	
0x108	ADC12B_ADTECS30 [B,H,W]		ADC12B_ADTECS31 [B,H,W]	
0x10C	ADC12B_ADRCLT0 [B,H,W]		ADC12B_ADRCLT1 [B,H,W]	
0x110	ADC12B_ADRCLT2 [B,H,W]		ADC12B_ADRCLT3 [B,H,W]	
0x114	ADC12B_ADRCCS0 [B,H,W]		ADC12B_ADRCCS1 [B,H,W]	
0x118	ADC12B_ADRCCS2 [B,H,W]		ADC12B_ADRCCS3 [B,H,W]	
0x11C	ADC12B_ADRCCS4 [B,H,W]	ADC12B_ADRCCS5 [B,H,W]	ADC12B_ADRCCS6 [B,H,W]	ADC12B_ADRCCS7 [B,H,W]
0x120	ADC12B_ADRCCS8 [B,H,W]	ADC12B_ADRCCS9 [B,H,W]	ADC12B_ADRCCS10 [B,H,W]	ADC12B_ADRCCS11 [B,H,W]
0x124	ADC12B_ADRCCS12 [B,H,W]	ADC12B_ADRCCS13 [B,H,W]	ADC12B_ADRCCS14 [B,H,W]	ADC12B_ADRCCS15 [B,H,W]
0x128	ADC12B_ADRCCS16 [B,H,W]	ADC12B_ADRCCS17 [B,H,W]	ADC12B_ADRCCS18 [B,H,W]	ADC12B_ADRCCS19 [B,H,W]
0x12C	ADC12B_ADRCCS20 [B,H,W]	ADC12B_ADRCCS21 [B,H,W]	ADC12B_ADRCCS22 [B,H,W]	ADC12B_ADRCCS23 [B,H,W]
0x130	ADC12B_ADRCCS24 [B,H,W]	ADC12B_ADRCCS25 [B,H,W]	ADC12B_ADRCCS26 [B,H,W]	ADC12B_ADRCCS27 [B,H,W]
0x134	ADC12B_ADRCCS28 [B,H,W]	ADC12B_ADRCCS29 [B,H,W]	ADC12B_ADRCCS30 [B,H,W]	ADC12B_ADRCCS31 [B,H,W]
0x138	ADC12B_ADRCCS32 [B,H,W]			
0x13C	ADC12B_ADRCCS33 [B,H,W]			
0x140	ADC12B_ADRCCS34 [B,H,W]			
0x144	ADC12B_ADRCCS35 [B,H,W]	-	-	
0x148	ADC12B_ADRCCS36 [B,H,W]	ADC12B_ADRCCS37 [B,H,W]	ADC12B_ADRCCS38 [B,H,W]	ADC12B_ADRCCS39 [B,H,W]
0x14C	ADC12B_ADRCCS40 [B,H,W]	ADC12B_ADRCCS41 [B,H,W]	ADC12B_ADRCCS42 [B,H,W]	ADC12B_ADRCCS43 [B,H,W]





H A R D W A R E M A N U A L

Offset Address	Register			
	+3	+2	+1	+0
0x150	ADC12B_ADNCS8 [B,H,W]	ADC12B_ADNCS9 [B,H,W]	ADC12B_ADNCS10 [B,H,W]	ADC12B_ADNCS11 [B,H,W]
0x154	ADC12B_ADNCS12 [B,H,W]	ADC12B_ADNCS13 [B,H,W]	ADC12B_ADNCS14 [B,H,W]	ADC12B_ADNCS15 [B,H,W]
0x158	ADC12B_ADPRTF [B,H,W]			
0x15C	ADC12B_ADEOCF [B,H,W]			
0x160	ADC12B_ADTCS0 [B,H,W]		ADC12B_ADTCS1 [B,H,W]	
0x164	ADC12B_ADTCS2 [B,H,W]		ADC12B_ADTCS3 [B,H,W]	
0x168	ADC12B_ADTCS4 [B,H,W]		ADC12B_ADTCS5 [B,H,W]	
0x16C	ADC12B_ADTCS6 [B,H,W]		ADC12B_ADTCS7 [B,H,W]	
0x170	ADC12B_ADTCS8 [B,H,W]		ADC12B_ADTCS9 [B,H,W]	
0x174	ADC12B_ADTCS10 [B,H,W]		ADC12B_ADTCS11 [B,H,W]	
0x178	ADC12B_ADTCS12 [B,H,W]		ADC12B_ADTCS13 [B,H,W]	
0x17C	ADC12B_ADTCS14 [B,H,W]		ADC12B_ADTCS15 [B,H,W]	
0x180	ADC12B_ADTCS16 [B,H,W]		ADC12B_ADTCS17 [B,H,W]	
0x184	ADC12B_ADTCS18 [B,H,W]		ADC12B_ADTCS19 [B,H,W]	
0x188	ADC12B_ADTCS20 [B,H,W]		ADC12B_ADTCS21 [B,H,W]	
0x18C	ADC12B_ADTCS22 [B,H,W]		ADC12B_ADTCS23 [B,H,W]	
0x190	ADC12B_ADTCS24 [B,H,W]		ADC12B_ADTCS25 [B,H,W]	
0x194	ADC12B_ADTCS26 [B,H,W]		ADC12B_ADTCS27 [B,H,W]	
0x198	ADC12B_ADTCS28 [B,H,W]		ADC12B_ADTCS29 [B,H,W]	
0x19C	ADC12B_ADTCS30 [B,H,W]		ADC12B_ADTCS31 [B,H,W]	
0x1A0	ADC12B_ADRCIFC [B,H,W]			
0x1A4	ADC12B_ADSCANSC0 [B,H,W]	-	-	
0x1A8 to 0x1AC	-			
0x1B0	ADC12B_ADCH [B,H,W]		ADC12B_ADCH [B,H,W]	ADC12B_ADMD [B,H,W]
0x1B4	ADC12B_ADSTPCS0 [B,H,W]	ADC12B_ADSTPCS1 [B,H,W]	ADC12B_ADSTPCS2 [B,H,W]	ADC12B_ADSTPCS3 [B,H,W]
0x1B8	ADC12B_ADSTPCS4 [B,H,W]	ADC12B_ADSTPCS5 [B,H,W]	ADC12B_ADSTPCS6 [B,H,W]	ADC12B_ADSTPCS7 [B,H,W]
0x1BC	ADC12B_ADTCSS0 [B,H,W]		ADC12B_ADTCSS1 [B,H,W]	
0x1C0	ADC12B_ADTCSS2 [B,H,W]		ADC12B_ADTCSS3 [B,H,W]	
0x1C4	ADC12B_ADTCSS4 [B,H,W]		ADC12B_ADTCSS5 [B,H,W]	
0x1C8	ADC12B_ADTCSS6 [B,H,W]		ADC12B_ADTCSS7 [B,H,W]	
0x1CC	ADC12B_ADTCSS8 [B,H,W]		ADC12B_ADTCSS9 [B,H,W]	
0x1D0	ADC12B_ADTCSS10 [B,H,W]		ADC12B_ADTCSS11 [B,H,W]	
0x1D4	ADC12B_ADTCSS12 [B,H,W]		ADC12B_ADTCSS13 [B,H,W]	
0x1D8	ADC12B_ADTCSS14 [B,H,W]		ADC12B_ADTCSS15 [B,H,W]	
0x1DC	ADC12B_ADTCSS16 [B,H,W]		ADC12B_ADTCSS17 [B,H,W]	
0x1E0	ADC12B_ADTCSS18 [B,H,W]		ADC12B_ADTCSS19 [B,H,W]	
0x1E4	ADC12B_ADTCSS20 [B,H,W]		ADC12B_ADTCSS21 [B,H,W]	
0x1E8	ADC12B_ADTCSS22 [B,H,W]		ADC12B_ADTCSS23 [B,H,W]	
0x1EC	ADC12B_ADTCSS24 [B,H,W]		ADC12B_ADTCSS25 [B,H,W]	
0x1F0	ADC12B_ADTCSS26 [B,H,W]		ADC12B_ADTCSS27 [B,H,W]	
0x1F4	ADC12B_ADTCSS28 [B,H,W]		ADC12B_ADTCSS29 [B,H,W]	
0x1F8	ADC12B_ADTCSS30 [B,H,W]		ADC12B_ADTCSS31 [B,H,W]	
0x1FC	ADC12B_ADSCANSS0 [B,H,W]	-	-	

**(36) A/D Converter Control**

Key code Base\_Address : 0xB101\_3000

Offset Address	Register			
	+3	+2	+1	+0
0x000	KEYCDR [W]			

Analog input control Base\_Address : 0xB101\_3004

Offset Address	Register			
	+3	+2	+1	+0
0x000	ADER [B,H,W]			

4ch analog input control Base\_Address : 0xB101\_3008

Offset Address	Register			
	+3	+2	+1	+0
0x000	-		ADER4CH_1 [B,H,W]	ADER4CH_0 [B,H,W]



### (37) Waveform Generator

Waveform generator Unit00(ch.0 to ch.5) Base\_Address : 0xB200\_0400(xx=00)

Waveform generator Unit01(ch.6 to ch.11) Base\_Address : 0xB100\_0400(xx=01)

Waveform generator Unit02(ch.12 to ch.17) Base\_Address : 0xB101\_0600(xx=02)

Waveform generator Unit03(ch.18 to ch.23) Base\_Address : 0xB101\_0640(xx=03)

Offset Address	Register			
	+3	+2	+1	+0
0x000	-			
0x004	WFGxx_TMRR0 [H,W]		WFGxx_TMRR1 [H,W]	
0x008	WFGxx_TMRR2 [H,W]		-	
0x00C	WFGxx_DTCR0 [B,H,W]	WFGxx_DTCR1 [B,H,W]	WFGxx_DTCR2 [B,H,W]	-
0x010	-	WFGxx_DTIR [B,H,W]	-	WFGxx_DTMNS [B,H,W]
0x014	-	WFGxx_SIGCR1 [B,H,W]	-	WFGxx_SIGCR2 [B,H,W]
0x018	WFGxx_PICS [B,H,W]	-	-	
0x01C	WFGxx_DTCRC0 [B,H,W]	WFGxx_DTCRC1 [B,H,W]	WFGxx_DTCRC2 [B,H,W]	-
0x020	-	WFGxx_DTIRC [B,H,W]	-	
0x024	-	WFGxx_SIGCR1C [B,H,W]	-	
0x028	WFGxx_DTCRS0 [B,H,W]	WFGxx_DTCRS1 [B,H,W]	WFGxx_DTCRS2 [B,H,W]	-
0x02C	-	WFGxx_DTIRS [B,H,W]	-	
0x030	-	WFGxx_SIGCR1S [B,H,W]	-	
0x034 to 0x03C	-			

### (38) Waveform Generator Control

DTTI select0 (WFG00/01) Base\_Address : 0xB101\_0600

Offset Address	Register			
	+3	+2	+1	+0
0x000	WFG02_DTSR [B,H,W]	-	-	-

DTTI select1 (WFG02/03) Base\_Address : 0xB101\_0640

Offset Address	Register			
	+3	+2	+1	+0
0x000	WFG03_DTSR [B,H,W]	-	-	-

Software DTTI control Base\_Address : 0xB101\_1000

Offset Address	Register			
	+3	+2	+1	+0
0x000	-	-	-	SDTCR2 [B,H,W]

Input setting of external DTTI pin Base\_Address: 0xB101\_1004

Offset Address	Register			
	+3	+2	+1	+0
0x000	-	-	-	EDTCR2 [B,H,W]

RTO output level conversion Base\_Address: 0xB101\_2000

Offset Address	Register			
	+3	+2	+1	+0
0x000	RTOSEL3 [B,H,W]	RTOSEL2 [B,H,W]	RTOSEL1 [B,H,W]	RTOSEL0 [B,H,W]



### (39) Up/Down Counter

Up/Down counter ch.0 Base\_Address : 0xB200\_0500(xx=00)

Up/Down counter ch.1 Base\_Address : 0xB200\_0580(xx=01)

Up/Down counter ch.2 Base\_Address : 0xB100\_0500(xx=02)

Up/Down counter ch.3 Base\_Address : 0xB100\_0580(xx=03)

