Please note that Cypress is an Infineon Technologies Company.
The document following this cover page is marked as “Cypress” document as this is the company that originally developed the product. Please note that Infineon will continue to offer the product to new and existing customers as part of the Infineon product portfolio.

Continuity of document content
The fact that Infineon offers the following product as part of the Infineon product portfolio does not lead to any changes to this document. Future revisions will occur when appropriate, and any changes will be set out on the document history page.

Continuity of ordering part numbers
Infineon continues to support existing part numbers. Please continue to use the ordering part numbers listed in the datasheet for ordering.

S6J3300 Series

32-bit Microcontroller

Traveo™ Family Hardware Manual

Document Number: 002-10185 Rev. *H

Cypress Semiconductor
An Infineon Technologies Company
198 Champion Court
San Jose, CA 95134-1709

www.cypress.com
www.infineon.com

Copyrights

© Cypress Semiconductor Corporation, 2015-2020. This document is the property of Cypress Semiconductor Corporation and its subsidiaries (“Cypress”). This document, including any software or firmware included or referenced in this document (“Software”), is owned by Cypress under the intellectual property laws and treaties of the United States and other countries worldwide. Cypress reserves all rights under such laws and treaties and does not, except as specifically stated in this paragraph, grant any license under its patents, copyrights, trademarks, or other intellectual property rights. If the Software is not accompanied by a license agreement and you do not otherwise have a written agreement with Cypress governing the use of the Software, then Cypress hereby grants you a personal, non-exclusive, nontransferable license (without the right to sublicense) (1) under its copyright rights in the Software (a) for Software provided in source code form, to modify and reproduce the Software solely for use with Cypress hardware products, only internally within your organization, and (b) to distribute the Software in binary code form externally to end users (either directly or indirectly through resellers and distributors), solely for use on Cypress hardware product units, and (2) under those claims of Cypress’s patents that are infringed by the Software (as provided by Cypress, unmodified) to make, use, distribute, and import the Software solely for use with Cypress hardware products. Any other use, reproduction, modification, translation, or compilation of the Software is prohibited.

TO THE EXTENT PERMITTED BY APPLICABLE LAW, CYPRESS MAKES NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS DOCUMENT OR ANY SOFTWARE OR ACCOMPANYING HARDWARE, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. No computing device can be absolutely secure. Therefore, despite security measures implemented in Cypress hardware or software products, Cypress shall have no liability arising out of any security breach, such as unauthorized access to or use of a Cypress product. CYPRESS DOES NOT REPRESENT, WARRANT, OR GUARANTEE THAT CYPRESS PRODUCTS, OR SYSTEMS CREATED USING CYPRESS PRODUCTS, WILL BE FREE FROM CORRUPTION, ATTACK, VIRUSES, INTERFERENCE, HACKING, DATA LOSS OR THEFT, OR OTHER SECURITY INTRUSION (collectively, “Security Breach”). Cypress disclaims any liability relating to any Security Breach, and you shall and hereby do release Cypress from any claim, damage, or other liability arising from any Security Breach. In addition, the products described in these materials may contain design defects or errors known as errata which may cause the product to deviate from published specifications. To the extent permitted by applicable law, Cypress reserves the right to make changes to this document without further notice. Cypress does not assume any liability arising out of the application or use of any product or circuit described in this document. Any information provided in this document, including any sample design information or programming code, is provided only for reference purposes. It is the responsibility of the user of this document to properly design, program, and test the functionality and safety of any application made of this information and any resulting product. “High-Risk Device” means any device or system whose failure could cause personal injury, death, or property damage. Examples of High-Risk Devices are weapons, nuclear installations, surgical implants, and other medical devices. “Critical Component” means any component of a High-Risk Device whose failure to perform can be reasonably expected to cause, directly or indirectly, the failure of the High-Risk Device, or to affect its safety or effectiveness. Cypress is not liable, in whole or in part, and you shall and hereby do release Cypress from any claim, damage, or other liability arising from any use of a Cypress product as a Critical Component in a High-Risk Device. You shall indemnify and hold Cypress, its directors, officers, employees, agents, affiliates, distributors, and assigns harmless from and against all claims, costs, damages, and expenses, arising out of any claim, including claims for product liability, personal injury or death, or property damage arising from any use of a Cypress product as a Critical Component in a High-Risk Device. Cypress products are not intended or authorized for use as a Critical Component in any High-Risk Device except to the limited extent that (i) Cypress’s published data sheet for the product explicitly states Cypress has qualified the product for use in a specific High-Risk Device, or (ii) Cypress has given you advance written authorization to use the product as a Critical Component in the specific High-Risk Device and you have signed a separate indemnification agreement.

Cypress, the Cypress logo, Spansion, the Spansion logo, and combinations thereof, WICED, PSoC, CapSense, EZ-USB, F-RAM, and Traveo are trademarks or registered trademarks of Cypress in the United States and other countries. For a more complete list of Cypress trademarks, visit cypress.com. Other names and brands may be claimed as property of their respective owners.

Arm® and Cortex® are registered trademarks of Arm Limited (or its subsidiaries) in the US and/or elsewhere.

Preface

Thank you for your continued use of Cypress semiconductor products.

Read this manual and "Data Sheet" thoroughly before using products in this family.

Purpose of This Manual and Intended Readers

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.

** This manual explains the configuration and operation of the peripheral functions, but does not cover the specifics of each device in the family.*

Users should refer to the respective data sheets of devices for device-specific details.

Microcontroller support information:

www.cypress.com/support/microcontrollers

Table of Contents

CHAPTER 1: Overview 19
1. Overview 20
2. Document Definition 20
3. Register Attribute 20
   3.1. Read and Write 20
   3.2. Protection 21
   3.3. Register Information 21
4. Vocabulary 22

CHAPTER 2: Function List 24
1. Function List 25
2. Optional Function 27
   2.1. Basic Option 27
   2.2. ID 28
   2.3. Restriction 28

CHAPTER 3: Product Description 30
1. Overview 31
2. Product Description 31
   2.1. Ethernet 36
   1.1. Reset Signal 37
3. Note 38
   3.1. Status Flag Clear 38
   3.2. Error Response 38
   3.3. Restriction 40

CHAPTER 4: Block Diagram 41
1. Block Diagram 42
2. Note 42

CHAPTER 5: Clock Configuration 43
1. Overview 44
2. Operation 44
   2.1. Spread Spectrum Clock Generator (SSCG) 47
3. Remark 47

CHAPTER 6: Operation Mode 48
1. Overview 49
2. Configuration 49
3. Registers 49

CHAPTER 7: Memory and Base Address Map 50
1. Overview 51
2. Memory Map 51
3. Base Address Map 53

CHAPTER 8: IRQ Map / NMI Map 60
1. IRQ MAP 61
2. NMI MAP 68

CHAPTER 9: DMA Channel Activation Factors 69
1. Factors List 70

2. Note 75
CHAPTER 10: Port Description 76
CHAPTER 11: Port Configuration 77
1. Overview 78
2. Configuration and Block Diagram 78
3. Operation 79
3.1. Resource Input Configuration Module 79
3.2. Port Output Function Configuration 151
3.3. The Analog I/O Setting 161
3.4. Input Level Setting 162
3.5. Output Drive Capacity Setting 166
3.6. Port Status 170
3.7. Function Port Group 171
3.8. Key Code Register 175
4. Registers 175
5. Configuration Procedure 175
5.1. Resource I/O Port (Both Direction) 176
5.2. Resource Input 177
5.3. Resource Output 178
5.4. Port Function Input 179
5.5. Port Function Output 180
5.6. Analog Function Input or Output 181
6. Precautions 182
6.1. Noise Filter 182
6.2. I^2C setting 183
CHAPTER 12: State Transition 184
1. Overview 185
2. Diagram of State Transition 186
3. Fetching the Operation Mode 188
4. Changes to PSS and RUN 192
CHAPTER 13: Low Voltage Detection 195
1. Overview 196
2. Configuration of Supervised Power Domain 196
3. Operation 197
3.1. LVD operation 197
3.2. Configurations 197
4. Registers 198
CHAPTER 14: Serial Programming 201
1. Overview 202
2. Memory Map 202
3. Flash Sector Configuration 202
4. Port Configuration 202
5. Operation 202
5.1. Timing Chart 202
5.2. RAM Executing Communication Protocol 203
5.3. Operation Flow 205
6. Note. 206
6.1. PSC Port State 206
CHAPTER 15: 12-/10-/8-Bit Analog to Digital Converter 207
1. Overview 208
2. Configuration and Block Diagram 210
3. Operation of A/D Converter 211

3.1. A/D Conversion Flow 211
3.2. Logical Channel Mapping to Analog Input Signals 212
3.3. Logical Channel Data Protection Function 213
3.4. Logical Channel Triggering and Priority 214
3.5. Group Processing 215
3.6. Multiple Conversion Logical Channels 223
3.7. Forced Stop 225
3.8. A/D Converter Calibration 226
3.9. DMA Transfer Function 228
3.10. Range Comparator Function 228
3.11. Pulse Detection Function 231
3.12. Debug Mode 233
4. Setup Procedure Examples 234
4.1. Control of A/D Conversion 234
4.2. Setting of Global A/D Conversion 235
4.3. Setting of Logical Channel 236
4.4. Setting of Range Comparator 237
4.5. Setting of Pulse Detection 238
4.6. A/D Conversion 238
4.7. Range Comparator 241
4.8. Pulse Detection 242
5. Registers 243
5.1. A/D Channel Control Registers (ADC12Bn_CHCTRL0 to 63) 244
5.2. A/D Channel Status Registers (ADC12Bn_CHSTAT0 to 63) 247
5.3. A/D Conversion Data Registers (ADC12Bn_CD0 to 63) 249
5.4. Pulse Counter Control Registers (ADC12Bn_PCCTRL0 to 63) 251
5.5. A/D Conversion Done Interrupt Flag Registers (ADC12Bn_CDONEIRQ0 to 1) 253
5.6. A/D Conversion Done Interrupt Enable Registers (ADC12Bn_CDONEIRQE0 to 1) 256
5.7. A/D Conversion Done Interrupt Clear Registers (ADC12Bn_CDONEIRQC0 to 1) 258
5.8. Group Interrupted Interrupt Flag Registers (ADC12Bn_GRPIRQ0 to 1) 260
5.9. Group Interrupted Interrupt Enable Registers (ADC12Bn_GRPIRQE0 to 1) 262
5.10. Group Interrupted Interrupt Clear Registers (ADC12Bn_GRPIRQC0 to 1) 264
5.11. Range Comparator Interrupt Flag Registers (ADC12Bn_RCIRQ0 to 1) 266
5.12. Range Comparator Interrupt Enable Registers (ADC12Bn_RCIRQE0 to 1) 268
5.13. Range Comparator Interrupt Clear Registers (ADC12Bn_RCIRQC0 to 1) 270
5.14. Pulse Counter Interrupt Flag Registers (ADC12Bn_PCIRQ0 to 1) 272
5.15. Pulse Counter Interrupt Enable Registers (ADC12Bn_PCIRQE0 to 1) 274
5.16. Pulse Counter Interrupt Clear Registers (ADC12Bn_PCIRQC0 to 1) 276
5.17. A/D Channel Trigger Status Flag Registers (ADC12Bn_TRGST0 to 1) 278
5.18. A/D Channel Trigger Clear Registers (ADC12Bn_TRGCL0 to 1) 281
5.19. A/D Channel Trigger Overrun Flag Registers (ADC12Bn_TRGOR0 to 1) 283
5.20. A/D Channel Trigger Overrun Clear Registers (ADC12Bn_TRGORC0 to 1) 285
5.21. Range Comparator Over Threshold Flag Registers (ADC12Bn_RCOTF0 to 1) 287
5.22. Conversion Done DMA Select Registers (ADC12Bn_CDDS0 to 3) 289
5.23. A/D Converter Comparison Time Setting Register (ADC12Bn_CT) 290
5.24. A/D Converter Resumption Time Setting Register (ADC12Bn_RT) 291
5.25. A/D Converter Sampling Time Setting Registers (ADC12Bn_ST0 to 3) 292
5.26. A/D Converter Offset Compensation Setting Register (ADC12Bn_OCV) 293
5.27. A/D Converter Gain Compensation Setting Register (ADC12B_GCV) 294
5.28. A/D Converter Global Control Register (ADC12Bn_CTRL) 295
5.29. A/D Converter Global Status Register (ADC12Bn_STAT) 297
5.30. Range Comparator Upper Threshold Registers (ADC12Bn_RCOH0 to 7) 299

5.31. Range Comparator Lower Threshold Registers (ADC12Bn_RCOL0 to 7) 300
5.32. Full Range Comparator Upper Threshold Registers (ADC12Bn_FRCOH0 to 7) 301
5.33. Full Range Comparator Lower Threshold Registers (ADC12Bn_FRCOL0 to 7) 302
5.34. A/D Multiple Conversion Channel Control Registers (ADC12Bn_MCCTRL0 to 3) 303
5.35. A/D Multiple Conversion Channel Status Registers (ADC12Bn_MCSTAT0 to 3) 305
CHAPTER 16: Stepper Motor Controller 306
1. Overview 307
2. Configuration and Block Diagram 307
3. Operation of the Stepper Motor Controller 309
3.1. Operation of PWM Pulse Generator 309
3.2. Selection of Motor Drive Signals 312
3.3. Synchronization System of Stepper Motor Controller 313
3.4. Operation of the SMC Trigger Delay Function 313
4. Registers 315
4.1. PWM Control Register (PWC) 316
4.2. PWM1 and PWM2 Compare Registers (PWC1, PWC2) 318
4.3. PWM Selection Register (PWS) 320
4.4. PWM Selection Set Register (PWSS) 323
4.5. SMC Trigger Delay Register (PTRGDL) 324
CHAPTER 17: Trigger Configuration of Stepper Motor Controller 325
1. Overview 326
2. Configuration and Block Diagram 326
3. Operation 327
4. Registers 327
4.1. SMC Trigger Selection Register (SMCTGg_PTRGS) 327
4.2. SMC Trigger Register (SMCTGg_PTRG) 330
CHAPTER 18: Sound Generator 332
1. Overview 333
2. Configuration 334
3. Operations 335
3.1. Relation between the Amplitude Data Register (SGAR) and PWM Pulse 336
3.2. Relation between the Frequency Data Register (SGFR), the Extended Frequency Data Register (SGEFR) and Tone Pulse Signal 337
3.3. Relation between the PWM Cycle Data Register (SGPCR) and PWM Cycle 339
3.4. Relation between the DMA Transfer Update Enable Register (SGDER) and DMA Settings 340
3.5. Sound Generator Operation 345
3.6. Interrupt of Sound Generator 371
4. Registers 372
4.1. DMA Transfer Update Enable Register (SGDER) 373
4.2. Sound Control Register (SGCR) 375
4.3. Amplitude Data Register (SGAR) 379
4.4. Frequency Data Register (SGFR) 380
4.5. Tone Output Number Register (SGNR) 381
4.6. Time Cycle Register (SGTCR) 382
4.7. Increase and Decrease Data Register (SGIDR) 383
4.8. PWM Cycle Data Register (SGPCR) 384
4.9. Extended Frequency Data Register (SGEFR) 385
4.10. DMA Transfer Intermediate Register (SGDMAR) 386
4.11. Interrupt Clear Register (SGCCR) 387
CHAPTER 19: Sound Waveform Generator 388
1. Overview 389

2. Configuration and Block Diagram 390
3. Operation of the Sound Waveform Generator 391
3.1. Sound Source Specifications 391
3.2. Interrupt 393
3.3. SWFG Sound Source Generation Start 393
3.4. SWFG Sound Source Generation Stop 393
3.5. Continuous SWFG Sound Source Generation 394
3.6. Filter 395
4. Registers 397
4.1. Waveform Generator Channel Enable Register (WGCHEN) 397
4.2. Waveform Generator Channel Start Register (WGCHSTART) 399
4.3. Waveform Generator Channel Address1 Register (WGCHADD1) 401
4.4. Waveform Generator Channel Address2 Register (WGCHADD2) 402
4.5. Waveform Generator Channel Address3 Register (WGCHADD3) 403
4.6. Waveform Generator Channel Monaural Register (WGCHMONO) 404
4.7. Waveform Generator Channel n Control Register1 (WGCHnCTRL1, n=0 to 4) 405
4.8. Waveform Generator Channel n Control Register2 (WGCHnCTRL2, n=0 to 4) 409
4.9. Waveform Generator Channel n Control Register3 (WGCHnCTRL3, n=0 to 4) 411
4.10. Waveform Generator Channel Clear Register (WGCHCLR) 413
4.11. Waveform Generator Interrupt Enable Register (WGINTREN) 415
4.12. Waveform Generator Interrupt State Register (WGINTRSTATE) 417
4.13. Waveform Generator Interrupt Clear Register (WGINTRCLR) 419
4.14. Waveform Generator AHB Bus Error Register (WGAHBERR) 421
CHAPTER 20: Sound Mixer 422
1. Overview 423
2. Configuration and Block Diagram 424
3. Operation of the Sound Mixer 425
3.1. Basic Mixing Operation Procedure 425
3.2. Volume Effects 427
3.3. Sound Source Input and Internal Mixing Process Mode 428
3.4. Interrupt 429
3.5. Data Request Control 430
3.6. Data Output Control 430
3.7. AHB Slave Interface 431
3.8. AHB Master Interface 432
3.9. Input Buffer (FIFO) 432
3.10. Sound Source Sampling Rate 433
4. Registers 434
4.1. Mixer Channel Register (MXCH) 434
4.2. Mixer Output Control Register (MXOCTRL) 436
4.3. Mixer Date Request Control Register (MXDRQCTRL) 437
4.4. Mixer Input Control Register (MXICTRL) 439
4.5. Mixer Channel Monaural Register (MXCHMONO) 441
4.6. Mixer Channel Volume1 Register (MXCHVOL1) 443
4.7. Mixer Channel Volume2 Register (MXCHVOL2) 444
4.8. Mixer Channel Volume3 Register (MXCHVOL3) 445
4.9. Mixer Channel Mute Register (MXCHMUTE) 446
4.10. Mixer Channel Fade_in/out1 Register (MXCHFADE1) 448
4.11. Mixer Channel Fade_in/out2 Register (MXCHFADE2) 449
4.12. Mixer Channel Fade_in/out3 Register (MXCHFADE3) 450
4.13. Mixer Channel Fade_in/out4 Register (MXCHFADE4) 452
4.14. Mixer Channel Fade_in/out5 Register (MXCHFADE5) 453

4.15. Mixer Mixed Fade_in/out Register (MXMXDFADE) 454
4.16. Mixer Channel Fade_in/out Enable Register (MXCHFADEEN) 455
4.17. Mixer Buffer Clear Register (MXBUFFCLR) 457
4.18. Mixer FADE_in/out Clear Register (MXFADECLR) 459
4.19. Mixer Interrupt Enable Register (MXINTREN) 461
4.20. Mixer Interrupt Status Register (MXINTRSTATE) 463
4.21. Mixer Interrupt Clear Register (MXINTRCLR) 466
4.22. Mixer Input Buffer Count1 Register (MXINBUFFCNT1) 468
4.23. Mixer Input Buffer Count2 Register (MXINBUFFCNT2) 471
4.24. Mixer Channel Buffer Count Register (MXCHBUFFCNT) 473
4.25. Mixer Output Buffer Count Register (MXOUTBUFFCNT) 475
4.26. Mixer AHB Bus Error Register (MXAHBERR) 476
4.27. Mixer WFGn Data Address Register (MXWFGnDADR, n=0 to 4) 478
4.28. Mixer PCM/I2S n Data Address Register0-15 (MXPMISnDADR0-15, n=0 to 4) 479
5. Appendix 480
5.1. Gain for Volume Control 480
5.2. Fade In/Fade Out Time 487
5.3. Digital Filter 488
CHAPTER 21: Ethernet MAC 491
1. Overview 492
2. Configuration and Block Diagram 492
3. Operation of the Ethernet MAC 494
3.1. Direct Memory Access Interface 494
3.2. MAC Transmitter Block 506
3.3. MAC Receiver Block 507
3.4. Checksum Offload for IP, TCP, and UDP 508
3.5. MAC Filtering Block 509
3.6. IEEE Std 1588 and IEEE Std 802.1AS Support 512
3.7. MAC IEEE Std 802.3 Pause Frame Support 517
3.8. MAC PFC Based Pause Frame Support 518
3.9. Energy Efficient Ethernet Support 520
3.10. LPI Operation in Ethernet MAC 520
3.11. IEEE Std 802.1Qav Support – Credit Based Shaping 521
3.12. PHY Interface 521
3.13. Jumbo Frames 521
4. Registers 522
4.1. Network Control Register (ETHERNETn_network_control) 528
4.2. Network Configuration Register (ETHERNETn_network_configuration) 532
4.3. Network StatusNetwork Status Register (ETHERNETn_network_status) 537
4.4. DMA Configuration Register (ETHERNETn_dma_config) 538
4.5. Transmit Status Register (ETHERNETn_transmit_status) 542
4.6. RX Buffer Queue Base Address Register (ETHERNETn_receive_q_ptr) 544
4.7. TX Buffer Queue Base Address Register (ETHERNETn_transmit_q_ptr) 545
4.8. Receive Status Register (ETHERNETn_receive_status) 546
4.9. Interrupt Status Register (ETHERNETn_int_status) 547
4.10. Interrupt Enable Register (ETHERNETn_int_enable) 551
4.11. Interrupt Disable Register (ETHERNETn_int_disable) 554
4.12. Interrupt Mask Register (ETHERNETn_int_mask) 557
4.13. PHY Maintenance Register (ETHERNETn_phy_management) 563
4.14. Receive Pause Quantum Register (ETHERNETn_pause_time) 565
4.15. Transmit Pause Quantum Register (ETHERNETn_tx_pause_quantum) 566
4.16. TX Partial Store and Forward Register (ETHERNETn_pbuf_txcutthru) 567

4.17. RX Partial Store and Forward Register (ETHERNETn_pbuf_rxcutthru) 568
4.18. Jumbo-Frame Maximum Length Register (ETHERNETn_jumbo_max_length) 569
4.19. AXI Maximum Pipeline Register (ETHERNETn_axi_max_pipeline) 570
4.20. Hash Bottom Register (ETHERNETn_hash_bottom) 572
4.21. Hash Top Register (ETHERNETn_hash_top) 573
4.22. Specific Address Bottom i Register (ETHERNETn_spec_add_bottom_i) (i=1 to 4) 574
4.23. Specific Address Top i Register (ETHERNETn_spec_add_top_i) (i=1 to 4) 575
4.24. Type ID Match i Register (ETHERNETn_spec_type_i) (i= 1 to 4) 576
4.25. IPG Stretch Register (ETHERNETn_stretch_ratio) 577
4.26. Stacked VLAN Register (ETHERNETn_stacked_vlan) 578
4.27. Transmit PFC Pause Register (ETHERNETn_tx_pfc_pause) 579
4.28. Specific Address Mask 1 Bottom Register (ETHERNETn_mask_add1_bottom) 580
4.29. Specific Address Mask 1 Top Register (ETHERNETn_mask_add1_top) 581
4.30. RX PTP Unicast IP Destination Address Register (ETHERNETn_rx_ptp_unicast) 582
4.31. TX PTP Unicast IP Destination Address Register (ETHERNETn_tx_ptp_unicast) 583
4.32. Receive DMA Data Buffer Address Mask Register (ETHERNETn_dma_addr_or_mask) 584
4.33. IEEE 1588 Timer Comparison Value Nanoseconds Register (ETHERNETn_tsu_nsec_cmp) 585
4.34. IEEE 1588 Timer Comparison Value Seconds Bottom Register (ETHERNETn_tsu_sec_cmp) 586
4.35. IEEE 1588 Timer Comparison Value Seconds Top Register (ETHERNETn_tsu_msb_sec_cmp) 587
4.36. PTP Event Frame Transmitted Seconds [47:32] Register (ETHERNETn_tsu_ptp_tx_msb_sec) 588
4.37. PTP Event Frame Received Seconds [47:32] Register (ETHERNETn_tsu_ptp_rx_msb_sec) 589
4.38. PTP Peer Event Frame Transmitted Seconds [47:32] Register (ETHERNETn_tsu_peer_tx_msb_sec) 590
4.39. PTP Peer Event Frame Received Seconds [47:32] Register (ETHERNETn_tsu_peer_rx_msb_sec) 591
4.40. Identification and Revision Register (ETHERNETn_revision_reg) 592
4.41. Octets Transmitted Bottom Register (ETHERNETn_octets_txed_bottom) 593
4.42. Octets Transmitted Top Register (ETHERNETn_octets_txed_top) 594
4.43. Frames Transmitted Register (ETHERNETn_frames_txed_ok) 595
4.44. Broadcast Frames Transmitted Register (ETHERNETn_broadcast_txed) 596
4.45. Multicast Frames Transmitted Register (ETHERNETn_multicast_txed) 597
4.46. Pause Frames Transmitted Register (ETHERNETn_pause_frames_txed) 598
4.47. 64 Byte Frames Transmitted Register (ETHERNETn_frames_txed_64) 599
4.48. 65 To 127 Byte Frames Transmitted Register (ETHERNETn_frames_txed_65) 600
4.49. 128 To 255 Byte Frames Transmitted Register (ETHERNETn_frames_txed_128) 601
4.50. 256 To 511 Byte Frames Transmitted Register (ETHERNETn_frames_txed_256) 602
4.51. 512 To 1023 Byte Frames Transmitted Register (ETHERNETn_frames_txed_512) 603
4.52. 1024 To 1518 Byte Frames Transmitted Register (ETHERNETn_frames_txed_1024) 604
4.53. Greater Than 1518 Byte Frames Transmitted Register (ETHERNETn_frames_txed_1519) 605
4.54. Transmit Under Runs Register (ETHERNETn_tx_underruns) 607
4.55. Single Collision Frames Register (ETHERNETn_single_collisions) 608
4.56. Multiple Collision Frames Register (ETHERNETn_multiple_collisions) 609
4.57. Excessive Collisions Register (ETHERNETn_excessive_collisions) 610
4.58. Late Collisions Register (ETHERNETn_late_collisions) 611

4.59. Deferred Transmission Frames Register (ETHERNETn_deferred_frames) 612
4.60. Carrier Sense Errors Register (ETHERNETn_crs_errors) 613
4.61. Octets Receive Bottom Register (ETHERNETn_octets_rxed_bottom) 614
4.62. Octets Receive TOP Register (ETHERNETn_octets_rxed_top) 615
4.63. Frames Received Register (ETHERNETn_frames_rxed_ok) 616
4.64. Broadcast Frames Received Register (ETHERNETn_broadcast_rxed) 617
4.65. Multicast Frames Received Register (ETHERNETn_multicast_rxed) 618
4.66. Pause Frames Received Register (ETHERNETn_pause_frames_rxed) 619
4.67. 64 Byte Frames Received Register (ETHERNETn_frames_rxed_64) 620
4.68. 65 To 127 Byte Frames Received Register (ETHERNETn_frames_rxed_65) 621
4.69. 128 To 255 Byte Frames Received Register (ETHERNETn_frames_rxed_128) 622
4.70. 256 To 511 Byte Frames Received Register (ETHERNETn_frames_rxed_256) 623
4.71. 512 To 1023 Byte Frames Received Register (ETHERNETn_frames_rxed_512) 624
4.72. 1024 To 1518 Byte Frames Received Register (ETHERNETn_frames_rxed_1024) 625
4.73. Greater than 1518 Byte Frames Received Register (ETHERNETn_frames_rxed_1519) 626
4.74. Undersized Frames Received Register (ETHERNETn_undersized_frames) 627
4.75. Oversize Frames Received Register (ETHERNETn_excessive_rx_length) 628
4.76. Jabbers Received Register (ETHERNETn_rx_jabbers) 629
4.77. Frame Check Sequence Errors Register (ETHERNETn_fcs_errors) 630
4.78. Length Field Frame Errors Register (ETHERNETn_rx_length_errors) 631
4.79. Receive Symbol_Errors Register (ETHERNETn_rx_symbol_errors) 632
4.80. Alignment Errors Register (ETHERNETn_alignment_errors) 633
4.81. Receive Resource Errors Register (ETHERNETn_rx_resource_errors) 634
4.82. Receive Over Runs Register (ETHERNETn_rx_overruns) 635
4.83. IP Header Checksum Errors Register (ETHERNETn_rx_ip_ck_errors) 636
4.84. TCP Checksum Errors Register (ETHERNETn_rx_tcp_ck_errors) 637
4.85. UDP Checksum Errors Register (ETHERNETn_rx_udp_ck_errors) 638
4.86. Receive DMA Flushed Packets Register (ETHERNETn_auto_flushed_pkts) 639
4.87. IEEE 1588 Timer Increment Sub Nanoseconds Register (ETHERNETn_tsu_timer_incr_sub_nsec) 640
4.88. IEEE 1588 Timer Seconds [47:32] Register (ETHERNETn_tsu_timer_msb_sec) 641
4.89. IEEE 1588 Timer Seconds [31:0] Register (ETHERNETn_tsu_timer_sec) 642
4.90. IEEE 1588 Timer Nanoseconds Register (ETHERNETn_tsu_timer_nsec) 643
4.91. IEEE 1588 Timer Adjust Register (ETHERNETn_tsu_timer_adjust) 644
4.92. IEEE 1588 Timer Increment Register (ETHERNETn_tsu_timer_incr) 645
4.93. PTP Event Frame Transmitted Seconds [31:0] Register (ETHERNETn_tsu_ptp_tx_sec) 646
4.94. PTP Event Frame Transmitted Nanoseconds Register (ETHERNETn_tsu_ptp_tx_nsec) 647
4.95. PTP Event Frame Received Seconds [31:0] Register (ETHERNETn_tsu_ptp_rx_sec) 648
4.96. PTP Event Frame Received Nanoseconds Register (ETHERNETn_tsu_ptp_rx_nsec) 649
4.97. PTP Peer Event Frame Transmitted Seconds [31:0] Register (ETHERNETn_tsu_peer_tx_sec) 650
4.98. PTP Peer Event Frame Transmitted Nanoseconds Register (ETHERNETn_tsu_peer_tx_nsec) 651
4.99. PTP Peer Event Frame Received Seconds [31:0] Register (ETHERNETn_tsu_peer_rx_sec) 652
4.100. PTP Peer Event Frame Received Nanoseconds Register (ETHERNETn_tsu_peer_rx_nsec) 653

4.101. Transmit Pause Quantum 1 Register (ETHERNETn_tx_pause_quantum1) 654
4.102. Transmit Pause Quantum 2 Register (ETHERNETn_tx_pause_quantum2) 655
4.103. Transmit Pause Quantum 3 Register (ETHERNETn_tx_pause_quantum3) 656
4.104. Receive LPI Transitions Register (ETHERNETn_rx_lpi) 657
4.105. Received LPI Time Register (ETHERNETn_rx_lpi_time) 658
4.106. Transmit LPI Transitions Register (ETHERNETn_tx_lpi) 659
4.107. Transmit LPI Time Register (ETHERNETn_tx_lpi_time) 660
4.108. Interrupt Status Queue i Register (ETHERNETn_int_status_qi) (i=1 to 3) 661
4.109. TX Buffer Queue 1 Base Address Register (ETHERNETn_transmit_q1_ptr) 663
4.110. TX Buffer Queue 2 Base Address Register (ETHERNETn_transmit_q2_ptr) 664
4.111. TX Buffer Queue 3 Base Address Register (ETHERNETn_transmit_q3_ptr) 665
4.112. RX Buffer Queue 1 Base Address Register (ETHERNETn_receive_q1_ptr) 666
4.113. RX Buffer Queue 2 Base Address Register (ETHERNETn_receive_q2_ptr) 667
4.114. RX Buffer Queue 3 Base Address Register (ETHERNETn_receive_q3_ptr) 668
4.115. RX Buffer Size Queue i Register (ETHERNETn_rxbuf_size_qi) (i=1 to 3) 669
4.116. CBS Control Register (ETHERNETn_cbs_control) 671
4.117. CBS IdleSlope Queue A Register (ETHERNETn_cbs_idleslope_q_a) 672
4.118. CBS IdleSlope Queue B Register (ETHERNETn_cbs_idleslope_q_b) 673
4.119. MSB Buffer Queue Base Address Register (ETHERNETn_msb_buff_q_base_addr_reg) 674
4.120. TX BD Control Register (ETHERNETn_tx_bd_control) 675
4.121. RX BD Control Register (ETHERNETn_rx_bd_control) 676
4.122. Screening Type 1 Register i (ETHERNETn_screening_type_1_register_i) (i=0 to 15) 677
4.123. Screening Type 2 Register i (ETHERNETn_screening_type_2_register_i) (i=0 to 15) 679
4.124. Interrupt Enable Queue i Register (ETHERNETn_int_enable_qi) (i=1 to 3) 681
4.125. Interrupt Disable Queue i Register (ETHERNETn_int_disable_qi) (i=1 to 3) 683
4.126. Interrupt Mask Queue i Register (ETHERNETn_int_mask_qi) (i=1 to 3) 685
4.127. EtherType 2 Register i (ETHERNETn_screening_type_2_ethertype_reg_i) (i=0 to 7) 687
4.128. Type2 Compare Word 0 i Register (ETHERNETn_type2_compare_i_word_0) (i=0 to 31) 688
4.129. Type2 Compare Word 1 i Register (ETHERNETn_type2_compare_i_word_1) (i=0 to 31) 689

CHAPTER 22: MEDIA Local Bus Interface (MediaLB) 692

1. Overview 693
2. Configuration and Block Diagram 694
3. Operation of the MediaLB 699
   3.1. Notes on Using MediaLB 699
   3.2. List of References 699
4. Registers 700
   4.1. MediaLBn Device Control Configuration Register (MLBn_DCCR) 705
   4.2. MediaLBn System Status Configuration Register (MLBn_SSCR) 708
   4.3. MediaLBn System Data Configuration Register (MLBn_SDCR) 711
   4.4. MediaLBn System Mask Configuration Register (MLBn_SMCR) 712
   4.5. MediaLBn Version Control Configuration Register (MLBn_VCCR) 714
   4.6. MediaLBn Synchronous Base Address Configuration Register (MLBn_SBCR) 716
   4.7. MediaLBn Asynchronous Base Address Configuration Register (MLBn_ABCR) 717
   4.8. MediaLBn Control Base Address Configuration Register (MLBn_CBCR) 718
   4.9. MediaLBn Isochronous Base Address Configuration Register (MLBn_IBCR) 719
   4.10. MediaLBn Channel Interrupt Configuration Register (MLBn_CICR) 720
   4.11. MediaLBn AHB Master Control Register (MLBn_AHBMCTL) 721
   4.12. MediaLBn Channel Entry Configuration Register (MLBn_CECR0 - MLBn_CECR15) 722
   4.13. MediaLBn Channel Status Configuration Register (MLBn_CSCR0 - MLBn_CSCR15) 726

4.14. MediaLBn Channel Current Buffer Configuration Register (MLBn_CCBCR0 - MLBn_CCBCR15) 732
4.15. MediaLBn Channel Next Buffer Configuration Register (MLBn_CNBCR0 - MLBn_CNBCR15) 734
4.16. MediaLBn Local Channel Buffer Configuration Register (MLBn_LCBCR0 - MLBn_LCBCR15) 736
4.17. MediaLBn Module Identification Register (MLBn_MID) 738
CHAPTER 23: Stereo Audio DAC 739
1. Overview 740
2. Configuration and Block Diagram 740
3. Operation 741
3.1. Initialization 741
3.2. Using DMA Transfers to Update DADR Register 742
3.3. Output Data Values of the Analog DAC 742
3.4. Switching Clock Frequencies of the Analog DAC 743
3.5. Minimum Writing Interval Between the DAOSR and DACR Registers 743
3.6. Interrupt 743
3.7. AHB Slave Interface 744
3.8. FIFO 744
3.9. Frequency of CLKPI and CLKDA 744
3.10. Sampling Frequency and Oversampling Setting of DA Converter 745
3.11. Output Stop Control of Analog DAC 746
3.12. Bridge-Tied Load (BTL) 747
4. Registers 748
4.1. DAC Over Sampling Register(DAOSR) 748
4.2. DAC Configuration (DACR) 750
4.3. DAC BUSY Register(DABUSY) 751
4.4. DAC Control Register(DACTRL) 752
4.5. DAC FLUSH Register(DAFLUSH) 754
4.6. Interrupt Enable Register(INTREN) 755
4.7. Interrupt State Register (INTRSTAT) 757
4.8. Interrupt Clear Register (INTCTR) 759
4.9. DAC Polarity Configuration Register (DPCR) 761
4.10. Data 0..15 Register (DADR) 762
CHAPTER 24: Inter IC Sound (I2S) 763
1. Overview 764
2. Configuration and Block Diagram 765
3. Operations of the I2S 766
3.1. I2S Frame Construction 766
3.2. I2S Configuration and Operation Modes 768
3.3. Bit Alignment 772
3.4. FIFO Construction 774
3.5. Caution Summary 777
4. Registers 777
4.1. Reception FIFO Data Register (I2Sn_RXFDAT0 to 15) 780
4.2. Transmission FIFO Data Register (I2Sn_TXFDAT0 to 15) 782
4.3. Control Register (I2Sn_CNTREG) 783
4.4. Channel Control Register 0 (I2Sn_MCR0REG) 787
4.5. Channel Control Register 1 (I2Sn_MCR1REG) 790
4.6. Channel Control Register 2 (I2Sn_MCR2REG) 791
4.7. Operation Control Register (I2Sn_OPRREG) 792
4.8. Software Reset Register (I2Sn_SRST) 794

4.9. Interrupt Control Register (I2Sn_INTCNT) 795
4.10. Status Register (I2Sn_STATUS) 799
4.11. DMA Activate Register (I2Sn_DMAACT) 803
4.12. Debug Register (I2Sn_DEBUG) 805
4.13. Module ID Register (I2Sn_MIDREG) 806
CHAPTER 25: Programmable CRC 807
1. Overview 808
2. Configuration and Block Diagram 809
3. Operation of the Programmable CRC 809
3.1. CRC Operation Flowcharts 809
3.2. CRC Input Data and Checksum Calculation Flow 813
3.3. CRC Calculation Example 816
4. Registers 817
4.1. CRC Polynomial Register (CRCn_POLY) 818
4.2. CRC Seed Register (CRCn_SEED) 819
4.3. CRC Final XOR Register (CRCn_FXOR) 820
4.4. CRC Configuration Register (CRCn_CFG) 821
4.5. CRC Write Register (CRCn_WR) 825
4.6. CRC Read Register (CRCn_RD) 826
CHAPTER 26: PCMPWM 827
1. Overview 828
2. Configuration and Block Diagram 829
2.1. Block Diagram 829
2.2. Configuration of the PCMPWM Module 830
2.3. Output Configuration 833
3. Operation of the PCMPWM 834
3.1. Description of the PCM to PWM Conversion Process 834
3.2. PWM Cycle Time Configuration 839
3.3. PCM Data Sample Input 840
3.4. Interrupts 841
3.5. Dead Timer Operation 841
4. Registers 842
4.1. PCMPWM Control Register (PCMPWMi_CONTROL) 843
4.2. PCMPWM Output Control Register (PCMPWMi_OCTRL) 846
4.3. PCMPWM Clock Select Register (PCMPWMi_CLKSEL) 848
4.4. PCMPWM Count Period Register (PCMPWMi_COUNTP) 849
4.5. PCM Offset Register (PCMPWMi_PCMOFFS) 850
4.6. PCM Interrupt Enable Register (PCMPWMi_INTREN) 851
4.7. PCM Interrupt Status Register (PCMPWMi_INTRSTAT) 853
4.8. PCM Interrupt Clear Register (PCMPWMi_INTRCLR) 855
4.9. PCM Data Register 0..15 (PCMPWMi_DATA) 857
CHAPTER 27: HyperBus Interface 858
1. Overview 859
2. Block Diagram 859
3. Operation of the HYPERBUSI 860
3.1. HyperBus Core 860
3.2. Tx/Rx Controller 862
3.3. C/A Coder 863
3.4. FIFO 864
3.5. CS REG (Control Status Register) 865
3.6. AXI Slave Interface 866
4. Registers 867

4.1. Controller Status Register (HYPERBUSIn_CSR) 868
4.2. Interrupt Enable Register (HYPERBUSIn_IEN) 870
4.3. Interrupt Status Register (HYPERBUSIn_ISR) 871
4.4. Memory Base Address Register (HYPERBUSIn_MBR0 to 1) 872
4.5. Memory Configuration Register (HYPERBUSIn_MCR0 to 1) 873
4.6. Memory Timing Register (HYPERBUSIn_MTR0 to 1) 875
4.7. General Purpose Output Register (HYPERBUSIn_GPOR) 878
4.8. Write Protection Register (HYPERBUSIn_WPR) 879
4.9. Test Register (HYPERBUSIn_TEST) 880

CHAPTER 28: LCD Controller 881

1. Overview 882
2. Features 883
3. Configuration 884
4. Operation 885
   4.1. Operation of LCD Controller/Driver (LCDC) 886
   4.2. 1/2 Duty Output Waveform 888
   4.3. 1/3 Duty Output Waveform 891
   4.4. 1/4 Duty Output Waveform 894
   4.5. Static Drive Output Waveform 897
5. Setting 900
6. Registers 902
   6.1. LCD Control Register 0: LCR0 903
   6.2. Data Memory for Display: VRAM 905
   6.3. LCDC Control Register 1: LCR1 907
   6.4. Common Pin Switching Register: LCDCMR 908
   6.5. LCDC Static Control Register: LCRS 909
   6.6. Static LCD Display Data Register: LDR 911
   6.7. Key Code Register: LCD_KEYCDR 913
   6.8. Segment Output Register: SEGER 915
   6.9. Common Output V Pin Control Register: COMVER 916
7. Q&A 918
   7.1. How can I Set Pins to COM Output Pins or SEG Output Pins? 919
   7.2. How to Set VRAM? 921
   7.3. How can I Setting the Frame Cycle? 922
   7.4. How can I Set the Bias? 923
   7.5. How can I Set the Duty? 923
   7.6. How can I Control the LCD Operation Start/Stop? 923
   7.7. How can I Execute/Cancel the Display? 924
   7.8. How can I Display during the PSS Timer Mode (Main Oscillation Operation/ Sub Oscillation Operation)? 925
   7.9. How can I Select Either Internal or External for the Division Resistor? 926
   7.10. How can I Select Pin of V3 Voltage? 926
   7.11. How can I Select Either Internal or External for the Division Resistor? 927
   7.12. How can I Adjust the Brightness when the Internal Division Resistor is Used? 929
   7.13. How can I Block the Current with the External Division Resistor when the LCD Stops? 929
   7.14. How can I Display/Non-Display the LCD with Static Drive (ST0 to ST8)? 929
8. Sample Program 930
9. Notes 936

CHAPTER 29: Indicator PWM 937

1. Overview 938

2. Configuration and Block Diagram 938
3. Operation of Indicator PWM 939
3.1. Indicator PWM Timer 939
3.2. PWM Timer Count Operation Start 940
3.3. Timer Count Operation Stop 941
3.4. Counter Value Updating while PWM Counter Operation is in Progress 942
3.5. Indicator PWM Interrupts 943
3.6. Operation Flowchart 945
4. Registers 947
4.1. Timer Control Register (ITMCR) 948
4.2. Count Control Register (ICNTCR) 950
4.3. Status Control Register (ISTC) 952
4.4. Status Control Clear Register (ISTCC) 954
4.5. Status Control Set Register (ISTCS) 956
4.6. PWM Cycle Setting Register (IPCSR) 958
4.7. PWM Duty Setting Register (IPDUT) 959
5. Precautions for Using This Device 960
5.1. Notes to Observe when Accessing a Register 960
5.2. Indicator PWM Operation Precautions 960
CHAPTER 30: Memory Protection Unit for AXI 961
1. Overview 962
2. Configuration and Block Diagram 963
3. Operation of the MPU AXI 964
4. Registers 970
4.1. MPU AXI Control Register (MPUXn_CTRL0) 973
4.2. MPU AXI NMI Enable Register (MPUXn_NMIEN) 975
4.3. MPU AXI Write Error Control Register (MPUXn_WERRC) 976
4.4. MPU AXI Write Error Address Register (MPUXn_WERRA) 977
4.5. MPU AXI Read Error Control Register (MPUXn_RERRC) 978
4.6. MPU AXI Read Error Address Register (MPUXn_RERRA) 979
4.7. MPU AXI Region Control Registers (MPUXn_CTRL1 to 8) 980
4.8. MPU AXI Start Address Registers (MPUXn_SADDR1 to 8) 981
4.9. MPU AXI End Address Registers (MPUXn_EADDR1 to 8) 982
4.10. MPU AXI Unlock Register (MPUXn_UNLOCK) 983
4.11. MPU AXI Module ID Register (MPUXn_MID) 984
5. Notes on Using MPU AXI 985
CHAPTER 31: Memory Protection Unit for AHB 987
1. Overview 988
2. Configuration and Block Diagram 989
3. Operation of the MPU AHB 990
4. Registers 994
4.1. MPU AHB Control Register (MPUHn_CTRL0) 996
4.2. MPU AHB NMI Enable Register (MPUHn_NMIEN) 998
4.3. MPU AHB Memory Error Control Register (MPUHn_MERRC) 999
4.4. MPU AHB Memory Error Address Register (MPUHn_MERRA) 1000
4.5. MPU AHB Region Control Registers (MPUHn_CTRL1 to 8) 1001
4.6. MPU AHB Start Address Registers (MPUHn_SADDR1 to 8) 1002
4.7. MPU AHB End Address Registers (MPUHn_EADDR1 to 8) 1003
4.8. MPU AHB Unlock Register (MPUHn_UNLOCK) 1004
4.9. MPU AHB Module ID Register (MPUHn_MID) 1005
5. Notes on Using the MPU AHB 1006
CHAPTER 32: Sound System Configuration 1008

1. Overview 1009
1.1. Input Module 1009
1.2. Sound Mixer 1009
1.3. Output Module 1009
2. Configuration and Block Diagram 1010
3. Operation 1011
3.1. Start Operations 1011
3.2. Addition of Sound Source Inputs 1013
3.3. Stop Operations 1014
3.4. Operation of a Sound Source 1016
4. Configuration and limitation of each module 1017
4.1. Interface between SMIX and DAC 1017
4.2. Interface between SMIX and PCM-PWM 1017
4.3. Interface between SMIX and I2S 1018
4.4. Interface between SWFG and SMIX 1018
4.5. Interface between DMAC and SMIX 1019
4.6. Interface between CPU and SMIX 1019
5. Note 1019
CHAPTER 33: Graphic Subsystem 1020
1. Perface 1021
1.1. About This Document 1021
1.2. Reference Document 1021
2. Function 1021
2.1. Feature Summary 1021
2.2. Block Diagrams 1022
2.3. Functional Limitations 1023
2.4. Clause Name 1024
3. Application 1025
3.1. Map Tables 1025
3.2. Common Setup 1027
3.3. 2D with LCD Bus Interface 1031
3.4. 2D Core 1033
3.5. LCD Bus Interface 1033
3.6. SW Interface 1033
CHAPTER 34: Automotive Remote Handler 1083
1. Outline of ARH 1084
2. ARH Registers 1087
3. Operation of ARH 1172
4. Notes on Using ARH 1180
5. Use Cases 1182
CHAPTER 35: Base Timer 1187
1. Overview 1188
2. Registers 1188
CHAPTER 36: Base Timer Port Definition 1191
1. Overview 1192
2. Definition 1192
2.1. Unit and Channel number definition 1192
2.2. Port name definition and Supported operation mode 1192
3. Supplementation 1194
3.1. Input port configuration 1194
3.2. Output port configuration 1194
3.3. Channel Group configuration 1194

4. Operation and Registers 1194
CHAPTER 37: LCD Bus Interface 1195
1. Preface 1196
1.1. Purpose and Scope 1196
1.2. Document Structure 1196
2. Function 1197
2.1. Feature Summary 1197
2.2. Block Diagram 1197
2.3. Functional Limitations 1197
2.4. Nomenclature 1198
2.5. Use Cases 1199
2.6. Protocol Engine 1200
2.7. Iris Graphics Core Interface 1204
2.8. Command Sequencer 1207
3. Application 1213
3.1. Map Tables 1213
3.2. Getting Started 1213
3.3. Display Setup 1214
3.4. Control Flow 1217
3.5. Fifo Operation 1219
3.6. Error Recovery 1220
3.7. SW Interface 1221
CHAPTER 38: Appendix 1232
1. Pin Status in Each CPU State 1233
CHAPTER 39: Major Changes 1236
Revision History 1294
Document Revision History 1294

CHAPTER 1: Overview

This chapter explains the product overview.

1.Overview
2.Document Definition
3.Register Attribute
4.Vocabulary

CODE: OVERVIEW-S6J3300-E1

1. Overview

S6J3300 is a microcontroller series which is to be applied to automotive systems.

S6J3310/20/30/40 is a microcontroller series for instrument clusters with small thin-film transistor (TFT) displays.

S6J3350 is a microcontroller series for body and gateway.

2. Document Definition

The related documents of S6J3300 are as follows:

Table 1-1 Related Documents

Document Type	Definition	Primary User	Document Code
Datasheet	The function and its characteristics are specified quantitatively.	Investigator and Hardware Engineer	002-10635 (S6J3310/20/30/40) 002-10634 (S6J3350)
S6J3300 Hardware Manual	The function and its operation of S6J3300 series are described.	Software Engineer	002-10185
Traveo™ Platform Hardware Manual	The function and its operation of CPU core platform are described.	Software Engineer	002-07884
Application note	The reference software, sample application, the reference board design, and so on are explained.	Software and Hardware Engineer	002-03898 002-04455 002-04446 002-09716 002-04452 002-04096 002-12061 002-02495

Note:

- *Refer all documents for the system development.*
- *"Primary user" is a most likely an engineer for whom the document is the most useful.*
- *The description of the datasheet and the S6J3300 Hardware Manual should precede the duplicated description of Traveo™ Platform Hardware Manual.*
- *Traveo™ Platform Hardware Manual is expected to be used as dictionary of platform specification.*

3. Register Attribute

3.1. Read and Write

Refer the table with the following definition.

Table 1-2 Register Attribute of Read and Write

Attribute	Definition
R	It can be read.
R0	"0" is to be read.
R1	"1" is to be read.
RX, RN	The read value is undefined.
W	It can be written.
W0	"0" must be written.
W1	"1" must be written.
WX	Writing doesn't affect the operation.
"/" (slash)	The read value is same as the written value.
"," (comma)	The read value is different from the written value.

■ **Example**

R/W : The written value is to be read.

R, W0 : "0" must be written, and the read value is different from the written value.

Note:

- *The register attribute of a status register which has its clear register and set register is conveniently described to be R/W though the written value cannot be read directly. In this case R/W has a meaning that the controlled value is to be read.*
- *A register attribute does not necessarily describe a prohibited operation of the register. Any prohibited operations should be referred within the line of the register description as well. See every description of the register specification together with every note.*

3.2. Protection

Refer the table with the following definition.

Table 1-3 Register Attribute of Protection

Attribute	Definition
RP	It can be read at the supervisor mode.
WP, RN	It can be written at the supervisor mode.
WS	It can be written after cancelling the sequence protection.
WPS	It can be written after cancelling the sequence protection at the supervisor mode.

3.3. Register Information

You can also see register information in a table as below.

REGISTER_NAME	The register name and its abbreviation.
OFFSET	The offset address of the register
ACCESS_SIZE	B: The byte (8-bit) access is possible. H: The half-word (16-bit) access is possible when the lower bit of the address is "0". W: The word (32-bit) access is possible when the lower bit of the address is "00".
MULTIPLE	No definition
NUMERIC_TYPE	No definition
OTHER	No definition

Register attribute for header file generation.

BIT_OFFSET	The bit offset (Ex. 31,30,29,,,,0)
BIT_NAME	The bit name
ACCESS_TYPE	The allowed access defined 3.1
PROT_TYPE	The allowed access defined 3.2
INITIAL_VALUE	The initial value

4. Vocabulary

Acronym	Description
A/D converter	Analog digital converter
ADC	Analog digital converter
AHB	Advanced high performance bus
AMBATM	Advanced microcontroller bus architecture
APB	Advanced peripheral bus
ATCM	TCM-A port
AXI	Advanced extensible interface
B0TCM	TCM B0 port
B1TCM	TCM B1 port
BBU	Bit banding unit
BDR	Boot description record
BTL	Bridge-tied load
CAN	Control are network
CD	Clock domain
CPU	Central processing unit
CR	CR Oscillator
CRC	Cyclic redundancy check
CSV	Clock supervisor
DAC	Digital analog converter
DAP	Debug access port
DED	Dual error detection
DMA	Direct memory access
DMAC	DMA controller
EAM	Exclusive access memory
ECC	Error correction code
ETM	Embedded trace macro
EXT-IRC	External interrupt controller
FIQ	Fast interrupt request
FPU	Floating point unit
FRT	Free run timer
GPIO	General purpose I/O
HPM	High performance matrix
HW-WDT	Hardware watchdog timer
I/O	Input or output
I2S	Inter-IC sound
ICU	Input capture unit
IPCU	Inter-processor communication unit
IRC	Interrupt controller
IRQ	Interrupt request
ISR	Interrupt service routine
JTAG	Joint test action group
LLPP	Low latency peripheral port
LVD	Low voltage detector
MCU	Microcontroller unit
MFS	Multi-function serial interface
NF	Noise filter
NMI	Non maskable interrupt

Acronym	Description
OCU	Output compare unit
OSC	Oscillator
PCM	Pulse coded module
PLL	Phase locked loop
PONR	Power on reset
PPC	Port pin configuration
PSS	Power saving state
PWM	Pulse width modulation
RAM	Random access memory
RIC	Resource input configuration
ROM	Read only memory
RTC	Real time clock
RVD	Low voltage detection and reset for RAM retention
SCT	Source clock timer
SEC	Single error correction
SECDED	Single error correction and dual error detection
SHE	Secure Hardware Extension
SMC	Stepper motor controller
SMIX	Sound mixer
SRAM	Static RAM
SWFG	Sound waveform generator
SW-WDT	Software watchdog timer
SYSC	System controller
TCFLASH	FLASH connected to TCM
TCM	Tightly coupled memory
TCRAM	RAM connected to TCM
TPU	Timing protection unit
UDC	Up-down counter
VIC	Vectored interrupt controller
VRAM	Video RAM
WDR	Watchdog description record
WDT	Watchdog timer
WFG	Waveform generator
WorkFLASH	Work FLASH memory

CHAPTER 2: Function List

This chapter explains the functions.

1. Function List
2. Optional Function

CODE: FUNCTIONLIST-S6J3300-E1

1. Function List

The table shows the functions which are implemented in S6J3300 series.

Table 2-1 Function List

Function	Description	Remark
CPU core	Arm® Cortex® R5F	
FPU	Available	
PPU	Available	
MPU	Available	
TPU	Available	
Endian	Little endian	
Core clock frequency	240 MHz	See the AC specification on the datasheet.
HPM bus frequency	200 MHz	See the AC specification on the datasheet.
LLPM bus frequency	240 MHz	See the AC specification on the datasheet.
Resource clock frequency	80 MHz (Max)	
Embedded CR oscillation	Slow clock:100 kHz, Fast clock: 4 MHz (Center frequency)	See the AC specification on the datasheet.
PLL	PLL0, 1, 2, 3	
SSCG PLL	SSCG0, 1, 2, 3	
Clock supervisor	Available	
DMA	16 ch	
Boot-ROM	16 Kbyte	
JTAG	Available	
Data cache	16 Kbyte	
Instruction cache	16 Kbyte	
Program FLASH	Option	See 2.1
Work FLASH	112 Kbyte	
TCRAM	128 Kbyte	
System SRAM	384 Kbyte	
Backup RAM	32 Kbyte	
Security (SHE)	Option	
Low latency interrupt	Available	
Power domain	5 domains	
External power supply	5 V(VCC5, VCC53), 3 V(VCC3, VCC53), 1. 2 V(VCC12)	See the DC specification on the datasheet.
Embedded LDO power supply for 5.0 V	Available	
Low voltage detection of external power supply	Available	
Low voltage detection of internal LDO output	Available	
Hardware watchdog timer	Available	
Software watchdog timer	Available	
Package	Option	See 2.1
AUTOSAR	AUTOSAR 4.0.3	
General Purpose I/O	Option	See 2.3
Up/down counter	2 ch	
I/O timer	(FRT 5 ch x ICU 6 ch x OCU 6 ch) + (FRT 3 ch x ICU 6 ch x OCU 6 ch)	
32-bit Reload timer	6 ch	
Real time clock	Available	Automatic calibration
Sound generator	5 ch	
Sound waveform generator	Option 1 unit x 5 outputs	See 2.1
Sound mixer	Option 1 unit x 10 inputs	See 2.1

Function	Description	Remark
Stereo audio DAC	Option 1 unit (L and R)	See 2.1
PCM-PWM	Option 1 unit (L and R)	See 2.1
Base timer	Option 32 units (64 ch)	See 2.3
Stepping motor controller (SMC)	For 6 gauges	
12-bit-A/D converter	Option 2 unit - 64 input ports (Max)	See 2.3
CRC	4 unit	
Programmable CRC	1 unit	
Source clock timer	4 ch	
NMI	Available	
External interrupt	Option 24 ch	See 2.3 See the AC specification on the datasheet.
Internal interrupt	512 vectors	
I2S	2 ch	One only supports an output as a function of the sound system.
DDR HSSPI	1 ch	A type of Quad SPI
Hyper BUS	1 ch	See the AC specification on the datasheet.
Multi-function serial interface	12 ch	See the AC specification on the datasheet.
CAN-FD	Option 8 ch	See 2.3
CAN-FD RAM (ECC supported)	16 KB/ch It equivalents to 128 message buffer per channel of MCAN module	
Ethernet AVB	Option	See 2.1
Media-LB (MOST50)	Option	See 2.1
LCD controller	Option 4 COM x 32 SEG (Max)	See 2.3
Indicator PWM	1 ch	
MPU for AHB	1 unit	
MPU for AXI	1 unit	
Graphic engine clock	80 MHz (Max)	
Graphic AXI clock	80 MHz (Max)	
Display clock	25 MHz	
Display clock source	Graphic display controller clock or external clock	
Target resolution	WQVGA 480 x 272	
Target frame rate	60 fps	
Number of display outputs	1 output	
TTL output (RGB888)	Option	See 2.1
2D Graphic engine	1 unit	
2D Driver API	CYPRESS proprietary	
External BUS	Option 1 ch	See 2.3
ARH (Automotive Remote Handler)	Option (2 ch)	See 2.1

Note:

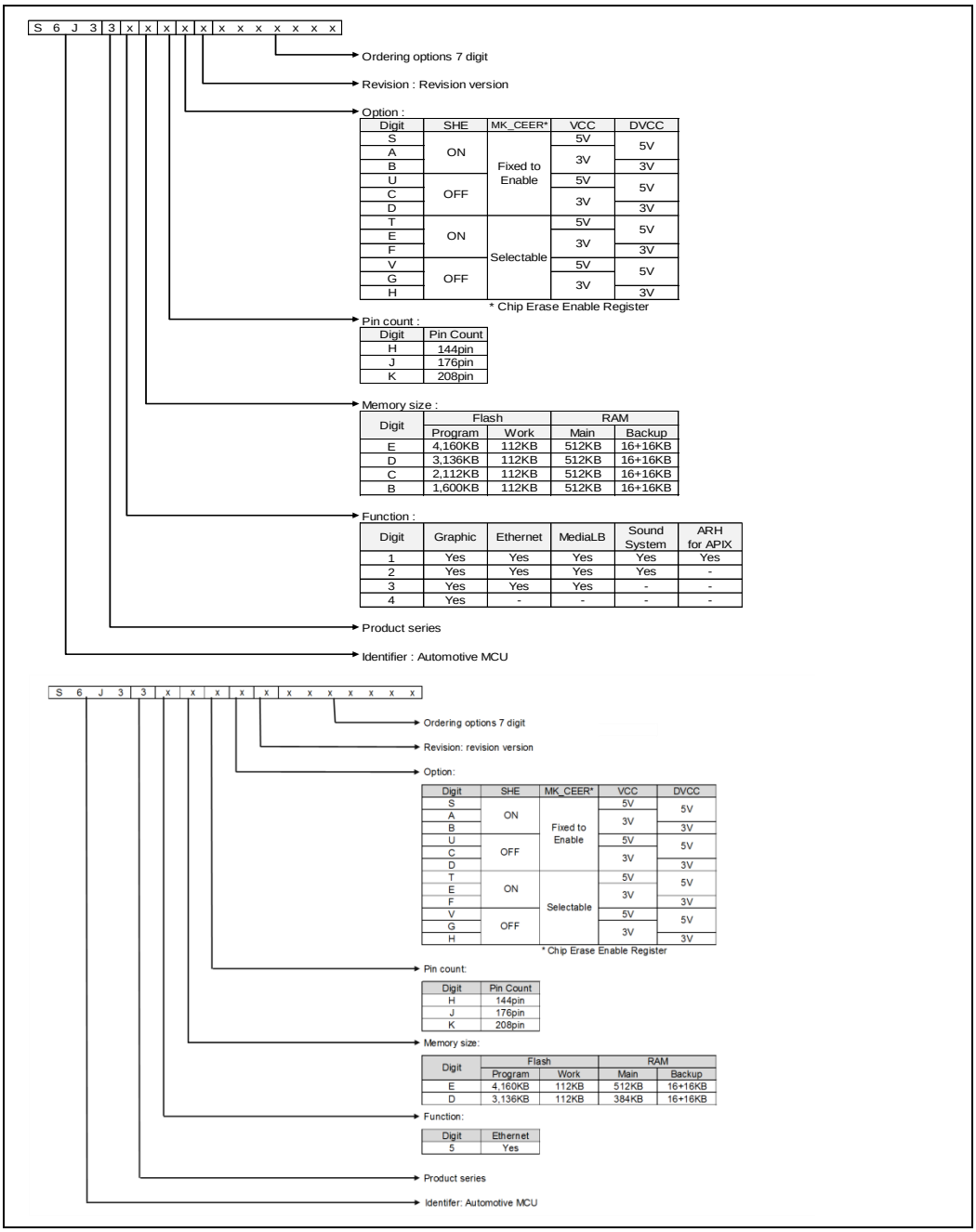
- *The options are described in 2 Optional Function.*
- *The described specifications in the table which are related the electric characteristics only show the typical values. They don't necessarily include the width of characteristics, errors, and so on. They should be seen in the datasheet in details.*

2. Optional Function

2.1. Basic Option

The figure shows the optional function and the part number relations of the series.

Figure 2-1 Optional Function and Part Number Relations



Note:

- *This table only shows the relations between the optional function and the part numbers. That is, all products are not necessarily available for orders. See the order number on the datasheet, and confirm actual availabilities of products.*
- *The sound system is composed of the sound waveform generator, the sound mixer, the audio DAC, PCM-PWM, and I2S0.*
- *In case of function digit 5. the main RAM size is configured as follows according to memory digit.*
  E : TCRAM 128 KB + System-RAM 384 KB
  D : TCRAM 128 KB + System-RAM 256 KB

2.2. ID

See the Chip ID specification on the Datasheet in detail.

2.3. Restriction

Some functions have restrictions which depend on package pin counts.

Table 2-2 Function Restriction of Package Pin Count

Function	TEQFP176	TEQFP144
Analog input port (12-bit-ADC)	AN0 to 3, AN19 to 20, AN22 to 23, AN27, AN33 to 38, AN48	AN0 to 7, AN10 to 11, AN14 to 15, AN19 to 20, AN22 to 23, AN25 to 30, AN33 to 38, AN48
SEG port of LCD controller	-	SEG0 to 3 SEG5 to 8
General Purpose I/O	P4_00 to to P4_31	P4_00 to P4_31 P3_00 to P3_31
CAN	RX0_2, TX0_2 RX3_2, TX3_2 RX4_0, TX4_0 RX7_0, TX7_0 RX7_1, TX7_1	RX0_1, TX0_1 RX0_2, TX0_2 RX1_1, TX1_1 RX2_1, TX2_1 RX3_1, TX3_1 RX3_2, TX3_2 RX4_0, TX4_0 RX5_1, TX5_1 RX6_1, TX6_1 RX7_0, TX7_0 RX7_1, TX7_1
BaseTimer	PPG16/17/18/19_TOUT0_0 PPG16/17/18/19_TOUT2_0 PPG20/21/22/23_TOUT0_0 PPG20/21/22/23_TOUT2_0 PPG24/25/26/27_TOUT0_0 PPG24/25/26/27_TOUT2_0 PPG28/29/30/31_TOUT0_0 PPG28/29/30/31_TOUT2_0 PPG16/17/18/19/20/21_TIN1_0 PPG22/23/24/25/26/27_TIN1_0 PPG28/29/30/31_TIN1_0	PPG16/17/18/19_TOUT0_0 PPG16/17/18/19_TOUT2_0 PPG20/21/22/23_TOUT0_0 PPG20/21/22/23_TOUT2_0 PPG24/25/26/27_TOUT0_0 PPG24/25/26/27_TOUT2_0 PPG28/29/30/31_TOUT0_0 PPG28/29/30/31_TOUT2_0 PPG16/17/18/19/20/21_TIN1_0 PPG22/23/24/25/26/27_TIN1_0 PPG28/29/30/31_TIN1_0 PPG4/5/6/7/8/9_TOUT0_1 PPG4/5/6/7/8/9_TOUT2_1 PPG10/11/12/13/15_TOUT0_1 PPG10/11/12/13/14/15_TOUT2_1 PPG0/1/2/3/4/5_TIN1_1 PPG6/7/8/9/10/11_TIN1_1 PPG12/13/14/15_TIN1_1

Function	TEQFP176	TEQFP144
ExtBus	-	MDQM1 MAD15 to 21 MDATA8 to 15
External Interrupt	EINT1_4, EINT1_5 EINT2_1, EINT2_2 EINT3_2, EINT4_2 EINT5_4, EINT5_5 EINT6_4, EINT7_1 EINT7_4, EINT8_4 EINT8_5, EINT9_1 EINT10_1, EINT10_4 EINT10_5, EINT13_2 EINT13_3, EINT14_2 EINT14_3, EINT15_3 EINT16_1, EINT16_3 EINT16_4, EINT19_4 EINT20_3, EINT21_3 EINT22_1, EINT22_3 EINT23_3, EINT23_4	EINT0_4, EINT1_1 EINT1_4, EINT1_5 EINT2_1, EINT2_2 EINT2_4, EINT3_1 EINT3_2, EINT3_4 EINT4_2, EINT4_4 EINT5_4, EINT5_5 EINT6_1, EINT6_4 EINT7_1, EINT7_4 EINT8_1, EINT8_4 EINT8_5, EINT9_1 EINT9_2, EINT10_1 EINT10_2, EINT10_4 EINT10_5, EINT11_2 EINT11_5, EINT12_1 EINT12_2, EINT12_5 EINT13_2, EINT13_3 EINT13_5, EINT14_1 EINT14_2, EINT14_3 EINT14_5, EINT15_2 EINT15_3, EINT16_1 EINT16_2, EINT16_3 EINT16_4, EINT16_5 EINT17_1, EINT17_3 EINT17_5, EINT18_1 EINT18_3, EINT18_5 EINT19_1, EINT19_3 EINT19_4, EINT20_1 EINT20_2, EINT20_3 EINT21_1, EINT21_3 EINT22_1, EINT22_3 EINT23_3, EINT23_4

Note:

- *See multiplexed functions on pin assignment sheet.*
- *The optional restriction will be added without notification.*

CHAPTER 3: Product Description

This chapter explains the function feature.

1. Overview
2. Product Description
3. Note

1. Overview

This chapter explains the product features of S6J3300 series. The description of this chapter should precede the duplicated description on Traveo™ Platform Hardware Manual.

2. Product Description

The table shows features.

Table 3-1 Features Description

Feature	Description
Technology	40-nm CMOS technology with embedded FLASH Fully automotive qualified according to ISO/TS 16949 and AEC-Q100 Developed according to ISO26262, safety target ASIL-B
Functional Safety	The product series has some functional safety features suited for ASIL-B application.
Peripherals	See function list.
Power Domain (PD)	See the Traveo™ Platform Hardware Manual and chapter STATE TRANSITION in detail. The product series supports the power off control of PD1, PD2 (including PD3 and 5), PD4_0, PD4_1 and PD6. The power domain resets of PD3 and PD5 included in PD2 are not supported in the product series, and "0" is always read from the reset factor flags of them. This series doesn't support partial wakeup for PD6.
Debug and Trace	See the Traveo™ Platform Hardware Manual in detail. − Standard 5-pin JTAG interface − 4 kB Embedded Trace Buffer 4-bit trace support for TEQFP package.
System Control	See the Traveo™ Platform Hardware Manual in detail. Main and sub oscillator is available. − A wide range of 3.6 - 16 MHz is available for main oscillator − 32 KHz is available for sub oscillator Sub clock is enable/disable by register settings
Clock	See the Traveo™ Platform Hardware Manual in detail. CLK_CLKO (Clock Output Function) is supported. Main Oscillation Stabilization Wait Time (at 4 MHz):8.19 ms (Initial value)
Embedded CR oscillation	See the Traveo™ Platform Hardware Manual in detail. Stabilization time is as followings. − 0.35 ms to 0.8 ms for 4 MHz (Fast clock) − 0.43 ms to 1.28 ms for 100 kHz (Slow clock)
Clock Supervisor	See the Traveo™ Platform Hardware Manual in detail. This product series doesn’t support clock supervisor output port. (Related register and internal circuit is implemented.)
Reset	RSTX pin + MD pin simultaneous assert INITX (Same as INITX pin input) − Occurrence factor: Simultaneously inputting “L” level to RSTX pin and inputting “L” level to MD pin − Release factor: Inputting “H” level to RSTX pin See the Traveo™ Platform Hardware Manual in detail. Following resets are not mounted on this device. − SRSTX (and nSRST pin) The product series does not support EX5VRST and writing EX5VRSTCNT bits in SYSC0_SPECFGR has no effect.
Hardware watchdog	See the Traveo™ Platform Hardware Manual in detail. Hardware watchdog function stops during PSS mode. In the related register of HWDG_CFG, the bit ALLOWSTOPCLK is always read as 1 (HWDG_CFG.ALLOWSTOPCLK=1). The product series doesn’t support Watchdog Counter Monitor Output port. (Related register and internal circuit is implemented.)
Standby mode	See the Traveo™ Platform Hardware Manual in detail. Standby mode with 5 V (or 3 V) single external power supply is available. Turning off the 1.2 V external power supply in standby mode is available. The long term pulse of the indicator PWM can be outputted during RTC Standby mode.

Feature	Description
PLL / SSCG PLL	See the Traveo™ Platform Hardware Manual in detail. Use case assumption is following. PLL − Sound system clock − Sound frequency master clock − Peripherals − Display clock − Trace clock SSCG − CPU core − GDC core − Hyper BUS − DDR-HSSPI Product supports down spread and center spread modes with the conditions defined in chapter "Internal Clock Timing" on the datasheet.
External Interrupts	See the Traveo™ Platform Hardware Manual in detail.
NMI	See the Traveo™ Platform Hardware Manual in detail. 1 NMI pin.
Memory Protection	MPU16 AHB: See the Traveo™ Platform Hardware Manual in detail. MPU for AXI: ch.0 MPU for AHB: ch.1 Additional MPU for Graphic sub system, MediaLB and Ethernet AVB. They are described on the chapter of MPU for AHB and MPU for AXI To configure Lock or Unlock for both MPUXn_UNLOCK and MPUHn_UNLOCK, − Lock: 0x112ABB56 − Unlock: 0xACCABB56
Peripheral Protection	See the Traveo™ Platform Hardware Manual in detail. Protected peripherals are described in the base address map.
Internal Memories System SRAM	384 kByte 1 wait cycle is necessary for RAM read at over 120 MHz.
Internal Memories TCRAM	128 kByte
Internal Memories Backup RAM	32 kByte (Backup-RAM0 : 16 KB, Backup-RAM1 : 16 KB) Backup RAM can only be operated in RUN mode (normal operation mode). In other mode the memory content should be retained, but it cannot be operated. SLEEP control for Backup RAM is not supported and cannot be used.
Embedded Program/Work Flash Memory	Embedded Program Flash can be accessed with 0-wait-cycle if CPU frequency is 80 MHz or less. 0-wait-cycle: 80 MHz or less. 1-wait-cycle: 160 MHz or less. 2-wait-cycle: more than 160 MHz. Work Flash can be accessed with 0-wait-cycle if CPU frequency is 12.5 MHz or less. 6-wait-cycle: 80 MHz or less. 12-wait-cycle: 160 MHz or less. The wait-cycle setting see the Traveo™ Platform Hardware Manual in details. The maximum frequency should be referred in datasheet. Erase suspend is supported. Reading and writing to the other sector are possible when Flash Erase is suspended. Serial Flash programing and Parallel Flash programing are supported. Margin mode is not supported.
Internal Power Domain	PD1: Always ON PD2: Cortex R5F platform/ GDC/ additional peripherals PD4: Backup RAM in Always On domain PD6: Peripherals in Always On domain * The chapter of the block diagram explains in detail.

Feature	Description
Power Supply	5 V, and 3 V, 1.2 V external power supply is required. Built in LDO provides internal power supply for Always On region (PD1). 1.2 V external power supply control pin is supported. 3 V external power supply should be controlled by GPIO. There are constraints of power on/off sequence.
Low Voltage Detection	LVD for external voltage is supported. LVD for internal voltage is supported. See the specification of the detected level on the datasheet.
Low voltage detection for RAM retention (RVD)	RVD for RAM retention is effective during the standby mode only. That is, it is only for the Backup RAM of 32 KB that the function is available.
Resource inter-connect	The output signal of some resources can be inputted to the other resource.
I/O Ports	5 V general purpose I/O 3 V general purpose I/O Multi input level and multi output drivability Pull-up, pull-down function is available. Resource input and output is multiplexed. +B input is allowed many pins of 3.3 V, 5 V and 3.3 V/5 V I/O domain.
A/D Converter	12-bit resolution, 2 unit(Unit0 is possible to select channels 0-31. Unit1 is possible to select channels 32-63.) 64 channels of analog input for TEQFP208 48 channels of analog input for TEQFP176 35 channel of analog input for TEQFP144 24 channels of them are shared with the SMC for TEQFP208/176/144 External trigger and timer trigger are available. The description of the A/D converter function should be referred in the S6J3300 Hardware Manual. Though the chapter of I/O port in Traveo™ Platform Hardware Manual describes another A/D converter function, do not refer it. A/D Channel Control Register (ADC12Bn_CHCTRL0) [bit5:0] ANIN[5:0] : Analog Input Selection bits. This register setting is possible of channel 0-31 (the register value is 00_0000 to 01_1111). AN39 to AN63 are not support for S6J33xxxAx, S6J33xxxCx, S6J33xxxEx, and S6J33xxxGx option.
CRC	See the Traveo™ Platform Hardware Manual in detail.
Programmable CRC	DMA support
Sound Generator	Produces sound/melody with varying frequency and amplitude for convenient duration Square wave sound output Automatic linear amplitude increment or decrement Interrupt request generated when specified sound length has ended
Sound Waveform generator	Sine waveform, saw-tooth waveform and Square waveform are generated with easy configuration of the parameters which specified sound sources. Fade-in and Fade-out control for reverberation.
Sound Mixer	The input channels of 0 - 4 are reserved for waveform generator. Mixing different sampling frequency sounds. Mixing Internal sounds and External I2S input sounds. Saturating addition function for keeping sound quality. Cut a specific frequency data by digital filter. LPF is support by FIR filter. Fade-in and Fade-out control.
PCM-PWM	Conversion of PCM audio streaming to Pulse Width Modulated signals. Supports 2 output channels for stereo and mono data Up to 16-bit output sample resolution Support for half and full H-bridges
Audio DAC	The sound source of the fixed 48 kHz sampling frequency can be outputted. 1 unit, L/R channels support. BTL connection is available.

Feature	Description
I2S	2 ch. − I2S0 can output sound sources which are processed by Sound System. − I2S1 can input sound sources which are processed by Sound System. See the "Sound System Configuration" of S6J3300 Hardware Manual in detail.
Base Timer	See the Traveo™ Platform Hardware Manual in detail. A unit consists of a pair of 16-bit base timers. 32 units, that is, 64 channels of base timers are available.
Reload Timer	See the Traveo™ Platform Hardware Manual in detail.
I/O Timer	See the Traveo™ Platform Hardware Manual in detail.
Up/Down Counter	See the Traveo™ Platform Hardware Manual in detail.
Multi-Functional Serial (MFS)	See the Traveo™ Platform Hardware Manual in detail. Only 2 ports of MFS have the dedicated I/O for I2C. See the datasheet in detail. The I2C is not designed to be hot swappable. CTS/RTS is not mounted (hardware flow control is not supported for this series.)
CAN-FD	Flexible data rate is supported. 16 KB/ch of message RAM is available. The clock output from CAN pre-scaler is supplied to every CAN. ECC error generation function of the message RAM is not supported for this device. Therefore CAN FD ECC Error Insertion Control Register (FDFECR) is not writeable.
Real Time Clock (RTC) with auto-calibration	See the Traveo™ Platform Hardware Manual in detail.
DDR High Speed SPI	ch.0: HSSPI as a MCU peripheral
Hyper BUS I/F	ch.0: Hyper Bus as a MCU peripheral The following register is not supported and cannot be used. − Controller Status Register (HYPERBUSIn_CSR) − Interrupt Enable Register (HYPERBUSIn_IEN) − Interrupt Status Register (HYPERBUSIn_ISR) − Write Protection Register (HYPERBUSIn_WPR) − Test Register (HYPERBUSIn_TEST) GPO signal can only be used for "Internal Control example by GPO" in this product, that is, it can select using HyperBus of PF or using HyperBus of Graphic Sub System.
Stepper Motor Control (SMC)	Each channel has 6 motor drivers with high output capability
External Interrupt Capture Unit (EICU)	See the Traveo™ Platform Hardware Manual in detail.
Ethernet AVB	10/100 Mbps MII-Interface Supports Audio-Video Bridging (AVB)
MediaLB	MOST50 (1024FS) 3 wires Maximum 15 ch is available.
LCD Controller	TEQFP208 : 4 com x 32 seg TEQFP176 : 4 com x 32 seg TEQFP144 : 4 com x 24 seg LCDC pins are initialized with Reset. (Stop LCDC alternating current output) Duty and Static of segment output is supported. (SEG23/ST0, SEG24/ST1, SEG25/ST2, SEG26/ST3, SEG27/ST4, SEG28/ST5, SEG29/ST6, SEG30/ST7, SEG31/ST8)
SHE	See the Traveo™ Platform Hardware Manual in detail.
Source Clock Timer	See the Traveo™ Platform Hardware Manual in detail.

Feature	Description
Graphics Subsystem	80 MHz maximum clock frequency Variable setting about GDC clock. (Asynchronous with CPU clock) 480 x 272 pixels maximum frame resolution Video modes up to 25 MHz pixel clock RGB888, Order replacement of RGB pins.
External BUS	TEQFP208 : 22 bit address and 16 bit data TEQFP176 : 22 bit address and 16 bit data TEQFP144 : 15 bit address and 8 bit data
ARH	2 ch This device does not have PHY macro and its function.
Power Supply Control (PSC)	PSC (PSC_1) output is used for external 1.2-V power supply module control and automatically switched with the following condition. "High": Request to supply VCC12 - "Power ON Reset" is released - CPU wakes up from PSS shutdown mode "Low": Request to stop supplying VCC12 - CPU transfers from RUN mode to PSS shutdown mode. For timing chart of output signals include PSC in detail, see the "S6J3300 Hardware Manual" and chapter "State Transition"

Note:

– *See the common information of the option features on the Traveo™ Platform Hardware Manual. In the case of option of default and Other Type, S6J3300 select the default.*

2.1. Ethernet

The following functions are not supported.

Functions	Remark
External FIFO Interface	
Additional Low Latency TX FIFO Interface for DMA configurations	
MAC Transmit Block - half-duplex - collision - back_pressure	
MAC Filtering Block - external address match - Wakeup On Lan	
Energy Efficient Ethernet support	
LPI Operation in Cadence IP	
PHY Interface - GMII - SGMII - TBI	
10/100/1000 Operation - 1000 M	
SGMII Operation	
Jumbo Frames	
Physical Control Sub-Layer	

2.2. Reset Signal

The following table shows the reset signal of each function.

See CHAPTER 4: Reset of the Traveo™ Platform Hardware Manual for details of reset signal "RSTX_*".

Functions	CHAPTER	Reset signal	Remark
12-/10-/8-bit Analog to Digital Converter	CHAPTER 15	RSTX_PD2	
Stepper Motor Controller	CHAPTER 16	RSTX_PD2	
Trigger Configuration of Stepper Motor Controller	CHAPTER 17	RSTX_PD2	
Sound Generator	CHAPTER 18	RSTX_PD2	
Sound Waveform Generator	CHAPTER 19	RSTX_PD2	
Sound Mixer	CHAPTER 20	RSTX_PD2	
Ethernet MAC	CHAPTER 21	RSTX_PD2	
Media Local Bus Interface	CHAPTER 22	RSTX_PD2	
Stereo Audio DAC	CHAPTER 23	RSTX_PD2	
I2S	CHAPTER 24	RSTX_PD2	
Programmable CRC	CHAPTER 25	RSTX_PD2	
PCMPWM	CHAPTER 26	RSTX_PD2	
HyperBus Interface	CHAPTER 27	RSTX_PD2	
LCD Controller	CHAPTER 28	RSTX_PD2	For LCDC
		RSTX_PD1	For LCDE
Indicator PWM	CHAPTER 29	RSTX_PD1	
Memory Protection Unit for AXI	CHAPTER 30	RSTX_PD2	
Memory Protection Unit for AHB	CHAPTER 31	RSTX_PD2	
Sound System Configuration	CHAPTER 32	RSTX_PD2	
Graphics Subsystem	CHAPTER 33	RSTX_PD2	
Automotive Remote Handler	CHAPTER 34	RSTX_PD2	
GPIO	Appendix	RSTX_IO_5V	For VCC5, DVCC
		RSTX_IO_3V	For VCC3
		RSTX_IO_35V	For VCC35

3. Note

3.1. Status Flag Clear

Note that the hardware operation of a write access to a register which has status flags may be later than program operations of software. The delay results from the fact that the write accesses and their signals are operated through multiple buses.

For example, when a software program intends to return from interrupt software routine (ISR) after clearing the interrupt flag, the return instruction may practically be executed before the completion of the write access as for hardware operation. That is, the CPU execution may immediately jump to the ISR again after returning from the ISR because the flag is not cleared.

To avoid such a phenomenon, the execution of Data Memory Barrier (DMB) instruction between the write access and the return instruction is recommended for the program. The return instruction is never executed before the completion of DMB execution.

It should be considered that the method takes a number of execution cycles and that it may cause some influence to application performance. For example, one time of the DMB execution is enough after a number of continuous flag clear operations.

3.2. Error Response

Error response is generated when access occurs to the following register and bit offset.