Offset Address	Register			
	+3	+2	+1	+0
0x000	-		UDC16Bxx_RCRH [B,H,W]	UDC16Bxx_RCRL [B,H,W]
0x004	-		UDC16Bxx_UDCRH [B,H,W]	UDC16Bxx_UDCRL [B,H,W]
0x008	-		UDC16Bxx_CCRH [B,H,W]	UDC16Bxx_CCRL [B,H,W]
0x00C	-		-	UDC16Bxx_CSRL [B,H,W]
0x010	-		UDC16Bxx_CMPRH0 [B,H,W]	UDC16Bxx_CMPRL0 [B,H,W]
0x014	-		UDC16Bxx_CMPRH1 [B,H,W]	UDC16Bxx_CMPRL1 [B,H,W]
0x018	-		UDC16Bxx_CMPRH2 [B,H,W]	UDC16Bxx_CMPRL2 [B,H,W]
0x01C	-		UDC16Bxx_CMPRH3 [B,H,W]	UDC16Bxx_CMPRL3 [B,H,W]
0x020	-		UDC16Bxx_CMPRH4 [B,H,W]	UDC16Bxx_CMPRL4 [B,H,W]
0x024	-		UDC16Bxx_CMPRH5 [B,H,W]	UDC16Bxx_CMPRL5 [B,H,W]
0x028	-		UDC16Bxx_CMPBRH0 [B,H,W]	UDC16Bxx_CMPBRL0 [B,H,W]
0x02C	-		UDC16Bxx_CMPBRH1 [B,H,W]	UDC16Bxx_CMPBRL1 [B,H,W]
0x030	-		UDC16Bxx_CMPBRH2 [B,H,W]	UDC16Bxx_CMPBRL2 [B,H,W]
0x034	-		UDC16Bxx_CMPBRH3 [B,H,W]	UDC16Bxx_CMPBRL3 [B,H,W]
0x038	-		UDC16Bxx_CMPBRH4 [B,H,W]	UDC16Bxx_CMPBRL4 [B,H,W]
0x03C	-		UDC16Bxx_CMPBRH5 [B,H,W]	UDC16Bxx_CMPBRL5 [B,H,W]
0x040	-		UDC16Bxx_CMPMSKRH0 [B,H,W]	UDC16Bxx_CMPMSKRL0 [B,H,W]
0x044	-		UDC16Bxx_CMPMSKRH1 [B,H,W]	UDC16Bxx_CMPMSKRL1 [B,H,W]
0x048	-		UDC16Bxx_CMPMSKRH2 [B,H,W]	UDC16Bxx_CMPMSKRL2 [B,H,W]
0x04C	-		UDC16Bxx_CMPMSKRH3 [B,H,W]	UDC16Bxx_CMPMSKRL3 [B,H,W]
0x050	-		UDC16Bxx_CMPMSKRH4 [B,H,W]	UDC16Bxx_CMPMSKRL4 [B,H,W]
0x054	-		UDC16Bxx_CMPMSKRH5 [B,H,W]	UDC16Bxx_CMPMSKRL5 [B,H,W]

Offset Address	Register			
	+3	+2	+1	+0
0x058	-		UDC16Bxx_CITER [B,H,W]	UDC16Bxx_CMPFR [B,H,W]
0x05C	-		-	UDC16Bxx_CBTR [B,H,W]
0x060	-		UDC16Bxx_CCCRH [B,H,W]	UDC16Bxx_CCCRL [B,H,W]
0x064	-		UDC16Bxx_CCSRH [B,H,W]	UDC16Bxx_CCSRL [B,H,W]
0x068	-		-	UDC16Bxx_CSCRL [B,H,W]
0x06C	-		-	UDC16Bxx_CSSRL [B,H,W]
0x070	-		-	UDC16Bxx_CMPFCR [B,H,W]
0x074 to x07C	-			



#### (40) R/D Converter

R/D converter Unit00 Base\_Address : 0xB200\_0C00(xx=00)

R/D converter Unit01 Base\_Address : 0xB100\_0C00(xx=01)

Offset Address	Register			
	+3	+2	+1	+0
0x000	RDCxx_RDCCTR0 [B,H,W]	RDCxx_RDCCTR1 [B,H,W]	RDCxx_RDCINTR [B,H,W]	RDCxx_RDCICER [B,H,W]
0x004	RDCxx_RDCINTR2 [B,H,W]	RDCxx_RDCCTR2 [B,H,W]	RDCxx_RDCIPR [H,W]	
0x008	RDCxx_RDCCPR1 [H,W]		RDCxx_RDCCPR2 [H,W]	
0x00C	RDCxx_RDCCPR3 [H,W]		RDCxx_RDCCPR4 [H,W]	
0x010	RDCxx_AGLDR [H,W]		RDCxx_AGVLDR [H,W]	
0x014	RDCxx_AGLDBR [H,W]		RDCxx_AGVLDBR [H,W]	
0x018	RDCxx_SCCIR [H,W]		-	
0x01C	RDCxx_SINDR [W]			
0x020	RDCxx_COSDR [W]			
0x024	-			
0x028	RDCxx_SINDR1 [W]			
0x02C	RDCxx_COSDR1 [W]			
0x030	RDCxx_ADTCDA [B,H,W]		RDCxx_ADTCDB [B,H,W]	
0x034	RDCxx_ADTCDC [B,H,W]		RDCxx_ADTCDD [B,H,W]	
0x038	RDCxx_AGLDR2 [H,W]		-	RDCxx_RDCICERC [B,H,W]
0x03C	RDCxx_SCCIRC [B,H,W]		-	
0x040	-		-	RDCxx_RDCICERS [B,H,W]
0x044	-		-	RDCxx_ADCCTR [B,H,W]
0x048 to 0x0FC	-			

#### (41) D/A Converter

D/A converter ch.0 Base\_Address : 0xB200\_0D00(xx=00)

D/A converter ch.1 Base\_Address : 0xB100\_0D00(xx=01)

Offset Address	Register			
	+3	+2	+1	+0
0x000	DACxx_DAER [B,H,W]	-	-	
0x004	DACxx_DACR [B,H,W]	-	DACxx_DADR [B,H,W]	
0x008	DACxx_KEYCDR [W]			
0x00C to 0x0FC	-			

\*: In the part number with R/D converter, this register controls output of MAG\_OUT pin. In the part number without R/D converter, this register selects function of DAOUT pin.

**(42) Motor Vector Operation Accelerator**

Motor vector operation accelerator Unit00 Base\_Address : 0xB200\_0800(xx=00)

Motor vector operation accelerator Unit01 Base\_Address : 0xB100\_0800(xx=01)

Offset Address	Register			
	+3	+2	+1	+0
0x000	-	MVAxx_MVASS [B,H,W]	-	MVAxx_MVACC [B,H,W]
0x004	MVAxx_MVAUSEL [B,H,W]		-	MVAxx_MVAOMS [B,H,W]
0x008	-	MVAxx_MVAIS [B,H,W]	MVAxx_MVASMS [B,H,W]	MVAxx_MVAPS [B,H,W]
0x00C	-	MVAxx_MVAES [B,H,W]	MVAxx_MVARS [B,H,W]	
0x010	-	MVAxx_MVAESCLR [B,H,W]	MVAxx_MVARSLR [B,H,W]	
0x014	-	MVAxx_MVAESIE [B,H,W]	MVAxx_MVARSIE [B,H,W]	
0x018	-	MVAxx_MVACES [B,H,W]	-	MVAxx_MVACS [B,H,W]
0x01C	MVAxx_MVAFDES [W]			
0x020	MVAxx_MVAECCES [W]			
0x024	MVAxx_MVAFES [B,H,W]		MVAxx_MVAFDDES [B,H,W]	
0x028	MVAxx_MVAUDSES [H,W]		-	MVAxx_MVAUDHES [B,H,W]
0x02C	MVAxx_MVADES [W]			
0x030	-		MVAxx_MVASCCIR [H,W]	
0x034	MVAxx_MVASCCSIN [W]			
0x038	MVAxx_MVASCCCOS [W]			
0x03C	MVAxx_MVAAGVL [H,W]		MVAxx_MVAAGL [H,W]	
0x040	MVAxx_MVARDCRS2D [H,W]		MVAxx_MVARDCRS1D [H,W]	
0x044	-		MVAxx_MVARDCTCDT [H,W]	
0x048	MVAxx_MVARRSN1D [W]			
0x04C	MVAxx_MVARRSN2D [W]			
0x050	MVAxx_MVALA [H,W]		MVAxx_MVACDAGL [H,W]	
0x054	MVAxx_MVA AVR [W]			
0x058	MVAxx_MVASIN [W]			
0x05C	MVAxx_MVACOS [W]			
0x060 to 0x064	-			
0x068	MVAxx_MVASINLA [W]			
0x06C	MVAxx_MVACOSLA [W]			
0x070	MVAxx_MVAADC DV [H,W]		MVAxx_MVAADC DU [H,W]	
0x074	-		MVAxx_MVAADC DW [H,W]	
0x078	MVAxx_MVAIU [W]			
0x07C	MVAxx_MVAIV [W]			
0x080	MVAxx_MVAIW [W]			
0x084	MVAxx_MVAID [W]			
0x088	MVAxx_MVAIQ [W]			
0x08C	MVAxx_MVAFID [W]			
0x090	MVAxx_MVAFIQ [W]			
0x094	MVAxx_MVACNTID [W]			
0x098	MVAxx_MVACNTIQ [W]			
0x09C	MVAxx_MVAVD [W]			
0x0A0	MVAxx_MVAVQ [W]			





H A R D W A R E M A N U A L

Offset Address	Register			
	+3	+2	+1	+0
0x0A4	MVAxx_MVAVU [W]			
0x0A8	MVAxx_MVAVV [W]			
0x0AC	MVAxx_MVAVW [W]			
0x0B0	MVAxx_MVACPB/MVAxx_MVACP [W]			
0x0B4	MVAxx_MVACATHB/MVAxx_MVACATH [W]			
0x0B8	MVAxx_MVAIDUTHB/MVAxx_MVAIDUTH [W]			
0x0BC	MVAxx_MVAIQUTHB/MVAxx_MVAIQUTH [W]			
0x0C0	MVAxx_MVAIDDTHB/MVAxx_MVAIDDTH [W]			
0x0C4	MVAxx_MVAIQDTHB/MVAxx_MVAIQDTH [W]			
0x0C8	MVAxx_MVATGIDB/MVAxx_MVATGID [W]			
0x0CC	MVAxx_MVATGIQB/MVAxx_MVATGIQ [W]			
0x0D0	MVAxx_MVAKPIDB/MVAxx_MVAKPID [W]			
0x0D4	MVAxx_MVAKPIQB/MVAxx_MVAKPIQ [W]			
0x0D8	MVAxx_MVAKIIDB/MVAxx_MVAKIID [W]			
0x0DC	MVAxx_MVAKIIQB/MVAxx_MVAKIIQ [W]			
0x0E0	MVAxx_MVAKDIDB/MVAxx_MVAKDID [W]			
0x0E4	MVAxx_MVAKDIQB/MVAxx_MVAKDIQ [W]			
0x0E8	MVAxx_MVAMAXIDB/MVAxx_MVAMAXID [W]			
0x0EC	MVAxx_MVAMAXIQB/MVAxx_MVAMAXIQ [W]			
0x0F0	MVAxx_MVAMINIDB/MVAxx_MVAMINID [W]			
0x0F4	MVAxx_MVAMINIQB/MVAxx_MVAMINIQ [W]			
0x0F8	MVAxx_MVADIDB1/MVAxx_MVADID1 [W]			
0x0FC	MVAxx_MVADIQB1/MVAxx_MVADIQ1 [W]			
0x100	MVAxx_MVADIDB2/MVAxx_MVADID2 [W]			
0x104	MVAxx_MVADIQB2/MVAxx_MVADIQ2 [W]			
0x108	MVAxx_MVACAGL [W]			
0x10C	MVAxx_MVAAGLOFST [H,W]		MVAxx_MVAAGLM [H,W]	
0x110	MVAxx_MVACAGVL [W]			
0x114	MVAxx_MVARRSR1 [W]			
0x118	MVAxx_MVARRSR2 [W]			
0x11C	MVAxx_MVARRSOFST1 [W]			
0x120	MVAxx_MVARRSOFST2 [W]			
0x124	-		MVAxx_MVARDCTOFST [H,W]	
0x128	MVAxx_MVARPMGP [W]			
0x12C	MVAxx_MVARDCAUTH [W]			
0x130	MVAxx_MVARDCADTH [W]			
0x134	MVAxx_MVARDCAGLTH [W]			
0x138	MVAxx_MVAADCRU [W]			
0x13C	MVAxx_MVAADCRV [W]			
0x140	MVAxx_MVAADCRW [W]			
0x144	MVAxx_MVAADOFSTU [W]			
0x148	MVAxx_MVAADOFSTV [W]			
0x14C	MVAxx_MVAADOFSTW [W]			
0x150	-			
0x154	MVAxx_MVAFIG0 [W]			
0x158	MVAxx_MVAFIG1 [W]			
0x15C	MVAxx_MVAFIG2 [W]			
0x160	MVAxx_MVAFIG3 [W]			
0x164	MVAxx_MVAFIG4 [W]			

Offset Address	Register			
	+3	+2	+1	+0
0x168	MVAxx_MVAFIG5 [W]			
0x16C	MVAxx_MVAFIG6 [W]			
0x170	MVAxx_MVAFIDD1 [W]			
0x174	MVAxx_MVAFIDD2 [W]			
0x178	MVAxx_MVAFIDD3 [W]			
0x17C	MVAxx_MVAFIDD4 [W]			
0x180	MVAxx_MVAFIDD5 [W]			
0x184	MVAxx_MVAFIDD6 [W]			
0x188	MVAxx_MVAFIDQ1 [W]			
0x18C	MVAxx_MVAFIDQ2 [W]			
0x190	MVAxx_MVAFIDQ3 [W]			
0x194	MVAxx_MVAFIDQ4 [W]			
0x198	MVAxx_MVAFIDQ5 [W]			
0x19C	MVAxx_MVAFIDQ6 [W]			
0x1A0	MVAxx_MVAMR [W]			
0x1A4	MVAxx_MVAMLQ [W]			
0x1A8	MVAxx_MVAMLQ [W]			
0x1AC	MVAxx_MVAMIF [W]			
0x1B0 to 0x1FC	-			



### (43) FlexRay

FlexRay customer registers, special register, interrupt-related registers,

Communication controller (CC) control registers,

Communication controller (CC) status registers,

Message buffer control registers,

Message buffer status registers,

Identification registers,

Input buffers, Output buffers

Base\_Address: 0xB200\_1000

Offset Address	Register			
	+3	+2	+1	+0
0x000	FLXRY_CIF0 [W]			
0x004	FLXRY_CIF1 [W]			
0x008	FLXRY_CIF1F [W]			
0x00C	FLXRY_CIF1C [W]			
0x010 to 0x018	-			
0x01C	FLXRY_LCK [W]			
0x020	FLXRY_EIR [W]			
0x024	FLXRY_SIR [W]			
0x028	FLXRY_EILS [W]			
0x02C	FLXRY_SILS [W]			
0x030	FLXRY_EIES [W]			
0x034	FLXRY_EIER [W]			
0x038	FLXRY_SIES [W]			
0x03C	FLXRY_SIER [W]			
0x040	FLXRY_ILE [W]			
0x044	FLXRY_T0C [W]			
0x048	FLXRY_T1C [W]			
0x04C	FLXRY_STPW1 [W]			
0x050	FLXRY_STPW2 [W]			
0x054 to 0x07C	-			
0x080	FLXRY_SUCC1 [W]			
0x084	FLXRY_SUCC2 [W]			
0x088	FLXRY_SUCC3 [W]			
0x08C	FLXRY_NEMC [W]			
0x090	FLXRY_PRTC1 [W]			
0x094	FLXRY_PRTC2 [W]			
0x098	FLXRY_MHDC [W]			
0x09C	-			
0x0A0	FLXRY_GTUC1 [W]			
0x0A4	FLXRY_GTUC2 [W]			
0x0A8	FLXRY_GTUC3 [W]			
0x0AC	FLXRY_GTUC4 [W]			
0x0B0	FLXRY_GTUC5 [W]			
0x0B4	FLXRY_GTUC6 [W]			
0x0B8	FLXRY_GTUC7 [W]			
0x0BC	FLXRY_GTUC8 [W]			
0x0C0	FLXRY_GTUC9 [W]			
0x0C4	FLXRY_GTUC10 [W]			

Offset Address	Register			
	+3	+2	+1	+0
0x0C8	FLXRY_GTUC11 [W]			
0x0CC to 0x0FC	-			
0x100	FLXRY_CCSV [W]			
0x104	FLXRY_CCEV [W]			
0x108 to 0x10C	-			
0x110	FLXRY_SCV [W]			
0x114	FLXRY_MTCCV [W]			
0x118	FLXRY_RCV [W]			
0x11C	FLXRY_OCV [W]			
0x120	FLXRY_SFS [W]			
0x124	FLXRY_SWNIT [W]			
0x128	FLXRY_ACS [W]			
0x12C	-			
0x130	FLXRY_ESID1 [W]			
0x134	FLXRY_ESID2 [W]			
0x138	FLXRY_ESID3 [W]			
0x13C	FLXRY_ESID4 [W]			
0x140	FLXRY_ESID5 [W]			
0x144	FLXRY_ESID6 [W]			
0x148	FLXRY_ESID7 [W]			
0x14C	FLXRY_ESID8 [W]			
0x150	FLXRY_ESID9 [W]			
0x154	FLXRY_ESID10 [W]			
0x158	FLXRY_ESID11 [W]			
0x15C	FLXRY_ESID12 [W]			
0x160	FLXRY_ESID13 [W]			
0x164	FLXRY_ESID14 [W]			
0x168	FLXRY_ESID15 [W]			
0x16C	-			
0x170	FLXRY_OSID1 [W]			
0x174	FLXRY_OSID2 [W]			
0x178	FLXRY_OSID3 [W]			
0x17C	FLXRY_OSID4 [W]			
0x180	FLXRY_OSID5 [W]			
0x184	FLXRY_OSID6 [W]			
0x188	FLXRY_OSID7 [W]			
0x18C	FLXRY_OSID8 [W]			
0x190	FLXRY_OSID9 [W]			
0x194	FLXRY_OSID10 [W]			
0x198	FLXRY_OSID11 [W]			
0x19C	FLXRY_OSID12 [W]			
0x1A0	FLXRY_OSID13 [W]			
0x1A4	FLXRY_OSID14 [W]			
0x1A8	FLXRY_OSID15 [W]			
0x1AC	-			
0x1B0	FLXRY_NMV1 [W]			
0x1B4	FLXRY_NMV2 [W]			
0x1B8	FLXRY_NMV3 [W]			
0x1BC to 0x2FC	-			



H A R D W A R E M A N U A L

Offset Address	Register			
	+3	+2	+1	+0
0x300	FLXRY_MRC [W]			
0x304	FLXRY_FRF [W]			
0x308	FLXRY_FRFM [W]			
0x30C	FLXRY_FCL [W]			
0x310	FLXRY_MHDS [W]			
0x314	FLXRY_LDTS [W]			
0x318	FLXRY_FSR [W]			
0x31C	FLXRY_MHDF [W]			
0x320	FLXRY_TXRQ1 [W]			
0x324	FLXRY_TXRQ2 [W]			
0x328	FLXRY_TXRQ3 [W]			
0x32C	FLXRY_TXRQ4 [W]			
0x330	FLXRY_NDAT1 [W]			
0x334	FLXRY_NDAT2 [W]			
0x338	FLXRY_NDAT3 [W]			
0x33C	FLXRY_NDAT4 [W]			
0x340	FLXRY_MBSC1 [W]			
0x344	FLXRY_MBSC2 [W]			
0x348	FLXRY_MBSC3 [W]			
0x34C	FLXRY_MBSC4 [W]			
0x350 to 0x3EC	-			
0x3F0	FLXRY_CREL [W]			
0x3F4	FLXRY_ENDN [W]			
0x3F8 to 0x3FC	-			
0x400	FLXRY_WRDS1 [W]			
0x404	FLXRY_WRDS2 [W]			
0x408	FLXRY_WRDS3 [W]			
0x40C	FLXRY_WRDS4 [W]			
0x410	FLXRY_WRDS5 [W]			
0x414	FLXRY_WRDS6 [W]			
0x418	FLXRY_WRDS7 [W]			
0x41C	FLXRY_WRDS8 [W]			
0x420	FLXRY_WRDS9 [W]			
0x424	FLXRY_WRDS10 [W]			
0x428	FLXRY_WRDS11 [W]			
0x42C	FLXRY_WRDS12 [W]			
0x430	FLXRY_WRDS13 [W]			
0x434	FLXRY_WRDS14 [W]			
0x438	FLXRY_WRDS15 [W]			
0x43C	FLXRY_WRDS16 [W]			
0x440	FLXRY_WRDS17 [W]			
0x444	FLXRY_WRDS18 [W]			
0x448	FLXRY_WRDS19 [W]			
0x44C	FLXRY_WRDS20 [W]			
0x450	FLXRY_WRDS21 [W]			
0x454	FLXRY_WRDS22 [W]			
0x458	FLXRY_WRDS23 [W]			
0x45C	FLXRY_WRDS24 [W]			
0x460	FLXRY_WRDS25 [W]			

Offset Address	Register			
	+3	+2	+1	+0
0x464				FLXRY_WRDS26 [W]
0x468				FLXRY_WRDS27 [W]
0x46C				FLXRY_WRDS28 [W]
0x470				FLXRY_WRDS29 [W]
0x474				FLXRY_WRDS30 [W]
0x478				FLXRY_WRDS31 [W]
0x47C				FLXRY_WRDS32 [W]
0x480				FLXRY_WRDS33 [W]
0x484				FLXRY_WRDS34 [W]
0x488				FLXRY_WRDS35 [W]
0x48C				FLXRY_WRDS36 [W]
0x490				FLXRY_WRDS37 [W]
0x494				FLXRY_WRDS38 [W]
0x498				FLXRY_WRDS39 [W]
0x49C				FLXRY_WRDS40 [W]
0x4A0				FLXRY_WRDS41 [W]
0x4A4				FLXRY_WRDS42 [W]
0x4A8				FLXRY_WRDS43 [W]
0x4AC				FLXRY_WRDS44 [W]
0x4B0				FLXRY_WRDS45 [W]
0x4B4				FLXRY_WRDS46 [W]
0x4B8				FLXRY_WRDS47 [W]
0x4BC				FLXRY_WRDS48 [W]
0x4C0				FLXRY_WRDS49 [W]
0x4C4				FLXRY_WRDS50 [W]
0x4C8				FLXRY_WRDS51 [W]
0x4CC				FLXRY_WRDS52 [W]
0x4D0				FLXRY_WRDS53 [W]
0x4D4				FLXRY_WRDS54 [W]
0x4D8				FLXRY_WRDS55 [W]
0x4DC				FLXRY_WRDS56 [W]
0x4E0				FLXRY_WRDS57 [W]
0x4E4				FLXRY_WRDS58 [W]
0x4E8				FLXRY_WRDS59 [W]
0x4EC				FLXRY_WRDS60 [W]
0x4F0				FLXRY_WRDS61 [W]
0x4F4				FLXRY_WRDS62 [W]
0x4F8				FLXRY_WRDS63 [W]
0x4FC				FLXRY_WRDS64 [W]
0x500				FLXRY_WRHS1 [W]
0x504				FLXRY_WRHS2 [W]
0x508				FLXRY_WRHS3 [W]
0x50C				-
0x510				FLXRY_IBCM [W]
0x514				FLXRY_IBCR [W]
0x518 to 0x5FC				-
0x600				FLXRY_RDDS1 [W]
0x604				FLXRY_RDDS2 [W]
0x608				FLXRY_RDDS3 [W]