Function	Register	Bit Offset	Access	Error Type
12/10/8-BIT ANALOG TO DIGITAL CONVERTER	ADC12Bn_CHSTAT0 to 63	Whole register	Write(B,H,W)	Bus error
	ADC12Bn_CD0 to 63	Whole register	Write(B,H,W)	Bus error
	ADC12Bn_CDONEIRQ0 to 1	Whole register	Write(B,H,W)	Bus error
	ADC12Bn_GRPIRQ0 to 1	Whole register	Write(B,H,W)	Bus error
	ADC12Bn_RCIRQ0 to 1	Whole register	Write(B,H,W)	Bus error
	ADC12Bn_PCIRQ0 to 1	Whole register	Write(B,H,W)	Bus error
	ADC12Bn_TRGST0 to 1	Whole register	Write(B,H,W)	Bus error
	ADC12Bn_RCOTF0 to 1	Whole register	Write(B,H,W)	Bus error
	ADC12Bn_TRGOR0 to 1	Whole register	Write(B,H,W)	Bus error
	ADC12Bn_MCSTAT0 to 3	Whole register	Write(B,H,W)	Bus error
	(Other reserved area)	(Other reserved area)	Read / Write(B,H,W)	Bus error
Stepper Motor Controller	SMCi_PWSS	bit [7:0] Reserved	Write(B)	Bus error
	SMCi_PTRGDL	bit [15:8] Reserved	Write(B)	Bus error
Trigger configuration of Stepper motor controller	SMCTGg_PTRG	bit [15:8] Reserved	Write(B)	Bus error
INTER IC SOUND (I2S)	I2Sn_RXFDAT0 to 15	Whole register	Write(B,H,W)	Bus error
	I2Sn_OPRREG	bit [15:8] Reserved	Write(B)	Bus error
	I2Sn_SRST	bit [31:8] Reserved	Write(B,H)	Bus error
	I2Sn_STATUS	bit [15:0]	Write(B,H,W)	Bus error
	I2Sn_DMAACT	bit [31:24] Reserved bit [15:8] Reserved	Write(B)	Bus error
	I2Sn_DEBUG	bit [31:8] Reserved	Write(B,H)	Bus error
	I2Sn_MIDREG	Whole register	Write(B,H,W)	Bus error

Function	Register	Bit Offset	Access	Error Type
MEDIA LOCAL BUS INTERFACE (Media LB)	MLBn_DCCR	bit [15:8] Reserved	Write(B)	Bus error
	MLBn_SSCR	bit [31:8] Reserved	Write(B,H)	Bus error
	MLBn_SDCR	Whole register	Write(B,H,W)	Bus error
	MLBn_SMCR	bit [31:8] Reserved	Write(B,H)	Bus error
	MLBn_VCCR	Whole register	Write(B,H,W)	Bus error
	MLBn_CICR	Whole register	Write(B,H,W)	Bus error
	MLBn_CSCR0-15	bit [31:24]	Write(B)	Bus error
	MLBn_CCBCR0-15	Whole register	Write(B,H,W)	Bus error
	MLBn_MID	Whole register	Write(B,H,W)	Bus error
Automotive Remote Handler	ARHn_RHCTRL	bit [23:0]	Write(B,H)	Bus error
	ARHn_CHCTRL0 to 1	bit [23:8]	Write(B,H)	Bus error
	ARHn_CHSTAT0 to 1	Whole register	Write(B,H,W)	Bus error
	ARHn_CHWDGCTL0 to 1	bit [23:8]	Write(B,H)	Bus error
	ARHn_CHWDGCNT0 to 1	Whole register	Write(B,H,W)	Bus error
	ARHn_TBCTRL0 to 15	bit [15:8]	Write(B)	Bus error
	ARHn_TBIRQ	Whole register	Write(B,H)	Bus error
	ARHn_TBIDX0 to 1	Whole register	Write(B)	Bus error
	ARHn_TFADDR0 to 15	bit [31:24] Reserved	Write(B)	Bus error
	ARHn_EVCTRL	bit [23:8]	Write(B,H)	Bus error
	ARHn_EVIRQC	bit [23:16] bit [7:0] Reserved	Write(B)	Bus error
	ARHn_EVBUF0	bit [15:0] Reserved	Write(B,H)	Bus error
	ARHn_APCFG12	bit [7:0] Reserved	Write(B)	Bus error
	ARHn_TST	bit [31:16] Reserved	Write(B,H)	Bus error
	ARHn_MID	Whole register	Write(B,H,W)	Bus error
	ARHn_APCFG04	bit [23:16]	Write(B)	Bus error
	ARHn_APCFG14	bit [23:16]	Write(B)	Bus error
	ARHn_EVAL0	bit [31:16]	Write(B,H)	Bus error
	ARHn_EVAL1	Whole register	Write(B,H,W)	Bus error
	ARHn_EVAL3	Whole register	Write(B,H,W)	Bus error
	ARHn_EVAL5	Whole register	Write(B,H,W)	Bus error
	ARHn_EVAL7	Whole register	Write(B,H,W)	Bus error

3.3. Restriction

Function	Related Register and Configuration	Restriction	Remark
DDRHSSPI	DDRHSSPI Peripheral Communication Configuration Registers (DDRHSSPIn_PCC0-3)	SS2CD[1:0] = 00 cannot be used. Configure delay as 01, 10, or 11.	
Power domain	SYSC0_RUNPDCFGR.PD5_xEN SYSC0_PSSPDCFGR.PD5_xEN SYSC0_APPPDCFGR.PD5_xEN SYSC0_STSPDCFGR.PD5_xEN	Configure them as same as PD2 registers because PD5 is a sub power domain of PD2.	As for PD3 related registers, they are specified as (R1, WX).
	SYSC0_RUNPDCFGR.PD6_1EN SYSC0_PSSPDCFGR.PD6_1EN SYSC0_APPPDCFGR.PD6_1EN SYSC0_STSPDCFGR.PD6_1EN	PD6 is controlled by PD6_0EN only. PD6 is not effected by PD6_1EN setting.	
Reset	-	All RAM data (include TCRAM, System RAM, Backup RAM) are not guaranteed after release of RSTX pin input reset.	This restriction is applied to the following product type. - S6J33xxxxC
Security	TCFCFG0_CSWP0.SWP0 to SWP31 bit TCFCFG0_CSWP1to8.SWP0 to SWP31 bit	In order to write successfully any sector of Code Flash, all Sectors must have their SWP (TCFCFG0_CSWP0.SWP0 to SWP31 , TCFCFG0_CSWP1.SWP0 to SWP31) set to 1	This restriction is applied to the following product type. - S6J33xxxxC

CHAPTER 4: Block Diagram

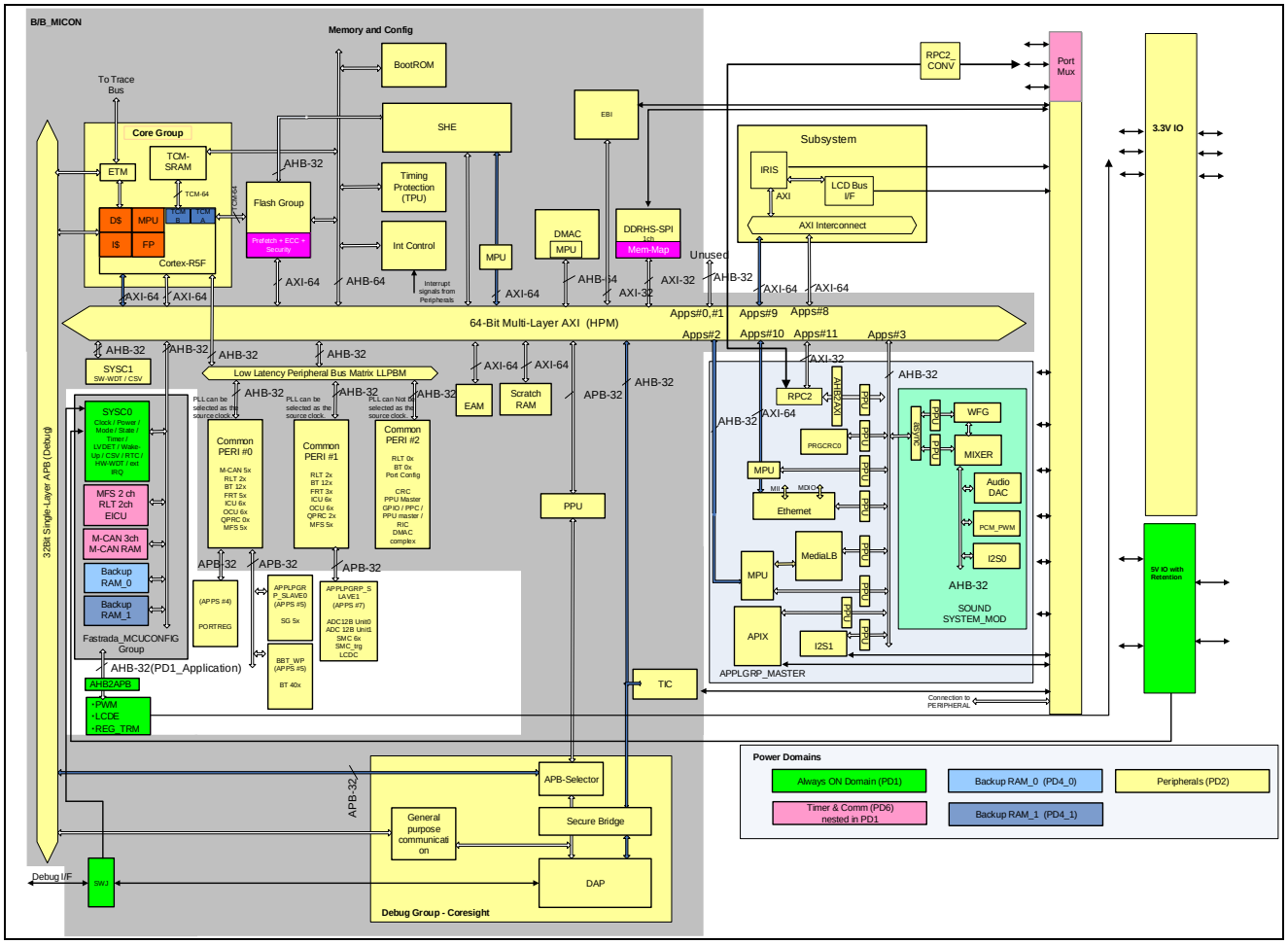
This chapter explains the block diagram.

1. Block Diagram
2. Note

CODE: BLOCK_DIAGRAM-S6J3300-E1

1. Block Diagram

Figure 4-1 Block Diagram



2. Note

No description.

CHAPTER 5: Clock Configuration

This chapter explains the clock configuration.

1. Overview
2. Operation
3. Remark

CODE: CLOCK_SYSTEM-S6J3300-E1

1. Overview

This chapter describes clock selection and configuration of each function.

See the chapter of the clock system before referring this chapter.

2. Operation

The source clock of each function should be selected as following table.

Table 5-1 Source Clock of Each Function

Function	Clock Name	Source Clock System	Configuration
CPU	CLK_CPU	SSCG0/PLL0	See Traveo™ Platform Hardware Manual
FLASH	CLK_FCLK	SSCG0/PLL0	See Traveo™ Platform Hardware Manual
HPM	CLK_HPM	SSCG0/PLL0	See Traveo™ Platform Hardware Manual
DMA	CLK_DMA	SSCG0/PLL0	See Traveo™ Platform Hardware Manual
LLPBM	CLK_LLPBM2	SSCG0/PLL0	See Traveo™ Platform Hardware Manual
MCU configuration	CLK_SYSC0H CLK_COMH CLK_RAM0H CLK_RAM1H	SSCG0/PLL0	See Traveo™ Platform Hardware Manual
DDR HSSPI	CLK_HSSPI	SSCG1	See Traveo™ Platform Hardware Manual
Graphics Core (Graphic)	CLK_CD3A0	SSCG2	See Traveo™ Platform Hardware Manual
Hyper FLASH	CLK_CD1	SSCG3	See Traveo™ Platform Hardware Manual
CAN	CLK_CAN	PLL0	See Traveo™ Platform Hardware Manual
Display clock	CLK_CD2A0	PLL1	See Traveo™ Platform Hardware Manual
CoreSight	CLK_TRC	PLL1	See Traveo™ Platform Hardware Manual
I2S0	CLK_CD4	PLL2	See Traveo™ Platform Hardware Manual It is described as "ECLK" in the chapter of I2S.
	CLK_CD5B0	PLL3	See Traveo™ Platform Hardware Manual
I2S1	CLK_HAPP1B0	SSCG0	See Traveo™ Platform Hardware Manual
	CLK_CD4	PLL2	See Traveo™ Platform Hardware Manual It is described as "ECLK" in the chapter of I2S.
Audio DAC	CLK_CD4	PLL2	See Traveo™ Platform Hardware Manual It is described as "CLKDACI" in the chapter of Audio DAC.
	CLK_CD5B0	PLL3	See Traveo™ Platform Hardware Manual It is described as "CLKPI" in the chapter of Audio DAC.
PCM-PWM	CLK_CD4	PLL2	See See Traveo™ Platform Hardware Manual It is described as "PWM_CLK" in the chapter of PCMPWM.
	CLK_CD5B0	PLL3	See Traveo™ Platform Hardware Manual
Sound waveform generator	CLK_CD5 CLK_CD5A0 CLK_CD5B0	PLL3	See Traveo™ Platform Hardware Manual
Sound mixer	CLK_CD5 CLK_CD5A0 CLK_CD5B0	PLL3	See Traveo™ Platform Hardware Manual
Ethernet Media-LB	CLK_HAPP1B0	SSCG0/PLL0	See Traveo™ Platform Hardware Manual
Registers of Indicator PWM and LCDE	CLK_SYSC0P	SSCG0/PLL0	See Traveo™ Platform Hardware Manual
Registers of LCDC	CLK_LCP1A	PPL0 or SSCG0	See Traveo™ Platform Hardware Manual
Hardware watchdog timer	CLK_HWWDT	Internal CR oscillator (High frequency) or Internal CR oscillator (Low frequency)	See Traveo™ Platform Hardware Manual

Function	Clock Name	Source Clock System	Configuration
Software watchdog timer	CLK_SWWDT	Internal CR oscillator (High frequency) or Internal CR oscillator (Low frequency) or Main clock or Sub clock	See Traveo™ Platform Hardware Manual
External interrupt capture unit	CLK_EICU	Main clock	See Traveo™ Platform Hardware Manual
Real time clock	CLK_RTC	Main clock	See Traveo™ Platform Hardware Manual
Real time clock	CLK_RTC	Sub clock	See Traveo™ Platform Hardware Manual

Note:

- *The frequency of PLLout (output of PLL/SSCG PLL multiplier circuit) should be 480 MHz or less.*
- *The frequency of CLK_CPU should be 240 MHz or less.*

The relation between source clock and local clock in product-specific functions is as follows.

Table 5-2: Source Clock - Local Clock List

Source Clock System	Clock Name	Function	Local Clock Name	Description
External input clock	External input clock	Ethernet MAC	TX_CLK	-
External input clock	External input clock	Ethernet MAC	RX_CLK	-
External input clock	External input clock	Inter IC Sound (I2S0/I2S1)	SCK	-
External input clock	External input clock	Inter IC Sound (I2S0/I2S1)	ECLK	-
External input clock	External input clock	Media Local Bus Interface (MediaLB)	MLBn_CLK	-
Main clock	Main clock	Indicator PWM	Main clock	-
Main clock	Main clock	LCD Controller	Main clock	-
PLL0 or SSCG0	CLK_LCP1A	12/10/8-BIT Analog to Digital Converter	No Define	-
PLL0 or SSCG0	CLK_LCP1A	LCD Controller	No Define	-
PLL0 or SSCG0	CLK_LCP1A	Trigger Configuration of Stepper Motor Controller	No Define	-
PLL0 or SSCG0	CLK_LCP0A	Sound Generator	BCLK	-
PLL0 or SSCG0	CLK_LCP1A	Stepper Motor Controller	CLKP	-
PLL1	CLK_CD2A0	Graphics Subsystem	No Define	Display clock
PLL2	CLK_CD4	Inter IC Sound (I2S0/I2S1)	ECLK	-
PLL2	CLK_CD4	PCMPWM	PWM_CLK	-
PLL2	CLK_CD4	Stereo Audio DAC	CLKDACI	-
PLL3	CLK_CD5B0	Inter IC Sound (I2S0)	Internal clock	-
PLL3	CLK_CD5B0	PCMPWM	No Define	-
PLL3	CLK_CD5	Sound Mixer	No Define	Waveform Generator and Sound Mixer bus interface clock
PLL3	CLK_CD5A0	Sound Mixer	No Define	Waveform Generator and Sound Mixer operation clock
PLL3	CLK_CD5	Sound Waveform Generator	No Define	Waveform Generator and Sound Mixer bus interface clock
PLL3	CLK_CD5A0	Sound Waveform Generator	No Define	Waveform Generator and Sound Mixer operation clock
PLL3	CLK_CD5B0	Stereo Audio DAC	CLKPI	-
PLL0 or SSCG0	CLK_HAPP1B0	Ethernet MAC	No Define	-
SSCG0	CLK_HAPP1B0	Inter IC Sound (I2S1)	Internal clock	

Source Clock System	Clock Name	Function	Local Clock Name	Description
PLL0 or SSCG0	CLK_HAPP1B0	Hyperbus Interface (MCU)	No Define	Operation Clock for Control register
PLL0 or SSCG0	CLK_HPM	Hyperbus Interface (MCU)	No Define	Operation Clock for Main controller
PLL0 or SSCG0	CLK_SYSC0P	CLK_LCP1A:or PWM	No Define	-
PLL0 or SSCG0	CLK_HAPP1B0	Media Local Bus Interface (MediaLB)	No Define	-
PLL0 or SSCG0	CLK_HAPP1B0	Memory Protection Unit for AHB	No Define	-
PLL0 or SSCG0	CLK_HAPP1B0	Memory Protection Unit for AHB	No Define	-
PLL0 or SSCG0	CLK_HAPP1B0	Memory Protection Unit for AXI	No Define	-
PLL0 or SSCG0	CLK_HAPP1B0	Memory Protection Unit for AXI	No Define	-
PLL0 or SSCG0	CLK_HAPP1B0	Programmable CRC	No Define	-
SSCG1	CLK_HSSPI	Graphics Subsystem	iHCLK	-
SSCG2	CLK_CD3A0	Graphics Subsystem	No Define	Graphics Core
SSCG2	CLK_CD3A0	Hyperbus Interface (GDC)	No Define	Operation Clock for Control register
SSCG2	CLK_CD3A0	Hyperbus Interface (GDC)	No Define	Operation Clock for Main controller
SSCG3	CLK_CD1	Hyperbus Interface (MCU)	No Define	External Clock for HyperBus memory interface
SSCG3	CLK_CD1	Hyperbus Interface (GDC)	No Define	External Clock for HyperBus memory interface
Sub clock	Sub Clock	LCD Controller	Sub clock	-

Notes:

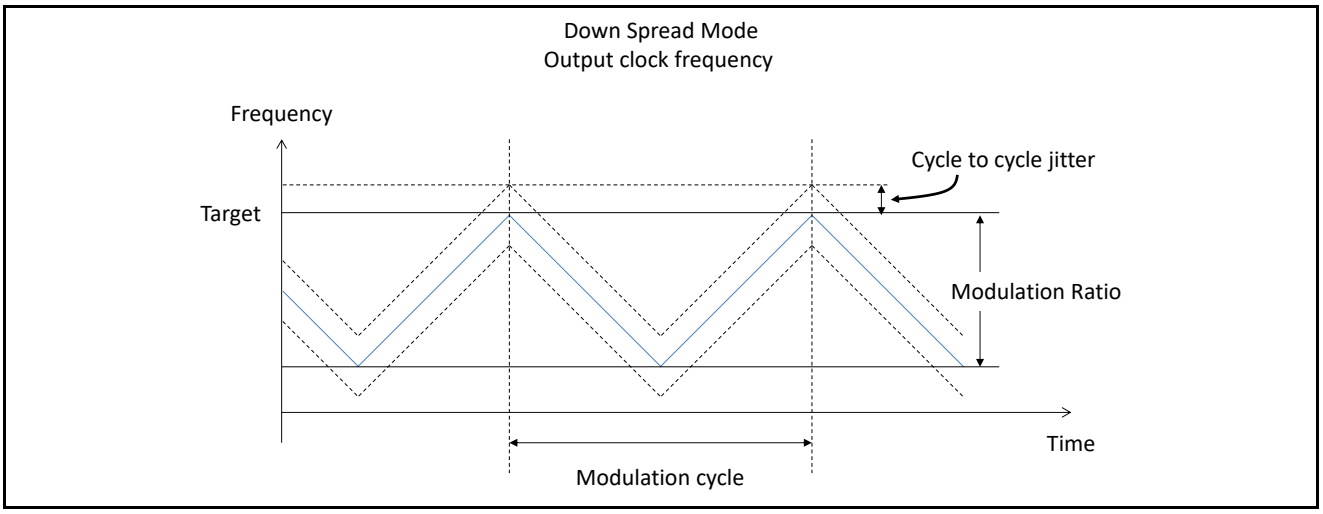
- *The configuration of the maximum clock frequency above should satisfy the values specified in Datasheet.*
- *The frequency of CLK_CD5 and CLK_CD5A0 should satisfy the following conditions.*
- *CLK_CD5 = 240 MHz or 120 MHz.*
- *CLK_CD5A0 = 120 MHz.*
- *Read/Write access from CPU or DMA may bring about dead-lock when the clock source is not supplied to the accessed clock domain because "handshake" for AXI transaction cannot be done. If you want to quit a clock generation for some clock domain, you also need to configure an access protection for the domain using MPU.*

2.1. Spread Spectrum Clock Generator (SSCG)

Target frequency of SSCG should be referred in Datasheet.

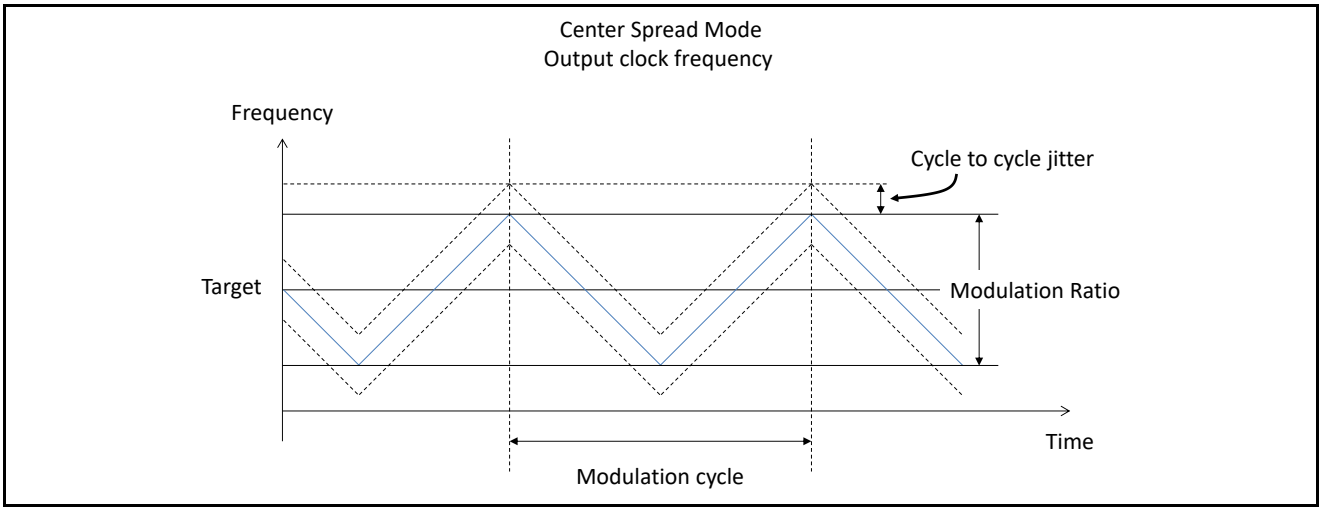
■ **Down Spread Mode**

Figure 5-1: Down Spread Mode



■ **Center Spread Mode**

Figure 5-2: Center Spread Mode



3. Remark

See the chapter of clock system before referring this chapter.

Note

− If you want to know a combination of a peripheral function and its source clock, you need to see the group name from the table of base address map on this manual at first.

− The group name and its clock source are described in PLATFORM OVERVIEW Configuration. See Traveo™ Platform Hardware Manual.

− Clock frequency can be seen in datasheet of S6J3300 series.

CHAPTER 6: Operation Mode

This chapter explains operation mode.

1. Overview
2. Configuration
3. Registers

CODE: MODE-S6J3300-E1

1. Overview

This section gives a description of operation mode.

The mode controller determines the operation mode of MCU. It supports,

- User mode
- Serial programming mode (with synchronous communication)
- Serial programming mode (with asynchronous communication)
- JTAG boundary scan mode

2. Configuration

Operation Mode	PORT		
	MODE	P1_05(SOT0)	P1_03(SIN0)
User Mode	1	-	-
Serial Programming Mode (Sync)	0	1	0
Serial Programming Mode (Async)	0	1	1
JTAG Boundary Scan Mode	0	0	0

Note:

- *User mode can be applied to Software debugging using JTAG interface with ICE.*
- *As for serial programming, see the chapter of SERIAL PROGRAMMING on this manual.*

3. Registers

See Traveo™ Platform Hardware Manual.

CHAPTER 7: Memory and Base Address Map

This chapter explains the memory map and the base address map of registers.

1. Overview
2. Memory Map
3. Base Address Map

CODE: MEMORYMAP-S6J3300-E1

1. Overview

When MPU attribute of Cortex-R5F is configured as "Normal", store buffer inside Cortex-R5F can operate and write data can be merged. To avoid influence of this data merger, MPU attribute "Device" or "Strongly Ordered" should be used.

MPU attribute "Device" or "Strongly Ordered" must be used for areas below, to avoid this influence.

- Peripheral area (Peri area) [B000_0000 to B7FF_FFFF]
- Error Config area (ERRCFG) [FFFE_E000 to FFFE_FFFF]
- AppS area [B800_0000 to BFFF_FFFF]
- Register area in Graphic subsystem [5020_0000 to 5FFF_FFFF]
- Backup RAM area (BACKUP_RAM) [0E80_0000 to 0E87_FFFF]

MPU attribute "Device" or "Strongly Ordered" is required for accesses to areas below, in particular situation.

- FLASH Memory (when writing commands)
- External Bus Interface (when writing commands to NAND FLASH)

2. Memory Map

START Address	END Address	Group	Part
0000_0000	0001_FFFF	Internal area for CR5 Complex	TCRAM 128 KB@Max
0002_0000	007F_FFFF		Reserved
0080_0000	009E_FFFF		Reserved
009F_0000	009F_FFFF		TCFLASH 64 KB (Small Sector 8 KB x 8)
00A0_0000	00DF_FFFF		TCFLASH 4 MB@Max
00E0_0000	00FF_FFFF		Reserved
0100_0000	019E_FFFF		Reserved
019F_0000	019F_FFFF		AXI_FLASH 64 KB (Small Sector 8 KB x 8)
01A0_0000	01DF_FFFF		AXI_FLASH 4 MB@Max
01E0_0000	01FF_FFFF		Reserved
0200_0000	0205_FFFF	Shared Flash and memory area	System RAM 384 KB@Max
0206_0000	027F_FFFF		Reserved
0280_0000	0280_0FFF		Exclusive Access Memory
0280_1000	03FF_FFFF		Reserved
0400_0000	05FF_FFFF		AXI_SLAVE_CORE0
0600_0000	07FF_FFFF		Reserved
0800_0000	09FF_FFFF		Reserved
0A00_0000	0BFF_FFFF		Reserved
0C00_0000	0DFF_FFFF		Reserved
0E00_0000	0E01_BFFF		WORK_FLASH 112 KB (mirror area 1)
0E01_C000	0E1F_FFFF		Reserved
0E20_0000	0E21_BFFF		WORK_FLASH 112 KB (mirror area 3)
0E21_C000	0E2F_FFFF		Reserved
0E30_0000	0E31_BFFF		WORK_FLASH 112 KB (mirror area 4)
0E31_C000	0E3F_FFFF		Reserved
0E40_0000	0E7F_FFFF		Reserved
0E80_0000	0E80_7FFF		BACKUP_RAM 32 KB@Max
0E80_8000	0EFF_FFFF		Reserved
0F00_0000	0FFF_FFFF		Reserved
1000_0000	1FFF_FFFF	Ext-Bus area	EBI_MEMORY(SRAM/FLASH)
2000_0000	2FFF_FFFF	Reserved	Reserved
3000_0000	3FFF_FFFF		

START Address	END Address	Group	Part
4000_0000	4FFF_FFFF	Reserved	Reserved
5000_0000	501F_FFFF	Graphic subsystem (AppS #8)	Reserved
5020_0000	5020_03FF		2D Graphics Core Subsystem control
5020_0400	5020_47FF		2D Graphics core(Blit,Drawing,Capture,Display,CmdSeq)
5020_4800	5020_4BFF		LCD Bus interface
5020_4C00	5021_3FFF		Reserved
5021_4000	5021_43FF		AXI Interconnect(aic_performance)
5021_4400	5021_47FF		AXI Interconnect(Error Monitor)
5021_4800	503F_FFFF		Reserved
5040_0000	504F_FFFF		AXI Interconnect(NIC301)
5050_0000	5FFF_FFFF		Reserved
6000_0000	7FFF_FFFF	Reserved	Reserved
8000_0000	8FFF_FFFF	HSSPI0_MEMORY	DDRHSSPI for CPU area
9000_0000	9FFF_FFFF	HSSPI1_MEMORY(Apps#11)	HyperBus Memory for CPU area
A000_0000	AFFF_FFFF	Reserved	Reserved
B000_0000	B7FF_FFFF	Peri area	Peri area
B800_0000	B802_87FF	Apps#3	Expansion area
B802_8800	BFFF_FFFF	Reserved	Reserved
C000_0000	EFFF_FFFF	Reserved	Reserved
F000_0000	FFFE_DFFF	BootROM/ERRCFG area	Reserved
FFFE_E000	FFFE_EFFF		ERRCFG
FFFE_F000	FFFE_FFFF		
FFFF_0000	FFFF_FFFF		BootROM

Memory map of BACKUP_RAM

START Address	END Address	Part
0E80_0000	0E80_3FFF	BACKUP_RAM in PD4_0
0E80_4000	0E80_7FFF	BACKUP_RAM in PD4_1

3. Base Address Map

An address of a certain register can be specified as below.

- Base address
  Look for the base address X of the function from the base address map below.
- Offset address
  Look for the offset address Y of the register from the offset address list which is described in the chapter of the function. Each function chapter has offset address list or its information.
- Specify
  The register address can be specified as X + Y.

Note:

- *Address area of not-implemented function is not supported. See the chapter of function list.*

START Address	END Address	Group	Function	PPU_No
B000_0000	B00F_FFFF	Reserved	Reserved / AppS area for LLPP-AXI32 (1 MB)	-
B010_0000	B010_03FF	EMEM GROUP(reg)	EBI registers	0
B010_0400	B010_0FFF	Reserved	Reserved	-
B010_1000	B010_13FF	EMEM GROUP(reg)	DDR_HSSPI	1
B010_1400	B010_7FFF	Reserved	Reserved	-
B010_8000	B010_80FF	System SRAM	SystemSRAM registers	2
B010_8100	B01F_FFFF	Reserved	Reserved	-
B020_0000	B02F_FFFF	Reserved	Reserved	-
B030_0000	B030_7FFF	SYSC1	System Controller #1	4
B030_8000	B03F_FFFF	SYSC1	SWDT	5
B040_0000	B040_7FFF	MEM Config GROUP	IRC0	21
B040_8000	B040_FFFF	MEM Config GROUP	TPU0	19
B041_0000	B041_0FFF	MEM Config GROUP	TCRAM Control Status Register	16
B041_1000	B041_1FFF	MEM Config GROUP	TCFlash Control Status Register	17
B041_2000	B041_20FF	MEM Config GROUP	Wflash Control Status Register	18
B041_2100	B04F_FFFF	Reserved	Reserved	-
B050_0000	B050_0FFF	DEBUG GROUP	DAP ROM table	-
B050_1000	B050_1FFF	DEBUG GROUP	ETB	-
B050_2000	B050_2FFF	DEBUG GROUP	CTI#4	-
B050_3000	B050_3FFF	DEBUG GROUP	TPIU	-
B050_4000	B057_FFFF	DEBUG GROUP	TRACE FUNNEL	-
B058_0000	B058_FFFF	DEBUG GROUP	CR5_RomTable	-
B059_0000	B059_1FFF	DEBUG GROUP	CORE0	-
B059_2000	B059_7FFF	Reserved	Reserved	-
B059_8000	B059_8FFF	DEBUG GROUP	CTI#0	-
B059_9000	B059_BFFF	Reserved	Reserved	-
B059_C000	B059_CFFF	DEBUG GROUP	ETM0	-
B059_D000	B05B_FFFF	Reserved	Reserved	-
B05C_0000		DEBUG GROUP	Security Checker	-
	B05F_FFFF	Reserved	Reserved	-
B060_0000	B060_007F	MCU_CONFIG_GROUP	Protection register area	51

START Address	END Address	Group	Function	PPU_No
B060_0080	B060_00FF	MCU_CONFIG_GROUP	RUN profile register area	51
B060_0100	B060_017F	MCU_CONFIG_GROUP	PSS profile register area	51
B060_0180	B060_01FF	MCU_CONFIG_GROUP	APP profile register area	51
B060_0200	B060_027F	MCU_CONFIG_GROUP	STS profile register area	51
B060_0280	B060_02FF	MCU_CONFIG_GROUP	System register area	51
B060_0300	B060_037F	MCU_CONFIG_GROUP	CSV	51
B060_0380	B060_03FF	MCU_CONFIG_GROUP	RESET	51
B060_0400	B060_047F	MCU_CONFIG_GROUP	SCT(Fast CR)	34
B060_0480	B060_04FF	MCU_CONFIG_GROUP	SCT(Slow CR)	33
B060_0500	B060_057F	MCU_CONFIG_GROUP	SCT(Main clock)	35
B060_0580	B060_05FF	MCU_CONFIG_GROUP	SCT(Sub clock)	36
B060_0600	B060_067F	MCU_CONFIG_GROUP	Clock System	51
B060_0680	B060_06FF	MCU_CONFIG_GROUP	Special register area	51
B060_0700	B060_07FF	MCU_CONFIG_GROUP	Debug register area	51
B060_0800	B060_BFFF	MCU_CONFIG_GROUP	MODEC	55
B060_C000	B060_FFFF	MCU_CONFIG_GROUP	HWDT	52
B061_0000	B061_7FFF	Reserved	Reserved	-
B061_8000	B061_FFFF	MCU_CONFIG_GROUP	RTC	32
B062_0000	B063_FFFF	MCU_CONFIG_GROUP	Ext-IRQ	53
B064_0000	B064_0FFF	MCU_CONFIG_GROUP	Indicator PWM	364
B064_1000	B064_1FFF	MCU_CONFIG_GROUP	LCDE	365
B064_2000	B064_2FFF	Reserved	Rserved	-
B064_3000	B065_FFFF	Reserved	Rserved	-
B066_0000	B067_FFFF	Reserved	Rserved	-
B068_0000	B068_7FFF	MCU_CONFIG_GROUP	BackupRam CSR	50
B068_8000	B068_83FF	MCU_CONFIG_GROUP	EICU	37
B068_8400	B068_87FF	MCU_CONFIG_GROUP	CR_CALIBRATION	38
B068_8800	B068_8BFF	MCU_CONFIG_GROUP	MCG_IRS	42
B068_8C00	B068_FFFF	MCU_CONFIG_GROUP	CAN_PRESCALER	43
B069_0000	B069_03FF	MCU_CONFIG_GROUP	ReloadTimer ch.48	39
B069_0400	B069_07FF	MCU_CONFIG_GROUP	ReloadTimer ch.49	40
B069_0800	B06A_7FFF	Reserved	Reserved	-
B06A_8000	B06A_83FF	MCU_CONFIG_GROUP	M.F.Serial ch.16	44
B06A_8400	B06A_87FF	MCU_CONFIG_GROUP	M.F.Serial ch.17	45
B06A_8800	B06B_FFFF	Reserved	Reserved	-
B06C_0000	B06C_FFFF	MCU_CONFIG_GROUP	CAN_FD ch.5	47
B06D_0000	B06D_FFFF	MCU_CONFIG_GROUP	CAN_FD ch.6	48
B06E_0000	B06E_FFFF	MCU_CONFIG_GROUP	CAN_FD ch.7	49
B06F_0000	B06F_FFFF	Reserved	Reserved	-
B070_0000	B07F_FFFF	Reserved	Reserved (1 MB)	-
B080_0000	B0FF_FFFF	Bit RMW alias	Bit RMW alias for MCU Config (Covers B060_0000 -- B06F_FFFF)	-
B100_0000	B10F_FFFF	Bit RMW alias	Bit RMW alias for SYSC1 (Covers B030_0000 -- B031_FFFF)	-
B110_0000	B11F_FFFF	Bit RMW alias	Bit RMW alias for MEMC (Covers B040_0000 -- B041_FFFF)	-

START Address	END Address	Group	Function	PPU_No
B120_0000	B1FF_FFFF	Reserved	Reserved (14 MB)	-
B200_0000	B20F_FFFF	SHE_CONFIG	SHE configuration registers (1 MB)	63
B210_0000	B3FF_FFFF	Reserved	Reserved / AppS area for LLPP-AXI32 (31 MB)	-
B400_0000	B46F_FFFF	Reserved	Reserved (7 MB)	-
B470_0000	B470_3FFF	Common PERI #2	DMAC #0 registers	64
B470_4000	B470_7FFF	Reserved	Reserved	-
B470_8000	B470_FFFF	Reserved	Reserved	-
B471_0000	B471_0FFF	Common PERI #2	MPU for DMAC#0	66
B471_1000	B471_1FFF	Reserved	Reserved	-
B471_2000	B471_3FFF	Reserved	Reserved	-
B471_4000	B471_4FFF	Common PERI #2	DMA Complex #0 registers (Additional registers, RLTs)	68
B471_5000	B471_5FFF	Reserved	Reserved	-
B471_6000	B471_7FFF	Reserved	Reserved	-
B471_8000	B471_83FF	Common PERI #2	CRC#0	70
B471_8400	B471_87FF	Common PERI #2	CRC#1	71
B471_8800	B471_8BFF	Common PERI #2	CRC#2	72
B471_8C00	B471_8FFF	Common PERI #2	CRC#3	73
B471_9000	B473_7FFF	Reserved	Reserved	-
B473_8000	B473_FFFF	Common PERI #2	GPIO	74
B474_0000	B474_7FFF	Common PERI #2	PPC	75
B474_8000	B474_FFFF	Common PERI #2	RIC	76
B475_0000	B475_7FFF	Common PERI #2	PPU	-
B475_8000	B478_FFFF	Reserved	Reserved	-
B479_0000	B479_03FF	Reserved	Reserved	-
B479_0000	B47F_FFFF	Reserved	Reserved	-
B480_0000	B480_03FF	Common PERI #0	M.F.Serial ch.0	176
B480_0400	B480_07FF	Common PERI #0	M.F.Serial ch.1	177
B480_0800	B480_0BFF	Common PERI #0	M.F.Serial ch.2	178
B480_0C00	B480_0FFF	Common PERI #0	M.F.Serial ch.3	179
B480_1000	B480_13FF	Common PERI #0	M.F.Serial ch.4	180
B480_1400	B480_7FFF	Reserved	Reserved	-
B480_8000	B480_83FF	Common PERI #0	BaseTimer ch.0	88
B480_8400	B480_87FF	Common PERI #0	BaseTimer ch.1	89
B480_8800	B480_8BFF	Common PERI #0	BaseTimer ch.2	90
B480_8C00	B480_8FFF	Common PERI #0	BaseTimer ch.3	91
B480_9000	B480_93FF	Common PERI #0	BaseTimer ch.4	92
B480_9400	B480_97FF	Common PERI #0	BaseTimer ch.5	93
B480_9800	B480_9BFF	Common PERI #0	BaseTimer ch.6	94
B480_9C00	B480_9FFF	Common PERI #0	BaseTimer ch.7	95
B480_A000	B480_A3FF	Common PERI #0	BaseTimer ch.8	96
B480_A400	B480_A7FF	Common PERI #0	BaseTimer ch.9	97
B480_A800	B480_ABFF	Common PERI #0	BaseTimer ch.10	98
B480_AC00	B480_AFFF	Common PERI #0	BaseTimer ch.11	99

START Address	END Address	Group	Function	PPU_No
B480_B000	B480_FFFF	Reserved	Reserved	-
B481_0000	B481_03FF	Common PERI #0	Reload Timer ch.0	128
B481_0400	B481_07FF	Common PERI #0	Reload Timer ch.1	129
B481_0800	B481_7FFF	Reserved	Reserved	-
B481_8000	B481_FFFF	Reserved	Reserved	-
B482_0000	B482_03FF	Common PERI #0	FRT ch.0	208
B482_0400	B482_07FF	Common PERI #0	FRT ch.1	209
B482_0800	B482_0BFF	Common PERI #0	FRT ch.2	210
B482_0C00	B482_0FFF	Common PERI #0	FRT ch.3	211
B482_1000	B482_13FF	Common PERI #0	FRT ch.4	212
B482_1400	B482_7FFF	Reserved	Reserved	-
B482_8000	B482_83FF	Common PERI #0	Input Capture 0	224
B482_8400	B482_87FF	Common PERI #0	Input Capture 1	225
B482_8800	B482_8BFF	Common PERI #0	Input Capture 2	226
B482_8C00	B482_FFFF	Reserved	Reserved	-
B483_0000	B483_03FF	Common PERI #0	Output Compare 0	240
B483_0400	B483_07FF	Common PERI #0	Output Compare 1	241
B483_0800	B483_0BFF	Common PERI #0	Output Compare 2	242
B483_0C00	B483_FBFF	Reserved	Reserved	-
B483_FC00	B483_FFFF	Common PERI #0	Reload Timer Simultaneous Soft Start for ch.0/1	80
B484_0000	B484_03FF	Common PERI #0	SG ch.0	266
B484_0400	B484_07FF	Common PERI #0	SG ch.1	267
B484_0800	B484_0BFF	Common PERI #0	SG ch.2	268
B484_0C00	B484_0FFF	Common PERI #0	SG ch.3	269
B484_1000	B484_13FF	Common PERI #0	SG ch.4	270
B484_1400	B484_5FFF	Reserved	Reserved (APPS#5 area)	-
B484_6000	B484_63FF	Common PERI #0	BaseTimer ch.24	290
B484_6400	B484_67FF	Common PERI #0	BaseTimer ch.25	291
B484_6800	B484_6BFF	Common PERI #0	BaseTimer ch.26	292
B484_6C00	B484_6FFF	Common PERI #0	BaseTimer ch.27	293
B484_7000	B484_73FF	Common PERI #0	BaseTimer ch.28	294
B484_7400	B484_77FF	Common PERI #0	BaseTimer ch.29	295
B484_7800	B484_7BFF	Common PERI #0	BaseTimer ch.30	296
B484_7C00	B484_7FFF	Common PERI #0	BaseTimer ch.31	297
B484_8000	B484_83FF	Common PERI #0	BaseTimer ch.32	298
B484_8400	B484_87FF	Common PERI #0	BaseTimer ch.33	299
B484_8800	B484_8BFF	Common PERI #0	BaseTimer ch.34	300
B484_8C00	B484_8FFF	Common PERI #0	BaseTimer ch.35	301
B484_9000	B484_93FF	Common PERI #0	BaseTimer ch.36	302
B484_9400	B484_97FF	Common PERI #0	BaseTimer ch.37	303
B484_9800	B484_9BFF	Common PERI #0	BaseTimer ch.38	304
B484_9C00	B484_9FFF	Common PERI #0	BaseTimer ch.39	305
B484_A000	B484_A3FF	Common PERI #0	BaseTimer ch.40	306
B484_A400	B484_A7FF	Common PERI #0	BaseTimer ch.41	307

START Address	END Address	Group	Function	PPU_No
B484_A800	B484_ABFF	Common PERI #0	BaseTimer ch.42	308
B484_AC00	B484_AFFF	Common PERI #0	BaseTimer ch.43	309
B484_B000	B484_B3FF	Common PERI #0	BaseTimer ch.44	310
B484_B400	B484_B7FF	Common PERI #0	BaseTimer ch.45	311
B484_B800	B484_BBFF	Common PERI #0	BaseTimer ch.46	312
B484_BC00	B484_BFFF	Common PERI #0	BaseTimer ch.47	313
B484_C000	B484_C3FF	Common PERI #0	BaseTimer ch.48	314
B484_C400	B484_C7FF	Common PERI #0	BaseTimer ch.49	315
B484_C800	B484_CBFF	Common PERI #0	BaseTimer ch.50	316
B484_CC00	B484_CFFF	Common PERI #0	BaseTimer ch.51	317
B484_D000	B484_D3FF	Common PERI #0	BaseTimer ch.52	318
B484_D400	B484_D7FF	Common PERI #0	BaseTimer ch.53	319
B484_D800	B484_DBFF	Common PERI #0	BaseTimer ch.54	320
B484_DC00	B484_DFFF	Common PERI #0	BaseTimer ch.55	321
B484_E000	B484_E3FF	Common PERI #0	BaseTimer ch.56	322
B484_E400	B484_E7FF	Common PERI #0	BaseTimer ch.57	323
B484_E800	B484_EBFF	Common PERI #0	BaseTimer ch.58	324
B484_EC00	B484_EFFF	Common PERI #0	BaseTimer ch.59	325
B484_F000	B484_F3FF	Common PERI #0	BaseTimer ch.60	326
B484_F400	B484_F7FF	Common PERI #0	BaseTimer ch.61	327
B484_F800	B484_FBFF	Common PERI #0	BaseTimer ch.62	328
B484_FC00	B484_FFFF	Common PERI #0	BaseTimer ch.63	329
B485_0000	B485_03FF	Reserved	Reserved (APPS#4 area)	-
B485_0400	B485_07FF	Reserved	Reserved	-
B486_0800	B487_FFFF	Reserved	Reserved (APPS#4 area)	-
B488_0000	B488_03FF	Common PERI #1	M.F.Serial ch.8	184
B488_0400	B488_07FF	Common PERI #1	M.F.Serial ch.9	185
B488_0800	B488_0BFF	Common PERI #1	M.F.Serial ch.10	186
B488_0C00	B488_0FFF	Common PERI #1	M.F.Serial ch.11	187
B488_1000	B488_13FF	Common PERI #1	M.F.Serial ch.12	188
B488_1400	B488_7FFF	Reserved	Reserved	-
B488_8000	B488_83FF	Common PERI #1	BaseTimer ch.12	100
B488_8400	B488_87FF	Common PERI #1	BaseTimer ch.13	101
B488_8800	B488_8BFF	Common PERI #1	BaseTimer ch.14	102
B488_8C00	B488_8FFF	Common PERI #1	BaseTimer ch.15	103
B488_9000	B488_93FF	Common PERI #1	BaseTimer ch.16	104
B488_9400	B488_97FF	Common PERI #1	BaseTimer ch.17	105
B488_9800	B488_9BFF	Common PERI #1	BaseTimer ch.18	106
B488_9C00	B488_9FFF	Common PERI #1	BaseTimer ch.19	107
B488_A000	B488_A3FF	Common PERI #1	BaseTimer ch.20	108
B488_A400	B488_A7FF	Common PERI #1	BaseTimer ch.21	109
B488_A800	B488_ABFF	Common PERI #1	BaseTimer ch.22	110
B488_AC00	B488_AFFF	Common PERI #1	BaseTimer ch.23	111
B488_B000	B488_FFFF	Reserved	Reserved	-

START Address	END Address	Group	Function	PPU_No
B489_0000	B489_03FF	Common PERI #1	Reload Timer ch.16	144
B489_0400	B489_07FF	Common PERI #1	Reload Timer ch.17	145
B489_0800	B489_7FFF	Reserved	Reserved	-
B489_8000	B489_83FF	Common PERI #1	QPRC ch.8	200
B489_8400	B489_87FF	Common PERI #1	QPRC ch.9	201
B489_8800	B489_FFFF	Reserved	Reserved	-
B48A_0000	B48A_03FF	Common PERI #1	FRT ch.8	216
B48A_0400	B48A_07FF	Common PERI #1	FRT ch.9	217
B48A_0800	B48A_0BFF	Common PERI #1	FRT ch.10	218
B48A_0C00	B48A_7FFF	Reserved	Reserved	-
B48A_8000	B48A_83FF	Common PERI #1	Input Capture 8	232
B48A_8400	B48A_87FF	Common PERI #1	Input Capture 9	233
B48A_8800	B48A_8BFF	Common PERI #1	Input Capture 10	234
B48A_8C00	B48A_FFFF	Reserved	Reserved	-
B48B_0000	B48B_03FF	Common PERI #1	Output Compare 8	248
B48B_0400	B48B_07FF	Common PERI #1	Output Compare 9	249
B48B_0800	B48B_0BFF	Common PERI #1	Output Compare 10	250
B48B_0C00	B48B_FBFF	Reserved	Reserved	-
B48B_FC00	B48B_FFFF	Common PERI #1	Reload Timer Simultaneous Soft Start for ch.16/17	81
B48C_0000	B48C_03FF	Common PERI #1	ADC12B0	330
B48C_0400	B48C_07FF	Common PERI #1	ADC12B1	331
B48C_0800	B48C_3FFF	Reserved	Reserved	-
B48C_4000	B48C_43FF	Common PERI #1	SMC ch.0	346
B48C_4400	B48C_47FF	Common PERI #1	SMC ch.1	347
B48C_4800	B48C_4BFF	Common PERI #1	SMC ch.2	348
B48C_4C00	B48C_4FFF	Common PERI #1	SMC ch.3	349
B48C_5000	B48C_53FF	Common PERI #1	SMC ch.4	350
B48C_5400	B48C_57FF	Common PERI #1	SMC ch.5	351
B48C_5800	B48C_5BFF	Common PERI #1	SMC_Trg_Reg	352
B48C_5C00	B48C_5FFF	Common PERI #1	LCDC	353
B48C_6000	B48C_FFFF	Reserved	Reserved (APPS#7 area)	-
B48D_0000	B48D_FFFF	Reserved	Reserved (APPS#6 area)	-
B48E_0000	B48F_FFFF	Reserved	Reserved	-
B490_0000	B490_FFFF	Common Peri#0	CAN_FD ch.0	256
B491_0000	B491_FFFF	Common Peri#0	CAN_FD ch.1	257
B492_0000	B492_FFFF	Common Peri#0	CAN_FD ch.2	258
B493_0000	B493_FFFF	Common Peri#0	CAN_FD ch.3	259
B494_0000	B494_FFFF	Common Peri#0	CAN_FD ch.4	260
B495_0000	B497_FFFF	Reserved	Reserved	-
B498_0000	B4BF_FFFF	Reserved	Reserved	-
B4C0_0000	B4FF_FFFF	Bit RMW alias	Bit RMW alias for CommonPERI#0 (Covers B490_0000 -- B497_FFFF)	-
B500_0000	B5FF_FFFF	AppS #0	AppS #0 (16 MB)	-
B600_0000	B6FF_FFFF	AppS #1	AppS #1 (16 MB)	-

START Address	END Address	Group	Function	PPU_No
B700_0000	B77F_FFFF	Bit RMW alias	Bit RMW alias for CommonPERI#2 (Covers B470_0000 -- B47F_FFFF)	-
B780_0000	B7BF_FFFF	Bit RMW alias	Bit RMW alias for CommonPERI#0 (Covers B480_0000 -- B487_FFFF)	-
B7C0_0000	B7FF_FFFF	Bit RMW alias	Bit RMW alias for CommonPERI#1 (Covers B488_0000 -- B48F_FFFF)	-
B800_0000	B800_07FF	Apps#3	ETHERNET	359
B800_0800	B800_0BFF	Apps#3	MPU_AXI	360
B800_0C00	B800_7FFF	Reserved	Reserved	-
B800_8000	B800_83FF	Apps#3	MLB	354
B800_8400	B800_87FF	Apps#3	MPU_AHB	355
B800_8800	B801_7FFF	Reserved	Reserved	-
B801_8000	B801_83FF	Apps#3	PRGCRC	356
B801_8400	B801_87FF	Reserved	Reserved	-
B801_8800	B801_FFFF	Reserved	Reserved	-
B802_0000	B802_03FF	Apps#3	SMIX (Sound System)	361
B802_0400	B802_07FF	Apps#3	Audio DAC (Sound System)	361
B802_0800	B802_0BFF	Apps#3	PCM-PWM (Sound System)	361
B802_0C00	B802_0FFF	Apps#3	I2S0 (Sound System)	361
B802_1000	B802_13FF	Apps#3	I2S1	358
B802_1400	B802_7FFF	Reserved	Reserved	-
B802_8000	B802_83FF	Apps#3	Wave Form Generator (Sound System)	357
B802_8400	B802_87FF	Apps#3	HyperBus Register for CPU	362
B802_8800	B802_FFFF	Reserved	Reserved	-
B803_0000	B803_03FF	Apps#3	ARH0	363
B803_0400	BFFF_FFFF	Reserved	Reserved	-
C000_0000	FFFE_DFFF	Reserved	Reserved	-
FFFE_E000	FFFE_E3FF	ERRCFG	IRC0 (IRC0_NMIVASBR)	-
FFFE_E400	FFFE_F7FF	Reserved	Reserved	-
FFFE_F800	FFFE_FBFF	ERRCFG	IRC (IRC_NMIVASBR) Mirror *	-
FFFE_FC00	FFFE_FFFF	ERRCFG	BootROM I/F	20

Note:

- *DMA doesn't support the output modules of the sound system; Audio-DAC, PCM-PWM, and I2S0.*

CHAPTER 8: IRQ Map / NMI Map

This chapter explains IRQ MAP and NMI MAP.

1. IRQ MAP
2. NMI MAP

CODE: IRQMAP-S6J3300-E1

1. IRQ MAP

This section shows list of interrupt vector.

This list shows the assignments of interrupt vectors / interrupt control registers.

IRQ No.	Detail	IRQ Priority Register	Vector Address Register
0	Reserved		
1	System Control Status	IRC0_IRQPL0 : IRQPL1	IRC0_IRQVA1
2	HW-WDT Pre-warning	IRC0_IRQPL0 : IRQPL2	IRC0_IRQVA2
3	SW-WDT Pre-warning	IRC0_IRQPL0 : IRQPL3	IRC0_IRQVA3
4 to 7	Reserved		
8	TCFLASH RDY, Hang up, Single Bit Error	IRC0_IRQPL2 : IRQPL8	IRC0_IRQVA8
9	Reserved		
10	Work FLASH Hang up	IRC0_IRQPL2 : IRQPL10	IRC0_IRQVA10
11 to 13	Reserved		
14	System RAM Single Bit Error	IRC0_IRQPL3 : IRQPL14	IRC0_IRQVA14
15	Backup RAM / CAN FD RAM(ch.0 to ch.7) Single Bit Error	IRC0_IRQPL3 : IRQPL15	IRC0_IRQVA15
16	IRC Vector Address RAM Single Bit Error	IRC0_IRQPL4 : IRQPL16	IRC0_IRQVA16
17 to 19	Reserved		
20	Work FLASH RDY, Write Enable Release, Single Bit Error	IRC0_IRQPL5 : IRQPL20	IRC0_IRQVA20
21 to 22	Reserved		
23	EICU	IRC0_IRQPL5 : IRQPL23	IRC0_IRQVA23
24	External Interrupt Request ch.0 / ch.16	IRC0_IRQPL6 : IRQPL24	IRC0_IRQVA24
25	External Interrupt Request ch.1 / ch.17	IRC0_IRQPL6 : IRQPL25	IRC0_IRQVA25
26	External Interrupt Request ch.2 / ch.18	IRC0_IRQPL6 : IRQPL26	IRC0_IRQVA26
27	External Interrupt Request ch.3 / ch.19	IRC0_IRQPL6 : IRQPL27	IRC0_IRQVA27
28	External Interrupt Request ch.4 / ch.20	IRC0_IRQPL7 : IRQPL28	IRC0_IRQVA28
29	External Interrupt Request ch.5 / ch.21	IRC0_IRQPL7 : IRQPL29	IRC0_IRQVA29
30	External Interrupt Request ch.6 / ch.22	IRC0_IRQPL7 : IRQPL30	IRC0_IRQVA30
31	External Interrupt Request ch.7 /ch.23	IRC0_IRQPL7 : IRQPL31	IRC0_IRQVA31
32	External Interrupt Request ch.8	IRC0_IRQPL8 : IRQPL32	IRC0_IRQVA32
33	External Interrupt Request ch.9	IRC0_IRQPL8 : IRQPL33	IRC0_IRQVA33
34	External Interrupt Request ch.10	IRC0_IRQPL8 : IRQPL34	IRC0_IRQVA34
35	External Interrupt Request ch.11	IRC0_IRQPL8 : IRQPL35	IRC0_IRQVA35
36	External Interrupt Request ch.12	IRC0_IRQPL9 : IRQPL36	IRC0_IRQVA36
37	External Interrupt Request ch.13	IRC0_IRQPL9 : IRQPL37	IRC0_IRQVA37
38	External Interrupt Request ch.14	IRC0_IRQPL9 : IRQPL38	IRC0_IRQVA38
39	External Interrupt Request ch.15	IRC0_IRQPL9 : IRQPL39	IRC0_IRQVA39
40	MFS RX ch.16	IRC0_IRQPL10 : IRQPL40	IRC0_IRQVA40
41	MFS TX ch.16	IRC0_IRQPL10 : IRQPL41	IRC0_IRQVA41
42	MFS RX ch.17	IRC0_IRQPL10 : IRQPL42	IRC0_IRQVA42
43	MFS TX ch.17	IRC0_IRQPL10 : IRQPL43	IRC0_IRQVA43
44 to 45	Reserved		
46	Reload Timer ch.48,49 OR-ed	IRC0_IRQPL11 : IRQPL46	IRC0_IRQVA46
47	Reserved		

IRQ No.	Detail	IRQ Priority Register	Vector Address Register
48	CAN FD ch.5	IRC0_IRQPL12 : IRQPL48	IRC0_IRQVA48
49	CAN FD ch.6	IRC0_IRQPL12 : IRQPL49	IRC0_IRQVA49
50	CAN FD ch.7	IRC0_IRQPL12 : IRQPL50	IRC0_IRQVA50
51 to 55	Reserved		
56	CAN FD ch.0	IRC0_IRQPL14 : IRQPL56	IRC0_IRQVA56
57	CAN FD ch.1	IRC0_IRQPL14 : IRQPL57	IRC0_IRQVA57
58	CAN FD ch.2	IRC0_IRQPL14 : IRQPL58	IRC0_IRQVA58
59	CAN FD ch.3	IRC0_IRQPL14 : IRQPL59	IRC0_IRQVA59
60	CAN FD ch.4	IRC0_IRQPL15 : IRQPL60	IRC0_IRQVA60
61 to 63	Reserved		
64	MFS RX ch.0	IRC0_IRQPL16 : IRQPL64	IRC0_IRQVA64
65	MFS TX ch.0	IRC0_IRQPL16 : IRQPL65	IRC0_IRQVA65
66	MFS RX ch.1	IRC0_IRQPL16 : IRQPL66	IRC0_IRQVA66
67	MFS TX ch.1	IRC0_IRQPL16 : IRQPL67	IRC0_IRQVA67
68	MFS RX ch.2	IRC0_IRQPL17 : IRQPL68	IRC0_IRQVA68
69	MFS TX ch.2	IRC0_IRQPL17 : IRQPL69	IRC0_IRQVA69
70	MFS RX ch.3	IRC0_IRQPL17 : IRQPL70	IRC0_IRQVA70
71	MFS TX ch.3	IRC0_IRQPL17 : IRQPL71	IRC0_IRQVA71
72	MFS RX ch.4	IRC0_IRQPL18 : IRQPL72	IRC0_IRQVA72
73	MFS TX ch.4	IRC0_IRQPL18 : IRQPL73	IRC0_IRQVA73
74 to 79	Reserved		
80	MFS RX ch.8	IRC0_IRQPL20 : IRQPL80	IRC0_IRQVA80
81	MFS TX ch.8	IRC0_IRQPL20 : IRQPL81	IRC0_IRQVA81
82	MFS RX ch.9	IRC0_IRQPL20 : IRQPL82	IRC0_IRQVA82
83	MFS TX ch.9	IRC0_IRQPL20 : IRQPL83	IRC0_IRQVA83
84	MFS RX ch.10	IRC0_IRQPL21 : IRQPL84	IRC0_IRQVA84
85	MFS TX ch.10	IRC0_IRQPL21 : IRQPL85	IRC0_IRQVA85
86	MFS RX ch.11	IRC0_IRQPL21 : IRQPL86	IRC0_IRQVA86
87	MFS TX ch.11	IRC0_IRQPL21 : IRQPL87	IRC0_IRQVA87
88	MFS RX ch.12	IRC0_IRQPL22 : IRQPL88	IRC0_IRQVA88
89	MFS TX ch.12	IRC0_IRQPL22 : IRQPL89	IRC0_IRQVA89
90 to 99	Reserved		
100	SHE Error	IRC0_IRQPL25 : IRQPL100	IRC0_IRQVA100
101	SHE	IRC0_IRQPL25 : IRQPL101	IRC0_IRQVA101
102	DDR HSSPI RX	IRC0_IRQPL25 : IRQPL102	IRC0_IRQVA102
103	DDR HSSPI TX	IRC0_IRQPL25 : IRQPL103	IRC0_IRQVA103
104 to 109	Reserved		
110	TCRAM	IRC0_IRQPL27 : IRQPL110	IRC0_IRQVA110
111	Reserved		
112	BACKUP RAM	IRC0_IRQPL28 : IRQPL112	IRC0_IRQVA112
113 to 115	Reserved		
116	RTC	IRC0_IRQPL29 : IRQPL116	IRC0_IRQVA116

IRQ No.	Detail	IRQ Priority Register	Vector Address Register
117	CR CARIBRATION	IRC0_IRQPL29 : IRQPL117	IRC0_IRQVA117
118 to 127	Reserved		
128	Base Timer ch.0/8/9/10/11	IRC0_IRQPL32 : IRQPL128	IRC0_IRQVA128
129	Base Timer ch.1	IRC0_IRQPL32 : IRQPL129	IRC0_IRQVA129
130	Base Timer ch.2	IRC0_IRQPL32 : IRQPL130	IRC0_IRQVA130
131	Base Timer ch.3	IRC0_IRQPL32 : IRQPL131	IRC0_IRQVA131
132	Base Timer ch.4	IRC0_IRQPL33 : IRQPL132	IRC0_IRQVA132
133	Base Timer ch.5	IRC0_IRQPL33 : IRQPL133	IRC0_IRQVA133
134	Base Timer ch.6	IRC0_IRQPL33 : IRQPL134	IRC0_IRQVA134
135	Base Timer ch.7	IRC0_IRQPL33 : IRQPL135	IRC0_IRQVA135
136	Base Timer ch.12/20/21/22/23	IRC0_IRQPL34 : IRQPL136	IRC0_IRQVA136
137	Base Timer ch.13	IRC0_IRQPL34 : IRQPL137	IRC0_IRQVA137
138	Base Timer ch.14	IRC0_IRQPL34 : IRQPL138	IRC0_IRQVA138
139	Base Timer ch.15	IRC0_IRQPL34 : IRQPL139	IRC0_IRQVA139
140	Base Timer ch.16	IRC0_IRQPL35 : IRQPL140	IRC0_IRQVA140
141	Base Timer ch.17	IRC0_IRQPL35 : IRQPL141	IRC0_IRQVA141
142	Base Timer ch.18	IRC0_IRQPL35 : IRQPL142	IRC0_IRQVA142
143	Base Timer ch.19	IRC0_IRQPL35 : IRQPL143	IRC0_IRQVA143
144 to 151	Reserved		
152	Reload Timer ch.0	IRC0_IRQPL38 : IRQPL152	IRC0_IRQVA152
153	Reload Timer ch.1	IRC0_IRQPL38 : IRQPL153	IRC0_IRQVA153
154 to 159	Reserved		
160	Reload Timer ch.16	IRC0_IRQPL40 : IRQPL160	IRC0_IRQVA160
161	Reload Timer ch.17	IRC0_IRQPL40 : IRQPL161	IRC0_IRQVA161
162 to 175	Reserved		
176	FRT ch.0	IRC0_IRQPL44 : IRQPL176	IRC0_IRQVA176
177	FRT ch.1	IRC0_IRQPL44 : IRQPL177	IRC0_IRQVA177
178	FRT ch.2	IRC0_IRQPL44 : IRQPL178	IRC0_IRQVA178
179	FRT ch.3	IRC0_IRQPL44 : IRQPL179	IRC0_IRQVA179
180	FRT ch.4	IRC0_IRQPL45 : IRQPL180	IRC0_IRQVA180
181 to 183	Reserved		
184	FRT ch.8	IRC0_IRQPL46 : IRQPL184	IRC0_IRQVA184
185	FRT ch.9	IRC0_IRQPL46 : IRQPL185	IRC0_IRQVA185
186	FRT ch.10	IRC0_IRQPL46 : IRQPL186	IRC0_IRQVA186
187 to 191	Reserved		
192	IRQ0 of Input Capture 0	IRC0_IRQPL48 : IRQPL192	IRC0_IRQVA192
193	IRQ0 of Input Capture 1	IRC0_IRQPL48 : IRQPL193	IRC0_IRQVA193
194	IRQ0 of Input Capture 2	IRC0_IRQPL48 : IRQPL194	IRC0_IRQVA194
195 to 199	Reserved		
200	IRQ0 of Input Capture 8	IRC0_IRQPL50 : IRQPL200	IRC0_IRQVA200
201	IRQ0 of Input Capture 9	IRC0_IRQPL50 : IRQPL201	IRC0_IRQVA201

IRQ No.	Detail	IRQ Priority Register	Vector Address Register
202	IRQ0 of Input Capture 10	IRC0_IRQPL50 : IRQPL202	IRC0_IRQVA202
203 to 207	Reserved		
208	IRQ0 of Output Compare 0	IRC0_IRQPL52 : IRQPL208	IRC0_IRQVA208
209	IRQ0 of Output Compare 1	IRC0_IRQPL52 : IRQPL209	IRC0_IRQVA209
210	IRQ0 of Output Compare 2	IRC0_IRQPL52 : IRQPL210	IRC0_IRQVA210
211 to 215	Reserved		
216	IRQ0 of Output Compare 8	IRC0_IRQPL54 : IRQPL216	IRC0_IRQVA216
217	IRQ0 of Output Compare 9	IRC0_IRQPL54 : IRQPL217	IRC0_IRQVA216
218	IRQ0 of Output Compare 10	IRC0_IRQPL54 : IRQPL218	IRC0_IRQVA216
219 to 231	Reserved		
232	QPRC ch.8	IRC0_IRQPL58 : IRQPL232	IRC0_IRQVA232
233	QPRC ch.9	IRC0_IRQPL58 : IRQPL233	IRC0_IRQVA233
234 to 239	Reserved		
240	IRQ1 of Input Capture 0	IRC0_IRQPL60 : IRQPL240	IRC0_IRQVA240
241	IRQ1 of Input Capture 1	IRC0_IRQPL60 : IRQPL241	IRC0_IRQVA241
242	IRQ1 of Input Capture 2	IRC0_IRQPL60 : IRQPL242	IRC0_IRQVA242
243 to 247	Reserved		
248	IRQ1 of Input Capture 8	IRC0_IRQPL62 : IRQPL248	IRC0_IRQVA248
249	IRQ1 of Input Capture 9	IRC0_IRQPL62 : IRQPL249	IRC0_IRQVA249
250	IRQ1 of Input Capture 10	IRC0_IRQPL62 : IRQPL250	IRC0_IRQVA250
251 to 255	Reserved		
256	IRQ1 of Output Compare 0	IRC0_IRQPL64 : IRQPL256	IRC0_IRQVA256
257	IRQ1 of Output Compare 1	IRC0_IRQPL64 : IRQPL257	IRC0_IRQVA257
258	IRQ1 of Output Compare 2	IRC0_IRQPL64 : IRQPL258	IRC0_IRQVA258
259 to 263			
264	IRQ1 of Output Compare 8	IRC0_IRQPL66 : IRQPL264	IRC0_IRQVA264
265	IRQ1 of Output Compare 9	IRC0_IRQPL66 : IRQPL265	IRC0_IRQVA265
266	IRQ1 of Output Compare 10	IRC0_IRQPL66 : IRQPL266	IRC0_IRQVA266
267 to 271	Reserved		
272	DMA Error	IRC0_IRQPL68 : IRQPL272	IRC0_IRQVA272
273	DMAC Completion ch.0	IRC0_IRQPL68 : IRQPL273	IRC0_IRQVA273
274	DMAC Completion ch.1	IRC0_IRQPL68 : IRQPL274	IRC0_IRQVA274
275	DMAC Completion ch.2	IRC0_IRQPL68 : IRQPL275	IRC0_IRQVA275
276	DMAC Completion ch.3	IRC0_IRQPL69 : IRQPL276	IRC0_IRQVA276
277	DMAC Completion ch.4	IRC0_IRQPL69 : IRQPL277	IRC0_IRQVA277
278	DMAC Completion ch.5	IRC0_IRQPL69 : IRQPL278	IRC0_IRQVA278
279	DMAC Completion ch.6	IRC0_IRQPL69 : IRQPL279	IRC0_IRQVA279
280	DMAC Completion ch.7	IRC0_IRQPL70 : IRQPL280	IRC0_IRQVA280
281	Reserved		
282	DMAC RLT(ch.0,1,2,3 OR-ed)	IRC0_IRQPL70 : IRQPL282	IRC0_IRQVA282
283 to 284	Reserved		

IRQ No.	Detail	IRQ Priority Register	Vector Address Register
285	DMAC Completion ch.8	IRC0_IRQPL71 : IRQPL285	IRC0_IRQVA285
286	DMAC Completion ch.9	IRC0_IRQPL71 : IRQPL286	IRC0_IRQVA286
287	DMAC Completion ch.10	IRC0_IRQPL71 : IRQPL287	IRC0_IRQVA287
288	DMAC Completion ch.11	IRC0_IRQPL72 : IRQPL288	IRC0_IRQVA288
289	DMAC Completion ch.12	IRC0_IRQPL72 : IRQPL289	IRC0_IRQVA289
290	DMAC Completion ch.13	IRC0_IRQPL72 : IRQPL290	IRC0_IRQVA290
291	DMAC Completion ch.14	IRC0_IRQPL72 : IRQPL291	IRC0_IRQVA291
292	DMAC Completion ch.15	IRC0_IRQPL73 : IRQPL292	IRC0_IRQVA292
293 to 307	Reserved		
308	SCT CR IRQ	IRC0_IRQPL77 : IRQPL308	IRC0_IRQVA308
309	SCT SRC IRQ	IRC0_IRQPL77 : IRQPL309	IRC0_IRQVA309
310	SCT Main OSC IRQ	IRC0_IRQPL77 : IRQPL310	IRC0_IRQVA310
311	SCT Sub OSC IRQ	IRC0_IRQPL77 : IRQPL311	IRC0_IRQVA311
312	CR5 Performance Monitor Unit IRQ	IRC0_IRQPL78 : IRQPL312	IRC0_IRQVA312
313 to 319	Reserved		
320	MFS ch.0 Error (Tx/Rx error, Status OR-ed)	IRC0_IRQPL80 : IRQPL320	IRC0_IRQVA320
321	MFS ch.1 Error (Tx/Rx error, Status OR-ed)	IRC0_IRQPL80 : IRQPL321	IRC0_IRQVA321
322	MFS ch.2 Error (Tx/Rx error, Status OR-ed)	IRC0_IRQPL80 : IRQPL322	IRC0_IRQVA322
323	MFS ch.3 Error (Tx/Rx error, Status OR-ed)	IRC0_IRQPL80 : IRQPL323	IRC0_IRQVA323
324	MFS ch.4 Error (Tx/Rx error, Status OR-ed)	IRC0_IRQPL81 : IRQPL324	IRC0_IRQVA324
325 to 327	Reserved		
328	MFS ch.8 Error (Tx/Rx error, Status OR-ed)	IRC0_IRQPL82 : IRQPL328	IRC0_IRQVA328
329	MFS ch.9 Error (Tx/Rx error, Status OR-ed)	IRC0_IRQPL82 : IRQPL329	IRC0_IRQVA329
330	MFS ch.10 Error (Tx/Rx error, Status OR-ed)	IRC0_IRQPL82 : IRQPL330	IRC0_IRQVA330
331	MFS ch.11 Error (Tx/Rx error, Status OR-ed)	IRC0_IRQPL82 : IRQPL331	IRC0_IRQVA331
332	MFS ch.12 Error (Tx/Rx error, Status OR-ed)	IRC0_IRQPL83 : IRQPL332	IRC0_IRQVA332
333 to 335	Reserved		
336	MFS ch.16 Error (Tx/Rx error, Status OR-ed)	IRC0_IRQPL84 : IRQPL336	IRC0_IRQVA336
337	MFS ch.17 Error (Tx/Rx error, Status OR-ed)	IRC0_IRQPL84 : IRQPL337	IRC0_IRQVA337
338 to 346	Reserved		
347	SG ch.0	IRC0_IRQPL86 : IRQPL347	IRC0_IRQVA347
348	SG ch.1	IRC0_IRQPL87 : IRQPL348	IRC0_IRQVA348
349	SG ch.2	IRC0_IRQPL87 : IRQPL349	IRC0_IRQVA349
350	SG ch.3	IRC0_IRQPL87 : IRQPL350	IRC0_IRQVA350
351	SG ch.4	IRC0_IRQPL87 : IRQPL351	IRC0_IRQVA351
352	Reserved		
353	ADC12B0 Conversion Done	IRC0_IRQPL88 : IRQPL353	IRC0_IRQVA353
354	ADC12B0 Group interrupt	IRC0_IRQPL88 : IRQPL354	IRC0_IRQVA354
355	ADC12B0 pulse detection function	IRC0_IRQPL88 : IRQPL355	IRC0_IRQVA355
356	ADC12B0 RCO	IRC0_IRQPL89 : IRQPL356	IRC0_IRQVA356
357	RPGCRC	IRC0_IRQPL89 : IRQPL357	IRC0_IRQVA357

IRQ No.	Detail	IRQ Priority Register	Vector Address Register
358	Reserved		
359	MLB channel interrupt	IRC0_IRQPL89 : IRQPL359	IRC0_IRQVA359
360	MLB system interrupt	IRC0_IRQPL90 : IRQPL360	IRC0_IRQVA360
361	ETHERNET IRQ	IRC0_IRQPL90 : IRQPL361	IRC0_IRQVA361
362	ETHERNET Q1 IRQ	IRC0_IRQPL90 : IRQPL362	IRC0_IRQVA362
363	ETHERNET Q2 IRQ	IRC0_IRQPL90 : IRQPL363	IRC0_IRQVA363
364	ETHERNET Q3 IRQ	IRC0_IRQPL91 : IRQPL364	IRC0_IRQVA364
365 to 367	Reserved		
368	Indicator PWM	IRC0_IRQPL92 : IRQPL368	IRC0_IRQVA368
369	PCMPWM_DREQ	IRC0_IRQPL92 : IRQPL369	IRC0_IRQVA369
370	PCMPWM_OVFL	IRC0_IRQPL92 : IRQPL370	IRC0_IRQVA370
371	PCMPWM_UDRN	IRC0_IRQPL92 : IRQPL371	IRC0_IRQVA371
372	PCMPWM_DMAE	IRC0_IRQPL93 : IRQPL372	IRC0_IRQVA372
373	AUDIO_DAC_DREQ	IRC0_IRQPL93 : IRQPL373	IRC0_IRQVA373
374	AUDIO_DAC_OVFL_IRQ	IRC0_IRQPL93 : IRQPL374	IRC0_IRQVA374
375	AUDIO_DAC_UDRN_IRQ	IRC0_IRQPL93 : IRQPL375	IRC0_IRQVA375
376	AUDIO_DAC_DMAE_IRQ	IRC0_IRQPL94 : IRQPL376	IRC0_IRQVA376
377	I2S0_IRQ	IRC0_IRQPL94 : IRQPL377	IRC0_IRQVA377
378	I2S1_IRQ	IRC0_IRQPL94 : IRQPL378	IRC0_IRQVA378
379 to 387	Reserved		
388	Base Timer ch.24/32/33/34/35	IRC0_IRQPL97 : IRQPL388	IRC0_IRQVA388
389	Base Timer ch.25	IRC0_IRQPL97 : IRQPL389	IRC0_IRQVA389
390	Base Timer ch.26	IRC0_IRQPL97 : IRQPL390	IRC0_IRQVA390
391	Base Timer ch.27	IRC0_IRQPL97 : IRQPL391	IRC0_IRQVA391
392	Base Timer ch.28	IRC0_IRQPL98 : IRQPL392	IRC0_IRQVA392
393	Base Timer ch.29	IRC0_IRQPL98 : IRQPL393	IRC0_IRQVA393
394	Base Timer ch.30	IRC0_IRQPL98 : IRQPL394	IRC0_IRQVA394
395	Base Timer ch.31	IRC0_IRQPL98 : IRQPL395	IRC0_IRQVA395
396 to 403	Reserved		
404	2D Graphics Core ContentStream0	IRC0_IRQPL101 : IRQPL404	IRC0_IRQVA404
405	2D Graphics Core Safety Stream0	IRC0_IRQPL101 : IRQPL405	IRC0_IRQVA405
406	2D Graphics Core Display Stream0	IRC0_IRQPL101 : IRQPL406	IRC0_IRQVA406
407	2D Graphics Core Signature0	IRC0_IRQPL101 : IRQPL407	IRC0_IRQVA407
408	2D Graphics Core Display0 Sync0	IRC0_IRQPL102 : IRQPL408	IRC0_IRQVA408
409	2D Graphics Core Display0 Sync1	IRC0_IRQPL102 : IRQPL409	IRC0_IRQVA409
410 to 420	Reserved		
421	2D Graphics Core LCDBusIf_Control	IRC0_IRQPL105 : IRQPL421	IRC0_IRQVA421
422	2D Graphics Core LCDBusIf_InstrFifo	IRC0_IRQPL105 : IRQPL422	IRC0_IRQVA422
423	2D Graphics Core LCDBusIf_RxFifo	IRC0_IRQPL105 : IRQPL423	IRC0_IRQVA423
424 to 431	Reserved		
432	WG_END_IRQ0	IRC0_IRQPL108 : IRQPL432	IRC0_IRQVA432

IRQ No.	Detail	IRQ Priority Register	Vector Address Register
433	WG_END_IRQ1	IRC0_IRQPL108 : IRQPL433	IRC0_IRQVA433
434	WG_END_IRQ2	IRC0_IRQPL108 : IRQPL434	IRC0_IRQVA434
435	WG_END_IRQ3	IRC0_IRQPL108 : IRQPL435	IRC0_IRQVA435
436	WG_END_IRQ4	IRC0_IRQPL109 : IRQPL436	IRC0_IRQVA436
437	WG_AHB_ERR_IRQ	IRC0_IRQPL109 : IRQPL437	IRC0_IRQVA437
438	MX_DATA_REQ_IRQ0	IRC0_IRQPL109 : IRQPL438	IRC0_IRQVA438
439	MX_DATA_REQ_IRQ1	IRC0_IRQPL109 : IRQPL439	IRC0_IRQVA439
440	MX_DATA_REQ_IRQ2	IRC0_IRQPL110 : IRQPL440	IRC0_IRQVA440
441	MX_DATA_REQ_IRQ3	IRC0_IRQPL110 : IRQPL441	IRC0_IRQVA441
442	MX_DATA_REQ_IRQ4	IRC0_IRQPL110 : IRQPL442	IRC0_IRQVA442
443	MX_OVFL_IRQ0	IRC0_IRQPL110 : IRQPL443	IRC0_IRQVA443
444	MX_OVFL_IRQ1	IRC0_IRQPL111 : IRQPL444	IRC0_IRQVA444
445	MX_OVFL_IRQ2	IRC0_IRQPL111 : IRQPL445	IRC0_IRQVA445
446	MX_OVFL_IRQ3	IRC0_IRQPL111 : IRQPL446	IRC0_IRQVA446
447	MX_OVFL_IRQ4	IRC0_IRQPL111 : IRQPL447	IRC0_IRQVA447
448 to 452	Reserved		
453	MX_DMA_ERR_IRQ0	IRC0_IRQPL113 : IRQPL453	IRC0_IRQVA453
454	MX_DMA_ERR_IRQ1	IRC0_IRQPL113 : IRQPL454	IRC0_IRQVA454
455	MX_DMA_ERR_IRQ2	IRC0_IRQPL113 : IRQPL455	IRC0_IRQVA455
456	MX_DMA_ERR_IRQ3	IRC0_IRQPL114 : IRQPL456	IRC0_IRQVA456
457	MX_DMA_ERR_IRQ4	IRC0_IRQPL114 : IRQPL457	IRC0_IRQVA457
458	MX_AHB_ERR_IRQ	IRC0_IRQPL114 : IRQPL458	IRC0_IRQVA458
459	Reserved		
460	ARH IRQ1 (Event IRQ; Event Buffer fifo Level; Event Buffer fifo overflow; AShell fatal error; watchdog timer)	IRC0_IRQPL115 : IRQPL460	IRC0_IRQVA460
461	ARH IRQ2 (Transaction Buffer Interrupt TBIRQ00 to 15)	IRC0_IRQPL115 : IRQPL461	IRC0_IRQVA461
462 to 463	Reserved		
464	ADC12B1 Conversion Done	IRC0_IRQPL116 : IRQPL464	IRC0_IRQVA464
465	ADC12B1 Group interrupt	IRC0_IRQPL116 : IRQPL465	IRC0_IRQVA465
466	ADC12B1 pulse detection function	IRC0_IRQPL116 : IRQPL466	IRC0_IRQVA466
467	ADC12B1 RCO	IRC0_IRQPL116 : IRQPL467	IRC0_IRQVA467
468	Base Timer ch.36/44/45/46/47	IRC0_IRQPL117 : IRQPL468	IRC0_IRQVA468
469	Base Timer ch.37	IRC0_IRQPL117 : IRQPL469	IRC0_IRQVA469
470	Base Timer ch.38	IRC0_IRQPL117 : IRQPL470	IRC0_IRQVA470
471	Base Timer ch.39	IRC0_IRQPL117 : IRQPL471	IRC0_IRQVA471
472	Base Timer ch.40	IRC0_IRQPL118 : IRQPL472	IRC0_IRQVA472
473	Base Timer ch.41	IRC0_IRQPL118 : IRQPL473	IRC0_IRQVA473
474	Base Timer ch.42	IRC0_IRQPL118 : IRQPL474	IRC0_IRQVA474
475	Base Timer ch.43	IRC0_IRQPL118 : IRQPL475	IRC0_IRQVA475
476	Base Timer ch.48/56/57/58/59	IRC0_IRQPL119 : IRQPL476	IRC0_IRQVA476
477	Base Timer ch.49	IRC0_IRQPL119 : IRQPL477	IRC0_IRQVA477

IRQ No.	Detail	IRQ Priority Register	Vector Address Register
478	Base Timer ch.50	IRC0_IRQPL119 : IRQPL478	IRC0_IRQVA478
479	Base Timer ch.51	IRC0_IRQPL119 : IRQPL479	IRC0_IRQVA479
480	Base Timer ch.52	IRC0_IRQPL120 : IRQPL480	IRC0_IRQVA480
481	Base Timer ch.53	IRC0_IRQPL120 : IRQPL481	IRC0_IRQVA481
482	Base Timer ch.54	IRC0_IRQPL120 : IRQPL482	IRC0_IRQVA482
483	Base Timer ch.55	IRC0_IRQPL120 : IRQPL483	IRC0_IRQVA483
484	Base Timer ch.60	IRC0_IRQPL121 : IRQPL484	IRC0_IRQVA484
485	Base Timer ch.61	IRC0_IRQPL121 : IRQPL485	IRC0_IRQVA485
486	Base Timer ch.62	IRC0_IRQPL121 : IRQPL486	IRC0_IRQVA486
487	Base Timer ch.63	IRC0_IRQPL121 : IRQPL487	IRC0_IRQVA487
488 to 511	Reserved		

2. NMI MAP

This section shows NMI map.

NMI No.	Detail	Priority Level	IRQ/NMI Vector Address
0	NMIX pin(Ext-IRC)	IRC0_NMIPL0 : NMIPL0	IRC0_NMIVA0
1 to 2	Reserved		
3	Reserved		
4	LVDs IRQ	IRC0_NMIPL1 : NMIPL4	IRC0_NMIVA4
5	CSV, Profile	IRC0_NMIPL1 : NMIPL5	IRC0_NMIVA5
6	HW-WDT	IRC0_NMIPL1 : NMIPL6	IRC0_NMIVA6
7	SW-WDT	IRC0_NMIPL1 : NMIPL7	IRC0_NMIVA7
8	IRC 2-bit ECC err detection	IRC0_NMIPL2 : NMIPL8	IRC0_NMIVA8
9 to 10	Reserved		
11	Backup RAM 2-bit ECC error detection	IRC0_NMIPL2 : NMIPL11	IRC0_NMIVA11
12	CAN-FD RAMs 2-bit ECC error detection	IRC0_NMIPL3 : NMIPL12	IRC0_NMIVA12
13	DMAC MPU #0 protection violation	IRC0_NMIPL3 : NMIPL13	IRC0_NMIVA13
14	Reserved		
15	SHE MPU	IRC0_NMIPL3 : NMIPL15	IRC0_NMIVA15
16 to 17	Reserved		
18	TPU protection violation	IRC0_NMIPL4 : NMIPL18	IRC0_NMIVA18
19	Reserved		
20	Graphics subsystem Memory Protection	IRC0_NMIPL5 : NMIPL20	IRC0_NMIVA20
21	Reserved		
22	MLB_MPU_NMI	IRC0_NMIPL5 : NMIPL22	IRC0_NMIVA22
23	ETHERNET_MPU_NMI	IRC0_NMIPL5 : NMIPL23	IRC0_NMIVA23
24 to 31	Reserved		

CHAPTER 9: DMA Channel Activation Factors

This chapter explains the DMA channel activation factors.

1. Factors List

CODE: DMAFACT-S6J3300-E1

1. Factors List

Client number	Peripheral Functions	Enable Setting of IRQ request	Remarks
0 to 8	Reserved	-	
9	WORK FLASH	Unnecessary	*2
10	Ext IRQ 0	Unnecessary	
11	Ext IRQ 1	Unnecessary	
12	Ext IRQ 2	Unnecessary	
13	Ext IRQ 3	Unnecessary	
14	Ext IRQ 4	Unnecessary	
15	Ext IRQ 5	Unnecessary	
16	Ext IRQ 6	Unnecessary	
17	Ext IRQ 7	Unnecessary	
18	Ext IRQ 8	Unnecessary	
19	Ext IRQ 9	Unnecessary	
20	Ext IRQ 10	Unnecessary	
21	Ext IRQ 11	Unnecessary	
22	Ext IRQ 12	Unnecessary	
23	Ext IRQ 13	Unnecessary	
24	Ext IRQ 14	Unnecessary	
25	Ext IRQ 15	Unnecessary	
26	Ext IRQ 16	Unnecessary	
27	Ext IRQ 17	Unnecessary	
28	Ext IRQ 18	Unnecessary	
29	Ext IRQ 19	Unnecessary	
30	Ext IRQ 20	Unnecessary	
31	Ext IRQ 21	Unnecessary	
32	Ext IRQ 22	Unnecessary	
33	Ext IRQ 23	Unnecessary	
34 to 41	Reserved	-	
42	MFS ch.16 RX	Necessary	
43	MFS ch.16 TX	Necessary	
44	MFS ch.17 RX	Necessary	
45	MFS ch.17 TX	Necessary	
46 to 50	Reserved	-	
51	Reload Timer ch.48	Unnecessary	
52	Reload Timer ch.49	Unnecessary	
53 to 57	Reserved	-	
58	DMAC #0 Reload Timer 0	Unnecessary	
59	DMAC #0 Reload Timer 1	Unnecessary	
60	DMAC #0 Reload Timer 2	Unnecessary	
61	DMAC #0 Reload Timer 3	Unnecessary	

Client number	Peripheral Functions	Enable Setting of IRQ request	Remarks
62 to 66	Reserved	-	
67	DDRHSSPI RX	Unnecessary	
68	DDRHSSPI TX	Unnecessary	
69 to 72	Reserved	-	
73	Base Timer ch.0-0	Necessary	*1
74	Base Timer ch.1-0	Necessary	*1
75	Base Timer ch.0-1	Necessary	*1
76	Base Timer ch.1-1	Necessary	*1
77	Base Timer ch.2-0	Necessary	*1
78	Base Timer ch.3-0	Necessary	*1
79	Base Timer ch.2-1	Necessary	*1
80	Base Timer ch.3-1	Necessary	*1
81	Base Timer ch.4-0	Necessary	*1
82	Base Timer ch.5-0	Necessary	*1
83	Base Timer ch.4-1	Necessary	*1
84	Base Timer ch.5-1	Necessary	*1
85	Base Timer ch.6-0	Necessary	*1
86	Base Timer ch.7-0	Necessary	*1
87	Base Timer ch.6-1	Necessary	*1
88	Base Timer ch.7-1	Necessary	*1
89	Base Timer ch.8-0	Necessary	*1
90	Base Timer ch.9-0	Necessary	*1
91	Base Timer ch.8-1	Necessary	*1
92	Base Timer ch.9-1	Necessary	*1
93	Base Timer ch.10-0	Necessary	*1
94	Base Timer ch.11-0	Necessary	*1
95	Base Timer ch.10-1	Necessary	*1
96	Base Timer ch.11-1	Necessary	*1
97	Base Timer ch.12-0	Necessary	*1
98	Base Timer ch.13-0	Necessary	*1
99	Base Timer ch.12-1	Necessary	*1
100	Base Timer ch.13-1	Necessary	*1
101	Base Timer ch.14-0	Necessary	*1
102	Base Timer ch.15-0	Necessary	*1
103	Base Timer ch.14-1	Necessary	*1
104	Base Timer ch.15-1	Necessary	*1
105	Base Timer ch.16-0	Necessary	*1
106	Base Timer ch.17-0	Necessary	*1
107	Base Timer ch.16-1	Necessary	*1
108	Base Timer ch.17-1	Necessary	*1
109	Base Timer ch.18-0	Necessary	*1
110	Base Timer ch.19-0	Necessary	*1

Client number	Peripheral Functions	Enable Setting of IRQ request	Remarks
111	Base Timer ch.18-1	Necessary	*1
112	Base Timer ch.19-1	Necessary	*1
113	Base Timer ch.20-0	Necessary	*1
114	Base Timer ch.21-0	Necessary	*1
115	Base Timer ch.20-1	Necessary	*1
116	Base Timer ch.21-1	Necessary	*1
117	Base Timer ch.22-0	Necessary	*1
118	Base Timer ch.23-0	Necessary	*1
119	Base Timer ch.22-1	Necessary	*1
120	Base Timer ch.23-1	Necessary	*1
121 to 144	Reserved	-	
145	Reload Timer ch.0	Unnecessary	
146	Reload Timer ch.1	Unnecessary	
147 to 160	Reserved	-	
161	Reload Timer ch.16	Unnecessary	
162	Reload Timer ch.17	Unnecessary	
163 to 192	Reserved	-	
193	MFS ch.0 RX	Necessary	
194	MFS ch.0 TX	Necessary	
195	MFS ch.1 RX	Necessary	
196	MFS ch.1 TX	Necessary	
197	MFS ch.2 RX	Necessary	
198	MFS ch.2 TX	Necessary	
199	MFS ch.3 RX	Necessary	
200	MFS ch.3 TX	Necessary	
201	MFS ch.4 RX	Necessary	
202	MFS ch.4 TX	Necessary	
203 to 208	Reserved	-	
209	MFS ch.8 RX	Necessary	
210	MFS ch.8 TX	Necessary	
211	MFS ch.9 RX	Necessary	
212	MFS ch.9 TX	Necessary	
213	MFS ch.10 RX	Necessary	
214	MFS ch.10 TX	Necessary	
215	MFS ch.11 RX	Necessary	
216	MFS ch.11 TX	Necessary	
217	MFS ch.12 RX	Necessary	
218	MFS ch.12 TX	Necessary	
219 to 224	Reserved	-	
225	FRT ch.0 Match	Necessary	*1
226	FRT ch.0 Zero	Necessary	*1
227	FRT ch.1 Match	Necessary	*1

Client number	Peripheral Functions	Enable Setting of IRQ request	Remarks
228	FRT ch.1 Zero	Necessary	*1
229	FRT ch.2 Match	Necessary	*1
230	FRT ch.2 Zero	Necessary	*1
231	FRT ch.3 Match	Necessary	*1
232	FRT ch.3 Zero	Necessary	*1
233	FRT ch.4 Match	Necessary	*1
234	FRT ch.4 Zero	Necessary	*1
235 to 240	Reserved	-	
241	FRT ch.8 Match	Necessary	*1
242	FRT ch.8 Zero	Necessary	*1
243	FRT ch.9 Match	Necessary	*1
244	FRT ch.9 Zero	Necessary	*1
245	FRT ch.10 Match	Necessary	*1
246	FRT ch.10 Zero	Necessary	*1
247 to 256	Reserved	-	
257	IRQ0 of Input Capture 0	Necessary	*1
258	IRQ1 of Input Capture 0	Necessary	*1
259	IRQ0 of Input Capture 1	Necessary	*1
260	IRQ1 of Input Capture 1	Necessary	*1
261	IRQ0 of Input Capture 2	Necessary	*1
262	IRQ1 of Input Capture 2	Necessary	*1
263 to 272	Reserved	-	
273	IRQ0 of Input Capture 8	Necessary	*1
274	IRQ1 of Input Capture 8	Necessary	*1
275	IRQ0 of Input Capture 9	Necessary	*1
276	IRQ1 of Input Capture 9	Necessary	*1
277	IRQ0 of Input Capture 10	Necessary	*1
278	IRQ1 of Input Capture 10	Necessary	*1
279 to 288	Reserved	-	
289	IRQ0 of Output Compare 0	Necessary	*1
290	IRQ1 of Output Compare 0	Necessary	*1
291	IRQ0 of Output Compare 1	Necessary	*1
292	IRQ1 of Output Compare 1	Necessary	*1
*1293	IRQ0 of Output Compare 2	Necessary	*1
294	IRQ1 of Output Compare 2	Necessary	*1
295 to 304	Reserved	-	
305	IRQ0 of Output Compare 8	Necessary	*1
306	IRQ1 of Output Compare 8	Necessary	*1
307	IRQ0 of Output Compare 9	Necessary	*1
308	IRQ1 of Output Compare 9	Necessary	*1
309	IRQ0 of Output Compare 10	Necessary	*1
310	IRQ1 of Output Compare 10	Necessary	*1
311 to 337	Reserved	-	

Client number	Peripheral Functions	Enable Setting of IRQ request	Remarks
338	Sound Generator 0	Necessary	*1
339	Sound Generator 1	Necessary	*1
340	Sound Generator 2	Necessary	*1
341	Sound Generator 3	Necessary	*1
342	Sound Generator 4	Necessary	*1
343 to 355	Reserved	-	
356	ADC12B0-0	Unnecessary	
357	ADC12B0-1	Unnecessary	
358	ADC12B0-2	Unnecessary	
359	ADC12B0-3	Unnecessary	
360	ADC12B1-0	Unnecessary	
361	ADC12B1-1	Unnecessary	
362	ADC12B1-2	Unnecessary	
363	ADC12B1-3	Unnecessary	
364	Mixer[0]	Unnecessary	
365	Mixer[1]	Unnecessary	
366	Mixer[2]	Unnecessary	
367	Mixer[3]	Unnecessary	
368	Mixer[4]	Unnecessary	
369 to 375	Reserved	-	
376	PRGCRC	Unnecessary	
377 to 378	Reserved	-	
379	I2S ch.1 RX	Unnecessary	
380	I2S ch.1 TX	Unnecessary	
381	ARH0 Remote Handler DMA request 0	Unnecessary	
382	ARH0 Remote Handler DMA request 1	Unnecessary	
383	ARH0 Remote Handler DMA request 2	Unnecessary	
384	ARH0 Remote Handler DMA request 3	Unnecessary	
385	ARH0 Remote Handler DMA request 4	Unnecessary	
386	ARH0 Remote Handler DMA request 5	Unnecessary	
387	ARH0 Remote Handler DMA request 6	Unnecessary	
388	ARH0 Remote Handler DMA request 7	Unnecessary	
389	ARH0 Remote Handler DMA request 8	Unnecessary	
390	ARH0 Remote Handler DMA request 9	Unnecessary	
391	ARH0 Remote Handler DMA request 10	Unnecessary	
392	ARH0 Remote Handler DMA request 11	Unnecessary	
393	ARH0 Remote Handler DMA request 12	Unnecessary	
394	ARH0 Remote Handler DMA request 13	Unnecessary	
395	ARH0 Remote Handler DMA request 14	Unnecessary	
396	ARH0 Remote Handler DMA request 15	Unnecessary	
397 to 511	Reserved	-	

*1: The interrupt factor flags of peripheral function are cleared automatically by acceptance of DMA transfer request.