H A R D W A R E M A N U A L

Offset Address	Register			
	+3	+2	+1	+0
0x60C				FLXRY_RDDS4 [W]
0x610				FLXRY_RDDS5 [W]
0x614				FLXRY_RDDS6 [W]
0x618				FLXRY_RDDS7 [W]
0x61C				FLXRY_RDDS8 [W]
0x620				FLXRY_RDDS9 [W]
0x624				FLXRY_RDDS10 [W]
0x628				FLXRY_RDDS11 [W]
0x62C				FLXRY_RDDS12 [W]
0x630				FLXRY_RDDS13 [W]
0x634				FLXRY_RDDS14 [W]
0x638				FLXRY_RDDS15 [W]
0x63C				FLXRY_RDDS16 [W]
0x640				FLXRY_RDDS17 [W]
0x644				FLXRY_RDDS18 [W]
0x648				FLXRY_RDDS19 [W]
0x64C				FLXRY_RDDS20 [W]
0x650				FLXRY_RDDS21 [W]
0x654				FLXRY_RDDS22 [W]
0x658				FLXRY_RDDS23 [W]
0x65C				FLXRY_RDDS24 [W]
0x660				FLXRY_RDDS25 [W]
0x664				FLXRY_RDDS26 [W]
0x668				FLXRY_RDDS27 [W]
0x66C				FLXRY_RDDS28 [W]
0x670				FLXRY_RDDS29 [W]
0x674				FLXRY_RDDS30 [W]
0x678				FLXRY_RDDS31 [W]
0x67C				FLXRY_RDDS32 [W]
0x680				FLXRY_RDDS33 [W]
0x684				FLXRY_RDDS34 [W]
0x688				FLXRY_RDDS35 [W]
0x68C				FLXRY_RDDS36 [W]
0x690				FLXRY_RDDS37 [W]
0x694				FLXRY_RDDS38 [W]
0x698				FLXRY_RDDS39 [W]
0x69C				FLXRY_RDDS40 [W]
0x6A0				FLXRY_RDDS41 [W]
0x6A4				FLXRY_RDDS42 [W]
0x6A8				FLXRY_RDDS43 [W]
0x6AC				FLXRY_RDDS44 [W]
0x6B0				FLXRY_RDDS45 [W]
0x6B4				FLXRY_RDDS46 [W]
0x6B8				FLXRY_RDDS47 [W]
0x6BC				FLXRY_RDDS48 [W]
0x6C0				FLXRY_RDDS49 [W]
0x6C4				FLXRY_RDDS50 [W]
0x6C8				FLXRY_RDDS51 [W]
0x6CC				FLXRY_RDDS52 [W]

Offset Address	Register			
	+3	+2	+1	+0
0x6D0	FLXRY_RDDS53 [W]			
0x6D4	FLXRY_RDDS54 [W]			
0x6D8	FLXRY_RDDS55 [W]			
0x6DC	FLXRY_RDDS56 [W]			
0x6E0	FLXRY_RDDS57 [W]			
0x6E4	FLXRY_RDDS58 [W]			
0x6E8	FLXRY_RDDS59 [W]			
0x6EC	FLXRY_RDDS60 [W]			
0x6F0	FLXRY_RDDS61 [W]			
0x6F4	FLXRY_RDDS62 [W]			
0x6F8	FLXRY_RDDS63 [W]			
0x6FC	FLXRY_RDDS64 [W]			
0x700	FLXRY_RDHS1 [W]			
0x704	FLXRY_RDHS2 [W]			
0x708	FLXRY_RDHS3 [W]			
0x70C	FLXRY_MBS [W]			
0x710	FLXRY_OBCM [W]			
0x714	FLXRY_OBCR [W]			
0x718 to 0x7FC	-			

#### (44) Clock Control for FlexRay/RDC

Clock control for FlexRay/RDC Base\_Address : 0xB201\_0000

Offset Address	Register			
	+3	+2	+1	+0
0x000	ERAYP_PLL2DIVM [B,H,W]	ERAYP_PLL2DIVN [B,H,W]	ERAYP_PLL2DIVG [B,H,W]	ERAYP_PLL2MULG [B,H,W]
0x004	ERAYP_PLL2CTRL [B,H,W]	ERAYP_PLL2DIVK [B,H,W]	ERAYP_CLKR2 [B,H,W]	-
0x008	ERAYP_PLL2CTRLF [B,H,W]	-	ERAYP_CLKR2F [B,H,W]	-
0x00C	ERAYP_PLL2CTRLC [B,H,W]	-	ERAYP_CLKR2C [B,H,W]	-
0x010 to 0x0FC	-			

Clock supervisor control for FlexRay/RDC Base\_Address: 0xB101\_3020

Offset Address	Register			
	+3	+2	+1	+0
0x000	-	-	-	ERAYP_CSVR [B,H,W]
0x004 to 0x0FC	-			





**(45) Clock Monitor**

Clock monitor Base\_Address: 0xB201\_0100

Offset Address	Register			
	+3	+2	+1	+0
0x000	-		CLKMN_CSCFG [B,H,W]	CLKMN_CMCFG [B,H,W]
0x004 to 0x0FC	-			

### 3. List of Interrupt Factors and DMA Activation Factors

This section explains interrupt factors and DMA activation factors.

#### 3.1. List of Interrupt Factors

This section shows list of interrupt factors.

The interrupt factors include the interrupt by individual CPU and by both CPU.

The former is called individual IRQ, the latter is called common IRQ.

List of interrupt factors shows the following.

IRQ Number	IRQ Factor	Priority Register	Vector Address Register	Type of IRQ
0	Reserved	-	-	-
1	Low-power consumption RUN profile update complete interrupt	IRCn_IRQPL0:IRQPL1[4:0]	IRCn_IRQVA1	Common IRQ
2	Hardware watchdog prior warning interrupt	IRCn_IRQPL0:IRQPL2[4:0]	IRCn_IRQVA2	Common IRQ
3	Software watchdog unit0, 1 prior warning interrupt	IRCn_IRQPL0:IRQPL3[4:0]	IRCn_IRQVA3	Individual IRQ
4 to 7	Reserved	-	-	-
8	TCFLASH unit0, 1 1-bit error correction interrupt/ready interrupt/hang interrupt	IRCn_IRQPL2:IRQPL8[4:0]	IRCn_IRQVA8	Individual IRQ
9	Reserved	-	-	-
10	WorkFLASH unit0 hang interrupt	IRCn_IRQPL2:IRQPL10[4:0]	IRCn_IRQVA10	Common IRQ
11	WorkFLASH unit1 hang interrupt	IRCn_IRQPL2:IRQPL11[4:0]	IRCn_IRQVA11	Common IRQ
12 to 15	Reserved	-	-	-
16	Interrupt controller unit0, 1 ECC 1-bit error interrupt	IRCn_IRQPL4:IRQPL16[4:0]	IRCn_IRQVA16	Individual IRQ
17 to 19	Reserved	-	-	-
20	WorkFLASH unit0 1-bit error correction interrupt/ready interrupt	IRCn_IRQPL5:IRQPL20[4:0]	IRCn_IRQVA20	Common IRQ
21	WorkFLASH unit1 1-bit error correction interrupt/ready interrupt	IRCn_IRQPL5:IRQPL21[4:0]	IRCn_IRQVA21	Common IRQ
22, 23	Reserved	-	-	-
24	External interrupt ch.0	IRCn_IRQPL6:IRQPL24[4:0]	IRCn_IRQVA24	Common IRQ
25	External interrupt ch.1	IRCn_IRQPL6:IRQPL25[4:0]	IRCn_IRQVA25	Common IRQ
26	External interrupt ch.2	IRCn_IRQPL6:IRQPL26[4:0]	IRCn_IRQVA26	Common IRQ
27	External interrupt ch.3	IRCn_IRQPL6:IRQPL27[4:0]	IRCn_IRQVA27	Common IRQ
28	External interrupt ch.4	IRCn_IRQPL7:IRQPL28[4:0]	IRCn_IRQVA28	Common IRQ
29	External interrupt ch.5	IRCn_IRQPL7:IRQPL29[4:0]	IRCn_IRQVA29	Common IRQ
30	External interrupt ch.6	IRCn_IRQPL7:IRQPL30[4:0]	IRCn_IRQVA30	Common IRQ
31	External interrupt ch.7	IRCn_IRQPL7:IRQPL31[4:0]	IRCn_IRQVA31	Common IRQ
32 to 55	Reserved	-	-	-
56	CAN ch.0	IRCn_IRQPL14:IRQPL56[4:0]	IRCn_IRQVA56	Common IRQ
57	CAN ch.1	IRCn_IRQPL14:IRQPL57[4:0]	IRCn_IRQVA57	Common IRQ
58	CAN ch.2	IRCn_IRQPL14:IRQPL58[4:0]	IRCn_IRQVA58	Common IRQ
59 to 63	Reserved	-	-	-
64	Multi-function serial interface ch.0 reception complete/status	IRCn_IRQPL16:IRQPL64[4:0]	IRCn_IRQVA64	Common IRQ
65	Multi-function serial interface ch.0 transmission complete	IRCn_IRQPL16:IRQPL65[4:0]	IRCn_IRQVA65	Common IRQ



IRQ Number	IRQ Factor	Priority Register	Vector Address Register	Type of IRQ
66	Multi-function serial interface ch.1 reception complete/status	IRCN_IRQPL16:IRQPL66[4:0]	IRCN_IRQVA66	Common IRQ
67	Multi-function serial interface ch.1 transmission complete	IRCN_IRQPL16:IRQPL67[4:0]	IRCN_IRQVA67	Common IRQ
68	Multi-function serial interface ch.2 reception complete/status	IRCN_IRQPL17:IRQPL68[4:0]	IRCN_IRQVA68	Common IRQ
69	Multi-function serial interface ch.2 transmission complete	IRCN_IRQPL17:IRQPL69[4:0]	IRCN_IRQVA69	Common IRQ
70	Multi-function serial interface ch.3 reception complete/status	IRCN_IRQPL17:IRQPL70[4:0]	IRCN_IRQVA70	Common IRQ
71	Multi-function serial interface ch.3 transmission complete	IRCN_IRQPL17:IRQPL71[4:0]	IRCN_IRQVA71	Common IRQ
72	Multi-function serial interface ch.4 reception complete/status	IRCN_IRQPL18:IRQPL72[4:0]	IRCN_IRQVA72	Common IRQ
73	Multi-function serial interface ch.4 transmission complete	IRCN_IRQPL18:IRQPL73[4:0]	IRCN_IRQVA73	Common IRQ
74 to 95	Reserved	-	-	-
96	Inter-processor communication (IPCU) interrupt	IRCN_IRQPL24:IRQPL96[4:0]	IRCN_IRQVA96	Individual IRQ
97 to 109	Reserved	-	-	-
110	TCRAM unit0, 1 RAM diagnosis interrupt	IRCN_IRQPL27:IRQPL110[4:0]	IRCN_IRQVA110	Individual IRQ
111 to 116	Reserved	-	-	-
117	CR calibration interrupt	IRCN_IRQPL29:IRQPL117[4:0]	IRCN_IRQVA117	Common IRQ
118 to 127	Reserved	-	-	-
128	Base timer ch.0 IRQ0/IRQ1	IRCN_IRQPL32:IRQPL128[4:0]	IRCN_IRQVA128	Common IRQ
129	Base timer ch.1 IRQ0/IRQ1	IRCN_IRQPL32:IRQPL129[4:0]	IRCN_IRQVA129	Common IRQ
130	Base timer ch.2 IRQ0/IRQ1	IRCN_IRQPL32:IRQPL130[4:0]	IRCN_IRQVA130	Common IRQ
131	Base timer ch.3 IRQ0/IRQ1	IRCN_IRQPL32:IRQPL131[4:0]	IRCN_IRQVA131	Common IRQ
132	Base timer ch.4 IRQ0/IRQ1	IRCN_IRQPL33:IRQPL132[4:0]	IRCN_IRQVA132	Common IRQ
133	Base timer ch.5 IRQ0/IRQ1	IRCN_IRQPL33:IRQPL133[4:0]	IRCN_IRQVA133	Common IRQ
134	Base timer ch.6 IRQ0/IRQ1	IRCN_IRQPL33:IRQPL134[4:0]	IRCN_IRQVA134	Common IRQ
135	Base timer ch.7 IRQ0/IRQ1	IRCN_IRQPL33:IRQPL135[4:0]	IRCN_IRQVA135	Common IRQ
136	Base timer ch.8 IRQ0/IRQ1	IRCN_IRQPL34:IRQPL136[4:0]	IRCN_IRQVA136	Common IRQ
137	Base timer ch.9 IRQ0/IRQ1	IRCN_IRQPL34:IRQPL137[4:0]	IRCN_IRQVA137	Common IRQ
138	Base timer ch.10 IRQ0/IRQ1	IRCN_IRQPL34:IRQPL138[4:0]	IRCN_IRQVA138	Common IRQ
139	Base timer ch.11 IRQ0/IRQ1	IRCN_IRQPL34:IRQPL139[4:0]	IRCN_IRQVA139	Common IRQ
140 to 175	Reserved	-	-	-
176	32-bit free-run timer ch.0 0 detection/compare clear interrupt	IRCN_IRQPL44:IRQPL176[4:0]	IRCN_IRQVA176	Common IRQ
177	32-bit free-run timer ch.1 0 detection/compare clear interrupt	IRCN_IRQPL44:IRQPL177[4:0]	IRCN_IRQVA177	Common IRQ
178	32-bit free-run timer ch.2 0 detection/compare clear interrupt	IRCN_IRQPL44:IRQPL178[4:0]	IRCN_IRQVA178	Common IRQ
179	32-bit free-run timer ch.3 0 detection/compare clear interrupt	IRCN_IRQPL44:IRQPL179[4:0]	IRCN_IRQVA179	Common IRQ
180	32-bit free-run timer ch.4 0 detection/compare clear interrupt	IRCN_IRQPL45:IRQPL180[4:0]	IRCN_IRQVA180	Common IRQ
181 to 191	Reserved	-	-	-