*2: DSTP_ACK function is set by DMAi_CMCICm:BEHSTPACK.

2. Note

This product supports 397 DMA clients.

- 'm' represents client number. m = 8, 9, …, 396
- 'M' represents number of client I/F. M = 397

This product does not support 'm = 397 to 511'. Accessing to those number of DMAi_CMICICm is responded with bus error. See the platform manual and chapter DMA controller in detail.

CHAPTER 10: Port Description

This chapter explains port functions.

Please for details refer to the Device Datasheet.
(The information on this chapter moved to Datasheet.)

CODE: PORT_DESCRIPTION-S6J3300-E1

CHAPTER 11: Port Configuration

This chapter explains the port configuration.

1. Overview
2. Configuration and Block Diagram
3. Operation
4. Registers
5. Configuration Procedure
6. Precautions

CODE: PORTCONFIG-S6J3300-E1

1. Overview

This chapter explains the particular port configuration of the product PKG pins.

The microcontroller has various functions such as general purpose I/O ports, input or output timers, analog input ports and so on. Some of these functions are multiplexed implemented in a pin, and the assignment to a pin is particular for the product.

A port configuration is to determine which function and which input/output direction is applied to a port.

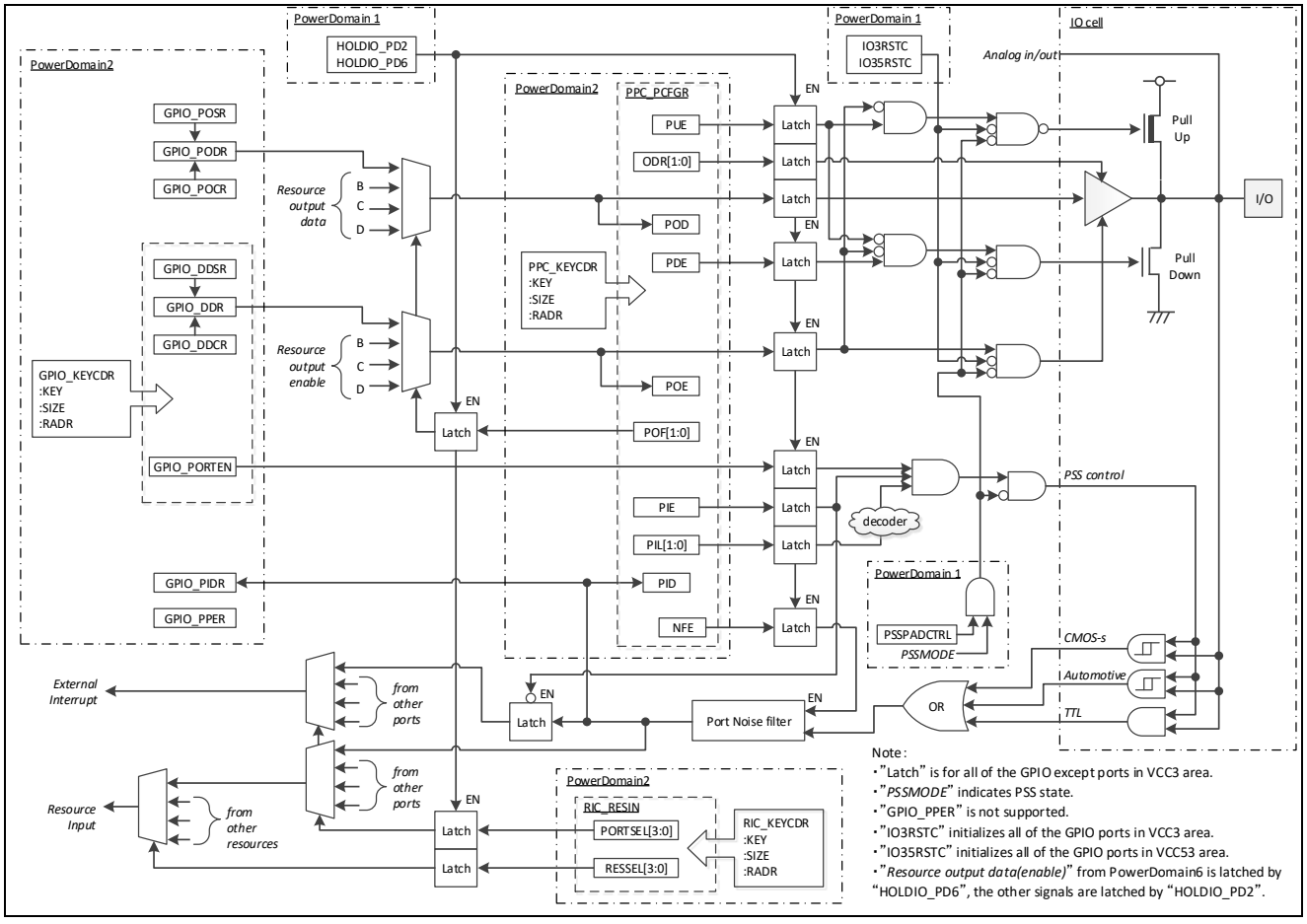
"3. Operation" of this chapter describes following.

- The particular port configurations of the resource input configuration register (RIC)
- The port output function configuration
- The analog I/O setting
- The input level setting
- The output drive capacity setting

See the common information of the registers on the Traveo™ Platform Hardware Manual.

2. Configuration and Block Diagram

Figure 11-1 Block Diagram of Port Function



3. Operation

The relation between configuration and operation is described.

3.1. Resource Input Configuration Module

The resource input configuration module (RIC) is a function to select input from an external or output from another internal resource as resource input. A resource which supports either a port input relocation or a resource inputs from the other resource has its RIC_RESIN register to configure resource input configuration.

Register (Offset)	Resource	RESSEL [3:0] /PORT SEL[3:0]	Source for Resource input							
			0	1	2	3	4	5	6	7
			8	9	10	11	12	13	14	15
RIC_RESIN000 (0x0000)	SIN16	RESSEL (0-7)	-	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P1_08	P4_19	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN001 (0x0002)	SCK16	RESSEL (0-7)	-	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P1_09	P4_21	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN002 (0x0004)	SCL16	RESSEL (0-7)	80 ns noise filter disable	80 ns noise filter enable	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN003 (0x0006)	SDA16	RESSEL (0-7)	80 ns noise filter disable	80 ns noise filter enable	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-

Register (Offset)	Resource	RESSEL [3:0] /PORT SEL[3:0]	Source for Resource input							
			0	1	2	3	4	5	6	7
			8	9	10	11	12	13	14	15
RIC_RESIN004 (0x0008)	MFS16_TRIGGER	RESSEL (0-7)	TOT48	TOT49	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN005 (0x000A)	SCS16	RESSEL (0-7)	-	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P1_11	P4_23	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN007 (0x000E)	SIN17	RESSEL (0-7)	-	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P1_14	P0_09	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN008 (0x0010)	SCK17	RESSEL (0-7)	-	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P1_15	P4_04	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN009 (0x0012)	SCL17	RESSEL (0-7)	80 ns noise filter disable	80 ns noise filter enable	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-

Register (Offset)	Resource	RESSEL [3:0] /PORT SEL[3:0]	Source for Resource input							
			0	1	2	3	4	5	6	7
			8	9	10	11	12	13	14	15
RIC_RESIN010 (0x0014)	SDA17	RESSEL (0-7)	80 ns noise filter disable	80 ns noise filter enable	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN011 (0x0016)	MFS17_TRIGGER	RESSEL (0-7)	TOT48	TOT49	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN012 (0x0018)	SCS17	RESSEL (0-7)	-	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P1_17	P4_03	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN021 (0x002A)	SIN0	RESSEL (0-7)	-	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P1_03	P0_04	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN022 (0x002C)	SCK0	RESSEL (0-7)	-	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P1_04	P0_06	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-

Register (Offset)	Resource	RESSEL [3:0] /PORT SEL[3:0]	Source for Resource input							
			0	1	2	3	4	5	6	7
			8	9	10	11	12	13	14	15
RIC_RESIN023 (0x002E)	SCL0	RESSEL (0-7)	80 ns noise filter disable	80 ns noise filter enable	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN024 (0x0030)	SDA0	RESSEL (0-7)	80 ns noise filter disable	80 ns noise filter enable	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN025 (0x0032)	MFS0_TRIGGER	RESSEL (0-7)	TOT0	TOT1	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN026 (0x0034)	SCS0	RESSEL (0-7)	-	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P1_06	P0_07	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN028 (0x0038)	SIN1	RESSEL (0-7)	-	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P0_01	P3_00	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-

Register (Offset)	Resource	RESSEL [3:0] /PORT SEL[3:0]	Source for Resource input							
			0	1	2	3	4	5	6	7
			8	9	10	11	12	13	14	15
RIC_RESIN029 (0x003A)	SCK1	RESSEL (0-7)	-	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P0_02	P3_01	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN030 (0x003C)	SCL1	RESSEL (0-7)	80 ns noise filter disable	80 ns noise filter enable	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN031 (0x003E)	SDA1	RESSEL (0-7)	80 ns noise filter disable	80 ns noise filter enable	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN032 (0x0040)	MFS1_TRIGGER	RESSEL (0-7)	TOT0	TOT1	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN033 (0x0042)	SCS1	RESSEL (0-7)	-	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P0_04	P3_03	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-

Register (Offset)	Resource	RESSEL [3:0] /PORTSEL[3:0]	Source for Resource input							
			0	1	2	3	4	5	6	7
			8	9	10	11	12	13	14	15
RIC_RESIN035 (0x0046)	SIN2	RESSEL (0-7)	-	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P0_21	P4_30	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN036 (0x0048)	SCK2	RESSEL (0-7)	-	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P0_22	P4_31	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN039 (0x004E)	MFS2_TRIGGER	RESSEL (0-7)	TOT0	TOT1	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN040 (0x0050)	SCS2	RESSEL (0-7)	-	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P0_24	P3_25	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN042 (0x0054)	SIN3	RESSEL (0-7)	-	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P0_28	P3_28	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN043 (0x0056)	SCK3	RESSEL (0-7)	-	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P0_29	P3_29	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-

Register (Offset)	Resource	RESSEL [3:0] /PORT SEL[3:0]	Source for Resource input							
			0	1	2	3	4	5	6	7
			8	9	10	11	12	13	14	15
RIC_RESIN046 (0x005C)	MFS3_TRIGGER	RESSEL (0-7)	TOT0	TOT1	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN047 (0x005E)	SCS3	RESSEL (0-7)	-	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P0_31	P3_31	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN049 (0x0062)	SIN4	RESSEL (0-7)	-	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P2_13	P3_05	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN050 (0x0064)	SCK4	RESSEL (0-7)	-	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P2_14	P0_14	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN051 (0x0066)	SCL4	RESSEL (0-7)	80 ns noise filter disable	80 ns noise filter enable	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-

Register (Offset)	Resource	RESSEL [3:0] /PORT SEL[3:0]	Source for Resource input							
			0	1	2	3	4	5	6	7
			8	9	10	11	12	13	14	15
RIC_RESIN052 (0x0068)	SDA4	RESSEL (0-7)	80 ns noise filter disable	80 ns noise filter enable	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN053 (0x006A)	MFS4_TRIGGER	RESSEL (0-7)	TOT0	TOT1	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN054 (0x006C)	SCS4	RESSEL (0-7)	-	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P2_16	P0_15	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN077 (0x009A)	SIN8	RESSEL (0-7)	-	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P1_19	P3_08	P0_27	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN078 (0x009C)	SCK8	RESSEL (0-7)	-	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P1_20	P3_09	P0_28	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-

Register (Offset)	Resource	RESSEL [3:0] /PORT SEL[3:0]	Source for Resource input							
			0	1	2	3	4	5	6	7
			8	9	10	11	12	13	14	15
RIC_RESIN079 (0x009E)	SCL8	RESSEL (0-7)	80 ns noise filter disable	80 ns noise filter enable	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN080 (0x00A0)	SDA8	RESSEL (0-7)	80 ns noise filter disable	80 ns noise filter enable	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN081 (0x00A2)	MFS8_TRIGGER	RESSEL (0-7)	TOT16	TOT17	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN082 (0x00A4)	SCS8	RESSEL (0-7)	-	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P1_22	P3_11	P0_30	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN084 (0x00A8)	SIN9	RESSEL (0-7)	-	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P1_23	P3_18	P0_21	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-

Register (Offset)	Resource	RESSEL [3:0] /PORT SEL[3:0]	Source for Resource input							
			0	1	2	3	4	5	6	7
			8	9	10	11	12	13	14	15
RIC_RESIN085 (0x00AA)	SCK9	RESSEL (0-7)	-	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P1_24	P3_19	P0_23	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN086 (0x00AC)	SCL9	RESSEL (0-7)	80 ns noise filter disable	80 ns noise filter enable	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN087 (0x00AE)	SDA9	RESSEL (0-7)	80 ns noise filter disable	80 ns noise filter enable	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN088 (0x00B0)	MFS9_TRIGGER	RESSEL (0-7)	TOT16	TOT17	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN089 (0x00B2)	SCS9	RESSEL (0-7)	-	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P1_26	P3_21	P0_24	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-

Register (Offset)	Resource	RESSEL [3:0] /PORT SEL[3:0]	Source for Resource input							
			0	1	2	3	4	5	6	7
			8	9	10	11	12	13	14	15
RIC_RESIN091 (0x00B6)	SIN10	RESSEL (0-7)	-	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P1_28	P3_12	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN092 (0x00B8)	SCK10	RESSEL (0-7)	-	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P1_29	P3_13	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN093 (0x00BA)	SCL10	RESSEL (0-7)	80 ns noise filter disable	80 ns noise filter enable	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN094 (0x00BC)	SDA10	RESSEL (0-7)	80 ns noise filter disable	80 ns noise filter enable	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN095 (0x00BE)	MFS10_TRIGGER	RESSEL (0-7)	TOT16	TOT17	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-

Register (Offset)	Resource	RESSEL [3:0] /PORT SEL[3:0]	Source for Resource input							
			0	1	2	3	4	5	6	7
			8	9	10	11	12	13	14	15
RIC_RESIN096 (0x00C0)	SCS10	RESSEL (0-7)	-	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P1_31	P3_15	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN100 (0x00C8)	SCL11	RESSEL (0-7)	80 ns noise filter disable	80 ns noise filter enable	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN101 (0x00CA)	SDA11	RESSEL (0-7)	80 ns noise filter disable	80 ns noise filter enable	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN102 (0x00CC)	MFS11_TRIGGER	RESSEL (0-7)	TOT16	TOT17	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN105 (0x00D2)	SIN12	RESSEL (0-7)	-	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P2_05	P4_18	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-

Register (Offset)	Resource	RESSEL [3:0] /PORT SEL[3:0]	Source for Resource input							
			0	1	2	3	4	5	6	7
			8	9	10	11	12	13	14	15
RIC_RESIN106 (0x00D4)	SCK12	RESSEL (0-7)	-	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P2_06	P1_15	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN107 (0x00D6)	SCL12	RESSEL (0-7)	80 ns noise filter disable	80 ns noise filter enable	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN108 (0x00D8)	SDA12	RESSEL (0-7)	80 ns noise filter disable	80 ns noise filter enable	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN109 (0x00DA)	MFS12_TRIGGER	RESSEL (0-7)	TOT16	TOT17	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN110 (0x00DC)	SCS12	RESSEL (0-7)	-	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P2_08	P3_23	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-

Register (Offset)	Resource	RESSEL [3:0] /PORT SEL[3:0]	Source for Resource input							
			0	1	2	3	4	5	6	7
			8	9	10	11	12	13	14	15
RIC_RESIN133 (0x010A)	RX5	RESSEL (0-7)	PORT_PIN	MCAN5_PIN_AND_TX	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P1_17	P3_19	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN134 (0x010C)	RX6	RESSEL (0-7)	PORT_PIN	MCAN6_PIN_AND_TX	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P1_21	P3_22	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN135 (0x010E)	RX7	RESSEL (0-7)	PORT_PIN	MCAN7_PIN_AND_TX	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P4_22	P2_18	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN136 (0x0110)	RX0	RESSEL (0-7)	PORT_PIN	MCAN0_PIN_AND_TX	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P1_05	P3_09	P4_00	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN137 (0x0112)	RX1	RESSEL (0-7)	PORT_PIN	MCAN1_PIN_AND_TX	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P1_07	P3_11	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-

Register (Offset)	Resource	RESSEL [3:0] /PORT SEL[3:0]	Source for Resource input							
			0	1	2	3	4	5	6	7
			8	9	10	11	12	13	14	15
RIC_RESIN138 (0x0114)	RX2	RESSEL (0-7)	PORT_PIN	MCAN2_PIN_AND_TX	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P1_11	P3_14	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN139 (0x0116)	RX3	RESSEL (0-7)	PORT_PIN	MCAN3_PIN_AND_TX	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P1_13	P3_16	P4_03	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN140 (0x0118)	RX4	RESSEL (0-7)	PORT_PIN	MCAN4_PIN_AND_TX	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN141 (0x011A)	TIN48	RESSEL (0-7)	PORT_PIN	TOT49	RLT49_UFSET	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P1_13	P3_16	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN142 (0x011C)	TIN49	RESSEL (0-7)	PORT_PIN	TOT48	RLT48_UFSET	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P1_15	P3_20	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-

Register (Offset)	Resource	RESSEL [3:0] /PORT SEL[3:0]	Source for Resource input							
			0	1	2	3	4	5	6	7
			8	9	10	11	12	13	14	15
RIC_RESIN144 (0x0120)	TIN0	RESSEL (0-7)	PORT_PIN	TOT1	RLT1_UFSET	-	PPG0_TOUT0	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P1_03	P3_08	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN145 (0x0122)	TIN1	RESSEL (0-7)	PORT_PIN	TOT0	RLT0_UFSET	-	PPG1_TOUT0	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P1_05	P3_10	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN160 (0x0140)	TIN16	RESSEL (0-7)	PORT_PIN	TOT17	RLT17_UFSET	-	PPG6_TOUT0	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P1_07	P3_12	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN161 (0x0142)	TIN17	RESSEL (0-7)	PORT_PIN	TOT16	RLT16_UFSET	-	PPG7_TOUT0	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P1_11	P3_14	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN192 (0x0180)	EINT0	RESSEL (0-7)	-	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P2_13	P0_00	P0_08	P0_20	P3_17	P2_01	P2_16	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN193 (0x0182)	EINT1	RESSEL (0-7)	-	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P0_01	P3_00	P0_09	P0_22	P4_16	P4_27	P2_17	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-

Register (Offset)	Resource	RESSEL [3:0] /PORT SEL[3:0]	Source for Resource input							
			0	1	2	3	4	5	6	7
			8	9	10	11	12	13	14	15
RIC_RESIN194 (0x0184)	EINT2	RESSEL (0-7)	-	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P0_21	P4_30	P4_03	P0_23	P3_20	P2_02	P2_18	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN195 (0x0186)	EINT3	RESSEL (0-7)	-	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P0_28	P3_28	P4_04	P0_24	P3_21	P2_03	P2_19	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN196 (0x0188)	EINT4	RESSEL (0-7)	-	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P1_03	P0_02	P4_05	P0_25	P3_23	P2_04	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN197 (0x018A)	EINT5	RESSEL (0-7)	-	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P1_04	P0_03	P0_10	P0_26	P4_17	P4_28	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN198 (0x018C)	EINT6	RESSEL (0-7)	-	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P1_05	P3_09	P0_11	P0_27	P4_18	P2_06	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN199 (0x018E)	EINT7	RESSEL (0-7)	-	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P1_06	P4_00	P0_12	P0_29	P4_20	P2_07	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-

Register (Offset)	Resource	RESSEL [3:0] /PORTSEL[3:0]	Source for Resource input							
			0	1	2	3	4	5	6	7
			8	9	10	11	12	13	14	15
RIC_RESIN200 (0x0190)	EINT8	RESSEL (0-7)	-	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P1_07	P3_11	P0_13	P0_30	P4_21	P4_29	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN201 (0x0192)	EINT9	RESSEL (0-7)	-	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P1_08	P4_01	P3_01	P0_31	P1_16	P2_08	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN202 (0x0194)	EINT10	RESSEL (0-7)	-	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P1_09	P4_02	P3_02	P1_00	P4_23	P4_31	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN203 (0x0196)	EINT11	RESSEL (0-7)	-	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P1_10	P0_04	P3_03	P1_01	P1_18	P3_24	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN204 (0x0198)	EINT12	RESSEL (0-7)	-	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P1_11	P3_14	P3_04	P1_02	P1_20	P3_25	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN205 (0x019A)	EINT13	RESSEL (0-7)	-	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P1_12	P0_05	P4_06	P4_08	P1_22	P3_26	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-

Register (Offset)	Resource	RESSEL [3:0] /PORT SEL[3:0]	Source for Resource input							
			0	1	2	3	4	5	6	7
			8	9	10	11	12	13	14	15
RIC_RESIN206 (0x019C)	EINT14	RESSEL (0-7)	-	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P1_13	P3_16	P4_07	P4_09	P1_24	P3_27	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN207 (0x019E)	EINT15	RESSEL (0-7)	-	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P1_14	P0_06	P3_05	P4_10	P1_25	P2_09	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN208 (0x01A0)	EINT16	RESSEL (0-7)	-	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P1_15	P4_19	P3_06	P4_11	P4_24	P3_29	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN209 (0x01A2)	EINT17	RESSEL (0-7)	-	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P1_17	P3_19	P0_14	P3_10	P1_26	P3_30	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN210 (0x01A4)	EINT18	RESSEL (0-7)	-	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P1_19	P3_08	P0_15	P3_13	P1_27	P3_31	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN211 (0x01A6)	EINT19	RESSEL (0-7)	-	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P1_21	P3_22	P0_16	P3_15	P4_25	P2_10	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-

Register (Offset)	Resource	RESSEL [3:0] /PORT SEL[3:0]	Source for Resource input							
			0	1	2	3	4	5	6	7
			8	9	10	11	12	13	14	15
RIC_RESIN212 (0x01A8)	EINT20	RESSEL (0-7)	-	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P1_23	P3_18	P3_07	P4_12	P1_29	P2_11	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN213 (0x01AA)	EINT21	RESSEL (0-7)	-	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P1_28	P3_12	P0_17	P4_13	P1_30	P2_12	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN214 (0x01AC)	EINT22	RESSEL (0-7)	-	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P2_00	P4_22	P0_18	P4_14	P1_31	P2_14	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN215 (0x01AE)	EINT23	RESSEL (0-7)	-	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P2_05	P0_07	P0_19	P4_15	P4_26	P2_15	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN216 (0x01B0)	TEXT0	RESSEL (0-7)	PORT_PIN	TOT0	TOT1	PPG0_TOUT2	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN217 (0x01B2)	TEXT1	RESSEL (0-7)	PORT_PIN	TOT0	TOT1	PPG1_TOUT2	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-

Register (Offset)	Resource	RESSEL [3:0] /PORT SEL[3:0]	Source for Resource input							
			0	1	2	3	4	5	6	7
			8	9	10	11	12	13	14	15
RIC_RESIN218 (0x01B4)	TEXT2	RESSEL (0-7)	PORT_PIN	TOT0	TOT1	PPG2_TOUT2	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN219 (0x01B6)	TEXT3	RESSEL (0-7)	PORT_PIN	TOT0	TOT1	PPG3_TOUT2	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN220 (0x01B8)	TEXT4	RESSEL (0-7)	PORT_PIN	TOT0	TOT1	PPG4_TOUT2	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN224 (0x01C0)	TEXT8	RESSEL (0-7)	PORT_PIN	RLT0_UFSET	RLT16_UFSET	PPG6_TOUT2	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN225 (0x01C2)	TEXT9	RESSEL (0-7)	PORT_PIN	RLT0_UFSET	RLT16_UFSET	PPG7_TOUT2	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN226 (0x01C4)	TEXT10	RESSEL (0-7)	PORT_PIN	RLT0_UFSET	RLT16_UFSET	PPG8_TOUT2	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-

Register (Offset)	Resource	RESSEL [3:0] /PORT SEL[3:0]	Source for Resource input							
			0	1	2	3	4	5	6	7
			8	9	10	11	12	13	14	15
IN232 (0x01D0)	OCU0_CK0	RESSEL (0-7)	FRT0	FRT1	FRT2	FRT3	FRT4	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
	OCU0_CK1	RESSEL (0-7)	FRT0	FRT1	FRT2	FRT3	FRT4	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
	OCU0_DOWNB0	RESSEL (0-7)	FRT0_DOWNB	FRT1_DOWNB	FRT2_DOWNB	FRT3_DOWNB	FRT4_DOWNB	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
	OCU0_DOWNB1	RESSEL (0-7)	FRT0_DOWNB	FRT1_DOWNB	FRT2_DOWNB	FRT3_DOWNB	FRT4_DOWNB	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
	OCU0_FCMD0	RESSEL (0-7)	FRT0_FCMD	FRT1_FCMD	FRT2_FCMD	FRT3_FCMD	FRT4_FCMD	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-

Register (Offset)	Resource	RESSEL [3:0] /PORT SEL[3:0]	Source for Resource input							
			0	1	2	3	4	5	6	7
			8	9	10	11	12	13	14	15
IN232 (0x01D0)	OCU0_FCMD1	RESSEL (0-7)	FRT0_FCMD	FRT1_FCMD	FRT2_FCMD	FRT3_FCMD	FRT4_FCMD	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
	OCU0_MTSF0	RESSEL (0-7)	FRT0_MTSF	FRT1_MTSF	FRT2_MTSF	FRT3_MTSF	FRT4_MTSF	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
	OCU0_MTSF1	RESSEL (0-7)	FRT0_MTSF	FRT1_MTSF	FRT2_MTSF	FRT3_MTSF	FRT4_MTSF	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
	OCU0_T0[31:0]	RESSEL (0-7)	FRT0_T[31:0]	FRT1_T[31:0]	FRT2_T[31:0]	FRT3_T[31:0]	FRT4_T[31:0]	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
	OCU0_T1[31:0]	RESSEL (0-7)	FRT0_T[31:0]	FRT1_T[31:0]	FRT2_T[31:0]	FRT3_T[31:0]	FRT4_T[31:0]	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-

Register (Offset)	Resource	RESSEL [3:0] /PORT SEL[3:0]	Source for Resource input							
			0	1	2	3	4	5	6	7
			8	9	10	11	12	13	14	15
IN232 (0x01D0)	OCU0_ZTSF0	RESSEL (0-7)	FRT0_ZTSF	FRT1_ZTSF	FRT2_ZTSF	FRT3_ZTSF	FRT4_ZTSF	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
	OCU0_ZTSF1	RESSEL (0-7)	FRT0_ZTSF	FRT1_ZTSF	FRT2_ZTSF	FRT3_ZTSF	FRT4_ZTSF	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN233 (0x01D2)	OCU0_MOD0	RESSEL (0-7)	set 1	set 0	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN234 (0x01D4)	OCU0_MOD1	RESSEL (0-7)	set 1	set 0	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN235 (0x01D6)	OCU1_CK0	RESSEL (0-7)	FRT0	FRT1	FRT2	FRT3	FRT4	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-

Register (Offset)	Resource	RESSEL [3:0] /PORT SEL[3:0]	Source for Resource input							
			0	1	2	3	4	5	6	7
			8	9	10	11	12	13	14	15
RIC_RESIN235 (0x01D6)	OCU1_CK1	RESSEL (0-7)	FRT0	FRT1	FRT2	FRT3	FRT4	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
	OCU1_DOWNB0	RESSEL (0-7)	FRT0_DOWNB	FRT1_DOWNB	FRT2_DOWNB	FRT3_DOWNB	FRT4_DOWNB	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
	OCU1_DOWNB1	RESSEL (0-7)	FRT0_DOWNB	FRT1_DOWNB	FRT2_DOWNB	FRT3_DOWNB	FRT4_DOWNB	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
	OCU1_FCMD0	RESSEL (0-7)	FRT0_FCMD	FRT1_FCMD	FRT2_FCMD	FRT3_FCMD	FRT4_FCMD	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
	OCU1_FCMD1	RESSEL (0-7)	FRT0_FCMD	FRT1_FCMD	FRT2_FCMD	FRT3_FCMD	FRT4_FCMD	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-

Register (Offset)	Resource	RESSEL [3:0] /PORT SEL[3:0]	Source for Resource input							
			0	1	2	3	4	5	6	7
			8	9	10	11	12	13	14	15
RIC_RESIN235 (0x01D6)	OCU1_MTSF0	RESSEL (0-7)	FRT0_MTSF	FRT1_MTSF	FRT2_MTSF	FRT3_MTSF	FRT4_MTSF	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
	OCU1_MTSF1	RESSEL (0-7)	FRT0_MTSF	FRT1_MTSF	FRT2_MTSF	FRT3_MTSF	FRT4_MTSF	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
	OCU1_T0[31:0]	RESSEL (0-7)	FRT0_T[31:0]	FRT1_T[31:0]	FRT2_T[31:0]	FRT3_T[31:0]	FRT4_T[31:0]	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
	OCU1_T1[31:0]	RESSEL (0-7)	FRT0_T[31:0]	FRT1_T[31:0]	FRT2_T[31:0]	FRT3_T[31:0]	FRT4_T[31:0]	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
	OCU1_ZTSF0	RESSEL (0-7)	FRT0_ZTSF	FRT1_ZTSF	FRT2_ZTSF	FRT3_ZTSF	FRT4_ZTSF	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
	OCU1_ZTSF1	RESSEL (0-7)	FRT0_ZTSF	FRT1_ZTSF	FRT2_ZTSF	FRT3_ZTSF	FRT4_ZTSF	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-

Register (Offset)	Resource	RESSEL [3:0] /PORT SEL[3:0]	Source for Resource input							
			0	1	2	3	4	5	6	7
			8	9	10	11	12	13	14	15
RIC_RESIN236 (0x01D8)	OCU1_MOD0	RESSEL (0-7)	set 1	set 0	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN237 (0x01DA)	OCU1_MOD1	RESSEL (0-7)	set 1	set 0	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN238 (0x01DC)	OCU2_CK0	RESSEL (0-7)	FRT0	FRT1	FRT2	FRT3	FRT4	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
	OCU2_CK1	RESSEL (0-7)	FRT0	FRT1	FRT2	FRT3	FRT4	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
	OCU2_DOWNB0	RESSEL (0-7)	FRT0_DOWNB	FRT1_DOWNB	FRT2_DOWNB	FRT3_DOWNB	FRT4_DOWNB	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-

Register (Offset)	Resource	RESSEL [3:0] /PORT SEL[3:0]	Source for Resource input							
			0	1	2	3	4	5	6	7
			8	9	10	11	12	13	14	15
RIC_RESIN238 (0x01DC)	OCU2_DOWNB1	RESSEL (0-7)	FRT0_DOWNB	FRT1_DOWNB	FRT2_DOWNB	FRT3_DOWNB	FRT4_DOWNB	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
	OCU2_FCMD0	RESSEL (0-7)	FRT0_FCMD	FRT1_FCMD	FRT2_FCMD	FRT3_FCMD	FRT4_FCMD	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
	OCU2_FCMD1	RESSEL (0-7)	FRT0_FCMD	FRT1_FCMD	FRT2_FCMD	FRT3_FCMD	FRT4_FCMD	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
	OCU2_MTSF0	RESSEL (0-7)	FRT0_MTSF	FRT1_MTSF	FRT2_MTSF	FRT3_MTSF	FRT4_MTSF	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
	OCU2_MTSF1	RESSEL (0-7)	FRT0_MTSF	FRT1_MTSF	FRT2_MTSF	FRT3_MTSF	FRT4_MTSF	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-

Register (Offset)	Resource	RESSEL [3:0] /PORT SEL[3:0]	Source for Resource input							
			0	1	2	3	4	5	6	7
			8	9	10	11	12	13	14	15
RIC_RESIN238 (0x01DC)	OCU2_T0[31:0]	RESSEL (0-7)	FRT0_T[31:0]	FRT1_T[31:0]	FRT2_T[31:0]	FRT3_T[31:0]	FRT4_T[31:0]	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
	OCU2_T1[31:0]	RESSEL (0-7)	FRT0_T[31:0]	FRT1_T[31:0]	FRT2_T[31:0]	FRT3_T[31:0]	FRT4_T[31:0]	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
	OCU2_ZTSF0	RESSEL (0-7)	FRT0_ZTSF	FRT1_ZTSF	FRT2_ZTSF	FRT3_ZTSF	FRT4_ZTSF	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
	OCU2_ZTSF1	RESSEL (0-7)	FRT0_ZTSF	FRT1_ZTSF	FRT2_ZTSF	FRT3_ZTSF	FRT4_ZTSF	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN239 (0x01DE)	OCU2_MOD0	RESSEL (0-7)	set 1	set 0	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN240 (0x01E0)	OCU2_MOD1	RESSEL (0-7)	set 1	set 0	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-

Register (Offset)	Resource	RESSEL [3:0] /PORT SEL[3:0]	Source for Resource input							
			0	1	2	3	4	5	6	7
			8	9	10	11	12	13	14	15
RIC_RESIN256 (0x0200)	OCU8_CK0	RESSEL (0-7)	FRT8	FRT9	FRT10	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
	OCU8_CK1	RESSEL (0-7)	FRT8	FRT9	FRT10	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
	OCU8_DOWNB0	RESSEL (0-7)	FRT8_DOWNB	FRT9_DOWNB	FRT10_DOWNB	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
	OCU8_DOWNB1	RESSEL (0-7)	FRT8_DOWNB	FRT9_DOWNB	FRT10_DOWNB	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
	OCU8_FCMD0	RESSEL (0-7)	FRT8_FCMD	FRT9_FCMD	FRT10_FCMD	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-

Register (Offset)	Resource	RESSEL [3:0] /PORT SEL[3:0]	Source for Resource input							
			0	1	2	3	4	5	6	7
			8	9	10	11	12	13	14	15
RIC_RESIN256 (0x0200)	OCU8_FCMD1	RESSEL (0-7)	FRT8_FCMD	FRT9_FCMD	FRT10_FCMD	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
	OCU8_MTSF0	RESSEL (0-7)	FRT8_MTSF	FRT9_MTSF	FRT10_MTSF	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
	OCU8_MTSF1	RESSEL (0-7)	FRT8_MTSF	FRT9_MTSF	FRT10_MTSF	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
	OCU8_T0[31:0]	RESSEL (0-7)	FRT8_T[31:0]	FRT9_T[31:0]	FRT10_T[31:0]	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
	OCU8_T1[31:0]	RESSEL (0-7)	FRT8_T[31:0]	FRT9_T[31:0]	FRT10_T[31:0]	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-

Register (Offset)	Resource	RESSEL [3:0] /PORT SEL[3:0]	Source for Resource input							
			0	1	2	3	4	5	6	7
			8	9	10	11	12	13	14	15
RIC_RESIN256 (0x0200)	OCU8_ZTSF0	RESSEL (0-7)	FRT8_ZTSF	FRT9_ZTSF	FRT10_ZTSF	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
	OCU8_ZTSF1	RESSEL (0-7)	FRT8_ZTSF	FRT9_ZTSF	FRT10_ZTSF	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN257 (0x0202)	OCU8_MOD0	RESSEL (0-7)	set 1	set 0	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN258 (0x0204)	OCU8_MOD1	RESSEL (0-7)	set 1	set 0	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN259 (0x0206)	OCU9_CK0	RESSEL (0-7)	FRT8	FRT9	FRT10	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
	OCU9_CK1	RESSEL (0-7)	FRT8	FRT9	FRT10	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-

Register (Offset)	Resource	RESSEL [3:0] /PORT SEL[3:0]	Source for Resource input							
			0	1	2	3	4	5	6	7
			8	9	10	11	12	13	14	15
RIC_RESIN259 (0x0206)	OCU9_DOWNB0	RESSEL (0-7)	FRT8_DOWNB	FRT9_DOWNB	FRT10_DOWNB	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
	OCU9_DOWNB1	RESSEL (0-7)	FRT8_DOWNB	FRT9_DOWNB	FRT10_DOWNB	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
	OCU9_FCMD0	RESSEL (0-7)	FRT8_FCMD	FRT9_FCMD	FRT10_FCMD	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
	OCU9_FCMD1	RESSEL (0-7)	FRT8_FCMD	FRT9_FCMD	FRT10_FCMD	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
	OCU9_MTSF0	RESSEL (0-7)	FRT8_MTSF	FRT9_MTSF	FRT10_MTSF	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
	OCU9_MTSF1	RESSEL (0-7)	FRT8_MTSF	FRT9_MTSF	FRT10_MTSF	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-

Register (Offset)	Resource	RESSEL [3:0] /PORT SEL[3:0]	Source for Resource input							
			0	1	2	3	4	5	6	7
			8	9	10	11	12	13	14	15
RIC_RESIN259 (0x0206)	OCU9_T0[31:0]	RESSEL (0-7)	FRT8_T[31:0]	FRT9_T[31:0]	FRT10_T[31:0]	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
	OCU9_T1[31:0]	RESSEL (0-7)	FRT8_T[31:0]	FRT9_T[31:0]	FRT10_T[31:0]	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
	OCU9_ZTSF0	RESSEL (0-7)	FRT8_ZTSF	FRT9_ZTSF	FRT10_ZTSF	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
	OCU9_ZTSF1	RESSEL (0-7)	FRT8_ZTSF	FRT9_ZTSF	FRT10_ZTSF	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN260 (0x0208)	OCU9_MOD0	RESSEL (0-7)	set 1	set 0	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN261 (0x020A)	OCU9_MOD1	RESSEL (0-7)	set 1	set 0	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-

Register (Offset)	Resource	RESSEL [3:0] /PORT SEL[3:0]	Source for Resource input							
			0	1	2	3	4	5	6	7
			8	9	10	11	12	13	14	15
RIC_RESIN262 (0x020C)	OCU10_CK0	RESSEL (0-7)	FRT8	FRT9	FRT10	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
	OCU10_CK1	RESSEL (0-7)	FRT8	FRT9	FRT10	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
	OCU10_DOWNB0	RESSEL (0-7)	FRT8_DOWNB	FRT9_DOWNB	FRT10_DOWNB	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
	OCU10_DOWNB1	RESSEL (0-7)	FRT8_DOWNB	FRT9_DOWNB	FRT10_DOWNB	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
	OCU10_FCMD0	RESSEL (0-7)	FRT8_FCMD	FRT9_FCMD	FRT10_FCMD	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
	OCU10_FCMD1	RESSEL (0-7)	FRT8_FCMD	FRT9_FCMD	FRT10_FCMD	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-

Register (Offset)	Resource	RESSEL [3:0] /PORT SEL[3:0]	Source for Resource input							
			0	1	2	3	4	5	6	7
			8	9	10	11	12	13	14	15
RIC_RESIN262 (0x020C)	OCU10_MTSF0	RESSEL (0-7)	FRT8_MTSF	FRT9_MTSF	FRT10_MTSF	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
	OCU10_MTSF1	RESSEL (0-7)	FRT8_MTSF	FRT9_MTSF	FRT10_MTSF	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
	OCU10_T0[31:0]	RESSEL (0-7)	FRT8_T[31:0]	FRT9_T[31:0]	FRT10_T[31:0]	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
	OCU10_T1[31:0]	RESSEL (0-7)	FRT8_T[31:0]	FRT9_T[31:0]	FRT10_T[31:0]	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
	OCU10_ZTSF0	RESSEL (0-7)	FRT8_ZTSF	FRT9_ZTSF	FRT10_ZTSF	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
	OCU10_ZTSF1	RESSEL (0-7)	FRT8_ZTSF	FRT9_ZTSF	FRT10_ZTSF	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-

Register (Offset)	Resource	RESSEL [3:0] /PORT SEL[3:0]	Source for Resource input							
			0	1	2	3	4	5	6	7
			8	9	10	11	12	13	14	15
RIC_RESIN263 (0x020E)	OCU10_MOD0	RESSEL (0-7)	set 1	set 0	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN264 (0x0210)	OCU10_MOD1	RESSEL (0-7)	set 1	set 0	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN280 (0x0230)	ICU0_IN0	RESSEL (0-7)	PORT_PIN	MFS0_LSYN	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P1_03	P3_08	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN281 (0x0232)	ICU0_IN1	RESSEL (0-7)	PORT_PIN	MFS1_LSYN	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P1_04	P3_09	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN282 (0x0234)	ICU0_T0[31:0]	RESSEL (0-7)	FRT0_T[31:0]	FRT1_T[31:0]	FRT2_T[31:0]	FRT3_T[31:0]	FRT4_T[31:0]	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
	ICU0_T1[31:0]	RESSEL (0-7)	FRT0_T[31:0]	FRT1_T[31:0]	FRT2_T[31:0]	FRT3_T[31:0]	FRT4_T[31:0]	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-

Register (Offset)	Resource	RESSEL [3:0] /PORT SEL[3:0]	Source for Resource input							
			0	1	2	3	4	5	6	7
			8	9	10	11	12	13	14	15
RIC_RESIN283 (0x0236)	ICU1_IN0	RESSEL (0-7)	PORT_PIN	MFS2_LSYN	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P1_05	P3_10	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN284 (0x0238)	ICU1_IN1	RESSEL (0-7)	PORT_PIN	MFS3_LSYN	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P1_06	P3_11	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN285 (0x023A)	ICU1_T0[31:0]	RESSEL (0-7)	FRT0_T[31:0]	FRT1_T[31:0]	FRT2_T[31:0]	FRT3_T[31:0]	FRT4_T[31:0]	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
	ICU1_T1[31:0]	RESSEL (0-7)	FRT0_T[31:0]	FRT1_T[31:0]	FRT2_T[31:0]	FRT3_T[31:0]	FRT4_T[31:0]	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN286 (0x023C)	ICU2_IN0	RESSEL (0-7)	PORT_PIN	MFS4_LSYN	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P1_07	P3_12	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN287 (0x023E)	ICU2_IN1	RESSEL (0-7)	PORT_PIN	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P1_08	P3_13	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-

Register (Offset)	Resource	RESSEL [3:0] /PORT SEL[3:0]	Source for Resource input							
			0	1	2	3	4	5	6	7
			8	9	10	11	12	13	14	15
RIC_RESIN288 (0x0240)	ICU2_T0[31:0]	RESSEL (0-7)	FRT0_T[31:0]	FRT1_T[31:0]	FRT2_T[31:0]	FRT3_T[31:0]	FRT4_T[31:0]	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
	ICU2_T1[31:0]	RESSEL (0-7)	FRT0_T[31:0]	FRT1_T[31:0]	FRT2_T[31:0]	FRT3_T[31:0]	FRT4_T[31:0]	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN304 (0x0260)	ICU8_IN0	RESSEL (0-7)	PORT_PIN	MFS8_LSYN	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P1_09	P3_14	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN305 (0x0262)	ICU8_IN1	RESSEL (0-7)	PORT_PIN	MFS9_LSYN	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P1_10	P3_15	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN306 (0x0264)	ICU8_T0[31:0]	RESSEL (0-7)	FRT8_T[31:0]	FRT9_T[31:0]	FRT10_T[31:0]	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
	ICU8_T1[31:0]	RESSEL (0-7)	FRT8_T[31:0]	FRT9_T[31:0]	FRT10_T[31:0]	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-

Register (Offset)	Resource	RESSEL [3:0] /PORT SEL[3:0]	Source for Resource input							
			0	1	2	3	4	5	6	7
			8	9	10	11	12	13	14	15
RIC_RESIN307 (0x0266)	ICU9_IN0	RESSEL (0-7)	PORT_PIN	MFS10_LSYN	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P1_11	P3_16	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN308 (0x0268)	ICU9_IN1	RESSEL (0-7)	PORT_PIN	MFS11_LSYN	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P1_12	P3_17	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN309 (0x026A)	ICU9_T0[31:0]	RESSEL (0-7)	FRT8_T[31:0]	FRT9_T[31:0]	FRT10_T[31:0]	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
	ICU9_T1[31:0]	RESSEL (0-7)	FRT8_T[31:0]	FRT9_T[31:0]	FRT10_T[31:0]	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN310 (0x026C)	ICU10_IN0	RESSEL (0-7)	PORT_PIN	MFS12_LSYN	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P1_14	P3_18	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN311 (0x026E)	ICU10_IN1	RESSEL (0-7)	PORT_PIN	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P1_15	P3_19	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-

Register (Offset)	Resource	RESSEL [3:0] /PORT SEL[3:0]	Source for Resource input							
			0	1	2	3	4	5	6	7
			8	9	10	11	12	13	14	15
RIC_RESIN312 (0x0270)	ICU10_T0[31:0]	RESSEL (0-7)	FRT8_T[31:0]	FRT9_T[31:0]	FRT10_T[31:0]	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
	ICU10_T1[31:0]	RESSEL (0-7)	FRT8_T[31:0]	FRT9_T[31:0]	FRT10_T[31:0]	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN352 (0x02C0)	AIN8	RESSEL (0-7)	PORT_PIN	TOT0	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN353 (0x02C2)	BIN8	RESSEL (0-7)	PORT_PIN	TOT1	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN354 (0x02C4)	ZIN8	RESSEL (0-7)	PORT_PIN	TOT16	PPG6_TOUT0	PPG6_TOUT2	PPG7_TOUT0	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN355 (0x02C6)	AIN9	RESSEL (0-7)	PORT_PIN	TOT16	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-

Register (Offset)	Resource	RESSEL [3:0] /PORT SEL[3:0]	Source for Resource input							
			0	1	2	3	4	5	6	7
			8	9	10	11	12	13	14	15
RIC_RESIN356 (0x02C8)	BIN9	RESSEL (0-7)	PORT_PIN	TOT17	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN357 (0x02CA)	ZIN9	RESSEL (0-7)	PORT_PIN	TOT0	PPG6_TOUT0	PPG6_TOUT2	PPG7_TOUT0	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN376 (0x02F0)	PPG0_TIN1	RESSEL (0-7)	PORT_PIN	TOT0	RLT0_UFSET	TOT0	RLT0_UFSET	FRT0_MTSF	OCU0_OTD0	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P1_16	P3_04	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN377 (0x02F2)	PPG0_TIN2	RESSEL (0-7)	set 0	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN378 (0x02F4)	PPG0_TIN3	RESSEL (0-7)	set 0	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN379 (0x02F6)	PPG1_TIN1	RESSEL (0-7)	PORT_PIN	TOT0	RLT0_UFSET	TOT0	RLT0_UFSET	FRT0_MTSF	OCU0_OTD0	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P1_16	P3_04	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-

Register (Offset)	Resource	RESSEL [3:0] /PORT SEL[3:0]	Source for Resource input							
			0	1	2	3	4	5	6	7
			8	9	10	11	12	13	14	15
RIC_RESIN380 (0x02F8)	PPG1_TIN2	RESSEL (0-7)	set 0	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN381 (0x02FA)	PPG1_TIN3	RESSEL (0-7)	set 0	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN382 (0x02FC)	PPG2_TIN1	RESSEL (0-7)	PORT_PIN	TOT0	RLT0_UFSET	TOT0	RLT0_UFSET	FRT0_MTSF	OCU0_OTD0	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P1_16	P3_04	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN383 (0x02FE)	PPG2_TIN2	RESSEL (0-7)	set 0	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN384 (0x0300)	PPG2_TIN3	RESSEL (0-7)	set 0	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN385 (0x0302)	PPG3_TIN1	RESSEL (0-7)	PORT_PIN	TOT0	RLT0_UFSET	TOT1	RLT1_UFSET	FRT0_MTSF	OCU0_OTD0	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P1_16	P3_04	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-

Register (Offset)	Resource	RESSEL [3:0] /PORT SEL[3:0]	Source for Resource input							
			0	1	2	3	4	5	6	7
			8	9	10	11	12	13	14	15
RIC_RESIN386 (0x0304)	PPG3_TIN2	RESSEL (0-7)	set 0	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN387 (0x0306)	PPG3_TIN3	RESSEL (0-7)	set 0	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN388 (0x0308)	PPG4_TIN1	RESSEL (0-7)	PORT_PIN	TOT0	RLT0_UFSET	TOT1	RLT1_UFSET	FRT0_MTSF	OCU0_OTD0	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P1_16	P3_04	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN389 (0x030A)	PPG4_TIN2	RESSEL (0-7)	set 0	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN390 (0x030C)	PPG4_TIN3	RESSEL (0-7)	set 0	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN391 (0x030E)	PPG5_TIN1	RESSEL (0-7)	PORT_PIN	TOT0	RLT0_UFSET	TOT1	RLT1_UFSET	FRT0_MTSF	OCU0_OTD0	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P1_16	P3_04	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-

Register (Offset)	Resource	RESSEL [3:0] /PORT SEL[3:0]	Source for Resource input							
			0	1	2	3	4	5	6	7
			8	9	10	11	12	13	14	15
RIC_RESIN392 (0x0310)	PPG5_TIN2	RESSEL (0-7)	set 0	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN393 (0x0312)	PPG5_TIN3	RESSEL (0-7)	set 0	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN394 (0x0314)	PPG6_TIN1	RESSEL (0-7)	PORT_PIN	TOT0	RLT0_UFSET	TOT16	RLT16_UFSET	FRT8_MTSF	OCU8_OTD0	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P1_29	P3_20	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN395 (0x0316)	PPG6_TIN2	RESSEL (0-7)	set 0	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN396 (0x0318)	PPG6_TIN3	RESSEL (0-7)	set 0	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN397 (0x031A)	PPG7_TIN1	RESSEL (0-7)	PORT_PIN	TOT0	RLT0_UFSET	TOT16	RLT16_UFSET	FRT8_MTSF	OCU8_OTD0	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P1_29	P3_20	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-

Register (Offset)	Resource	RESSEL [3:0] /PORT SEL[3:0]	Source for Resource input							
			0	1	2	3	4	5	6	7
			8	9	10	11	12	13	14	15
RIC_RESIN398 (0x031C)	PPG7_TIN2	RESSEL (0-7)	set 0	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN399 (0x031E)	PPG7_TIN3	RESSEL (0-7)	set 0	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN400 (0x0320)	PPG8_TIN1	RESSEL (0-7)	PORT_PIN	TOT0	RLT0_UFSET	TOT16	RLT16_UFSET	FRT8_MTSF	OCU8_OTD0	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P1_29	P3_20	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN401 (0x0322)	PPG8_TIN2	RESSEL (0-7)	set 0	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN402 (0x0324)	PPG8_TIN3	RESSEL (0-7)	set 0	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN403 (0x0326)	PPG9_TIN1	RESSEL (0-7)	PORT_PIN	TOT0	RLT0_UFSET	TOT17	RLT17_UFSET	FRT8_MTSF	OCU8_OTD0	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P1_29	P3_20	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-

Register (Offset)	Resource	RESSEL [3:0] /PORT SEL[3:0]	Source for Resource input							
			0	1	2	3	4	5	6	7
			8	9	10	11	12	13	14	15
RIC_RESIN404 (0x0328)	PPG9_TIN2	RESSEL (0-7)	set 0	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN405 (0x032A)	PPG9_TIN3	RESSEL (0-7)	set 0	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN406 (0x032C)	PPG10_TIN1	RESSEL (0-7)	PORT_PIN	TOT0	RLT0_UFSET	TOT17	RLT17_UFSET	FRT8_MTSF	OCU8_OTD0	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P1_29	P3_20	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN407 (0x032E)	PPG10_TIN2	RESSEL (0-7)	set 0	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN408 (0x0330)	PPG10_TIN3	RESSEL (0-7)	set 0	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN409 (0x0332)	PPG11_TIN1	RESSEL (0-7)	PORT_PIN	TOT0	RLT0_UFSET	TOT17	RLT17_UFSET	FRT8_MTSF	OCU8_OTD0	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P1_29	P3_20	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-

Register (Offset)	Resource	RESSEL [3:0] /PORT SEL[3:0]	Source for Resource input							
			0	1	2	3	4	5	6	7
			8	9	10	11	12	13	14	15
RIC_RESIN410 (0x0334)	PPG11_TIN2	RESSEL (0-7)	set 0	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN411 (0x0336)	PPG11_TIN3	RESSEL (0-7)	set 0	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN430 (0x035C)	PPG12_TIN1	RESSEL (0-7)	PORT_PIN	TOT0	RLT0_UFSET	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P2_06	P3_31	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN431 (0x035E)	PPG12_TIN2	RESSEL (0-7)	set 0	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN432 (0x0360)	PPG12_TIN3	RESSEL (0-7)	set 0	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN433 (0x0362)	PPG13_TIN1	RESSEL (0-7)	PORT_PIN	TOT0	RLT0_UFSET	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P2_06	P3_31	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-

Register (Offset)	Resource	RESSEL [3:0] /PORT SEL[3:0]	Source for Resource input							
			0	1	2	3	4	5	6	7
			8	9	10	11	12	13	14	15
RIC_RESIN434 (0x0364)	PPG13_TIN2	RESSEL (0-7)	set 0	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN435 (0x0366)	PPG13_TIN3	RESSEL (0-7)	set 0	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN436 (0x0368)	PPG14_TIN1	RESSEL (0-7)	PORT_PIN	TOT0	RLT0_UFSET	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P2_06	P3_31	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN437 (0x036A)	PPG14_TIN2	RESSEL (0-7)	set 0	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN438 (0x036C)	PPG14_TIN3	RESSEL (0-7)	set 0	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN439 (0x036E)	PPG15_TIN1	RESSEL (0-7)	PORT_PIN	TOT0	RLT0_UFSET	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P2_06	P3_31	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-

Register (Offset)	Resource	RESSEL [3:0] /PORT SEL[3:0]	Source for Resource input							
			0	1	2	3	4	5	6	7
			8	9	10	11	12	13	14	15
RIC_RESIN440 (0x0370)	PPG15_TIN2	RESSEL (0-7)	set 0	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN441 (0x0372)	PPG15_TIN3	RESSEL (0-7)	set 0	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN442 (0x0374)	PPG16_TIN1	RESSEL (0-7)	PORT_PIN	TOT0	RLT0_UFSET	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN443 (0x0376)	PPG16_TIN2	RESSEL (0-7)	set 0	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN444 (0x0378)	PPG16_TIN3	RESSEL (0-7)	set 0	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN445 (0x037A)	PPG17_TIN1	RESSEL (0-7)	PORT_PIN	TOT0	RLT0_UFSET	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-

Register (Offset)	Resource	RESSEL [3:0] /PORT SEL[3:0]	Source for Resource input							
			0	1	2	3	4	5	6	7
			8	9	10	11	12	13	14	15
RIC_RESIN446 (0x037C)	PPG17_TIN2	RESSEL (0-7)	set 0	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN447 (0x037E)	PPG17_TIN3	RESSEL (0-7)	set 0	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN448 (0x0380)	PPG18_TIN1	RESSEL (0-7)	PORT_PIN	TOT0	RLT0_UFSET	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN449 (0x0382)	PPG18_TIN2	RESSEL (0-7)	set 0	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN450 (0x0384)	PPG18_TIN3	RESSEL (0-7)	set 0	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN451 (0x0386)	PPG19_TIN1	RESSEL (0-7)	PORT_PIN	TOT0	RLT0_UFSET	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-

Register (Offset)	Resource	RESSEL [3:0] /PORT SEL[3:0]	Source for Resource input							
			0	1	2	3	4	5	6	7
			8	9	10	11	12	13	14	15
RIC_RESIN452 (0x0388)	PPG19_TIN2	RESSEL (0-7)	set 0	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN453 (0x038A)	PPG19_TIN3	RESSEL (0-7)	set 0	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN454 (0x038C)	PPG20_TIN1	RESSEL (0-7)	PORT_PIN	TOT0	RLT0_UFSET	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN455 (0x038E)	PPG20_TIN2	RESSEL (0-7)	set 0	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN456 (0x0390)	PPG20_TIN3	RESSEL (0-7)	set 0	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN457 (0x0392)	PPG21_TIN1	RESSEL (0-7)	PORT_PIN	TOT0	RLT0_UFSET	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-

Register (Offset)	Resource	RESSEL [3:0] /PORT SEL[3:0]	Source for Resource input							
			0	1	2	3	4	5	6	7
			8	9	10	11	12	13	14	15
RIC_RESIN458 (0x0394)	PPG21_TIN2	RESSEL (0-7)	set 0	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN459 (0x0396)	PPG21_TIN3	RESSEL (0-7)	set 0	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN460 (0x0398)	PPG22_TIN1	RESSEL (0-7)	PORT_PIN	TOT0	RLT0_UFSET	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN461 (0x039A)	PPG22_TIN2	RESSEL (0-7)	set 0	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN462 (0x039C)	PPG22_TIN3	RESSEL (0-7)	set 0	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN463 (0x039E)	PPG23_TIN1	RESSEL (0-7)	PORT_PIN	TOT0	RLT0_UFSET	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-

Register (Offset)	Resource	RESSEL [3:0] /PORT SEL[3:0]	Source for Resource input							
			0	1	2	3	4	5	6	7
			8	9	10	11	12	13	14	15
RIC_RESIN464 (0x03A0)	PPG23_TIN2	RESSEL (0-7)	set 0	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN465 (0x03A2)	PPG23_TIN3	RESSEL (0-7)	set 0	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN466 (0x03A4)	PPG24_TIN1	RESSEL (0-7)	PORT_PIN	TOT0	RLT0_UFSET	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN467 (0x03A6)	PPG24_TIN2	RESSEL (0-7)	set 0	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN468 (0x03A8)	PPG24_TIN3	RESSEL (0-7)	set 0	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN469 (0x03AA)	PPG25_TIN1	RESSEL (0-7)	PORT_PIN	TOT0	RLT0_UFSET	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-

Register (Offset)	Resource	RESSEL [3:0] /PORT SEL[3:0]	Source for Resource input							
			0	1	2	3	4	5	6	7
			8	9	10	11	12	13	14	15
RIC_RESIN470 (0x03AC)	PPG25_TIN2	RESSEL (0-7)	set 0	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN471 (0x03AE)	PPG25_TIN3	RESSEL (0-7)	set 0	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN472 (0x03B0)	PPG26_TIN1	RESSEL (0-7)	PORT_PIN	TOT0	RLT0_UFSET	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN473 (0x03B2)	PPG26_TIN2	RESSEL (0-7)	set 0	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN474 (0x03B4)	PPG26_TIN3	RESSEL (0-7)	set 0	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN475 (0x03B6)	PPG27_TIN1	RESSEL (0-7)	PORT_PIN	TOT0	RLT0_UFSET	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-

Register (Offset)	Resource	RESSEL [3:0] /PORT SEL[3:0]	Source for Resource input							
			0	1	2	3	4	5	6	7
			8	9	10	11	12	13	14	15
RIC_RESIN476 (0x03B8)	PPG27_TIN2	RESSEL (0-7)	set 0	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN477 (0x03BA)	PPG27_TIN3	RESSEL (0-7)	set 0	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN478 (0x03BC)	PPG28_TIN1	RESSEL (0-7)	PORT_PIN	TOT0	RLT0_UFSET	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN479 (0x03BE)	PPG28_TIN2	RESSEL (0-7)	set 0	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN480 (0x03C0)	PPG28_TIN3	RESSEL (0-7)	set 0	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN481 (0x03C2)	PPG29_TIN1	RESSEL (0-7)	PORT_PIN	TOT0	RLT0_UFSET	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-

Register (Offset)	Resource	RESSEL [3:0] /PORT SEL[3:0]	Source for Resource input							
			0	1	2	3	4	5	6	7
			8	9	10	11	12	13	14	15
RIC_RESIN482 (0x03C4)	PPG29_TIN2	RESSEL (0-7)	set 0	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN483 (0x03C6)	PPG29_TIN3	RESSEL (0-7)	set 0	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN484 (0x03C8)	PPG30_TIN1	RESSEL (0-7)	PORT_PIN	TOT0	RLT0_UFSET	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN485 (0x03CA)	PPG30_TIN2	RESSEL (0-7)	set 0	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN486 (0x03CC)	PPG30_TIN3	RESSEL (0-7)	set 0	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN487 (0x03CE)	PPG31_TIN1	RESSEL (0-7)	PORT_PIN	TOT0	RLT0_UFSET	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-

Register (Offset)	Resource	RESSEL [3:0] /PORTSEL[3:0]	Source for Resource input							
			0	1	2	3	4	5	6	7
			8	9	10	11	12	13	14	15
RIC_RESIN488 (0x03D0)	PPG31_TIN2	RESSEL (0-7)	set 0	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN489 (0x03D2)	PPG31_TIN3	RESSEL (0-7)	set 0	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN490 (0x03D4)	ADC12B0_HWTRG0	RESSEL (0-7)	PORT_PIN	RLT0_UFSET	RLT1_UFSET	OCU0_OTD0	OCU1_OTD0	PPG0_TOUT0	PPG1_TOUT2	PPG3_TOUT2
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN491 (0x03D6)	ADC12B0_HWTRG1	RESSEL (0-7)	PORT_PIN	RLT1_UFSET	RLT16_UFSET	OCU1_OTD0	OCU2_OTD0	PPG0_TOUT2	PPG2_TOUT0	PPG4_TOUT0
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN492 (0x03D8)	ADC12B0_HWTRG2	RESSEL (0-7)	PORT_PIN	RLT16_UFSET	RLT17_UFSET	OCU2_OTD0	OCU8_OTD0	PPG1_TOUT0	PPG2_TOUT2	PPG4_TOUT2
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN493 (0x03DA)	ADC12B0_HWTRG3	RESSEL (0-7)	PORT_PIN	RLT17_UFSET	RLT0_UFSET	OCU8_OTD0	OCU9_OTD0	PPG1_TOUT2	PPG3_TOUT0	PPG5_TOUT0
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-

Register (Offset)	Resource	RESSEL [3:0] /PORT SEL[3:0]	Source for Resource input							
			0	1	2	3	4	5	6	7
			8	9	10	11	12	13	14	15
RIC_RESIN494 (0x03DC)	ADC12B0_HWTRG4	RESSEL (0-7)	PORT_PIN	RLT0_UFSET	RLT16_UFSET	OCU9_OTD0	OCU10_OTD0	PPG2_TOUT0	PPG3_TOUT2	PPG5_TOUT2
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN495 (0x03DE)	ADC12B0_HWTRG5	RESSEL (0-7)	PORT_PIN	RLT1_UFSET	RLT17_UFSET	OCU10_OTD0	OCU0_OTD0	PPG2_TOUT2	PPG4_TOUT0	PPG6_TOUT0
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN496 (0x03E0)	ADC12B0_HWTRG6	RESSEL (0-7)	PORT_PIN	RLT16_UFSET	RLT0_UFSET	OCU0_OTD0	OCU2_OTD0	PPG3_TOUT0	PPG4_TOUT2	PPG6_TOUT2
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN497 (0x03E2)	ADC12B0_HWTRG7	RESSEL (0-7)	PORT_PIN	RLT17_UFSET	RLT1_UFSET	OCU1_OTD0	OCU8_OTD0	PPG3_TOUT2	PPG5_TOUT0	PPG7_TOUT0
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN498 (0x03E4)	ADC12B0_HWTRG8	RESSEL (0-7)	PORT_PIN	RLT0_UFSET	RLT17_UFSET	OCU2_OTD0	OCU9_OTD0	PPG4_TOUT0	PPG5_TOUT2	PPG7_TOUT2
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN499 (0x03E6)	ADC12B0_HWTRG9	RESSEL (0-7)	PORT_PIN	RLT1_UFSET	RLT0_UFSET	OCU8_OTD0	OCU10_OTD0	PPG4_TOUT2	PPG6_TOUT0	PPG8_TOUT0
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-

Register (Offset)	Resource	RESSEL [3:0] /PORT SEL[3:0]	Source for Resource input							
			0	1	2	3	4	5	6	7
			8	9	10	11	12	13	14	15
RIC_RESIN500 (0x03E8)	ADC12B0_HWTRG10	RESSEL (0-7)	PORT_PIN	RLT16_UFSET	RLT1_UFSET	OCU9_OTD0	OCU0_OTD0	PPG5_TOUT0	PPG6_TOUT2	PPG8_TOUT2
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN501 (0x03EA)	ADC12B0_HWTRG11	RESSEL (0-7)	PORT_PIN	RLT17_UFSET	RLT16_UFSET	OCU10_OTD0	OCU1_OTD0	PPG5_TOUT2	PPG7_TOUT0	PPG9_TOUT0
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN502 (0x03EC)	ADC12B0_HWTRG12	RESSEL (0-7)	PORT_PIN	RLT0_UFSET	RLT1_UFSET	OCU0_OTD0	OCU8_OTD0	PPG6_TOUT0	PPG7_TOUT2	PPG9_TOUT2
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN503 (0x03EE)	ADC12B0_HWTRG13	RESSEL (0-7)	PORT_PIN	RLT1_UFSET	RLT16_UFSET	OCU1_OTD0	OCU9_OTD0	PPG6_TOUT2	PPG8_TOUT0	PPG10_TOUT0
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN504 (0x03F0)	ADC12B0_HWTRG14	RESSEL (0-7)	PORT_PIN	RLT16_UFSET	RLT17_UFSET	OCU2_OTD0	OCU10_OTD0	PPG7_TOUT0	PPG8_TOUT2	PPG10_TOUT2
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN505 (0x03F2)	ADC12B0_HWTRG15	RESSEL (0-7)	PORT_PIN	RLT17_UFSET	RLT0_UFSET	OCU8_OTD0	OCU0_OTD0	PPG7_TOUT2	PPG9_TOUT0	PPG11_TOUT0
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-

Register (Offset)	Resource	RESSEL [3:0] /PORT SEL[3:0]	Source for Resource input							
			0	1	2	3	4	5	6	7
			8	9	10	11	12	13	14	15
RIC_RESIN506 (0x03F4)	ADC12B0_HWTRG16	RESSEL (0-7)	PORT_PIN	RLT0_UFSET	RLT16_UFSET	OCU9_OTD0	OCU1_OTD0	PPG8_TOUT0	PPG9_TOUT2	PPG11_TOUT2
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN507 (0x03F6)	ADC12B0_HWTRG17	RESSEL (0-7)	PORT_PIN	RLT1_UFSET	RLT17_UFSET	OCU10_OTD0	OCU2_OTD0	PPG8_TOUT2	PPG10_TOUT0	PPG12_TOUT0
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN508 (0x03F8)	ADC12B0_HWTRG18	RESSEL (0-7)	PORT_PIN	RLT16_UFSET	RLT0_UFSET	OCU0_OTD0	OCU9_OTD0	PPG9_TOUT0	PPG10_TOUT2	PPG12_TOUT2
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN509 (0x03FA)	ADC12B0_HWTRG19	RESSEL (0-7)	PORT_PIN	RLT17_UFSET	RLT1_UFSET	OCU1_OTD0	OCU10_OTD0	PPG9_TOUT2	PPG11_TOUT0	PPG13_TOUT0
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN510 (0x03FC)	ADC12B0_HWTRG20	RESSEL (0-7)	PORT_PIN	RLT0_UFSET	RLT17_UFSET	OCU2_OTD0	OCU0_OTD0	PPG10_TOUT0	PPG11_TOUT2	PPG13_TOUT2
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN511 (0x03FE)	ADC12B0_HWTRG21	RESSEL (0-7)	PORT_PIN	RLT1_UFSET	RLT0_UFSET	OCU8_OTD0	OCU1_OTD0	PPG10_TOUT2	PPG12_TOUT0	PPG14_TOUT0
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-

Register (Offset)	Resource	RESSEL [3:0] /PORT SEL[3:0]	Source for Resource input							
			0	1	2	3	4	5	6	7
			8	9	10	11	12	13	14	15
RIC_RESIN512 (0x0400)	ADC12B0_HWTRG 22	RESSEL (0-7)	PORT_PIN	RLT16_UFSET	RLT1_UFSET	OCU9_OTD0	OCU2_OTD0	PPG11_TOUT0	PPG12_TOUT2	PPG14_TOUT2
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN513 (0x0402)	ADC12B0_HWTRG 23	RESSEL (0-7)	PORT_PIN	RLT17_UFSET	RLT16_UFSET	OCU10_OTD0	OCU8_OTD0	PPG11_TOUT2	PPG13_TOUT0	PPG15_TOUT0
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN514 (0x0404)	ADC12B0_HWTRG 24	RESSEL (0-7)	PORT_PIN	RLT0_UFSET	RLT1_UFSET	OCU0_OTD0	OCU10_OTD0	PPG12_TOUT0	PPG13_TOUT2	PPG15_TOUT2
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN515 (0x0406)	ADC12B0_HWTRG 25	RESSEL (0-7)	PORT_PIN	RLT1_UFSET	RLT16_UFSET	OCU1_OTD0	OCU0_OTD0	PPG12_TOUT2	PPG14_TOUT0	PPG16_TOUT0
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN516 (0x0408)	ADC12B0_HWTRG 26	RESSEL (0-7)	PORT_PIN	RLT16_UFSET	RLT17_UFSET	OCU2_OTD0	OCU1_OTD0	PPG13_TOUT0	PPG14_TOUT2	PPG16_TOUT2
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN517 (0x040A)	ADC12B0_HWTRG 27	RESSEL (0-7)	PORT_PIN	RLT17_UFSET	RLT0_UFSET	OCU8_OTD0	OCU2_OTD0	PPG13_TOUT2	PPG15_TOUT0	PPG17_TOUT0
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-

Register (Offset)	Resource	RESSEL [3:0] /PORT SEL[3:0]	Source for Resource input							
			0	1	2	3	4	5	6	7
			8	9	10	11	12	13	14	15
RIC_RESIN518 (0x040C)	ADC12B0_HWTRG28	RESSEL (0-7)	PORT_PIN	RLT0_UFSET	RLT16_UFSET	OCU9_OTD0	OCU8_OTD0	PPG14_TOUT0	PPG15_TOUT2	PPG17_TOUT2
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN519 (0x040E)	ADC12B0_HWTRG29	RESSEL (0-7)	PORT_PIN	RLT1_UFSET	RLT17_UFSET	OCU10_OTD0	OCU9_OTD0	PPG14_TOUT2	PPG16_TOUT0	PPG18_TOUT0
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN520 (0x0410)	ADC12B0_HWTRG30	RESSEL (0-7)	PORT_PIN	RLT16_UFSET	RLT0_UFSET	OCU0_OTD0	OCU1_OTD0	PPG15_TOUT0	PPG16_TOUT2	PPG18_TOUT2
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN521 (0x0412)	ADC12B0_HWTRG31	RESSEL (0-7)	PORT_PIN	RLT17_UFSET	RLT1_UFSET	OCU1_OTD0	OCU2_OTD0	PPG15_TOUT2	PPG17_TOUT0	PPG19_TOUT0
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN586 (0x0494)	ADC12B1_HWTRG32	RESSEL (0-7)	PORT_PIN	RLT0_UFSET	RLT17_UFSET	OCU2_OTD0	OCU8_OTD0	PPG16_TOUT0	PPG17_TOUT2	PPG19_TOUT2
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN587 (0x0496)	ADC12B1_HWTRG33	RESSEL (0-7)	PORT_PIN	RLT1_UFSET	RLT0_UFSET	OCU8_OTD0	OCU9_OTD0	PPG16_TOUT2	PPG18_TOUT0	PPG20_TOUT0
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-

Register (Offset)	Resource	RESSEL [3:0] /PORT SEL[3:0]	Source for Resource input							
			0	1	2	3	4	5	6	7
			8	9	10	11	12	13	14	15
RIC_RESIN588 (0x0498)	ADC12B1_HWTRG34	RESSEL (0-7)	PORT_PIN	RLT16_UFSET	RLT1_UFSET	OCU9_OTD0	OCU10_OTD0	PPG17_TOUT0	PPG18_TOUT2	PPG20_TOUT2
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN589 (0x049A)	ADC12B1_HWTRG35	RESSEL (0-7)	PORT_PIN	RLT17_UFSET	RLT16_UFSET	OCU10_OTD0	OCU0_OTD0	PPG17_TOUT2	PPG19_TOUT0	PPG21_TOUT0
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN590 (0x049C)	ADC12B1_HWTRG36	RESSEL (0-7)	PORT_PIN	RLT0_UFSET	RLT1_UFSET	OCU0_OTD0	OCU2_OTD0	PPG18_TOUT0	PPG19_TOUT2	PPG21_TOUT2
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN591 (0x049E)	ADC12B1_HWTRG37	RESSEL (0-7)	PORT_PIN	RLT1_UFSET	RLT16_UFSET	OCU1_OTD0	OCU8_OTD0	PPG18_TOUT2	PPG20_TOUT0	PPG22_TOUT0
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN592 (0x04A0)	ADC12B1_HWTRG38	RESSEL (0-7)	PORT_PIN	RLT16_UFSET	RLT17_UFSET	OCU2_OTD0	OCU9_OTD0	PPG19_TOUT0	PPG20_TOUT2	PPG22_TOUT2
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN593 (0x04A2)	ADC12B1_HWTRG39	RESSEL (0-7)	PORT_PIN	RLT17_UFSET	RLT0_UFSET	OCU8_OTD0	OCU10_OTD0	PPG19_TOUT2	PPG21_TOUT0	PPG23_TOUT0
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-

Register (Offset)	Resource	RESSEL [3:0] /PORT SEL[3:0]	Source for Resource input							
			0	1	2	3	4	5	6	7
			8	9	10	11	12	13	14	15
RIC_RESIN594 (0x04A4)	ADC12B1_HWTRG40	RESSEL (0-7)	PORT_PIN	RLT0_UFSET	RLT16_UFSET	OCU9_OTD0	OCU0_OTD0	PPG20_TOUT0	PPG21_TOUT2	PPG23_TOUT2
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN595 (0x04A6)	ADC12B1_HWTRG41	RESSEL (0-7)	PORT_PIN	RLT1_UFSET	RLT17_UFSET	OCU10_OTD0	OCU1_OTD0	PPG20_TOUT2	PPG22_TOUT0	PPG24_TOUT0
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN596 (0x04A8)	ADC12B1_HWTRG42	RESSEL (0-7)	PORT_PIN	RLT16_UFSET	RLT0_UFSET	OCU0_OTD0	OCU8_OTD0	PPG21_TOUT0	PPG22_TOUT2	PPG24_TOUT2
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN597 (0x04AA)	ADC12B1_HWTRG43	RESSEL (0-7)	PORT_PIN	RLT17_UFSET	RLT1_UFSET	OCU1_OTD0	OCU9_OTD0	PPG21_TOUT2	PPG23_TOUT0	PPG25_TOUT0
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN598 (0x04AC)	ADC12B1_HWTRG44	RESSEL (0-7)	PORT_PIN	RLT0_UFSET	RLT17_UFSET	OCU2_OTD0	OCU10_OTD0	PPG22_TOUT0	PPG23_TOUT2	PPG25_TOUT2
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN599 (0x04AE)	ADC12B1_HWTRG45	RESSEL (0-7)	PORT_PIN	RLT1_UFSET	RLT0_UFSET	OCU8_OTD0	OCU0_OTD0	PPG22_TOUT2	PPG24_TOUT0	PPG26_TOUT0
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-

Register (Offset)	Resource	RESSEL [3:0] /PORT SEL[3:0]	Source for Resource input							
			0	1	2	3	4	5	6	7
			8	9	10	11	12	13	14	15
RIC_RESIN600 (0x04B0)	ADC12B1_HWTRG 46	RESSEL (0-7)	PORT_PIN	RLT16_UFSET	RLT1_UFSET	OCU9_OTD0	OCU1_OTD0	PPG23_TOUT0	PPG24_TOUT2	PPG26_TOUT2
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN601 (0x04B2)	ADC12B1_HWTRG 47	RESSEL (0-7)	PORT_PIN	RLT17_UFSET	RLT16_UFSET	OCU10_OTD0	OCU2_OTD0	PPG23_TOUT2	PPG25_TOUT0	PPG27_TOUT0
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN602 (0x04B4)	ADC12B1_HWTRG 48	RESSEL (0-7)	PORT_PIN	RLT0_UFSET	RLT1_UFSET	OCU0_OTD0	OCU9_OTD0	PPG24_TOUT0	PPG25_TOUT2	PPG27_TOUT2
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN603 (0x04B6)	ADC12B1_HWTRG 49	RESSEL (0-7)	PORT_PIN	RLT1_UFSET	RLT16_UFSET	OCU1_OTD0	OCU10_OTD0	PPG24_TOUT2	PPG26_TOUT0	PPG28_TOUT0
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN604 (0x04B8)	ADC12B1_HWTRG 50	RESSEL (0-7)	PORT_PIN	RLT16_UFSET	RLT17_UFSET	OCU2_OTD0	OCU0_OTD0	PPG25_TOUT0	PPG26_TOUT2	PPG28_TOUT2
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN605 (0x04BA)	ADC12B1_HWTRG 51	RESSEL (0-7)	PORT_PIN	RLT17_UFSET	RLT0_UFSET	OCU8_OTD0	OCU1_OTD0	PPG25_TOUT2	PPG27_TOUT0	PPG29_TOUT0
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-

Register (Offset)	Resource	RESSEL [3:0] /PORT SEL[3:0]	Source for Resource input							
			0	1	2	3	4	5	6	7
			8	9	10	11	12	13	14	15
RIC_RESIN606 (0x04BC)	ADC12B1_HWTRG52	RESSEL (0-7)	PORT_PIN	RLT0_UFSET	RLT16_UFSET	OCU9_OTD0	OCU2_OTD0	PPG26_TOUT0	PPG27_TOUT2	PPG29_TOUT2
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN607 (0x04BE)	ADC12B1_HWTRG53	RESSEL (0-7)	PORT_PIN	RLT1_UFSET	RLT17_UFSET	OCU10_OTD0	OCU8_OTD0	PPG26_TOUT2	PPG28_TOUT0	PPG30_TOUT0
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN608 (0x04C0)	ADC12B1_HWTRG54	RESSEL (0-7)	PORT_PIN	RLT16_UFSET	RLT0_UFSET	OCU0_OTD0	OCU10_OTD0	PPG27_TOUT0	PPG28_TOUT2	PPG30_TOUT2
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN609 (0x04C2)	ADC12B1_HWTRG55	RESSEL (0-7)	PORT_PIN	RLT17_UFSET	RLT1_UFSET	OCU1_OTD0	OCU0_OTD0	PPG27_TOUT2	PPG29_TOUT0	PPG31_TOUT0
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN610 (0x04C4)	ADC12B1_HWTRG56	RESSEL (0-7)	PORT_PIN	RLT0_UFSET	RLT17_UFSET	OCU2_OTD0	OCU1_OTD0	PPG28_TOUT0	PPG29_TOUT2	PPG31_TOUT2
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN611 (0x04C6)	ADC12B1_HWTRG57	RESSEL (0-7)	PORT_PIN	RLT1_UFSET	RLT0_UFSET	OCU8_OTD0	OCU2_OTD0	PPG28_TOUT2	PPG30_TOUT0	PPG0_TOUT0
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-

Register (Offset)	Resource	RESSEL [3:0] /PORT SEL[3:0]	Source for Resource input							
			0	1	2	3	4	5	6	7
			8	9	10	11	12	13	14	15
RIC_RESIN612 (0x04C8)	ADC12B1_HWTRG58	RESSEL (0-7)	PORT_PIN	RLT16_UFSET	RLT1_UFSET	OCU9_OTD0	OCU8_OTD0	PPG29_TOUT0	PPG30_TOUT2	PPG0_TOUT2
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN613 (0x04CA)	ADC12B1_HWTRG59	RESSEL (0-7)	PORT_PIN	RLT17_UFSET	RLT16_UFSET	OCU10_OTD0	OCU9_OTD0	PPG29_TOUT2	PPG31_TOUT0	PPG1_TOUT0
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN614 (0x04CC)	ADC12B1_HWTRG60	RESSEL (0-7)	PORT_PIN	RLT0_UFSET	RLT1_UFSET	OCU0_OTD0	OCU1_OTD0	PPG30_TOUT0	PPG31_TOUT2	PPG1_TOUT2
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN615 (0x04CE)	ADC12B1_HWTRG61	RESSEL (0-7)	PORT_PIN	RLT1_UFSET	RLT16_UFSET	OCU1_OTD0	OCU2_OTD0	PPG30_TOUT2	PPG0_TOUT0	PPG2_TOUT0
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN616 (0x04D0)	ADC12B1_HWTRG62	RESSEL (0-7)	PORT_PIN	RLT16_UFSET	RLT17_UFSET	OCU2_OTD0	OCU8_OTD0	PPG31_TOUT0	PPG0_TOUT2	PPG2_TOUT2
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN617 (0x04D2)	ADC12B1_HWTRG63	RESSEL (0-7)	PORT_PIN	RLT17_UFSET	RLT0_UFSET	OCU8_OTD0	OCU9_OTD0	PPG31_TOUT2	PPG1_TOUT0	PPG3_TOUT0
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-

Register (Offset)	Resource	RESSEL [3:0] /PORT SEL[3:0]	Source for Resource input							
			0	1	2	3	4	5	6	7
			8	9	10	11	12	13	14	15
RIC_RESIN626 (0x04E4)	DDRHSSPI_MSTART	RESSEL (0-7)	-	TOT0	TOT16	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN629 (0x04EA)	MDIO	RESSEL (0-7)	-	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P0_31	P3_06	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN630 (0x04EC)	CRS	RESSEL (0-7)	-	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P1_02	P0_20	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN631 (0x04EE)	RXD0	RESSEL (0-7)	-	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P0_27	P3_04	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN632 (0x04F0)	RXD1	RESSEL (0-7)	-	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P0_28	P4_06	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN633 (0x04F2)	RXD2	RESSEL (0-7)	-	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P0_29	P4_07	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-

Register (Offset)	Resource	RESSEL [3:0] /PORT SEL[3:0]	Source for Resource input							
			0	1	2	3	4	5	6	7
			8	9	10	11	12	13	14	15
RIC_RESIN634 (0x04F4)	RXD3	RESSEL (0-7)	-	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P0_30	P3_05	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN635 (0x04F6)	COL	RESSEL (0-7)	-	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P1_01	P0_19	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN636 (0x04F8)	RXDV	RESSEL (0-7)	-	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P0_19	P4_02	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN637 (0x04FA)	RXER	RESSEL (0-7)	-	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P0_18	P4_01	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN638 (0x04FC)	RXCLK	RESSEL (0-7)	-	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P0_17	P4_00	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN639 (0x04FE)	TXCLK	RESSEL (0-7)	-	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P0_20	P4_03	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-

Register (Offset)	Resource	RESSEL [3:0] /PORT SEL[3:0]	Source for Resource input							
			0	1	2	3	4	5	6	7
			8	9	10	11	12	13	14	15
RIC_RESIN643 (0x0506)	I2S0_WS	RESSEL (0-7)	-	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P0_12	P3_26	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN644 (0x0508)	I2S0_SD	RESSEL (0-7)	-	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P0_11	P3_25	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN645 (0x050A)	I2S0_SCK	RESSEL (0-7)	-	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P0_13	P3_27	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN646 (0x050C)	I2S0_ECLK	RESSEL (0-7)	PORT_PIN	SYSC1_CLK_CD4	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P0_10	P3_24	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN650 (0x0514)	I2S1_ECLK	RESSEL (0-7)	PORT_PIN	SYSC1_CLK_CD4	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	-	-	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-
RIC_RESIN685 (0x055A)	ADTRG0	RESSEL (0-7)	-	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P1_16	P2_10	-	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-

Register (Offset)	Resource	RESSEL [3:0] /PORT SEL[3:0]	Source for Resource input							
			0	1	2	3	4	5	6	7
			8	9	10	11	12	13	14	15
RIC_RESIN686 (0x055C)	ADTRG1	RESSEL (0-7)	-	-	-	-	-	-	-	-
		RESSEL (8-15)	-	-	-	-	-	-	-	-
		PORTSEL (0-7)	P4_22	P1_08	P2_11	-	-	-	-	-
		PORTSEL (8-15)	-	-	-	-	-	-	-	-

Notes:

- *When both GPIO_PORTEN.GPORTEN and PPC_PCFGR.PIE are configured as 0, the input signal is disconnected and external interrupt cannot be detected. During disconnecting, I/O internally outputs "low" to internal logic, and if ELVR is configured as low-level-detection, falling-edge-detection, or both-edge-detection it will be detected as external interrupt with EIRR=1.*
- *"Set 0" (Set 1) means that "0" ("1") is inputted.*
- *OCUx_MODn is described as MODn pin in Traveo™ Platform Hardware Manual.*

3.2. Port Output Function Configuration

The port output function configuration (POF) is a function to select a function to output to a port.

A resource which supports a port output relocation has its PPC_PCFGR.POF to configure resource output.

3.2.1. Standard Configuration

Register (Offset)	Port	Resource Functional Outputs							
		POF=0	POF=1	POF=2	POF=3	POF=4	POF=5	POF=6	POF=7
PPC_PCFGR000 (0x0000)	P0_00	GPIO_PODR0:POD00	-	LCDD5	-	-	DSP0_R7_0	-	MAD1
PPC_PCFGR001 (0x0002)	P0_01	GPIO_PODR0:POD01	-	LCDD6	ARH0_AIC1_DNCLK	-	DSP0_G0_0	-	MAD2
PPC_PCFGR002 (0x0004)	P0_02	GPIO_PODR0:POD02	SCK1_0	LCDD7	ARH0_AIC1_TDA1	ARH0_AIC1_DNDATA1	DSP0_G1_0	SCL1	MAD3
PPC_PCFGR003 (0x0006)	P0_03	GPIO_PODR0:POD03	SOT1_0	LCDD8	ARH0_AIC1_dbg_out_1	-	DSP0_G2_0	SDA1	MAD4
PPC_PCFGR004 (0x0008)	P0_04	GPIO_PODR0:POD04	SCS10_0	LCDD9	ARH0_AIC1_dbg_out_0	-	DSP0_G3_0	-	MAD5
PPC_PCFGR005 (0x000A)	P0_05	GPIO_PODR0:POD05	SCS11_0	LCDD10	ARH0_AIC1_TDA0	ARH0_AIC1_DNDATA0	DSP0_G4_0	SOT0_1	MAD6
PPC_PCFGR006 (0x000C)	P0_06	GPIO_PODR0:POD06	SCS12_0	LCDD11	SCK0_1	-	DSP0_G5_0	PPG0_TOUT0_1	MAD7
PPC_PCFGR007 (0x000E)	P0_07	GPIO_PODR0:POD07	SCS13_0	LCDD12	SCS00_1	I2S1_SD_0	DSP0_G6_0	PPG0_TOUT2_1	MAD8
PPC_PCFGR008 (0x0010)	P0_08	GPIO_PODR0:POD08	-	LCDD13	-	I2S1_WS_0	DSP0_G7_0	PPG1_TOUT0_1	MAD9
PPC_PCFGR009 (0x0012)	P0_09	GPIO_PODR0:POD09	-	LCDD14	ARH0_AIC0_DNCLK	I2S1_SCK_0	DSP0_B0_0	PPG1_TOUT2_1	MAD10
PPC_PCFGR010 (0x0014)	P0_10	GPIO_PODR0:POD10	SCS171_1	LCDD15	ARH0_AIC0_TDA1	ARH0_AIC0_DNDATA1	DSP0_B1_0	PPG2_TOUT0_1	MAD11
PPC_PCFGR011 (0x0016)	P0_11	GPIO_PODR0:POD11	-	LCDD16	ARH0_AIC0_dbg_out_1	I2S0_SD_0	DSP0_B2_0	PPG2_TOUT2_1	MAD12
PPC_PCFGR012 (0x0018)	P0_12	GPIO_PODR0:POD12	-	LCDD17	ARH0_AIC0_dbg_out_0	I2S0_WS_0	DSP0_B3_0	PPG3_TOUT0_1	MAD13
PPC_PCFGR013 (0x001A)	P0_13	GPIO_PODR0:POD13	-	CS#	-	I2S0_SCK_0	DSP0_B4_0	PPG3_TOUT2_1	MAD14
PPC_PCFGR014 (0x001C)	P0_14	GPIO_PODR0:POD14	-	WR#	-	-	DSP0_B5_0	SCK4_1	MOEX
PPC_PCFGR015 (0x001E)	P0_15	GPIO_PODR0:POD15	-	RD#	-	-	DSP0_B6_0	SCS40_1	MWEX

Register (Offset)	Port	Resource Functional Outputs							
		POF=0	POF=1	POF=2	POF=3	POF=4	POF=5	POF=6	POF=7
PPC_PCFGR016 (0x0020)	P0_16	GPIO_PODR0:POD16	DSP0_B7_1	-	ARH0_AIC0_TDA0	ARH0_AIC0_DNDATA0	-	SCS41_1	MCLK
PPC_PCFGR017 (0x0022)	P0_17	GPIO_PODR0:POD17	-	-	-	-	DSP0_B7_0	SCS43_1	MDQM0
PPC_PCFGR018 (0x0024)	P0_18	GPIO_PODR0:POD18	-	RS	-	MDC_1	-	-	MCSX2
PPC_PCFGR019 (0x0026)	P0_19	GPIO_PODR0:POD19	-	RES#	-	-	-	-	MCSX3
PPC_PCFGR020 (0x0028)	P0_20	GPIO_PODR0:POD20	-	-	-	-	-	-	-
PPC_PCFGR021 (0x002A)	P0_21	GPIO_PODR0:POD21	-	M_SDATA0_0	TXEN_0	M_DQ3	-	-	-
PPC_PCFGR022 (0x002C)	P0_22	GPIO_PODR0:POD22	SCK2_0	M_SDATA0_2	TXD0_0	M_DQ2	SOT9_2	-	-
PPC_PCFGR023 (0x002E)	P0_23	GPIO_PODR0:POD23	SOT2_0	M_SDATA0_1	TXD1_0	M_DQ1	SCK9_2	-	-
PPC_PCFGR024 (0x0030)	P0_24	GPIO_PODR0:POD24	SCS20_0	M_SSEL0	TXD2_0	M_DQ0	SCS90_2	-	-
PPC_PCFGR025 (0x0032)	P0_25	GPIO_PODR0:POD25	SCS21_0	M_SDATA0_3	TXD3_0	M_CS#_1	SCS91_2	-	-
PPC_PCFGR026 (0x0034)	P0_26	GPIO_PODR0:POD26	SCS22_0	M_SCLK0	TXER_0	M_CK	-	-	-
PPC_PCFGR027 (0x0036)	P0_27	GPIO_PODR0:POD27	SCS23_0	M_SDATA1_0	-	M_RWDS	-	-	-
PPC_PCFGR028 (0x0038)	P0_28	GPIO_PODR0:POD28	-	M_SDATA1_2	-	M_DQ4	SCK8_2	-	-
PPC_PCFGR029 (0x003A)	P0_29	GPIO_PODR0:POD29	SCK3_0	M_SDATA1_1	-	M_DQ5	SOT8_2	-	-
PPC_PCFGR030 (0x003C)	P0_30	GPIO_PODR0:POD30	SOT3_0	M_SSEL1	-	M_DQ6	SCS80_2	-	-
PPC_PCFGR031 (0x003E)	P0_31	GPIO_PODR0:POD31	SCS30_0	M_SDATA1_3	MDIO_0	M_DQ7	-	-	-
PPC_PCFGR100 (0x0040)	P1_00	GPIO_PODR1:POD00	SCS31_0	-	MDC_0	M_CS#_2	-	-	-
PPC_PCFGR101 (0x0042)	P1_01	GPIO_PODR1:POD01	SCS32_0	-	-	-	-	-	MLBSIG
PPC_PCFGR102 (0x0044)	P1_02	GPIO_PODR1:POD02	SCS33_0	-	-	-	-	-	MLBDAT

Register (Offset)	Port	Resource Functional Outputs							
		POF=0	POF=1	POF=2	POF=3	POF=4	POF=5	POF=6	POF=7
PPC_PCFGR103 (0x0046)	P1_03	GPIO_PODR1:POD03	-	-	-	OCU0_OTD0_0	-	PPG0_TOUT0_0	BN0(BL0)
PPC_PCFGR104 (0x0048)	P1_04	GPIO_PODR1:POD04	SCK0_0	-	SCL0	OCU0_OTD1_0	TOT0_0	PPG0_TOUT2_0	BP0(BH0)
PPC_PCFGR105 (0x004A)	P1_05	GPIO_PODR1:POD05	SOT0_0	SGA0_0	SDA0	TRACE0_0	-	PPG1_TOUT0_0	AN0(AL0)
PPC_PCFGR106 (0x004C)	P1_06	GPIO_PODR1:POD06	SCS00_0	SGO0_0	TX0_0	TRACE1_0	TOT1_0	PPG1_TOUT2_0	AP0(AH0)
PPC_PCFGR107 (0x004E)	P1_07	GPIO_PODR1:POD07	-	SGA1_0	TRACE2_0	OCU1_OTD0_0	-	PPG2_TOUT0_0	BN1(BL1)
PPC_PCFGR108 (0x0050)	P1_08	GPIO_PODR1:POD08	TRACE3_0	SGO1_0	TX1_0	OCU1_OTD1_0	TOT16_0	PPG2_TOUT2_0	BP1(BH1)
PPC_PCFGR109 (0x0052)	P1_09	GPIO_PODR1:POD09	SCK16_0	SGA2_0	SCL16	OCU2_OTD0_0	TRACE_CTL_0	PPG3_TOUT0_0	AN1(AL1)
PPC_PCFGR110 (0x0054)	P1_10	GPIO_PODR1:POD10	SOT16_0	SGO2_0	SDA16	OCU2_OTD1_0	TRACE_CLK_0	PPG3_TOUT2_0	AP1(AH1)
PPC_PCFGR111 (0x0056)	P1_11	GPIO_PODR1:POD11	SCS160_0	INDICATOR0_0	-	OCU8_OTD0_0	-	PPG4_TOUT0_0	-
PPC_PCFGR112 (0x0058)	P1_12	GPIO_PODR1:POD12	SCS161_0	-	-	OCU8_OTD1_0	TOT17_0	-	TX2_0
PPC_PCFGR113 (0x005A)	P1_13	GPIO_PODR1:POD13	-	SGA3_0	-	OCU9_OTD0_0	-	-	-
PPC_PCFGR114 (0x005C)	P1_14	GPIO_PODR1:POD14	SYSC0_CLK_1	SGO3_0	-	OCU9_OTD1_0	TOT48_0	PPG5_TOUT0_0	TX3_0
PPC_PCFGR115 (0x005E)	P1_15	GPIO_PODR1:POD15	SCK17_0	SGA4_0	SCL17	OCU10_OTD0_0	SCK12_1	PPG5_TOUT2_0	INDICATOR0_1
PPC_PCFGR116 (0x0060)	P1_16	GPIO_PODR1:POD16	SOT17_0	SGO4_0	SDA17	OCU10_OTD1_0	TOT49_0	SYSC0_CLK_0	WOT
PPC_PCFGR117 (0x0062)	P1_17	GPIO_PODR1:POD17	SCS170_0	-	-	-	PWM1P0	PPG6_TOUT0_0	-
PPC_PCFGR118 (0x0064)	P1_18	GPIO_PODR1:POD18	SCS171_0	-	-	-	PWM1M0	PPG6_TOUT2_0	TX5_0
PPC_PCFGR119 (0x0066)	P1_19	GPIO_PODR1:POD19	-	-	-	-	PWM2P0	PPG7_TOUT0_0	-
PPC_PCFGR120 (0x0068)	P1_20	GPIO_PODR1:POD20	SCK8_0	-	SCL8	-	PWM2M0	PPG7_TOUT2_0	-
PPC_PCFGR121 (0x006A)	P1_21	GPIO_PODR1:POD21	SOT8_0	-	SDA8	-	PWM1P1	PPG8_TOUT0_0	-

Register (Offset)	Port	Resource Functional Outputs							
		POF=0	POF=1	POF=2	POF=3	POF=4	POF=5	POF=6	POF=7
PPC_PCFGR122 (0x006C)	P1_22	GPIO_PODR1:POD22	SCS80_0	-	-	-	PWM1M1	PPG8_TOUT2_0	TX6_0
PPC_PCFGR123 (0x006E)	P1_23	GPIO_PODR1:POD23	-	-	-	-	PWM2P1	PPG9_TOUT0_0	-
PPC_PCFGR124 (0x0070)	P1_24	GPIO_PODR1:POD24	SCK9_0	-	SCL9	-	PWM2M1	PPG9_TOUT2_0	-
PPC_PCFGR125 (0x0072)	P1_25	GPIO_PODR1:POD25	SOT9_0	-	SDA9	-	PWM1P2	PPG10_TOUT0_0	-
PPC_PCFGR126 (0x0074)	P1_26	GPIO_PODR1:POD26	SCS90_0	-	-	-	PWM1M2	PPG10_TOUT2_0	-
PPC_PCFGR127 (0x0076)	P1_27	GPIO_PODR1:POD27	SCS91_0	-	-	-	PWM2P2	PPG11_TOUT0_0	-
PPC_PCFGR128 (0x0078)	P1_28	GPIO_PODR1:POD28	-	-	-	-	PWM2M2	PPG11_TOUT2_0	-
PPC_PCFGR129 (0x007A)	P1_29	GPIO_PODR1:POD29	SCK10_0	-	SCL10	-	PWM1P3	-	-
PPC_PCFGR130 (0x007C)	P1_30	GPIO_PODR1:POD30	SOT10_0	-	SDA10	-	PWM1M3	PPG12_TOUT0_0	-
PPC_PCFGR131 (0x007E)	P1_31	GPIO_PODR1:POD31	SCS100_0	-	-	-	PWM2P3	PPG12_TOUT2_0	-
PPC_PCFGR200 (0x0080)	P2_00	GPIO_PODR2:POD00	-	-	-	-	PWM2M3	PPG13_TOUT0_0	-
PPC_PCFGR201 (0x0082)	P2_01	GPIO_PODR2:POD01	SCK11_0	-	SCL11	-	PWM1P4	PPG13_TOUT2_0	-
PPC_PCFGR202 (0x0084)	P2_02	GPIO_PODR2:POD02	SOT11_0	-	SDA11	-	PWM1M4	PPG14_TOUT0_0	-
PPC_PCFGR203 (0x0086)	P2_03	GPIO_PODR2:POD03	SCS110_0	-	-	-	PWM2P4	PPG14_TOUT2_0	-
PPC_PCFGR204 (0x0088)	P2_04	GPIO_PODR2:POD04	SCS111_0	-	-	-	PWM2M4	PPG15_TOUT0_0	-
PPC_PCFGR205 (0x008A)	P2_05	GPIO_PODR2:POD05	-	-	-	-	PWM1P5	PPG15_TOUT2_0	-
PPC_PCFGR206 (0x008C)	P2_06	GPIO_PODR2:POD06	SCK12_0	-	SCL12	-	PWM1M5	-	-
PPC_PCFGR207 (0x008E)	P2_07	GPIO_PODR2:POD07	SOT12_0	-	SDA12	-	PWM2P5	-	-
PPC_PCFGR208 (0x0090)	P2_08	GPIO_PODR2:POD08	SCS120_0	-	-	-	PWM2M5	-	-

Register (Offset)	Port	Resource Functional Outputs							
		POF=0	POF=1	POF=2	POF=3	POF=4	POF=5	POF=6	POF=7
PPC_PCFGR209 (0x0092)	P2_09	GPIO_PODR2:POD09	SCS23_1	-	-	-	DSP0_EN_0	PPG14_TOUT0_1	MCSX0
PPC_PCFGR210 (0x0094)	P2_10	GPIO_PODR2:POD10	SCS31_1	-	-	-	DSP0_HSYNC_0	-	MCSX1
PPC_PCFGR211 (0x0096)	P2_11	GPIO_PODR2:POD11	SCS32_1	-	-	-	DSP0_VSYNC_0	-	MDATA0
PPC_PCFGR212 (0x0098)	P2_12	GPIO_PODR2:POD12	SCS33_1	-	-	-	DSP0_CLK_0	-	MDATA1
PPC_PCFGR213 (0x009A)	P2_13	GPIO_PODR2:POD13	-	-	-	-	DSP0_R0_0	-	MDATA2
PPC_PCFGR214 (0x009C)	P2_14	GPIO_PODR2:POD14	SCK4_0	-	SCL4	-	DSP0_R1_0	-	MDATA3
PPC_PCFGR215 (0x009E)	P2_15	GPIO_PODR2:POD15	SOT4_0	LCDD0	SDA4	-	DSP0_R2_0	-	MDATA4
PPC_PCFGR216 (0x00A0)	P2_16	GPIO_PODR2:POD16	SCS40_0	LCDD1	-	-	DSP0_R3_0	-	MDATA5
PPC_PCFGR217 (0x00A2)	P2_17	GPIO_PODR2:POD17	SCS41_0	LCDD2	-	-	DSP0_R4_0	-	MDATA6
PPC_PCFGR218 (0x00A4)	P2_18	GPIO_PODR2:POD18	SCS42_0	LCDD3	-	-	DSP0_R5_0	-	MDATA7
PPC_PCFGR219 (0x00A6)	P2_19	GPIO_PODR2:POD19	SCS43_0	LCDD4	-	-	DSP0_R6_0	TX7_1	MAD0
PPC_PCFGR300 (0x00C0)	P3_00	GPIO_PODR3:POD00	-	TXD1_1	-	-	-	PPG4_TOUT0_1	MAD15
PPC_PCFGR301 (0x00C2)	P3_01	GPIO_PODR3:POD01	SCK1_1	TXD2_1	-	-	-	PPG4_TOUT2_1	MAD16
PPC_PCFGR302 (0x00C4)	P3_02	GPIO_PODR3:POD02	SOT1_1	TXD3_1	-	-	-	PPG5_TOUT0_1	MAD17
PPC_PCFGR303 (0x00C6)	P3_03	GPIO_PODR3:POD03	SCS10_1	TXER_1	-	-	-	PPG5_TOUT2_1	MAD18
PPC_PCFGR304 (0x00C8)	P3_04	GPIO_PODR3:POD04	SCS11_1	-	-	-	-	-	MAD19
PPC_PCFGR305 (0x00CA)	P3_05	GPIO_PODR3:POD05	SCS12_1	-	-	-	-	-	MAD20
PPC_PCFGR306 (0x00CC)	P3_06	GPIO_PODR3:POD06	SCS13_1	MDIO_1	-	-	-	SOT4_1	MAD21
PPC_PCFGR307 (0x00CE)	P3_07	GPIO_PODR3:POD07	-	-	-	-	-	SCS42_1	MDQM1

Register (Offset)	Port	Resource Functional Outputs							
		POF=0	POF=1	POF=2	POF=3	POF=4	POF=5	POF=6	POF=7
PPC_PCFGR308 (0x00D0)	P3_08	GPIO_PODR3:POD08	-	SGA0_1	PPG6_TOUT0_1	OCU0_OTD0_1	-	-	-
PPC_PCFGR309 (0x00D2)	P3_09	GPIO_PODR3:POD09	SCK8_1	SGO0_1	PPG6_TOUT2_1	OCU0_OTD1_1	TOT0_1	PPG22_TOUT0_0	-
PPC_PCFGR310 (0x00D4)	P3_10	GPIO_PODR3:POD10	SOT8_1	SGA1_1	PPG7_TOUT0_1	OCU1_OTD0_1	TRACE0_1	PPG22_TOUT2_0	TX0_1
PPC_PCFGR311 (0x00D6)	P3_11	GPIO_PODR3:POD11	SCS80_1	SGO1_1	PPG7_TOUT2_1	OCU1_OTD1_1	TOT1_1	TRACE1_1	-
PPC_PCFGR312 (0x00D8)	P3_12	GPIO_PODR3:POD12	-	SGA2_1	PPG8_TOUT0_1	OCU2_OTD0_1	-	TRACE2_1	TX1_1
PPC_PCFGR313 (0x00DA)	P3_13	GPIO_PODR3:POD13	SCK10_1	SGO2_1	PPG8_TOUT2_1	OCU2_OTD1_1	TOT16_1	TRACE3_1	-
PPC_PCFGR314 (0x00DC)	P3_14	GPIO_PODR3:POD14	SOT10_1	SGA3_1	PPG9_TOUT0_1	OCU8_OTD0_1	-	TRACE_CTL_1	-
PPC_PCFGR315 (0x00DE)	P3_15	GPIO_PODR3:POD15	SCS100_1	SGO3_1	PPG9_TOUT2_1	OCU8_OTD1_1	TOT17_1	TRACE_CLK_1	TX2_1
PPC_PCFGR316 (0x00E0)	P3_16	GPIO_PODR3:POD16	-	SGA4_1	PPG10_TOUT0_1	OCU9_OTD0_1	-	-	-
PPC_PCFGR317 (0x00E2)	P3_17	GPIO_PODR3:POD17	-	SGO4_1	PPG10_TOUT2_1	OCU9_OTD1_1	TOT48_1	-	TX3_1
PPC_PCFGR318 (0x00E4)	P3_18	GPIO_PODR3:POD18	-	-	PPG11_TOUT0_1	OCU10_OTD0_1	-	-	-
PPC_PCFGR319 (0x00E6)	P3_19	GPIO_PODR3:POD19	SCK9_1	-	PPG11_TOUT2_1	OCU10_OTD1_1	-	-	-
PPC_PCFGR320 (0x00E8)	P3_20	GPIO_PODR3:POD20	SOT9_1	-	-	-	-	-	TX5_1
PPC_PCFGR321 (0x00EA)	P3_21	GPIO_PODR3:POD21	SCS90_1	-	-	-	TOT49_1	-	-
PPC_PCFGR322 (0x00EC)	P3_22	GPIO_PODR3:POD22	SCS91_1	-	-	-	-	-	-
PPC_PCFGR323 (0x00EE)	P3_23	GPIO_PODR3:POD23	-	-	-	-	SCS120_1	-	TX6_1
PPC_PCFGR324 (0x00F0)	P3_24	GPIO_PODR3:POD24	SOT2_1	-	-	-	-	PPG12_TOUT0_1	MDATA8
PPC_PCFGR325 (0x00F2)	P3_25	GPIO_PODR3:POD25	SCS20_1	-	-	-	I2S0_SD_1	PPG12_TOUT2_1	MDATA9
PPC_PCFGR326 (0x00F4)	P3_26	GPIO_PODR3:POD26	SCS21_1	-	-	-	I2S0_WS_1	PPG13_TOUT0_1	MDATA10

Register (Offset)	Port	Resource Functional Outputs							
		POF=0	POF=1	POF=2	POF=3	POF=4	POF=5	POF=6	POF=7
PPC_PCFGR327 (0x00F6)	P3_27	GPIO_PODR3:POD27	SCS22_1	-	-	-	I2S0_SCK_1	PPG13_TOUT2_1	MDATA11
PPC_PCFGR328 (0x00F8)	P3_28	GPIO_PODR3:POD28	-	-	-	-	-	PPG14_TOUT2_1	MDATA12
PPC_PCFGR329 (0x00FA)	P3_29	GPIO_PODR3:POD29	SCK3_1	-	-	-	-	PPG15_TOUT0_1	MDATA13
PPC_PCFGR330 (0x00FC)	P3_30	GPIO_PODR3:POD30	SOT3_1	-	-	-	-	PPG15_TOUT2_1	MDATA14
PPC_PCFGR331 (0x00FE)	P3_31	GPIO_PODR3:POD31	SCS30_1	-	-	-	-	-	MDATA15
PPC_PCFGR400 (0x0100)	P4_00	GPIO_PODR4:POD00	-	-	-	-	-	PPG16_TOUT0_0	-
PPC_PCFGR401 (0x0102)	P4_01	GPIO_PODR4:POD01	-	-	-	-	TX0_2	PPG16_TOUT2_0	-
PPC_PCFGR402 (0x0104)	P4_02	GPIO_PODR4:POD02	-	-	-	-	-	PPG17_TOUT0_0	-
PPC_PCFGR403 (0x0106)	P4_03	GPIO_PODR4:POD03	-	-	SCS170_1	-	-	PPG17_TOUT2_0	-
PPC_PCFGR404 (0x0108)	P4_04	GPIO_PODR4:POD04	-	TXEN_1	SCK17_1	-	TX3_2	PPG18_TOUT0_0	-
PPC_PCFGR405 (0x010A)	P4_05	GPIO_PODR4:POD05	-	TXD0_1	SOT17_1	-	-	PPG18_TOUT2_0	-
PPC_PCFGR406 (0x010C)	P4_06	GPIO_PODR4:POD06	-	-	-	-	-	PPG19_TOUT0_0	-
PPC_PCFGR407 (0x010E)	P4_07	GPIO_PODR4:POD07	-	-	-	-	-	PPG19_TOUT2_0	-
PPC_PCFGR408 (0x0110)	P4_08	GPIO_PODR4:POD08	-	-	-	-	-	PPG20_TOUT0_0	-
PPC_PCFGR409 (0x0112)	P4_09	GPIO_PODR4:POD09	-	-	-	-	-	PPG20_TOUT2_0	-
PPC_PCFGR410 (0x0114)	P4_10	GPIO_PODR4:POD10	-	-	-	-	-	PPG21_TOUT0_0	-
PPC_PCFGR411 (0x0116)	P4_11	GPIO_PODR4:POD11	-	-	-	-	-	PPG21_TOUT2_0	-
PPC_PCFGR412 (0x0118)	P4_12	GPIO_PODR4:POD12	-	-	-	-	-	PPG23_TOUT0_0	-
PPC_PCFGR413 (0x011A)	P4_13	GPIO_PODR4:POD13	-	-	-	-	-	PPG23_TOUT2_0	-

Register (Offset)	Port	Resource Functional Outputs							
		POF=0	POF=1	POF=2	POF=3	POF=4	POF=5	POF=6	POF=7
PPC_PCFGR414 (0x011C)	P4_14	GPIO_PODR4:POD14	-	-	-	-	-	PPG24_TOUT0_0	-
PPC_PCFGR415 (0x011E)	P4_15	GPIO_PODR4:POD15	-	-	-	-	-	PPG24_TOUT2_0	-
PPC_PCFGR416 (0x0120)	P4_16	GPIO_PODR4:POD16	-	-	-	-	-	PPG25_TOUT0_0	-
PPC_PCFGR417 (0x0122)	P4_17	GPIO_PODR4:POD17	-	-	-	-	SOT12_1	PPG25_TOUT2_0	-
PPC_PCFGR418 (0x0124)	P4_18	GPIO_PODR4:POD18	-	-	-	-	-	PPG26_TOUT0_0	-
PPC_PCFGR419 (0x0126)	P4_19	GPIO_PODR4:POD19	-	-	-	-	-	PPG26_TOUT2_0	-
PPC_PCFGR420 (0x0128)	P4_20	GPIO_PODR4:POD20	-	-	-	-	SOT16_1	PPG27_TOUT0_0	TX4_0
PPC_PCFGR421 (0x012A)	P4_21	GPIO_PODR4:POD21	-	-	-	-	SCK16_1	PPG27_TOUT2_0	-
PPC_PCFGR422 (0x012C)	P4_22	GPIO_PODR4:POD22	-	-	-	-	SCS161_1	-	-
PPC_PCFGR423 (0x012E)	P4_23	GPIO_PODR4:POD23	-	-	-	-	SCS160_1	PPG28_TOUT0_0	TX7_0
PPC_PCFGR424 (0x0130)	P4_24	GPIO_PODR4:POD24	-	-	-	-	-	PPG28_TOUT2_0	-
PPC_PCFGR425 (0x0132)	P4_25	GPIO_PODR4:POD25	-	-	-	-	-	PPG29_TOUT0_0	-
PPC_PCFGR426 (0x0134)	P4_26	GPIO_PODR4:POD26	-	-	-	-	-	PPG29_TOUT2_0	-
PPC_PCFGR427 (0x0136)	P4_27	GPIO_PODR4:POD27	-	-	-	-	-	PPG30_TOUT0_0	-
PPC_PCFGR428 (0x0138)	P4_28	GPIO_PODR4:POD28	-	-	-	-	-	PPG30_TOUT2_0	-
PPC_PCFGR429 (0x013A)	P4_29	GPIO_PODR4:POD29	-	-	-	-	-	PPG31_TOUT0_0	-
PPC_PCFGR430 (0x013C)	P4_30	GPIO_PODR4:POD30	-	-	-	-	-	PPG31_TOUT2_0	-
PPC_PCFGR431 (0x013E)	P4_31	GPIO_PODR4:POD31	SCK2_1	-	-	-	-	-	-

Notes:

- *The hyphen indicates that setting is prohibited. If setting the port will be operated as input independent on the register value of the GPIO_DDR.*
- *The register for P0_20 for POF exists though the port only supports input not supports output. The configuration of POF=0 for the port doesn't affect anything.*

3.2.2. The Port Additional Configuration

This section shows the port registers that must be set in addition to POF register setting.

- The I2SCLK Function Setting
  If the I2SCLK function on ports is used, set the value to 1 of PPC_PCFGR:PIE(Port input enable bit) of that port. Target ports: P0_06, P0_10

3.3. The Analog I/O Setting

If analog terminals assignment of the ADC and the LCDC are used, must set the following before using ADC and LCDC.

ADC

- The PPC_PCFGR:PIE register of target port makes disable digital input to set to 0.
- The corresponding POF. (typically set to 0)
- The GPIO_DDR register of target port. (typically set to 0)
- The PPC_PCFGR:PDE/PUE register of target port. (typically set 0)

LCDC

- The PPC_PCFGR:PIE register of target port makes disable digital input to set to 0.
- The corresponding POF should be set to 0.
- The GPIO_DDR register of target port makes input control to set to 0.
- The PPC_PCFGR:PDE/PUE register of target port makes disable Pull-Up/Pull-Down to set to 0.

Note:

- *Regarding the analog switch setting, see 'CHAPTER of 12/10/8-BIT Analog to Digital Converter' and 'CHAPTER of LCDC Controller'.*

3.4. Input Level Setting

This section shows the I/O port input level settings.

Pin No. of Package			PORT No.	Value of PPC_PCFGR:PIL[1:0]				Remarks
TEQFP-144	TEQFP-176	TEQFP-208		2'b00	2'b01	2'b10	2'b11	
2	2	2	P0_00	CMOS-s	Automotive	TTL	-	
3	3	3	P0_01	CMOS-s	Automotive	TTL	-	
4	4	4	P0_02	CMOS-s	Automotive	TTL	-	
5	5	5	P0_03	CMOS-s	Automotive	TTL	-	
-	-	6	P4_00	CMOS-s	Automotive	TTL	-	
-	-	7	P4_01	CMOS-s	Automotive	TTL	-	
-	-	8	P4_02	CMOS-s	Automotive	TTL	-	
6	6	9	P0_04	CMOS-s	Automotive	TTL	-	
7	7	10	P0_05	CMOS-s	Automotive	TTL	-	
8	8	11	P0_06	CMOS-s	Automotive	TTL	-	
9	9	12	P0_07	CMOS-s	Automotive	TTL	-	
10	10	13	P0_08	CMOS-s	Automotive	TTL	-	
11	11	14	P0_09	CMOS-s	Automotive	TTL	-	
-	-	15	P4_03	CMOS-s	Automotive	TTL	-	
-	-	16	P4_04	CMOS-s	Automotive	TTL	-	
-	-	17	P4_05	CMOS-s	Automotive	TTL	-	
12	12	18	P0_10	CMOS-s	Automotive	TTL	-	
13	13	19	P0_11	CMOS-s	Automotive	TTL	-	
14	14	20	P0_12	CMOS-s	Automotive	TTL	-	
15	15	21	P0_13	CMOS-s	Automotive	TTL	-	
-	19	25	P3_00	CMOS-s	Automotive	TTL	-	
-	20	26	P3_01	CMOS-s	Automotive	TTL	-	
-	21	27	P3_02	CMOS-s	Automotive	TTL	-	
-	22	28	P3_03	CMOS-s	Automotive	TTL	-	
-	23	29	P3_04	CMOS-s	Automotive	TTL	-	
-	-	30	P4_06	CMOS-s	Automotive	TTL	-	
-	-	31	P4_07	CMOS-s	Automotive	TTL	-	
-	24	32	P3_05	CMOS-s	Automotive	TTL	-	
-	25	33	P3_06	CMOS-s	Automotive	TTL	-	
19	26	34	P0_14	CMOS-s	Automotive	TTL	-	
20	27	35	P0_15	CMOS-s	Automotive	TTL	-	
21	28	36	P0_16	CMOS-s	Automotive	TTL	-	
-	29	37	P3_07	CMOS-s	Automotive	TTL	-	
22	30	38	P0_17	CMOS-s	Automotive	TTL	-	
23	31	39	P0_18	CMOS-s	Automotive	TTL	-	
24	32	40	P0_19	CMOS-s	Automotive	TTL	-	
25	33	41	P0_20	CMOS-s	Automotive	TTL	-	
38	46	54	P0_21	CMOS-s	-	TTL	-	
39	47	55	P0_22	CMOS-s	-	TTL	-	
40	48	56	P0_23	CMOS-s	-	TTL	-	
41	49	57	P0_24	CMOS-s	-	TTL	-	
42	50	58	P0_25	CMOS-s	-	TTL	-	
44	52	60	P0_26	CMOS-s	-	TTL	-	
47	55	63	P0_27	CMOS-s	-	TTL	-	

Pin No. of Package			PORT No.	Value of PPC_PCFGR:PIL[1:0]				Remarks
TEQFP-144	TEQFP-176	TEQFP-208		2'b00	2'b01	2'b10	2'b11	
48	56	64	P0_28	CMOS-s	-	TTL	-	
49	57	65	P0_29	CMOS-s	-	TTL	-	
50	58	66	P0_30	CMOS-s	-	TTL	-	
51	59	67	P0_31	CMOS-s	-	TTL	-	
54	62	70	P1_00	CMOS-s	-	-	Media LB	
55	63	71	P1_01	CMOS-s	-	-	Media LB	
56	64	72	P1_02	CMOS-s	-	-	Media LB	
-	-	77	P4_08	CMOS-s	Automotive	-	-	
-	-	78	P4_09	CMOS-s	Automotive	-	-	
-	-	79	P4_10	CMOS-s	Automotive	-	-	
-	-	80	P4_11	CMOS-s	Automotive	-	-	
-	69	81	P3_08	CMOS-s	Automotive	-	-	
-	70	82	P3_09	CMOS-s	Automotive	-	-	
-	71	83	P3_10	CMOS-s	Automotive	-	-	
-	72	84	P3_11	CMOS-s	Automotive	-	-	
61	73	85	P1_03	CMOS-s	Automotive	-	-	
62	74	86	P1_04	CMOS-s	Automotive	-	-	
-	75	87	P3_12	CMOS-s	Automotive	-	-	
-	76	88	P3_13	CMOS-s	Automotive	-	-	
63	77	89	P1_05	CMOS-s	Automotive	-	-	
64	78	90	P1_06	CMOS-s	Automotive	-	-	
-	79	91	P3_14	CMOS-s	Automotive	-	-	
-	80	92	P3_15	CMOS-s	Automotive	-	-	
65	81	93	P1_07	CMOS-s	Automotive	-	-	
66	82	94	P1_08	CMOS-s	Automotive	-	-	
67	83	95	P1_09	CMOS-s	Automotive	TTL	-	
-	-	96	P4_12	CMOS-s	Automotive	-	-	
-	-	97	P4_13	CMOS-s	Automotive	-	-	
68	84	98	P1_10	CMOS-s	Automotive	TTL	-	
-	-	99	P4_14	CMOS-s	Automotive	-	-	
-	-	100	P4_15	CMOS-s	Automotive	-	-	
70	86	102	P1_11	CMOS-s	Automotive	-	-	
71	87	103	P1_12	CMOS-s	Automotive	-	-	
81	97	113	P1_13	CMOS-s	Automotive	-	-	
-	98	114	P3_16	CMOS-s	Automotive	-	-	
-	99	115	P3_17	CMOS-s	Automotive	-	-	
-	-	116	P4_16	CMOS-s	Automotive	-	-	
-	105	122	P3_18	CMOS-s	Automotive	-	-	
-	106	123	P3_19	CMOS-s	Automotive	-	-	
-	107	124	P3_20	CMOS-s	Automotive	-	-	
-	108	125	P3_21	CMOS-s	Automotive	-	-	
-	109	126	P3_22	CMOS-s	Automotive	-	-	
-	110	127	P3_23	CMOS-s	Automotive	-	-	
87	111	128	P1_14	CMOS-s	Automotive	-	-	
88	112	129	P1_15	CMOS-s	Automotive	TTL	-	
-	-	130	P4_17	CMOS-s	Automotive			
-	-	131	P4_18	CMOS-s	Automotive			
-	-	136	P4_19	CMOS-s	Automotive	-	-	
-	-	137	P4_20	CMOS-s	Automotive	-	-	

Pin No. of Package			PORT No.	Value of PPC_PCFGR:PIL[1:0]				Remarks
TEQFP-144	TEQFP-176	TEQFP-208		2'b00	2'b01	2'b10	2'b11	
-	-	138	P4_21	CMOS-s	Automotive	-	-	
93	117	139	P1_16	CMOS-s	Automotive	TTL	-	
-	-	140	P4_22	CMOS-s	Automotive	-	-	
-	-	141	P4_23	CMOS-s	Automotive	-	-	
100	124	148	P1_17	CMOS-s	Automotive	-	-	
101	125	149	P1_18	CMOS-s	Automotive	-	-	
102	126	150	P1_19	CMOS-s	Automotive	-	-	
103	127	151	P1_20	CMOS-s	Automotive	-	-	
104	128	152	P1_21	CMOS-s	Automotive	-	-	
105	129	153	P1_22	CMOS-s	Automotive	-	-	
106	130	154	P1_23	CMOS-s	Automotive	-	-	
107	131	155	P1_24	CMOS-s	Automotive	-	-	
110	134	158	P1_25	CMOS-s	Automotive	-	-	
-	-	159	P4_24	CMOS-s	Automotive	-	-	
111	135	160	P1_26	CMOS-s	Automotive	-	-	
112	136	161	P1_27	CMOS-s	Automotive	-	-	
113	137	162	P1_28	CMOS-s	Automotive	-	-	
-	-	163	P4_25	CMOS-s	Automotive	-	-	
114	138	164	P1_29	CMOS-s	Automotive	-	-	
115	139	165	P1_30	CMOS-s	Automotive	-	-	
116	140	166	P1_31	CMOS-s	Automotive	-	-	
-	-	167	P4_26	CMOS-s	Automotive	-	-	
117	141	168	P2_00	CMOS-s	Automotive	-	-	
120	144	171	P2_01	CMOS-s	Automotive	-	-	
-	-	172	P4_27	CMOS-s	Automotive	-	-	
121	145	173	P2_02	CMOS-s	Automotive	-	-	
122	146	174	P2_03	CMOS-s	Automotive	-	-	
123	147	175	P2_04	CMOS-s	Automotive	-	-	
-	-	176	P4_28	CMOS-s	Automotive	-	-	
124	148	177	P2_05	CMOS-s	Automotive	-	-	
125	149	178	P2_06	CMOS-s	Automotive	-	-	
126	150	179	P2_07	CMOS-s	Automotive	-	-	
-	-	180	P4_29	CMOS-s	Automotive	-	-	
127	151	181	P2_08	CMOS-s	Automotive	-	-	
-	-	184	P4_30	CMOS-s	Automotive	-	-	
-	-	185	P4_31	CMOS-s	Automotive	-	-	
-	154	186	P3_24	CMOS-s	Automotive	TTL	-	
-	155	187	P3_25	CMOS-s	Automotive	TTL	-	
-	156	188	P3_26	CMOS-s	Automotive	TTL	-	
-	157	189	P3_27	CMOS-s	Automotive	TTL	-	
130	158	190	P2_09	CMOS-s	Automotive	TTL	-	
-	162	194	P3_28	CMOS-s	Automotive	TTL	-	
-	163	195	P3_29	CMOS-s	Automotive	TTL	-	
-	164	196	P3_30	CMOS-s	Automotive	TTL	-	
-	165	197	P3_31	CMOS-s	Automotive	TTL	-	
134	166	198	P2_10	CMOS-s	Automotive	TTL	-	
135	167	199	P2_11	CMOS-s	Automotive	TTL	-	
136	168	200	P2_12	CMOS-s	Automotive	TTL	-	
137	169	201	P2_13	CMOS-s	Automotive	TTL	-	

Pin No. of Package			PORT No.	Value of PPC_PCFGR:PIL[1:0]				Remarks
TEQFP-144	TEQFP-176	TEQFP-208		2'b00	2'b01	2'b10	2'b11	
138	170	202	P2_14	CMOS-s	Automotive	TTL	-	
139	171	203	P2_15	CMOS-s	Automotive	TTL	-	
140	172	204	P2_16	CMOS-s	Automotive	TTL	-	
141	173	205	P2_17	CMOS-s	Automotive	TTL	-	
142	174	206	P2_18	CMOS-s	Automotive	TTL	-	
143	175	207	P2_19	CMOS-s	Automotive	TTL	-	

Note:

- *The hyphen of "Value of PPC_PCFGR:PIL[1:0]" indicates that setting is prohibited.*
- *"CMOS-s" is CMOS hysteresis input level.*
- *"Automotive" is Automotive input level.*
- *"TTL" is TTL input level.*
- *"MediaLB" is MediaLB input level.*
- *To get detailed information about the input level, see the DC characteristics of the Datasheet.*

3.5. Output Drive Capacity Setting

This section shows the I/O port output drive capacity settings.
Note that the drive capacity depends on power supply voltage. The table only describes the representative value when 5 V is used as standard value of power supply usage. Please see the actual characteristics in datasheet.

Pin No. of Package			PORT	Value of PPC_PCFGR:ODR[1:0]				Remarks
TEQFP-144	TEQFP-176	TEQFP-208	No.	2'b00	2'b01	2'b10	2'b11	
2	2	2	P0_00	1 mA	2 mA	5 mA	-	
3	3	3	P0_01	1 mA	2 mA	5 mA	-	
4	4	4	P0_02	1 mA	2 mA	5 mA	-	*1-2
5	5	5	P0_03	1 mA	2 mA	5 mA	-	*1-2
-	-	6	P4_00	1 mA	2 mA	5 mA	-	
-	-	7	P4_01	1 mA	2 mA	5 mA	-	
-	-	8	P4_02	1 mA	2 mA	5 mA	-	
6	6	9	P0_04	1 mA	2 mA	5 mA	-	
7	7	10	P0_05	1 mA	2 mA	5 mA	-	
8	8	11	P0_06	1 mA	2 mA	5 mA	-	
9	9	12	P0_07	1 mA	2 mA	5 mA	-	
10	10	13	P0_08	1 mA	2 mA	5 mA	-	
11	11	14	P0_09	1 mA	2 mA	5 mA	-	
-	-	15	P4_03	1 mA	2 mA	5 mA	-	
-	-	16	P4_04	1 mA	2 mA	5 mA	-	
-	-	17	P4_05	1 mA	2 mA	5 mA	-	
12	12	18	P0_10	1 mA	2 mA	5 mA	-	
13	13	19	P0_11	1 mA	2 mA	5 mA	-	
14	14	20	P0_12	1 mA	2 mA	5 mA	-	
15	15	21	P0_13	1 mA	2 mA	5 mA	-	
-	19	25	P3_00	1 mA	2 mA	5 mA	-	
-	20	26	P3_01	1 mA	2 mA	5 mA	-	
-	21	27	P3_02	1 mA	2 mA	5 mA	-	
-	22	28	P3_03	1 mA	2 mA	5 mA	-	
-	23	29	P3_04	1 mA	2 mA	5 mA	-	
-	-	30	P4_06	1 mA	2 mA	5 mA	-	
-	-	31	P4_07	1 mA	2 mA	5 mA	-	
-	24	32	P3_05	1 mA	2 mA	5 mA	-	
-	25	33	P3_06	1 mA	2 mA	5 mA	-	
19	26	34	P0_14	1 mA	2 mA	5 mA	-	
20	27	35	P0_15	1 mA	2 mA	5 mA	-	
21	28	36	P0_16	1 mA	2 mA	5 mA	-	
-	29	37	P3_07	1 mA	2 mA	5 mA	-	
22	30	38	P0_17	1 mA	2 mA	5 mA	-	
23	31	39	P0_18	1 mA	2 mA	5mA	-	
24	32	40	P0_19	1 mA	2 mA	5 mA	-	
25	33	41	P0_20	-	-	-	-	Input only
38	46	54	P0_21	2 mA	5 mA	6 mA	15 mA	
39	47	55	P0_22	2 mA	5 mA	6 mA	15 mA	
40	48	56	P0_23	2 mA	5 mA	6 mA	15 mA	
41	49	57	P0_24	2 mA	5 mA	6 mA	15 mA	
42	50	58	P0_25	2 mA	5 mA	6 mA	15 mA	

Pin No. of Package			PORT No.	Value of PPC_PCFGR:ODR[1:0]				Remarks
TEQFP-144	TEQFP-176	TEQFP-208		2'b00	2'b01	2'b10	2'b11	
44	52	60	P0_26	2 mA	5 mA	6 mA	15 mA	
47	55	63	P0_27	2 mA	5 mA	6 mA	15 mA	
48	56	64	P0_28	2 mA	5 mA	6 mA	15 mA	
49	57	65	P0_29	2 mA	5 mA	6 mA	15 mA	
50	58	66	P0_30	2 mA	5 mA	6 mA	15 mA	
51	59	67	P0_31	2 mA	5 mA	6 mA	15 mA	
54	62	70	P1_00	2 mA	5 mA	6 mA	15 mA	
55	63	71	P1_01	2 mA	5 mA	6 mA	15 mA	
56	64	72	P1_02	2 mA	5 mA	6 mA	15 mA	
-	-	77	P4_08	1 mA	2 mA	-	5 mA	
-	-	78	P4_09	1 mA	2 mA	-	5 mA	
-	-	79	P4_10	1 mA	2 mA	-	5 mA	
-	-	80	P4_11	1 mA	2 mA	-	5 mA	
-	69	81	P3_08	1 mA	2 mA	-	5 mA	
-	70	82	P3_09	1 mA	2 mA	-	5 mA	
-	71	83	P3_10	1 mA	2 mA	-	5 mA	
-	72	84	P3_11	1 mA	2 mA	-	5 mA	
61	73	85	P1_03	1 mA	2 mA	-	5 mA	
62	74	86	P1_04	1 mA	2 mA	-	5 mA	*1-1
-	75	87	P3_12	1 mA	2 mA	-	5 mA	
-	76	88	P3_13	1 mA	2 mA	-	5 mA	
63	77	89	P1_05	1 mA	2 mA	-	5 mA	*1-1
64	78	90	P1_06	1 mA	2 mA	-	5 mA	
-	79	91	P3_14	1 mA	2 mA	-	5 mA	
-	80	92	P3_15	1 mA	2 mA	-	5 mA	
65	81	93	P1_07	1 mA	2 mA	-	5 mA	
66	82	94	P1_08	1 mA	2 mA	-	5 mA	
67	83	95	P1_09	1 mA	2 mA	-	5 mA	*1-1
-	-	96	P4_12	1 mA	2 mA	-	5 mA	
-	-	97	P4_13	1 mA	2 mA	-	5 mA	
68	84	98	P1_10	1 mA	2 mA	-	5 mA	*1-1
-	-	99	P4_14	1 mA	2 mA	-	5 mA	
-	-	100	P4_15	1 mA	2 mA	-	5 mA	
70	86	102	P1_11	1 mA	2 mA	-	5 mA	
71	87	103	P1_12	1 mA	2 mA	-	5 mA	
81	97	113	P1_13	1 mA	2 mA	-	5 mA	
-	98	114	P3_16	1 mA	2 mA	-	5 mA	
-	99	115	P3_17	1 mA	2 mA	-	5 mA	
-	-	116	P4_16	1 mA	2 mA	-	5 mA	
-	105	122	P3_18	1 mA	2 mA	-	5 mA	
-	106	123	P3_19	1 mA	2 mA	-	5 mA	
-	107	124	P3_20	1 mA	2 mA	-	5 mA	
-	108	125	P3_21	1 mA	2 mA	-	5 mA	
-	109	126	P3_22	1 mA	2 mA	-	5 mA	
-	110	127	P3_23	1 mA	2 mA	-	5 mA	
87	111	128	P1_14	1 mA	2 mA	-	5 mA	
88	112	129	P1_15	1 mA	2 mA	-	5 mA	*1-1
-	-	130	P4_17	1 mA	2 mA	-	5 mA	

Pin No. of Package			PORT	Value of PPC_PCFGR:ODR[1:0]				Remarks
TEQFP-144	TEQFP-176	TEQFP-208	No.	2'b00	2'b01	2'b10	2'b11	
-	-	131	P4_18	1 mA	2 mA	-	5 mA	
-	-	136	P4_19	1 mA	2 mA	-	5 mA	
-	-	137	P4_20	1 mA	2 mA	-	5 mA	
-	-	138	P4_21	1 mA	2 mA	-	5 mA	
93	117	139	P1_16	1 mA	2 mA	-	5 mA	*1-1
-	-	140	P4_22	1 mA	2 mA	-	5 mA	
-	-	141	P4_23	1 mA	2 mA	-	5 mA	
100	124	148	P1_17	1 mA	2 mA	5 mA	30 mA	
101	125	149	P1_18	1 mA	2 mA	5 mA	30 mA	
102	126	150	P1_19	1 mA	2 mA	5 mA	30 mA	
103	127	151	P1_20	1 mA	2 mA	5 mA	30 mA	*1-2
104	128	152	P1_21	1 mA	2 mA	5 mA	30 mA	*1-2
105	129	153	P1_22	1 mA	2 mA	5 mA	30 mA	
106	130	154	P1_23	1 mA	2 mA	5 mA	30 mA	
107	131	155	P1_24	1 mA	2 mA	5 mA	30 mA	*1-2
110	134	158	P1_25	1 mA	2 mA	5 mA	30 mA	*1-2
-	-	159	P4_24	1 mA	2 mA	5 mA	30 mA	
111	135	160	P1_26	1 mA	2 mA	5 mA	30 mA	
112	136	161	P1_27	1 mA	2 mA	5 mA	30 mA	
113	137	162	P1_28	1 mA	2 mA	5 mA	30 mA	
-	-	163	P4_25	1 mA	2 mA	5 mA	30 mA	
114	138	164	P1_29	1 mA	2 mA	5 mA	30 mA	*1-2
115	139	165	P1_30	1 mA	2 mA	5 mA	30 mA	*1-2
116	140	166	P1_31	1 mA	2 mA	5 mA	30 mA	
-	-	167	P4_26	1 mA	2 mA	5 mA	30 mA	
117	141	168	P2_00	1 mA	2 mA	5 mA	30 mA	
120	144	171	P2_01	1 mA	2 mA	5 mA	30 mA	*1-2
-	-	172	P4_27	1 mA	2 mA	5 mA	30 mA	
121	145	173	P2_02	1 mA	2 mA	5 mA	30 mA	*1-2
122	146	174	P2_03	1 mA	2 mA	5 mA	30 mA	
123	147	175	P2_04	1 mA	2 mA	5 mA	30 mA	
-	-	176	P4_28	1 mA	2 mA	5 mA	30 mA	
124	148	177	P2_05	1 mA	2 mA	5 mA	30 mA	
125	149	178	P2_06	1 mA	2 mA	5 mA	30 mA	*1-2
126	150	179	P2_07	1 mA	2 mA	5 mA	30 mA	*1-2
-	-	180	P4_29	1 mA	2 mA	5 mA	30 mA	
127	151	181	P2_08	1 mA	2 mA	5 mA	30 mA	
-	-	184	P4_30	1 mA	2 mA	5 mA	30 mA	
-	-	185	P4_31	1 mA	2 mA	5 mA	30 mA	
-	154	186	P3_24	1 mA	2 mA	5 mA	-	
-	155	187	P3_25	1 mA	2 mA	5 mA	-	
-	156	188	P3_26	1 mA	2 mA	5 mA	-	
-	157	189	P3_27	1 mA	2 mA	5 mA	-	
130	158	190	P2_09	1 mA	2 mA	5 mA	-	
-	162	194	P3_28	1 mA	2 mA	5 mA	-	
-	163	195	P3_29	1 mA	2 mA	5 mA	-	
-	164	196	P3_30	1 mA	2 mA	5 mA	-	
-	165	197	P3_31	1 mA	2 mA	5 mA	-	

Pin No. of Package			PORT No.	Value of PPC_PCFGR:ODR[1:0]				Remarks
TEQFP-144	TEQFP-176	TEQFP-208		2'b00	2'b01	2'b10	2'b11	
134	166	198	P2_10	1 mA	2 mA	5 mA	-	
135	167	199	P2_11	1 mA	2 mA	5 mA	-	
136	168	200	P2_12	1 mA	2 mA	5 mA	15 mA	
137	169	201	P2_13	1 mA	2 mA	5 mA	-	
138	170	202	P2_14	1 mA	2 mA	5 mA	-	*1-2
139	171	203	P2_15	1 mA	2 mA	5 mA	-	*1-2
140	172	204	P2_16	1 mA	2 mA	5 mA	-	
141	173	205	P2_17	1 mA	2 mA	5 mA	-	
142	174	206	P2_18	1 mA	2 mA	5 mA	-	
143	175	207	P2_19	1 mA	2 mA	5 mA	-	

See the note at the next page.

Note:

- *The hyphen of "Value of PPC_PCFGR:ODR[1:0]" indicates that setting is prohibited.*
- **1-1 When the PPC_PCFGR:POF[2:0] value setting is "3"(SDA or SCL function setting), regardless of the value of the PPC_PCGR:ODR[1:0], The drive capacity will be "I2C". Then, the port status is to be Pseudo Open Drain, and IOL is to be 3 mA.*
- **1-2 When the PPC_PCFGR:POF[2:0] value setting is "3","6"(SDA or SCL function setting), the port status is to be Pseudo Open Drain, and IOL is to be the configured value for ODR.*

To get detailed information about the drive capability, see the DC characteristics of the Datasheet.

3.6. Port Status

3.6.1. Hi-z Control

Traveo™ Platform Hardware Manual has description of System Special Setting Register (SYSC0_SPECFGR).

The [bit23] PSSPADCTRL: PSS-time port configuring bit should be configured as below.

- 0: Do not perform Hi-z control.
- 1: Perform Hi-z control.

Notes:

- *At RUN mode, if configured as Hi-z control, it doesn't affect the port status immediately, but after executing WFI instruction to update the profile registers, then it turns out Hi-z status during PSS mode.*
- *As opposite control from PSS to RUN, the port status of Hi-z will automatically be released without reconfiguration of SYSC0_SPECFGR.PSSPADCTRL = 0.*

3.6.2. Port Status Hold during PSS Mode

All of the GPIO except ports in VCC3 area can be kept the port status during PSS mode by System Special Setting Register (SYSC0_SPECFGR).

The [bit31] to [bit24] HOLDIO_PDx: HOLD data latch bit should be configured as below.

- 0: Do not retain control.
- 1: Retain control.

Notes:

- *During MCU RUN mode, I/O port status will be latched immediately after SYSC0_SPECFGR.HOLDIO_PDx= 1 (Retain control) configured.*
- *After SYSC0_SPECFGR.HOLDIO_PDx= 1,the status of followings will be latched. Please note that PID is not included, that is, input data cannot be latched.*
  - *PPC_PCFGRijj POD, POE, PIL, PUE, PDE, ODR, NFE, and POF. (excluding PID: Input data cannot be latched)*
  - *RIC_RESINx.PORTSEL and RESSEL*
- *At turning from MCU PSS mode to RUN, the latched status will not be released automatically. Configuration SYSC0_SPECFGR.HOLDIO_PDx = 0 should be necessary for releasing the status.*

3.7. Function Port Group

A port group specifies an I/O port combination for a peripheral function. A peripheral function has some port groups and should be configured and used within a port group of them which is defined in the following table in order to satisfy AC specification.

Do not take a port combination which is not described in the following table as a port group.

Table 11-1: Function Port Group List

Function	Port Group
Multi-Function Serial Ch.0	Group1 -P1_03(SIN0_0) -P1_05(SOT0_0) -P1_04(SCK0_0) -P1_06(SCS00_0) Group2 -P0_04(SIN0_1) -P0_05(SOT0_1) -P0_06(SCK0_1) -P0_07(SCS00_1)
Multi-Function Serial Ch.1	Group1 -P0_01(SIN1_0) -P0_02(SCK1_0) -P0_03(SOT1_0) -P0_04(SCS10_0) -P0_05(SCS11_0) -P0_06(SCS12_0) -P0_07(SCS13_0) Group2 -P3_00(SIN1_1) -P3_01(SCK1_1) -P3_02(SOT1_1) -P3_03(SCS10_1) -P3_04(SCS11_1) -P3_05(SCS12_1) -P3_06(SCS13_1)
Multi-Function Serial Ch.2	Group1 -P0_21(SIN2_0) -P0_23(SOT2_0) -P0_22(SCK2_0) -P0_24(SCS20_0) -P0_25(SCS21_0) -P0_26(SCS22_0) -P0_27(SCS23_0) Group2 -P4_30(SIN2_1) -P3_24(SOT2_1) -P4_31(SCK2_1) -P3_25(SCS20_1) -P3_26(SCS21_1) -P3_27(SCS22_1) -P2_09(SCS23_1)

Function	Port Group
Multi-Function Serial Ch.3	Group1 -P0_28(SIN3_0) -P0_29(SCK3_0) -P0_30(SOT3_0) -P0_31(SCS30_0) -P1_00(SCS31_0) -P1_01(SCS32_0) -P1_02(SCS33_0) Group2 -P3_28(SIN3_1) -P3_29(SCK3_1) -P3_30(SOT3_1) -P3_31(SCS30_1) -P2_10(SCS31_1) -P2_11(SCS32_1) -P2_12(SCS33_1)
Multi-Function Serial Ch.4	Group1 -P2_13(SIN4_0) -P2_14(SCK4_0) -P2_15(SOT4_0) -P2_16(SCS40_0) -P2_17(SCS41_0) -P2_18(SCS42_0) -P2_19(SCS43_0) Group2 -P3_05(SIN4_1) -P0_14(SCK4_1) -P3_06(SOT4_1) -P0_15(SCS40_1) -P0_16(SCS41_1) -P3_07(SCS42_1) -P0_17(SCS43_1)
Multi-Function Serial Ch.8	Group1 -P1_19(SIN8_0) -P1_20(SCK8_0) -P1_21(SOT8_0) -P1_22(SCS80_0) Group2 -P3_08(SIN8_1) -P3_09(SCK8_1) -P3_10(SOT8_1) -P3_11(SCS80_1) Group3 -P0_27(SIN8_2) -P0_28(SCK8_2) -P0_29(SOT8_2) -P0_30(SCS80_2)
Multi-Function Serial Ch.9	Group1 -P1_23(SIN9_0) -P1_24(SCK9_0) -P1_25(SOT9_0) -P1_26(SCS90_0) -P1_27(SCS91_0) Group2 -P3_18(SIN9_1) -P3_19(SCK9_1) -P3_20(SOT9_1) -P3_21(SCS90_1) -P3_22(SCS91_1) Group3 -P0_21(SIN9_2) -P0_22(SOT9_2) -P0_23(SCK9_2) -P0_24(SCS90_2) -P0_25(SCS91_2)

Function	Port Group
Multi-Function Serial Ch.10	Group1 -P1_28(SIN10_0) -P1_29(SCK10_0) -P1_30(SOT10_0) -P1_31(SCS100_0) Group2 -P3_12(SIN10_1) -P3_13(SCK10_1) -P3_14(SOT10_1) -P3_15(SCS100_1)
Multi-Function Serial Ch.11	Group1 -P2_00(SIN11_0) -P2_01(SCK11_0) -P2_02(SOT11_0) -P2_03(SCS110_0) -P2_04(SCS111_0)
Multi-Function Serial Ch.12	Group1 -P2_05(SIN12_0) -P2_06(SCK12_0) -P2_07(SOT12_0) -P2_08(SCS120_0) Group2 -P4_18(SIN12_1) -P1_15(SCK12_1) -P4_17(SOT12_1) -P3_23(SCS120_1)
Multi-Function Serial Ch.16	Group1 -P1_08(SIN16_0) -P1_09(SCK16_0) -P1_10(SOT16_0) -P1_11(SCS160_0) -P1_12(SCS161_0) Group2 -P4_19(SIN16_1) -P4_21(SCK16_1) -P4_20(SOT16_1) -P4_23(SCS160_1) -P4_22(SCS161_1)
Multi-Function Serial Ch.17	Group1 -P1_14(SIN17_0) -P1_16(SOT17_0) -P1_15(SCK17_0) -P1_17(SCS170_0) -P1_18(SCS171_0) Group2 -P0_09(SIN17_1) -P4_05(SOT17_1) -P4_04(SCK17_1) -P4_03(SCS170_1) -P0_10(SCS171_1)
CAN Ch.0	Group1 -P1_05(RX0_0) -P1_06(TX0_0) Group2 -P3_09(RX0_1) -P3_10(TX0_1) Group3 -P4_00(RX0_2) -P4_01(TX0_2)

Function	Port Group
CAN Ch.1	Group1 -P1_07(RX1_0) -P1_08(TX1_0) Group2 -P3_11(RX1_1) -P3_12(TX1_1)
CAN Ch.2	Group1 -P1_11(RX2_0) -P1_12(TX2_0) Group2 -P3_14(RX2_1) -P3_15(TX2_1)
CAN Ch.3	Group1 -P1_13(RX3_0) -P1_14(TX3_0) Group2 -P3_16(RX3_1) -P3_17(TX3_1) Group3 -P4_03(RX3_2) -P4_04(TX3_2)
CAN Ch.4	Group1 -P4_19(RX4_0) -P4_20(TX4_0)
CAN Ch.5	Group1 -P1_17(RX5_0) -P1_18(TX5_0) Group2 -P3_19(RX5_1) -P3_20(TX5_1)
CAN Ch.6	Group1 -P1_21(RX6_0) -P1_22(TX6_0) Group2 -P3_22(RX6_1) -P3_23(TX6_1)
CAN Ch.7	Group1 -P4_22(RX7_0) -P4_23(TX7_0) Group2 -P2_18(RX7_1) -P2_19(TX7_1)

3.8. Key Code Register

The write access to I/O Port register is protected by Key Code Register.

Table 11-2 Relationship between I/O Port Register and Key Code Register

I/O Port Register	Key Code Register	Remark
Data Direction Register (GPIO_DDRi)	GPIO Key Code Register (GPIO_KEYCDR)	-
Data Direction Set Register (GPIO_DDSRi)	GPIO Key Code Register (GPIO_KEYCDR)	-
Data Direction Clear Register (GPIO_DDCRi)	GPIO Key Code Register (GPIO_KEYCDR)	-
Port Output Data Register (GPIO_PODRi)	-	-
Port Output Set Register (GPIO_POSRi)	-	-
Port Output Clear Register (GPIO_POCRi)	-	-
Port Input Enable Register (GPIO_PORTEN)	GPIO Key Code Register (GPIO_KEYCDR)	-
Port Input Data Register (GPIO_PIDRi)	-	-
Port Enable Register (GPIO_PPERi)	-	-
GPIO Key Code Register (GPIO_KEYCDR)	-	-
Port Setting Register (PPC_PCFGRijj)	PPC Key Code Register (PPC_KEYCDR)	-
PPC Key Code Register (PPC_KEYCDR)	-	-
Resource Input Setting Register (RIC_RESINn)	RIC Key Code Register (RIC_KEYCDR)	-
RIC Key Code Register (RIC_KEYCDR)	-	-

4. Registers

See the common information of the registers on the Traveo™ Platform Hardware Manual.

5. Configuration Procedure

The configuration procedure of resource I/O port is described.

Category	I/O Direction	Referable Procedure	Remark
Resource I/O Port	Both Direction	See 5.1	
	Input	See 5.2	
	Output	See 5.3	
Port Function	Input	See 5.4	
	Output	See 5.5	
Analog Function	Input or Output	See 5.6	

Notes:

- *Glitch at output port may sometime be observed when the following case.*
  1. *from input to output*
  2. *from output to input*
  3. *from input to input*
  4. *from output to output*

5.1. Resource I/O Port (Both Direction)

Figure 11-2: Procedure of Resource I/O Port (Both Direction)

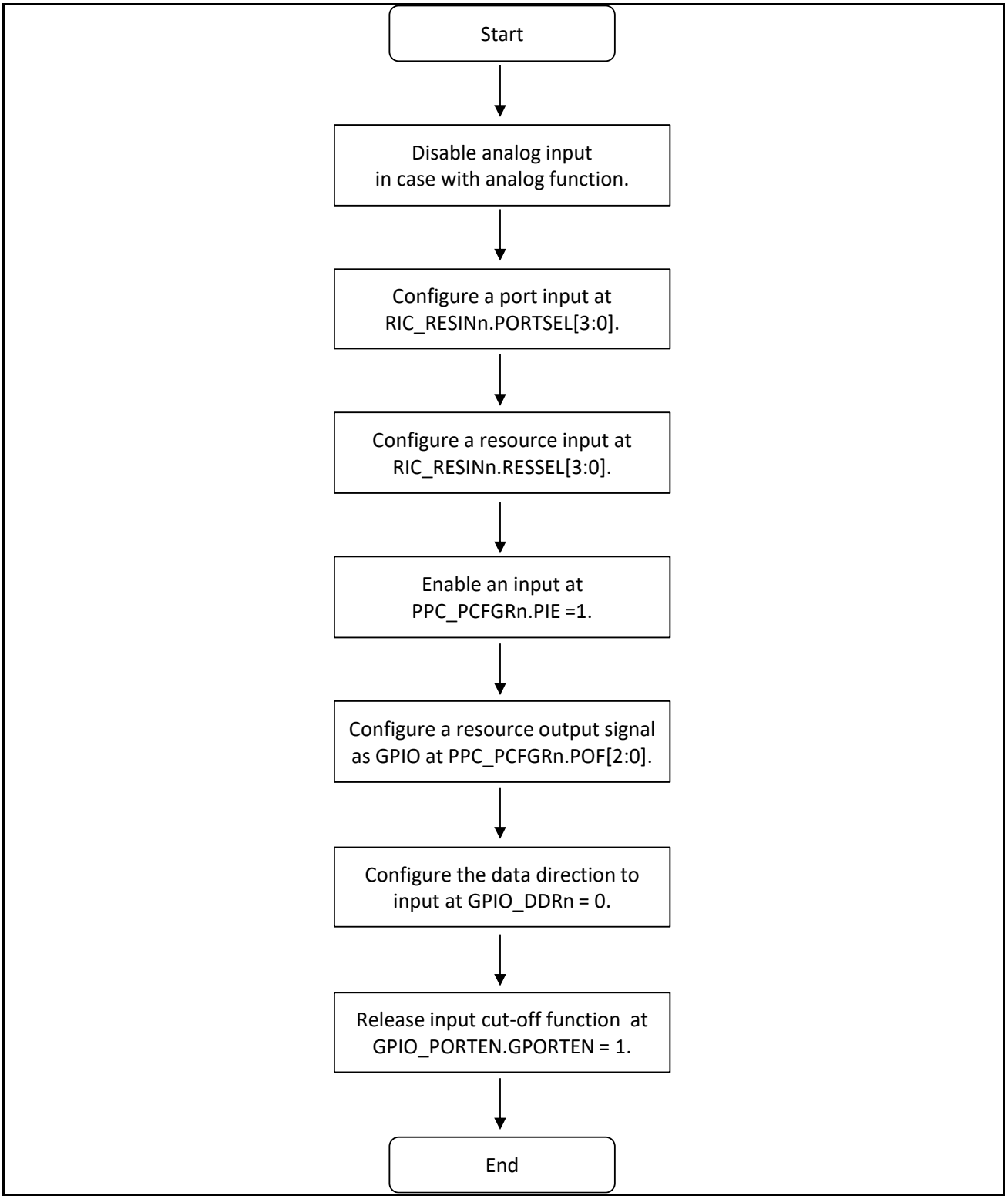


Notes:

- *The dedicated input/output direction configuration should be necessary for SCK (MFS).*

5.2. Resource Input

Figure 11-3: Procedure of Resource Input

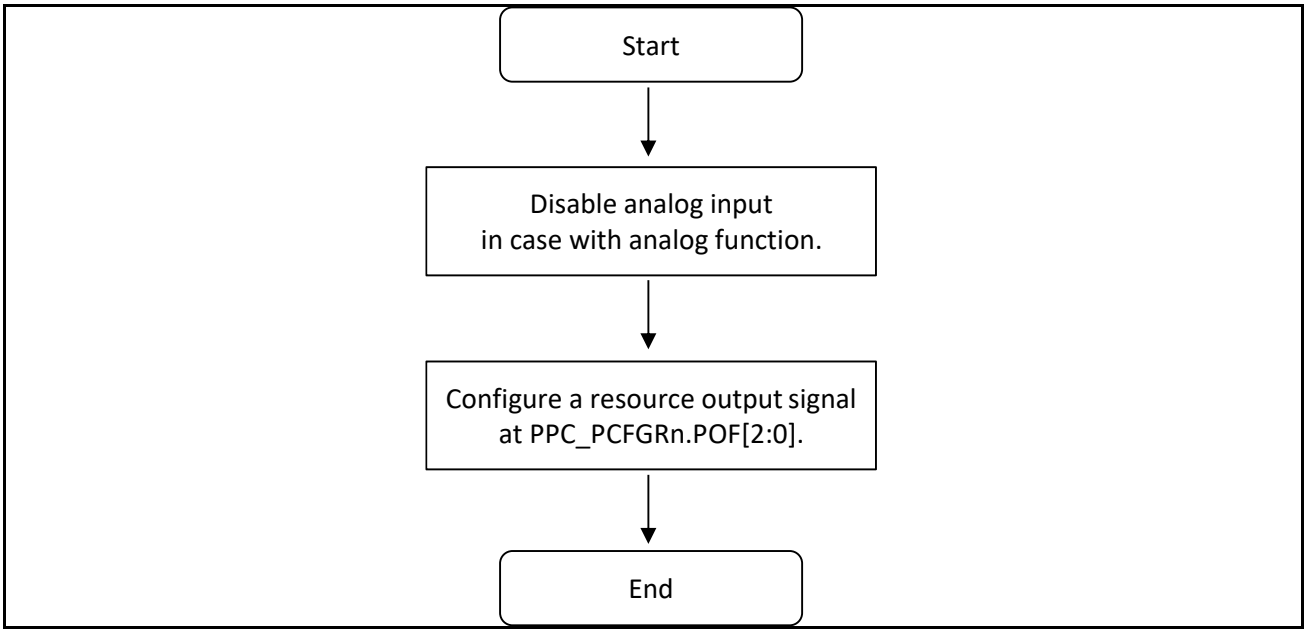


Notes:

- *RIC_RESINn register is only used for a resource which supports relocation function or resource input function from the other resource. The RIC_RESINn should be configured as above.*
- *A resource which supports neither relocation nor the other resource input doesn't have its RIC_RESINn register. Configurations can be skipped in above procedure.*

5.3. Resource Output

Figure 11-4: Procedure of Resource Output

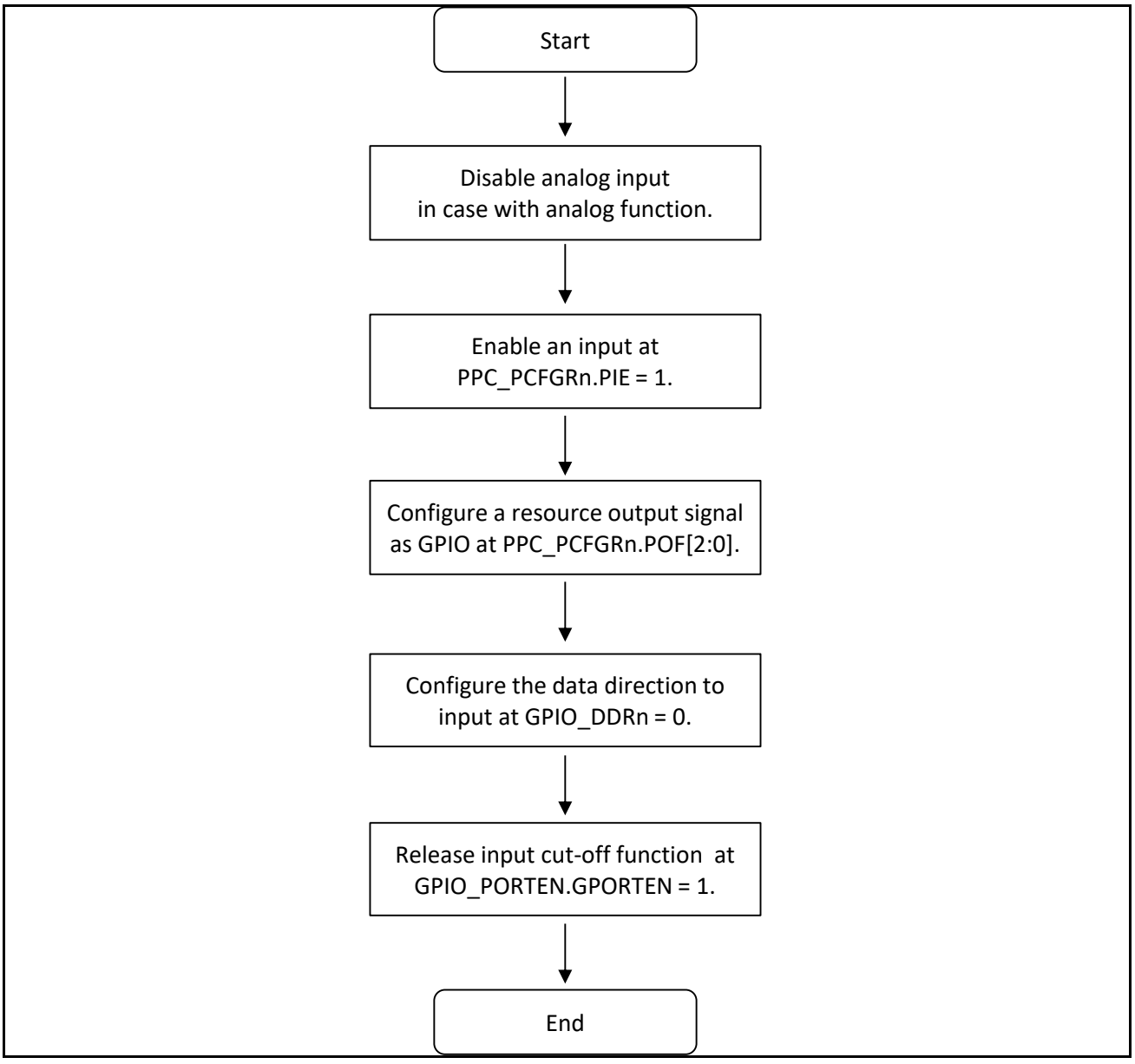


Notes:

- *The dedicated output control configuration as well as POF should be necessary for belows.*
  - *SOUT(MFS),*
  - *SGA,SGO(SG)*
  - *WOT(RTC)*
  - *MFS_CS (MFS)*

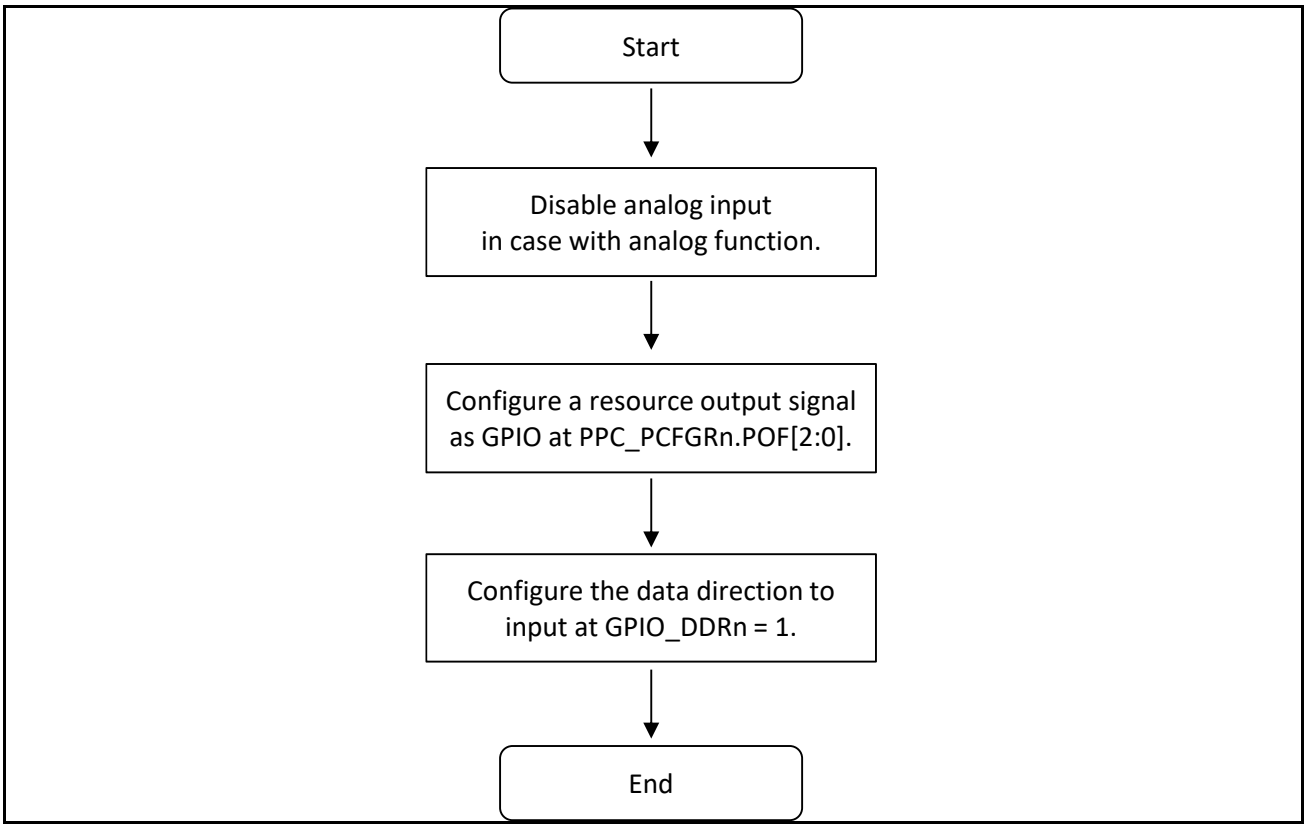
5.4. Port Function Input

Figure 11-5: Procedure of Port Input



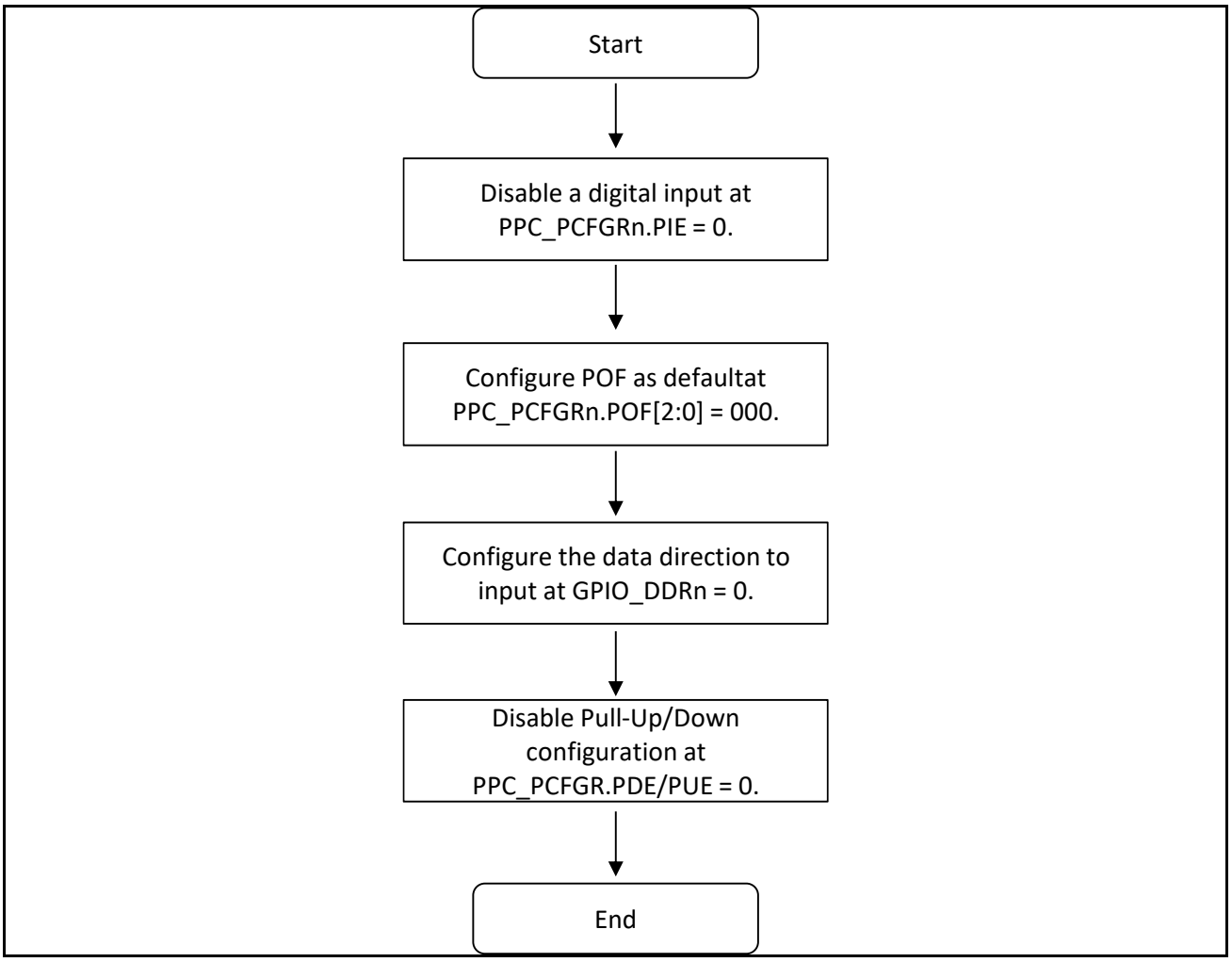
5.5. Port Function Output

Figure 11-6: Procedure of Port Output



5.6. Analog Function Input or Output

Figure 11-7: Procedure of Analog Function Input or Output



Notes:

- *The procedure is for analog input or output function of A/D converter and LCDC of which the ports have multiplexed usage.*
- *Regarding the analog switch setting, see 'CHAPTER of 12/10/8-BIT Analog to Digital Converter' and 'CHAPTER of LCDC Controller'.*
- *D/A Converter and FPD-Link has dedicated Output Ports. They don't have POF. The configuration is described in each chapter.*

6. Precautions

6.1. Noise Filter

Each GPIO has the noise filter for noise removal from external input. Also, some resources have additional dedicated noise filter for resource input. See the following Configuration and Diagram.

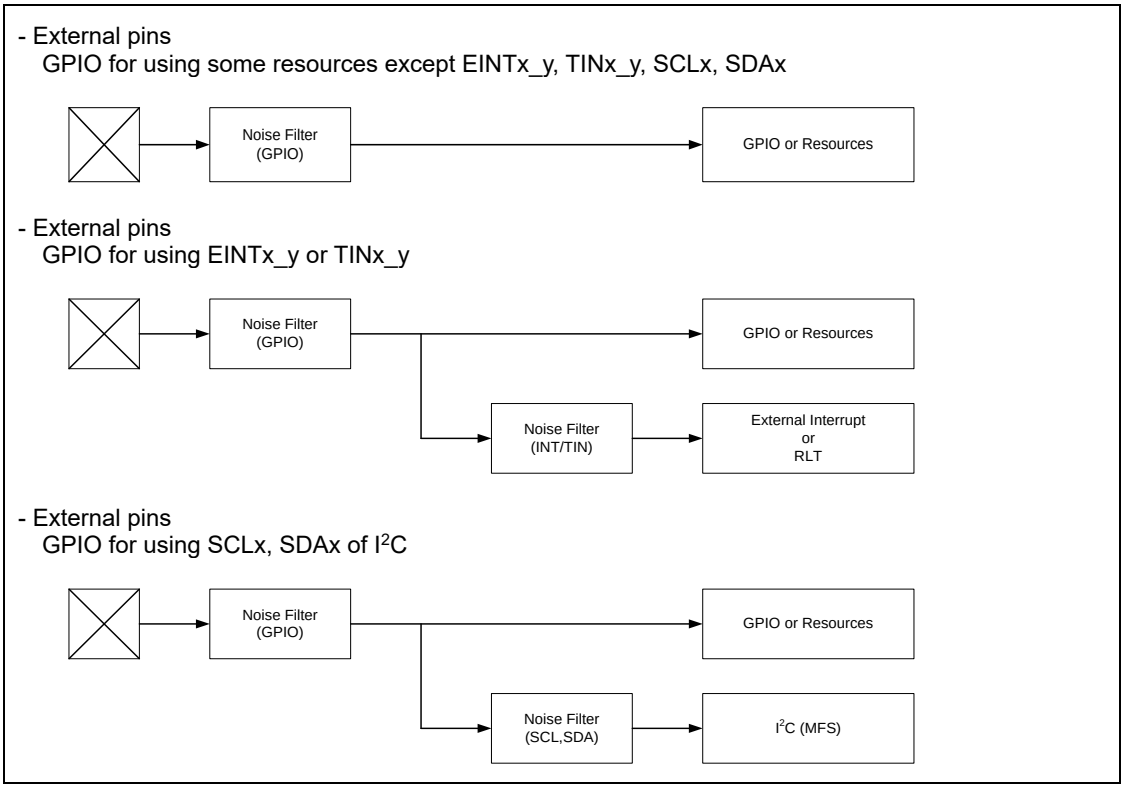
Table 11-3: Configuration for Noise Filter

Noise filter for resource	PPC_PCFGRiij: NFE	EICxx_NFER: NFEn	RLTn_TMCSR: NFE	RIC_RESINn: RESSEL
GPIO	Enable/Disable	- *1	- *1	- *1
External interrupt (EINT)	Disable *2	Enable/Disable	- *1	- *1
Reload Timer input (TIN)	Disable *2	- *1	Enable/Disable	- *1
I^2C interface (SCL, SDA)	Disable *2	- *1	- *1	Enable/Disable

*1: It is unrelated to register's setting.

*2: When selecting a Noise Filter other than for GPIO, set the Port Setting Register (PPC_PCFGRiij: NFE) to Disable.

Figure 11-8: Diagram of Noise Filter



6.2. I²C setting

For I2C fast-mode output pins“P1_09 (SCL16), P1_10 (SDA16), P1_15 (SCL17), P1_16 (SDA17)”to output Hi-Z after reset, the registers must be set in the following order.

(1) Set the MD (Operation Mode Setting Bits) of the SMR (Serial Mode Register) to I2C mode.
(2) Set the POF (Port Output Function Selection Bit) of the PPC_PCFGRijj (Port Setting Register) to I2C.