IRQ Number	IRQ Factor	Priority Register	Vector Address Register	Type of IRQ
192	32-bit input capture ch.0, ch.1 fetching	IRCN_IRQPL48:IRQPL192[4:0]	IRCN_IRQVA192	Common IRQ
193	32-bit input capture ch.2, ch.3 fetching	IRCN_IRQPL48:IRQPL193[4:0]	IRCN_IRQVA193	Common IRQ
194	32-bit input capture ch.4, ch.5 fetching	IRCN_IRQPL48:IRQPL194[4:0]	IRCN_IRQVA194	Common IRQ
195 to 271	Reserved	-	-	-
272	DMAC transfer error interrupt	IRCN_IRQPL68:IRQPL272[4:0]	IRCN_IRQVA272	Common IRQ
273	DMAC ch.0, ch.8 transfer complete interrupt	IRCN_IRQPL68:IRQPL273[4:0]	IRCN_IRQVA273	Common IRQ
274	DMAC ch.1, ch.9 transfer complete interrupt	IRCN_IRQPL68:IRQPL274[4:0]	IRCN_IRQVA274	Common IRQ
275	DMAC ch.2, ch.10 transfer complete interrupt	IRCN_IRQPL68:IRQPL275[4:0]	IRCN_IRQVA275	Common IRQ
276	DMAC ch.3, ch.11 transfer complete interrupt	IRCN_IRQPL69:IRQPL276[4:0]	IRCN_IRQVA276	Common IRQ
277	DMAC ch.4, ch.12 transfer complete interrupt	IRCN_IRQPL69:IRQPL277[4:0]	IRCN_IRQVA277	Common IRQ
278	DMAC ch.5, ch.13 transfer complete interrupt	IRCN_IRQPL69:IRQPL278[4:0]	IRCN_IRQVA278	Common IRQ
279	DMAC ch.6, ch.14 transfer complete interrupt	IRCN_IRQPL69:IRQPL279[4:0]	IRCN_IRQVA279	Common IRQ
280	DMAC ch.7, ch.15 transfer complete interrupt	IRCN_IRQPL70:IRQPL280[4:0]	IRCN_IRQVA280	Common IRQ
281 to 307	Reserved	-	-	-
308	Fast-CR clock timer interrupt	IRCN_IRQPL77:IRQPL308[4:0]	IRCN_IRQVA308	Common IRQ
309	Slow-CR clock timer interrupt	IRCN_IRQPL77:IRQPL309[4:0]	IRCN_IRQVA309	Common IRQ
310	Main clock timer interrupt	IRCN_IRQPL77:IRQPL310[4:0]	IRCN_IRQVA310	Common IRQ
311	Reserved	-	-	-
312	PMU interrupt	IRCN_IRQPL78:IRQPL312[4:0]	IRCN_IRQVA312	Individual IRQ
313 to 319	Reserved	-	-	-
320	FlexRay0	IRCN_IRQPL80:IRQPL320[4:0]	IRCN_IRQVA320	Common IRQ
321	FlexRay1	IRCN_IRQPL80:IRQPL321[4:0]	IRCN_IRQVA321	Common IRQ
322	FlexRay timer 0	IRCN_IRQPL80:IRQPL322[4:0]	IRCN_IRQVA322	Common IRQ
323	FlexRay timer 1	IRCN_IRQPL80:IRQPL323[4:0]	IRCN_IRQVA323	Common IRQ
324	16-bit free-run timer ch.0 0 detection/compare clear interrupt	IRCN_IRQPL81:IRQPL324[4:0]	IRCN_IRQVA324	Common IRQ
325	16-bit free-run timer ch.1 0 detection/compare clear interrupt	IRCN_IRQPL81:IRQPL325[4:0]	IRCN_IRQVA325	Common IRQ
326	16-bit free-run timer ch.2 0 detection/compare clear interrupt	IRCN_IRQPL81:IRQPL326[4:0]	IRCN_IRQVA326	Common IRQ
327	16-bit free-run timer ch.3 0 detection/compare clear interrupt	IRCN_IRQPL81:IRQPL327[4:0]	IRCN_IRQVA327	Common IRQ
328	16-bit free-run timer ch.4 0 detection/compare clear interrupt	IRCN_IRQPL82:IRQPL328[4:0]	IRCN_IRQVA328	Common IRQ
329	16-bit free-run timer ch.5 0 detection/compare clear interrupt	IRCN_IRQPL82:IRQPL329[4:0]	IRCN_IRQVA329	Common IRQ
330	16-bit free-run timer ch.6 0 detection/compare clear interrupt	IRCN_IRQPL82:IRQPL330[4:0]	IRCN_IRQVA330	Common IRQ
331	16-bit free-run timer ch.7 0 detection/compare clear interrupt	IRCN_IRQPL82:IRQPL331[4:0]	IRCN_IRQVA331	Common IRQ
332	16-bit free-run timer ch.8 0 detection/compare clear interrupt	IRCN_IRQPL83:IRQPL332[4:0]	IRCN_IRQVA332	Common IRQ
333	16-bit free-run timer ch.9 0 detection/compare clear interrupt	IRCN_IRQPL83:IRQPL333[4:0]	IRCN_IRQVA333	Common IRQ
334	16-bit free-run timer ch.10 0 detection/compare clear interrupt	IRCN_IRQPL83:IRQPL334[4:0]	IRCN_IRQVA334	Common IRQ
335	16-bit free-run timer ch.11 0 detection/compare clear interrupt	IRCN_IRQPL83:IRQPL335[4:0]	IRCN_IRQVA335	Common IRQ



IRQ Number	IRQ Factor	Priority Register	Vector Address Register	Type of IRQ
336	16-bit free-run timer ch.12 0 detection/compare clear interrupt	IRCN_IRQPL84:IRQPL336[4:0]	IRCN_IRQVA336	Common IRQ
337	16-bit free-run timer ch.13 0 detection/compare clear interrupt	IRCN_IRQPL84:IRQPL337[4:0]	IRCN_IRQVA337	Common IRQ
338	16-bit free-run timer ch.14 0 detection/compare clear interrupt	IRCN_IRQPL84:IRQPL338[4:0]	IRCN_IRQVA338	Common IRQ
339	16-bit free-run timer ch.15 0 detection/compare clear interrupt	IRCN_IRQPL84:IRQPL339[4:0]	IRCN_IRQVA339	Common IRQ
340	16-bit free-run timer ch.16 0 detection/compare clear interrupt	IRCN_IRQPL85:IRQPL340[4:0]	IRCN_IRQVA340	Common IRQ
341	16-bit free-run timer ch.17 0 detection/compare clear interrupt	IRCN_IRQPL85:IRQPL341[4:0]	IRCN_IRQVA341	Common IRQ
342	MVA0 angular calculation end interrupt	IRCN_IRQPL85:IRQPL342[4:0]	IRCN_IRQVA342	Common IRQ
343	MVA0 three-phase current normalization end interrupt	IRCN_IRQPL85:IRQPL343[4:0]	IRCN_IRQVA343	Common IRQ
344	MVA0 three-phase to two-phase DC conversion end interrupt	IRCN_IRQPL86:IRQPL344[4:0]	IRCN_IRQVA344	Common IRQ
345	MVA0 PID control end interrupt	IRCN_IRQPL86:IRQPL345[4:0]	IRCN_IRQVA345	Common IRQ
346	MVA0 current to voltage conversion end interrupt	IRCN_IRQPL86:IRQPL346[4:0]	IRCN_IRQVA346	Common IRQ
347	MVA0 two-phase to three-phase AC conversion end interrupt	IRCN_IRQPL86:IRQPL347[4:0]	IRCN_IRQVA347	Common IRQ
348	MVA0 overflow interrupt	IRCN_IRQPL87:IRQPL348[4:0]	IRCN_IRQVA348	Common IRQ
349	MVA0 underflow interrupt	IRCN_IRQPL87:IRQPL349[4:0]	IRCN_IRQVA349	Common IRQ
350	MVA0 floating-point non-normalized error interrupt	IRCN_IRQPL87:IRQPL350[4:0]	IRCN_IRQVA350	Common IRQ
351	MVA0 failure detection error interrupt	IRCN_IRQPL87:IRQPL351[4:0]	IRCN_IRQVA351	Common IRQ
352	MVA0 calculation data update error interrupt	IRCN_IRQPL88:IRQPL352[4:0]	IRCN_IRQVA352	Common IRQ
353	MVA0 R/D converter diagnosis error interrupt	IRCN_IRQPL88:IRQPL353[4:0]	IRCN_IRQVA353	Common IRQ
354	MVA0 Cumulative three-phase current abnormality detection error interrupt	IRCN_IRQPL88:IRQPL354[4:0]	IRCN_IRQVA354	Common IRQ
355	Reserved	-	-	-
356	MVA0 three-phase to two-phase DC value abnormality detection error interrupt	IRCN_IRQPL89:IRQPL356[4:0]	IRCN_IRQVA356	Common IRQ
357	MVA0 calculation overtime error interrupt	IRCN_IRQPL89:IRQPL357[4:0]	IRCN_IRQVA357	Common IRQ
358	MVA1 angular calculation end interrupt	IRCN_IRQPL89:IRQPL358[4:0]	IRCN_IRQVA358	Common IRQ
359	MVA1 three-phase current normalization end interrupt	IRCN_IRQPL89:IRQPL359[4:0]	IRCN_IRQVA359	Common IRQ
360	MVA1 three-phase to two-phase DC conversion end interrupt	IRCN_IRQPL90:IRQPL360[4:0]	IRCN_IRQVA360	Common IRQ
361	MVA1 PID control end interrupt	IRCN_IRQPL90:IRQPL361[4:0]	IRCN_IRQVA361	Common IRQ
362	MVA1 current to voltage conversion end interrupt	IRCN_IRQPL90:IRQPL362[4:0]	IRCN_IRQVA362	Common IRQ
363	MVA1 two-phase to three-phase AC conversion end interrupt	IRCN_IRQPL90:IRQPL363[4:0]	IRCN_IRQVA363	Common IRQ
364	MVA1 overflow interrupt	IRCN_IRQPL91:IRQPL364[4:0]	IRCN_IRQVA364	Common IRQ
365	MVA1 underflow interrupt	IRCN_IRQPL91:IRQPL365[4:0]	IRCN_IRQVA365	Common IRQ
366	MVA1 floating-point non-normalized error interrupt	IRCN_IRQPL91:IRQPL366[4:0]	IRCN_IRQVA366	Common IRQ
367	MVA1 failure detection error interrupt	IRCN_IRQPL91:IRQPL367[4:0]	IRCN_IRQVA367	Common IRQ
368	MVA1 calculation data update error interrupt	IRCN_IRQPL92:IRQPL368[4:0]	IRCN_IRQVA368	Common IRQ