If not set in the above order, the MCU will output "H" level before outputting Hi-Z to each port.

CHAPTER 12: State Transition

This chapter explains the state transition.

1. Overview
2. Diagram of State Transition
3. Fetching the Operation Mode
4. Changes to PSS and RUN

CODE: STATE_TRANSITION-S6J3300-E1

1. Overview

This section gives a brief overview of State transition

Refer to the low-power chapter for the detailed information for performing a change state.

2. Diagram of State Transition

This section shows diagram of state transitions.

The device state transitions for this series are shown below.

Figure 12-1 Diagram of Device State Transitions

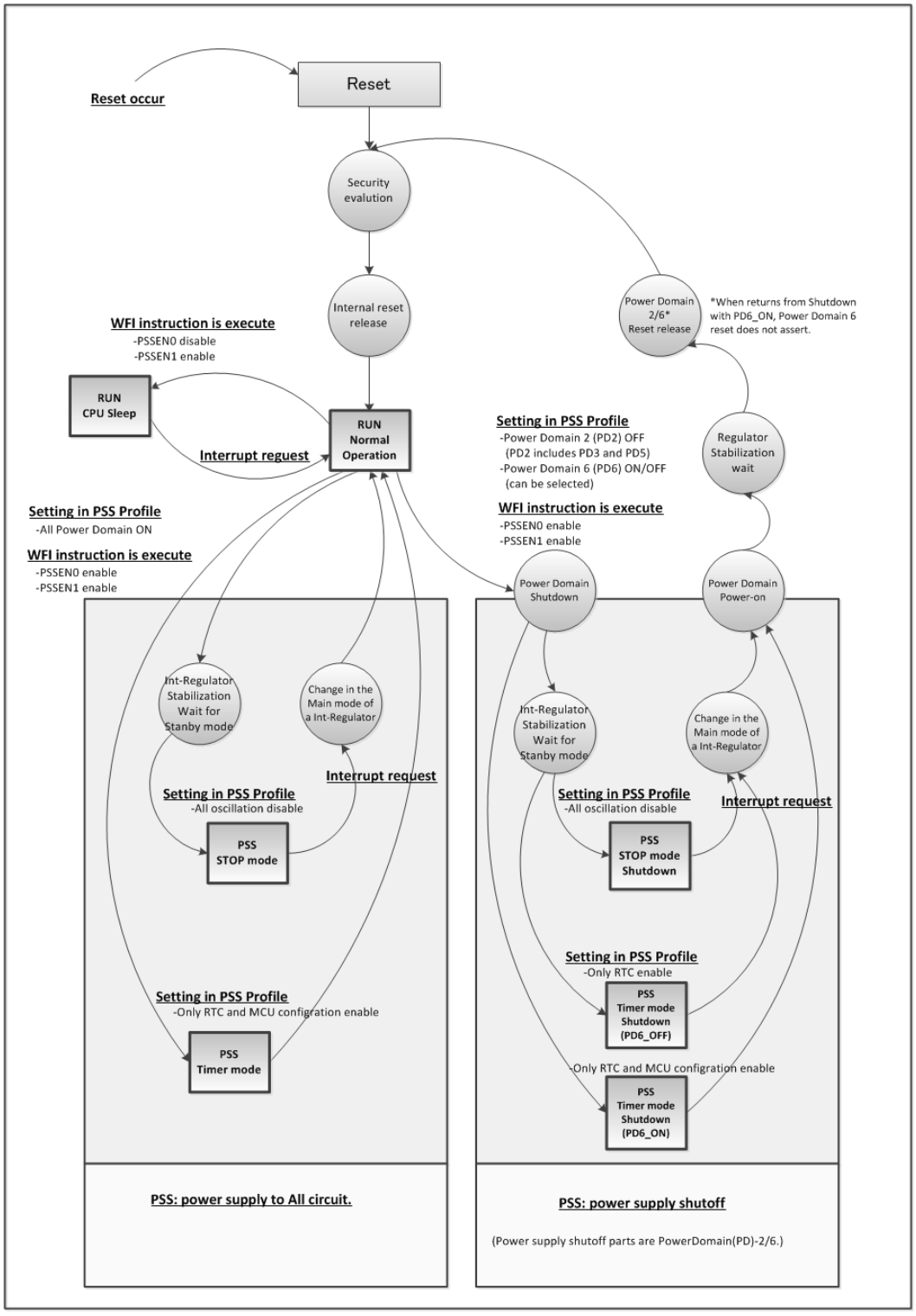


Figure 12-2 RUN/PSS State definitions

Internal state and a frequency definition

		RUN Normal Operation	RUN CPU Sleep	PSS Timer mode	PSS STOP mode	PSS Timer mode shutdown (PD6_ON)	PSS Timer mode shutdown (PD6_OFF)	PSS STOP mode Shutdown
Clocks	CLK_CPU	Customer any frequency	Customer any frequency	Prohibition of an oscillation	Prohibition of an oscillation	Prohibition of an oscillation	Prohibition of an oscillation	Prohibition of an oscillation
	CLK_FCLK							
	CLK_ATB							
	CLK_DBG							
	CLK_HPM							
	CLK_HPM2							
	CLK_DMA							
	CLK_MEMC							
	CLK_SYSC1							
	CLK_TRC							
	CLK_SYSC0H			Customer any frequency	Prohibition of an oscillation	Customer any frequency		
	CLK_COMH							
	CLK_RAM0H							
	CLK_RAM1H							
	CLK_SYSC0P							
	CLK_COMP							
	CLK_HAPP0A0			Prohibition of an oscillation		Prohibition of an oscillation		
	CLK_HAPP0A1							
	CLK_HAPP1B0							
	CLK_HAPP1B1							
	CLK_LLPBM							
	CLK_LLPBM2							
	CLK_LCP							
	CLK_LCP0							
	CLK_LCP0A							
	CLK_LCP1							
	CLK_LCP1A							
	CLK_LAPP0A0							
	CLK_LAPP0A1							
	CLK_LAPP1B0							
	CLK_LAPP1AB1							
oscillation state of a source clock	Slow-CR	100KHz	100KHz	100KHz / disable	Prohibition of an oscillation	100KHz / disable	100KHz / disable	Prohibition of an oscillation
	Fast-CR	4MHz	4MHz	4MHz	4MHz	4MHz	Prohibition of an oscillation	
	Main oscillator	4MHz / disable	4MHz / disable	4MHz / disable	Prohibition of an oscillation	4MHz / disable	4MHz / disable	
	Sub oscillator	32KHz / disable	32KHz / disable	32KHz / disable		32KHz / disable	32KHz / disable	
	PLL	Customer any frequency	Customer any frequency	Prohibition of an oscillation		Prohibition of an oscillation	Prohibition of an oscillation	
	SSCG_PLL	Customer any frequency	Customer any frequency					

The macro operating state in each mode, and the state of the power domain.

		RUN Normal Operation	RUN CPU Sleep	PSS Timer mode	PSS STOP mode	PSS Timer mode shutdown (PD6_ON)	PSS Timer mode shutdown (PD6_OFF)	PSS STOP mode Shutdown
CPU(PD2)	-	enable	disable	disable	disable	Power down	Power down	Power down
FLASH(PD2)	-	Operation	NOP	Deep Sleep	Deep Sleep	Power down	Power down	Power down
Backup RAM(PD4)	-	Operation	NOP	NOP	NOP	NOP	NOP	NOP
Int-Regulator	-	Main mode	Main mode	Main mode	Standby mode	Main mode	Standby mode	Standby mode
Power Domain	PD2(/PD3/PD5)	ON	ON	ON	ON	OFF	OFF	OFF
	PD4_0	ON	ON	ON	ON	ON or OFF	ON or OFF	ON or OFF
	PD4_1	ON	ON	ON	ON	ON or OFF	ON or OFF	ON or OFF
	PD6	ON	ON	ON	ON	ON	OFF	OFF

The conditions for changing in each state.

		RUN Normal Operation	RUN CPU Sleep	PSS Timer mode	PSS STOP mode	PSS Timer mode shutdown (PD6_ON)	PSS Timer mode shutdown (PD6_OFF)	PSS STOP mode Shutdown
Setup for changes	PSSEN0	-	disable	enable	enable	enable	enable	enable
	PSSEN1	-	enable	enable	enable	enable	enable	enable
Changes start command		-	Execution of a WFI command.					

3. Fetching the Operation Mode

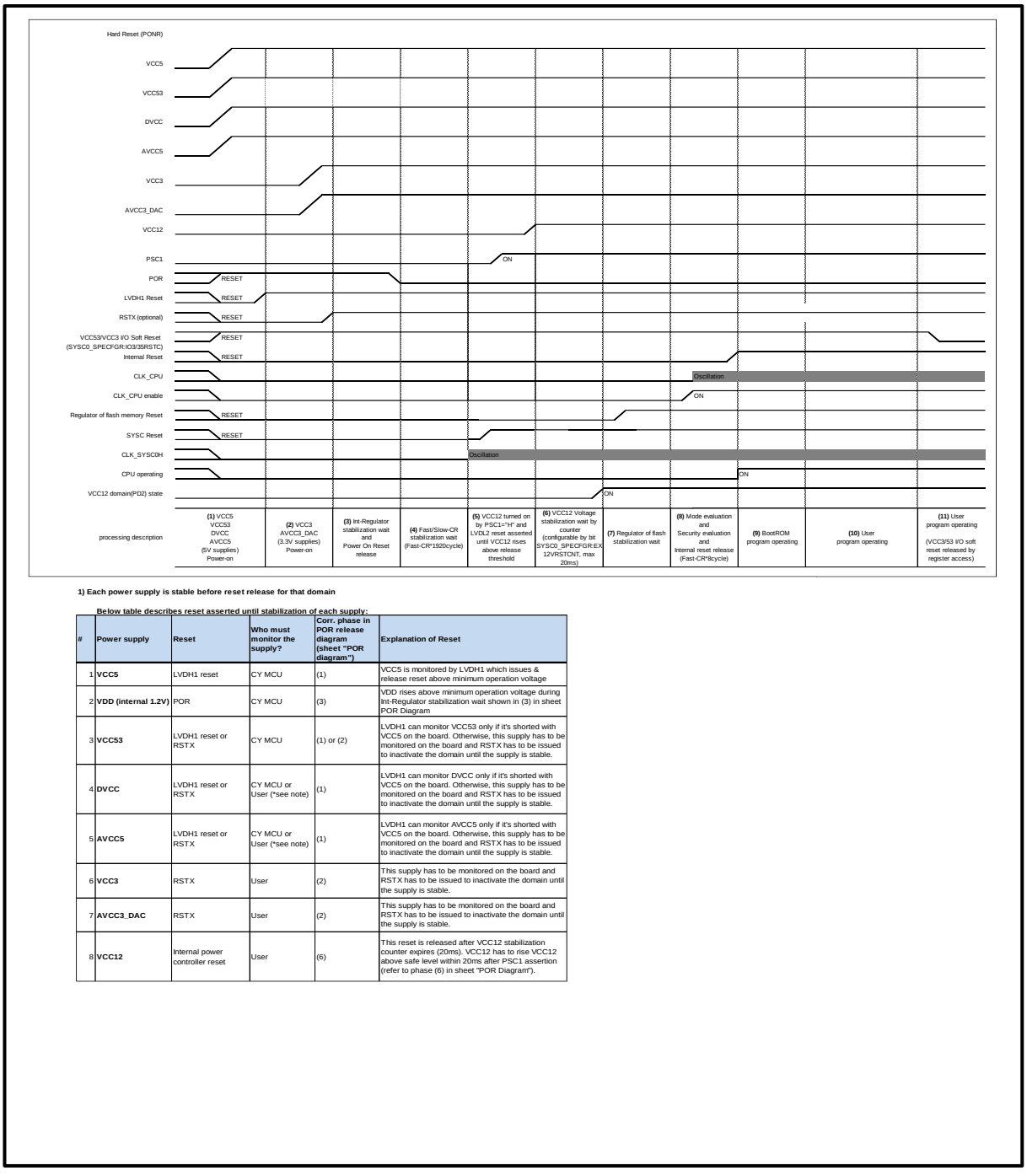
This section describes the Fetching the Operation Mode.

The operation mode is fetched by sampling the RST (Reset).

The following shows an operation sequence from an occurrence of reset cause to the determination of an operation mode.

The conditions for starting of a power supply and the correlation of a power supply are shown in Figure 12-3. Operation Mode Fetch Timing Chart of RESET other than PONR is shown in Figure 12-4.

Figure 12-3 Operation Mode Fetch Timing Chart of PONR



1) Each power supply is stable before reset release for that domain

Below table describes reset asserted until stabilization of each supply:

#	Power supply	Reset	Who must monitor the supply?	Corr. phase in POR release diagram (sheet "POR diagram")	Explanation of Reset
1	**VCC5**	LVDH1 reset	CY MCU	(1)	VCC5 is monitored by LVDH1 which issues & release reset above minimum operation voltage
2	**VDD (internal 1.2V)**	POR	CY MCU	(3)	VDD rises above minimum operation voltage during Int-Regulator stabilization wait shown in (3) in sheet POR Diagram
3	**VCC53**	LVDH1 reset or RSTX	CY MCU	(1) or (2)	LVDH1 can monitor VCC53 only if it's shorted with VCC5 on the board. Otherwise, this supply has to be monitored on the board and RSTX has to be issued to inactivate the domain until the supply is stable.
4	**DVCC**	LVDH1 reset or RSTX	CY MCU or User (*see note)	(1)	LVDH1 can monitor DVCC only if it's shorted with VCC5 on the board. Otherwise, this supply has to be monitored on the board and RSTX has to be issued to inactivate the domain until the supply is stable.
5	**AVCC5**	LVDH1 reset or RSTX	CY MCU or User (*see note)	(1)	LVDH1 can monitor AVCC5 only if it's shorted with VCC5 on the board. Otherwise, this supply has to be monitored on the board and RSTX has to be issued to inactivate the domain until the supply is stable.
6	**VCC3**	RSTX	User	(2)	This supply has to be monitored on the board and RSTX has to be issued to inactivate the domain until the supply is stable.
7	**AVCC3_DAC**	RSTX	User	(2)	This supply has to be monitored on the board and RSTX has to be issued to inactivate the domain until the supply is stable.
8	**VCC12**	Internal power controller reset	User	(6)	This reset is released after VCC12 stabilization counter expires (20ms). VCC12 has to rise VCC12 above safe level within 20ms after PSC1 assertion (refer to phase (6) in sheet "POR Diagram").

2) Isolation is valid before each pair of adjacent supplies is stable during power-up sequence

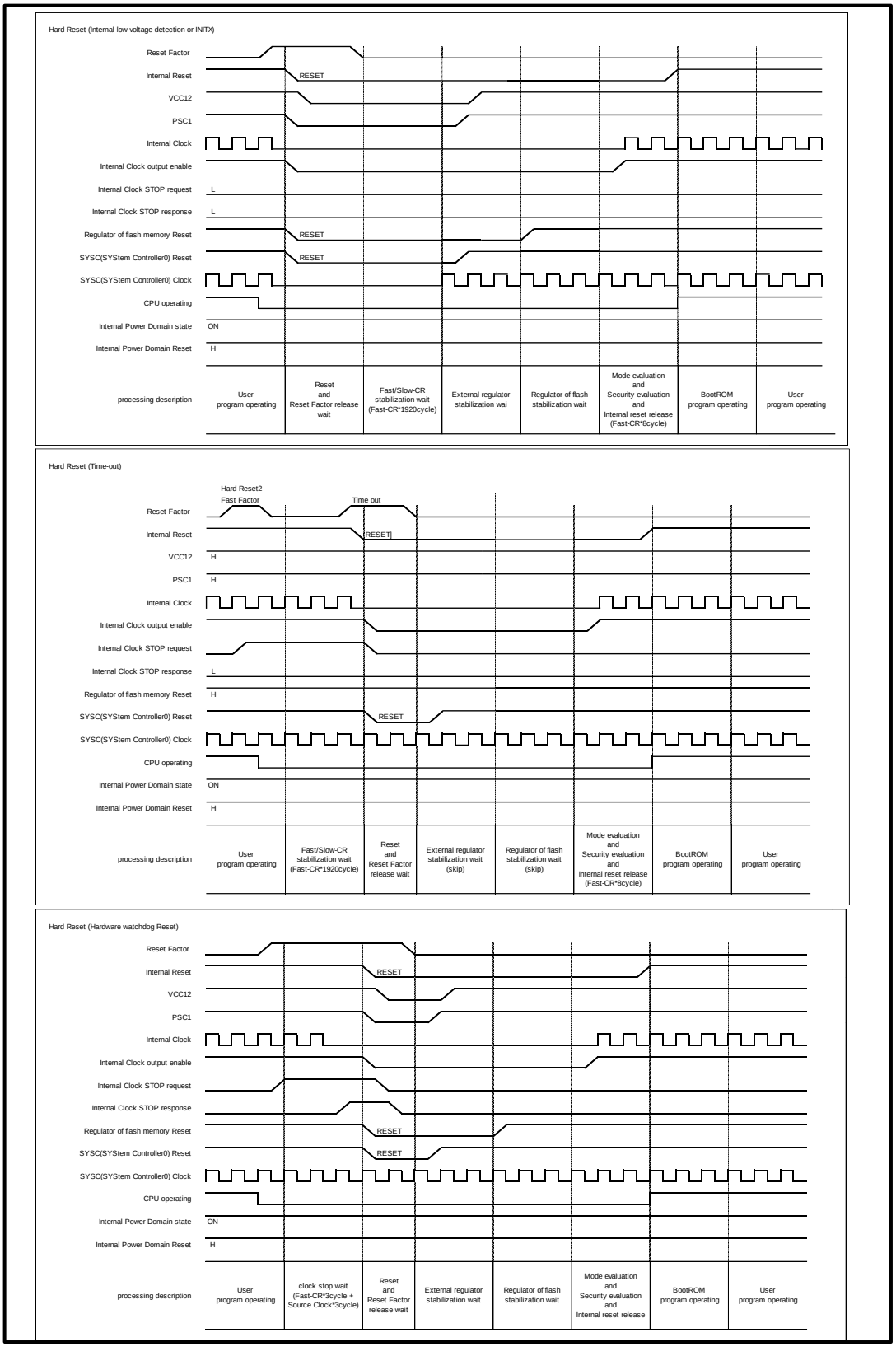
Below table describes isolation available for each adjacent supply pair:

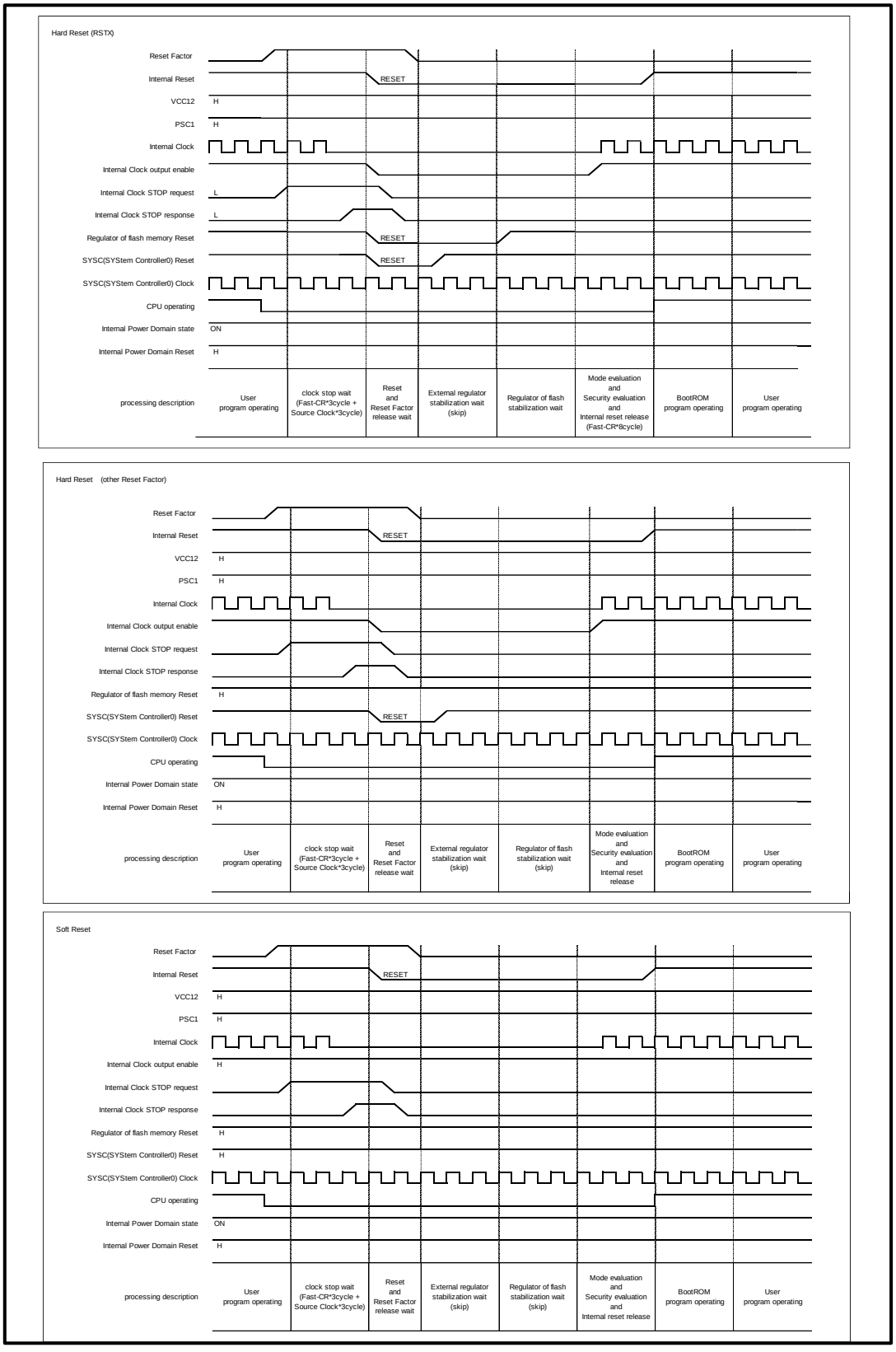
#	From	To	Isolation method	Isolator assert timing (sheet "POR diagram")	Isolator release timing (sheet "POR diagram")
1	**VCC5**	**VDD**	POR & LVDH1 initialize both domains until VCC5 stablization	(1) POR assertion	(2) LVDH1 release
2	**VCC53**	**VDD**	3.3/5V I/O reset (SYSC0_SPECFGR:IO35RSTC) works as isolation until VCC53 powers up and user inverts the bit in software	(1) POR assertion	(11) Reset release by software (SYSC0_SPECFGR:IO35RSTC=0)
3	**DVCC**	**VDD**	LVDH1 reset (if DVCC shorted w/ VCC5 on board) or RSTX, etc. (otherwise) works as isolation until DVCC powers up and user inverts the bit in software	(1) POR assertion or RSTX etc. assertion	(1) LVDH1 or RSTX etc. release
4	**AVCC5**	**VCC12**	LVDH1 reset (if AVCC5 shorted w/ VCC5 on board) or RSTX, etc. (otherwise) works as isolation until AVCC5 powers up and user inverts the bit in software	(1) POR assertion or RSTX etc. assertion	(1) LVDH1 or RSTX etc. release
5	**AVCC5**	**VDD**	LVDH1 reset (if AVCC5 shorted w/ VCC5 on board) or RSTX, etc. (otherwise) works as isolation until AVCC5 powers up and user inverts the bit in software	(1) POR assertion or RSTX etc. assertion	(1) LVDH1 or RSTX etc. release
6	**VCC3**	**VCC12**	3V I/O reset (SYSC0_SPECFGR:IO3RSTC) works as isolation until VCC3 powers up and user inverts the bit in software	(1) POR assertion	(11) Reset release by software (SYSC0_SPECFGR:IO3RSTC=0)
7	**AVCC3_DAC**	**VCC12**	3V I/O reset (SYSC0_SPECFGR:IO3RSTC) works as isolation until AVCC3_DAC powers up and user inverts the bit in software	(1) POR assertion	(11) Reset release by software (SYSC0_SPECFGR:IO3RSTC=0)
8	**VCC12**	**AVCC5**	VCC12 stabilization wait reset initializes both domains	(1) POR assertion	(6) VCC12 stabilization wait done
9	**VCC12**	**VCC3**	3V I/O reset (SYSC0_SPECFGR:IO3RSTC) works as isolation until VCC3 powers up and user inverts the bit in software	(1) POR assertion	(11) Reset release by software (SYSC0_SPECFGR:IO3RSTC=0)
10	**VCC12**	**AVCC3_DAC**	VCC12 stabilization wait reset initializes both domains	(1) POR assertion	(6) VCC12 stabilization wait done
11	**VCC12**	**VDD**	Power sequence controller in VDD domain asserts isolation until VCC12 stablization	(1) POR assertion	(6) VCC12 stabilization wait done
12	**VDD**	**VCC5**	POR initialized both domains until VDD stabilization	(1) POR assertion	(3) POR release
13	**VDD**	**VCC53**	POR initialized both domains until VDD stabilization	(1) POR assertion	(3) POR release
14	**VDD**	**DVCC**	POR initialized both domains until VDD stabilization	(1) POR assertion	(3) POR release
15	**VDD**	**AVCC5**	POR initialized both domains until VDD stabilization	(1) POR assertion	(3) POR release
16	**VDD**	**VCC12**	POR initialized both domains until VDD stabilization	(1) POR assertion	(3) POR release

Existing power supply pairs (# in parenthesis = corresp. entry in left table):

From \ To	VCC5	VDD	VCC53	DVCC	AVCC5	VCC3	AVCC3_DAC	VCC12	
VCC5		x (1)							VCC5 interacts only with regulated domain
VDD	x (12)		x (13)	x (14)	x (15)			x (16)	VDD interacts with Always-ON domain I/Os
VCC53		x (2)							VCC53 powers only I/Os in VDD domain
DVCC		x (3)							DVCC powers only I/Os & analog SWs in VDD domain
AVCC5		x (5)						x (4)	AVCC5 powers analog SWs & A/D in VCC12 domain
VCC3								x (6)	VCC3 powers only I/Os in VCC12 domain
AVCC3_DAC								x (7)	AVCC3_DAC powers only AudioDAC macro in VCC12 domain
VCC12		x (11)			x (8)	x (9)	x (10)		VCC12 domain includes I/Os & analog macro supplies that can be turned off. Also, there is a boundary b/w VDD (Always-ON) domain & VCC12 (switchable) domain

Figure 12-4 Operation Mode Fetch Timing Chart of other Reset

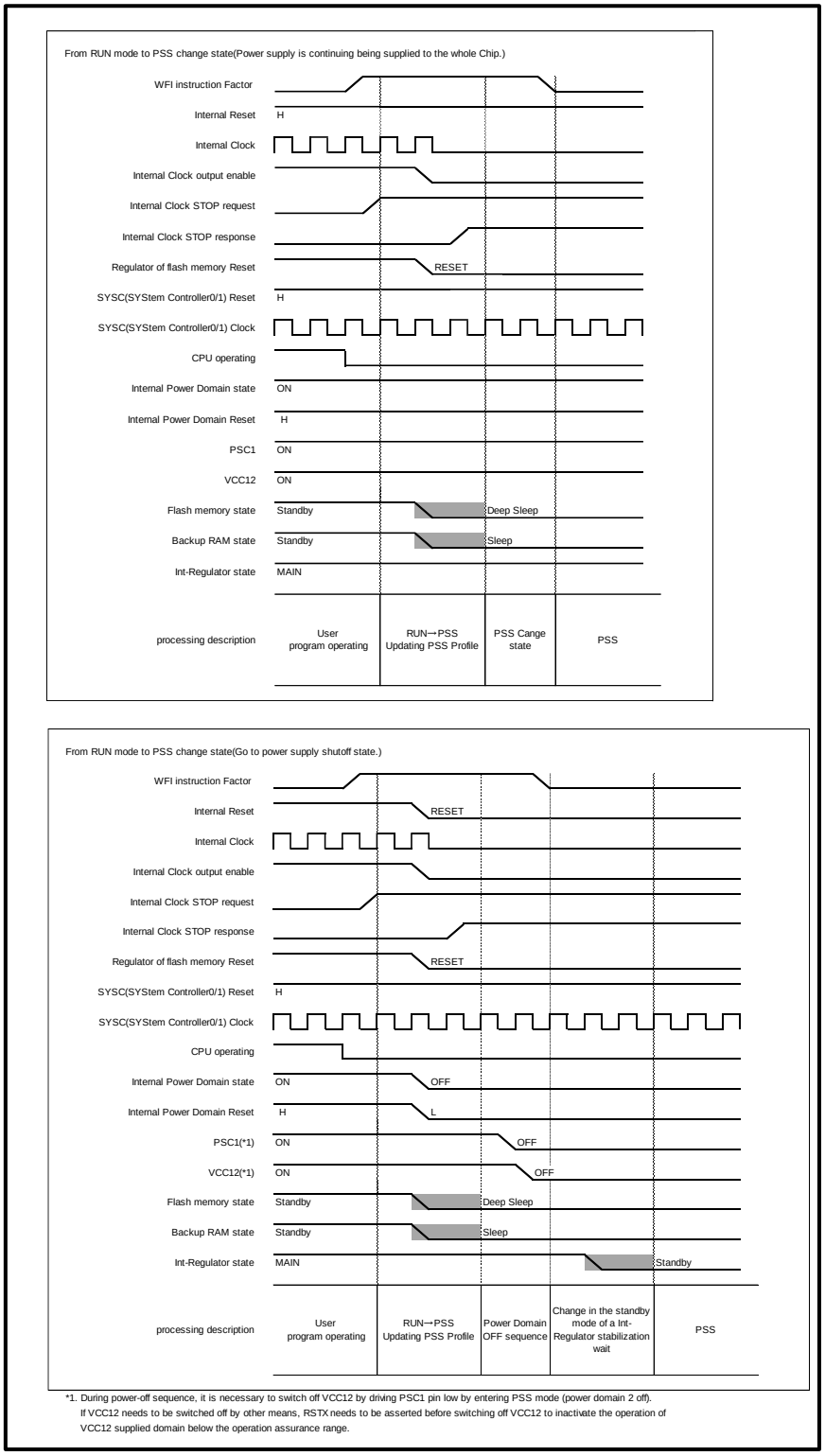



Hard Reset (RSTX)
Reset Factor
Internal Reset
VCC12
PSC1
Internal Clock
Internal Clock output enable
Internal Clock STOP request
Internal Clock STOP response
Regulator of flash memory Reset
SYSC(SYStem Controller0) Reset
SYSC(SYStem Controller0) Clock
CPU operating
Internal Power Domain state
Internal Power Domain Reset
processing description
RESET
H
L
ON
User program operating
clock stop wait (Fast-CR*3cycle + Source Clock*3cycle)
Reset and Reset Factor release wait
External regulator stabilization wait (skip)
Regulator of flash stabilization wait
Mode evaluation and Security evaluation and Internal reset release (Fast-CR*8cycle)
BootROM program operating
Hard Reset (other Reset Factor)
Mode evaluation and Security evaluation and Internal reset release
Soft Reset

4. Changes to PSS and RUN

This section shows the sequence which changes to PSS and RUN.

Figure 12-5 Changes to PSS Timing Chart



*1. During power-off sequence, it is necessary to switch off VCC12 by driving PSC1 pin low by entering PSS mode (power domain 2 off).
If VCC12 needs to be switched off by other means, RSTX needs to be asserted before switching off VCC12 to inactivate the operation of VCC12 supplied domain below the operation assurance range.

Figure 12-6 Changes to PSS Timing Chart

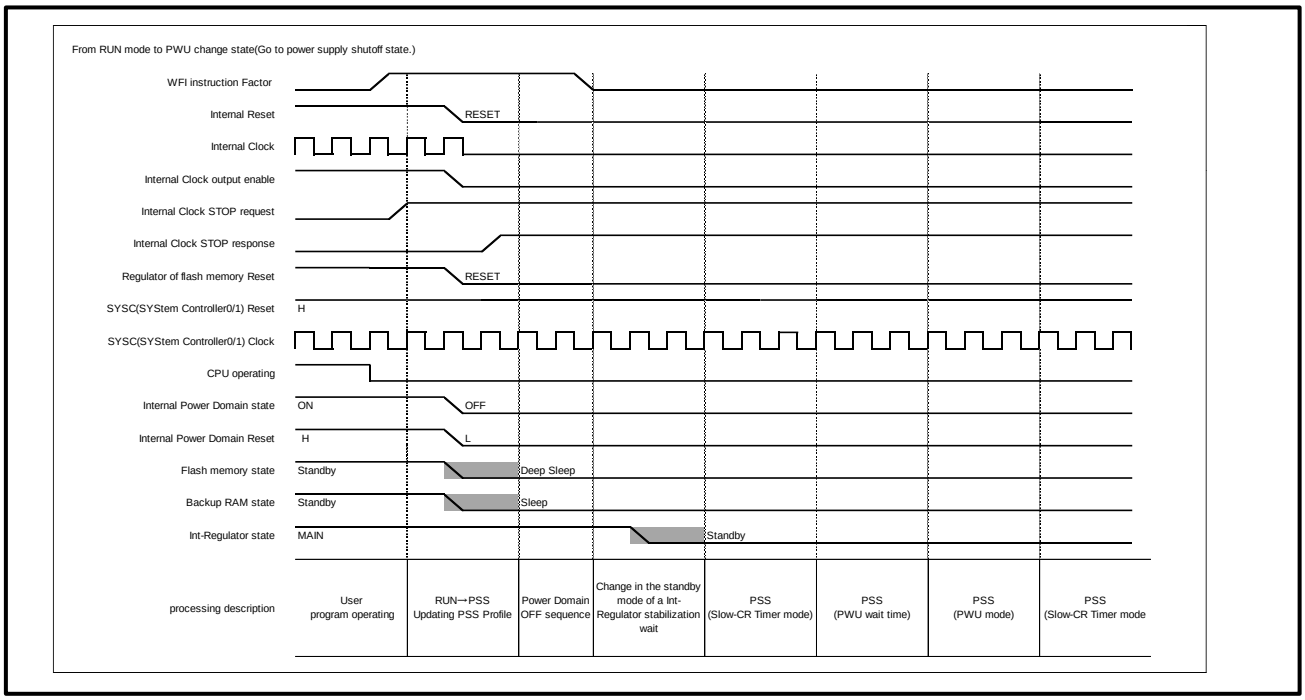
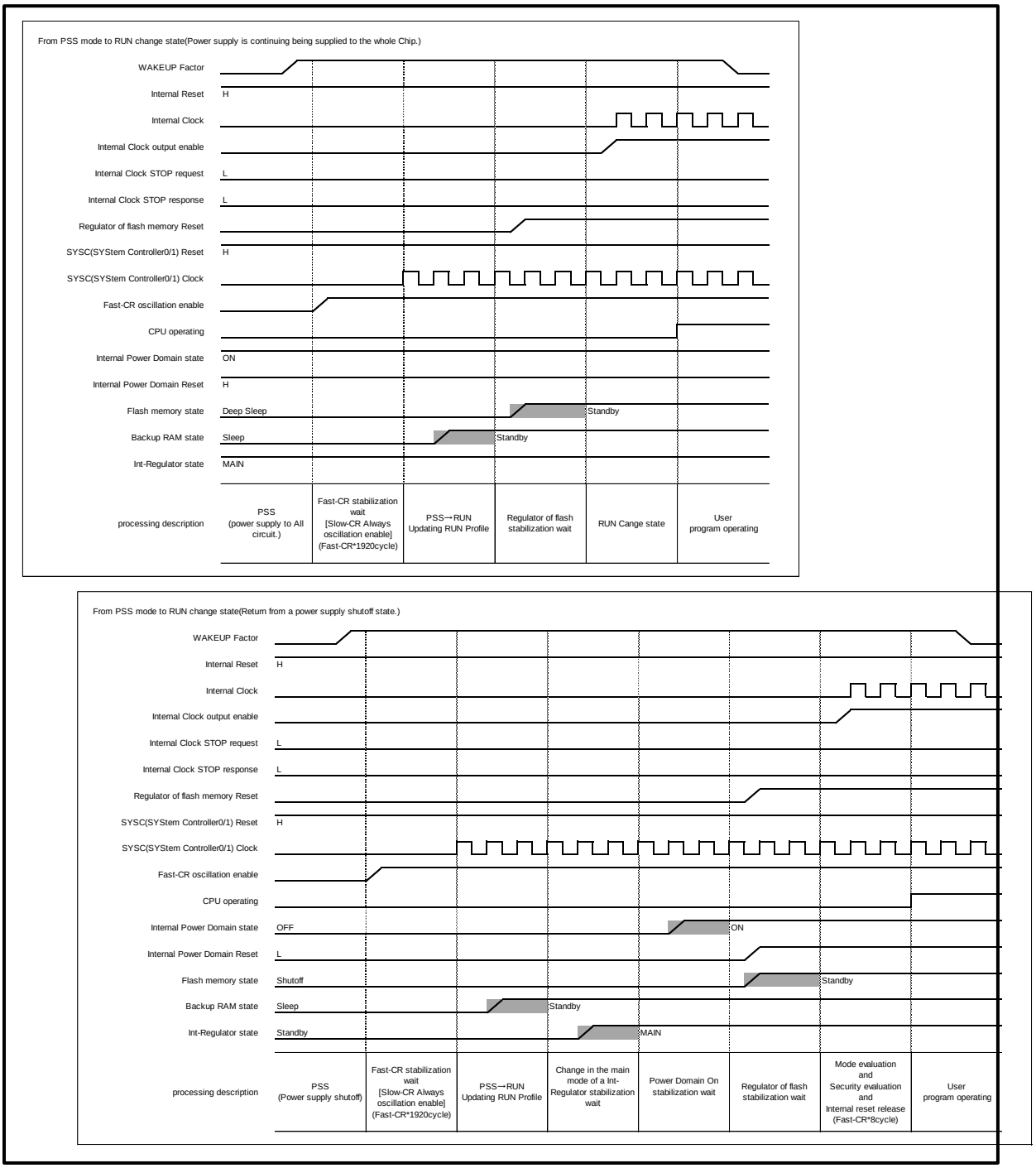


Figure 12-7 Changes to RUN Timing Chart



CHAPTER 13: Low Voltage Detection

This chapter explains the function of low voltage detection.

1. Overview
2. Configuration of Supervised Power Domain
3. Operation
4. Registers

CODE: LVD-S6J3300-E1

1. Overview

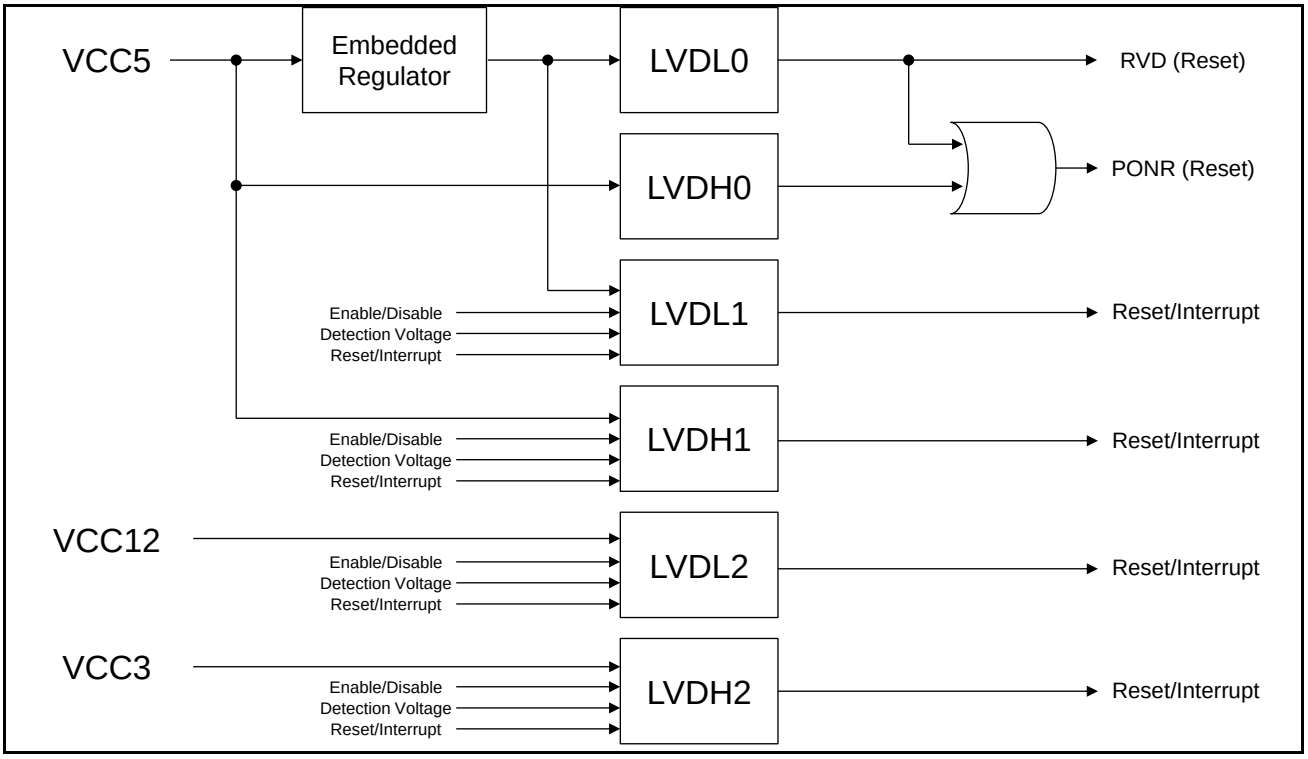
Low-Voltage Detection (LVD) gives the following functions.

- Power on reset (PONR) generation.
- RAM retention low-voltage detection reset (RVD) generation.
- Internal regulator output supervision.
- External power supply supervision.

2. Configuration of Supervised Power Domain

The block diagram shows LVDs hardware and software configuration.

Figure 13-1 LVD Hardware and Software Configuration



3. Operation

3.1. LVD operation

Table 13-1: LVD Description

Channel	Monitor	Detected Voltage	Operation after detection	Enable/Disable	Description (Purpose)
LVDL0	Embedded regulator output	Specific	Reset only	Always enable	RAM retention voltage supervision. Power on condition supervision.
LVDH0	VCC5	Specific	Reset only	Always enable	Power on condition supervision.
LVDL1	Embedded regulator output	Configurable	Reset or interrupt	Configurable	PD1 power supply supervision.
LVDH1	VCC5	Configurable	Reset or interrupt	Configurable	Internal regulator input and 5 V I/O power supply supervision.
LVDL2	VCC12	Configurable	Reset or interrupt	Configurable	PD2 power supply supervision.
LVDH2	VCC3	Configurable	Reset or interrupt	Configurable	3 V I/O power supply supervision.

Notes:

- *PONR will be*
  - *generated by means of either LVDL0 or LVDH0 detected.*
  - *released by means of both LVDL0 and LVDH0 released.*
- *The voltage for both to detect and to release is specified in the datasheet.*
- *The detection/release threshold values of following LVD channels are potentially below supply range defined in Recommended operating condition of Datasheet.*
  LVDL0
  LVDL1
  LVDL2
  LVDH0
  LVDH1
  LVDH2
- *Please use these LVD channels with your own risk*
- *Please monitor the external power supplies on the PCB if needed*

3.2. Configurations

Function enable/disable, detected voltage threshold, and operation option after low-voltage detection are selectable and configurable for LVDL1, LVDH1, LVDL2, and LVDH2 by software.

As for LVDL0 and LVDH0 anything cannot be configured by software. The functions are always enabled, detect specific voltage, and only generate reset.

The voltage range of detect and release for each is specified in datasheet.

Bit name	Description
LVDL1S, LVDH1S, LVDL2S, LVDH2S	0: Reset, 1: Interrupt
LVDL1V, LVDH1V, LVDL2V, LVDH2V	Threshold voltage can be configured to the bit range.
LVDL1E, LVDH1E, LVDL2E, LVDH2E	0: Disable, 1: Enable

4. Registers

See the RUN/PSS/APP/STS Low-voltage Detection setting Register on Traveo™ Platform Hardware Manual.

Here, configurable value and its operation are described.

[Bit30] LVDL1S Internal low-voltage detection voltage operation selection bit

Bit 30	Description
0	Reset (Initial value)
1	Interrupt

[Bit26:25] LVDL1V Internal low-voltage detection voltage setting bits

Bit 26:25	Voltage [V]	Guaranteed MCU operation voltage range
00 *	0.875(Initial value)	No
01 *	0.95	
10 *	1.00	
11 *	1.05	

*: This LVD cannot be used to reliably generate a reset before voltage dips below minimum guaranteed MCU operation voltage, as these detection levels are below the minimum guaranteed MCU operation voltage.

[Bit24] LVDL1E Internal low-voltage detection operation enable bit

Bit 24	Description
0	STOP operation
1	Enable operation (Initial value)

Note:

- *LVDL1 corresponds to 1.2 V power supply low-voltage detection ch.0*

[Bit22] LVDL2S Internal low-voltage detection voltage operation selection bit

Bit 22	Description
0	Reset (Initial value)
1	Interrupt

[Bit18:17] LVDL2V Expanded internal low-voltage detection voltage setting bits

Bit 18:17	Voltage [V]	Guaranteed MCU operation voltage range
00 *	0.875	No
01 *	0.95	
10 *	0.925	
11 *	0.8125(Initial value)	

*: This LVD cannot be used to reliably generate a reset before voltage dips below minimum guaranteed MCU operation voltage, as these detection levels are below the minimum guaranteed MCU operation voltage.

[Bit16] LVDL2E Internal low-voltage detection operation enable bit

Bit 16	Description
0	STOP operation
1	Enable operation (Initial value)

[Bit14] LVDH1S Internal low-voltage detection voltage operation selection bit

Bit 14	Description
0	Reset (Initial value)
1	Interrupt

Note:

- *LVDH1 Reset level can be changed by EXVRSTCNT bit in SYSC0_SPECFGR register. If VCC12 needs to be switched off without entering PSS mode, please set LVDH1S bit to "0", and moreover set EXVRSTCNT bit to "0".*

[Bit11:9] LVDH1V External low-voltage detection voltage setting bits

Bit 11:9	Voltage [V]	Guaranteed MCU operation voltage range	
		Chip ID 0x10122xxx product	Chip ID 0x10128xxx, 0x10120xxx product
000 *	2.7	No	No
001	2.8	Yes	Yes
010	3.6		Not supported
011	3.8		
100	4.0(initial value for Chip ID : 0x10122xxx product)		
101	4.2		
110 *	2.5	No	No
111 *	2.6(initial value for Chip ID: 0x10128xxx product and 0x10120xxx product)		

*: This LVD cannot be used to reliably generate a reset before voltage dips below minimum guaranteed MCU operation voltage, as these detection levels are below the minimum guaranteed MCU operation voltage.

Note:

- *Regarding the Chip ID, see FUNCTION LIST on the datasheet.*

[Bit8] LVDH1E Internal low-voltage detection operation enable bit

Bit 8	Description
0	STOP operation
1	Enable operation (Initial value)

[Bit6] LVDH2S Internal low-voltage detection voltage operation selection bit

Bit 6	Description
0	Reset
1	Interrupt(Initial value)

[Bit3:1] LVDH2V Expanded external low-voltage detection voltage setting bits

Bit 3:1	Voltage [V]	Guaranteed MCU operation voltage range
000 *	2.7	No
001	2.8	Yes
010	3.6	Not supported
011	3.8	
100	4.0	
101	4.2	
110 *	2.5	No
111 *	2.6(Initial value)	

*: This LVD cannot be used to reliably generate a reset before voltage dips below minimum guaranteed MCU operation voltage, as these detection levels are below the minimum guaranteed MCU operation voltage.

[Bit0] LVDH2E Internal low-voltage detection operation enable bit

Bit 0	Description
0	STOP operation (Initial value)
1	Enable operation

CHAPTER 14: Serial Programming

This chapter explains serial programming.

1. Overview
2. Memory Map
3. Flash Sector Configuration
4. Port Configuration
5. Operation
6. Note.

CODE: SERIAL_PRG-S6J3300-E1

1. Overview

This chapter describes the FLASH serial programming. Please refer Traveo™ Platform Hardware Manual if necessary.

2. Memory Map

See the chapter of Memory AND BASE ADDRESS MAP on this Hardware Manual.

3. Flash Sector Configuration

See the chapter of TCFLASH and WORKFLASH of Traveo™ Platform Hardware Manual.

4. Port Configuration

Table 14-1: Port Configuration and Usage

Port Name	Configuration	Remark
MODE	Pull-down	-
RSTX	Reset input	-
X0	Oscillation input	-
X1	Oscillation output	-
P1_03/SIN0	Pull-up: Clock asynchronous mode Pull-down: Clock synchronous mode Used for a serial input function after reset operation.	-
P1_05/SOT0	Pull-up: necessary until reset release. Used for a serial output function after reset operation.	-
P1_04/SCK0	Used for synchronous mode.	-

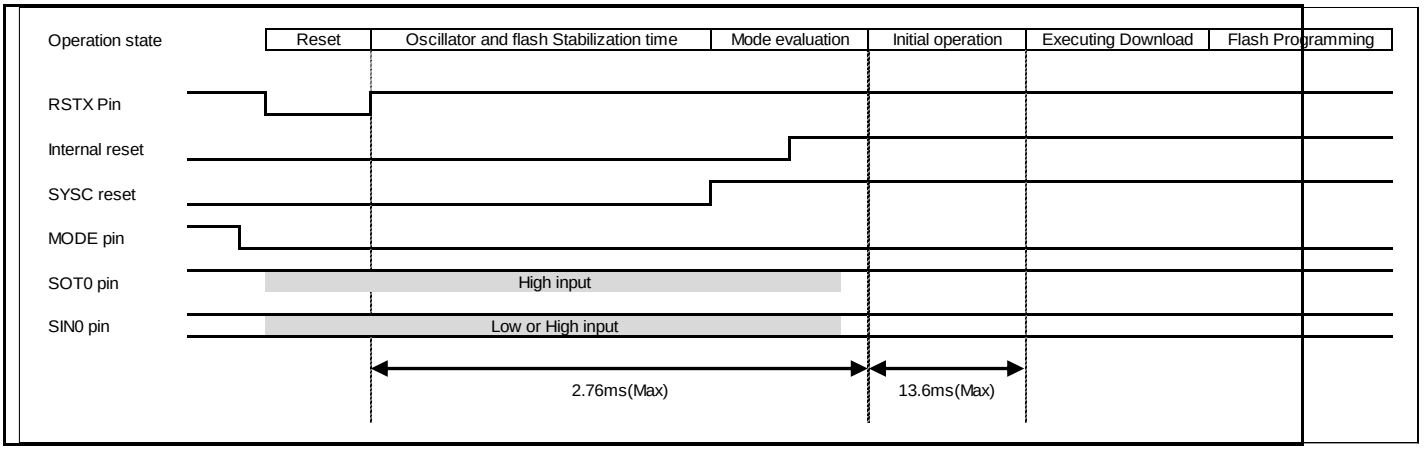
Note:

- *See the chapter of PORT DESCRIPTION for Port name.*
- *See the PIN ASSIGNMENT on Datasheet.*

5. Operation

5.1. Timing Chart

Figure 14-1 Timing Chart of State Transition for Serial Programming.



5.2. RAM Executing Communication Protocol

5.2.1. Command Format

5.2.1.1. Download Command

Byte Position	Byte Value	Explanation
1	00H	Download command
2	XXH	Download start address (A7-A0)
3	XXH	Download start address (A15-A8)
4	XXH	Download start address (A23-A16)
5	XXH	Download start address (A31-A24)
6	XXH	Download count number (BC7-BC0)
7	XXH	Download count number (BC15-BC8)
8	XXH	Download count number (BC23-BC16)
9	XXH	Download count number (BC31-BC24)
10	XXH	Sum value of download command(*1)

(*1) The SUM value is calculated in 8-bits, and the overflow of simple addition is ignored.

5.2.1.2. Executing Command

Byte Position	Byte Value	Explanation
1	C0H	Executing command
2	00H	Dummy data (For adjusting of command length)
3	00H	Dummy data (For adjusting of command length)
4	00H	Dummy data (For adjusting of command length)
5	00H	Dummy data (For adjusting of command length)
6	00H	Dummy data (For adjusting of command length)
7	00H	Dummy data (For adjusting of command length)
8	00H	Dummy data (For adjusting of command length)
9	00H	Dummy data (For adjusting of command length)
10	C0H	SUM value of executing command

5.2.1.3. Reset Command

Byte Position	Byte Value	Explanation
1	18H	Reset command

5.2.1.4. Full Chip Erase Command

Word Position	Word Value	Explanation
1	C0356EA5H	Key0
2	2E830596H	Key1
3	01A00000H	Key2
4	F3A033EDH	Key3
5	370E6A51H	Key4
6	B0412000H	Key5
7	00000002H	Key6
8	AA805510H	Key7

5.2.2. Command Sequence

5.2.2.1. Download Command Sequence

Byte Count	Host(Tool) ⇒ Micom	Micom ⇒ Host(Tool)
1 Byte	Receipt of down load command (00H)	
4 Byte	Receipt of download start address	
4 Byte	Receipt of download byte number	
1 Byte	SUM value of command	

Byte Count	Host(Tool) ⇒ Micom	Micom ⇒ Host(Tool)	
1 Byte		Command response Normal end SUM malfunction Command malfunction	 : 01H : X2H : X4H
N Byte	Receipt of download data	Data input procedure(Little Endian) D7 to D0 D15 to D8 D23 to D16 D31 to D24	
1 Byte	SUM value of download data (*1)		
1 Byte		Download processing response	
		Busy Normal end SUM malfunction	: 00H : 01H : 02H

(*1) About downloaded N byte data, SUM value is calculated in 8-bits, and the overflow of simple addition is ignored.

5.2.2.2. Executing Command Sequence

Byte Count	Host(Tool) ⇒ Micom	Micom ⇒ Host(Tool)	
1 Byte	Receipt of executing command (C0H)		
8 Byte	Receipt of dummy data (00H)		
1 Byte	SUM value of the command (C0H)		
(1 Byte)		Command response (when SUM value is malfunction) (*1)	
		SUM malfunction	: C2H

(*1) Only when the SUM malfunction occurs, command response is returned. Command response is not returned except the SUM malfunction occurs. After receiving executing command, it jumps to the download start address specified by the download command if the SUM is not malfunctioning.

5.2.2.3. Reset Command Sequence

Byte Count	Host(Tool) ⇒ Micom	Micom ⇒ Host(Tool)	
1 Byte	Download Command (18H)		
1 Byte		Command response (*1) Normal end	: 11H

(*1) Response of reset command is always normal end. (11H)

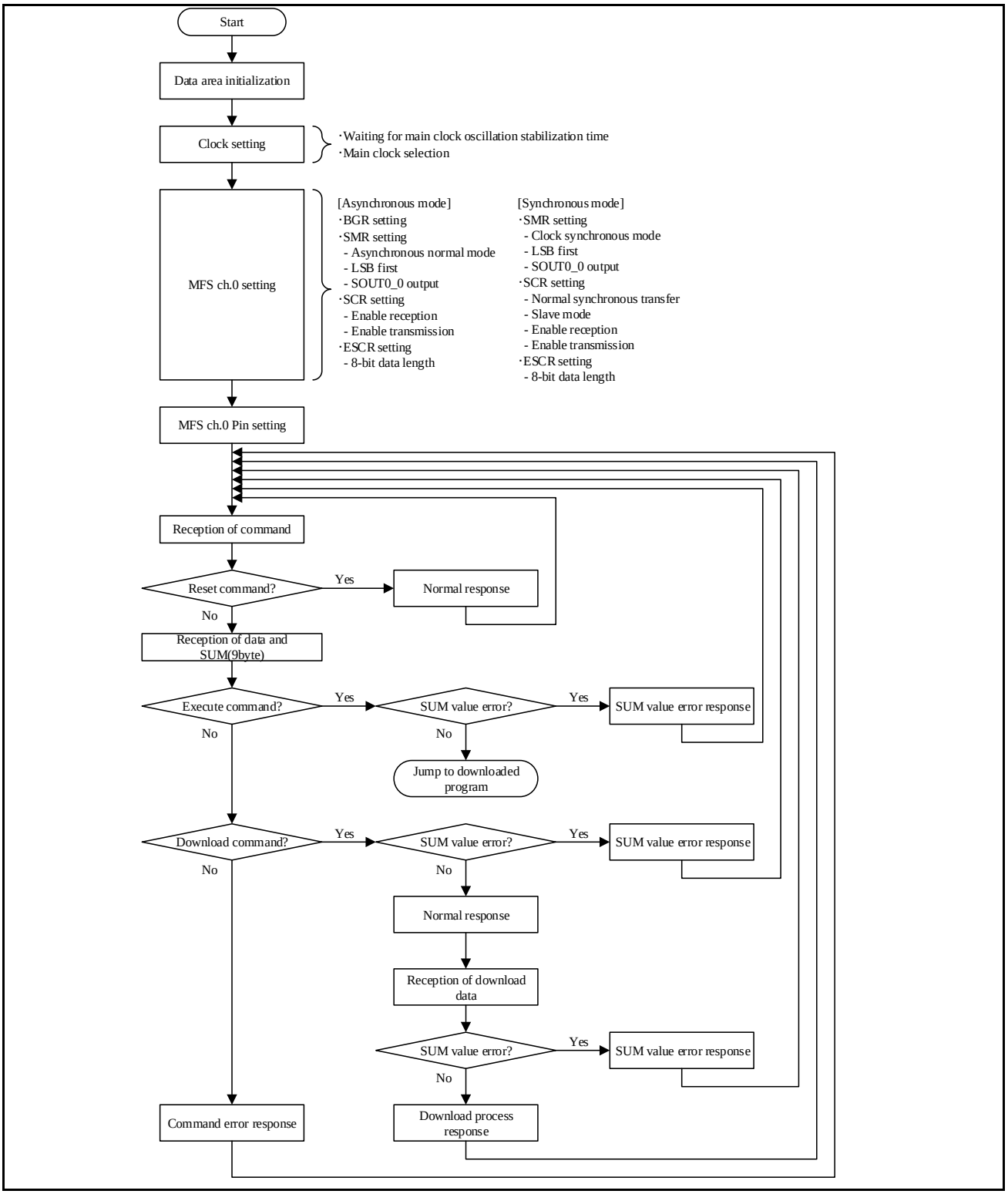
5.2.2.4. Full Chip Erase Command Sequence

Byte Count	Host(Tool) ⇒ Micom	Micom ⇒ Host(Tool)	
1-32 Byte	Key0-7Receive	-	
4 Byte	-	Command response (*1) Normal end	: 12345678H
4 Byte	-	Command response (*1) Normal end	: BABEFACEH

(*1) Command response is returned only when it is normal end. In other cases, hard reset is operated and command response is not returned.

5.3. Operation Flow

Figure 14-2: Operation Flow Chart



6. Note.

See Traveo™ Platform Hardware Manual.

6.1. PSC Port State

PSC port which is expected to control switching of external power supply devises will output "low" during external reset ("low" is inputted to RSTX) in SERIAL PROGRAMMING MODE. After reset ("high" is inputted to RSTX), PSC port will output "high".

If 1.2 V power supply is supervised and RSTX is controlled with its low-voltage state, PSC cannot output "high", that is, external power supply devises would not start power supplying, and MCU cannot go to a power-on sequence before a serial programming operation.

CHAPTER 15: 12-/10-/8-Bit Analog to Digital Converter

This chapter explains the functions and operations of the 12/10/8-bit A/D Converter.

1. Overview
2. Configuration and Block Diagram
3. Operation of A/D Converter
4. Setup Procedure Examples
5. Registers

CODE: FIP022-E02.2

1. Overview

The A/D Converter converts analog input voltages into digital values. The A/D Converter features eight range comparators, 64 pulse detection units, 64 separate conversion data registers and four multiple conversion channels.

Features of the A/D Converter

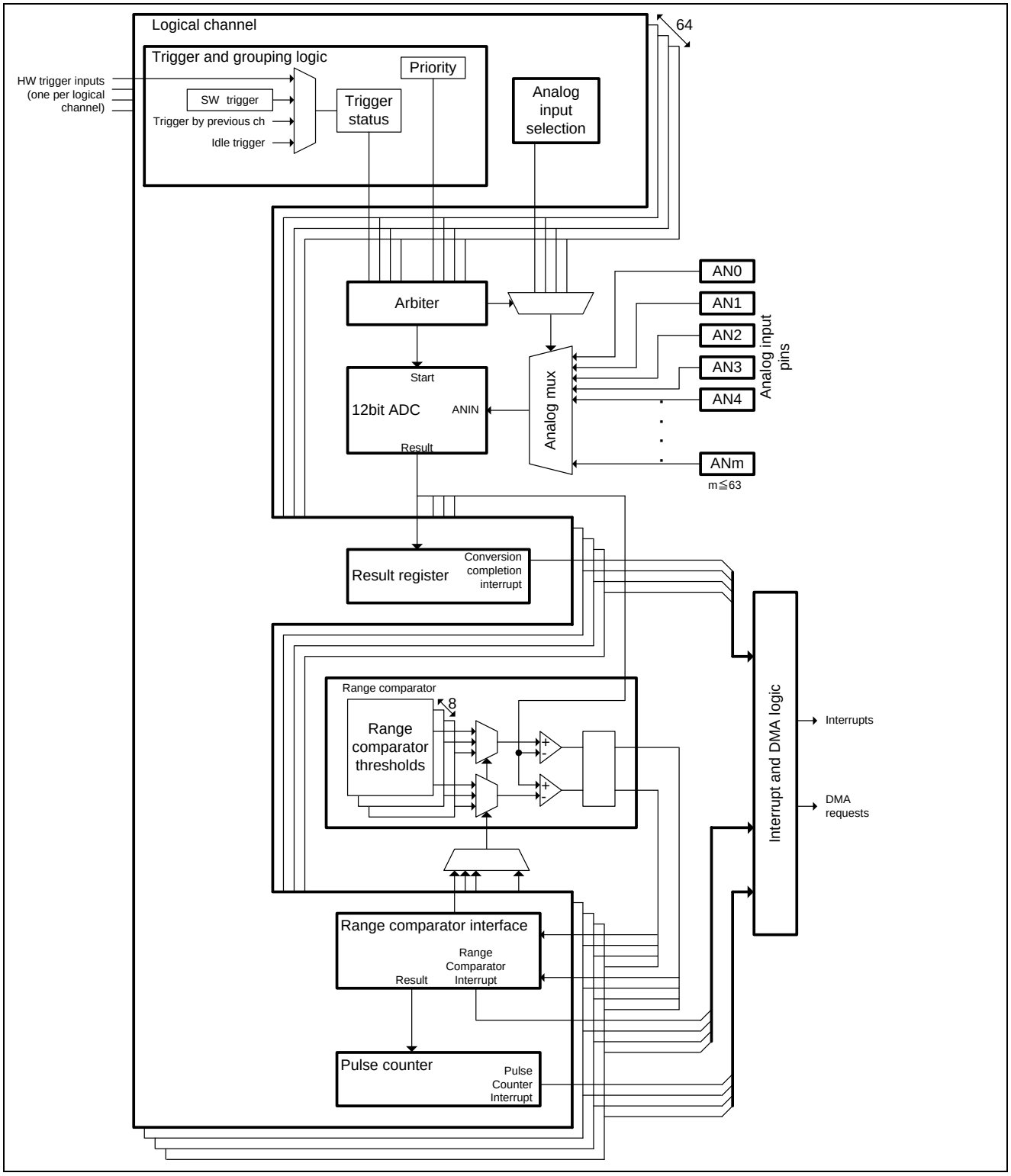
- Conversion time: Refer to the datasheet
- 12/10/8-bit resolution
- RC type successive approximation conversion with sample & hold circuit
- Programmable configuration of 64 logical channels by mapping them to up to 64 analog inputs
  - Each logical channel is mapped to exactly one analog input
  - Several logical channels can be mapped to the same analog input
- 64 dedicated logical channel conversion data registers
- Programmable selection of one of the four global sampling times for each logical channel
- Conversion request for each logical channel i is possible by software trigger only, hardware or software trigger, trigger by conversion completion of the preceding logical channel i-1 and idle trigger
- All active conversion requests are compared according to the corresponding logical channel priorities (selectable range is 0-15, 0 is the highest and 15 is the lowest priority) and A/D conversion is started for the logical channel with the highest priority
- When the higher priority conversion requests during an active A/D conversion, it is possible to select the following behavior
  - An active A/D conversion can be interrupted, and the higher priority conversion starts after interrupt operation
  - An active A/D conversion cannot be interrupted, hence the wait time between the higher priority conversion request and its conversion start can be up to the maximum A/D conversion period. In case of multiple conversion channel processing that are configured to be non-interruptible, the wait time can increase to the time needed to the maximum multiple channel conversion period
- The consecutive logical channels can be configured as a group, where the start channel conversion request is set to software trigger only, hardware or software trigger or idle trigger while all other channel conversion requests are set to the preceding logical channel conversion completion
- If a group conversion is interrupted by a higher priority conversion request, after the higher priority request is processed the following group configurations are possible:
  - Resume with the next logical channel in the group (before the conversion start of the channel the group is interrupted)
  - Restart with the group start channel or the last converted channel configured as "resume" channel within a group
  - Stop the group processing until next start channel conversion request is issued
- First four logical channels can be configured as:
  - Multiple conversion channels offering the possibility to perform 1 to 16 conversions of the same logical channel and accumulate the result
  - A/D Converter calibration channels providing the configuration for conversion of A/D Converter reference voltages in order to calculate the offset and gain corrections
- Eight range comparators are selectable for every logical channel, comparing the full range (12-bit)/upper 8-bit of the conversion result
- Programmable upper and lower thresholds for each range comparator
- The comparison results will set flags per logical channel, depending on the configuration. Possible configurations are:
  - “Outside range”: The flags are set if the A/D result is below the lower OR above the upper threshold
  - “Inside range”: The flags are set if the A/D result is above the lower AND below the upper threshold
- The results of the range comparator can be filtered to ignore short spikes
- During an active A/D conversion, it can be forced stop by software
- It is possible to select the operation after A/D conversion finished (and next conversion is not requested)
  - A/D converter goes to idle (power-down) state after A/D conversion finished
  - A/D converter does not go idle (power-down) state, it can start A/D sampling without the resumption time

- Interrupt request generation to CPU is provided for:
  - Conversion done interrupt (end of a logical channel conversion)
  - Group interrupted interrupt (a group processing is interrupted just before the logical channel conversion is to be started or in case of multiple conversion channel before the last conversion is started)
  - Range comparison interrupt
  - Pulse detection interrupt
- Conversion done interrupts of up to four logical channels can trigger DMA requests to transfer the A/D conversion results to memory. The conversion done interrupts of the group last logical channels are good candidates for DMA burst setup, since the group result registers can be read linearly
- Debug mode provides the possibility to freeze further A/D conversion processing at the end of the current conversion

2. Configuration and Block Diagram

This section shows a block diagram of A/D converter.

Figure 15-1 Block Diagram of A/D Converter



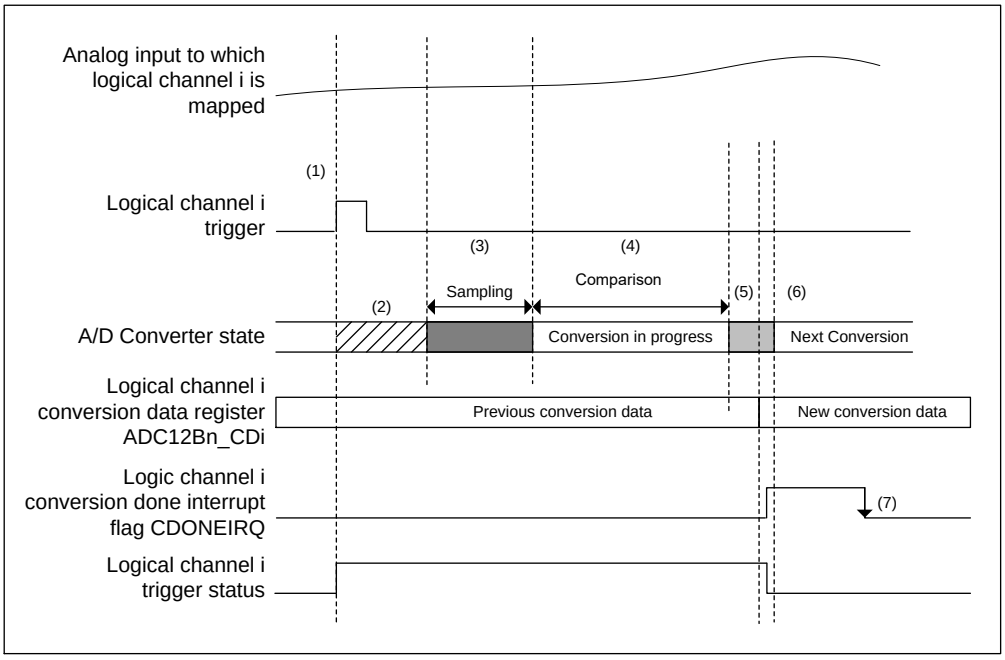
3. Operation of A/D Converter

A/D Converter operates using the successive approximation method with 12-bit or 10-bit or 8-bit resolution. There is one A/D Conversion Data Register per logical channel which is updated each time the assigned logical channel is converted. First four logical channels can be configured as multiple conversion channels or used for A/D Converter calibration.
The range comparator compares the converted values with the configured values in the threshold registers and accordingly generates an interrupt for "inside range" or "outside range", depending on the configuration. Any of eight range comparators can be configured for any of 64 logical channels.
A/D Pulse Detection function detects events of desired length and also filters parasitic inverted events. Each logical channel has a dedicated pulse detection function.

3.1. A/D Conversion Flow

The basic A/D conversion flow is shown on Figure 15-2.

Figure 15-2 A/D Conversion Flow



The important steps marked on the figure are:

(1) A/D conversion request (trigger) is issued for particular logical channel i and the corresponding trigger status is set.

(2) Period between the trigger and actual conversion start depends on current A/D Converter state condition

- If A/D Converter is in idle (power-down) state (power-down mode is enabled (ADC12Bn_CTRL.PDDMD = "0")) i.e. there are no trigger status set at the moment the trigger is issued, it is first waited until the resumption time (configured as A/D converter resumption time register (ADC12Bn_RT)) elapses. At the end of resumption time, the priority arbiter compares the priorities of all logical channels with set trigger status and inactive data protection feature. The conversion of the logical channel i starts when this channel wins the arbitration, i.e. after all logical channels with higher priority are converted.
- If A/D converter is active at the moment the trigger is issued, the operation is dependent on the setting of forced stop mode (ADC12Bn_CTRL.FSMD).

- Forced stop is enabled (ADC12Bn_CTRL.FSMD = "1"): Even if the sampling phase is started, an active A/D conversion can be interrupted. So, the logical channel i starts after interrupt operations instead of currently performed conversion when this channel wins the priority arbitration.
- Forced stop is disabled (ADC12Bn_CTRL.FSMD = "0"): The conversion of the logical channel i starts when this channel wins the priority arbitration and currently preformed conversion is finished, since an A/D conversion cannot be interrupted once its sampling phase is started. Accordingly, even if the logical channel i has the highest priority, in worst case it can happen that its conversion is delayed by the maximum configured time needed to convert one logical channel (including the case of multiple conversion channel that cannot be interrupted).

(3) Sampling period. A/D converter internal input signal level goes to the level of the analog input to which the logical channel i is mapped.

(4) A/D conversion.

(5) A/D conversion finalization period. The conversion result is stored in the dedicated conversion data register ADC12Bn_CDi, the conversion done interrupt flag is set and the trigger status is cleared.

(6) Started next conversion of the logical channel with the highest priority at the moment. If there is no trigger for any logical channel, the operation is dependent on the setting of power-down disable mode (ADC12Bn_CTRL.PDDMD).

- Power down disable mode (ADC12Bn_CTRL.PDDMD = "1"): Even if the A/D conversion is finished (and next conversion is not requested), A/D converter does not go idle (power-down) state. It can start A/D sampling without A/D converter resumption time.
- Power down enable mode (ADC12Bn_CTRL.PDDMD = "0"): A/D converter goes to idle (power-down) state after A/D conversion finished (and next conversion is not requested).

If the trigger status of any logical channel is already set, next conversion is started without A/D converter resumption time.

(7) After the conversion data register (ADC12Bn_CDi) is read or by writing "1" to the corresponding conversion done interrupt clear bit (ADC12Bn_CDONEIRQC0 to 1.CDONEIRQCi), the conversion done interrupt flag is cleared.

In case A/D Converter reconfiguration takes place during operation (A/D Converter is not in power-down state), the following rules are to be obeyed:

- Do not reconfigure logical channels belonging to the group that is currently converted.
- Set the trigger types (ADC12B_CHCTRL0 to 63.TRGTYP[1:0]) of the logical channels affected by the reconfiguration to software trigger only ("00" setting).
- Clear all the trigger status flags (ADC12Bn_TRGST0 to 1.TRGST and ADC12Bn_CHSTAT0 to 63.TRGST) of the affected logical channels, by writing "1" to the corresponding bits of ADC12Bn_TRGCL0 to 1 (or ADC12B_CHCTRL0 to 63.TRGCL) registers.
- Reconfigure the logical channels.

3.2. Logical Channel Mapping to Analog Input Signals

64 logical channels are mapped to up to 64 analog input signals, that need to be converted into the digital values. The mapping is done over ADC12Bn_CHCTRL0 to 63.CHNUM configuration fields in the way that CHNUM value controls the number of the analog input signal AN, that is propagated to the A/D Converter input in case the logical channel is converted.

Figure 15-3 illustrates an example of possible logical channel mapping to analog inputs. Each logical channel is mapped to exactly one analog input and it is allowed to map several logical channels to the same analog input (logical channels 4, 5, 6 and 62 are mapped to the analog input AN8).

Figure 15-3 Example of Logical Channel Mapping to Analog Inputs

Logical channel number	0	1	2	3	4	5	6	7	…	58	59	60	61	62	63
CHNUM bit field setting	52	14	63	0	8	8	8	32	…	1	2	3	28	8	4
Analog input signal	AN52	AN14	AN63	AN0	AN8	AN8	AN8	AN32	…	AN1	AN2	AN3	AN28	AN8	AN4

The described way of logical channel organization provides following benefits:

- Configurable grouping and consecutive conversion of channels independent on physical pin order.
- Mapping of several logical channels to the same analog input allows repetitive conversion of the same analog input with only one interrupt at the end.

3.3. Logical Channel Data Protection Function

Every logical channel has its own conversion data register ADC12Bn_CD. They are written by hardware at the end of conversion of the dedicated channel. The CPU can read the data registers any time. If a conversion is finished and the data of the previous conversion of the same channel has not been read out, previous data could be overwritten and previous conversion result lost. To avoid this the data protection function can be enabled so that the next conversion of this logical channel is not started until the previous data has been read out. The data protection function is controlled by the corresponding ADC12Bn_CHCTRL0 to 63.DP bits:

- If for some logical channel the dedicated ADC12Bn_CHCTRLi.DP bit is equal to "0", then conversion is continued and former conversion data are overwritten.
- In case of a regular logical channel, if the dedicated ADC12Bn_CHCTRLi.DP bit is equal to "1" and conversion done interrupt flag is set to "1", this logical channel cannot be selected for the conversion even though its trigger status may be set. The channel can be selected for conversion again after its conversion done interrupt flag has been cleared, e.g. by reading the conversion data register.
- In case of a multiple conversion logical channel
  - if the dedicated DP bit is equal to "1" and conversion done interrupt flag is set to "1" or
  - if the dedicated DP bit is equal to "1", group interrupted interrupt flag is set to "1", group interrupted interrupt is enabled and multiple conversions are already started, then the channel cannot be selected for conversion until its conversion done interrupt flag or group interrupted interrupt flag have been cleared.

Note:

1. It should set DP to "1" only for the logical channels configured as a group (for details about group configuration please see section "Group processing" in chapter "3. Operation of A/D ") having all its logical channels set to "Resume" (ADC12Bn_CHCTRL0 to 63,RSMRST = "01"). In case of other group configurations:

- *"Restart" setting (ADC12Bn_CHCTRL0 to 63.RSMRST = "10") - the group may be restarted before the conversion done interrupt of the last channel in the group (meaning the interrupt that indicates that the conversion of the entire group has been completed) is asserted, so the CPU does not know that there is already a result available in this channel.*
- *"Stop" setting (ADC12Bn_CHCTRL0 to 63.RSMRST = "00") - with the group interrupted interrupt enabled, it could be possible for the CPU to clear the conversion done interrupt flags of the affected channels, but the use case of both "Stop" and "Restart" is that the results of an interrupted group are not interesting, so they do not need to be protected.*

2. For groups consisting of only one multiple conversion channel, it should use DP as follows:

- *"Resume" setting (ADC12Bn_CHCTRL0 to 63.RSMRST = "01") - when setting DP to "1", disable the group interrupted interrupt (set the corresponding ADC12Bn_GRPIRQE0.GRPIRQE0 to 3 to "0"): The conversion result will be protected only after all conversions of the multiple conversion channel have been performed.*

If the corresponding ADC12Bn_GRPIRQE0.GRPIRQE0 to 3 is set to "1", then the conversion result will be protected at the time the channel is interrupted, and the conversions will not be resumed before the group interrupted interrupt flag has been cleared. When the conversion is resumed, the conversion counter is reset and first conversion is started. It means the multiple conversion cannot resume from the interrupted state.

- *"Restart" setting (ADC12Bn_CHCTRL0 to 63.RSMRST = "10") - when setting DP to "1", disable the group interrupted interrupt (set the corresponding ADC12Bn_GRPIRQE0.GRPIRQE0 to 3 to "0"): The conversion result will be protected only after all conversions of the multiple conversion channel have been performed. Even if the corresponding ADC12Bn_GRPIRQE0.GRPIRQE0 to 3 is set to "1", the conversion result will be protected only after all conversions of the multiple conversion channel have been performed.*
- *"Stop" setting (ADC12Bn_CHCTRL0 to 63.RSMRST = "00") - when setting DP to "1", enable the group interrupted interrupt (set the corresponding ADC12Bn_GRPIRQE0.GRPIRQE0 to 3 to "1"): The conversion result will be protected when the channel is interrupted or when all conversions have been completed.*

3.4. Logical Channel Triggering and Priority

Each logical channel has the configuration bit field ADC12B_CHCTRL0 to 63.TRGTYP[1:0], controlling the way a conversion request (trigger) can be issued. The following settings are possible:

- Software trigger only (TRGTYP = "00") - in order to set the dedicated trigger status flags (ADC12Bn_CHSTAT0 to 63.TRGST and ADC12Bn_TRGST0 to 1.TRGST) the corresponding software trigger bit (ADC12Bn_CHCTRL0 to 63.SWTRG) must be set to "1".
- Hardware or software trigger (TRGTYP = "01") - the corresponding trigger status flags can be set not only through software trigger bit, but also in case of the rising edge event on dedicated hardware trigger input.
- Trigger by completion of the preceding logical channel (TRGTYP = "10") - the trigger status flags are set only at the end of preceding logical channel conversion, i.e. updating of conversion data register ADC12Bn_CD of the preceding channel triggers this channel if the trigger status flags of the preceding channel are still set. This allows to combine several consecutive logical channels into a group that will be sequentially converted after the start channel is triggered. To configure a group, the start channel trigger type is set to software trigger only, hardware or software trigger or idle trigger, while all other channel trigger types are set to the preceding logical channel conversion completion. It does not make sense to configure the first logical channel to trigger type 2, since the channels would never be triggered.
- Idle trigger (TRGTYP = "11") - the trigger status flags are set whenever there is no active conversion request, i.e. there is no logical channel having trigger status flag set and inactive data protection function. Accordingly, trigger status flags of all logical channels with idle trigger type are set at the same time. The further processing depends on their priority. The idle trigger type allows to form a regular group (the start channel is set to idle trigger and consecutive channels have preceding channel completion trigger type).

Once the logical channel trigger status is set to "1", the channel priority (bit fields ADC12Bn_CHCTRL0 to 63.CHPRI) configures the conversion order of logical channels with the inactive data protection function. In case of the same priority, active trigger status and inactive data protection function, the logical channel having lower number wins the priority arbitration. The priority can be set in the range 0 - 15. The setting 0 (CHPRI = "0000") is the highest priority, whereas the lowest possible priority setting is 15 (CHPRI = "1111").

Forced stop mode is enabled (ADC12Bn_CTRL.FSMD = "1"):

When the higher priority conversion was requested during an active A/D conversion, the operation of interrupt is dependent on the setting of the channel priority (ADC12Bn_CHCTRL0 to 63.CHPRI) only.

- If the priority (CHPRI) of the requested channel is lower than active channel, it can be interrupted and the conversion of the channel with the higher priority is started.
- If the priority (CHPRI) is higher or same, it cannot be interrupted.

Figure 15-4 Example of Logical Channel Priority Configuration

Logical channel number	0	1	2	3	4	5	6	7	…	58	59	60	61	62	63
CHPRI bit field setting	0	12	13	5	1	6	3	10	…	1	2	3	14	15	15
Data protection function	1	0	0	0	0	0	0	0	…	0	1	0	0	0	0
Trigger status flag	1	1	1	0	1	0	0	1	…	1	0	1	1	1	1

The example on Figure 15-4 would result in following priority arbitration and corresponding logical channel conversion sequence:

(1) Logical channel 4.

Logical channel 0 has higher priority, but its data protection function is active.

(2) Logical channel 58.

This logical channel has the same priority as logical channel 4, but it is converted as second since its number is higher.

(3) Logical channel 60.

Although the channel 59 has higher priority, its trigger status is not set and even the data protection is active. Moreover, logical channel 6 has the same priority and even lower number, but its trigger status is not set.

(4) Logical channel 7.

Even though the logical channels 3 and 5 have higher priority, their trigger status is not set.

(5) Logical channel 1.

(6) Logical channel 2.

(7) Logical channel 61.

(8) Logical channel 62.

It has lower number then the channel 63.

(9) Logical channel 63.

3.5. Group Processing

A group is defined by the trigger type configuration of consecutive logical channels. The first channel of the group has the trigger type set to software trigger only, hardware or software trigger or idle trigger. If the following channel trigger type is not set to preceding logical channel conversion completion, the group consists of only one channel. Otherwise, the group continues until the last channel in a row having trigger type set to preceding logical channel conversion completion. After the first channel of the group is triggered and converted, automatically the second channel is triggered and so on until the whole group is converted.

Figure 15-5 Group Configuration Examples

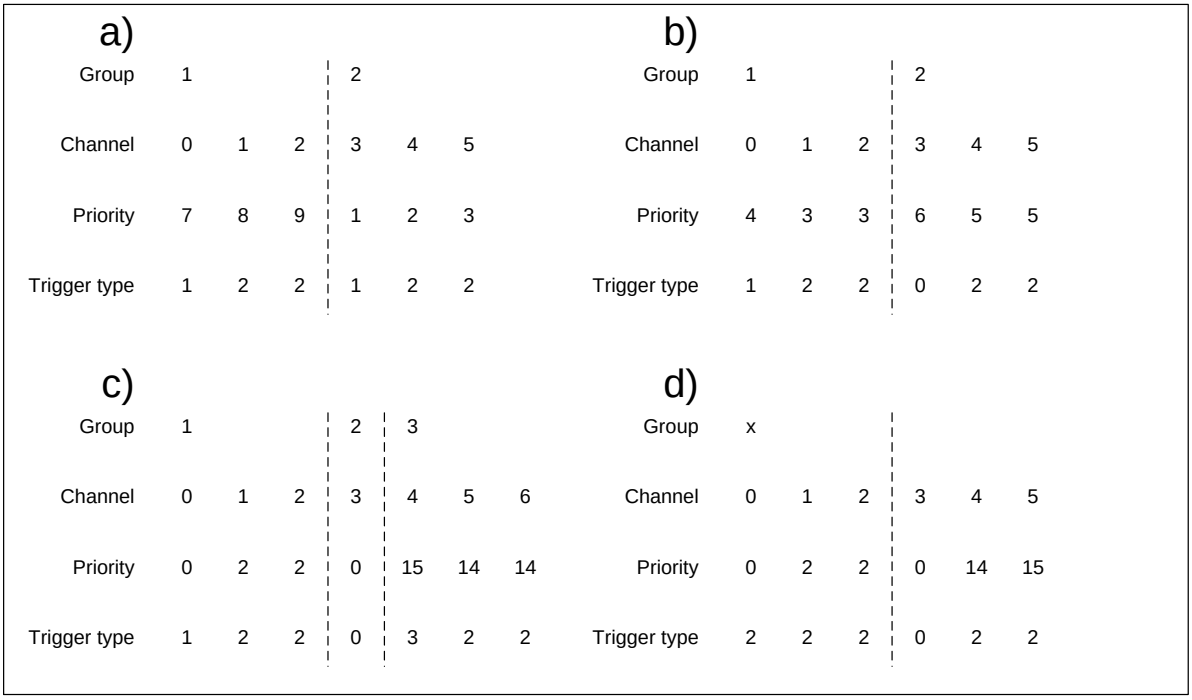


Figure 15-5 shows some possible group configurations:

a) In the example two groups are configured, the group 1 consists of logical channels 0, 1 and 2 and the group 2 includes logical channels 3, 4 and 5. If both groups are triggered (trigger status is set for the starting channels 0 and 3), the group 2 would be converted first due to higher priority setting. Since the first channel in the groups have the highest priority, the groups are "re-triggerable" (if the next starting channel trigger appears before the group conversion is finished, the conversion of the group would be restarted).

b) Here the first channel of the groups has a lower priority then the remaining channels within the groups, hence it is not possible to re-trigger the group processing. The processing of the group 2 can be started only by the software trigger of the logical channel 3. The group 1 allows the triggering by the software or hardware trigger of the channel 0.

c) This example contains three group, where the group 2 consists of only one logical channel 3. The group 3 shows the configuration of the group starting with the idle trigger type, i.e. the trigger status of the logical channel 4 is set to "1" if there is no logical channel having trigger status flag set and inactive data protection function.

d) In this case first three logical channels are set to trigger type 2 and since there will be no event of preceding channel conversion completion, the logical channels 0, 1 and 2 will never be converted.

Between the different groups (the group 1, the group 2): the priority of all the logical channels should be configured as [the group 1 > the group 2] or [the group 1 < the group 2].

(Example: When the priorities of all the channels included in the group 1 are set as 7-9, all the channels of the group 2 should be set lower (or higher) than 7-9.)

After a group processing is started (at least the conversion of the first channel of the group is started), it can be interrupted.

Forced stop mode is enabled (ADC12Bn_CTRL.FSMD = "1"): If the triggered channel is higher priority than the current channel in the group, current conversion of group processing is interrupted after interrupt operation and the conversion of the channel with the higher priority is started.

Forced stop mode is disabled (ADC12Bn_CTRL.FSMD = "0"): If an A/D conversion is already started, it cannot be stopped. Hence if the triggered channel is higher priority than the next channel in the group, the group processing is interrupted before the conversion start of the next logical channel in the group.

The configuration bit fields ADC12B_CHCTRL0 to 63.RSMRST define how the group processing is to be continued after the higher priority conversion requests are executed. Note that RSMRST setting of the first channel in the group does not matter, since the group cannot be interrupted before the conversion of its first channel started.

If stop group setting (RSMRST[1:0] = "00") is configured for a logical channel and the group is interrupted just before conversion start of that channel, the current processing of the group is stopped. Consequently, the trigger status flag of the logical channel (set by the conversion completion of the preceding channel in the group) is cleared.

Figure 15-6 shows an example of stopping a group processing.

(1) Logic channel 0 trigger is issued and processing of the group 0 is initiated.

(2) Processing of the group 0, conversion of logical channels 0 -> 1 -> 2 -> start of channel 3 conversion.

(3) Logic channel 6 trigger is issued during the conversion of channel 3.

(4) The operation is dependent on the setting of forced stop mode (ADC12Bn_CTRL.FSMD).

Forced stop mode is enabled (ADC12Bn_CTRL.FSMD = "1"):

Since the priority of the channel 6 is higher than priority of the channel 3, the processing of the group 0 is interrupted during the conversion of channel 3. After interrupted operation, processing continues with the conversion of the channel 6.

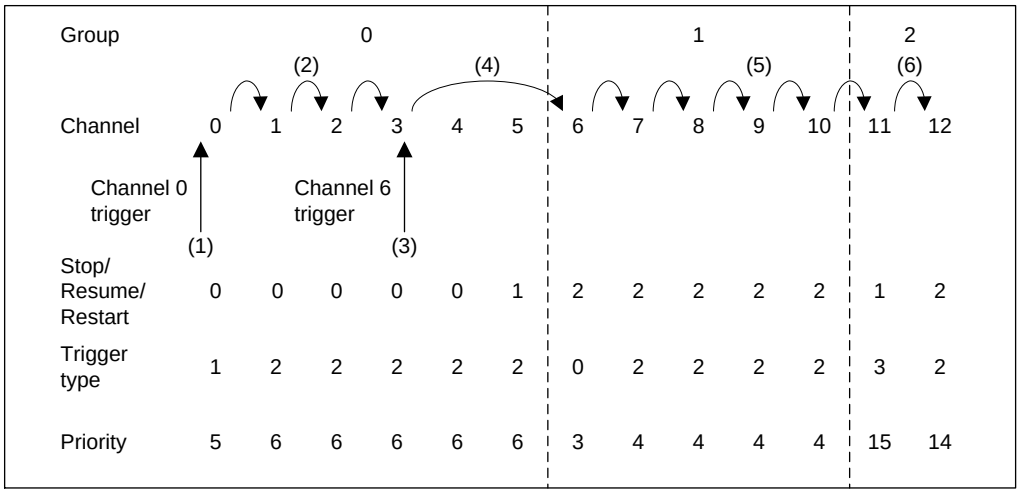
Forced stop mode is disabled (ADC12Bn_CTRL.FSMD = "0"):

Since the priority of the channel 6 is higher than priority of the channel 4, the processing of the group 0 is interrupted after the conversion of channel 3 is finished. Moreover, the trigger status flag of the channel 4 is cleared and processing continues with the conversion of the channel 6.

(5) Group 1 is converted.

(6) Idle trigger group 2 is converted. Group 0 will not be processed until next channel 0 trigger appears.

Figure 15-6 Stopping of the Group Processing



Resume group setting (RSMRST[1:0] = "01") for a logical channel means that if the group is interrupted just before conversion start of that channel, the current processing of the group will be resumed with this channel after higher priority conversions are done.

When the channel which is configured as resume (ADC12Bn_CHCTRL0 to 63.RSMRST = "01") is in interrupted operation, group interrupted interrupt flag (corresponding bits of ADC12Bn_CHSTAT0 to 63.GRPIRQ and ADC12Bn_GRPIRQ0 to 1.GRPIRQ) is set to "1" whenever converting of other channel occurred.

Figure 15-7 shows an example of resuming a group processing (Forced stop mode is enabled (ADC12Bn_CTRL.FSMD = "1")).

Figure 15-8 shows an example of resuming a group processing (Forced stop mode is disabled (ADC12Bn_CTRL.FSMD = "0")).

(1) Logic channel 0 trigger is issued and processing of the group 0 is initiated.

(2) Processing of the group 0, conversion of logical channels 0 -> 1 -> 2 -> start of channel 3 conversion.

(3) Logic channel 6 trigger is issued during the conversion of channel 3.

(4) The operation is dependent on the setting of forced stop mode (ADC12Bn_CTRL.FSMD).

Forced stop mode is enabled (ADC12Bn_CTRL.FSMD = "1", Figure 15-7):

Since the priority of the channel 6 is higher than priority of the channel 3, the processing of the group 0 is interrupted during the conversion of channel 3.

Forced stop mode is disabled (ADC12Bn_CTRL.FSMD = "0", Figure 15-8):

Since the priority of the channel 6 is higher than priority of the channel 4, the processing of the group 0 is interrupted after the conversion of channel 3 is finished.

(5) Group 1 is converted.

(6) The operation is dependent on the setting of forced stop mode (ADC12Bn_CTRL.FSMD).

Forced stop mode is enabled (ADC12Bn_CTRL.FSMD = "1", Figure 15-7):

After group 1 is finished, processing of group 0 is continued with conversion of channel 3 since its trigger status flag is still set.

Forced stop mode is disabled (ADC12Bn_CTRL.FSMD = "0", Figure 15-8):

After group 1 is finished, processing of group 0 is continued with conversion of channel 4 since its trigger status flag is still set.

(7) At the end of group 0 processing, idle trigger group 2 is converted.

Figure 15-7 Resuming of the Group Processing (Forced Stop Mode is Enabled (ADC12Bn_CTRL.FSMD = "1"))

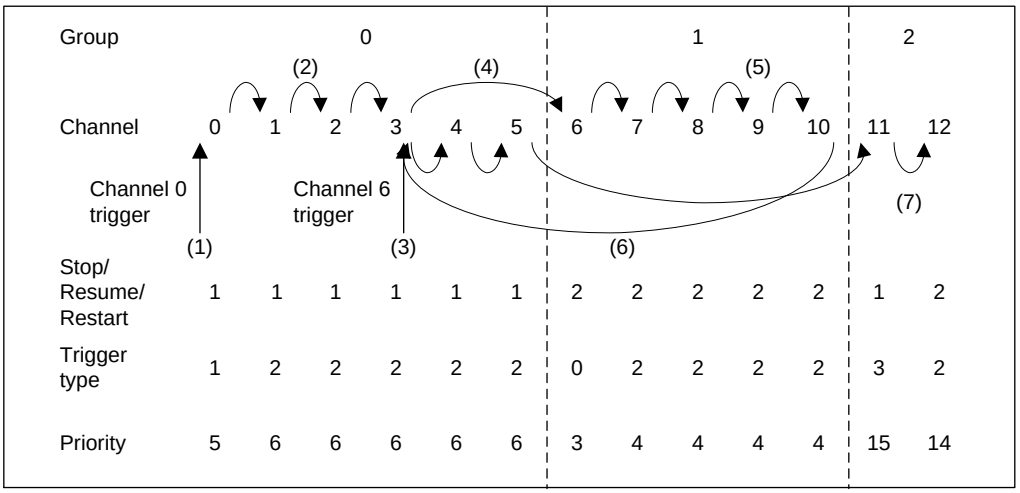
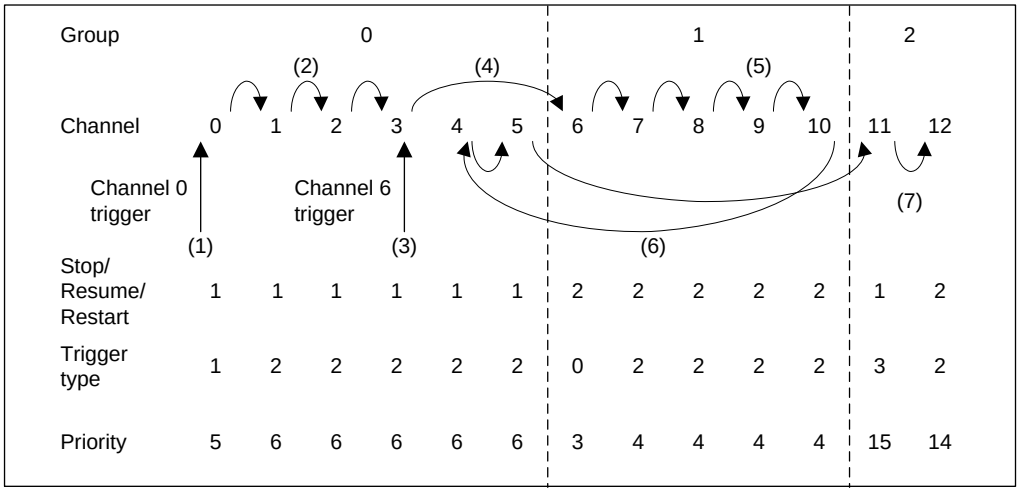


Figure 15-8 Resuming of the Group Processing (Forced Stop Mode is Disabled (ADC12Bn_CTRL.FSMD = "0"))



Restart group setting (RSMRST[1:0] = "10") for a logical channel configures that if the group is interrupted just before conversion start of that channel, its trigger status is cleared and the processing of the group will be restarted after higher priority conversions are done:

- with the closest previous channel of the group set to resume (RSMRST = "1"), accordingly the trigger status flag of that channel is set
- with the first channel of the group (consequently, the trigger status flag of the first channel is set) if there are no previous channels set to resume.

Figure 15-9 shows an example of restarting a group processing with its first channel.

(1) Logic channel 0 trigger is issued and processing of the group 0 is initiated.

(2) Processing of the group 0, conversion of logical channels 0 -> 1 -> 2 -> start of channel 3 conversion.

(3) Logic channel 6 trigger is issued during the conversion of channel 3.

(4) The operation is dependent on the setting of forced stop mode (ADC12Bn_CTRL.FSMD).

Forced stop mode is enabled (ADC12Bn_CTRL.FSMD = "1"):

Since the priority of the channel 6 is higher than priority of the channel 3, the processing of the group 0 is interrupted during the conversion of channel 3 and trigger status of channel 0 is set.

Forced stop mode is disabled (ADC12Bn_CTRL.FSMD = "0"):

Since the priority of the channel 6 is higher than priority of the channel 4, the processing of the group 0 is interrupted after the conversion of channel 3 is finished and trigger status of channel 0 is set.

(5) Group 1 is converted.

(6) After group 1 is finished, processing of group 0 is restarted with conversion of channel 0 since its trigger status flag is set to "1".

(7) Group 0 is converted.

(8) At the end of group 0 processing, idle trigger group 2 is converted.

Figure 15-9 Restarting of the whole Group Processing

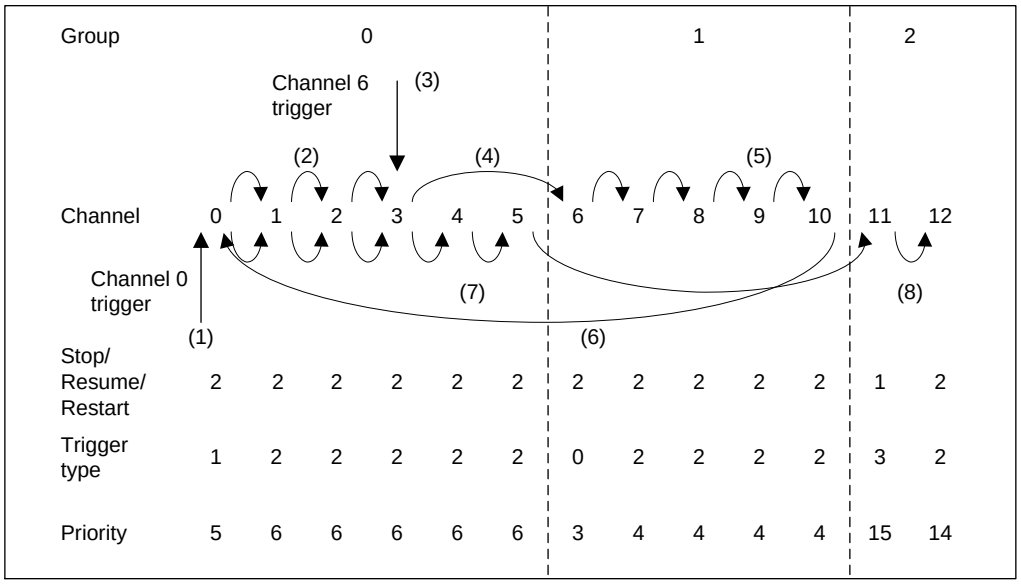


Figure 15-10 shows an example of restarting a group processing with its first channel during the conversion of the last channel of the group processing (Forced stop mode is enabled (ADC12Bn_CTRL.FSMD = "1")).

Figure 15-11 shows an example of restarting a group processing with its first channel during the conversion of the last channel of the group processing (Forced stop mode is disabled (ADC12Bn_CTRL.FSMD = "0")).

(1) Logic channel 0 trigger is issued and processing of the group 0 is initiated.

(2) Processing of the group 0, conversion of logical channels 0 -> 1 -> 2 -> 3 -> 4 -> start of channel 5 conversions (last channel of the group 0).

(3) Logic channel 6 trigger is issued during the conversion of channel 5.

The operations after (3) are dependent on the setting of forced stop mode (ADC12Bn_CTRL.FSMD).

Forced stop mode is enabled (ADC12Bn_CTRL.FSMD = "1", Figure 15-10):

(4) Since the priority of the channel 6 is higher than priority of the channel 5, the processing of the group 0 is interrupted during the conversion of channel 5 and trigger status of channel 0 is set.

(5) Group 1 is converted.

(6) After group 1 is finished, processing of group 0 is restarted with conversion of channel 0 since its trigger status flag is set to "1".

(7) Group 0 is converted.

(8) At the end of group 0 processing, idle trigger group 2 is converted.

Forced stop mode is disabled (ADC12Bn_CTRL.FSMD = "0", Figure 15-11):

(4) After the conversion of channel 5 is finished (group 0 is converted), processing continues with the conversion of the channel 6.

(5) Group 1 is converted.

(6) At the end of group 1 processing, idle trigger group 2 is converted (group 0 is not restarted).

Figure 15-10 Restarting a Group Processing with Its First Channel during the Conversion of the Last Channel of the Group Processing (Forced Stop Mode is Enabled (ADC12Bn_CTRL.FSMD = "1"))

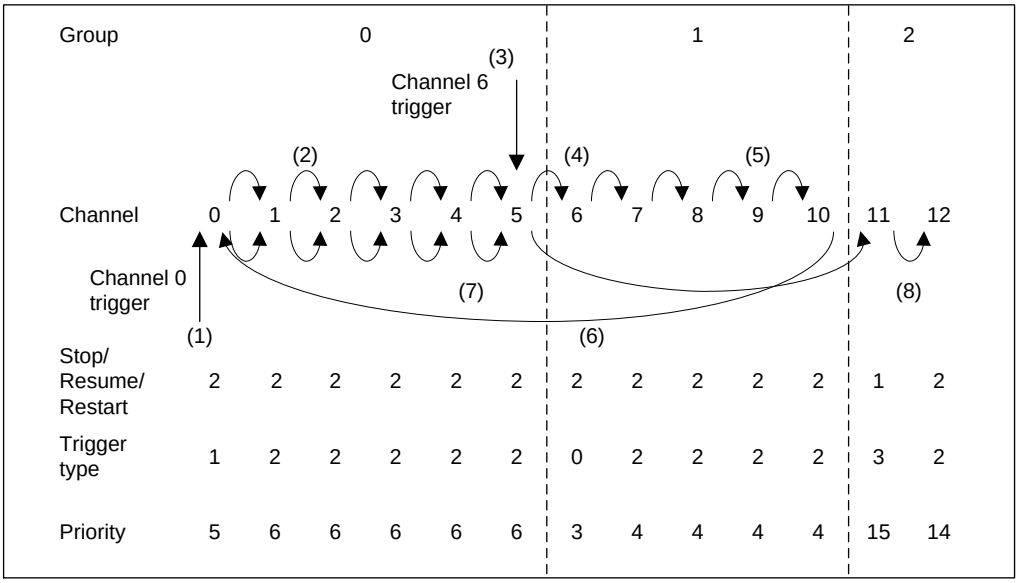


Figure 15-11 Restarting a Group Processing with Its First Channel during the Conversion of the Last Channel of the Group Processing (Forced Stop Mode is Disabled (ADC12Bn_CTRL.FSMD = "0"))

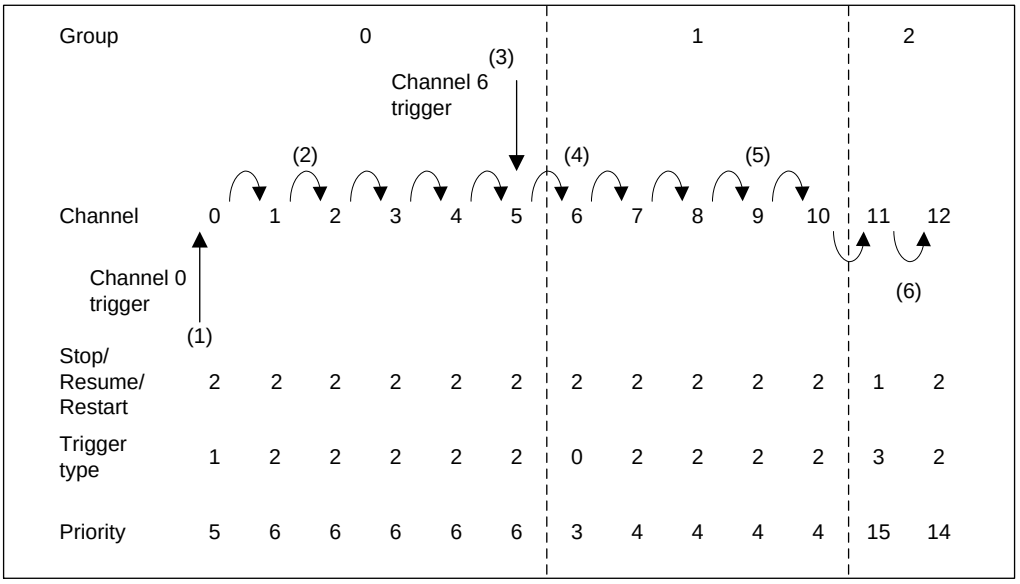


Figure 15-12 shows an example of restarting a group processing with its last converted channel set to resume. This way subgroups can be formed within a group.

(1) Logic channel 0 trigger is issued and processing of the group 0 is initiated.

(2) Processing of the group 0, conversion of logical channels 0 -> 1 -> 2 -> start of channel 3 conversion.

(3) Logic channel 6 trigger is issued during the conversion of channel 3.

(4) The operation is dependent on the setting of forced stop mode (ADC12Bn_CTRL.FSMD).

Forced stop mode is enabled (ADC12Bn_CTRL.FSMD = "1"):

Since the priority of the channel 6 is higher than priority of the channel 3, the processing of the group 0 is interrupted during the conversion of channel 3 and trigger status of channel 1 is set (channel 1 is the last converted channel set to resume).

Forced stop mode is disabled (ADC12Bn_CTRL.FSMD = "0"):

Since the priority of the channel 6 is higher than priority of the channel 4, the processing of the group 0 is interrupted after the conversion of channel 3 is finished and trigger status of channel 1 is set (channel 1 is the last converted channel set to resume).

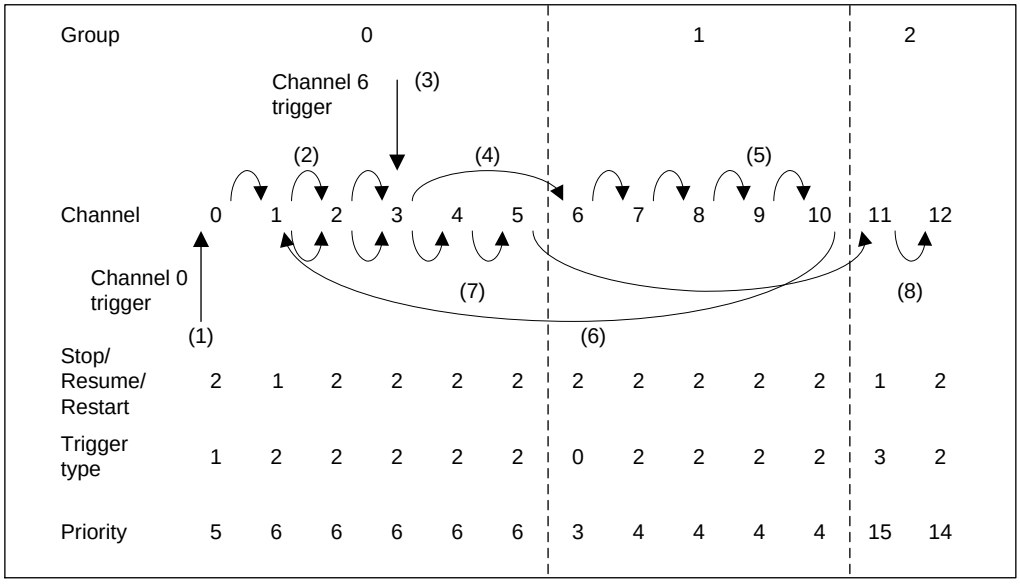
(5) Group 1 is converted.

(6) After group 1 is finished, processing of group 0 is restarted with conversion of channel 1 since its trigger status flag is set to "1".

(7) Subgroup (channels 1 till 5) of the group 0 is converted.

(8) At the end of group 0 processing, idle trigger group 2 is converted.

Figure 15-12 Restarting of the Group Processing with a Subgroup



3.6. Multiple Conversion Logical Channels

First four logical channels can be configured so that after the channel is triggered, several conversions (up to 16) are performed and conversion result is accumulated. These channels are referred to as multiple conversion channels. The following is valid for multiple conversion channels:

- All features and rules provided for regular logical channels (mapping to analog inputs, data protection, triggering rules, priority, channel grouping and behavior within a group) apply also to multiple conversion channels.
- The number of conversions to be performed when the channel is triggered is defined by ADC12Bn_MCCTRL0 to 3.CNVNUM bit field. If CNVNUM is set to "0", a multiple conversion channel behaves exactly like a regular channel.
- Dedicated A/D conversion data registers ADC12Bn_CD0 to 3 hold the sum of the single conversion results. Accordingly, ADC12Bn_CD0 to 3 registers are extended to 16 bits.
- Forced stop mode is enabled (ADC12Bn_CTRL.FSMD = "1"): In case the conversion request with higher priority is issued during multiple conversion, the multiple conversion channel is interrupted after interrupt operation, and the conversion of the channel with the higher priority is started. Forced stop mode is disabled (ADC12Bn_CTRL.FSMD = "0"): Once the first conversion of multiple

conversion channel is started, the channel processing cannot be interrupted by higher priority requests until CNVNUM+1 conversions are finished, if the dedicated bit ADC12Bn_MCCTRL0 to 3.ICIRQY (intra-channel interruptability) is set to "1". The setting ADC12Bn_MCCTRL0 to 3.ICIRQY = "0" allows to interrupt multiple conversion channel processing between single conversions, if a higher priority request appears.

- In the case that a multiple conversion channel is interrupted between/during single conversions, the dedicated group interrupted interrupt flags (ADC12Bn_CHSTAT0 to 3.GRPIRQ and ADC12Bn_GRPIRQ0.GRPIRQ) are set. The corresponding Resume/Restart/Stop bit field (ADC12Bn_CHCTRL0 to 3.RSMRST) and intra-channel interruptibility setting (ADC12Bn_MCCTRL0 to 3.ICIRQY) determine the further channel processing (after higher priority requests are converted).
  Table 15-1 shows the description about multiple conversion after interrupted (ADC12Bn_CTRL.FSMD = "0").
  Table 15-2 shows the description about multiple conversion after interrupted (ADC12Bn_CTRL.FSMD = "1").

Table 15-1 Description About Multiple Conversion after Interrupted (ADC12Bn_CTRL.FSMD = "0")

Settings		Description of Trigger Status and Conversion Counter after Interrupted
ADC12Bn_MCCTRL0 to 3. ICIRQY	ADC12Bn_CHCTRL0 to 63. RSMRST[1:0]	
"0" (can be interrupted between single conversion)	"00" (Stop)	Imposes the trigger status flag clearing. Keeps current value of the conversion counter. It is cleared by next conversion request.
	"01" (Resume)	Keeps the trigger status flag set and current value of the conversion counter.
	"10" (Restart)	The trigger status flag stays set only if the channel is first channel in the group, otherwise the trigger status of the channel is cleared and instead the trigger status flag of the first channel in the subgroup is set. The conversion counter is cleared immediately. It means that the accumulated conversion result before the interruption is not really usable because it is not known how many conversions are done.
"1" (cannot be interrupted between single conversion)	Don't care	Don't care. Multiple conversion is not interrupted.

Table 15-2 Description About Multiple Conversion after Interrupted (ADC12Bn_CTRL.FSMD = "1")

Settings		Description of Trigger Status and Conversion Counter after Interrupted
ADC12Bn_MCCTRL0 to 3. ICIRQY	ADC12Bn_CHCTRL0 to 63. RSMRST[1:0]	
"0" (can be interrupted between single conversion)	"00" (Stop)	Imposes the trigger status flag clearing. Keeps current value of the conversion counter. It is cleared by next conversion request.
	"01" (Resume)	Keeps the trigger status flag set. The conversion counter is cleared immediately. It means that the accumulated conversion result before the interruption is not really usable because it is not known how many conversions are done.
	"10" (Restart)	The trigger status flag stays set only if the channel is first channel in the group, otherwise the trigger status of the channel is cleared and instead the trigger status flag of the first channel in the subgroup is set. The conversion counter is cleared immediately. It means that the accumulated conversion result before the interruption is not really usable because it is not known how many conversions are done.
"1" (cannot be interrupted between single conversion)	"00" (Stop)	Imposes the trigger status flag clearing. Keeps current value of the conversion counter. It is cleared by next conversion request.
	"01" (Resume)	Keeps the trigger status flag set and current value of the conversion counter.
	"10" (Restart)	The trigger status flag stays set only if the channel is first channel in the group, otherwise the trigger status of the channel is cleared and instead the trigger status flag of the first channel in the subgroup is set. The conversion counter is cleared immediately. It means that the accumulated conversion result before the interruption is not really usable because it is not known how many conversions are done.

- In the case that range comparison is enabled for a multiple conversion channel, every conversion result is passed to the range comparator.
- For notes of data protection about multiple conversion, refer to section "Logical channel data protection function" in chapter "3. Operation of A/D ".

3.7. Forced Stop

■ Forced stop mode (ADC12Bn_CTRL.FSMD = "1"):
During an active A/D conversion, it can be forced stop by the following ways.

- Writing ADC12Bn_CTRL.FSTP = "1":
  All channel's trigger status flags are reset.
  The current A/D conversion is stopped after interrupt operation.
- Writing "1" to the corresponding bits of ADC12Bn_TRGCL1 to 0.TRGCL or ADC12Bn_CHCTRLi.TRGCL:
  The corresponding trigger status flag is reset immediately.
  The current A/D conversion is stopped after interrupt operation,

■ Not forced stop mode (ADC12Bn_CTRL.FSMD = "0"):
The A/D conversion cannot stop.

- Writing ADC12Bn_CTRL.FSTP = "1": invalid.
- Writing "1" to the corresponding bits of ADC12Bn_TRGCL1 to 0.TRGCL or ADC12Bn_CHCTRLi.TRGCL:
  The corresponding trigger status flag is reset immediately.
  The current A/D conversion is not stopped, but A/D conversion result (the following) is not update.
  - A/D conversion done interrupt flag (the corresponding bits of ADC12Bn_CDONEIRQ1 to 0.CDONEIRQ or ADC12Bn_CHSTATi.CDONEIRQ)
  - A/D conversion data (the corresponding register of ADC12Bn_CD0 to 63)

- Converted logical channel number (ADC12Bn_STAT.ACH[5:0] (ADC12Bn_CTRL.ACHMD is set as "1"))

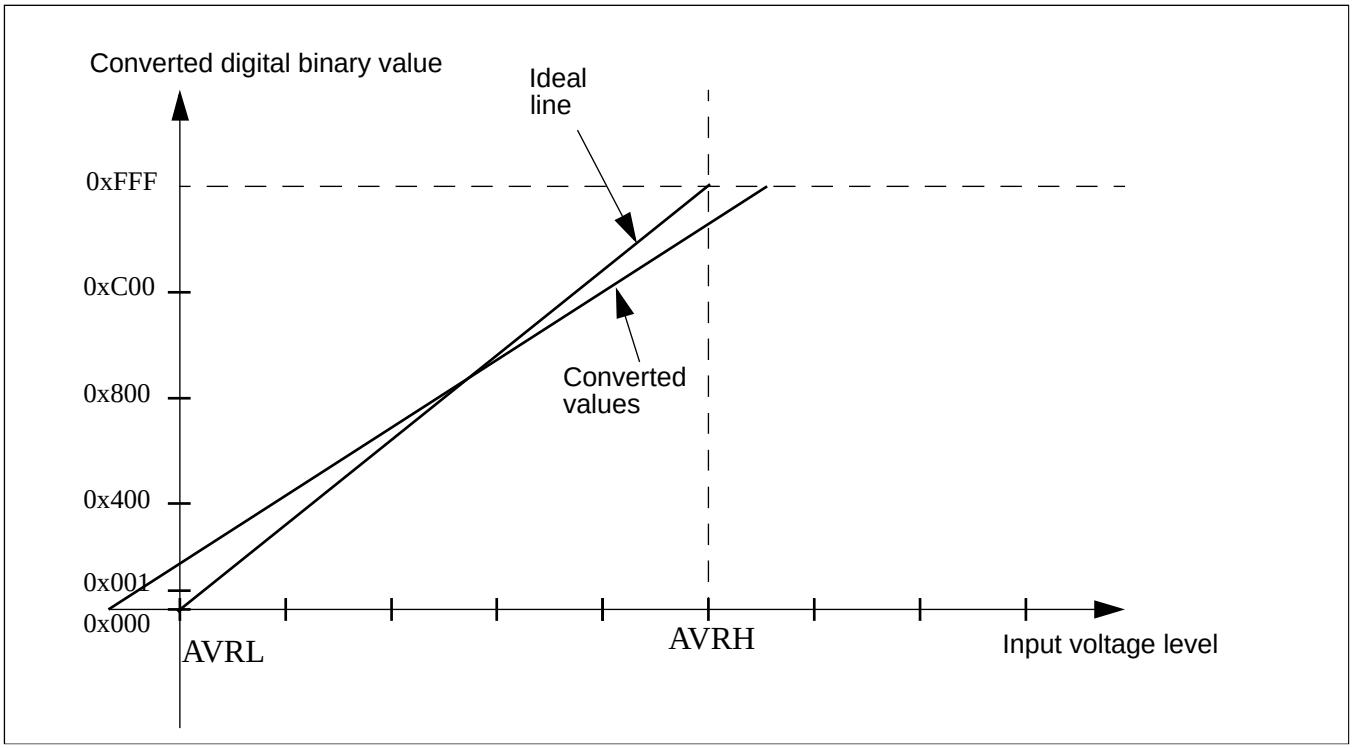
Even if the trigger status flag of the idle trigger channel i (ADC12Bn_CHCTRLi.TRGTYP[1:0] = "11") is cleared (by writing "1" to the corresponding bits of ADC12Bn_TRGCL1 to 0 or ADC12Bn_CHCTRLi.TRGCL), it is set "1" again immediately when the trigger status flag of all the channels is "0".

3.8. A/D Converter Calibration

Additional feature of multiple conversion logical channels is selection of A/D Converter reference voltages AVRH or AVRL for conversion instead of dedicated analog input signals. If AVRL conversion is wanted it is enough to set one of ADC12Bn_MCCTRL0 to 3.AVRLSEL bits to "1" and trigger corresponding channel. For AVRH conversion it is needed to set one of ADC12Bn_MCCTRL0 to 3.AVRHSEL bits to "1", set corresponding ADC12Bn_MCCTRL0 to 3.AVRLSEL bit to "0" and trigger the dedicated channel.

If AVRHSEL or ARVRLSEL bits is set to "1", during the conversion of the corresponding channel mapping of the logical channel to an analog inputs is disabled.

Figure 15-13 Example of Converted Digital Value Dependence on Input Voltage Level



This feature can be used to perform A/D Converter calibration. As it is shown on Figure 15-13 ideal characteristics of A/D Converter would have:

- transition between digital values 0x000 and 0x001 at AVRL + 0.5 LSB input voltage level and
- transition between digital values 0xFFEh and 0xFFF at AVRH - 1.5 LSB input voltage level.

If this is not the case, A/D Converter can be calibrated by performing following steps:

- Set gain correction setting (ADC12Bn_GCV.GCV) to "0".
- Set a multiple conversion channel to highest priority and its AVRLSEL bit to "1".
- For different ADC12Bn_OCV.OCV settings trigger the multiple conversion channel, i.e. perform AVRL voltage conversion. It is better to configure multiple conversions of AVRL and calculate average result (accumulated result divided by the number of executed conversions).
- Find ADC12Bn_OCV.OCV setting x for which the transition between digital values 0x000 and 0x001 occurs.
- Set AVRLSEL bit back to "0".
- Set AVRHSEL bit to "1".
- For different ADC12Bn_OCV.OCV settings trigger the multiple conversion channel, i.e. perform AVRH voltage conversion. It is better to configure multiple conversions of AVRH and calculate average result.
- Find ADC12Bn_OCV.OCV setting y for which the transition between digital values 0xFFE and 0xFFF occurs.
- Set A/D Converter offset compensation setting register ADC12Bn_OCV to the value calculated as "(x+y)/2 + 2".
- Set AVRHSEL bit back to "0".
- Set AVRLSEL bit to "1".
- For different ADC12Bn_GCV.GCV settings trigger the multiple conversion channel, i.e. perform AVRL voltage conversion. It is better to configure multiple conversions of AVRL and calculate average result.

- Find ADC12Bn_GCV.GCV setting z for which the transition between digital values 0x000 and 0x001 occurs.
- Set A/D Converter gain compensation setting register ADC12Bn_GCV to the value calculated as "z + 1".
- Set AVRLSEL and AVRHSEL bits to "0".
- Trigger and perform further logical channel conversions (analog inputs AN to which are logical channels mapped are converted by calibrated A/D Converter).

3.9. DMA Transfer Function

There are four conversion done DMA triggers available. They can be configured through the corresponding ADC12Bn_CDDS0 to 3 registers:

- The bit field CDCHNUM specifies the logical channel number whose conversion done interrupt flag issues a DMA request,
- CDCHEN bit controls enabling/disabling of the DMA request.

The feature provides an efficient way to transfer A/D conversion results to memory. Selecting of the group last logical channels through the four CDCHNUM bit fields can configure four different groups for DMA transfers. Since the group is always made of consecutive logical channels and the corresponding conversion data registers can be read linearly, the DMA transfer for the whole group is possible after the conversion of the last channel of the group is finished.

3.10. Range Comparator Function

The range comparator offers eight comparison groups with an upper and a lower threshold register each. The 64 logical channels can be enabled for range comparison and assigned to one of the eight comparators individually.

If range comparator is enabled for a logical channel, it provides two result flags:

- Interrupt flags ADC12Bn_RCIRQ0 to 1.RCIRQ and ADC12Bn_CHSTAT0 to 63.RCIRQ, signaling that the A/D conversion result is outside the range or inside the range ("inverted" operation, i.e. ADC12Bn_CHCTRL0 to 63.RCINVSEL = "1").
- Over threshold flags ADC12Bn_RCOTF0 to 1.RCOTF and ADC12Bn_CHSTAT0 to 63.RCOTF, showing that the A/D conversion value exceeded the upper threshold in the case of outside range detection.

Furthermore, each logical channel can be enabled to send an interrupt request to the CPU, if dedicated flags ADC12Bn_RCIRQ0 to 1.RCIRQ63 to 0 and ADC12Bn_CHSTAT0 to 63.RCIRQ are set.

The range comparator can choose 12/8-bit mode by the full range comparator mode (ADC12Bn_CTRL.FRCMD).

ADC12Bn_CTRL.FRCMD	Description
"1" : 12-bit range comparator	ADC12Bn_FRCOL0 to 11/FRCOH0 to 11 are used for range comparator upper/lower threshold value. Selection of one of eight available range comparator threshold values for the logical channel is configured by corresponding ADC12Bn_CHCTRL0 to 63.RCSEL[2:0] bit fields. The upper/lower comparator compares the 12 bits, 10 bits or 8 bits of the A/D conversion result. ADC12Bn_CTRL.RES[1:0] define this resolution. − If 12-bit resolution (ADC12Bn_CTRL.RES[1:0] = "x0"): Compares the 12 bits of A/D conversion result. − If 10-bit resolution (ADC12Bn_CTRL.RES[1:0] = "01"): Compares the 10 bits of A/D conversion result. − If 8-bit resolution (ADC12Bn_CTRL.RES[1:0] = "11"): Compares the 8 bits of A/D conversion result.
"0" : 8-bit range comparator	ADC12Bn_RCOL0 to 7/RCOH0 to 7 are used for range comparator upper/lower threshold value. Selection of one of eight available range comparator threshold values for the logical channel is configured by corresponding ADC12Bn_CHCTRL0 to 63.RCSEL[2:0] bit fields. The upper/lower comparator compares the upper 8 bits of the A/D conversion result.

Figure 15-14 shows the 8-bit range comparator structure.

Figure 15-15 shows the 12-bit range comparator structure.

Figure 15-14 8-Bit Range Comparator Structure

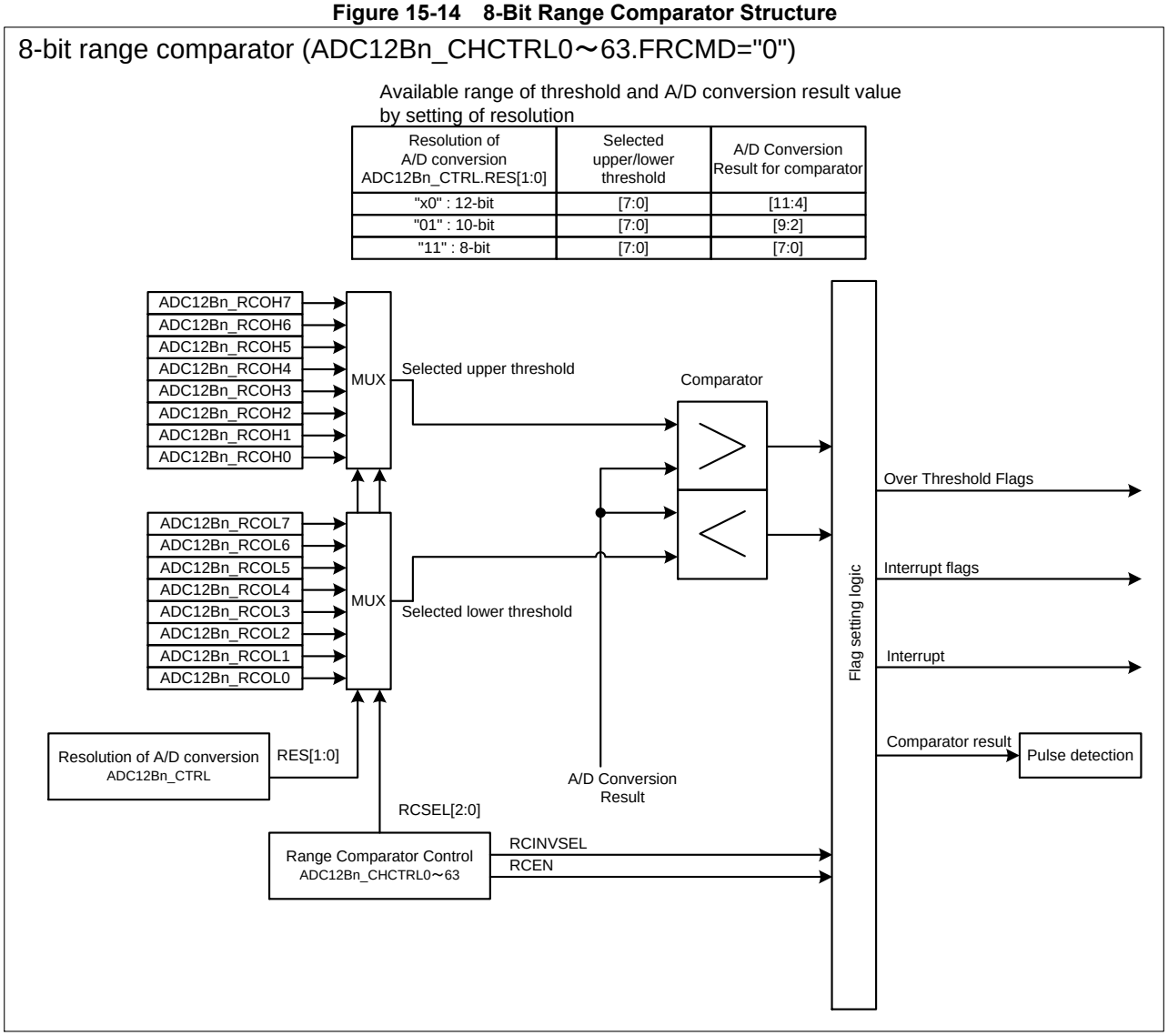
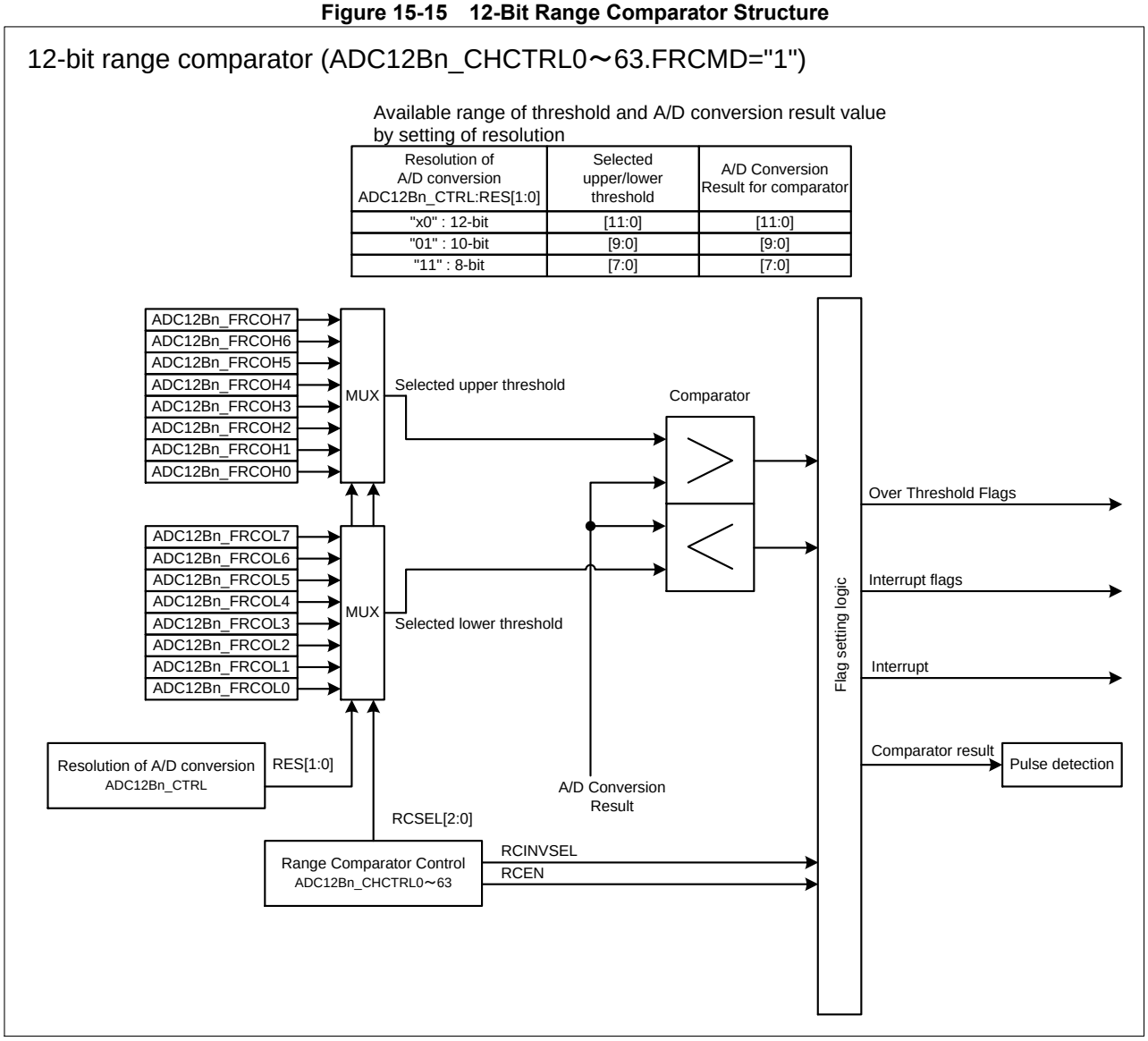


Figure 15-15 12-Bit Range Comparator Structure



3.11. Pulse Detection Function

The result of the comparison from the range comparator can be filtered by the pulse detection function. For every logical channel, the pulse detection function has a pair of reload registers to store the initial value for the positive and negative down counters (ADC12Bn_PCCTRL0 to 63.PCTPRL[7:0] and ADC12Bn_PCCTRL0 to 63.PCTNRL[4:0]). The positive and the negative counters decrement on positive and negative events obtained from the result of the comparison done by the range comparator.

The features of the pulse detection function are:

- Detect events with desired length
- Filter parasitic inverted events
- Each logical channel has a pulse detection function module associated with it
- Interrupt can be generated on detection of a pulse.

The output of the range comparator for particular logical channel signifies either a positive event or a negative event depending on the configuration of ADC12Bn_CHCTRL0 to 63.RCINVSEL register in the dedicated channel control register and the converted digital value of the A/D Converter as explained in Table 15-3 "Generation of positive/negative events". Whenever a positive event occurs the corresponding positive counter ADC12B_PCCTRL0 to 63.PCTPCT is decremented. Similarly, the dedicated negative counter ADC12B_PCCTRL0 to 63.PCTNCT decrements with each negative event. The purpose of the positive counter is to detect consecutive range comparator events of desired length. The negative counter can be used to force a restart of the positive counter if a negative events of a certain length are detected due to spikes, noise etc.

Table 15-3 Generation of Positive/Negative Events

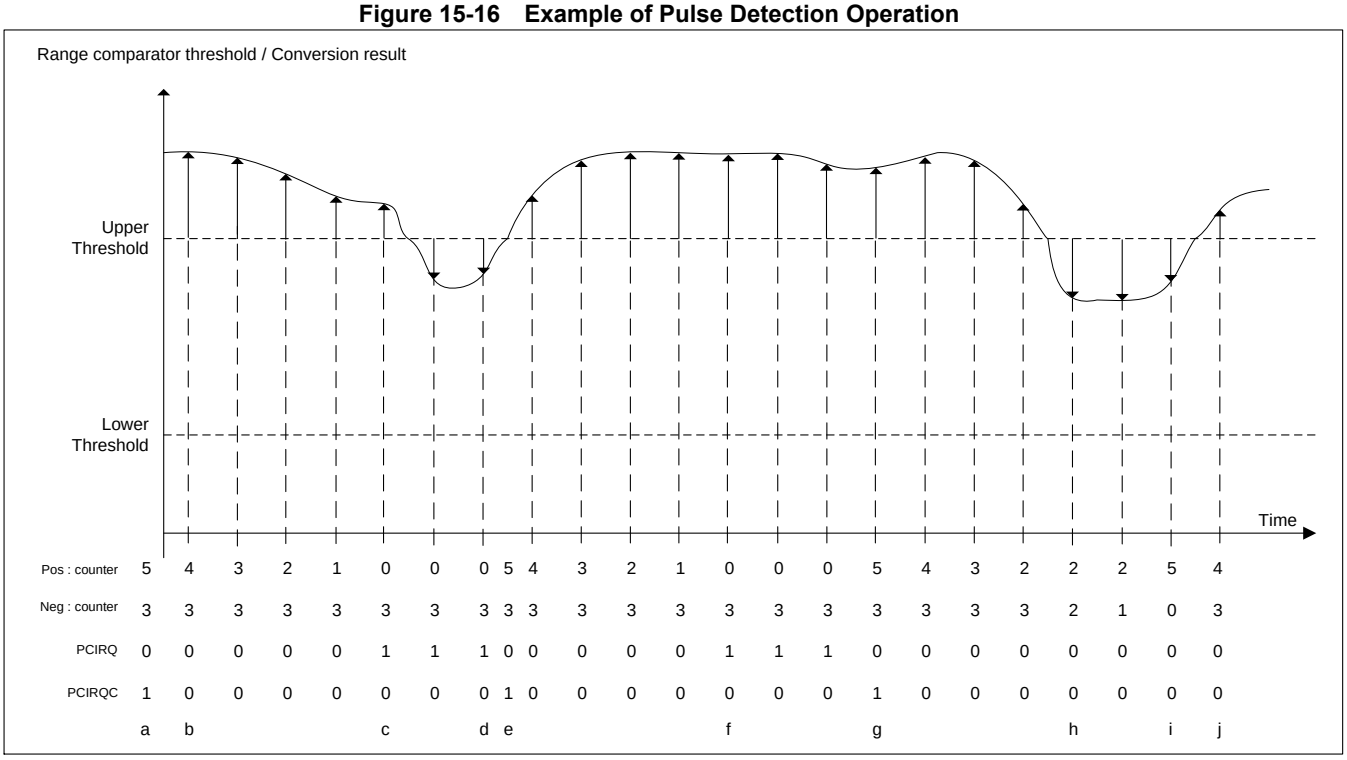
Inverted Range Selection RCINVSEL Bit	Range Comparator Output	Events
0 (configured for “outside range”)	inside range	Negative event
	outside range	Positive event
1 (configured for “inside range”)	outside range	Negative event
	inside range	Positive event

The following steps describe the working principle.

- The positive counter is decremented with each positive event of the corresponding logical channel.
- The corresponding pulse counter interrupt flags (ADC12Bn_CHSTAT0 to 63.PCIRQ and ADC12Bn_PCIRQ0 to 1.PCIRQ63 to 0) are set as the positive counter reaches zero. This flag remains set until it is cleared through writing dedicated ADC12Bn_PCIRQC0 to 1.PCIRQC63 to 0 bit to "1". The positive counter and the negative counter are stopped as long as PCIRQ flag of the channel is "1".
- If PCIRQ is set and the corresponding enable bit ADC12B_PCIRQE0 to 1.PCIRQE63 to 0 is equal to "1", an interrupt is generated.
- The negative counter is decremented with each negative event of the corresponding logical channel except while the corresponding PCIRQ flag is set.
- Positive counter is reloaded with the value set in the reload register ADC12Bn_PCCTRL0 to 63.PCTPRL when:
  - Negative counter reaches zero,
  - "1" is written to dedicated ADC12Bn_PCIRQC0 to 1.PCIRQC63 to 0 bit.
- Negative counter is reloaded with the value set in the reload register ADC12Bn_PCCTRL0 to 63.PCTNRL when:
  - Any positive event occurs
  - "1" is written to dedicated ADC12Bn_PCIRQC0 to 1.PCIRQC63 to 0 bit
  - The positive counter reaches zero and PCIRQ flag is set. The negative counter will hold the reload value as long as PCIRQ flag is not cleared.

The Figure 15-16 shows the operation of the pulse detection function for channel 0 with ADC12Bn_CHCTRL0.RCINVSEL = "0" configured for outside range, reload register ADC12Bn_PCCTRL0.PCTPRL = "101" and reload register ADC12Bn_PCCTRL0.PCTNRL = "011".

Figure 15-16 Example of Pulse Detection Operation



a) Reload counters with appropriate reload value by writing "1" to ADC12Bn_PCIRQC0.PCIRQC0.

b) Positive counter decrements with positive events.

c) Positive counter expires (becomes equal to "0"), the pulse counter interrupt flag PCIRQ is set.

d) A series of negative events does not decrement the negative counter as the PCIRQ flag is set.

e) The pulse counter interrupt flag PCIRQ is cleared and the positive as well as the negative counter are reloaded.

f) Positive counter expires and PCIRQ flag is to "1".

g) Software clears PCIRQ flag and reloads positive and negative counter.

h) Negative event decrements negative counter.

i) Negative counter expires and reloads positive counter.

j) Positive counter decrements and negative counter reloads with positive event.

3.12. Debug Mode

When the ADC12Bn_CTRL.DBGE bit is set to "1" and the processor is in debug state, the A/D Converter completes the current conversion, but further conversion is stopped. When the processor leaves debug state or ADC12Bn_CTRL.DBGE is set to "0", conversion continues with the next channel from where it had stopped.

For the definition of debug state, refer to Section 12.8 of the Arm Cortex-R5 Technical Reference Manual.

The ADC12Bn_STAT.BUSY flag is not affected even while ADC12Bn_CTRL.DBGE bit is set to "1" and the processor is in debug state: If all trigger status bits are cleared, ADC12Bn_STAT.BUSY flag is set to "0", and the A/D Converter goes to idle (power-down) state ; if any trigger status bit is set again, the A/D Converter leaves idle state, and the ADC12Bn_STAT.BUSY flag is set to "1" after the resumption time elapses.

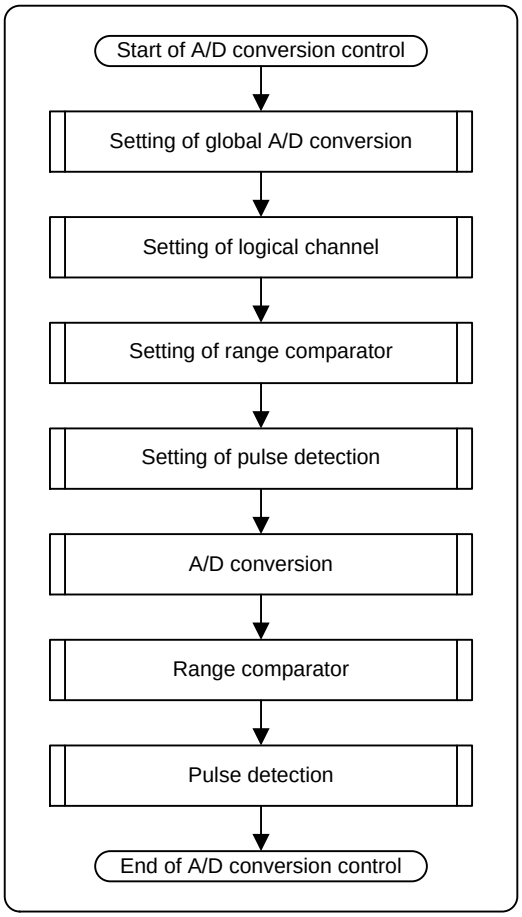
4. Setup Procedure Examples

This section shows examples of setup procedure of A/D converter.

4.1. Control of A/D Conversion

Figure 15-17 shows main flow of controlling A/D converter.

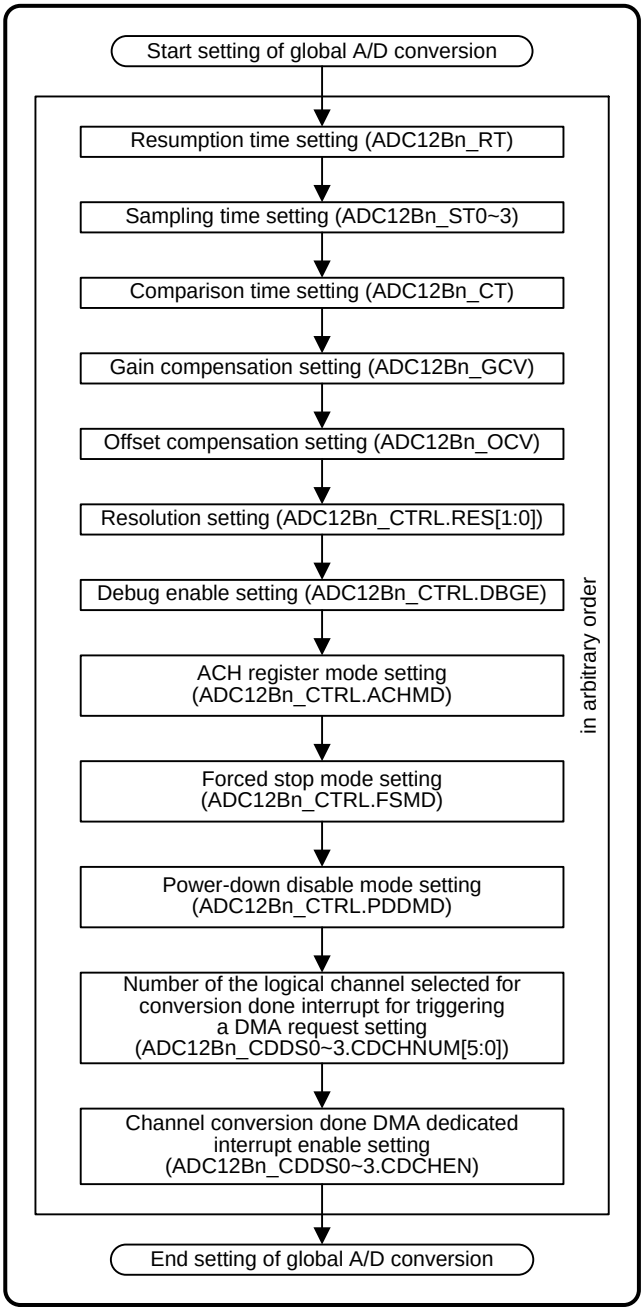
Figure 15-17 Main Flow of Controlling A/D Converter



4.2. Setting of Global A/D Conversion

Figure 15-18 shows the setting procedure of global A/D conversion.

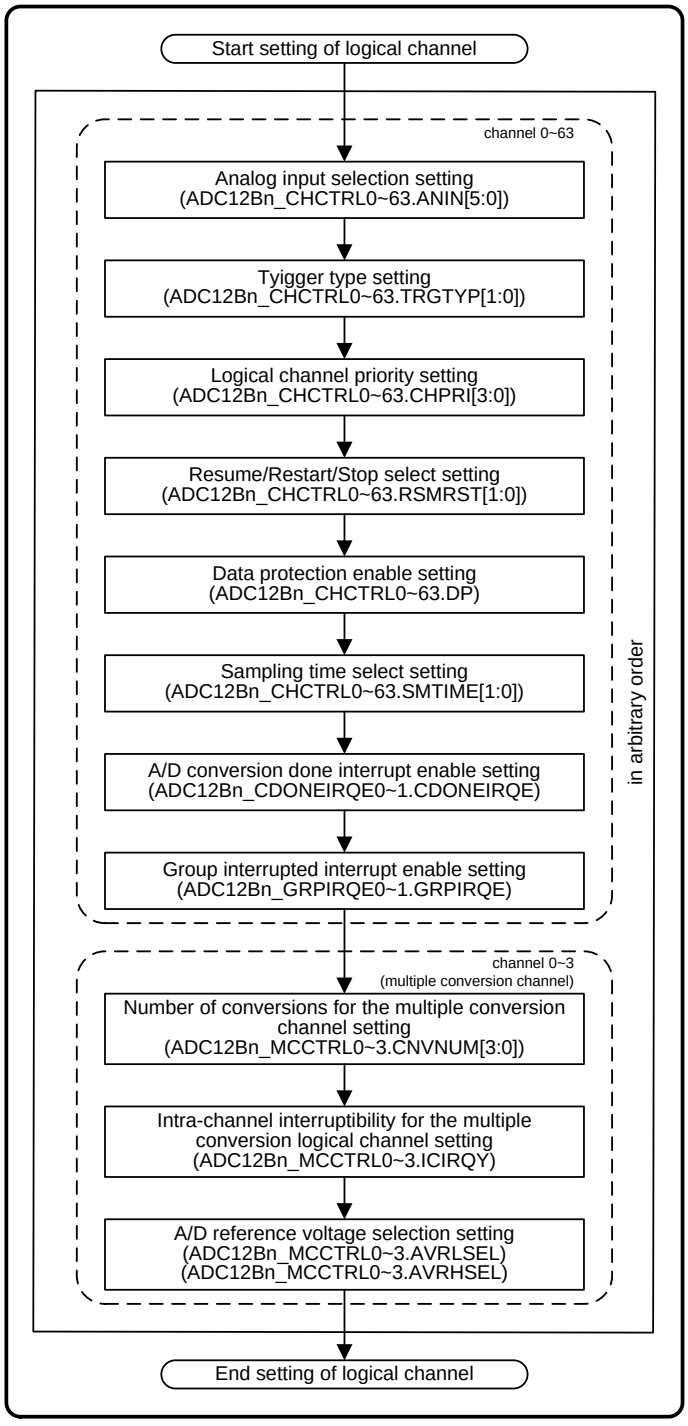
Figure 15-18 Setting of Global A/D Conversion



4.3. Setting of Logical Channel

Figure 15-19 shows the setting procedure of logical channel.

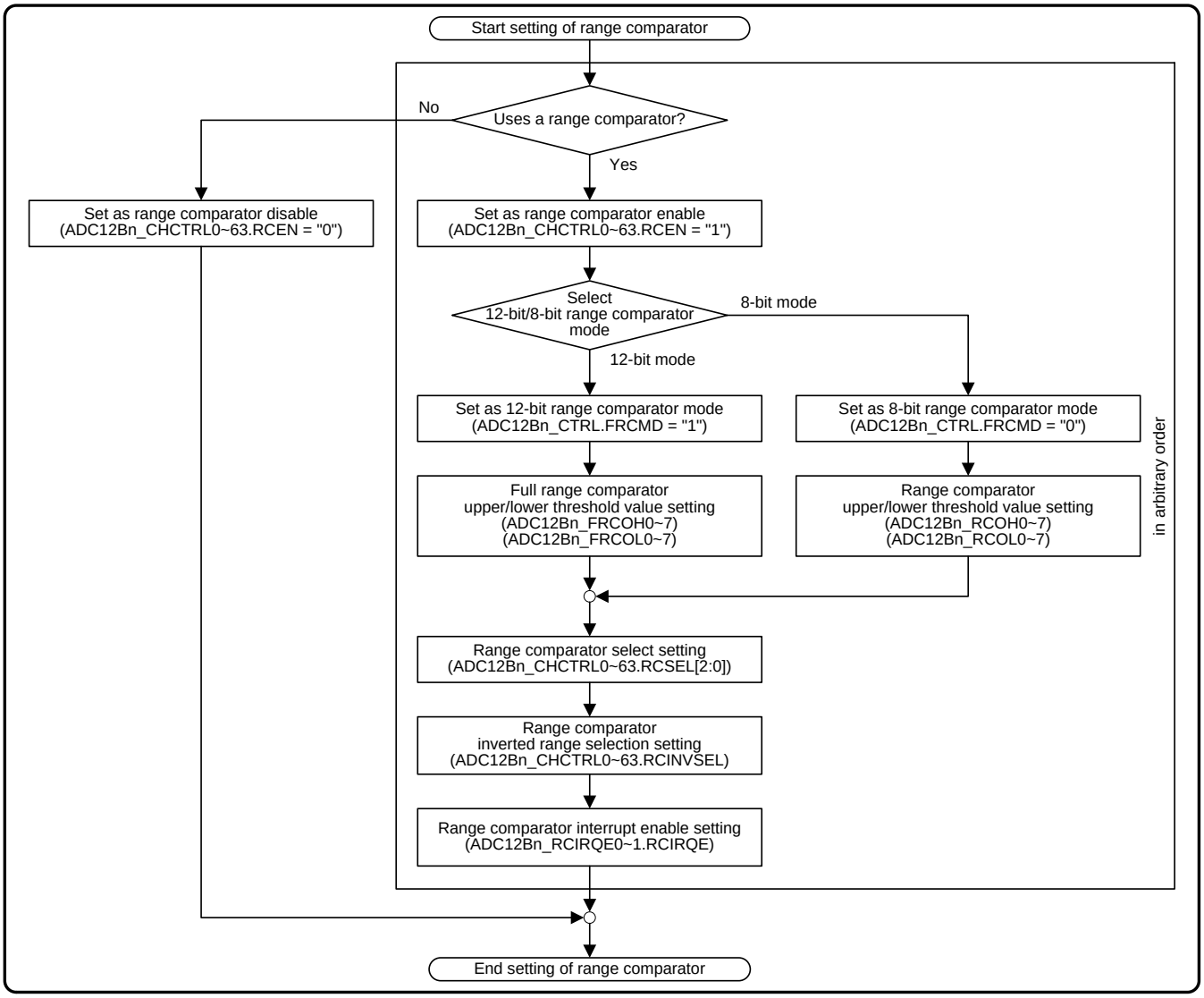
Figure 15-19 Setting of Logical Channel



4.4. Setting of Range Comparator

Figure 15-20 shows the setting procedure of range comparator.

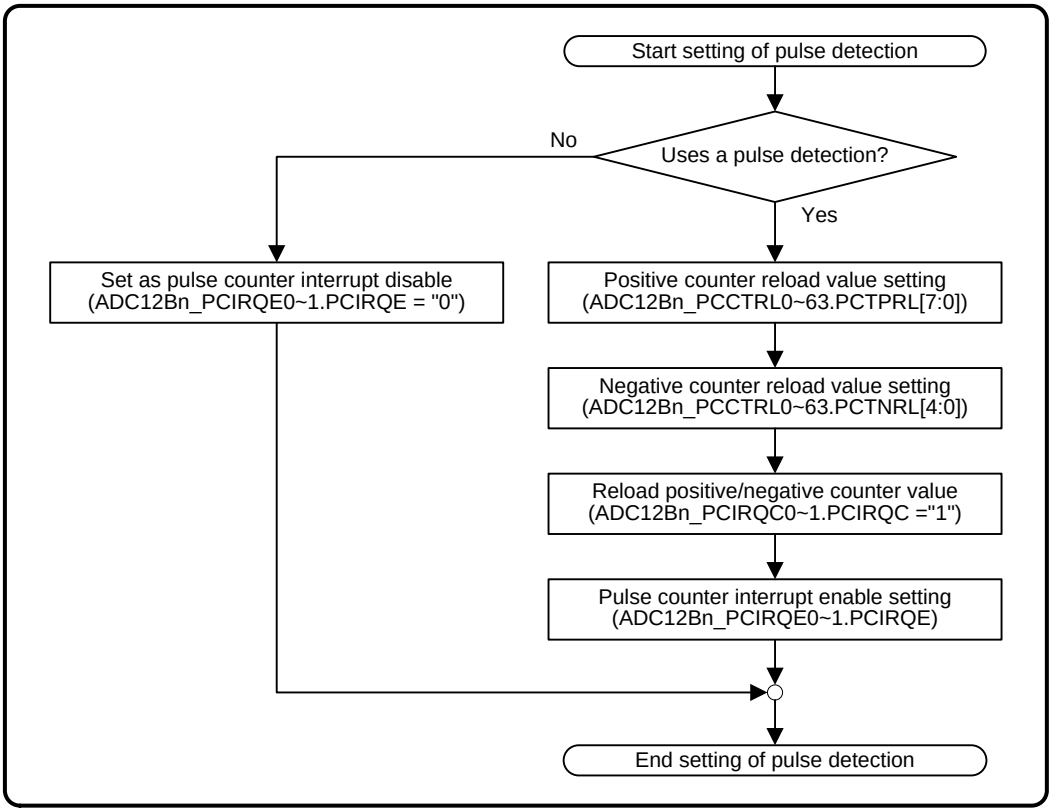
Figure 15-20 Setting of Range Comparator



4.5. Setting of Pulse Detection

Figure 15-21 shows setting procedure of pulse detection.

Figure 15-21 Setting of Pulse Detection

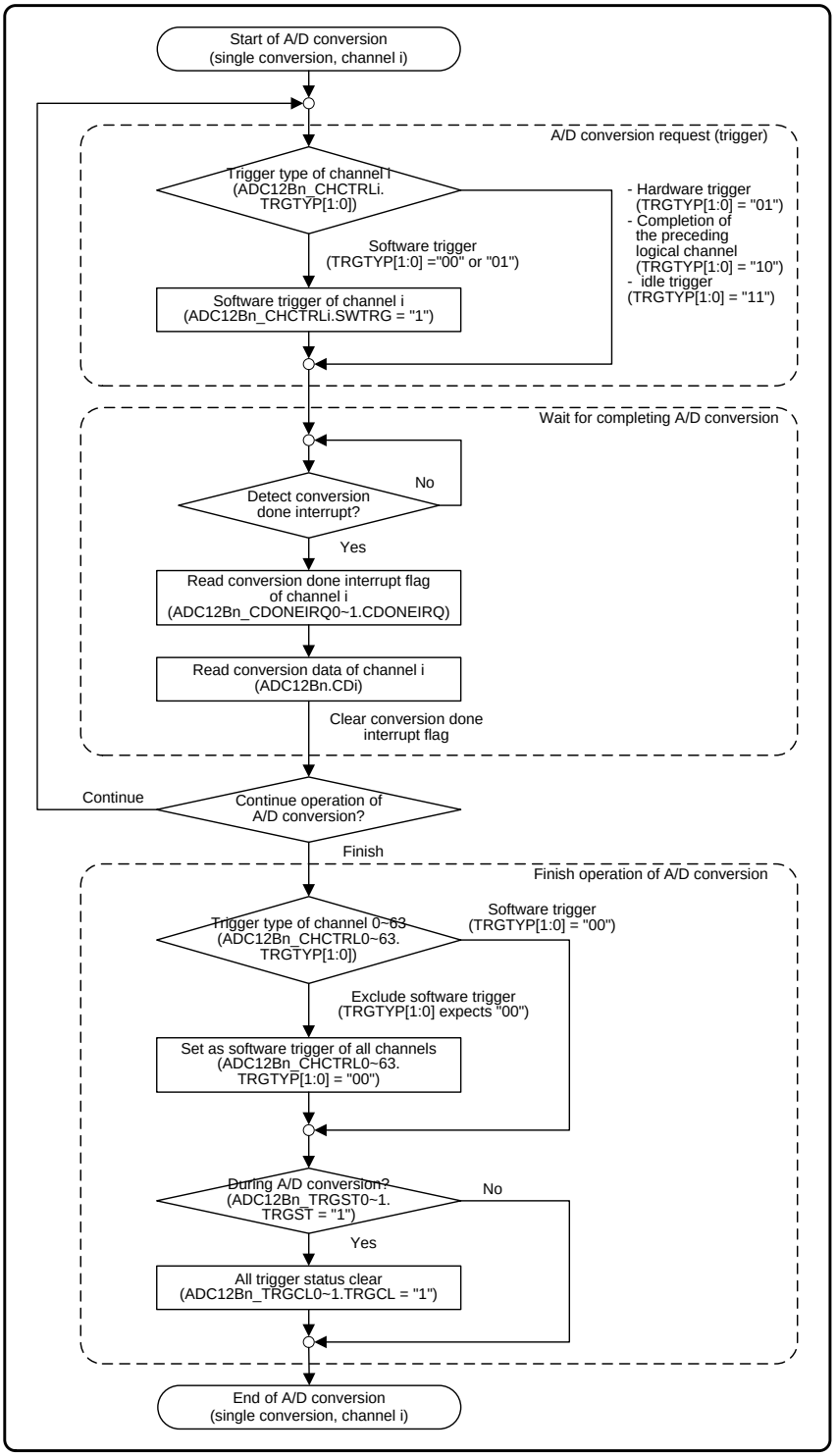


4.6. A/D Conversion

- Single conversion

Figure 15-22 shows the example of A/D conversion (single conversion) operation.

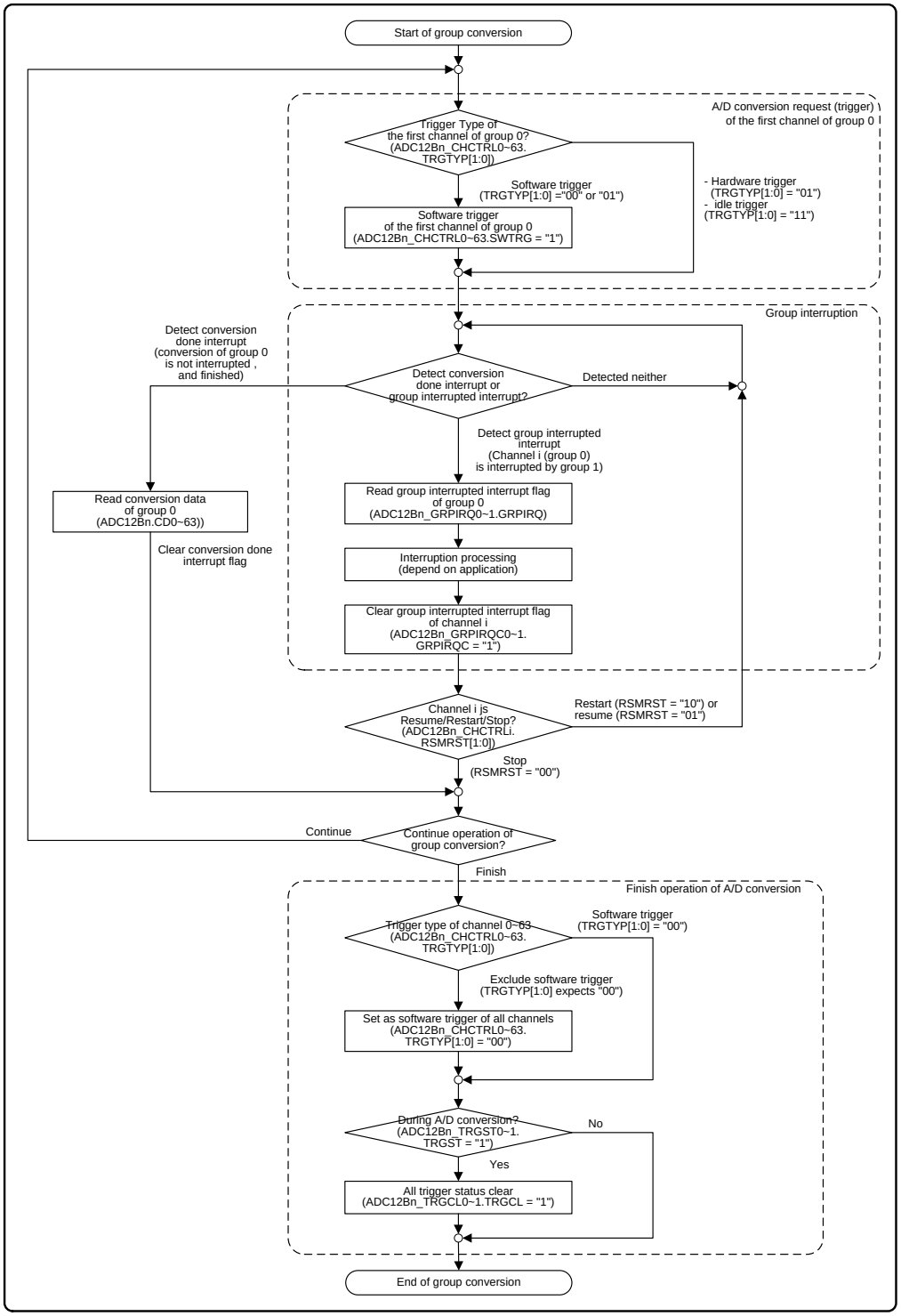
Figure 15-22 A/D Conversion (Single Conversion, Channel i)



■ Group conversion

Figure 15-23 shows the example of group conversion operation.

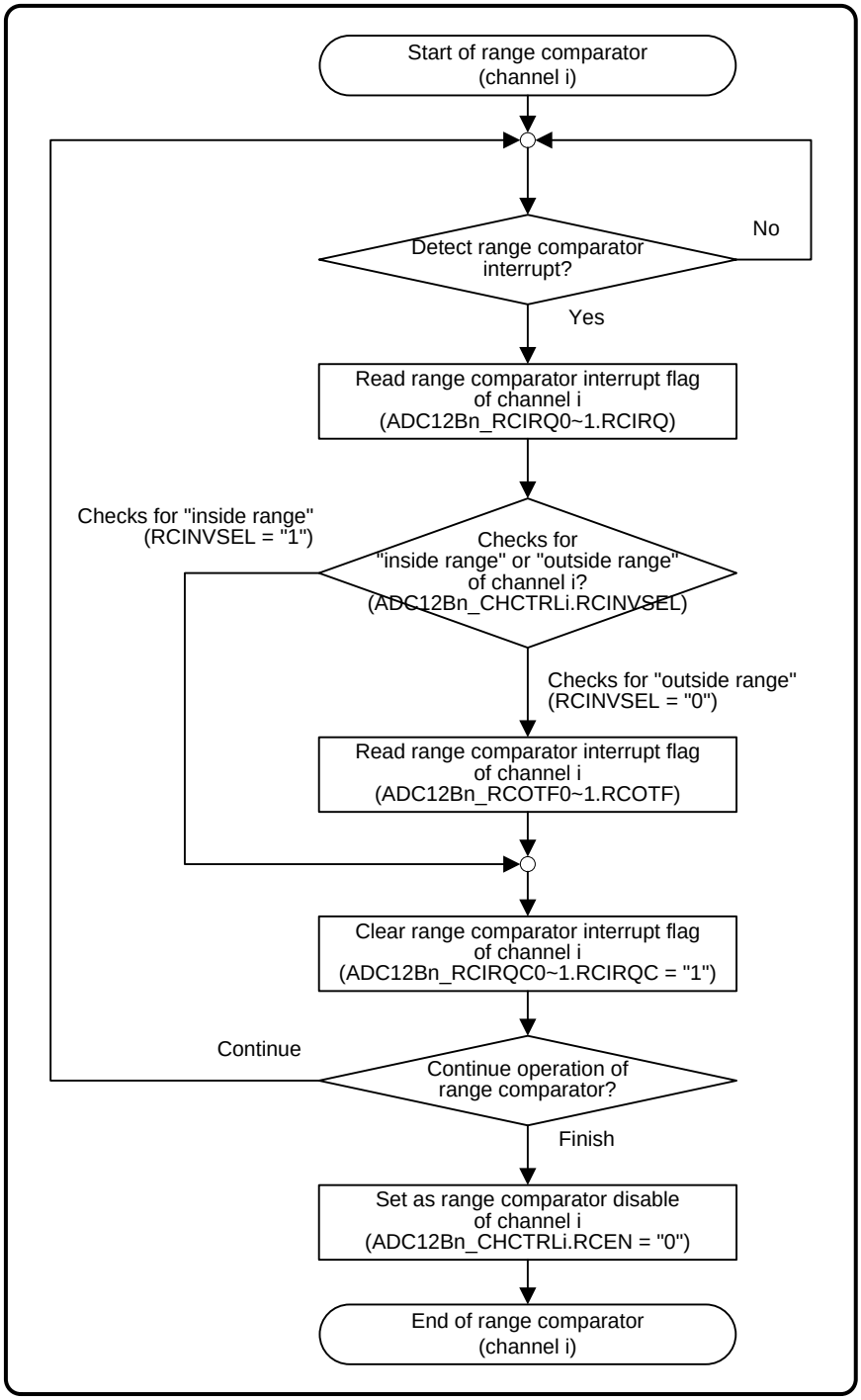
Figure 15-23 Group Conversion



4.7. Range Comparator

Figure 15-24 shows the example of range comparator operation.

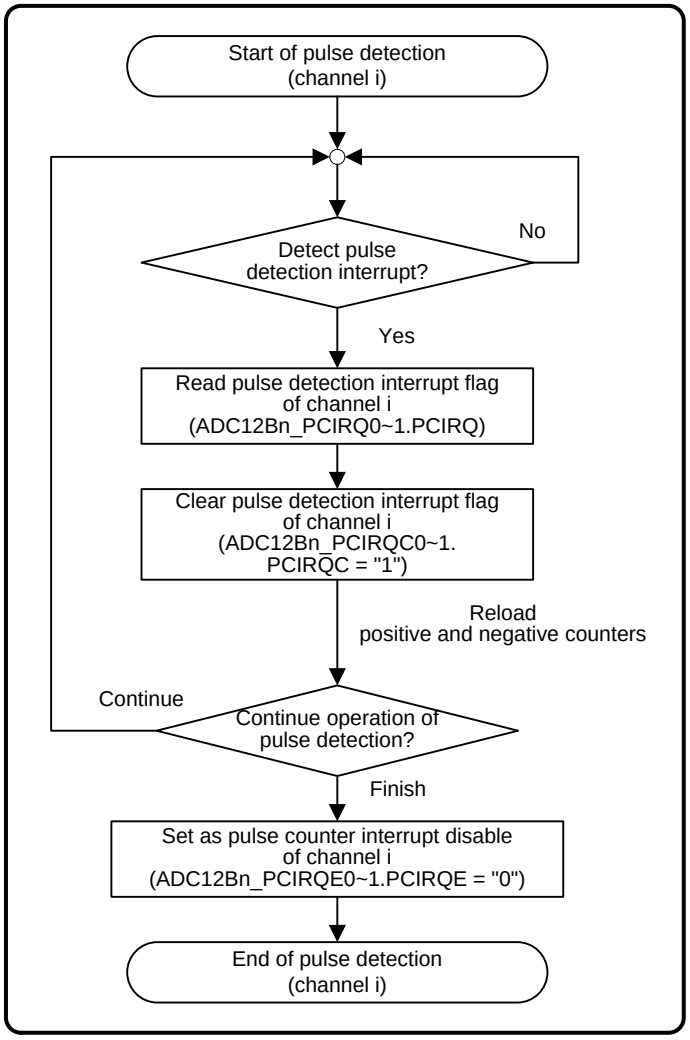
Figure 15-24 Range Comparator (Channel i)



4.8. Pulse Detection

Figure 15-25 shows the example of pulse detection operation.

Figure 15-25 Pulse Detection (Channel i)



5. Registers

The A/D converter contains registers to configure the operation of A/D conversion and to store the converted values. It also contains registers to configure the range comparators and registers to control and store the status of the pulse detection function. This section describes the registers of the A/D Converter in details.

The suffix "n" in the register name indicates that the register is an instance "n" of the module.

Registers of A/D Converter

Logical channel related registers are:

- A/D Channel Control Registers (ADC12Bn_CHCTRL0 to 63)
- A/D Channel Status Registers (ADC12Bn_CHSTAT0 to 63)
- A/D Conversion Data Registers (ADC12Bn_CD0 to 63)
- Pulse Counter Control Registers (ADC12Bn_PCCTRL0 to 63)
- A/D Conversion Done Interrupt Flags (ADC12Bn_CDONEIRQ0 to 1)
- A/D Conversion Done Interrupt Enable Registers (ADC12Bn_CDONEIRQE0 to 1)
- A/D Conversion Done Interrupt Clear Registers (ADC12Bn_CDONEIRQC0 to 1)
- Group Interrupted Interrupt Flags (ADC12Bn_GRPIRQ0 to 1)
- Group Interrupted Interrupt Enable Registers (ADC12Bn_GRPIRQE0 to 1)
- Group Interrupted Interrupt Clear Registers (ADC12Bn_GRPIRQC0 to 1)
- Range Comparator Interrupt Flags (ADC12Bn_RCIRQ0 to 1)
- Range Comparator Interrupt Enable Registers (ADC12Bn_RCIRQE0 to 1)
- Range Comparator Interrupt Clear Registers (ADC12Bn_RCIRQC0 to 1)
- Pulse Counter Interrupt Flags (ADC12Bn_PCIRQ0 to 1)
- Pulse Counter Interrupt Enable Registers (ADC12Bn_PCIRQE0 to 1)
- Pulse Counter Interrupt Clear Registers (ADC12Bn_PCIRQC0 to 1)
- A/D Channel Trigger Status Flags (ADC12Bn_TRGST0 to 1)
- A/D Channel Trigger Status Clear Registers (ADC12Bn_TRGCL0 to 1)
- A/D Channel Trigger Overrun Flags (ADC12Bn_TRGOR0 to 1)
- A/D Channel Trigger Overrun Clear Registers (ADC12Bn_TRGORC0 to 1)
- Range Comparator Over Threshold Flags (ADC12Bn_RCOTF0 to 1)

Global A/D Converter registers are:

- Conversion Done DMA Select Registers (ADC12Bn_CDDS0 to 3)
- A/D Converter Resumption Time Setting Register (ADC12Bn_RT)
- A/D Converter Comparison Time Setting Register (ADC12Bn_CT)
- A/D Converter Sampling Time Setting Registers (ADC12Bn_ST0 to 3)
- A/D Converter Offset Compensation Setting Register (ADC12Bn_OCV)
- A/D Converter Gain Compensation Setting Register (ADC12Bn_GCV)
- A/D Converter Global Control Register (ADC12Bn_CTRL)
- A/D Converter Global Status Register (ADC12Bn_STAT)
- Range Comparator Upper Threshold Registers (ADC12Bn_RCOH0 to 7)
- Range Comparator Lower Threshold Registers (ADC12Bn_RCOL0 to 7)
- Full Range Comparator Upper Threshold Registers (ADC12Bn_FRCOH0 to 7)
- Full Range Comparator Lower Threshold Registers (ADC12Bn_FRCOL0 to 7)

Multiple conversion logical channel related registers are:

- A/D Multiple Conversion Channel Control Registers (ADC12Bn_MCCTRL0 to 3)
- A/D Multiple Conversion Channel Status Registers (ADC12Bn_MCSTAT0 to 3)

5.1. A/D Channel Control Registers (ADC12Bn_CHCTRL0 to 63)

The A/D Channel Control Registers configure the logical channel specific settings. ADC12Bn_CHCTRL0 (described here) is dedicated to channel 0, ADC12Bn_CHCTRL63 is dedicated to channel 63. Other registers (ADC12Bn_CHCTRL1,.....ADC12Bn_CHCTRL62) have similar bit fields.

REGISTER_NAME	ADC12Bn_CHCTRLi (i = 0 to 63)
OFFSET	0x0000 + i*4
ACCESS_SIZE	B H W
MULTIPLE	0:63
NUMERIC_TYPE	-
OTHER	-

A/D Channel Control Register (ADC12Bn_CHCTRL0)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved						TRGCL	SWTRG
ACCESS_TYPE	R0,W0						R0,W	R0,W
PROT_TYPE								
INITIAL_VALUE	000000						0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	RCEN	RCINVSEL	Reserved	RCSEL[2]	RCSEL[1]	RCSEL[0]	SMTIME[1]	SMTIME[0]
ACCESS_TYPE	R/W	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	0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved	DP	RSMRST[1]	RSMRST[0]	CHPRI[3]	CHPRI[2]	CHPRI[1]	CHPRI[0]
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	TRGTYP[1]	TRGTYP[0]	ANIN[5]	ANIN[4]	ANIN[3]	ANIN[2]	ANIN[1]	ANIN[0]
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

When writing, always write "0".

When reading, "0" is always read.

[bit25] TRGCL : Trigger Status Clear bit

Bit	Description
0	No effect.
1	Clears corresponding trigger status bits ADC12Bn_CHSTAT0.TRGST and ADC12Bn_TRGST0.TRGST0.

Reading this bit always returns "0".
This bit is identical to the corresponding bit ADC12Bn_TRGCL0.TRGCL0.

[bit24] SWTRG : Software trigger bit

Bit	Description
0	No effect.
1	Sets corresponding trigger status bits ADC12Bn_CHSTAT0.TRGST and ADC12Bn_TRGST0.TRGST0. When trigger status bits are cleared and set at the same time, clearing has priority. When TRGTYP bits are set to "10" or "11", writing SWTRG bit to "1" has no effect.

Reading this bit always returns "0".

Note:

- *Do not write "1" to this bit and reconfigure TRGTYP with the same access.*

[bit23] RCEN : Range Comparator Enable bit

Bit	Description
0	Range comparator disabled.
1	Range comparator enabled.

[bit22] RCINVSEL : Range Comparator Inverted Range Selection bit

Bit	Description
0	The comparison checks for "outside range", i.e the over threshold and interrupt flags (ADC12Bn_RCOTF0.RCOTF0, ADC12Bn_CHSTAT0.RCOTF, ADC12Bn_RCIRQ0.RCIRQ0 and ADC12Bn_CHSTAT0.RCIRQ) are set when the ADC result is above the upper threshold OR below the lower threshold. That is called "outside range".
1	The comparison checks for "inside range" i.e., the interrupt flags (ADC12Bn_CHSTAT0.RCIRQ and ADC12Bn_RCIRQ0.RCIRQ0) are set when the ADC result is below or equal the upper threshold AND above or equal the lower threshold. That is called "inside range" mode.