IRQ Number	IRQ Factor	Priority Register	Vector Address Register	Type of IRQ
369	MVA1 R/D converter diagnosis error interrupt	IRCN_IRQPL92:IRQPL369[4:0]	IRCN_IRQVA369	Common IRQ
370	MVA1 Cumulative three-phase current abnormality detection error interrupt	IRCN_IRQPL92:IRQPL370[4:0]	IRCN_IRQVA370	Common IRQ
371	Reserved	-	-	-
372	MVA1 three-phase to two-phase DC value abnormality detection error interrupt	IRCN_IRQPL93:IRQPL372[4:0]	IRCN_IRQVA372	Common IRQ
373	MVA1 calculation overtime error interrupt	IRCN_IRQPL93:IRQPL373[4:0]	IRCN_IRQVA373	Common IRQ
374	MVA0 free-run timer 0 detection/compare clear interrupt	IRCN_IRQPL93:IRQPL374[4:0]	IRCN_IRQVA374	Common IRQ
375	MVA1 free-run timer 0 detection/compare clear interrupt	IRCN_IRQPL93:IRQPL375[4:0]	IRCN_IRQVA375	Common IRQ
376	Waveform generator dead timer underflow 0	IRCN_IRQPL94:IRQPL376[4:0]	IRCN_IRQVA376	Common IRQ
377	Waveform generator dead timer reload 0	IRCN_IRQPL94:IRQPL377[4:0]	IRCN_IRQVA377	Common IRQ
378	Waveform generator dead timer underflow 1	IRCN_IRQPL94:IRQPL378[4:0]	IRCN_IRQVA378	Common IRQ
379	Waveform generator dead timer reload 1	IRCN_IRQPL94:IRQPL379[4:0]	IRCN_IRQVA379	Common IRQ
380	Waveform generator dead timer underflow 2	IRCN_IRQPL95:IRQPL380[4:0]	IRCN_IRQVA380	Common IRQ
381	Waveform generator dead timer reload 2	IRCN_IRQPL95:IRQPL381[4:0]	IRCN_IRQVA381	Common IRQ
382	Waveform generator DTTI0, 1, 2	IRCN_IRQPL95:IRQPL382[4:0]	IRCN_IRQVA382	Common IRQ
383	Waveform generator dead timer underflow 3	IRCN_IRQPL95:IRQPL383[4:0]	IRCN_IRQVA383	Common IRQ
384	Waveform generator dead timer reload 3	IRCN_IRQPL96:IRQPL384[4:0]	IRCN_IRQVA384	Common IRQ
385	Waveform generator dead timer underflow 4	IRCN_IRQPL96:IRQPL385[4:0]	IRCN_IRQVA385	Common IRQ
386	Waveform generator dead timer reload 4	IRCN_IRQPL96:IRQPL386[4:0]	IRCN_IRQVA386	Common IRQ
387	Waveform generator dead timer underflow 5	IRCN_IRQPL96:IRQPL387[4:0]	IRCN_IRQVA387	Common IRQ
388	Waveform generator dead timer reload 5	IRCN_IRQPL97:IRQPL388[4:0]	IRCN_IRQVA388	Common IRQ
389	Waveform generator DTTI3, 4, 5	IRCN_IRQPL97:IRQPL389[4:0]	IRCN_IRQVA389	Common IRQ
390	Waveform generator dead timer underflow 6	IRCN_IRQPL97:IRQPL390[4:0]	IRCN_IRQVA390	Common IRQ
391	Waveform generator dead timer reload 6	IRCN_IRQPL97:IRQPL391[4:0]	IRCN_IRQVA391	Common IRQ
392	Waveform generator dead timer underflow 7	IRCN_IRQPL98:IRQPL392[4:0]	IRCN_IRQVA392	Common IRQ
393	Waveform generator dead timer reload 7	IRCN_IRQPL98:IRQPL393[4:0]	IRCN_IRQVA393	Common IRQ
394	Waveform generator dead timer underflow 8	IRCN_IRQPL98:IRQPL394[4:0]	IRCN_IRQVA394	Common IRQ
395	Waveform generator dead timer reload 8	IRCN_IRQPL98:IRQPL395[4:0]	IRCN_IRQVA395	Common IRQ
396	Waveform generator DTTI6, 7, 8	IRCN_IRQPL99:IRQPL396[4:0]	IRCN_IRQVA396	Common IRQ
397	Waveform generator dead timer underflow 9	IRCN_IRQPL99:IRQPL397[4:0]	IRCN_IRQVA397	Common IRQ
398	Waveform generator dead timer reload 9	IRCN_IRQPL99:IRQPL398[4:0]	IRCN_IRQVA398	Common IRQ
399	Waveform generator dead timer underflow 10	IRCN_IRQPL99:IRQPL399[4:0]	IRCN_IRQVA399	Common IRQ
400	Waveform generator dead timer reload 10	IRCN_IRQPL100:IRQPL400[4:0]	IRCN_IRQVA400	Common IRQ
401	Waveform generator dead timer underflow 11	IRCN_IRQPL100:IRQPL401[4:0]	IRCN_IRQVA401	Common IRQ
402	Waveform generator dead timer reload 11	IRCN_IRQPL100:IRQPL402[4:0]	IRCN_IRQVA402	Common IRQ
403	Waveform generator DTTI9, 10, 11	IRCN_IRQPL100:IRQPL403[4:0]	IRCN_IRQVA403	Common IRQ
404	16-bit output compare ch.0, ch.1 compare match	IRCN_IRQPL101:IRQPL404[4:0]	IRCN_IRQVA404	Common IRQ
405	16-bit output compare ch.2, ch.3 compare match	IRCN_IRQPL101:IRQPL405[4:0]	IRCN_IRQVA405	Common IRQ
406	16-bit output compare ch.4, ch.5 compare match	IRCN_IRQPL101:IRQPL406[4:0]	IRCN_IRQVA406	Common IRQ
407	16-bit output compare ch.6, ch.7 compare match	IRCN_IRQPL101:IRQPL407[4:0]	IRCN_IRQVA407	Common IRQ
408	16-bit output compare ch.8, ch.9 compare match	IRCN_IRQPL102:IRQPL408[4:0]	IRCN_IRQVA408	Common IRQ
409	16-bit output compare ch.10, ch.11 compare match	IRCN_IRQPL102:IRQPL409[4:0]	IRCN_IRQVA409	Common IRQ
410	16-bit output compare ch.12, ch.13 compare match	IRCN_IRQPL102:IRQPL410[4:0]	IRCN_IRQVA410	Common IRQ



IRQ Number	IRQ Factor	Priority Register	Vector Address Register	Type of IRQ
411	16-bit output compare ch.14, ch.15 compare match	IRCN_IRQPL102:IRQPL411[4:0]	IRCN_IRQVA411	Common IRQ
412	16-bit output compare ch.16, ch.17 compare match	IRCN_IRQPL103:IRQPL412[4:0]	IRCN_IRQVA412	Common IRQ
413	16-bit output compare ch.18, ch.19 compare match	IRCN_IRQPL103:IRQPL413[4:0]	IRCN_IRQVA413	Common IRQ
414	16-bit output compare ch.20, ch.21 compare match	IRCN_IRQPL103:IRQPL414[4:0]	IRCN_IRQVA414	Common IRQ
415	16-bit output compare ch.22, ch.23 compare match	IRCN_IRQPL103:IRQPL415[4:0]	IRCN_IRQVA415	Common IRQ
416	Up/Down counter unit0 interrupt	IRCN_IRQPL104:IRQPL416[4:0]	IRCN_IRQVA416	Common IRQ
417	Up/Down counter unit0 comparison result match detection 0	IRCN_IRQPL104:IRQPL417[4:0]	IRCN_IRQVA417	Common IRQ
418	Up/Down counter unit0 comparison result match detection 1	IRCN_IRQPL104:IRQPL418[4:0]	IRCN_IRQVA418	Common IRQ
419	Up/Down counter unit0 comparison result match detection 2	IRCN_IRQPL104:IRQPL419[4:0]	IRCN_IRQVA419	Common IRQ
420	Up/Down counter unit0 comparison result match detection 3	IRCN_IRQPL105:IRQPL420[4:0]	IRCN_IRQVA420	Common IRQ
421	Up/Down counter unit0 comparison result match detection 4	IRCN_IRQPL105:IRQPL421[4:0]	IRCN_IRQVA421	Common IRQ
422	Up/Down counter unit0 comparison result match detection 5	IRCN_IRQPL105:IRQPL422[4:0]	IRCN_IRQVA422	Common IRQ
423	Up/Down counter unit1 interrupt	IRCN_IRQPL105:IRQPL423[4:0]	IRCN_IRQVA423	Common IRQ
424	Up/Down counter unit1 comparison result match detection 0	IRCN_IRQPL106:IRQPL424[4:0]	IRCN_IRQVA424	Common IRQ
425	Up/Down counter unit1 comparison result match detection 1	IRCN_IRQPL106:IRQPL425[4:0]	IRCN_IRQVA425	Common IRQ
426	Up/Down counter unit1 comparison result match detection 2	IRCN_IRQPL106:IRQPL426[4:0]	IRCN_IRQVA426	Common IRQ
427	Up/Down counter unit1 comparison result match detection 3	IRCN_IRQPL106:IRQPL427[4:0]	IRCN_IRQVA427	Common IRQ
428	Up/Down counter unit1 comparison result match detection 4	IRCN_IRQPL107:IRQPL428[4:0]	IRCN_IRQVA428	Common IRQ
429	Up/Down counter unit1 comparison result match detection 5	IRCN_IRQPL107:IRQPL429[4:0]	IRCN_IRQVA429	Common IRQ
430	Up/Down counter unit2 interrupt	IRCN_IRQPL107:IRQPL430[4:0]	IRCN_IRQVA430	Common IRQ
431	Up/Down counter unit2 comparison result match detection 0	IRCN_IRQPL107:IRQPL431[4:0]	IRCN_IRQVA431	Common IRQ
432	Up/Down counter unit2 comparison result match detection 1	IRCN_IRQPL108:IRQPL432[4:0]	IRCN_IRQVA432	Common IRQ
433	Up/Down counter unit2 comparison result match detection 2	IRCN_IRQPL108:IRQPL433[4:0]	IRCN_IRQVA433	Common IRQ
434	Up/Down counter unit2 comparison result match detection 3	IRCN_IRQPL108:IRQPL434[4:0]	IRCN_IRQVA434	Common IRQ
435	Up/Down counter unit2 comparison result match detection 4	IRCN_IRQPL108:IRQPL435[4:0]	IRCN_IRQVA435	Common IRQ
436	Up/Down counter unit2 comparison result match detection 5	IRCN_IRQPL109:IRQPL436[4:0]	IRCN_IRQVA436	Common IRQ
437	Up/Down counter unit3 interrupt	IRCN_IRQPL109:IRQPL437[4:0]	IRCN_IRQVA437	Common IRQ