[bit21] Reserved : Reserved bit

When writing, always write "0".

When reading, "0" is always read.

[bit20:18] RCSEL[2:0] : Range Comparator Select bit

RCSEL[2:0]	Description
000	Select range comparator 0, defined by ADC12Bn_RCOH0 and ADC12Bn_RCOL0 registers.
...	...
111	Select range comparator 7, defined by ADC12Bn_RCOH7 and ADC12Bn_RCOL7 registers.

[bit17:16] SMTIME[1:0] : Sampling Time Select bits

SMTIME[1:0]	Description
00	ADC12Bn_ST0 register value is selected as channel sampling time.
01	ADC12Bn_ST1 register value is selected as channel sampling time.
10	ADC12Bn_ST2 register value is selected as channel sampling time.
11	ADC12Bn_ST3 register value is selected as channel sampling time.

[bit15] Reserved : Reserved bit

When writing, always write "0".

When reading, "0" is always read.

[bit14] DP : Data protection Enable bit

Bit	Description
0	Data protection function disabled.
1	Data protection function enabled.

[bit13:12] RSMRST[1:0] : Resume/Restart/Stop Select bits

RSMRST[1:0]	Description
00	Stop the group processing until next group start channel conversion request is issued.
01	Resume. If the group is interrupted just before the conversion of this channel is started, resume the group processing with this channel.
10	Restart. After the group is interrupted, restart with the start channel or the last converted channel configured as "resume" channel.
11	Reserved.

RSMRST[1:0] are not allowed to update during A/D conversion operation (ADC12Bn_TRGST0 to 1.TRGST and ADC12Bn_CHSTAT0 to 63.TRGST="1").

[bit11:8] CHPRI[3:0] : Logical channel priority

CHPRI[3:0]	Description
0000	Highest priority.
...	...
1111	Lowest priority.

CHPRI[3:0] are not allowed to update during A/D conversion operation (ADC12Bn_TRGST0 to 1.TRGST and ADC12Bn_CHSTAT0 to 63.TRGST="1").

[bit7:6] TRGTYP[1:0] : Trigger Type bits

TRGTYP[1:0]	Description
00	Software trigger only.
01	Software or hardware trigger.
10	Trigger by conversion completion and updating of conversion data register ADC12Bn_CD of the preceding channel.
11	Idle trigger, the channel trigger status is set if there is no channel having trigger status flag set and inactive data protection function.

TRGTYP[1:0] are not allowed to update during A/D conversion operation (ADC12Bn_TRGST0 to 1.TRGST and ADC12Bn_CHSTAT0 to 63.TRGST="1").

[bit5:0] ANIN[5:0] : Analog Input Selection bits

ANIN[5:0]	Description
000000	Analog input AN0 is selected.
...	...
111111	Analog input AN63 is selected.

5.2. A/D Channel Status Registers (ADC12Bn_CHSTAT0 to 63)

These registers store the status information of the corresponding logical channels related to interrupt flags and trigger status. ADC12Bn_CHSTAT0 (described here) is dedicated to channel 0, ADC12Bn_CHSTAT63 is dedicated to channel 63. Other registers (ADC12Bn_CHSTAT1,.....ADC12Bn_CHSTAT62) have similar bit fields.

REGISTER_NAME	ADC12Bn_CHSTATi (i = 0 to 63)
OFFSET	0x0100 + i*2
ACCESS_SIZE	B H W
MULTIPLE	0:63
NUMERIC_TYPE	-
OTHER	-

A/D Channel Status Register (ADC12Bn_CHSTAT0)

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

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved		RCOTF	PCIRQ	RCIRQ	GRPIRQ	CDONEIRQ	TRGST
ACCESS_TYPE	RX,WX		R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE								
INITIAL_VALUE	00		0	0	0	0	0	0

[bit15:6] Reserved : Reserved bits

Reading this bit returns an undefined.

Writing data to these bits has no effect on the operation.

Writing data to these bits has no effect on the operation.[bit5] RCOTF : Range Comparator Over Threshold flag

Bit	Description
0	The conversion result is less than or equal to the upper threshold.
1	The conversion result is above the upper threshold.

This bit is identical to the corresponding bit in the ADC12Bn_RCOTF0 register. For more details, see ADC12Bn_RCOTF0 to 1 register description.

[bit4] PCIRQ : Pulse Counter Interrupt flag

Bit	Description
0	Not detected.
1	Pulse counter interrupt detected.

This bit is identical to the corresponding bit in the ADC12Bn_PCIRQ0 register. For more details, see ADC12Bn_PCIRQ0 to 1 register description.

[bit3] RCIRQ : Range Comparator Interrupt flag

Bit	Description
0	Not detected.
1	Range comparator interrupt detected.

This bit is identical to the corresponding bit in the ADC12Bn_RCIRQ0 register. For more details, see ADC12Bn_RCIRQ0 to 1 register description

[bit2] GRPIRQ : Group Interrupted Interrupt flag

Bit	Description
0	Not detected.
1	Group interrupted interrupt detected.

This bit is identical to the corresponding bit in the ADC12Bn_GRPIRQ0 register. For more details, see ADC12Bn_GRPIRQ0 to 1 register description.

[bit1] CDONEIRQ : Conversion done Interrupt flag

Bit	Description
0	Not detected
1	Conversion done Interrupt detected.

This bit is identical to the corresponding bit in the ADC12Bn_CDONEIRQ0 register. For more details, see ADC12Bn_CDONEIRQ0 to 1 register description.

[bit0] TRGST : Trigger Status flag

Bit	Description
0	Conversion request not detected.
1	Conversion request detected.

This bit is identical to the corresponding bit in the ADC12Bn_TRGST0 register. For more details, see ADC12Bn_TRGST0 to 1 register description.

5.3. A/D Conversion Data Registers (ADC12Bn_CD0 to 63)

There are 64 A/D conversion data registers, one per logical channel. The registers are written by hardware at the end of conversion if the trigger status is still set. Registers ADC12B_CD0 to 3 are dedicated to multiple conversion channels so their width is 16 bits. Width of registers ADC12B_CD4 to 63 is 12 bits.

REGISTER_NAME	ADC12Bn_CDi (i = 0 to 63)
OFFSET	0x0180 + i*2
ACCESS_SIZE	B H W
MULTIPLE	0:63
NUMERIC_TYPE	-
OTHER	-

A/D Conversion Data Result Registers (ADC12Bn_CD0 to 3)

Here is the register ADC12Bn_CD0 described, the registers ADC12Bn_CD1 to 3 have similar bit fields.

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	D[15]	D[14]	D[13]	D[12]	D[11]	D[10]	D[9]	D[8]
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE								
INITIAL_VALUE	X	X	X	X	X	X	X	X

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	D[7]	D[6]	D[5]	D[4]	D[3]	D[2]	D[1]	D[0]
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE								
INITIAL_VALUE	X	X	X	X	X	X	X	X

[bit15:0] D[15:0] : A/D Conversion Data bits

These bits store the conversion data.

The register is updated at the end of the A/D conversion only in the case the corresponding trigger status (ADC12Bn_TRGST0.TRGST0 and ADC12Bn_CHSTAT0.TRGST bits) is still "1".

The logical channel can be configured as multiple conversion channel, if ADC12Bn_MCCTRL0.CNVNUM is greater than 0. In that case the result of conversions is accumulated until the number of conversions reaches ADC12Bn_MCCTRL0.CNVNUM (the result of the first conversion is always directly stored and the following conversion results are added on current register value). Accordingly, if 12-bit conversion resolution is selected the conversion data can become 16-bit wide.

If ADC12Bn_MCCTRL0.CNVNUM is equal 0 (only one consecutive conversion is requested), the conversion data are provided in range of:

- bit[7:0] for 8-bit conversion resolution (bits 15 to 8 are "0"),
- bit[9:0] for 10-bit conversion resolution (bits 15 to 10 are "0"),
- bit[11:0] for 12-bit conversion resolution (bits 15 to 12 are "0").

A/D Conversion Data Registers (ADC12Bn_CD4 to 63)

Here is the register ADC12Bn_CD4 described, the registers ADC12Bn_CD5 to 63 have similar bit fields.

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved				D[11]	D[10]	D[9]	D[8]
ACCESS_TYPE	RX,WX				R,WX	R,WX	R,WX	R,WX
PROT_TYPE								
INITIAL_VALUE	0000				X	X	X	X

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	D[7]	D[6]	D[5]	D[4]	D[3]	D[2]	D[1]	D[0]
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R.WX
PROT_TYPE								
INITIAL_VALUE	X	X	X	X	X	X	X	X

[bit15:12] Reserved : Reserved bits

Reading this bit returns an undefined.

Writing data to these bits has no effect on the operation.

[bit11:0] D[11:0] : A/D Conversion Data bits

These bits store the conversion data, provided in range of:

- bit[7:0] for 8-bit conversion resolution (bits 11 to 8 are "0"),
- bit[9:0] for 10-bit conversion resolution (bits 11 to 10 are "0") and
- bit[11:0] for 12-bit conversion resolution.

The register is updated at the end of the A/D conversion only in the case the corresponding trigger status (ADC12Bn_TRGST0.TRGST4 and ADC12Bn_CHSTAT4.TRGST bits) is still "1".

5.4. Pulse Counter Control Registers (ADC12Bn_PCCTRL0 to 63)

These registers hold the reload and current values of positive and negative counters of the pulse detection function for the corresponding logical channel. The positive counters count down the positive events of the range comparator and negative counters count down the negative events of the range comparator. ADC12Bn_PCCTRL0 (described here) is dedicated to channel 0, ADC12Bn_PCCTRL63 is dedicated to channel 63. Other registers (ADC12Bn_PCCTRL1,.....ADC12Bn_PCCTRL62) have similar bit fields.

REGISTER_NAME	ADC12Bn_PCCTRLi (i = 0 to 63)
OFFSET	0x0200 + i*4
ACCESS_SIZE	B H W
MULTIPLE	0:63
NUMERIC_TYPE	-
OTHER	-

Pulse Counter Control Register (ADC12Bn_PCCTRL0)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved			PCTNCT[4]	PCTNCT[3]	PCTNCT[2]	PCTNCT[1]	PCTNCT[0]
ACCESS_TYPE	RX,WX			R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE								
INITIAL_VALUE	000			0	0	0	0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved			PCTNRL[4]	PCTNRL[3]	PCTNRL[2]	PCTNRL[1]	PCTNRL[0]
ACCESS_TYPE	R0,W0			R/W	R/W	R/W	R/W	R/W
PROT_TYPE								
INITIAL_VALUE	000			0	0	0	1	0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	PCTPCT[7]	PCTPCT[6]	PCTPCT[5]	PCTPCT[4]	PCTPCT[3]	PCTPCT[2]	PCTPCT[1]	PCTPCT[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

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	PCTPRL[7]	PCTPRL[6]	PCTPRL[5]	PCTPRL[4]	PCTPRL[3]	PCTPRL[2]	PCTPRL[1]	PCTPRL[0]
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	1	0

[bit31:29] Reserved : Reserved bits
Reading this bit returns an undefined.

Writing data to these bits has no effect on the operation.

[bit28:24] PCTNCT[4:0] : Pulse Negative Counter Register
This register reflects the current counter value of the pulse detection negative counter. Reload value is determined by PCTNRL.

The negative counter is reloaded under one of the following conditions:

- Writing "1" to the corresponding ADC12Bn_PCIRQC0.PCIRQC bit.
- Any positive event from the appropriate range comparator.

Therefore, in order to reload negative counter immediately after set to PCTNRL[4:0], the corresponding ADC12Bn_PCIRQC0 to 1.PCIRQC bit should be written "1".

[bit23:21] Reserved : Reserved bits
When writing, always write "0".

When reading, "0" is always read.

[bit20:16] PCTNRL[4:0] : Pulse Negative Counter Reload Register
These register bits hold the reload value of the negative counter PCTNCT used for counting negative events of the range comparator. Do not set this bit field to "00000".

The negative counter is reloaded with the value from this register when "1" is written to the corresponding ADC12Bn_PCIRQC0 to 1.PCIRQC bit or on any positive event from the appropriate range comparator.

For further explanation of negative events and operation of pulse detection function refer to section "Pulse Detection Function" in chapter "3 Operation of A/D "

[bit15:8] PCTPCT[7:0] : Pulse Positive Counter Register
This register reflects the current counter value of the pulse detection positive counter. Reload value is determined by PCTPRL.

The positive counter is reloaded under one of the following conditions:

- Writing "1" to the corresponding ADC12Bn_PCIRQC0 to 1.PCIRQ bit.
- Expiration of the corresponding negative counter (PCTNCT).

Therefore, in order to reload positive counter immediately after set to PCTPRL[7:0], the corresponding ADC12Bn_PCIRQC0 to 1.PCIRQC bit should be written "1".

.

[bit7:0] PCTPRL[7:0] : Pulse Positive Counter Reload Register
These register bits hold the reload value of the positive counter PCTPCT used for counting positive events of the range comparator. Do not set this bit field to "00000000".

The positive counter is reloaded with the value from this register when "1" is written to the corresponding ADC12Bn_PCIRQC0 to 1.PCIRQC bit or on expiration of the corresponding negative counter (PCTNCT).

For further explanation of negative events and operation of pulse detection function refer to section "Pulse Detection Function" in chapter "3 Operation of A/D "

5.5. A/D Conversion Done Interrupt Flag Registers (ADC12Bn_CDONEIRQ0 to 1)

A/D Conversion Done Interrupt Flag Registers ADC12Bn_CDONEIRQ0 to 1 contain the status of conversion done interrupt flags for all 64 logical channels.

REGISTER_NAME	ADC12Bn_CDONEIRQi (i = 0 to 1)
OFFSET	0x300 + i*4
ACCESS_SIZE	B H W
MULTIPLE	0:1
NUMERIC_TYPE	-
OTHER	-

A/D Conversion Done Interrupt Flag Register (ADC12Bn_CDONEIRQ0)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	CDONEIRQ31	CDONEIRQ30	CDONEIRQ29	CDONEIRQ28	CDONEIRQ27	CDONEIRQ26	CDONEIRQ25	CDONEIRQ24
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	CDONEIRQ23	CDONEIRQ22	CDONEIRQ21	CDONEIRQ20	CDONEIRQ19	CDONEIRQ18	CDONEIRQ17	CDONEIRQ16
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	CDONEIRQ15	CDONEIRQ14	CDONEIRQ13	CDONEIRQ12	CDONEIRQ11	CDONEIRQ10	CDONEIRQ9	CDONEIRQ8
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	CDONEIRQ7	CDONEIRQ6	CDONEIRQ5	CDONEIRQ4	CDONEIRQ3	CDONEIRQ2	CDONEIRQ1	CDONEIRQ0
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] CDONEIRQ31 to 0 : Conversion Done Interrupt flags

Bit	Description
0	Conversion done interrupt request not detected.
1	Conversion done interrupt request detected.

This bit is set when conversion data is stored in corresponding conversion data register ADC12Bn_CD0 to 31 and corresponding trigger status (ADC12Bn_TRGST0.TRGST and ADC12Bn_CHSTAT0 to 31.TRGST bits) is still "1".

In case of multiple conversion channels this bit is set when the last conversion is done, final conversion result is accumulated and corresponding trigger status (ADC12Bn_TRGST0.TRGST and ADC12Bn_CHSTAT0 to 31.TRGST bits) is still "1".

This bit is cleared by writing "1" to the corresponding ADC12Bn_CDONEIRQC0 bits or by reading the corresponding conversion data register ADC12Bn_CD0 to 31 (except by the debug master, DAP). All ADC12Bn_CD0 to 31 read access types (8/16/32-bit) clear the flag.

If this bit is set and cleared at the same time, clearing has higher priority.

This bit is identical to the CDONEIRQ bit in the corresponding ADC12Bn_CHSTAT0 to 31 registers.

A/D Conversion Done Interrupt Flag Register (ADC12Bn_CDONEIRQ1)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	CDONEIRQ63	CDONEIRQ62	CDONEIRQ61	CDONEIRQ60	CDONEIRQ59	CDONEIRQ58	CDONEIRQ57	CDONEIRQ56
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	CDONEIRQ55	CDONEIRQ54	CDONEIRQ53	CDONEIRQ52	CDONEIRQ51	CDONEIRQ50	CDONEIRQ49	CDONEIRQ48
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	CDONEIRQ47	CDONEIRQ46	CDONEIRQ45	CDONEIRQ44	CDONEIRQ43	CDONEIRQ42	CDONEIRQ41	CDONEIRQ40
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	CDONEIRQ39	CDONEIRQ38	CDONEIRQ37	CDONEIRQ36	CDONEIRQ35	CDONEIRQ34	CDONEIRQ33	CDONEIRQ32
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] CDONEIRQ63 to 32 : Conversion Done Interrupt flags

Bit	Description
0	Conversion done interrupt request not detected.
1	Conversion done interrupt request detected.

This bit is set when conversion data is stored in corresponding conversion data register ADC12Bn_CD32 to 63 and corresponding trigger status (ADC12Bn_TRGST1.TRGST and ADC12Bn_CHSTAT32 to 63.TRGST bits) is still "1".

This bit is cleared by writing "1" to the corresponding ADC12Bn_CDONEIRQC1 bits or by reading the corresponding conversion data register ADC12Bn_CD32 to 63 (except by the debug master, DAP). All ADC12Bn_CD32 to 63 read access types (8/16/32-bit) clear the flag.

If this bit is set and cleared at the same time, clearing has higher priority.

This bit is identical to the CDONEIRQ bit in the corresponding ADC12Bn_CHSTAT32 to 63 registers.

5.6. A/D Conversion Done Interrupt Enable Registers (ADC12Bn_CDONEIRQE0 to 1)

These registers contain enable bits for all 64 logical channels, dedicated to the generation of conversion done interrupt.

REGISTER_NAME	ADC12Bn_CDONEIRQEi (i = 0 to 1)
OFFSET	0x0308 + i*4
ACCESS_SIZE	B H W
MULTIPLE	0:1
NUMERIC_TYPE	-
OTHER	-

A/D Conversion Done Interrupt Enable Register (ADC12Bn_CDONEIRQE0)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	CDONEIRQE31	CDONEIRQE30	CDONEIRQE29	CDONEIRQE28	CDONEIRQE27	CDONEIRQE26	CDONEIRQE25	CDONEIRQE24
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	CDONEIRQE23	CDONEIRQE22	CDONEIRQE21	CDONEIRQE20	CDONEIRQE19	CDONEIRQE18	CDONEIRQE17	CDONEIRQE16
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	CDONEIRQE15	CDONEIRQE14	CDONEIRQE13	CDONEIRQE12	CDONEIRQE11	CDONEIRQE10	CDONEIRQE9	CDONEIRQE8
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	CDONEIRQE7	CDONEIRQE6	CDONEIRQE5	CDONEIRQE4	CDONEIRQE3	CDONEIRQE2	CDONEIRQE1	CDONEIRQE0
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] CDONEIRQE31 to 0 : Conversion Done Interrupt Enable bits

Bit	Description
0	Conversion done interrupt disabled
1	Conversion done interrupt enabled.

Conversion done interrupt is issued when the bit is "1" and the corresponding interrupt flags ADC12Bn_CDONEIRQ0.CDONEIRQ31 to 0 and ADC12Bn_CHSTAT0 to 31.CDONEIRQ are set to "1".

A/D Conversion Done Interrupt Enable Register (ADC12Bn_CDONEIRQE1)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	CDONEIRQE63	CDONEIRQE62	CDONEIRQE61	CDONEIRQE60	CDONEIRQE59	CDONEIRQE58	CDONEIRQE57	CDONEIRQE56
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	CDONEIRQE55	CDONEIRQE54	CDONEIRQE53	CDONEIRQE52	CDONEIRQE51	CDONEIRQE50	CDONEIRQE49	CDONEIRQE48
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	CDONEIRQE47	CDONEIRQE46	CDONEIRQE45	CDONEIRQE44	CDONEIRQE43	CDONEIRQE42	CDONEIRQE41	CDONEIRQE40
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	CDONEIRQE39	CDONEIRQE38	CDONEIRQE37	CDONEIRQE36	CDONEIRQE35	CDONEIRQE34	CDONEIRQE33	CDONEIRQE32
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] CDONEIRQE63 to 32 : Conversion Done Interrupt Enable bits

Bit	Description
0	Conversion done interrupt disabled
1	Conversion done interrupt enabled.

Conversion done interrupt is issued when the bit is "1" and the corresponding interrupt flags ADC12Bn_CDONEIRQ1.CDONEIRQ63 to 32 and ADC12Bn_CHSTAT32 to 63.CDONEIRQ are set to "1".

5.7. A/D Conversion Done Interrupt Clear Registers (ADC12Bn_CDONEIRQC0 to 1)

These registers contain bits for clearing corresponding conversion done interrupt flags in the ADC12Bn_CDONEIRQ0 to 1 registers. The 64 bits are assigned to 64 logical channels.

REGISTER_NAME	ADC12Bn_CDONEIRQCi (i = 0 to 1)
OFFSET	0x0310 + i*4
ACCESS_SIZE	B H W
MULTIPLE	0:1
NUMERIC_TYPE	-
OTHER	-

A/D Conversion Done Interrupt Clear Register (ADC12Bn_CDONEIRQC0)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	CDONEIRQC31	CDONEIRQC30	CDONEIRQC29	CDONEIRQC28	CDONEIRQC27	CDONEIRQC26	CDONEIRQC25	CDONEIRQC24
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	CDONEIRQC23	CDONEIRQC22	CDONEIRQC21	CDONEIRQC20	CDONEIRQC19	CDONEIRQC18	CDONEIRQC17	CDONEIRQC16
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	CDONEIRQC15	CDONEIRQC14	CDONEIRQC13	CDONEIRQC12	CDONEIRQC11	CDONEIRQC10	CDONEIRQC9	CDONEIRQC8
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	CDONEIRQC7	CDONEIRQC6	CDONEIRQC5	CDONEIRQC4	CDONEIRQC3	CDONEIRQC2	CDONEIRQC1	CDONEIRQC0
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] CDONEIRQC31 to 0 : Conversion Done Interrupt Clear bits

Bit	Description
0	No effect.
1	Conversion Done Interrupt cleared.

When this bit is set to "1", the corresponding bit in the ADC12Bn_CDONEIRQ0 register and CDONEIRQ bit in the corresponding ADC12Bn_CHSTAT0 to 31 register are cleared.

A/D Conversion Done Interrupt Clear Register (ADC12Bn_CDONEIRQC1)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	CDONEIRQC63	CDONEIRQC62	CDONEIRQC61	CDONEIRQC60	CDONEIRQC59	CDONEIRQC58	CDONEIRQC57	CDONEIRQC56
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	CDONEIRQC55	CDONEIRQC54	CDONEIRQC53	CDONEIRQC52	CDONEIRQC51	CDONEIRQC50	CDONEIRQC49	CDONEIRQC48
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	CDONEIRQC47	CDONEIRQC46	CDONEIRQC45	CDONEIRQC44	CDONEIRQC43	CDONEIRQC42	CDONEIRQC41	CDONEIRQC40
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	CDONEIRQC39	CDONEIRQC38	CDONEIRQC37	CDONEIRQC36	CDONEIRQC35	CDONEIRQC34	CDONEIRQC33	CDONEIRQC32
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] CDONEIRQC63 to 32 : Conversion Done Interrupt Clear bits

Bit	Description
0	No effect.
1	Conversion Done Interrupt cleared.

When this bit is set to "1", the corresponding bit in the ADC12Bn_CDONEIRQ1 register and CDONEIRQ bit in the corresponding ADC12Bn_CHSTAT32 to 63 register are cleared.

5.8. Group Interrupted Interrupt Flag Registers (ADC12Bn_GRPIRQ0 to 1)

These registers contain the status of group interrupted interrupt flags for all 64 logical channels.

REGISTER_NAME	ADC12Bn_GRQIRQi (i = 0 to 1)
OFFSET	0x0318 + i*4
ACCESS_SIZE	B H W
MULTIPLE	0:1
NUMERIC_TYPE	-
OTHER	-

Group Interrupted Interrupt Flag Register (ADC12Bn_GRPIRQ0)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	GRPIRQ31	GRPIRQ30	GRPIRQ29	GRPIRQ28	GRPIRQ27	GRPIRQ26	GRPIRQ25	GRPIRQ24
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	GRPIRQ23	GRPIRQ22	GRPIRQ21	GRPIRQ20	GRPIRQ19	GRPIRQ18	GRPIRQ17	GRPIRQ16
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	GRPIRQ15	GRPIRQ14	GRPIRQ13	GRPIRQ12	GRPIRQ11	GRPIRQ10	GRPIRQ9	GRPIRQ8
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	GRPIRQ7	GRPIRQ6	GRPIRQ5	GRPIRQ4	GRPIRQ3	GRPIRQ2	GRPIRQ1	GRPIRQ0
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] GRPIRQ31 to 0

Group Interrupted Interrupt flags for the case the group gets interrupted and stopped before finishing.

Bit	Description
0	Group interrupted interrupt not detected.
1	Group interrupted interrupt detected.

The bit is set to "1" if following conditions are fulfilled:

- The corresponding trigger status flags of the channel (ADC12Bn_CHSTAT0 to 31.TRGST and ADC12Bn_TRGST0.TRGST31 to 0) are set to "1"
- The channel is not first in the group i.e. the trigger type bits TRGTYP are set to "10" in the corresponding ADC12Bn_CHCTRL0 to 31 register
- The channel did not win arbitration for the next conversion, i.e. there was a channel with active trigger status and higher priority.

For multiple conversion logical channels, this bit is set to "1" also in the case the multiple conversions are started and they are interrupted before the last conversion is performed.

This bit is cleared by writing "1" to the corresponding ADC12Bn_GRPIRQC0 bits.

This bit is identical to the GRPIRQ bit in the corresponding ADC12Bn_CHSTAT0 to 31 registers

Group Interrupted Interrupt Flag Register (ADC12Bn_GRPIRQ1)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	GRPIRQ63	GRPIRQ62	GRPIRQ61	GRPIRQ60	GRPIRQ59	GRPIRQ58	GRPIRQ57	GRPIRQ56
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	GRPIRQ55	GRPIRQ54	GRPIRQ53	GRPIRQ52	GRPIRQ51	GRPIRQ50	GRPIRQ49	GRPIRQ48
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	GRPIRQ47	GRPIRQ46	GRPIRQ45	GRPIRQ44	GRPIRQ43	GRPIRQ42	GRPIRQ41	GRPIRQ40
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	GRPIRQ39	GRPIRQ38	GRPIRQ37	GRPIRQ36	GRPIRQ35	GRPIRQ34	GRPIRQ33	GRPIRQ32
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] GRPIRQ63 to 32

Group Interrupted Interrupt flags for the case the group gets interrupted and stopped before finishing.

Bit	Description
0	Group interrupted interrupt not detected.
1	Group interrupted interrupt detected.

The bit is set to "1" if following conditions are fulfilled:

- The corresponding trigger status flags of the channel (ADC12Bn_CHSTAT32 to 63.TRGST and ADC12Bn_TRGST1.TRGST63 to 32) are set to "1"
- The channel is not first in the group i.e. the trigger type bits TRGTYP are set to "10" in the corresponding ADC12Bn_CHCTRL32 to 63 register
- The channel did not win arbitration for the next conversion, i.e. there was a channel with active trigger status and higher priority.

For multiple conversion logical channels, this bit is set to "1" also in the case the multiple conversions are started and they are interrupted before the last conversion is performed.

This bit is cleared by writing "1" to the corresponding ADC12Bn_GRPIRQC1 bits.

This bit is identical to the GRPIRQ bit in the corresponding ADC12Bn_CHSTAT32 to 63 registers

5.9. Group Interrupted Interrupt Enable Registers (ADC12Bn_GRPIRQE0 to 1)

These registers contain enable bits for all 64 logical channels, dedicated to the generation of group interrupted interrupt.

REGISTER_NAME	ADC12Bn_GRPIRQEi (i = 0 to 1)
OFFSET	0x0320 + i*4
ACCESS_SIZE	B H W
MULTIPLE	0:1
NUMERIC_TYPE	-
OTHER	-

Group Interrupted Interrupt Enable Register (ADC12Bn_GRPIRQE0)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	GRPIRQE31	GRPIRQE30	GRPIRQE29	GRPIRQE28	GRPIRQE27	GRPIRQE26	GRPIRQE25	GRPIRQE24
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	GRPIRQE23	GRPIRQE22	GRPIRQE21	GRPIRQE20	GRPIRQE19	GRPIRQE18	GRPIRQE17	GRPIRQE16
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	GRPIRQE15	GRPIRQE14	GRPIRQE13	GRPIRQE12	GRPIRQE11	GRPIRQE10	GRPIRQE9	GRPIRQE8
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	GRPIRQE7	GRPIRQE6	GRPIRQE5	GRPIRQE4	GRPIRQE3	GRPIRQE2	GRPIRQE1	GRPIRQE0
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] GRPIRQE31 to 0 : Group Interrupted Interrupt Enable bits

Bit	Description
0	Group interrupted interrupt disabled.
1	Group interrupted interrupt enabled.

Group interrupted interrupt is issued when this bit is "1" and the corresponding interrupt flags (ADC12Bn_GRPIRQ0.GRPIRQ31 to 0 and ADC12Bn_CHSTAT0 to 31.GRPIRQ) are set to "1".

Group Interrupted Interrupt Enable Register (ADC12Bn_GRPIRQE1)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	GRPIRQE63	GRPIRQE62	GRPIRQE61	GRPIRQE60	GRPIRQE59	GRPIRQE58	GRPIRQE57	GRPIRQE56
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	GRPIRQE55	GRPIRQE54	GRPIRQE53	GRPIRQE52	GRPIRQE51	GRPIRQE50	GRPIRQE49	GRPIRQE48
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	GRPIRQE47	GRPIRQE46	GRPIRQE45	GRPIRQE44	GRPIRQE43	GRPIRQE42	GRPIRQE41	GRPIRQE40
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	GRPIRQE39	GRPIRQE38	GRPIRQE37	GRPIRQE36	GRPIRQE35	GRPIRQE34	GRPIRQE33	GRPIRQE32
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] GRPIRQE63 to 32 : Group Interrupted Interrupt Enable bits

Bit	Description
0	Group interrupted interrupt disabled.
1	Group interrupted interrupt enabled.

Group interrupted interrupt is issued when this bit is "1" and the corresponding interrupt flags (ADC12Bn_GRPIRQ1.GRPIRQ63 to 32 and ADC12Bn_CHSTAT32 to 63.GRPIRQ) are set to "1".

5.10. Group Interrupted Interrupt Clear Registers (ADC12Bn_GRPIRQC0 to 1)

These registers contain bits for clearing corresponding group interrupted interrupt flags in the ADC12Bn_GRPIRQ0 to 1 registers. The 64 bits are assigned to 64 logical channels.

REGISTER_NAME	ADC12Bn_GRPIRQCi (i = 0 to 1)
OFFSET	0x0328 + i*4
ACCESS_SIZE	B H W
MULTIPLE	0:1
NUMERIC_TYPE	-
OTHER	-

Group interrupted Interrupt Clear Register (ADC12Bn_GRPIRQC0)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	GRPIRQC31	GRPIRQC30	GRPIRQC29	GRPIRQC28	GRPIRQC27	GRPIRQC26	GRPIRQC25	GRPIRQC24
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	GRPIRQC23	GRPIRQC22	GRPIRQC21	GRPIRQC20	GRPIRQC19	GRPIRQC18	GRPIRQC17	GRPIRQC16
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	GRPIRQC15	GRPIRQC14	GRPIRQC13	GRPIRQC12	GRPIRQC11	GRPIRQC10	GRPIRQC9	GRPIRQC8
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	GRPIRQC7	GRPIRQC6	GRPIRQC5	GRPIRQC4	GRPIRQC3	GRPIRQC2	GRPIRQC1	GRPIRQC0
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] GRPIRQC31 to 0 : Group Interrupted Clear bits

Bit	Description
0	No effect.
1	Group interrupted interrupt cleared.

When this bit is set to "1", the corresponding bit in the ADC12Bn_GRPIRQ0 register and GRPIRQ bit in the corresponding ADC12Bn_CHSTAT0 to 31 register are cleared.

Group Interrupted Interrupt Clear Register (ADC12Bn_GRPIRQC1)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	GRPIRQC63	GRPIRQC62	GRPIRQC61	GRPIRQC60	GRPIRQC59	GRPIRQC58	GRPIRQC57	GRPIRQC56
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	GRPIRQC55	GRPIRQC54	GRPIRQC53	GRPIRQC52	GRPIRQC51	GRPIRQC50	GRPIRQC49	GRPIRQC48
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	GRPIRQC47	GRPIRQC46	GRPIRQC45	GRPIRQC44	GRPIRQC43	GRPIRQC42	GRPIRQC41	GRPIRQC40
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	GRPIRQC39	GRPIRQC38	GRPIRQC37	GRPIRQC36	GRPIRQC35	GRPIRQC34	GRPIRQC33	GRPIRQC32
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] GRPIRQC63 to 32 : Group Interrupted Clear bits

Bit	Description
0	No effect.
1	Group interrupted interrupt cleared.

When this bit is set to "1", the corresponding bit in the ADC12Bn_GRPIRQ1 register and GRPIRQ bit in the corresponding ADC12Bn_CHSTAT32 to 63 register are cleared.

5.11. Range Comparator Interrupt Flag Registers (ADC12Bn_RCIRQ0 to 1)

Range Comparator Interrupt Flag Registers ADC12Bn_RCIRQ0 to 1 contain the status of range comparator interrupt flags for all 64 logical channels.

REGISTER_NAME	ADC12Bn_RCIRQi (i = 0 to 1)
OFFSET	0x0330 + i*4
ACCESS_SIZE	B H W
MULTIPLE	0:1
NUMERIC_TYPE	-
OTHER	-

Range Comparator Interrupt Flag Register (ADC12Bn_RCIRQ0)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	RCIRQ31	RCIRQ30	RCIRQ29	RCIRQ28	RCIRQ27	RCIRQ26	RCIRQ25	RCIRQ24
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	RCIRQ23	RCIRQ22	RCIRQ21	RCIRQ20	RCIRQ19	RCIRQ18	RCIRQ17	RCIRQ16
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	RCIRQ15	RCIRQ14	RCIRQ13	RCIRQ12	RCIRQ11	RCIRQ10	RCIRQ9	RCIRQ8
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	RCIRQ7	RCIRQ6	RCIRQ5	RCIRQ4	RCIRQ3	RCIRQ2	RCIRQ1	RCIRQ0
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] RCIRQ31 to 0 : Range Comparator Interrupt flags

Bit	Description
0	Range comparator interrupt not detected.
1	Range comparator interrupt detected.

This flag shows that an outside range or inside range condition has been found on the corresponding logical channel.

This bit is set under the following conditions:

- The range comparison for this channel is enabled ADC12B_CHCTRL0 to 31.RCEN is set
- The conversion of the logical channel is just finished
- An interrupt condition is met (see Table 15-4)

This bit is cleared by writing "1" to the corresponding ADC12B_RCIRQC0.RCIRQC bit.

This bit is identical to the RCIRQ bit in the corresponding ADC12Bn_CHSTAT0 to 31 register.

Range Comparator Interrupt Flag Register (ADC12Bn_RCIRQ1)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	RCIRQ63	RCIRQ62	RCIRQ61	RCIRQ60	RCIRQ59	RCIRQ58	RCIRQ57	RCIRQ56
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	RCIRQ55	RCIRQ54	RCIRQ53	RCIRQ52	RCIRQ51	RCIRQ50	RCIRQ49	RCIRQ48
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	RCIRQ47	RCIRQ46	RCIRQ45	RCIRQ44	RCIRQ43	RCIRQ42	RCIRQ41	RCIRQ40
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	RCIRQ39	RCIRQ38	RCIRQ37	RCIRQ36	RCIRQ35	RCIRQ34	RCIRQ33	RCIRQ32
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] RCIRQ63 to 32 : Range Comparator Interrupt flags

Bit	Description
0	Range comparator interrupt not detected.
1	Range comparator interrupt detected.

This flag shows that an outside range or inside range condition has been found on the corresponding logical channel.

This bit is set under the following conditions:

- The range comparison for this channel is enabled ADC12B_CHCTRL32 to 63.RCEN is set
- The conversion of the logical channel is just finished
- An interrupt condition is met (see Table 15-4)

This bit is cleared by writing "1" to the corresponding ADC12B_RCIRQC1.RCIRQC bit.

This bit is identical to the RCIRQ bit in the corresponding ADC12Bn_CHSTAT32 to 63 register.

Table 15-4 Range Comparator Interrupt Condition

Mode	Inverted Range Selection ADC12Bn_CHCTRL0 to 63.RCINVSEL	Conversion Result Above Upper Threshold	Conversion Result below Lower Threshold	Interrupt Condition
outside range	0	1	x	INT condition: above range, ADC12Bn_RCOTF0 to 1.RCOTF and ADC12Bn_CHSTAT0 to 63.RCOTF are set.
		0	0	-
		0	1	INT condition: below range, ADC12Bn_RCOTF0 to 1.RCOTF and ADC12Bn_CHSTAT0 to 63.RCOTF are cleared.
inside range	1	1	x	-
		0	0	INT condition: inside range
		0	1	-

5.12. Range Comparator Interrupt Enable Registers (ADC12Bn_RCIRQE0 to 1)

These registers contain enable bits for all 64 logical channels, dedicated to the generation of range comparator interrupt.

REGISTER_NAME	ADC12Bn_RCIRQEi (i = 0 to 1)
OFFSET	0x0338 + i*4
ACCESS_SIZE	B H W
MULTIPLE	0:1
NUMERIC_TYPE	-
OTHER	-

Range Comparator Interrupt Enable Register (ADC12Bn_RCIRQE0)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	RCIRQE31	RCIRQE30	RCIRQE29	RCIRQE28	RCIRQE27	RCIRQE26	RCIRQE25	RCIRQE24
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	RCIRQE23	RCIRQE22	RCIRQE21	RCIRQE20	RCIRQE19	RCIRQE18	RCIRQE17	RCIRQE16
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	RCIRQE15	RCIRQE14	RCIRQE13	RCIRQE12	RCIRQE11	RCIRQE10	RCIRQE9	RCIRQE8
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	RCIRQE7	RCIRQE6	RCIRQE5	RCIRQE4	RCIRQE3	RCIRQE2	RCIRQE1	RCIRQE0
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] RCIRQE31 to 0 : Range Comparator Interrupt Enable bits

Bit	Description
0	Range comparator interrupt disabled.
1	Range comparator interrupt enabled

Range comparator interrupt is issued when this bit is "1" and the corresponding interrupt flags ADC12Bn_RCIRQ0.RCIRQ31 to 0 and ADC12Bn_CHSTAT0 to 31.RCIRQ are set to "1".

Range Comparator Interrupt Enable Register (ADC12Bn_RCIRQE1)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	RCIRQE63	RCIRQE62	RCIRQE61	RCIRQE60	RCIRQE59	RCIRQE58	RCIRQE57	RCIRQE56
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	RCIRQE55	RCIRQE54	RCIRQE53	RCIRQE52	RCIRQE51	RCIRQE50	RCIRQE49	RCIRQE48
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	RCIRQE47	RCIRQE46	RCIRQE45	RCIRQE44	RCIRQE43	RCIRQE42	RCIRQE41	RCIRQE40
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	RCIRQE39	RCIRQE38	RCIRQE37	RCIRQE36	RCIRQE35	RCIRQE34	RCIRQE33	RCIRQE32
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] RCIRQE63 to 32 : Range Comparator Interrupt Enable bits

Bit	Description
0	Range comparator interrupt disabled.
1	Range comparator interrupt enabled

Range comparator interrupt is issued when this bit is "1" and the corresponding interrupt flags ADC12Bn_RCIRQ1.RCIRQ63 to 32 and ADC12Bn_CHSTAT32 to 63.RCIRQ are set to "1".

5.13. Range Comparator Interrupt Clear Registers (ADC12Bn_RCIRQC0 to 1)

These registers contain bits for clearing corresponding range comparator interrupt flags in the ADC12Bn_RCIRQ0 to 1 registers. The 64 bits are assigned to 64 logical channels.

REGISTER_NAME	ADC12Bn_RCIRQCi (i = 0 to 1)
OFFSET	0x0340 + i*4
ACCESS_SIZE	B H W
MULTIPLE	0:1
NUMERIC_TYPE	-
OTHER	-

Range Comparator Interrupt Clear Register (ADC12Bn_RCIRQC0)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	RCIRQC31	RCIRQC30	RCIRQC29	RCIRQC28	RCIRQC27	RCIRQC26	RCIRQC25	RCIRQC24
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	RCIRQC23	RCIRQC22	RCIRQC21	RCIRQC20	RCIRQC19	RCIRQC18	RCIRQC17	RCIRQC16
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	RCIRQC15	RCIRQC14	RCIRQC13	RCIRQC12	RCIRQC11	RCIRQC10	RCIRQC9	RCIRQC8
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	RCIRQC7	RCIRQC6	RCIRQC5	RCIRQC4	RCIRQC3	RCIRQC2	RCIRQC1	RCIRQC0
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] RCIRQC31 to 0 : Range Comparator Interrupt Clear bits

Bit	Description
0	No effect.
1	Range comparator interrupt cleared.

When this bit is set to "1", the corresponding bit in the ADC12Bn_RCIRQ0 register and RCIRQ bit in the corresponding ADC12Bn_CHSTAT0 to 31 register are cleared.

Range Comparator Interrupt Clear Register (ADC12Bn_RCIRQC1)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	RCIRQC63	RCIRQC62	RCIRQC61	RCIRQC60	RCIRQC59	RCIRQC58	RCIRQC57	RCIRQC56
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	RCIRQC55	RCIRQC54	RCIRQC53	RCIRQC52	RCIRQC51	RCIRQC50	RCIRQC49	RCIRQC48
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	RCIRQC47	RCIRQC46	RCIRQC45	RCIRQC44	RCIRQC43	RCIRQC42	RCIRQC41	RCIRQC40
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	RCIRQC39	RCIRQC38	RCIRQC37	RCIRQC36	RCIRQC35	RCIRQC34	RCIRQC33	RCIRQC32
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] RCIRQC63 to 32 : Range Comparator Interrupt Clear bits

Bit	Description
0	No effect.
1	Range comparator interrupt cleared.

When this bit is set to "1", the corresponding bit in the ADC12Bn_RCIRQ1 register and RCIRQ bit in the corresponding ADC12Bn_CHSTAT32 to 63 register are cleared.

5.14. Pulse Counter Interrupt Flag Registers (ADC12Bn_PCIRQ0 to 1)

Pulse Comparator Interrupt Flag Registers ADC12Bn_PCIRQ0 to 1 contain the status of pulse counter interrupt flags for all 64 logical channels.

REGISTER_NAME	ADC12Bn_PCIRQi (i = 0 to 1)
OFFSET	0x0348 + i*4
ACCESS_SIZE	B H W
MULTIPLE	0:1
NUMERIC_TYPE	-
OTHER	-

Pulse Counter Interrupt Flag Register (ADC12Bn_PCIRQ0)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	PCIRQ31	PCIRQ30	PCIRQ29	PCIRQ28	PCIRQ27	PCIRQ26	PCIRQ25	PCIRQ24
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	PCIRQ23	PCIRQ22	PCIRQ21	PCIRQ20	PCIRQ19	PCIRQ18	PCIRQ17	PCIRQ16
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	PCIRQ15	PCIRQ14	PCIRQ13	PCIRQ12	PCIRQ11	PCIRQ10	PCIRQ9	PCIRQ8
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	PCIRQ7	PCIRQ6	PCIRQ5	PCIRQ4	PCIRQ3	PCIRQ2	PCIRQ1	PCIRQ0
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] PCIRQ31 to 0 : Pulse Counter Interrupt flags

Bit	Description
0	Pulse counter interrupt not detected.
1	Pulse counter interrupt detected.

This register returns the status of the pulse counter interrupt flag which is set when positive counter ADC12B_PCCTRL0 to 31.PCTPCT of the corresponding logical channel decrements to zero. The positive counter and negative counter are stopped as long as the pulse counter interrupt flag of the appropriate channel is set.

This bit is cleared by writing "1" to the corresponding bit in the ADC12Bn_PCIRQC0 register.

This bit is identical to the PCIRQ bit in the corresponding ADC12Bn_CHSTAT0 to 31 register.

Pulse Counter Interrupt Flag Register (ADC12Bn_PCIRQ1)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	PCIRQ63	PCIRQ62	PCIRQ61	PCIRQ60	PCIRQ59	PCIRQ58	PCIRQ57	PCIRQ56
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	PCIRQ55	PCIRQ54	PCIRQ53	PCIRQ52	PCIRQ51	PCIRQ50	PCIRQ49	PCIRQ48
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	PCIRQ47	PCIRQ46	PCIRQ45	PCIRQ44	PCIRQ43	PCIRQ42	PCIRQ41	PCIRQ40
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	PCIRQ39	PCIRQ38	PCIRQ37	PCIRQ36	PCIRQ35	PCIRQ34	PCIRQ33	PCIRQ32
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] PCIRQ63 to 32 : Pulse Counter Interrupt flags

Bit	Description
0	Pulse counter interrupt not detected.
1	Pulse counter interrupt detected.

This register returns the status of the pulse counter interrupt flag which is set when positive counter ADC12B_PCCTRL32 to 63.PCTPCT of the corresponding logical channel decrements to zero. The positive counter and negative counter are stopped as long as the pulse counter interrupt flag of the appropriate channel is set.

This bit is cleared by writing "1" to the corresponding bit in the ADC12Bn_PCIRQC1 register.

This bit is identical to the PCIRQ bit in the corresponding ADC12Bn_CHSTAT32 to 63 register.

5.15. Pulse Counter Interrupt Enable Registers (ADC12Bn_PCIRQE0 to 1)

These registers contain enable bits for all 64 logical channels, dedicated to the generation of pulse counter interrupt.

REGISTER_NAME	ADC12Bn_PCIRQEi (i = 0 to 1)
OFFSET	0x0350 + i*4
ACCESS_SIZE	B H W
MULTIPLE	0:1
NUMERIC_TYPE	-
OTHER	-

Pulse Counter Interrupt Enable Register (ADC12Bn_PCIRQE0)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	PCIRQE31	PCIRQE30	PCIRQE29	PCIRQE28	PCIRQE27	PCIRQE26	PCIRQE25	PCIRQE24
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	PCIRQE23	PCIRQE22	PCIRQE21	PCIRQE20	PCIRQE19	PCIRQE18	PCIRQE17	PCIRQE16
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	PCIRQE15	PCIRQE14	PCIRQE13	PCIRQE12	PCIRQE11	PCIRQE10	PCIRQE9	PCIRQE8
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	PCIRQE7	PCIRQE6	PCIRQE5	PCIRQE4	PCIRQE3	PCIRQE2	PCIRQE1	PCIRQE0
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] PCIRQE31 to 0 : Pulse Counter Interrupt Enable bits

Bit	Description
0	Pulse counter interrupt is disabled.
1	Pulse counter interrupt is enabled.

Pulse counter interrupt is issued when this bit is "1" and the corresponding interrupt flags ADC12Bn_PCIRQ0.PCIRQ31 to 0 and ADC12Bn_CHSTAT0 to 31.PCIRQ are set to "1".

Pulse Counter Interrupt Enable Register (ADC12Bn_PCIRQE1)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	PCIRQE63	PCIRQE62	PCIRQE61	PCIRQE60	PCIRQE59	PCIRQE58	PCIRQE57	PCIRQE56
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	PCIRQE55	PCIRQE54	PCIRQE53	PCIRQE52	PCIRQE51	PCIRQE50	PCIRQE49	PCIRQE48
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	PCIRQE47	PCIRQE46	PCIRQE45	PCIRQE44	PCIRQE43	PCIRQE42	PCIRQE41	PCIRQE40
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	PCIRQE39	PCIRQE38	PCIRQE37	PCIRQE36	PCIRQE35	PCIRQE34	PCIRQE33	PCIRQE32
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] PCIRQE63 to 32 : Pulse Counter Interrupt Enable bits

Bit	Description
0	Pulse counter interrupt is disabled.
1	Pulse counter interrupt is enabled.

Pulse counter interrupt is issued when this bit is "1" and the corresponding interrupt flags ADC12Bn_PCIRQ1.PCIRQ63 to 32 and ADC12Bn_CHSTAT32 to 63.PCIRQ are set to "1".

5.16. Pulse Counter Interrupt Clear Registers (ADC12Bn_PCIRQC0 to 1)

These registers contain bits for clearing corresponding pulse counter interrupt flags in the ADC12Bn_PCIRQ0 to 1 registers. The 64 bits are assigned to 64 logical channels

REGISTER_NAME	ADC12Bn_RCIRQCi (i = 0 to 1)
OFFSET	0x0358 + i*4
ACCESS_SIZE	B H W
MULTIPLE	0:1
NUMERIC_TYPE	-
OTHER	-

Pulse Counter Interrupt Clear Register (ADC12Bn_PCIRQC0)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	PCIRQC31	PCIRQC30	PCIRQC29	PCIRQC28	PCIRQC27	PCIRQC26	PCIRQC25	PCIRQC24
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	PCIRQC23	PCIRQC22	PCIRQC21	PCIRQC20	PCIRQC19	PCIRQC18	PCIRQC17	PCIRQC16
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	PCIRQC15	PCIRQC14	PCIRQC13	PCIRQC12	PCIRQC11	PCIRQC10	PCIRQC9	PCIRQC8
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	PCIRQC7	PCIRQC6	PCIRQC5	PCIRQC4	PCIRQC3	PCIRQC2	PCIRQC1	PCIRQC0
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] PCIRQC31 to 0 : ADC Pulse Counter Interrupt Clear bits

Bit	Description
0	No effect.
1	Pulse counter interrupt cleared.

When this bit is set to "1", the corresponding bit in the ADC12Bn_PCIRQ0 register and PCIRQ bit in the corresponding ADC12Bn_CHSTAT0 to 31 register are cleared.

Additionally, the corresponding positive and negative counter (ADC12Bn_PCCTRL0 to 31.PCTPCT and ADC12Bn_PCCTRL0 to 31.PCTNCT) are reloaded with their reload values defined in the ADC12Bn_PCCTRL0 to 31.PCTPRL and ADC12Bn_PCCTRL0 to 31.PCTNRL.

Pulse Counter Interrupt Clear Register (ADC12Bn_PCIRQC1)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	PCIRQC63	PCIRQC62	PCIRQC61	PCIRQC60	PCIRQC59	PCIRQC58	PCIRQC57	PCIRQC56
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	PCIRQC55	PCIRQC54	PCIRQC53	PCIRQC52	PCIRQC51	PCIRQC50	PCIRQC49	PCIRQC48
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	PCIRQC47	PCIRQC46	PCIRQC45	PCIRQC44	PCIRQC43	PCIRQC42	PCIRQC41	PCIRQC40
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	PCIRQC39	PCIRQC38	PCIRQC37	PCIRQC36	PCIRQC35	PCIRQC34	PCIRQC33	PCIRQC32
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] PCIRQC63 to 32 : ADC Pulse Counter Interrupt Clear bits

Bit	Description
0	No effect.
1	Pulse counter interrupt cleared.

When this bit is set to "1", the corresponding bit in the ADC12Bn_PCIRQ1 register and PCIRQ bit in the corresponding ADC12Bn_CHSTAT32 to 63 register are cleared.

Additionally, the corresponding positive and negative counter (ADC12Bn_PCCTRL32 to 63.PCTPCT and ADC12Bn_PCCTRL32 to 63.PCTNCT) are reloaded with their reload values defined in the ADC12Bn_PCCTRL32 to 63.PCTPRL and ADC12Bn_PCCTRL32 to 63.PCTNRL.

5.17. A/D Channel Trigger Status Flag Registers (ADC12Bn_TRGST0 to 1)

A/D Channel Trigger Status Flag Registers ADC12Bn_TRGST0 to 1 contain the status of conversion requests for all 64 logical channels.

REGISTER_NAME	ADC12Bn_TRGSTi (i = 0 to 1)
OFFSET	0x0360 + i*4
ACCESS_SIZE	B H W
MULTIPLE	0:1
NUMERIC_TYPE	-
OTHER	-

A/D Channel Trigger Status Flag Register (ADC12Bn_TRGST0)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	TRGST31	TRGST30	TRGST29	TRGST28	TRGST27	TRGST26	TRGST25	TRGST24
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	TRGST23	TRGST22	TRGST21	TRGST20	TRGST19	TRGST18	TRGST17	TRGST16
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	TRGST15	TRGST14	TRGST13	TRGST12	TRGST11	TRGST10	TRGST9	TRGST8
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	TRGST7	TRGST6	TRGST5	TRGST4	TRGST3	TRGST2	TRGST1	TRGST0
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] TRGST31 to 0 : A/D Channel Trigger Status flags

Bit	Description
0	No conversion request.
1	Conversion request is issued.

This bit is set if any conversion request, dedicated to the corresponding logical channel, occurs (software, hardware, trigger by completion of preceding channel or idle trigger). Moreover, the bit is set if:

- The group of the corresponding logical channel is interrupted just before a channel configured as "restart" channel (ADC12Bn_CHCTRL32 to 63.RSMRST = "10")

The channel is the start channel of the group or the last already converted channel configured as "resume" channel (ADC12Bn_CHCTRL32 to 63.RSMRST = "01").

This bit is cleared under one of the following conditions:

- Completion of the logical channel conversion (the last conversion in the case of multiple conversion channels), at the same time the conversion done interrupt flag is set
- The corresponding group processing is interrupted just before the conversion of this channel is started and the channel is not configured as "resume" channel (ADC12Bn_CHCTRL0 to 31.RSMRST = "01"). Instead, the group interrupted interrupt flag is set
- Writing "1" to the corresponding bit in the ADC12Bn_TRGCL0 register or writing "1" to the corresponding ADC12Bn_CHCTRL0 to 31.TRGCL bit.

When setting and clearing of the bit takes place at the same time, clearing has priority.

This bit is identical to the TRGST bit in the corresponding ADC12Bn_CHSTAT0 to 31 register.

A/D Channel Trigger Status Flag Register (ADC12Bn_TRGST1)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	TRGST63	TRGST62	TRGST61	TRGST60	TRGST59	TRGST58	TRGST57	TRGST56
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	TRGST55	TRGST54	TRGST53	TRGST52	TRGST51	TRGST50	TRGST49	TRGST48
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	TRGST47	TRGST46	TRGST45	TRGST44	TRGST43	TRGST42	TRGST41	TRGST40
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	TRGST39	TRGST38	TRGST37	TRGST36	TRGST35	TRGST34	TRGST33	TRGST32
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] TRGST63 to 32 : A/D Channel Trigger Status flags

Bit	Description
0	No conversion request.
1	Conversion request is issued.

This bit is set if any conversion request, dedicated to the corresponding logical channel, occurs (software, hardware, trigger by completion of preceding channel or idle trigger). Moreover, the bit is set if:

- The group of the corresponding logical channel is interrupted just before a channel configured as "restart" channel (ADC12Bn_CHCTRL32 to 63.RSMRST = "10")
- The channel is the start channel of the group or the last already converted channel configured as "resume" channel (ADC12Bn_CHCTRL32 to 63.RSMRST = "01").

This bit is cleared under one of the following conditions:

- Completion of the logical channel conversion, at the same time the conversion done interrupt flag is set
- The corresponding group processing is interrupted just before the conversion of this channel is started and the channel is not configured as "resume" channel (ADC12Bn_CHCTRL32 to 63.RSMRST = "01"). Instead, the group interrupted interrupt flag is set
- Writing "1" to the corresponding bit in the ADC12Bn_TRGCL1 register or writing "1" to the corresponding ADC12Bn_CHCTRL32 to 63.TRGCL bit.

When setting and clearing of the bit takes place at the same time, clearing has priority.

This bit is identical to the TRGST bit in the corresponding ADC12Bn_CHSTAT32 to 63 register.

5.18. A/D Channel Trigger Clear Registers (ADC12Bn_TRGCL0 to 1)

These registers contain bits for clearing corresponding A/D channel trigger status flags in the ADC12Bn_TRGST0 to 1 registers. The 64 bits are assigned to 64 logical channels.

REGISTER_NAME	ADC12Bn_TRGCLi (i = 0 to 1)
OFFSET	0x0368 + i*4
ACCESS_SIZE	B H W
MULTIPLE	0:1
NUMERIC_TYPE	-
OTHER	-

A/D Channel Trigger Clear Register (ADC12Bn_TRGCL0)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	TRGCL31	TRGCL30	TRGCL29	TRGCL28	TRGCL27	TRGCL26	TRGCL25	TRGCL24
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	TRGCL23	TRGCL22	TRGCL21	TRGCL20	TRGCL19	TRGCL18	TRGCL17	TRGCL16
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	TRGCL15	TRGCL14	TRGCL13	TRGCL12	TRGCL11	TRGCL10	TRGCL9	TRGCL8
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	TRGCL7	TRGCL6	TRGCL5	TRGCL4	TRGCL3	TRGCL2	TRGCL1	TRGCL0
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] TRGCL31 to 0 : Trigger Status Clear bits

Bit	Description
0	No effect.
1	A/D channel trigger status is cleared.

When this bit is set to "1", the corresponding bit in the ADC12Bn_TRGST0 register and TRGST bit in the corresponding ADC12Bn_CHSTAT0 to 31 register are cleared.

If the corresponding trigger status is cleared during A/D conversion, the operation is dependent on the setting of forced stop mode (ADC12Bn_CTRL.FSMD).

- Forced stop is enabled (ADC12Bn_CTRL.FSMD = "1"): A/D conversion is stopped after interrupt operation.
- Forced stop is disabled (ADC12Bn_CTRL.FSMD = "0"): A/D conversion is not stopped (but A/D conversion result is not updated). Hence, when the next conversion is already requested, the wait time from trigger status flag clear to the start of next conversion can be up to the maximum A/D conversion period.

This bit is identical to the TRGCL bit in the corresponding ADC12Bn_CHCTRL0 to 31 register.

Note:

- *If forced stop is disabled (ADC12Bn_CTRL.FSMD = "0"), do not set trigger status flag again during the same conversion after the trigger status is cleared.*
  Please set again after the end timing of the cleared conversion.

A/D Channel Trigger Clear Register (ADC12Bn_TRGCL1)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	TRGCL63	TRGCL62	TRGCL61	TRGCL60	TRGCL59	TRGCL58	TRGCL57	TRGCL56
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	TRGCL55	TRGCL54	TRGCL53	TRGCL52	TRGCL51	TRGCL50	TRGCL49	TRGCL48
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	TRGCL47	TRGCL46	TRGCL45	TRGCL44	TRGCL43	TRGCL42	TRGCL41	TRGCL40
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	TRGCL39	TRGCL38	TRGCL37	TRGCL36	TRGCL35	TRGCL34	TRGCL33	TRGCL32
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] TRGCL63 to 32 : Trigger Status Clear bits

Bit	Description
0	No effect.
1	A/D channel trigger status is cleared.

When this bit is set to "1", the corresponding bit in the ADC12Bn_TRGST1 register and TRGST bit in the corresponding ADC12Bn_CHSTAT32 to 63 register are cleared.

If the corresponding trigger status is cleared during A/D conversion, the operation is dependent on the setting of forced stop mode (ADC12Bn_CTRL.FSMD).

- Forced stop is enabled (ADC12Bn_CTRL.FSMD = "1"): A/D conversion is stopped after interrupt operation.
- Forced stop is disabled (ADC12Bn_CTRL.FSMD = "0"): A/D conversion is not stopped (but A/D conversion result is not updated). Hence, when the next conversion is already requested, the wait time from trigger status flag clear to the start of next conversion can be up to the maximum A/D conversion period.

This bit is identical to the TRGCL bit in the corresponding ADC12Bn_CHCTRL32 to 63 register.

Note:

- *If forced stop is disabled (ADC12Bn_CTRL.FSMD = "0"), do not set trigger status flag again during the same conversion after the trigger status is cleared.*
  Please set again after the end timing of the cleared conversion.

5.19. A/D Channel Trigger Overrun Flag Registers (ADC12Bn_TRGOR0 to 1)

A/D Channel Trigger Overrun Flag Registers ADC12Bn_TRGOR0 to 1 register the possible trials to set already active (set to "1") trigger status for all 64 logical channels.

REGISTER_NAME	ADC12Bn_TRGORi (i = 0 to 1)
OFFSET	0x0378 + i*4
ACCESS_SIZE	B H W
MULTIPLE	0:1
NUMERIC_TYPE	-
OTHER	-

A/D Channel Trigger Overrun Flag Register (ADC12Bn_TRGOR0)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	TRGOR31	TRGOR30	TRGOR29	TRGOR28	TRGOR27	TRGOR26	TRGOR25	TRGOR24
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	TRGOR23	TRGOR22	TRGOR21	TRGOR20	TRGOR19	TRGOR18	TRGOR17	TRGOR16
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	TRGOR15	TRGOR14	TRGOR13	TRGOR12	TRGOR11	TRGOR10	TRGOR9	TRGOR8
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	TRGOR7	TRGOR6	TRGOR5	TRGOR4	TRGOR3	TRGOR2	TRGOR1	TRGOR0
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] TRGOR31 to 0 : A/D Channel Trigger Overrun flags

Bit	Description
0	No trigger overrun happened
1	Trigger overrun occured

This bit is set to "1" under following conditions:

- Conversion request is issued although the corresponding trigger status bits ADC12Bn_TRGST0.TRGST and ADC12Bn_CHSTAT0 to 31.TRGST are already set to "1"
- Software and hardware trigger are issued at the same cycle and corresponding trigger type ADC12Bn_CHCTRL0 to 31.TRGTYP[1:0] is set to "01"

Writing "1" to the corresponding bit in the ADC12Bn_TRGORC0 register clears this bit..

A/D Channel Trigger Overrun Flag Register (ADC12Bn_TRGOR1)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	TRGOR63	TRGOR62	TRGOR61	TRGOR60	TRGOR59	TRGOR58	TRGOR57	TRGOR56
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	TRGOR55	TRGOR54	TRGOR53	TRGOR52	TRGOR51	TRGOR50	TRGOR49	TRGOR48
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	TRGOR47	TRGOR46	TRGOR45	TRGOR44	TRGOR43	TRGOR42	TRGOR41	TRGOR40
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	TRGOR39	TRGOR38	TRGOR37	TRGOR36	TRGOR35	TRGOR34	TRGOR33	TRGOR32
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] TRGOR63 to 32 : A/D Channel Trigger Overrun flags

Bit	Description
0	No trigger overrun happened
1	Trigger overrun occured

This bit is set to "1" under following conditions:

- Conversion request is issued although the corresponding trigger status bits ADC12Bn_TRGST1.TRGST and ADC12Bn_CHSTAT32 to 63.TRGST are already set to "1"
- Software and hardware trigger are issued at the same cycle and corresponding trigger type ADC12Bn_CHCTRL32 to 63.TRGTYP[1:0] is set to "01"

Writing "1" to the corresponding bit in the ADC12Bn_TRGORC1 register clears this bit.

5.20. A/D Channel Trigger Overrun Clear Registers (ADC12Bn_TRGORC0 to 1)

These registers contain bits for clearing corresponding A/D channel trigger overrun flags in the ADC12Bn_TRGOR0 to 1 registers. The 64 bits are assigned to 64 logical channels.

REGISTER_NAME	ADC12Bn_TRGORCi (i = 0 to 1)
OFFSET	0x0380 + i*4
ACCESS_SIZE	B H W
MULTIPLE	0:1
NUMERIC_TYPE	-
OTHER	-

A/D Channel Trigger Overrun Clear Register (ADC12Bn_TRGORC0)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	TRGORC31	TRGORC30	TRGORC29	TRGORC28	TRGORC27	TRGORC26	TRGORC25	TRGORC24
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	TRGORC23	TRGORC22	TRGORC21	TRGORC20	TRGORC19	TRGORC18	TRGORC17	TRGORC16
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	TRGORC15	TRGORC14	TRGORC13	TRGORC12	TRGORC11	TRGORC10	TRGORC9	TRGORC8
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	TRGORC7	TRGORC6	TRGORC5	TRGORC4	TRGORC3	TRGORC2	TRGORC1	TRGORC0
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] TRGORC31 to 0 : Trigger Overrun Clear bits

Bit	Description
0	No effect.
1	Trigger overrun flag is cleared.

When this bit is set to "1", the corresponding bit in the ADC12Bn_TRGOR0 register is cleared.

A/D Channel Trigger Overrun Clear Register (ADC12Bn_TRGORC1)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	TRGORC63	TRGORC62	TRGORC61	TRGORC60	TRGORC59	TRGORC58	TRGORC57	TRGORC56
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	TRGORC55	TRGORC54	TRGORC53	TRGORC52	TRGORC51	TRGORC50	TRGORC49	TRGORC48
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	TRGORC47	TRGORC46	TRGORC45	TRGORC44	TRGORC43	TRGORC42	TRGORC41	TRGORC40
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	TRGORC39	TRGORC38	TRGORC37	TRGORC36	TRGORC35	TRGORC34	TRGORC33	TRGORC32
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] TRGORC63 to 32 : Trigger Overrun Clear bits

Bit	Description
0	No effect.
1	Trigger overrun flag is cleared.

When this bit is set to "1", the corresponding bit in the ADC12Bn_TRGOR1 register is cleared.

5.21. Range Comparator Over Threshold Flag Registers (ADC12Bn_RCOTF0 to 1)

The flag bits (ADC12B_RCOTF0 to 1) are set when the result of the range comparator is "outside range" and the converted value is above the value of the upper threshold register.

REGISTER_NAME	ADC12Bn_RCOTFi (i = 0 to 1)
OFFSET	0x0370 + i*4
ACCESS_SIZE	B H W
MULTIPLE	0:1
NUMERIC_TYPE	-
OTHER	-

Range Comparator Over Threshold Flag Register (ADC12Bn_RCOTF0)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	RCOTF31	RCOTF30	RCOTF29	RCOTF28	RCOTF27	RCOTF26	RCOTF25	RCOTF24
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	RCOTF23	RCOTF22	RCOTF21	RCOTF20	RCOTF19	RCOTF18	RCOTF17	RCOTF16
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	RCOTF15	RCOTF14	RCOTF13	RCOTF12	RCOTF11	RCOTF10	RCOTF9	RCOTF8
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	RCOTF7	RCOTF6	RCOTF5	RCOTF4	RCOTF3	RCOTF2	RCOTF1	RCOTF0
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] RCOTF31 to 0 : Range Comparator Over Threshold flags

The flag is only applicable in "outside range" mode i.e. while the RCINVSEL bit in the corresponding ADC12Bn_CHCTRL0 to 31 register is "0". If a range comparator interrupt is signaled (corresponding bits ADC12Bn_RCIRQ0.RCIRQ31 to 0 = ADC12Bn_CHSTAT0 to 31.RCIRQ = "1"), this flag has the following meaning:

Bit	Description
0	The conversion result is less than or equal to the upper threshold.
1	The conversion result is above the upper threshold.

This bit is updated only in the case the corresponding interrupt flag (ADC12Bn_RCIRQ0.RCIRQ, ADC12Bn_CHSTAT0 to 31.RCIRQ) has a rising edge.

This bit is identical to the RCOTF bit in the corresponding ADC12Bn_CHSTAT0 to 31 register.

Range Comparator Over Threshold Flag Register (ADC12Bn_RCOTF1)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	RCOTF63	RCOTF62	RCOTF61	RCOTF60	RCOTF59	RCOTF58	RCOTF57	RCOTF56
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	RCOTF55	RCOTF54	RCOTF53	RCOTF52	RCOTF51	RCOTF50	RCOTF49	RCOTF48
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	RCOTF47	RCOTF46	RCOTF45	RCOTF44	RCOTF43	RCOTF42	RCOTF41	RCOTF40
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	RCOTF39	RCOTF38	RCOTF37	RCOTF36	RCOTF35	RCOTF34	RCOTF33	RCOTF32
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] RCOTF63 to 32 : Range Comparator Over Threshold flags

The flag is only applicable in "outside range" mode i.e. while the RCINVSEL bit in the corresponding ADC12Bn_CHCTRL32 to 63 register is "0". If a range comparator interrupt is signaled (corresponding bits ADC12Bn_RCIRQ1.RCIRQ63 to 32 = ADC12Bn_CHSTAT32 to 63.RCIRQ = "1"), this flag has the following meaning:

Bit	Description
0	The conversion result is less than or equal to the upper threshold.
1	The conversion result is above the upper threshold.

This bit is updated only in the case the corresponding interrupt flag (ADC12Bn_RCIRQ1.RCIRQ, ADC12Bn_CHSTAT32 to 63.RCIRQ) has a rising edge.

This bit is identical to the RCOTF bit in the corresponding ADC12Bn_CHSTAT32 to 63 register.

5.22. Conversion Done DMA Select Registers (ADC12Bn_CDDS0 to 3)

These four registers specify four logical channels whose conversion done interrupt flags can initiate DMA requests to transfer A/D conversion results to memory. Only ADC12Bn_CDDS0 register is described here. Other registers (ADC12Bn_CDDS1, ADC12Bn_CDDS2 and ADC12Bn_CDDS3) have identical bit fields.

REGISTER_NAME	ADC12Bn_CDDSi (i = 0 to 3)
OFFSET	0x0388 + i*2
ACCESS_SIZE	B H W
MULTIPLE	0:3
NUMERIC_TYPE	-
OTHER	-

Conversion Done DMA Select Register (ADC12Bn_CDDS0)

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	CDCHEN	CDCHNUM[5]	CDCHNUM[4]	CDCHNUM[3]	CDCHNUM[2]	CDCHNUM[1]	CDCHNUM[0]
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

[bit15:7] Reserved : Reserved bits

When writing, always write "0".

When reading, "0" is always read.

[bit6] CDCHEN : Channel conversion done DMA dedicated interrupt enable bit

Bit	Description
0	Interrupt for triggering DMA request on conversion done interrupt flag (ADC12Bn_CHSTAT0 to 63.CDONEIRQ = ADC12Bn_CDONEIRQ0 to 1.CDONEIRQ = "1") of the logical channel defined by CDCHNUM bits is disabled.
1	Interrupt for triggering DMA request on conversion done interrupt flag (ADC12Bn_CHSTAT0 to 63.CDONEIRQ = ADC12Bn_CDONEIRQ0 to 1.CDONEIRQ = "1") of the logical channel defined by CDCHNUM bits is enabled.

[bit5:0] CDCHNUM[5:0] : Number of the logical channel selected for conversion done interrupt for triggering a DMA request

CDCHNUM[5:0]	Description
000000	Logical channel 0 selected.
000001	Logical channel 1 selected.
...	...
111111	Logical channel 63 selected.

The conversion done interrupts of the group last logical channels are good candidates for DMA burst setup, since the group result registers can be read linearly.

5.23. A/D Converter Comparison Time Setting Register (ADC12Bn_CT)

ADC12Bn_CT register specifies the comparison time of the A/D converter. It is not allowed to update the value of this register during A/D conversion operation (ADC12Bn_TRGST0 to 1.TRGST and ADC12Bn_CHSTAT0 to 63.TRGST="1").

REGISTER_NAME	ADC12Bn_CT
OFFSET	0x0390
ACCESS_SIZE	B H W
MULTIPLE	1
NUMERIC_TYPE	-
OTHER	-

A/D Converter Comparison Time Setting Register (ADC12Bn_CT)

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	CT[15]	CT[14]	CT[13]	CT[12]	CT[11]	CT[10]	CT[9]	CT[8]
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	CT[7]	CT[6]	CT[5]	CT[4]	CT[3]	CT[2]	CT[1]	CT[0]
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	1	0	0

[bit15:0] CT[15:0] : A/D Converter Comparison Time Setting bits

These bits specify the comparison phase time.

If [CT value < 4]: Comparison time = (CT value x 12 + 4) x Peripheral clock period

If [CT value >= 4]: Comparison time = (CT value x 13) x Peripheral clock period

Do not set CT value to 0.

For specific values of minimum and maximum comparison time, please refer to the Device Data Sheet.

5.24. A/D Converter Resumption Time Setting Register (ADC12Bn_RT)

ADC12Bn_RT register specifies the resumption time of the A/D converter. It is not allowed to update the value of this register during A/D conversion operation (ADC12Bn_TRGST0 to 1.TRGST and ADC12Bn_CHSTAT0 to 63.TRGST="1").

REGISTER_NAME	ADC12Bn_RT
OFFSET	0x0392
ACCESS_SIZE	B H W
MULTIPLE	1
NUMERIC_TYPE	-
OTHER	-

A/D Converter Resumption Time Setting Register (ADC12Bn_RT)

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	RT[7]	RT[6]	RT[5]	RT[4]	RT[3]	RT[2]	RT[1]	RT[0]
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	1	1	1	1	1

[bit15:8] Reserved : Reserved bits

When writing, always write "0".

When reading, "0" is always read.

[bit7:0] RT[7:0] : A/D Converter Resumption Time Setting bits

These bits specify the resumption phase time (power-up wait time).

Please set RT value is longer than the specific values of maximum resumption time.

Do not set RT value to 0.

RT value >= Maximum resumption time / Peripheral clock period.

For specific values of maximum resumption time, please refer to the Device Data Sheet.

5.25. A/D Converter Sampling Time Setting Registers (ADC12Bn_ST0 to 3)

ADC12Bn_ST0 to 3 registers specify four different settings for the selection of the sampling time of the A/D converter. It is not allowed to update the value of these registers during A/D sampling operation (ADC12Bn_TRGST0 to 1.TRGST and ADC12Bn_CHSTAT0 to 63.TRGST="1"). Only ADC12Bn_ST0 register is described here. Other registers (ADC12Bn_ST1, ADC12Bn_ST2 and ADC12Bn_ST3) have similar bit fields.

REGISTER_NAME	ADC12Bn_STi (i = 0 to 3)
OFFSET	0x0394 + i*2
ACCESS_SIZE	B H W
MULTIPLE	0:3
NUMERIC_TYPE	-
OTHER	-

A/D Converter Sampling Time Setting Register (ADC12Bn_ST0)

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	ST0[15]	ST0[14]	ST0[13]	ST0[12]	ST0[11]	ST0[10]	ST0[9]	ST0[8]
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	ST0[7]	ST0[6]	ST0[5]	ST0[4]	ST0[3]	ST0[2]	ST0[1]	ST0[0]
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	1	1	0	0

[bit15:0] ST0[15:0] : A/D Converter Sampling Time Setting bits

These bits specify one of the four available settings for A/D converter sampling phase time.

This time is selected for a logical channel conversion by configuring the corresponding logical channel register field ADC12Bn_CHCTRL0 to 63.SMTIME to "00".

Sampling time = ST value x Peripheral clock period

Do not set ST value below 6.

For specific values of minimum sampling time, please refer to the Device Data Sheet.

5.26. A/D Converter Offset Compensation Setting Register (ADC12Bn_OCV)

The global register ADC12Bn_OCV specifies the setting for offset compensation value.

REGISTER_NAME	ADC12Bn_OCV
OFFSET	0x039C
ACCESS_SIZE	B H W
MULTIPLE	1
NUMERIC_TYPE	-
OTHER	-

A/D Converter Offset Compensation Setting Register (ADC12Bn_OCV)

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	OCV[7]	OCV[6]	OCV[5]	OCV[4]	OCV[3]	OCV[2]	OCV[1]	OCV[0]
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	1	0

[bit15:8] Reserved : Reserved bits

When writing, always write "0".

When reading, "0" is always read.

[bit7:0] OCV[7:0] : Offset Compensation Value

The A/D Converter can be configured to convert the values of its reference voltages AVRH and AVRL in order to calculate the offset compensation value.

After A/D Converter calibration, the calculated offset compensation value must be written to this register.

For further explanation of A/D Converter calibration refer to section "A/D Converter Calibration" in chapter "3 Operation of A/D ".

5.27. A/D Converter Gain Compensation Setting Register (ADC12B_GCV)

This register ADC12Bn_GCV specifies the setting for gain compensation value.

REGISTER_NAME	ADC12Bn_GCV
OFFSET	0x039E
ACCESS_SIZE	B H W
MULTIPLE	1
NUMERIC_TYPE	-
OTHER	-

A/D Converter Gain Compensation Setting Register (ADC12Bn_GCV)

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			GCV[4]	GCV[3]	GCV[2]	GCV[1]	GCV[0]
ACCESS_TYPE	R0,W0			R/W	R/W	R/W	R/W	R/W
PROT_TYPE								
INITIAL_VALUE	000			0	0	0	0	0

[bit15:5] Reserved : Reserved bits

When writing, always write "0".

When reading, "0" is always read.

[bit4:0] GCV[4:0] : Gain Compensation Value

The A/D Converter can be configured to convert the values of its reference voltages AVRH and AVRL in order to calculate the gain compensation value.

After A/D Converter calibration, the calculated gain compensation value must be written to this register.

For further explanation of A/D Converter calibration refer to section "A/D Converter Calibration" in chapter "3 Operation of A/D ".

5.28. A/D Converter Global Control Register (ADC12Bn_CTRL)

The register configures the global control settings of the A/D Converter: A/D conversion resolution, enabling of the debug feature and the mode of ADC12Bn_STAT.ACH bits, forced stop mode and request, full range comparator mode, and power-down disable mode.

REGISTER_NAME	ADC12Bn_CTRL
OFFSET	0x3A0
ACCESS_SIZE	B H W
MULTIPLE	1
NUMERIC_TYPE	-
OTHER	-

A/D Converter Global Control Register (ADC12Bn_CTRL)

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	PDDMD	FSTP	FRCMD	FSMD	ACHMD	DBGE	RES[1]	RES[0]
ACCESS_TYPE	R/W	R0,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

When writing, always write "0".

When reading, "0" is always read.

[bit7] PDDMD : Power-down disable mode

Bit	Description
0	A/D converter goes to idle (power-down) state after A/D conversion finished (and next conversion is not requested).
1	A/D convertor does not go idle (power-down) state after A/D conversion finished (and next conversion is not requested). It can start A/D sampling without the resumption time.

Note:

- *At the following case, even if the power-down mode is disabled (PDDMD="1"), A/D converter is waited until resumption time.*
- *A/D converter is in idle (power-down) state.*
- *PDDMD is change "0" to "1".*

[bit6] FSTP : Forced stop

When the forced stop mode (FSMD = "1"):

Bit	Description	
	Read	Write
0	The value is always "0"	No effect.
1		Request forced stop of A/D conversion.

When the "not" forced stop mode (FSMD = "0"):

"0" is always read from this bit. Writing "0" or "1" to this bit has no effect.

[bit5] FRCMD : Full Range Comparator mode

Bit	Description
0	8-bit range comparator mode. ADC12Bn_RCOH and ADC12Bn_RCOL are used for 8-bit range comparator. ADC12Bn_FRCOH0 to 7 and ADC12Bn_FRCOL0 to 7 are not used.
1	12-bit range comparator mode. ADC12Bn_RCOH and ADC12Bn_RCOL are not used. ADC12Bn_FRCOH0 to 7 and ADC12Bn_FRCOL0 to 7 are used for 12-bit range comparator.

[bit4] FSMD : Forced stop mode

Bit	Description
0	The forced stop mode is disabled. − An active A/D conversion cannot be interrupted. − Request forced stop of A/D conversion is disabled.
1	The forced stop mode is enabled. − An active A/D conversion can be interrupted. − Request forced stop of A/D conversion is enabled.

[bit3] ACHMD : ACH register mode

Bit	Description
0	Direct ACH register mode. ADC12Bn_STAT.ACH shows the number of currently converted logical channel.
1	Latched ACH register mode. ADC12Bn_STAT.ACH shows the number of the logical channel whose conversion was finished last.

[bit2] DBGE : Debug Enable bit

Bit	Description
0	Debug mode disabled
1	Debug mode enabled.

When this bit is set to "1" and the processor is in debug state, the A/D Converter completes the current conversion, but further conversion is stopped. When the processor leaves debug state or DBGE is set to "0", conversion continues with the next channel from where it had stopped.

For the definition of debug state, refer to Section 12.8 of the Arm Cortex-R5 Technical Reference Manual.

[bit1:0] RES[1:0] : Resolution of A/D conversion

RES[1:0]	Description
x0	12-bit resolution
01	10-bit resolution
11	8-bit resolution.

Conversion result is stored in lower 10 bits of ADC12Bn_CD0 to 63 registers in case of 10-bit resolution and in lower 8 bits of ADC12Bn_CD0 to 63 registers if the 8-bit resolution is configured.

In case of 10-bit or 8-bit resolution, the lower 2 or 4 bits of the 12-bit conversion result are truncated (not rounded).

5.29. A/D Converter Global Status Register (ADC12Bn_STAT)

The register ADC12B_STAT is responsible for storing global status information of A/D Converter, such as currently converted channel and activity indication bit.

REGISTER_NAME	ADC12Bn_STAT
OFFSET	0x03A2
ACCESS_SIZE	B H W
MULTIPLE	1
NUMERIC_TYPE	-
OTHER	-

A/D Converter Global Status Register (ADC12Bn_STAT)

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

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved	BUSY	ACH[5]	ACH[4]	ACH[3]	ACH[2]	ACH[1]	ACH[0]
ACCESS_TYPE	RX,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

[bit15:7] Reserved : Reserved bits

Reading this bit returns an undefined.

Writing data to these bits has no effect on the operation.

[bit6] BUSY : A/D Converter Busy flag

Bit	Description
0	A/D Converter is not active, i. e. no A/D conversion is ongoing.
1	A/D Converter is active, i. e. A/D conversion is ongoing.

- Power-down disable mode (ADC12Bn_CTRL.PDDMD = "1"):

  This bit set to "1" when first conversion request (trigger status set to "1") occurs (without A/D converter resumption time).
  This bit clear to "0" when the all trigger status are cleared.

- Power-down enable mode (ADC12Bn_CTRL.PDDMD = "0"):

  This bit is set to "1" when first conversion request (trigger status set to "1") occurs and A/D Converter resumption time (power-up wait time configured as A/D converter resumption time register (ADC12Bn_RT)) elapses.
  This bit clear to "0" when the all trigger status are cleared.

If there is no idle trigger type set for any of logical channels, after all conversion requests are processed A/D Converter goes to idle (power-down) state and BUSY flag becomes "0" until next conversion request appears and resumption time elapses.

In case that idle trigger type is set (ADC12Bn_CHCTRL0 to 63.TRGTYP[1:0] = "11"), there will be always at least one conversion request active and after BUSY flag is set first time to "1", it will not change.

[bit5:0] ACH[5:0] : Converted logical channel number

This bit field depends on ADC12Bn_CTRL.ACHMD field setting.

- If ACHMD is equal to "0", ACH represents currently converted logical channel number.
- If ACHMD is equal to "1", ACH represents the last logical channel number whose conversion has finished. ACH is updated at the end of the A/D conversion only in the case the corresponding trigger status (ADC12Bn_TRGST0 to 1.TRGST and ADC12Bn_CHSTAT0 to 63.TRGST bits) is still "1".

5.30. Range Comparator Upper Threshold Registers (ADC12Bn_RCOH0 to 7)

When the 8-bit range comparator mode (ADC12Bn_CTRL.FRCMD = "0"), these registers are used for 8-bit range comparator.

If the 12-bit range comparator mode (ADC12Bn_CTRL.FRCMD = "1"), these registers are not used.

These registers specify the upper threshold values that can be selected for the 8-bit range comparator to which the output of the A/D Converter is compared. Only ADC12Bn_RCOH0 is described here. Other registers (ADC12Bn_RCOH1,..., ADC12Bn_RCOH6 and ADC12Bn_RCOH7) have similar bit fields.

REGISTER_NAME	ADC12Bn_RCOHi (i = 0 to 7)
OFFSET	0x03B1 + i*2
ACCESS_SIZE	B H W
MULTIPLE	0:7
NUMERIC_TYPE	-
OTHER	-

Range Comparator Upper Threshold Register 0 (ADC12Bn_RCOH0)

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	RCOH[7]	RCOH[6]	RCOH[5]	RCOH[4]	RCOH[3]	RCOH[2]	RCOH[1]	RCOH[0]
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

[bit7:0] RCOH[7:0] : Range Comparator Upper Threshold value

The RCOH bits define the upper comparison threshold of the range comparator 0.

The upper comparator compares the upper 8 bits of the A/D conversion result. If the value is higher than RCOH[7:0], then the conversion result is outside range.

Selection of one of eight available range comparator threshold values for the logical channel is configured by corresponding ADC12Bn_CHCTRL0 to 63.RCSEL[2:0] bit fields.

5.31. Range Comparator Lower Threshold Registers (ADC12Bn_RCOL0 to 7)

When the 8-bit range comparator mode (ADC12Bn_CTRL.FRCMD = "0"), these registers are used for 8-bit range comparator.

If the 12-bit range comparator mode (ADC12Bn_CTRL.FRCMD = "1"), these registers are not used.

These registers specify the lower threshold values that can be selected for the 8-bit range comparator to which the output of the A/D Converter is compared. Only ADC12Bn_RCOL0 is described here. Other registers (ADC12Bn_RCOL1,...,ADC12Bn_RCOL6, ADC12Bn_RCOL7) have similar bit fields.

REGISTER_NAME	ADC12Bn_RCOLi (i = 0 to 7)
OFFSET	0x03B0 + i*2
ACCESS_SIZE	B H W
MULTIPLE	0:7
NUMERIC_TYPE	-
OTHER	-

Range Comparator Lower Threshold Register 0 (ADC12Bn_RCOL0)

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	RCOL[7]	RCOL[6]	RCOL[5]	RCOL[4]	RCOL[3]	RCOL[2]	RCOL[1]	RCOL[0]
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] RCOL[7:0] : Range Comparator Lower Threshold value

The RCOL bits define the lower comparison threshold of the range comparator 0.

The lower comparator compares the upper 8 bits of the A/D conversion result. If the value is lower than RCOL[7:0], then the conversion result is outside range.

Selection of one of eight available range comparator threshold values for the logical channel is configured by corresponding ADC12Bn_CHCTRL0 to 63.RCSEL[2:0] bit fields.

5.32. Full Range Comparator Upper Threshold Registers (ADC12Bn_FRCOH0 to 7)

When the 12-bit range comparator mode (ADC12Bn_CTRL.FRCMD = "1"), these registers are used for 12-bit range comparator.

If the 8-bit range comparator mode (ADC12Bn_CTRL.FRCMD = "0"), these registers are not used.

These registers specify the upper threshold values that can be selected for the 12-bit range comparator to which the output of the A/D Converter is compared. Only ADC12Bn_FRCOH0 is described here. Other registers (ADC12Bn_FRCOH1,..., ADC12Bn_FRCOH6 and ADC12Bn_FRCOH7) have similar bit fields.

REGISTER_NAME	ADC12Bn_FRCOHi (i = 0 to 7)
OFFSET	0x03F0 + i*2
ACCESS_SIZE	B H W
MULTIPLE	0:7
NUMERIC_TYPE	-
OTHER	-

Full Range Comparator Upper Threshold Register 0 (ADC12Bn_FRCOH0)

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved				FRCOH[11]	FRCOH[10]	FRCOH[9]	FRCOH[8]
ACCESS_TYPE	R0,W0				R/W	R/W	R/W	R/W
PROT_TYPE								
INITIAL_VALUE	0000				1	1	1	1

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	FRCOH[7]	FRCOH[6]	FRCOH[5]	FRCOH[4]	FRCOH[3]	FRCOH[2]	FRCOH[1]	FRCOH[0]
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

[bit15:12] Reserved : Reserved bits

When writing, always write "0".

When reading, "0" is always read.

[bit11:0] FRCOH[11:0] : Full Range Comparator Upper Threshold value

The FRCOH bits define the upper comparison threshold of the range comparator 0.

The upper comparator compares the 12 bits, 10 bits or 8 bits of the A/D conversion result. ADC12Bn_CTRL.RES[1:0] define this resolution.