IRQ Number	IRQ Factor	Priority Register	Vector Address Register	Type of IRQ
438	Up/Down counter unit3 comparison result match detection 0	IRCN_IRQPL109:IRQPL438[4:0]	IRCN_IRQVA438	Common IRQ
439	Up/Down counter unit3 comparison result match detection 1	IRCN_IRQPL109:IRQPL439[4:0]	IRCN_IRQVA439	Common IRQ
440	Up/Down counter unit3 comparison result match detection 2	IRCN_IRQPL110:IRQPL440[4:0]	IRCN_IRQVA440	Common IRQ
441	Up/Down counter unit3 comparison result match detection 3	IRCN_IRQPL110:IRQPL441[4:0]	IRCN_IRQVA441	Common IRQ
442	Up/Down counter unit3 comparison result match detection 4	IRCN_IRQPL110:IRQPL442[4:0]	IRCN_IRQVA442	Common IRQ
443	Up/Down counter unit3 comparison result match detection 5	IRCN_IRQPL110:IRQPL443[4:0]	IRCN_IRQVA443	Common IRQ
444	4ch A/D converter unit0 conversion complete 0	IRCN_IRQPL111:IRQPL444[4:0]	IRCN_IRQVA444	Common IRQ
445	4ch A/D converter unit0 conversion complete 1	IRCN_IRQPL111:IRQPL445[4:0]	IRCN_IRQVA445	Common IRQ
446	4ch A/D converter unit0 conversion complete 2	IRCN_IRQPL111:IRQPL446[4:0]	IRCN_IRQVA446	Common IRQ
447	4ch A/D converter unit0 conversion complete 3	IRCN_IRQPL111:IRQPL447[4:0]	IRCN_IRQVA447	Common IRQ
448	4ch A/D converter unit0 conversion complete 4	IRCN_IRQPL112:IRQPL448[4:0]	IRCN_IRQVA448	Common IRQ
449	4ch A/D converter unit0 conversion complete 5	IRCN_IRQPL112:IRQPL449[4:0]	IRCN_IRQVA449	Common IRQ
450	4ch A/D converter unit0 conversion complete 6	IRCN_IRQPL112:IRQPL450[4:0]	IRCN_IRQVA450	Common IRQ
451	4ch A/D converter unit0 conversion complete 7	IRCN_IRQPL112:IRQPL451[4:0]	IRCN_IRQVA451	Common IRQ
452	4ch A/D converter unit0 range comparison 0/1/2/3/4/5/6/7	IRCN_IRQPL113:IRQPL452[4:0]	IRCN_IRQVA452	Common IRQ
453	4ch A/D converter unit1 conversion complete 0	IRCN_IRQPL113:IRQPL453[4:0]	IRCN_IRQVA453	Common IRQ
454	4ch A/D converter unit1 conversion complete 1	IRCN_IRQPL113:IRQPL454[4:0]	IRCN_IRQVA454	Common IRQ
455	4ch A/D converter unit1 conversion complete 2	IRCN_IRQPL113:IRQPL455[4:0]	IRCN_IRQVA455	Common IRQ
456	4ch A/D converter unit1 conversion complete 3	IRCN_IRQPL114:IRQPL456[4:0]	IRCN_IRQVA456	Common IRQ
457	4ch A/D converter unit1 conversion complete 4	IRCN_IRQPL114:IRQPL457[4:0]	IRCN_IRQVA457	Common IRQ
458	4ch A/D converter unit1 conversion complete 5	IRCN_IRQPL114:IRQPL458[4:0]	IRCN_IRQVA458	Common IRQ
459	4ch A/D converter unit1 conversion complete 6	IRCN_IRQPL114:IRQPL459[4:0]	IRCN_IRQVA459	Common IRQ
460	4ch A/D converter unit1 conversion complete 7	IRCN_IRQPL115:IRQPL460[4:0]	IRCN_IRQVA460	Common IRQ
461	4ch A/D converter unit1 range comparison 0/1/2/3/4/5/6/7	IRCN_IRQPL115:IRQPL461[4:0]	IRCN_IRQVA461	Common IRQ
462	16-bit input capture ch.0, ch.1 fetching	IRCN_IRQPL115:IRQPL462[4:0]	IRCN_IRQVA462	Common IRQ
463	16-bit input capture ch.2, ch.3 fetching	IRCN_IRQPL115:IRQPL463[4:0]	IRCN_IRQVA463	Common IRQ
464	16-bit input capture ch.4, ch.5 fetching	IRCN_IRQPL116:IRQPL464[4:0]	IRCN_IRQVA464	Common IRQ
465	16-bit input capture ch.6, ch.7 fetching	IRCN_IRQPL116:IRQPL465[4:0]	IRCN_IRQVA465	Common IRQ
466	16-bit input capture ch.8, ch.9 fetching	IRCN_IRQPL116:IRQPL466[4:0]	IRCN_IRQVA466	Common IRQ
467	16-bit input capture ch.10, ch.11 fetching	IRCN_IRQPL116:IRQPL467[4:0]	IRCN_IRQVA467	Common IRQ
468	16-bit input capture ch.12, ch.13 fetching	IRCN_IRQPL117:IRQPL468[4:0]	IRCN_IRQVA468	Common IRQ
469	16-bit input capture ch.14 fetching	IRCN_IRQPL117:IRQPL469[4:0]	IRCN_IRQVA469	Common IRQ
470	12-bit ADC conversion complete 0	IRCN_IRQPL117:IRQPL470[4:0]	IRCN_IRQVA470	Common IRQ
471	12-bit ADC conversion complete 1	IRCN_IRQPL117:IRQPL471[4:0]	IRCN_IRQVA471	Common IRQ
472	12-bit ADC conversion complete 2	IRCN_IRQPL118:IRQPL472[4:0]	IRCN_IRQVA472	Common IRQ
473	12-bit ADC conversion complete 3	IRCN_IRQPL118:IRQPL473[4:0]	IRCN_IRQVA473	Common IRQ
474	12-bit ADC conversion complete 4	IRCN_IRQPL118:IRQPL474[4:0]	IRCN_IRQVA474	Common IRQ
475	12-bit ADC conversion complete 5	IRCN_IRQPL118:IRQPL475[4:0]	IRCN_IRQVA475	Common IRQ
476	12-bit ADC conversion complete 6	IRCN_IRQPL119:IRQPL476[4:0]	IRCN_IRQVA476	Common IRQ
477	12-bit ADC conversion complete 7	IRCN_IRQPL119:IRQPL477[4:0]	IRCN_IRQVA477	Common IRQ
478	12-bit ADC conversion complete 8	IRCN_IRQPL119:IRQPL478[4:0]	IRCN_IRQVA478	Common IRQ





IRQ Number	IRQ Factor	Priority Register	Vector Address Register	Type of IRQ
479	12-bit ADC conversion complete 9	IRCN_IRQPL119:IRQPL479[4:0]	IRCN_IRQVA479	Common IRQ
480	12-bit ADC conversion complete 10	IRCN_IRQPL120:IRQPL480[4:0]	IRCN_IRQVA480	Common IRQ
481	12-bit ADC conversion complete 11	IRCN_IRQPL120:IRQPL481[4:0]	IRCN_IRQVA481	Common IRQ
482	12-bit ADC conversion complete 12	IRCN_IRQPL120:IRQPL482[4:0]	IRCN_IRQVA482	Common IRQ
483	12-bit ADC conversion complete 13	IRCN_IRQPL120:IRQPL483[4:0]	IRCN_IRQVA483	Common IRQ
484	12-bit ADC conversion complete 14	IRCN_IRQPL121:IRQPL484[4:0]	IRCN_IRQVA484	Common IRQ
485	12-bit ADC conversion complete 15	IRCN_IRQPL121:IRQPL485[4:0]	IRCN_IRQVA485	Common IRQ
486	12-bit ADC conversion complete 16	IRCN_IRQPL121:IRQPL486[4:0]	IRCN_IRQVA486	Common IRQ
487	12-bit ADC conversion complete 17	IRCN_IRQPL121:IRQPL487[4:0]	IRCN_IRQVA487	Common IRQ
488	12-bit ADC conversion complete 18	IRCN_IRQPL122:IRQPL488[4:0]	IRCN_IRQVA488	Common IRQ
489	12-bit ADC conversion complete 19	IRCN_IRQPL122:IRQPL489[4:0]	IRCN_IRQVA489	Common IRQ
490	12-bit ADC conversion complete 20	IRCN_IRQPL122:IRQPL490[4:0]	IRCN_IRQVA490	Common IRQ
491	12-bit ADC conversion complete 21	IRCN_IRQPL122:IRQPL491[4:0]	IRCN_IRQVA491	Common IRQ
492	12-bit ADC conversion complete 22	IRCN_IRQPL123:IRQPL492[4:0]	IRCN_IRQVA492	Common IRQ
493	12-bit ADC conversion complete 23	IRCN_IRQPL123:IRQPL493[4:0]	IRCN_IRQVA493	Common IRQ
494	12-bit ADC conversion complete 24	IRCN_IRQPL123:IRQPL494[4:0]	IRCN_IRQVA494	Common IRQ
495	12-bit ADC conversion complete 25	IRCN_IRQPL123:IRQPL495[4:0]	IRCN_IRQVA495	Common IRQ
496	12-bit ADC conversion complete 26	IRCN_IRQPL124:IRQPL496[4:0]	IRCN_IRQVA496	Common IRQ
497	12-bit ADC conversion complete 27	IRCN_IRQPL124:IRQPL497[4:0]	IRCN_IRQVA497	Common IRQ
498	12-bit ADC conversion complete 28	IRCN_IRQPL124:IRQPL498[4:0]	IRCN_IRQVA498	Common IRQ
499	12-bit ADC conversion complete 29	IRCN_IRQPL124:IRQPL499[4:0]	IRCN_IRQVA499	Common IRQ
500	12-bit ADC conversion complete 30	IRCN_IRQPL125:IRQPL500[4:0]	IRCN_IRQVA500	Common IRQ
501	12-bit ADC conversion complete 31	IRCN_IRQPL125:IRQPL501[4:0]	IRCN_IRQVA501	Common IRQ
502	12-bit ADC range comparison 0/1/2/3/4/5/6/7	IRCN_IRQPL125:IRQPL502[4:0]	IRCN_IRQVA502	Common IRQ
503	12-bit ADC range comparison 8/9/10/11/12/13/14/15	IRCN_IRQPL125:IRQPL503[4:0]	IRCN_IRQVA503	Common IRQ
504	12-bit ADC range comparison 16/17/18/19/20/21/22/23	IRCN_IRQPL126:IRQPL504[4:0]	IRCN_IRQVA504	Common IRQ
505	12-bit ADC range comparison 24/25/26/27/28/29/30/31	IRCN_IRQPL126:IRQPL505[4:0]	IRCN_IRQVA505	Common IRQ
506	12-bit ADC scan conversion complete	IRCN_IRQPL126:IRQPL506[4:0]	IRCN_IRQVA506	Common IRQ
507	PLL gear for FlexRay	IRCN_IRQPL126:IRQPL507[4:0]	IRCN_IRQVA507	Common IRQ
508	PLL alarm for FlexRay	IRCN_IRQPL127:IRQPL508[4:0]	IRCN_IRQVA508	Common IRQ
509 to 511	Reserved	-	-	-

### 3.2. List of NMI Factors

This section shows list of NMI factors.

The NMI factors include the NMI by individual CPU and by both CPU. The former is called individual NMI, the latter is called common NMI.

List of NMI factors shows the following.

NMI Number	NMI Factor	Priority Register	Vector Address Register	Type of NMI
0	NMIX pin	IRCN_NMIPL0:NMIPL0[4:0]	IRCN_NMIVA0	Common NMI
1 to 3	Reserved	-	-	-
4	Low-voltage detection	IRCN_NMIPL1:NMIPL4[4:0]	IRCN_NMIVA4	Common NMI
5	Clock supervisor, profile	IRCN_NMIPL1:NMIPL5[4:0]	IRCN_NMIVA5	Common NMI
6	Hardware watchdog	IRCN_NMIPL1:NMIPL6[4:0]	IRCN_NMIVA6	Common NMI
7	Software watchdog unit0, 1	IRCN_NMIPL1:NMIPL7[4:0]	IRCN_NMIVA7	Individual NMI
8	Interrupt controller unit0, 1 ECC 2-bit error detection	IRCN_NMIPL2:NMIPL8[4:0]	IRCN_NMIVA8	Individual NMI
9	CPU Livelock	IRCN_NMIPL2:NMIPL9[4:0]	IRCN_NMIVA9	Individual NMI
10 to 12	Reserved	-	-	-
13	Memory protection(MPU) protection violation interrupt	IRCN_NMIPL3:NMIPL13[4:0]	IRCN_NMIVA13	Common NMI
14 to 17	Reserved	-	-	-
18	Time protection(TPU) unit0, 1 protection violation interrupt	IRCN_NMIPL4:NMIPL18[4:0]	IRCN_NMIVA18	Individual NMI
19 to 23	Reserved	-	-	-
24	R/D converter ch.0 abnormal	IRCN_NMIPL6:NMIPL24[4:0]	IRCN_NMIVA24	Common NMI
25	R/D converter ch.1 abnormal	IRCN_NMIPL6:NMIPL25[4:0]	IRCN_NMIVA25	Common NMI
26 to 31	Reserved	-	-	-

(n=0, 1)



### 3.3. List of DMA Activation Factors

This section shows list of DMA activation factors.

DMA Client Number	DMA Activation Factor
0 to 7	Reserved
8	WorkFLASH 0 DMA transfer request
9	WorkFLASH 1 DMA transfer request
10	External interrupt ch.4
11	External interrupt ch.5
12	External interrupt ch.6
13	External interrupt ch.7
14	Multi-function serial interface ch.0 reception complete
15	Multi-function serial interface ch.0 transmission complete
16	Multi-function serial interface ch.3 reception complete
17	Multi-function serial interface ch.3 transmission complete
18	Multi-function serial interface ch.4 reception complete
19	Multi-function serial interface ch.4 transmission complete
20	Base timer ch.4 IRQ0
21	Base timer ch.4 IRQ1
22	Base timer ch.5 IRQ0
23	Base timer ch.5 IRQ1
24	Base timer ch.8 IRQ0
25	Base timer ch.8 IRQ1
26	Base timer ch.9 IRQ0
27	Base timer ch.9 IRQ1
28	Base timer ch.10 IRQ0
29	Base timer ch.10 IRQ1
30	Base timer ch.11 IRQ0
31	Base timer ch.11 IRQ1
32	32-bit free-run timer ch.0 compare clear interrupt
33	32-bit free-run timer ch.0 0 detection interrupt
34	32-bit free-run timer ch.1 compare clear interrupt
35	32-bit free-run timer ch.1 0 detection interrupt
36	32-bit input capture ch.0 fetching
37	32-bit input capture ch.1 fetching
38	32-bit input capture ch.2 fetching
39	FlexRay output buffer DMA transfer request
40	FlexRay input buffer DMA transfer request
41	16-bit free-run timer ch.0 compare clear interrupt
42	16-bit free-run timer ch.0 0 detection interrupt
43	16-bit free-run timer ch.1 compare clear interrupt
44	16-bit free-run timer ch.1 0 detection interrupt
45	16-bit free-run timer ch.2 compare clear interrupt
46	16-bit free-run timer ch.2 0 detection interrupt
47	16-bit free-run timer ch.6 compare clear interrupt
48	16-bit free-run timer ch.6 0 detection interrupt
49	16-bit free-run timer ch.7 compare clear interrupt
50	16-bit free-run timer ch.7 0 detection interrupt
51	16-bit free-run timer ch.8 compare clear interrupt

DMA Client Number	DMA Activation Factor
52	16-bit free-run timer ch.8 0 detection interrupt
53	16-bit free-run timer ch.12 compare clear interrupt
54	16-bit free-run timer ch.12 0 detection interrupt
55	16-bit free-run timer ch.13 compare clear interrupt
56	16-bit free-run timer ch.13 0 detection interrupt
57	16-bit free-run timer ch.14 compare clear interrupt
58	16-bit free-run timer ch.14 0 detection interrupt
59	MVA0 angular calculation end interrupt
60	MVA0 three-phase current normalization end interrupt
61	MVA0 three-phase to two-phase DC conversion end interrupt
62	MVA0 PID control end interrupt
63	MVA0 current to voltage conversion end interrupt
64	MVA0 two-phase to three-phase AC conversion end interrupt
65	MVA0 free-run timer compare clear interrupt
66	MVA1 angular calculation end interrupt
67	MVA1 three-phase current normalization end interrupt
68	MVA1 three-phase to two-phase DC conversion end interrupt
69	MVA1 PID control end interrupt
70	MVA1 current to voltage conversion end interrupt
71	MVA1 two-phase to three-phase AC conversion end interrupt
72	MVA1 free-run timer compare clear interrupt
73	Waveform generator DTTI0, 1, 2
74	Waveform generator DTTI3, 4, 5
75	Waveform generator dead timer underflow 6
76	Waveform generator dead timer reload 6
77	Waveform generator dead timer underflow 7
78	Waveform generator dead timer reload 7
79	Waveform generator dead timer underflow 8
80	Waveform generator dead timer reload 8
81	Waveform generator DTTI6, 7, 8
82	Waveform generator DTTI9, 10, 11
83	16-bit output compare ch.0 compare match
84	16-bit output compare ch.1 compare match
85	16-bit output compare ch.2 compare match
86	16-bit output compare ch.3 compare match
87	16-bit output compare ch.4 compare match
88	16-bit output compare ch.5 compare match
89	16-bit output compare ch.6 compare match
90	16-bit output compare ch.7 compare match
91	16-bit output compare ch.8 compare match
92	16-bit output compare ch.9 compare match
93	16-bit output compare ch.10 compare match
94	16-bit output compare ch.11 compare match
95	16-bit output compare ch.12 compare match
96	16-bit output compare ch.13 compare match
97	16-bit output compare ch.14 compare match
98	16-bit output compare ch.15 compare match
99	16-bit output compare ch.16 compare match
100	16-bit output compare ch.17 compare match

DMA Client Number	DMA Activation Factor
101	Up/Down counter unit0 compare result match detection 0
102	Up/Down counter unit0 compare result match detection 1
103	Up/Down counter unit0 compare result match detection 2
104	Up/Down counter unit1 compare result match detection 0
105	Up/Down counter unit1 compare result match detection 1
106	Up/Down counter unit1 compare result match detection 2
107	Up/Down counter unit1 compare result match detection 3
108	Up/Down counter unit1 compare result match detection 4
109	Up/Down counter unit1 compare result match detection 5
110	Up/Down counter unit3 compare result match detection 0
111	Up/Down counter unit3 compare result match detection 1
112	Up/Down counter unit3 compare result match detection 2
113	Up/Down counter unit3 compare result match detection 3
114	Up/Down counter unit3 compare result match detection 4
115	Up/Down counter unit3 compare result match detection 5
116	4ch A/D converter unit0 conversion complete 0
117	4ch A/D converter unit0 conversion complete 4
118	4ch A/D converter unit1 conversion complete 0
119	4ch A/D converter unit1 conversion complete 4
120	16-bit input capture ch.6 fetching
121	16-bit input capture ch.7 fetching
122	16-bit input capture ch.8 fetching
123	16-bit input capture ch.9 fetching
124	16-bit input capture ch.10 fetching
125	16-bit input capture ch.11 fetching
126	16-bit input capture ch.12 fetching
127	12-bit ADC conversion complete 0
128	12-bit ADC conversion complete 1
129	12-bit ADC conversion complete 2
130	12-bit ADC conversion complete 3
131	12-bit ADC conversion complete 4
132	12-bit ADC conversion complete 5
133	12-bit ADC conversion complete 6
134	12-bit ADC conversion complete 7
135	12-bit ADC conversion complete 8
136	12-bit ADC conversion complete 9
137	12-bit ADC conversion complete 10
138	12-bit ADC conversion complete 11
139	12-bit ADC conversion complete 12
140	12-bit ADC conversion complete 13
141	12-bit ADC conversion complete 14
142	12-bit ADC conversion complete 15

**Note:**

- Since the interrupt request used transfer request also handles as request to CPU, set the interrupt controller to the interrupt disabled.

## 4. Master Access

This section shows restriction each bus master access to each slave module.

Slave Module	Bus Master		
	CPU0	CPU1	DMAC
CPU0	○	○	○
CPU1	○	○	○
TCFLASH0(AXI)	○	○	○
TCFLASH1(AXI)	○	○	○
EAM	○	○	○
SYSC	○	○*	×
MODEC	○	○	×
HW-WDT	○	× (2CPU mode) ○ (1CPU1 mode)	×
SW-WDT0	○	×	×
SW-WDT1	×	○	×
EXT-IRC	○	○	○
WorkFLASH0(Memory)	○	○	○
WorkFLASH0(I/O)	○	○	○
WorkFLASH1(Memory)	○	○	○
WorkFLASH1(I/O)	○	○	○
BootROM(Memory)	○	○	○
BootROM(IO)	○	○	○
TPU0	○	×	○
TPU1	×	○	○
IRC0	○	×	○
IRC1	×	○	○
NMI Distributor	○	○	○
IPCU	○	○	○
TCFLASH0(I/O)	○	×	○
TCRAM0(I/O)	○	×	○
TCFLASH1(I/O)	×	○	○
TCRAM1(I/O)	×	○	○
DMAC	○	○	×
MPU	○	○	×
CR Calibration	○	○	○
CAN Prescaler	○	○	○
CAN	○	○	○
CRC	○	○	○
GPIO	○	○	○
PPC	○	○	○
RIC	○	○	○
MFS	○	○	○
BASE TIMER	○	○	○
32-bitFRT	○	○	○
32-bitICU	○	○	○
Application Specific Peripheral Group A	○	○	○
Application Specific Peripheral Group B	○	○	○

○: accessible , ×: non-accessible

\*: In 2CPU mode, some bits are accessible from CPU1.



## 5. Low-power Mode

This section shows setting for low-power mode in MB9D560 series.

### Low-power Mode Setting

Category	Item	PSS Setting	Stop Mode	Watch Mode
Clock enable	PERI0 clock oscillation enable	Programmable	Oscillation disable	Oscillation disable
	PERI1 clock oscillation enable	Programmable	Oscillation disable	Oscillation disable
	PERI4 clock oscillation enable	Programmable	Oscillation disable	Oscillation disable
	PERI5 clock oscillation enable	Programmable	Oscillation disable	Oscillation disable
	PERI6 clock oscillation enable	Programmable	Oscillation disable	Oscillation disable
	PERI7 clock oscillation enable	Programmable	Oscillation disable	Oscillation disable
	SYSCPD1 clock oscillation enable	Programmable	Oscillation disable	Oscillation disable
	MEMC clock oscillation enable	Programmable	Oscillation disable	Oscillation disable
	DMA clock oscillation enable	Programmable	Oscillation disable	Oscillation disable
	HPMPD2 clock oscillation enable	Programmable	Oscillation disable	Oscillation disable
	TRC clock oscillation enable	Programmable	Oscillation disable	Oscillation disable
	ATB clock oscillation enable	Programmable	Oscillation disable	Oscillation disable
	DBG clock oscillation enable	Programmable	Oscillation disable	Oscillation disable
Source clock oscillation	Slow-CR	Programmable	Oscillation disable	Oscillation disable
	Fast-CR	Programmable	Oscillation disable	Oscillation disable
	Main oscillation	Programmable	Oscillation disable	Oscillation enable
	PLL	Programmable	Oscillation disable	Oscillation disable
Clock select	Clock domain0 clock select	Programmable	Fixed at "L"	Fixed at "L"

MB9D560 series support only stop mode and watch mode.



6. Major Changes

Page	Section	Change Results
Revision 4.0		
-	-	Initial release





MN708-00002-4v0-E

---

**Spansion • Controller Manual**

MB9D560 Series

32-bit Microcontroller

Spansion® Traveo™ Family

MB9DF564/F565/F566

Hardware Manual

---

March 2015 Rev. 4.0

Published:    Spansion Inc.

Edited:        Communications

---

**Colophon**

The products described in this document are designed, developed and manufactured as contemplated for general use, including without limitation, ordinary industrial use, general office use, personal use, and household use, but are not designed, developed and manufactured as contemplated (1) for any use that includes fatal risks or dangers that, unless extremely high safety is secured, could have a serious effect to the public, and could lead directly to death, personal injury, severe physical damage or other loss (i.e., nuclear reaction control in nuclear facility, aircraft flight control, air traffic control, mass transport control, medical life support system, missile launch control in weapon system), or (2) for any use where chance of failure is intolerable (i.e., submersible repeater and artificial satellite). Please note that Spansion will not be liable to you and/or any third party for any claims or damages arising in connection with above-mentioned uses of the products. Any semiconductor devices have an inherent chance of failure. You must protect against injury, damage or loss from such failures by incorporating safety design measures into your facility and equipment such as redundancy, fire protection, and prevention of over-current levels and other abnormal operating conditions. If any products described in this document represent goods or technologies subject to certain restrictions on export under the Foreign Exchange and Foreign Trade Law of Japan, the US Export Administration Regulations or the applicable laws of any other country, the prior authorization by the respective government entity will be required for export of those products.

**Trademarks and Notice**

The contents of this document are subject to change without notice. This document may contain information on a Spansion product under development by Spansion. Spansion reserves the right to change or discontinue work on any product without notice. The information in this document is provided as is without warranty or guarantee of any kind as to its accuracy, completeness, operability, fitness for particular purpose, merchantability, non-infringement of third-party rights, or any other warranty, express, implied, or statutory. Spansion assumes no liability for any damages of any kind arising out of the use of the information in this document.

Copyright © 2014-2015 Spansion All rights reserved. Spansion®, the Spansion logo, MirrorBit®, MirrorBit® Eclipse™, ORNAND™, Easy DesignSim™, Traveo™ and combinations thereof, are trademarks and registered trademarks of Spansion LLC in the United States and other countries. ARM is the registered trademark of ARM Limited in the EU and other countries. Cortex, MPCore, CoreSight are the trademarks of ARM Limited in the EU and other countries. Other names used are for informational purposes only and may be trademarks of their respective owners.