- If 12-bit resolution (ADC12Bn_CTRL.RES[1:0] = "x0"):

FRCOH[11:0]	Compares the 12 bits of A/D conversion result.

- If 10-bit resolution (ADC12Bn_CTRL.RES[1:0] = "01"):

FRCOH[11:10]	Not used for 10-bit range comparator.
FRCOH[9:0]	Compares the 10 bits of A/D conversion result.

- If 8-bit resolution (ADC12Bn_CTRL.RES[1:0] = "11"):

FRCOH[11:8]	Not used for 8-bit range comparator.
FRCOH[7:0]	Compares the 8 bits of A/D conversion result.

Selection of one of eight available range comparator threshold values for the logical channel is configured by corresponding ADC12Bn_CHCTRL0 to 63.RCSEL[2:0] bit fields.

5.33. Full Range Comparator Lower Threshold Registers (ADC12Bn_FRCOL0 to 7)

When the 12-bit range comparator mode (ADC12Bn_CTRL.FRCMD = "1"), these registers are used for 12-bit range comparator.

If the 8-bit range comparator mode (ADC12Bn_CTRL.FRCMD = "0"), these registers are not used.

These registers specify the lower threshold values that can be selected for the 12-bit range comparator to which the output of the A/D Converter is compared. Only ADC12Bn_FRCOL0 is described here. Other registers (ADC12Bn_FRCOL1,..., ADC12Bn_FRCOL6 and ADC12Bn_FRCOL7) have similar bit fields.

REGISTER_NAME	ADC12Bn_FRCOLi (i = 0 to 7)
OFFSET	0x03D0 + i*2
ACCESS_SIZE	B H W
MULTIPLE	0:7
NUMERIC_TYPE	-
OTHER	-

Full Range Comparator Lower Threshold Register 0 (ADC12Bn_FRCOL0)

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved				FRCOL[11]	FRCOL[10]	FRCOL[9]	FRCOL[8]
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	FRCOL[7]	FRCOL[6]	FRCOL[5]	FRCOL[4]	FRCOL[3]	FRCOL[2]	FRCOL[1]	FRCOL[0]
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

When writing, always write "0".

When reading, "0" is always read.

[bit11:0] FRCOL[11:0] : Full Range Comparator Lower Threshold value

The FRCOL bits define the lower comparison threshold of the range comparator 0.

The lower comparator compares the 12 bits, 10 bits or 8 bits of the A/D conversion result. ADC12Bn_CTRL.RES[1:0] define this resolution.

- If 12-bit resolution (ADC12Bn_CTRL.RES[1:0] = "x0"):

FRCOL[11:0]	Compares the 12 bits of A/D conversion result.

- If 10-bit resolution (ADC12Bn_CTRL.RES[1:0] = "01"):

FRCOL[11:10]	Not used for 10-bit range comparator.
FRCOL[9:0]	Compares the 10 bits of A/D conversion result.

- If 8-bit resolution (ADC12Bn_CTRL.RES[1:0] = "11"):

FRCOL[11:8]	Not used for 8-bit range comparator.
FRCOL[7:0]	Compares the 8 bits of A/D conversion result.

Selection of one of eight available range comparator threshold values for the logical channel is configured by corresponding ADC12Bn_CHCTRL0 to 63.RCSEL[2:0] bit fields.

5.34. A/D Multiple Conversion Channel Control Registers (ADC12Bn_MCCTRL0 to 3)

These registers ADC12Bn_MCCTRL0 to 3 contain additional configuration bits for the first four logical channels (multiple conversion logical channels). They control number of conversions, multiple conversion interruption and A/D Converter calibration setup. Only ADC12Bn_MCCTRL0 is described here. Other registers (ADC12Bn_MCCTRL1, ADC12Bn_MCCTRL2 and ADC12Bn_MCCTRL3) have identical bit fields.

REGISTER_NAME	ADC12Bn_MCCTRLi (i = 0 to 3)
OFFSET	0x03C0 + i
ACCESS_SIZE	B H W
MULTIPLE	0:3
NUMERIC_TYPE	-
OTHER	-

A/D Multiple Conversion Channel Control Register (ADC12Bn_MCCTRL0)

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved	AVRHSEL	AVRLSEL	ICIRQY	CNVNUM[3]	CNVNUM[2]	CNVNUM[1]	CNVNUM[0]
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

When writing, always write "0".

When reading, "0" is always read.

[bit6] AVRHSEL : A/D reference voltage AVRH selection bit

Bit	Description
0	AVRH voltage is not selected for A/D conversion
1	AVRH voltage is selected for A/D conversion.

In case that both bits AVRHSEL and AVRLSEL bits are set to "1", AVRLSEL has higher priority and AVRL voltage is converted.

If any of AVRHSEL/ARVRLSEL bits is set to "1", conversion of a regular analog input AN is not possible.

For further explanation of A/D Converter calibration refer to section "A/D Converter Calibration" in chapter "3 Operation of A/D ".

[bit5] AVRLSEL : A/D reference voltage AVRL selection bit

Bit	Description
0	AVRL voltage is not selected for A/D conversion
1	AVRL voltage is selected for A/D conversion

In case that both bits AVRHSEL and AVRLSEL bits are set to "1", AVRLSEL has higher priority and AVRL voltage is converted.

If any of AVRHSEL/ARVRLSEL bits is set to "1", conversion of a regular analog input AN is not possible.

For further explanation of A/D Converter calibration refer to section "A/D Converter Calibration" in chapter "3 Operation of A/D ".

[bit4] ICIRQY : Intra-channel interruptibility for the multiple conversion logical channel

This bit determines if the multiple conversion logical channel can be interrupted between single conversions, in case the conversion request with higher priority is issued.

Bit	Description
0	Multiple conversion logical channel can be interrupted between single conversions.
1	Multiple conversion logical channel cannot be interrupted between single conversions.

This bit has no effect if CNVNUM[3:0] is configured to "0000".

[bit3:0] CNVNUM[3:0] : Number of conversions for the multiple conversion channel

This bit field specifies the number of A/D conversions to be performed if the conversion request for the multiple conversion logical channel is issued.

CNVNUM[3:0]	Description
0000	1 conversion. With this setting multiple conversion logical channel behaves like any other logical channel.
0001	2 conversions.
0010	3 conversions.
...	...
1111	16 conversions

CNVNUM[3:0] are not allowed to update during A/D conversion operation (ADC12Bn_TRGST0 to 1.TRGST and ADC12Bn_CHSTAT0 to 63.TRGST="1").

5.35. A/D Multiple Conversion Channel Status Registers (ADC12Bn_MCSTAT0 to 3)

These registers contain the current status of finished conversion number for first four multiple conversion logical channels. Only ADC12Bn_MCSTAT0 is described here. Other registers (ADC12Bn_MCSTAT1, ADC12Bn_MCSTAT2 and ADC12Bn_MCSTAT3) have identical bit fields.

REGISTER_NAME	ADC12Bn_MCSTATi (i = 0 to 3)
OFFSET	0x03E0 + i
ACCESS_SIZE	B H W
MULTIPLE	0:3
NUMERIC_TYPE	-
OTHER	-

A/D Multiple Conversion Channel Status Register (ADC12Bn_MCSTAT0)

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved			MCCNT[4]	MCCNT[3]	MCCNT[2]	MCCNT[1]	MCCNT[0]
ACCESS_TYPE	RX,WX			R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE								
INITIAL_VALUE	000			0	0	0	0	0

[bit7:5] Reserved : Reserved bits

Reading this bit returns an undefined.

Writing data to these bits has no effect on the operation.

[bit4:0] MCCNT[4:0]

Conversion counter counting the number of already finished A/D conversions of the multiple conversion logical channel.

After the set event of the conversion done interrupt flag, the ADC12Bn_MCSTAT.MCCNT[4:0] are meaningless.

MCCNT[4:0]	Description
00000	No conversion is finished.
00001	1 conversion is finished.
...	...
10000	16 conversions are finished.
10001-11111	Reserved

CHAPTER 16: Stepper Motor Controller

This chapter explains the functions and operations of the Stepper Motor Controller (SMC).

1. Overview
2. Configuration and Block Diagram
3. Operation of the Stepper Motor Controller
4. Registers

CODE: SMC-S6J3300-E1

1. Overview

This section lists the main features for Stepper Motor Controller.

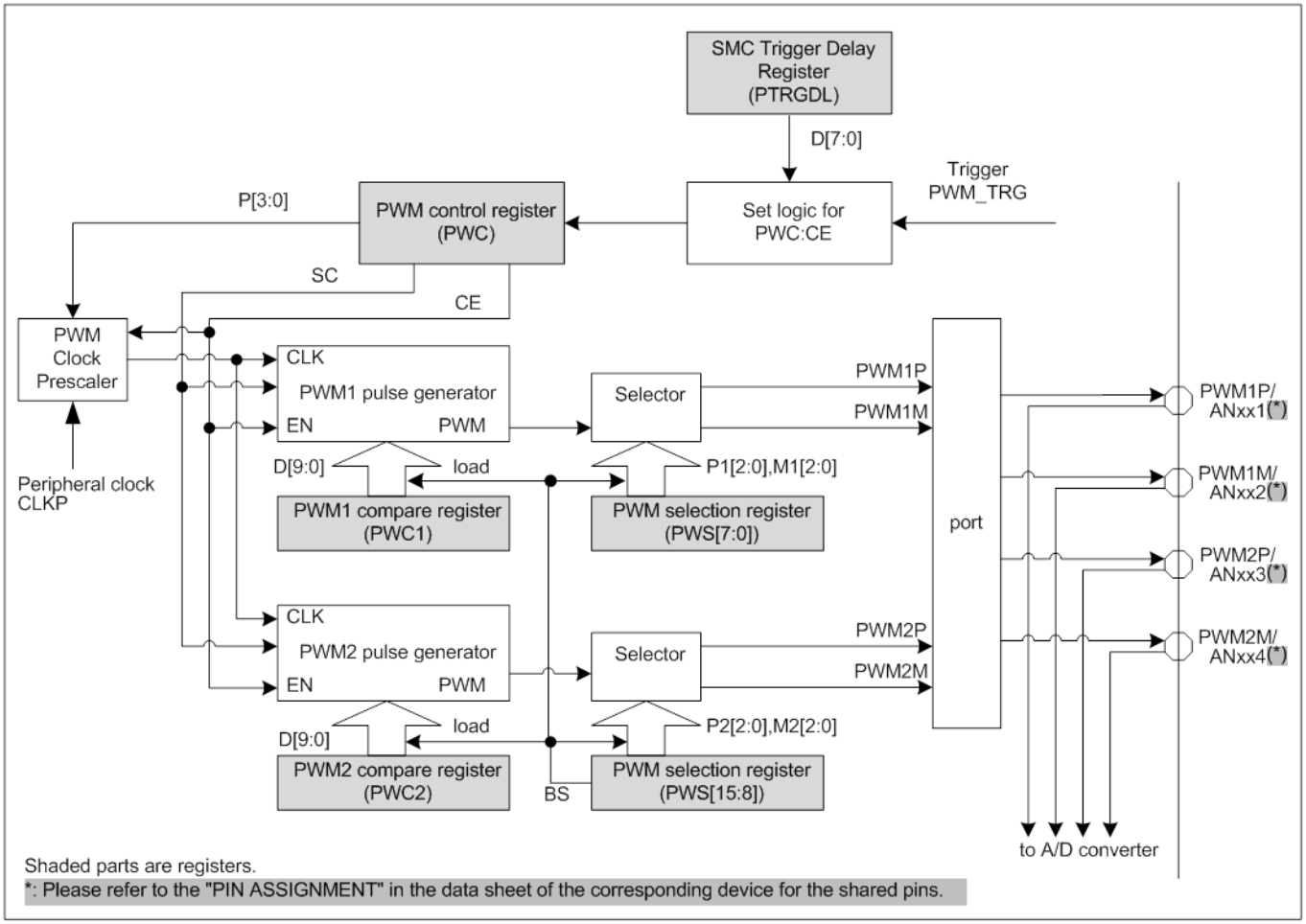
■ **Features of the Stepper Motor Controller:**

- Four high-current output drivers: Stepper Motor Controller provides four driver outputs with high current driving capability. Two motor coils can be connected directly to four stepper motor controller pins. The motor rotation is controlled by a combination of the PWM Pulse Generator and selector logic circuits.
- Two PWM Pulse Generators with 10-bit/8-bit operation mode (selected by software):
  - For the PWM pulse width in 8-bit operation mode, the duty can be specified within the range from 0% to 99.6%.
  - For the PWM pulse width in 10-bit operation mode, the duty can be specified within the range from 0% to 99.9%.
- A synchronization mechanism that enables synchronous operation of PWM Pulse Generators.
- A variable operating Clock Prescaler (the division ratio from a Peripheral clock can be selected by software).
- Output selector logic ('High', 'Low', PWM pulse and 'Hi-Z' signal levels can be selected).

2. Configuration and Block Diagram

This section shows block diagrams of the Stepper Motor Controller.

Figure 16-1 Block Diagram of the Stepper Motor Controller



PWM Clock Prescaler

The PWM Clock Prescaler divides the Peripheral clock (CLKP) and generates the PWM operating clock. The PWM operating clock is used by the PWM Pulse Generators.

PWM1 Pulse Generator / PWM2 Pulse Generator

The PWM Pulse Generators generate the pulse-width modulated signals according to the waveform configuration and the selected operation mode.

Selector

This sub-module, based on configuration, selects the level of the signal that will be sent to the Stepper Motor Controller output pins.

Port

The Port controls the I/O port for the Stepper Motor Controller pins.

PWM Control Register (PWC)

The PWM Control Register (PWC) starts/stops the Stepper Motor Controller, sets the PWM Prescaler and selects the operation mode for the PWM Pulse Generators.

PWM1 and PWM2 Compare Registers (PWC1, PWC2)

The PWM1 and PWM2 Compare Registers determine the PWM pulse width.

PWM Selection Register (PWS)

The PWM Selection Register selects the signal levels (‘H’ and ‘L’, PWM pulse and ‘Hi-Z’) to be output to the Stepper Motor Controller pins. It also sets the update instruction for the system to synchronize the PWM Pulse Generators.

SMC Trigger Delay Register (PTRGDL)

The SMC Trigger Delay Register (PTRGDL) is used to delay the start of the operation by 0 to 255 clock cycles.

Set Logic for PWC.CE

The Set logic for PWC.CE receives the trigger and waits for a configured delay time and sets automatically PWC.CE bit.

3. Operation of the Stepper Motor Controller

This section describes the operation of the Stepper Motor Controller.

3.1 Operation of PWM Pulse Generator
3.2 Selection of Motor Drive Signals
3.3 Synchronization System of Stepper Motor Controller
3.4 Operation of the SMC Trigger Delay Function

3.1. Operation of PWM Pulse Generator

The PWM period is determined by the selected division ratio from a Peripheral clock (CLKP) and the operation mode (8-bit/10-bit) of the PWM Pulse Generator:

- During configuration, software can select the PWM operating clock by specifying the division ratio for the Peripheral clock (CLKP). The division ratio for the Peripheral clock is defined by setting the PWM operating clock prescaler bits (PWC.P[3:0]).
- Software can also configure the operation mode for the counter that controls the PWM period. When the Operation Mode Switching bit (PWC.SC) is set to ‘0’, 8-bit operation mode is selected (PWM period = 256 counts). If this bit is set to ‘1’, 10-bit operation mode is selected (PWM period = 1024 counts).

Examples of operating clock values for different configurations are given in Table 16-1.

Table 16-1 Relation between PWM Operating Clock and PWM Period for Different Operation Modes

PWC P[3:0]	PWM Operating Clock (CLK)	PWM Period [µs] (CLKP = 16 MHz)		PWM Period [µs] (CLKP = 60 MHz)	
		PWC SC = ‘0’	PWC SC = ‘1’	PWC SC = ‘0’	PWC SC = ‘1’
0000	CLKP	16	64	4.267	17.067
0001	CLKP/4	64	256	17.067	68.267
0010	CLKP/5	80	320	21.333	85.333
0011	CLKP/6	96	384	25.6	102.4
0100	CLKP/8	128	512	34.133	136.533
0101	CLKP/10	160	640	42.667	170.667
0110	CLKP/12	192	768	51.2	204.8
0111	CLKP/16	256	1024	68.267	273.067
1000	CLKP/2	32	128	8.533	34.133
1001	CLKP/8	128	512	34.133	136.533
1010	CLKP/10	160	640	42.667	170.667
1011	CLKP/12	192	768	51.2	204.8
1100	CLKP/16	256	1024	68.267	273.067
1101	CLKP/20	320	1280	85.333	341.333
1110	CLKP/24	384	1536	102.4	409.6
1111	CLKP/32	512	2048	136.533	546.133

CLKP: Peripheral clock

The PWM pulse width (duty cycle) is determined by the PWM Compare Data Value (PWC1.D[9:0], PWC2.D[9:0]). In 8-bit operation mode (PWC.SC = ‘0’), the PWM Compare Data Value is taken from the PWC1.D[7:0], PWC2.D[7:0] bits and bits PWC1.D[9:8], PWC2.D[9:8] are 'don’t care'. In 10-bit operation mode (PWC.SC = ‘1’), the PWM Compare Data Value is taken from the PWC1.D[9:0], PWC2.D[9:0] bits. Below are the ranges in which PWM Compare Data Values (PWC1.D[9:0], PWC2.D[9:0]) can be set:

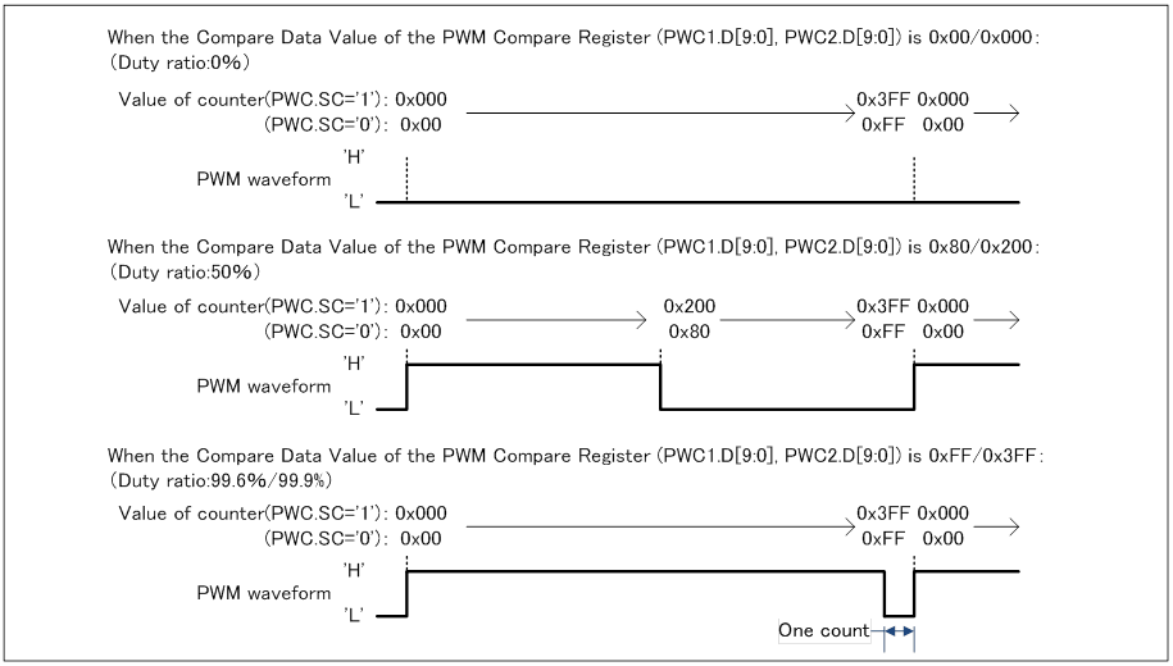
- In 8-bit operation mode (PWC.SC = ‘0’) 0% to 99.6% (0x00 to 0xFF)
- In 10-bit operation mode (PWC.SC = ‘1’) 0% to 99.9% (0x000 to 0x3FF)

The PWM Pulse Generator is started/stopped using the Count Enable bit (PWC.CE). When this bit is set to ‘1’, the PWM Pulse Generator starts the counter that controls the PWM period. The counter

increments its value from 0x00 (0x000) at the rising edge of the PWM operating clock. The PWM Pulse Generator output remains at the 'H' level until the counter value matches the PWM Compare Data Value. Once the PWM Compare Data Value is reached, the PWM Pulse Generator output changes to 'L' and remains at the 'L' level until the counter value overflows: 0xFF → 0x00 (0x3FF → 0x000).

Figure 16-2 shows the PWM waveforms generated by the PWM Pulse Generator.

Figure 16-2 Examples of PWM1 and PWM2 Output Waveforms



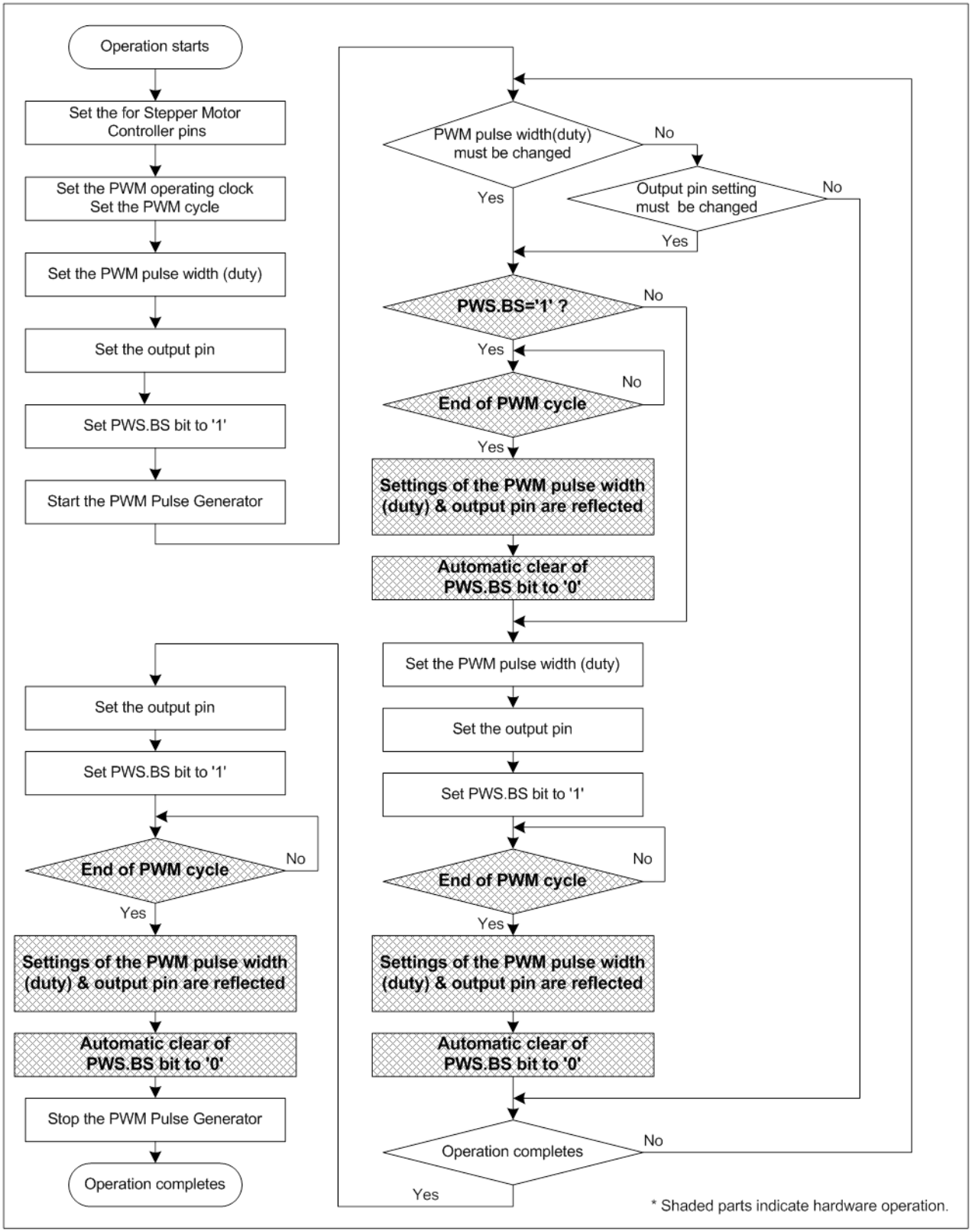
If the PWM Pulse Generator is stopped (PWC.CE = '0'), the setting of the PWM pulse width (duty cycle) and the output pin selection settings are not reflected on the Stepper Motor Controller outputs.

Notes:

- If the PWM Pulse Generator is started(PWC.CE='1'), operate following steps because the PWM Pulse Generator.Compare Data Value must be initialized.
  1. Set the PWM Compare Register.Compare Data Value(PWC1.D[9:0], PWC2.D[9:0])
  2. Set the PWM Selection Register.Output Selection bits(PWS.BS='1', PWS.P2[2:0], PWS.M2[2:0], PWS.P1[2:0], PWS.M1[2:0])
  3. Set the PWM Control Register.Count Enable bit(PWC.CE='1')
- If the PWM Pulse Generator is stopped when the Output Update bit (PWS.BS) is '1', the PWS.BS bit will retain the value '1' until the PWM Pulse Generator is started and the beginning of the new PWM cycle is detected. As a result, the setting of the PWM pulse width (duty cycle) and the output pin selection settings are reflected on the Stepper Motor Controller outputs. To avoid this scenario, clear PWC.CE only after PWS.BS is automatically cleared by the SMC.
- If the PWS.BS bit is '1', keep the following registers value because the setting of the PWM pulse width (duty cycle) and the output pin selection settings are not reflected at the same time.
  - PWM1 Compare Register.Compare Data Value(PWC1.D[9:0])
  - PWM2 Compare Register.Compare Data Value(PWC2.D[9:0])
  - PWM Selection Register.Output Selection bits(PWS.P2[2:0], PWS.M2[2:0], PWS.P1[2:0], PWS.M1[2:0])

An example for the Stepper Motor Controller setting procedure is given on the Figure 16-3.

Figure 16-3 Example of the Setting Procedure for the Stepper Motor Controller



3.2. Selection of Motor Drive Signals

The user can select the signal at each output pin of the Stepper Motor Controller according to the Table 16-2 and Table 16-3. The selection of output pin is done by configuring the PWM Selection Register (PWS.P2[2:0], PWS.M2[2:0], PWS.P1[2:0], PWS.M1[2:0]).

Table 16-2 Selection of Motor Drive Signals and Setting of PWM1 Selection Bits

PWS.P1[2:0]	PWM1P
000	L
001	H
01X	PWM pulse
1XX	Hi-Z

PWS.M1[2:0]	PWM1M
000	L
001	H
01X	PWM pulse
1XX	Hi-Z

X: don’t care

Table 16-3 Selection of Motor Drive Signals and Setting of PWM2 Selection Bits

PWS.P2[2:0]	PWM2P
000	L
001	H
01X	PWM pulse
1XX	Hi-Z

PWS.M2[2:0]	PWM2M
000	L
001	H
01X	PWM pulse
1XX	Hi-Z

X: don’t care

3.3. Synchronization System of Stepper Motor Controller

The Stepper Motor Controller contains a dedicated system developed to enable synchronization of PWM1 and PWM2 Pulse Generators, as well as Stepper Motor Controller output pin selection. The synchronization system is controlled by the PWM Selection Register Output Update bit (PWS.BS). Until the Output Update bit (PWS.BS) is set to '1', changes made to the PWM Compare Registers (PWC1, PWC2) and the PWM Selection Register (PWS) are not reflected on the output. Configuring the Stepper Motor Controller output pin selection and setting the Output Update bit (PWS.BS) simultaneously (in the same software access) is allowed.

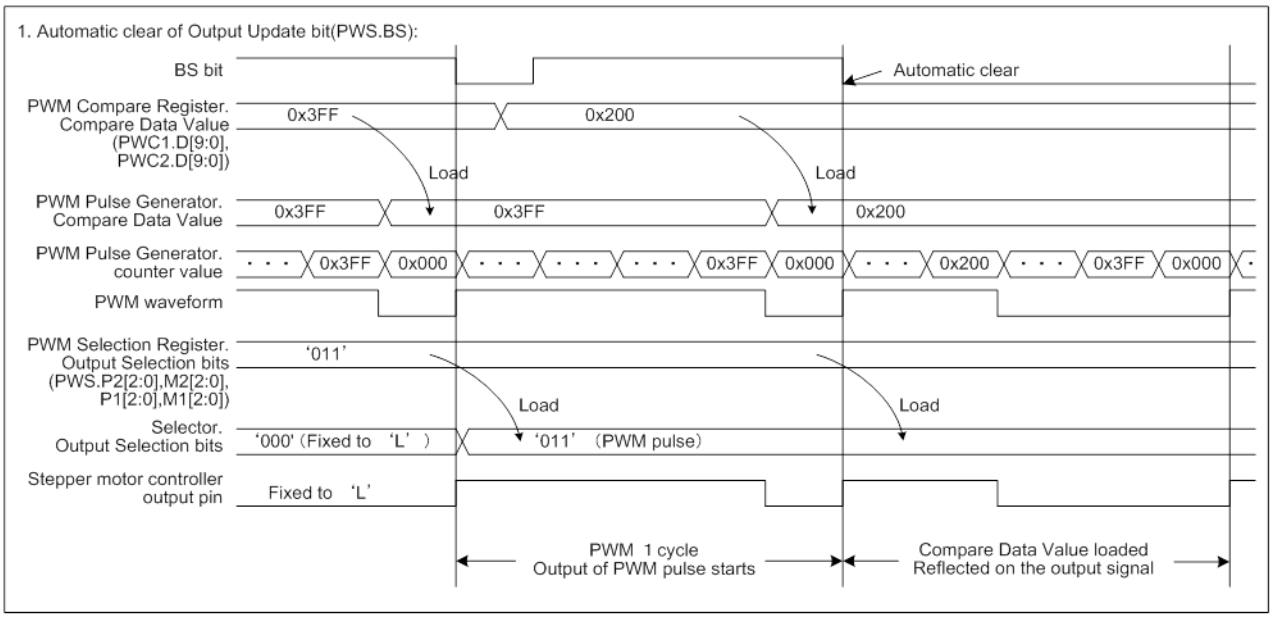
When the Output Update bit (PWS.BS) is set to '1' by software, and the end of current PWM cycle is detected, PWM Pulse Generator and motor drive selection logic will load values of the corresponding registers. The Output Update bit (PWS.BS) is automatically cleared to '0' at the beginning of the next PWM cycle.

If the Output Update bit (PWS.BS) is equal to '1' and the software access is initiated to set this bit to '1' at the beginning of the PWM cycle (at the same time when an automatic clear of Output Update bit (PWS.BS) is issued), the Output Update bit (PWS.BS) will retain the value '1' (i.e. no change is made to the PWS.BS bit) and the automatic clearing is canceled for this PWM cycle.

The following scenario is shown on Figure 16-4:

1. Automatic clear of the Output Update bit (PWS.BS): load operation is executed and reflected on the output.

Figure 16-4 Load Timing of PWM Compare Register and PWM Selection Register



3.4. Operation of the SMC Trigger Delay Function

After the Stepper Motor Controller receives a trigger input (PWM_TRG), it waits for a configured delay time to elapse before the Count Enable bit (PWC.CE) is automatically set and the Stepper Motor Controller starts operating. The delay time is configured in the SMC Trigger Delay Register (PTRGDL). The Stepper Motor Controller will continue to operate until the Count Enable bit (PWC.CE) is reset by software.

Figure 16-5 demonstrates the software flow for receiving a trigger input (PWM_TRG). An example, illustrating how the trigger starts Stepper Motor Controller, is shown on Figure 16-6.

Note:

- The trigger should only be received when the Stepper Motor Controller are not running (i.e. the Count Enable bit (PWC.CE) is cleared to '0').

Figure 16-5 Flowchart for Triggering Stepper Motor Controller

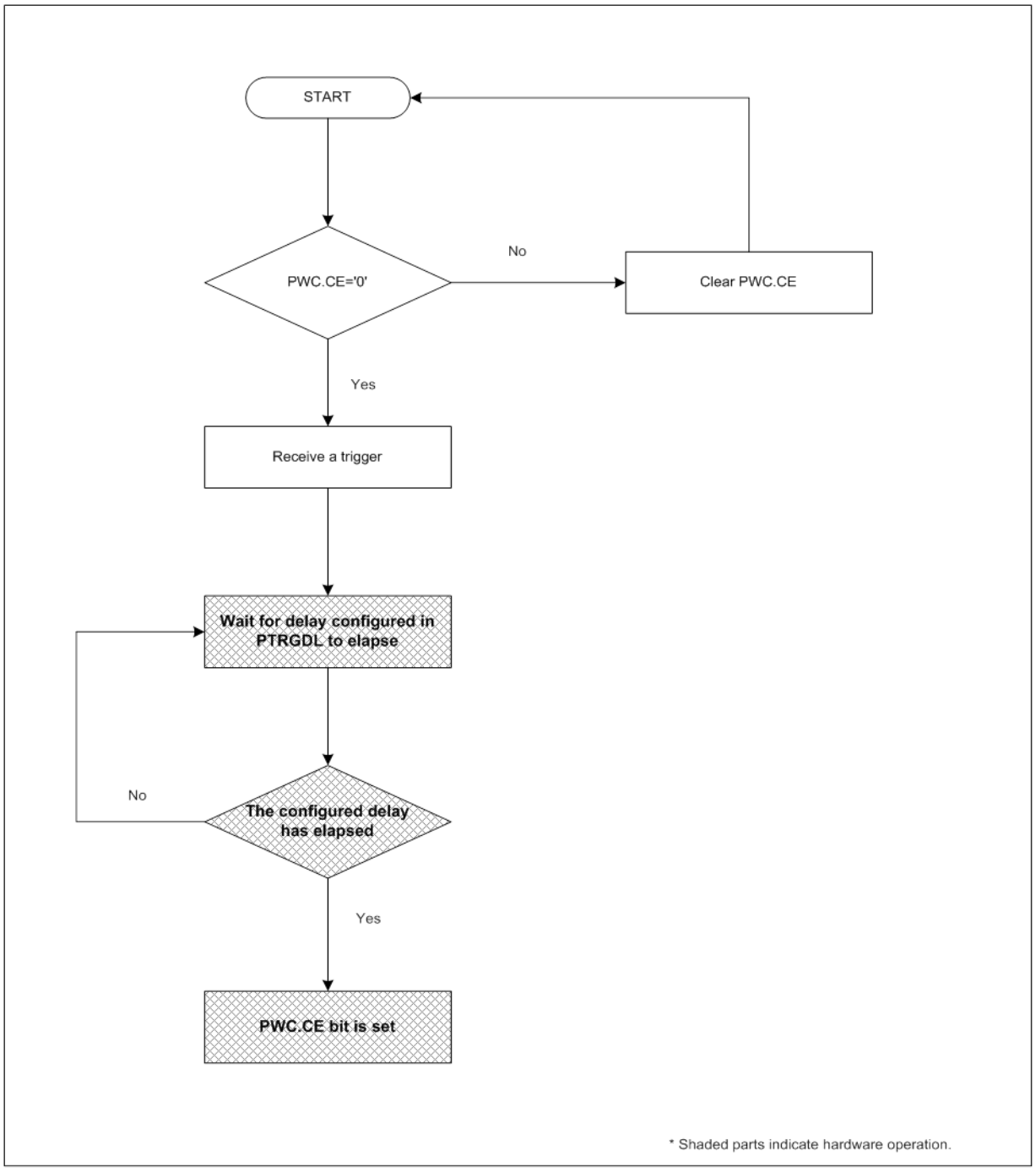


Figure 16-6 Timing Diagram for Triggering of Stepper Motor Controller

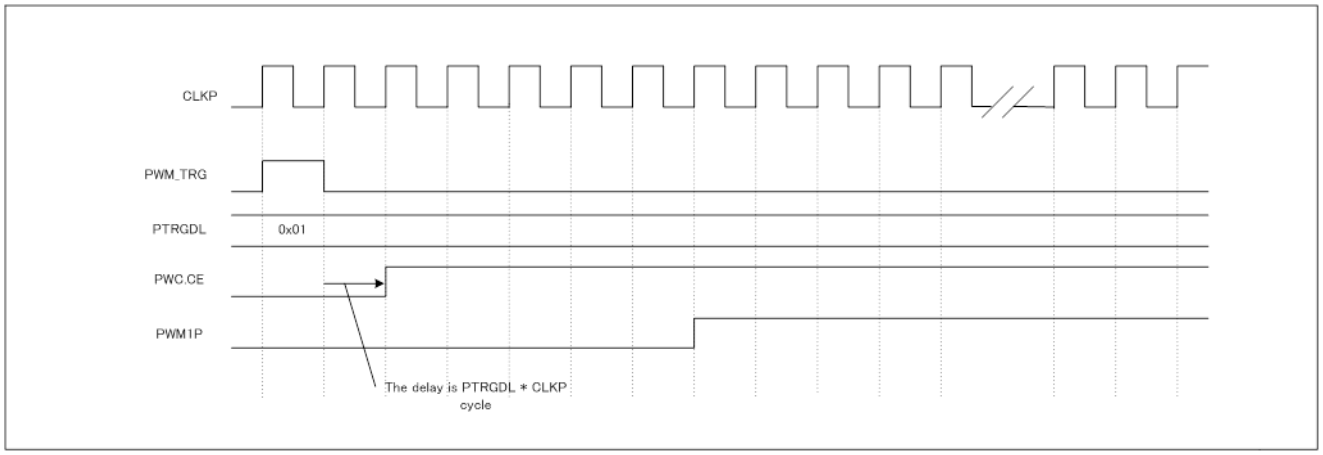


Figure 16-6 shows how the trigger starts Stepper Motor Controller (PTRGDL=0x01). If the SMC Trigger Delay Register (PTRGDL) setting is 0x00, PWC.CE is set immediately after receiving the trigger input (PWM_TRG). If the SMC Trigger Delay Register (PTRGDL) setting is 0x01, PWC.CE is set with a delay of one clock cycle after receiving the trigger input (PWM_TRG).

4. Registers

This section describes the registers of the Stepper Motor Controller.

■ **The Following Registers are Available for One Stepper Motor Controller:**

PWM Control Register (PWC)

PWM1 and PWM2 Compare Registers (PWC1, PWC2)

PWM Selection Register (PWS)

PWM Selection Set Register (PWSS)

SMC Trigger Delay Register (PTRGDL)

Table 16-4 List of Registers for the Stepper Motor Controller

Abbreviation	Register Name	Reference
PWC	PWM Control Register	4.1
PWC1	PWM1 Compare Register	4.2.1
PWC2	PWM2 Compare Register	4.2.2
PWS	PWM Selection Register	4.3
PWSS	PWM Selection Set Register	4.4
PTRGDL	SMC Trigger Delay Register	4.5

4.1. PWM Control Register (PWC)

The PWM Control Register starts/stops the Stepper Motor Controller, sets the PWM prescaler and selects the operation mode for the generation of the PWM pulse.

REGISTER_NAME	SMCi_PWC
OFFSET	0x00000000
ACCESS_SIZE	B, H
MULTIPLE	1
NUMERIC_TYPE	-
OTHER	-

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved						P[3]	Reserved
ACCESS_TYPE	R0,W0						R/W	R0/W0
PROT_TYPE	Rp/Wp							
INITIAL_VALUE	0						0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved	P[2:0]			CE	SC	Reserved	Reserved
ACCESS_TYPE	R0/W0	R/W			R,W	R/W	R0,W0	R0/W0
PROT_TYPE	Rp/Wp							
INITIAL_VALUE	0	0			0	0	0	0

[bit15:10] Reserved: Reserved bits

When writing, always write '0'.

When reading, '0' is always read.

[bit9] P[3]: PWM Operating Clock Prescaler bit

P[3] bit, in combination with P[2:0] bits, specifies the clock input signal to the PWM Pulse Generators.

[bit8:7] Reserved: Reserved bits

When writing, always write '0'.

When reading, '0' is always read.

[bit6:4] P[2:0]: PWM Operating Clock Prescaler bits

P[2:0] bits, in combination with P[3] bit, specify the clock input signal to the PWM Pulse Generators.

P[3]	P[2:0]	Description
0	000	PWM Operating clock is equal to the Peripheral clock (CLKP)
0	001	PWM Operating clock is equal to CLKP/4
0	010	PWM Operating clock is equal to CLKP/5
0	011	PWM Operating clock is equal to CLKP/6
0	100	PWM Operating clock is equal to CLKP/8
0	101	PWM Operating clock is equal to CLKP/10
0	110	PWM Operating clock is equal to CLKP/12
0	111	PWM Operating clock is equal to CLKP/16
1	000	PWM Operating clock is equal to CLKP/2
1	001	PWM Operating clock is equal to CLKP/8
1	010	PWM Operating clock is equal to CLKP/10
1	011	PWM Operating clock is equal to CLKP/12
1	100	PWM Operating clock is equal to CLKP/16
1	101	PWM Operating clock is equal to CLKP/20
1	110	PWM Operating clock is equal to CLKP/24
1	111	PWM Operating clock is equal to CLKP/32

CLKP: Peripheral clock

Note:

- *Configuration of the PWM Operating Clock Prescaler bits (P[3:0]) shall be done while counting operation is disabled (PWC.CE = '0').*

[bit3] CE: Count Enable bit

The CE bit enables operation of the PWM Pulse Generators. When CE bit is set to '1', the PWM Pulse Generator starts its operating. The PWM2 Pulse Generator starts with the delay of one Peripheral clock cycle (CLKP) after the PWM1 Pulse Generator is started.

The CE bit can be set and cleared by the software. This bit also can be set to '1' by receiving trigger. To receive trigger, the CE bit must first be cleared to '0'.

Bit	Description
0	Initializes and stops the operation of the PWM Pulse Generators
1	Starts the operation of the PWM Pulse Generators

Notes:

- If the PWM Pulse Generator is started(CE='1'), operate following steps because the PWM Pulse Generator.Compare Data Value must be initialized.
  1. Set the PWM Compare Register.Compare Data Value(PWC1.D[9:0], PWC2.D[9:0])
  2. Set the PWM Selection Register.Output Selection bits(PWS.BS='1', P2[2:0], M2[2:0], P1[2:0], M1[2:0])
  3. Set the PWM Control Register.Count Enable bit(CE='1')
- If the PWM Pulse Generator is stopped (CE='0') when the PWM pulse is selected by the Output Selection bits (PWS.P2[2:0], M2[2:0], P1[2:0], M1[2:0]), the pin for which the PWM pulse is selected is fixed to the 'L' level.
- If the trigger input is used, write '0' to the CE bit only when the CE bit is '1'. Because the CE bit can be cleared at the same time when the CE bit is set by the trigger.
- The CE bit shall be set to '1' after the setting of the PWM Operating Clock Prescaler bits (PWC.P[3:0]) and the Operation Mode Switching bit (PWC.SC) is completed.

[bit2] SC: Operation Mode Switching bit

This bit specifies the operation mode of the PWM Pulse Generators

bit	Description
0	8-bit operation mode is selected
1	10-bit operation mode is selected

Note:

- The PWC.CE bit shall be set to '1' after the setting of the PWM Operating Clock Prescaler bits (PWC.P[3:0]) and the Operation Mode Switching bit (SC) is completed.

[bit1:0] Reserved: Reserved bits

When writing, always write '0'.

When reading, '0' is always read.

4.2. PWM1 and PWM2 Compare Registers (PWC1, PWC2)

The value of the 8/10 bit PWM1 and PWM2 Compare Registers determines the width of the PWM pulse. A value of 0x00 (0x000) indicates that the PWM duty cycle is 0%, while a value of 0xFF (0x3FF) indicates that the PWM duty cycle is 99.6% (99.9%).

4.2.1. PWM1 Compare Register(PWC1)

REGISTER_NAME	SMCi_PWC1
OFFSET	0x00000002
ACCESS_SIZE	B, H
MULTIPLE	1
NUMERIC_TYPE	-
OTHER	-

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved						D[9:8]	
ACCESS_TYPE	R0,W0						R/W	
PROT_TYPE	Rp/Wp							
INITIAL_VALUE	0						X	

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	D[7:0]							
ACCESS_TYPE	R/W							
PROT_TYPE	Rp/Wp							
INITIAL_VALUE	X							

[bit15:10] Reserved: Reserved bits

When writing, always write '0'.

When reading, '0' is always read.

[bit9:0] D[9:0]: PWM1 Compare Data Value

The PWM1 Compare Data Value D[9:0] are used to set the PWM1 duty cycle. When the Operation Mode Switching bit is set to 8-bit operation mode (PWC.SC= '0'), the compare data value is taken from D[7:0] bits and D[9:8] bits are 'don't care'. When the Operation Mode Switching bit is set to 10-bit operation mode (PWC.SC= '1'), the compare data value is taken from D[9:0] bits.

D[9:0]	Description
0x(0)00	0%
…	Set the PWM pulse width
0x(3)FF	(99.9%) 99.6%

Notes:

- PWM Pulse Generator.Compare Data Value is updated at the end of PWM cycle, after PWS.BS bit is set to '1'.
- When PWC.SC bit is set to '0' (PWM Pulse Generators are operating in 8-bit mode), D[9:8], PWC2.D[9:8] bits are 'don't care'.
- The software must always make a 16-bit write access to PWM1 and PWM2 Compare Registers (PWC1, PWC2) to ensure consistency of data. However, 8-bit write access to PWM1 or PWM2 Compare Register (PWC1, PWC2) is possible and will not result in error response.
- If the PWS.BS bit is '1' , don't change the PWM Compare Register.Compare Data Value(D[9:0], PWC2.D[9:0]) and the PWM Selection Register.Output Selection bits (PWS.P2[2:0], PWS.M2[2:0], PWS.P1[2:0], PWS.M1[2:0]).

4.2.2. PWM2 Compare Register(PWC2)

REGISTER_NAME	SMCi_PWC2
OFFSET	0x00000004
ACCESS_SIZE	B, H
MULTIPLE	1
NUMERIC_TYPE	-
OTHER	-

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved						D[9:8]	
ACCESS_TYPE	R0,W0						R/W	
PROT_TYPE	Rp/Wp							
INITIAL_VALUE	0						X	

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	D[7:0]							
ACCESS_TYPE	R/W							
PROT_TYPE	Rp/Wp							
INITIAL_VALUE	X							

[bit15:10] Reserved: Reserved bits

When writing, always write '0'.

When reading, '0' is always read.

[bit9:0] D[9:0]: PWM2 Compare Data Value

The PWM2 Compare Data Value D[9:0] bits are used to set the PWM2 duty cycle. When the Operation Mode Switching bit is set to 8-bit operation mode (PWC.SC= '0'), the compare data value is taken from D[7:0] bits and D[9:8] bits are 'don't care'. When the Operation Mode Switching bit is set to 10-bit operation mode (PWC.SC= '1'), the compare data value is taken from D[9:0] bits.

D[9:0]	Description
0x(0)00	0%
…	Set the PWM pulse width
0x(3)FF	(99.9%) 99.6%

Notes:

- PWM Pulse Generator.Compare Data Value is updated at the end of PWM cycle, after PWS.BS bit is set to '1'.
- When PWC.SC bit is set to '0' (PWM Pulse Generators are operating in 8-bit mode), PWC1.D[9:8], D[9:8] bits are 'don't care'.
- The software must always make a 16-bit write access to PWM1 and PWM2 Compare Registers (PWC1, PWC2) to ensure consistency of data. However, 8-bit write access to PWM1 or PWM2 Compare Register (PWC1, PWC2) is possible and will not result in error response.
- If the PWS.BS bit is '1' , don't change the PWM Compare Register.Compare Data Value(PWC1.D[9:0], D[9:0]) and the PWM Selection Register.Output Selection bits (PWS.P2[2:0], PWS.M2[2:0], PWS.P1[2:0], PWS.M1[2:0]).

4.3. PWM Selection Register (PWS)

The PWM Selection Register (PWS) sets the output state of the Stepper Motor Controller pins (‘L’, ‘H’, PWM pulse or ‘Hi-Z’) and allows the update of the PWM duty cycle or output pin states.

REGISTER_NAME	SMCi_PWS
OFFSET	0x00000006
ACCESS_SIZE	B, H
MULTIPLE	1
NUMERIC_TYPE	-
OTHER	-

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved	BS	P2[2:0]			M2[2:0]		
ACCESS_TYPE	R0,W0	R/W	R/W			R/W		
PROT_TYPE	Rp/Wp							
INITIAL_VALUE	0	0	0			0		

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved		P1[2:0]			M1[2:0]		
ACCESS_TYPE	R0/W0		R/W			R/W		
PROT_TYPE	Rp/Wp							
INITIAL_VALUE	0		0			0		

[bit15] Reserved: Reserved bit

When writing, always write '0'.

When reading, '0' is always read.

[bit14] BS: Output Update bit

The Output Update bit (BS) synchronously updates the Selector.Output Selection bits and PWM Pulse Generator.Compare Data Value (See Figure 16-4). To change the settings for the PWM duty cycle or output pin states, the BS bit must be set to ‘1’ after (or at the same time) a new configuration is written to the PWM1 and PWM2 Compare Registers (PWC1, PWC2) and PWM Selection Register (PWS). If the BS bit is set to ‘1’ while PWM Pulse Generators are operating, the new settings are loaded at the end of the current PWM cycle. The BS bit is cleared automatically at the beginning of the following PWM cycle. As long as BS bit is '0', the PWM duty cycle and output pin states are not changed.

This bit can also be set by the PWSS.BSS bit.

Bit	Description
0	Disables the update of the setting for the PWM output
1	Enables the update of the setting for the PWM output

Notes:

- If the PWM Pulse Generator is started(PWC.CE='1'), operate following steps because the PWM Pulse Generator.Compare Data Value must be initialized.
  4. Set the PWM Compare Register.Compare Data Value(PWC1.D[9:0], PWC2.D[9:0])
  5. Set the PWM Selection Register.Output Selection bits(BS='1', PWS.P2[2:0], PWS.M2[2:0], PWS.P1[2:0], PWS.M1[2:0])
  6. Set the PWM Control Register:Count Enable bit(PWC.CE='1')
- If the BS bit is '1' , don't change the PWM Compare Register.Compare Data Value(PWC1.D[9:0], PWC2.D[9:0]) and the PWM Selection Register.Output Selection bits (PWS.P2[2:0], PWS.M2[2:0], PWS.P1[2:0], PWS.M1[2:0]).

[bit13:11] P2[2:0]: Plus Output 2 Selection bits

The P2[2:0] bits select the output signal for PWM2P.

P2[2:0]	Description
000	'L': PWM2P output is fixed to low level
001	'H': PWM2P output is fixed to high level
01X	PWM pulse is driven on the PWM2P output
1XX	'Hi-Z': PWM2P output is set to high impedance

X: don't care

Note:
− If the PWS.BS bit is '1' , don't change the PWM Compare Register.Compare Data Value(PWC1.D[9:0], PWC2.D[9:0]) and the PWM Selection Register.Output Selection bits (P2[2:0], PWS.M2[2:0], PWS.P1[2:0], PWS.M1[2:0]).

[bit10:8] M2[2:0]: Minus Output 2 Selection bits

The M2[2:0] bits selects the output signal for PWM2M.

M2[2:0]	Description
000	'L': PWM2M output is fixed to low level
001	'H': PWM2M output is fixed to high level
01X	PWM pulse is driven on the PWM2M output
1XX	'Hi-Z': PWM2M output is set to high impedance

X: don't care

Note:
- *If the PWS.BS bit is '1' , don't change the PWM Compare Register.Compare Data Value(PWC1.D[9:0], PWC2.D[9:0]) and the PWM Selection Register.Output Selection bits (PWS.P2[2:0], M2[2:0], PWS.P1[2:0], PWS.M1[2:0]).*

[bit7:6] Reserved: Reserved bit

When writing, always write '0'.

When reading, '0' is always read.

[bit5:3] P1[2:0]: Plus Output 1 Selection bits

The P1[2:0] bits select the output signal for PWM1P.

P1[2:0]	Description
000	'L': PWM1P output is fixed to low level
001	'H': PWM1P output is fixed to high level
01X	PWM pulse is driven on the PWM1P output
1XX	'Hi-Z': PWM1P output is set to high impedance

X: don't care

Note:
− If the PWS.BS bit is '1' , don't change the PWM Compare Register.Compare Data Value(PWC1.D[9.0], PWC2.D[9:0]) and the PWM Selection Register.Output Selection bits (PWS.P2[2:0], PWS.M2[2:0], P1[2:0], PWS.M1[2:0]).

[bit2:0] M1[2:0]: Minus Output 1 Selection bits

The M1[2:0] bits selects the output signal for PWM1M.

M1[2:0]	Description
000	'L': PWM1M output is fixed to low level
001	'H': PWM1M output is fixed to high level
01X	PWM pulse is driven on the PWM1M output
1XX	'Hi-Z': PWM1M output is set to high impedance

X: don’t care

Note:

- *If the PWS.BS bit is '1' , don't change the PWM Compare Register.Compare Data Value(PWC1.D[9:0], PWC2.D[9:0]) and the PWM Selection Register.Output Selection bits (PWS.P2[2:0], PWS.M2[2:0], PWS.P1[2:0], M1[2:0]).*

4.4. PWM Selection Set Register (PWSS)

The PWM Selection Set Register is used to set the value of the PWS.BS bit.

REGISTER_NAME	SMCi_PWSS
OFFSET	0x00000008
ACCESS_SIZE	B,H
MULTIPLE	1
NUMERIC_TYPE	-
OTHER	-

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved	BSS	Reserved					
ACCESS_TYPE	R0,W0	R0,W	R0,W0					
PROT_TYPE	Rp/Wp							
INITIAL_VALUE	0	0	0					

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved							
ACCESS_TYPE	R0,W0							
PROT_TYPE	Rp/Wp							
INITIAL_VALUE	0							

[bit15] Reserved: Reserved bit

When writing, always write '0'.

When reading, '0' is always read.

[bit14] BSS: Set bit for the Output Update bit

This bit sets the value of PWS.BS bit.

Bit	Description
0	No effect
1	Sets the PWS.BS bit to '1'. Reading of this bit returns '0'.

Notes:

- If the PWM Pulse Generators is started(PWC.CE='1'), operate following steps because the PWM Pulse Generator.Compare Data Value must be initialized.
  7. Set the PWM Compare Register.Compare Data Value(PWC1.D[9:0], PWC2.D[9:0])
  8. Set the PWM Selection Register.Output Selection bits(PWS.BS='1', PWS.P2[2:0], PWS.M2[2:0], PWS.P1[2:0], PWS.M1[2:0])
  9. Set the PWM Control Register.Count Enable bit(PWC.CE='1')
- If the PWS.BS bit is '1' , don't change the PWM Compare Register.Compare Data Value(PWC1.D[9:0], PWC2.D[9:0]) and the PWM Selection Register.Output Selection bits (PWS.P2[2:0], PWS.M2[2:0], PWS.P1[2:0], PWS.M1[2:0]).

[bit13:0] Reserved: Reserved bits

When writing, always write '0'.

When reading, '0' is always read.

4.5. SMC Trigger Delay Register (PTRGDL)

The SMC Trigger Delay Register (PTRGDL) is used to delay the start of the operation by 0 to 255 clock cycles.

REGISTER_NAME	SMCi_PTRGDL
OFFSET	0x0000000A
ACCESS_SIZE	B, H
MULTIPLE	1
NUMERIC_TYPE	-
OTHER	-

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved							
ACCESS_TYPE	R0,W0							
PROT_TYPE	Rp/Wp							
INITIAL_VALUE	0							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	D[7:0]							
ACCESS_TYPE	R/W							
PROT_TYPE	Rp/Wp							
INITIAL_VALUE	0							

[bit15:8] Reserved: Reserved bits

When writing, always write '0'.

When reading, '0' is always read.

[bit7:0] D[7:0]: Trigger Delay bits

The Trigger Delay bits (D[7:0]) configure a delay of the trigger input that starts the PWM operation from 0 to 255 clock cycles.

D[7:0]	Description
0x00	No delay (delay disabled)
0x01	Delay of 1 clock cycle between the trigger and the clock cycle in which the PWM generation starts
…	…
0xFF	Delay of 255 clock cycles between the trigger and the clock cycle in which the PWM generation starts

CHAPTER 17: Trigger Configuration of Stepper Motor Controller

This chapter explains the trigger configuration of stepper motor controller.

1. Overview
2. Configuration and Block Diagram
3. Operation
4. Registers

CODE: SMCTRG-S6J3300-E1

1. Overview

The chapter describes the function and its operation of both SMC trigger selection register and SMC trigger register. See the chapter of Stepper motor controller with this chapter.

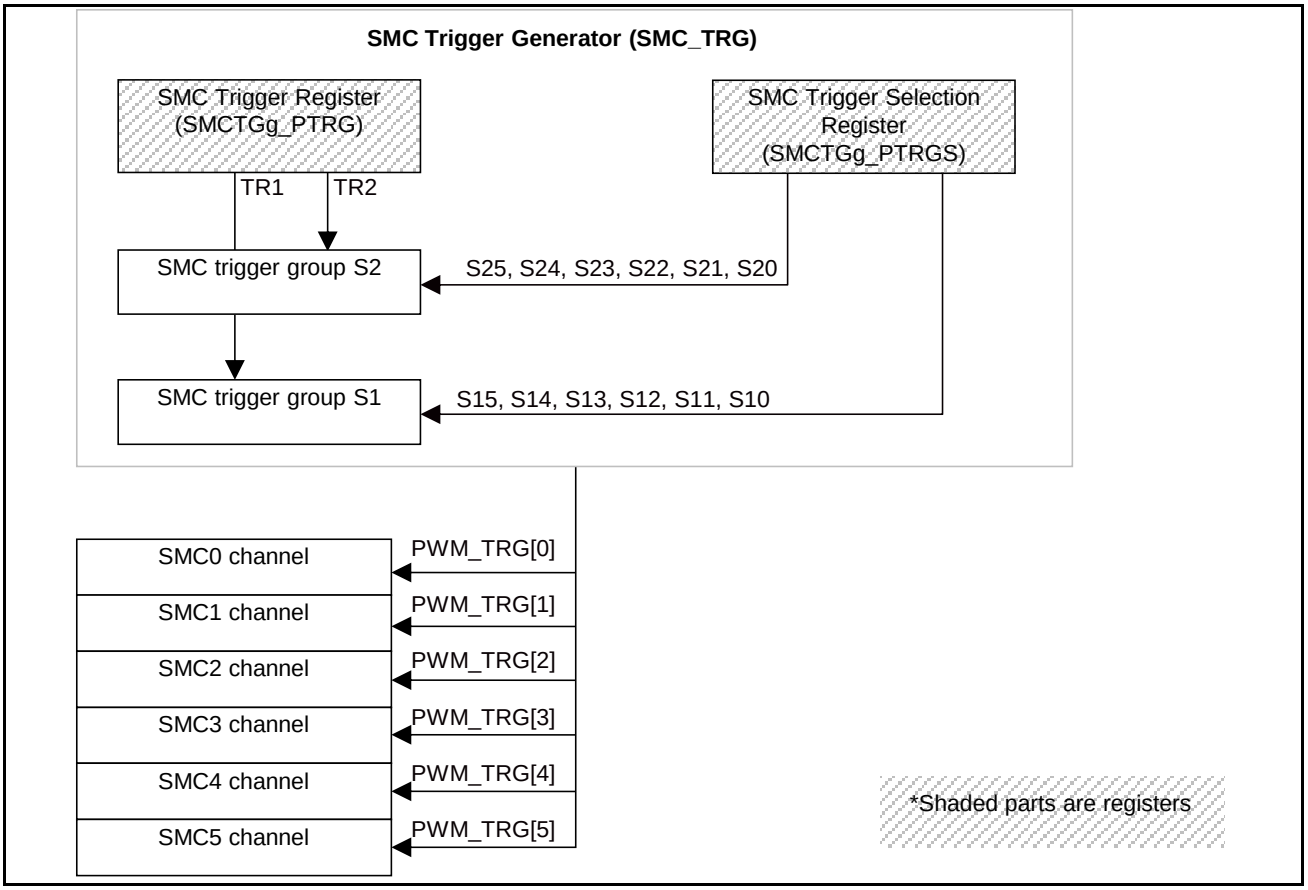
2. Configuration and Block Diagram

See the block diagram below.

Notes:

- *The suffix 'g' in the register name indicates that the register is an instance 'g' of the SMC Trigger Generator.*
- *The suffix 'i' in the register name indicates that the register is an instance 'i' of the SMC channel.*

Figure 17-1 Block Diagram of the SMC Module



3. Operation

The trigger configuration and its operation is described in 4.

4. Registers

Offset	Register Name
0x00000000	SMCTGg_PTRGS
0x00000002	SMCTGg_PTRG

4.1. SMC Trigger Selection Register (SMCTGg_PTRGS)

The SMC Trigger Selection Register (SMCTGg_PTRGS) selects multiple SMC channels that shall be triggered by the SMC Trigger Register to start operation synchronously or with a delay. The SMC channels can be combined into two trigger groups.

REGISTER_NAME	SMCTGg_PTRGS
OFFSET	0x00000000
ACCESS_SIZE	B, H
MULTIPLE	0
NUMERIC_TYPE	-
OTHER	-

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved		S25	S24	S23	S22	S21	S20
ACCESS_TYPE	R0,WX		R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	Rp/Wp							
INITIAL_VALUE	0		0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved		S15	S14	S13	S12	S11	S10
ACCESS_TYPE	R0,WX		R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	Rp/Wp							
INITIAL_VALUE	0		0	0	0	0	0	0

[bit13] S25 : Trigger Enable for Operation of SMC<6*g+5>

This bit selects SMC<6*g+5> for the SMC trigger group S2 (triggered by SMCTGg_PTRG:TR2).

S25	Description
0	SMC<6*g+5> is not triggered by SMCTGg_PTRG:TR2
1	SMC<6*g+5> is triggered by SMCTGg_PTRG:TR2

[bit12] S24 : Trigger Enable for Operation of SMC<6*g+4>

This bit selects SMC<6*g+4> for the SMC trigger group S2 (triggered by SMCTGg_PTRG:TR2).

S24	Description
0	SMC<6*g+4> is not triggered by SMCTGg_PTRG:TR2
1	SMC<6*g+4> is triggered by SMCTGg_PTRG:TR2

[bit11] S23 : Trigger Enable for Operation of SMC<6*g+3>

This bit selects SMC<6*g+3> for the SMC trigger group S2 (triggered by SMCTGg_PTRG:TR2).

S23	Description
0	SMC<6*g+3> is not triggered by SMCTGg_PTRG:TR2
1	SMC<6*g+3> is triggered by SMCTGg_PTRG:TR2

[bit10] S22 : Trigger Enable for Operation of SMC<6*g+2>

This bit selects SMC<6*g+2> for the SMC trigger group S2 (triggered by SMCTGg_PTRG:TR2).

S22	Description
0	SMC<6*g+2> is not triggered by SMCTGg_PTRG:TR2
1	SMC<6*g+2> is triggered by SMCTGg_PTRG:TR2

[bit9] S21 : Trigger Enable for Operation of SMC<6*g+1>

This bit selects SMC<6*g+1> for the SMC trigger group S2 (triggered by SMCTGg_PTRG:TR2).

S21	Description
0	SMC<6*g+1> is not triggered by SMCTGg_PTRG:TR2
1	SMC<6*g+1> is triggered by SMCTGg_PTRG:TR2

[bit8] S20 : Trigger Enable for Operation of SMC<6*g+0>

This bit selects SMC<6*g+0> for the SMC trigger group S2 (triggered by SMCTGg_PTRG:TR2).

S20	Description
0	SMC<6*g+0> is not triggered by SMCTGg_PTRG:TR2
1	SMC<6*g+0> is triggered by SMCTGg_PTRG:TR2

[bit5] S15 : Trigger Enable for Operation of SMC<6*g+5>

This bit selects SMC<6*g+5> for the SMC trigger group S1 (triggered by SMCTGg_PTRG:TR1).

S15	Description
0	SMC<6*g+5> is not triggered by SMCTGg_PTRG:TR1
1	SMC<6*g+5> is triggered by SMCTGg_PTRG:TR1

[bit4] S14 : Trigger Enable for Operation of SMC<6*g+4>

This bit selects SMC<6*g+4> for the SMC trigger group S1 (triggered by SMCTGg_PTRG:TR1).

S14	Description
0	SMC<6*g+4> is not triggered by SMCTGg_PTRG:TR1
1	SMC<6*g+4> is triggered by SMCTGg_PTRG:TR1

[bit3] S13 : Trigger Enable for Operation of SMC<6*g+3>

This bit selects SMC<6*g+3> for the SMC trigger group S1 (triggered by SMCTGg_PTRG:TR1).

S13	Description
0	SMC<6*g+3> is not triggered by SMCTGg_PTRG:TR1
1	SMC<6*g+3> is triggered by SMCTGg_PTRG:TR1

[bit2] S12 : Trigger Enable for Operation of SMC<6*g+2>

This bit selects SMC<6*g+2> for the SMC trigger group S1 (triggered by SMCTGg_PTRG:TR1).

S12	Description
0	SMC<6*g+2> is not triggered by SMCTGg_PTRG:TR1
1	SMC<6*g+2> is triggered by SMCTGg_PTRG:TR1

[bit1] S11 : Trigger Enable for Operation of SMC<6*g+1>

This bit selects SMC<6*g+1> for the SMC trigger group S1 (triggered by SMCTGg_PTRG:TR1).

S11	Description
0	SMC<6*g+1> is not triggered by SMCTGg_PTRG:TR1
1	SMC<6*g+1> is triggered by SMCTGg_PTRG:TR1

[bit0] S10 : Trigger Enable for Operation of SMC<6*g+0>

This bit selects SMC<6*g+0> for the SMC trigger group S1 (triggered by SMCTGg_PTRG:TR1).

S10	Description
0	SMC<6*g+0> is not triggered by SMCTGg_PTRG:TR1
1	SMC<6*g+0> is triggered by SMCTGg_PTRG:TR1

4.2. SMC Trigger Register (SMCTGg_PTRG)

The SMC Trigger Register is used to generate trigger signals for the two trigger groups configured in the SMC Trigger Selection Register (SMCTGg_PTRGS). The trigger group can be used to synchronously start several SMC channels. The operation (count enable) starts depending on the delay configured in the SMCi_PTRGDL register.

REGISTER_NAME	SMCTGg_PTRG
OFFSET	0x00000002
ACCESS_SIZE	B, H
MULTIPLE	0
NUMERIC_TYPE	-
OTHER	-

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Rp/Wp							
INITIAL_VALUE	0							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved						TR2	TR1
ACCESS_TYPE	R0,WX						R0,W	R0,W
PROT_TYPE	Rp/Wp							
INITIAL_VALUE	0						0	0

[bit1] TR2 : SMC Trigger 2

This bit triggers the SMC trigger group S2 of SMCTGg. The configuration in SMCTGg_PTRGS:S2x determines which SMC channels are triggered. The trigger signal starts the delay counters (if enabled in SMCi_PTRGDL). If the delay has elapsed or the delay is disabled, the Count Enable (SMCi_PWC:CE) bits of the selected SMC channels will be set and the operation starts (PWM generation and output control).

TR2	Description
0	No effect
1	Triggers selected SMC channels via the Set logic for SMCi_PWC:CE

Notes:

- *This bit is cleared automatically to '0' after one clock cycle.*
- *A second trigger should not be applied to a group which has already been triggered. If this happens, the running delay counter will reset its value.*

[bit0] TR1 : SMC Trigger 1

This bit triggers the SMC trigger group S1 of SMCTGg. The configuration in SMCTGg_PTRGS:S1x determines which SMC channels are triggered. The trigger signal starts the delay counters (if enabled in SMCi_PTRGDL). If the delay has elapsed or the delay is disabled, the Count Enable (SMCi_PWC:CE) bits of the selected SMC channels will be set and the operation starts (PWM generation and output control).

TR1	Description
0	No effect
1	Triggers selected SMC channels via the Set logic for SMCi_PWC:CE

Notes:

- *This bit is cleared automatically to '0' after one clock cycle.*
- *A second trigger should not be applied to a group which has already been triggered. If this happens, the running delay counter will reset its value.*

CHAPTER 18: Sound Generator

This chapter explains the functions and operations of the Sound Generator(SG).

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

CODE: FS12C-E03.02

1. Overview

The Sound Generator generates and outputs a tone pulse signal (or mixed signal of a tone pulse and a PWM pulse) and a PWM pulse signal according to the setting by MCU. It is enabled to set the frequency of the tone pulse signal, sound volume (amplitude of PWM pulse), and length of sound. The Sound Generator consists of the following registers and counters.

- DMA Transfer Update Enable Register (SGDER)
- Sound Control Register (SGCR)
- Amplitude Data Register (SGAR)
- Frequency Data Register (SGFR)
- Time Cycle Register (SGTCR)
- Tone Output Number Register (SGNR)
- Increase and Decrease Data Register (SGIDR)
- PWM Cycle Data Register (SGPCR)
- Extended Frequency Data Register (SGEFR)
- DMA Transfer Intermediate Register (SGDMAR)
- Interrupt Clear Register (SGCCR)
- PWM pulse generation counter
- Frequency counter
- Decrement counter
- Tone pulse counter

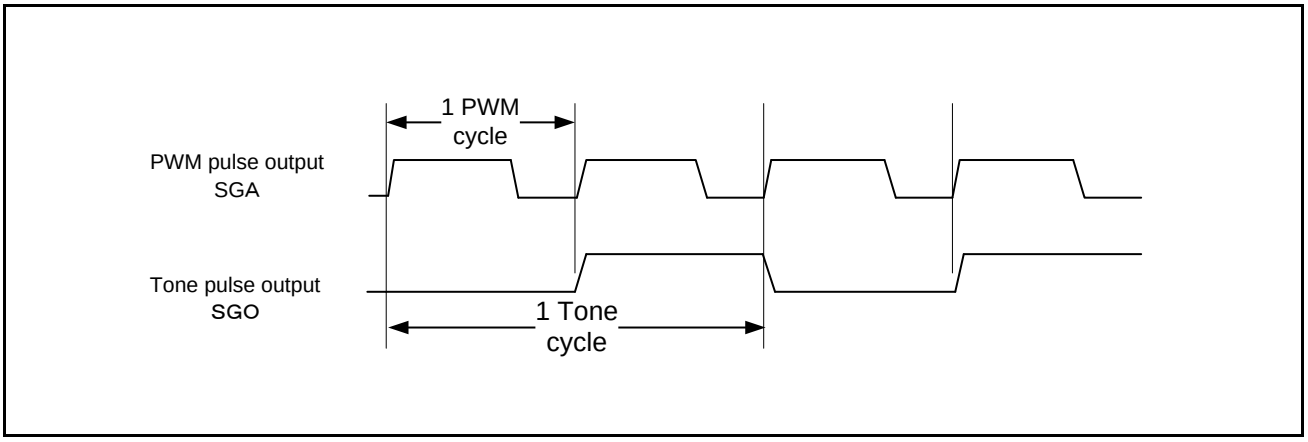
Functions of Sound Generator

Table 18-1 Functions of Sound Generator

No	Item	Function
1	Operation clock	Bus clock (BCLK)
2	Clock input	For clock input of the Sound Generator, divide the bus clock (BCLK) and use it. - 1 x BCLK - 1/2 x BCLK - 1/4 x BCLK - 1/8 x BCLK
3	Waveform	Rectangular wave for sound (tone pulse output, sound output from SGO pin)
4	Volume of sound	It is enabled to set their arbitrary level (PWM pulse output). (Amplitude output from SGA pin)
5	Frequency	It is enabled to set their arbitrary level for sound signal frequency. (Frequency setting and PWM cycle setting) The duty ratio (high side) of sound signal is programmable up to 50%. The high width of sound signal is controlled by the PWM cycle unit. (Extended Frequency setting)
6	Length of sound	It is enabled to set their arbitrary level.
7	Interrupt	- It is enabled to generate interrupt requests at the end of outputting sound at programmed length (An overflow of Tone pulse counter). - It is enabled to generate interrupts by writing "1" to the Start bit (SGCR.ST) when in DMA mode (SGCR.DMA="1").

Output of Sound Generator

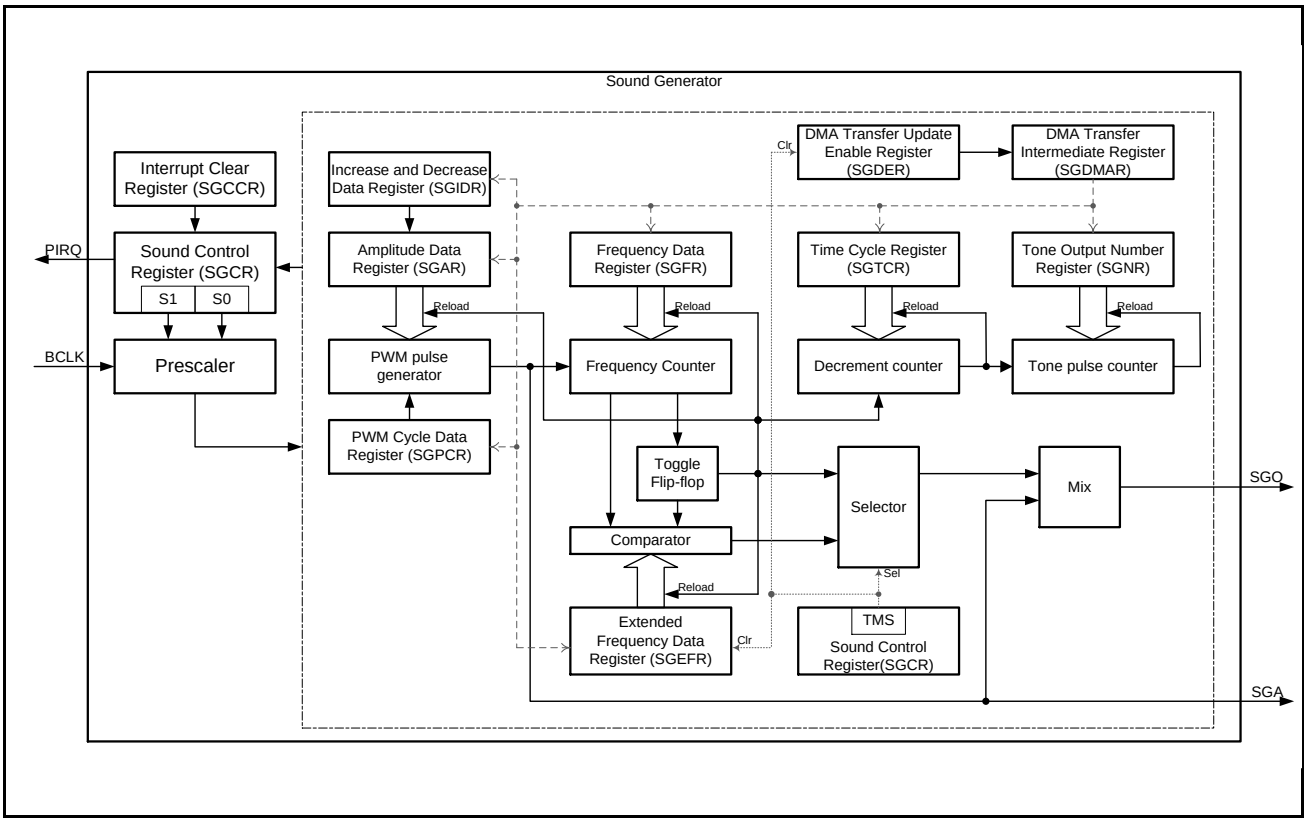
Figure 18-1 Example of Output



2. Configuration

This chapter shows the block diagram of Sound Generator.

Figure 18-2 Block Diagram

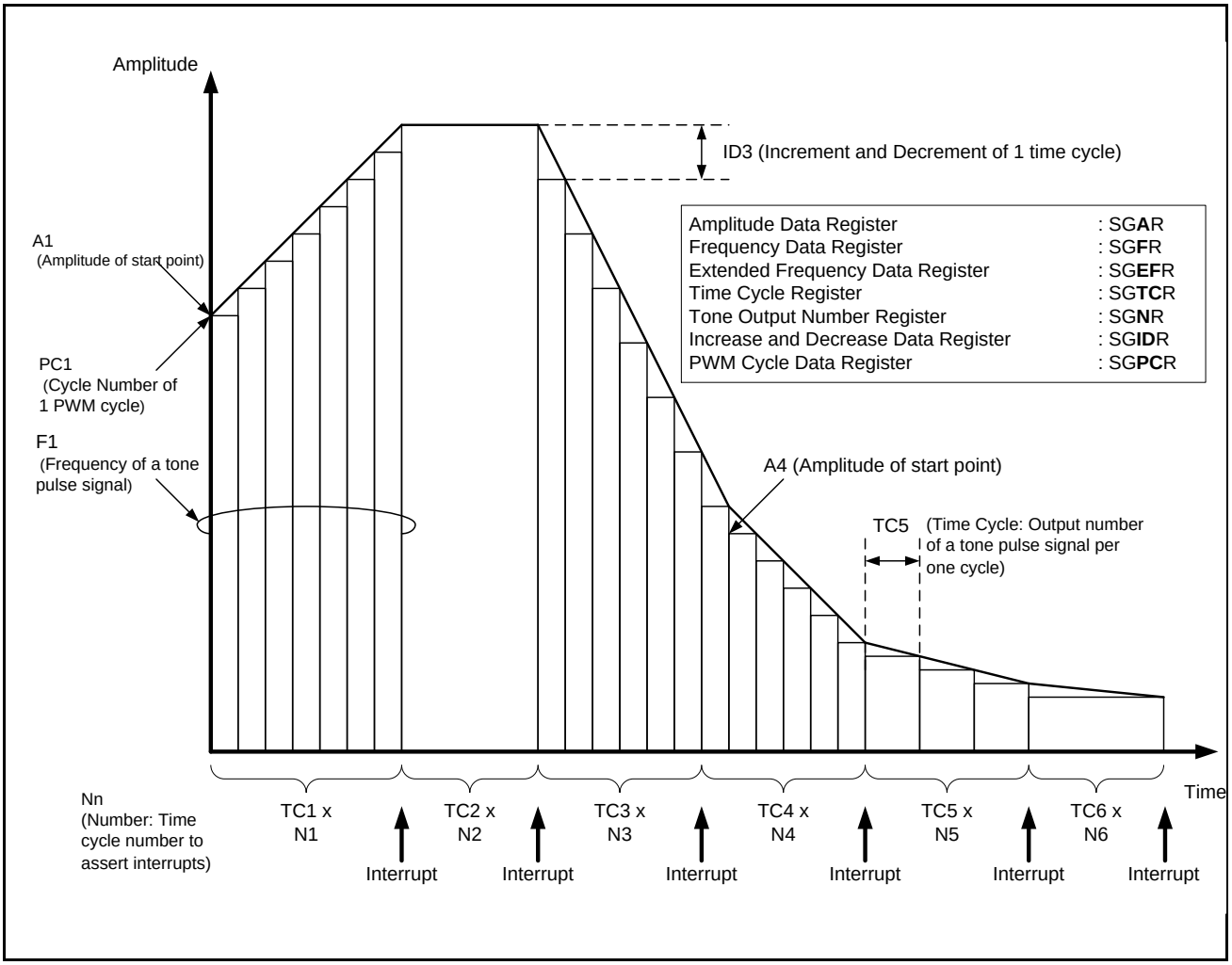


3. Operations

This chapter describes the Sound Generator operation.

The concept of Sound Generator operation is shown in the diagram as below.

Figure 18-3 The Operation of Sound Generator (Conceptual Diagram)



Set the various values in the register as follows:

Amplitude of start point (An) (SGAR)	- to the Amplitude Data Register
Frequency of a tone pulse signal (Fn) (SGFR)	- to the Frequency Data Register
Duty ratio of a tone pulse signal	- to the Extended Frequency Data Register (SGEFR)
Output number of a tone pulse signal per one cycle (TCn)	- to the Time Cycle Register (SGTCR)
Time cycle number to assert interrupts (Nn)	- to the Tone Output Number Register (SGNR)

Increment and Decrement of 1 time cycle (IDn)	- to the Increase and Decrease Data Register (SGIDR)
Cycle Number of 1 PWM cycle (PCn)	- to the PWM Cycle Data Register (SGPCR)
Other information about Sound Generator control (SGCR)	- to the Sound Control Register

The Sound Generator outputs the tone pulse signal and the amplitude data according to the setting above.

In the conceptual diagram shown above, it outputs 6 steps of signal.

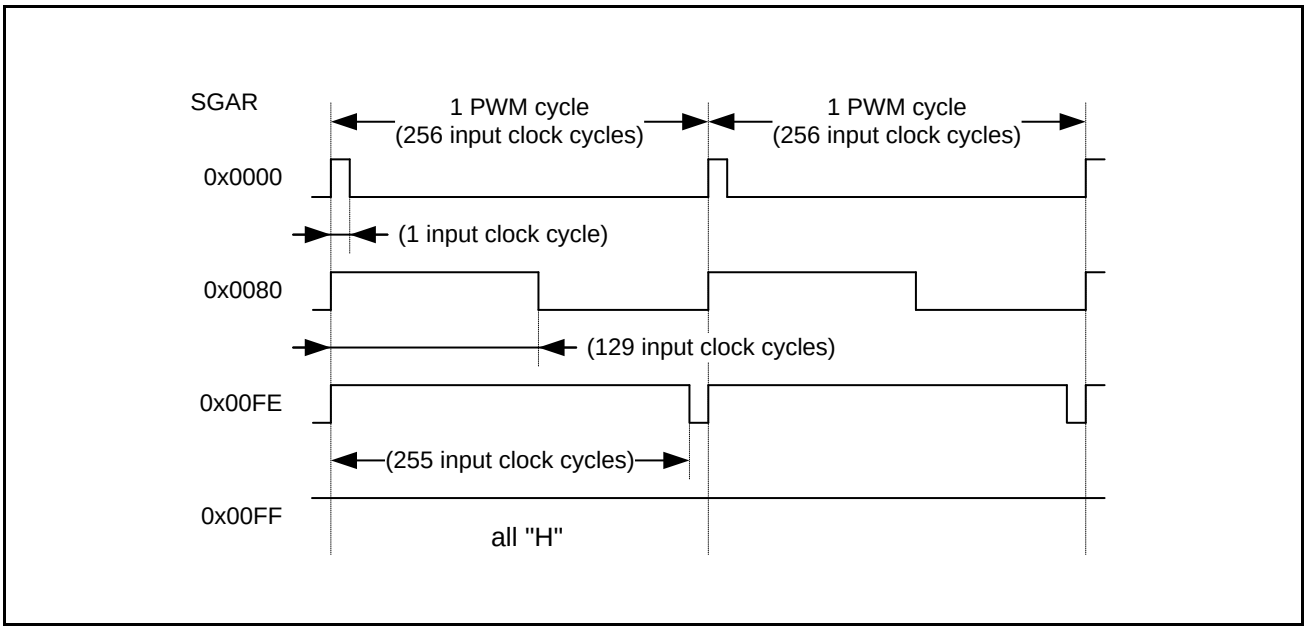
Each register, such as Amplitude Data Register, is supposed to be written at the following timing.

- Before starting signal output
- Each time after asserting an interrupt

3.1. Relation between the Amplitude Data Register (SGAR) and PWM Pulse

This section describes the relation between the Amplitude Data Register (SGAR) and a PWM pulse.

Figure 18-4 The Relation between the Amplitude Data Register (SGAR) and PWM Pulse



The amplitude data is output as a PWM (Pulse Width Modulation) waveform on the SGA pin.

The length of a PWM cycle is programmable in the PWM Cycle Data Register (SGPCR). This length is based on the count of input clock. The initial value is 256 input clock cycle (SGPCR=0x00FF).

The value [Amplitude Data Register (SGAR) + 1] means the number of input clock cycles in which the SGA pin is "H" during 1 PWM cycle. Moreover, when the Amplitude Data Register (SGAR) is greater than or equal to the PWM Cycle Data Register (SGPCR), the SGA pin output is always "H".

3.2. Relation between the Frequency Data Register (SGFR), the Extended Frequency Data Register (SGEFR) and Tone Pulse Signal

This section describes the relation between the Frequency Data Register (SGFR), the Extended Frequency Data Register (SGFR) and a tone pulse signal.

Sound generator has two modes of Normal mode and Extended mode for generation of a tone pulse signal.

The mode is selected by the Tone output mode select bit (TMS) of the Sound Control Register (SGCR).

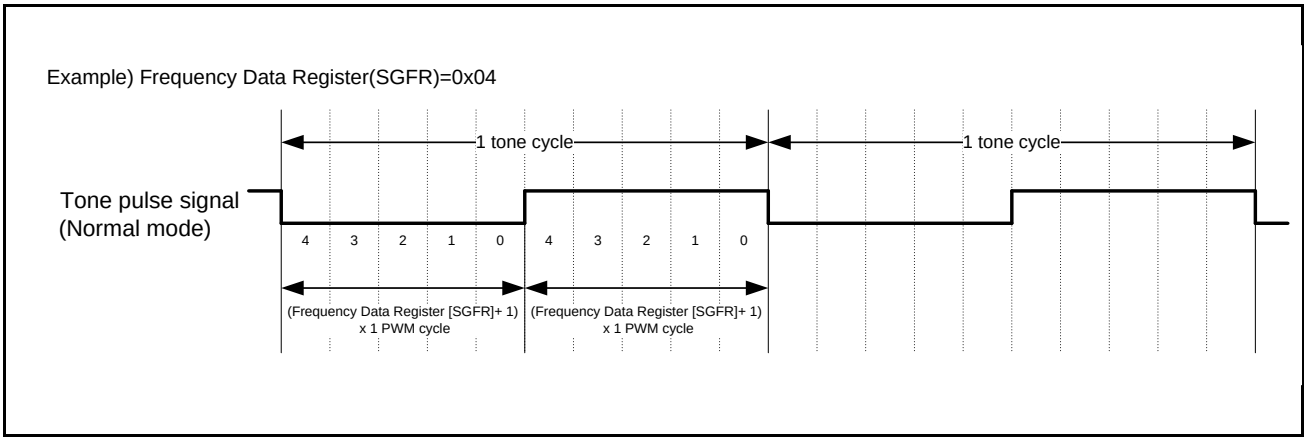
When the Tone output mode select bit (TMS) of the Sound Control Register (SGCR) is "0" (Normal mode), the duty ratio of a tone pulse signal is fixed to 50%.

When the Tone output mode select bit (TMS) of the Sound Control Register (SGCR) is "1" (Extended mode), the duty ratio (high side) of a tone pulse signal is programmable up to 50%.

3.2.1. Generation of a Tone Pulse Signal in Normal Mode

When in Normal mode (SGCR.TMS="0"), a sound generator generates a tone pulse signal by the Frequency Data Register (SGFR).

Figure 18-5 Tone Pulse Signal in Normal Mode (SGCR.TMS="0")



A tone cycle of a tone pulse signal is controlled by the Frequency Data Register (SGFR) and the PWM cycle.

A tone cycle is "(Frequency Data Register [SGFR] + 1) x 1 PWM cycle x 2".

The tone pulse signal transits between "L" and "H" at every cycle of "(Frequency Data Register [SGFR] + 1) x 1 PWM cycle)". This signal is generated by an internal toggle flip-flop.

The duty ratio of a tone pulse signal of Normal mode (SGCR.TMS="0") is fixed to 50%.

When the Tone output bit (SGCR.TONE) of the Sound Control Register (SGCR) is "0", the tone pulse signal is mixed with a PWM pulse, and is output from the SGO pin. And when the Tone output bit (SGCR.TONE) of the Sound Control Register (SGCR) is "1", the tone pulse signal is output without being mixed.

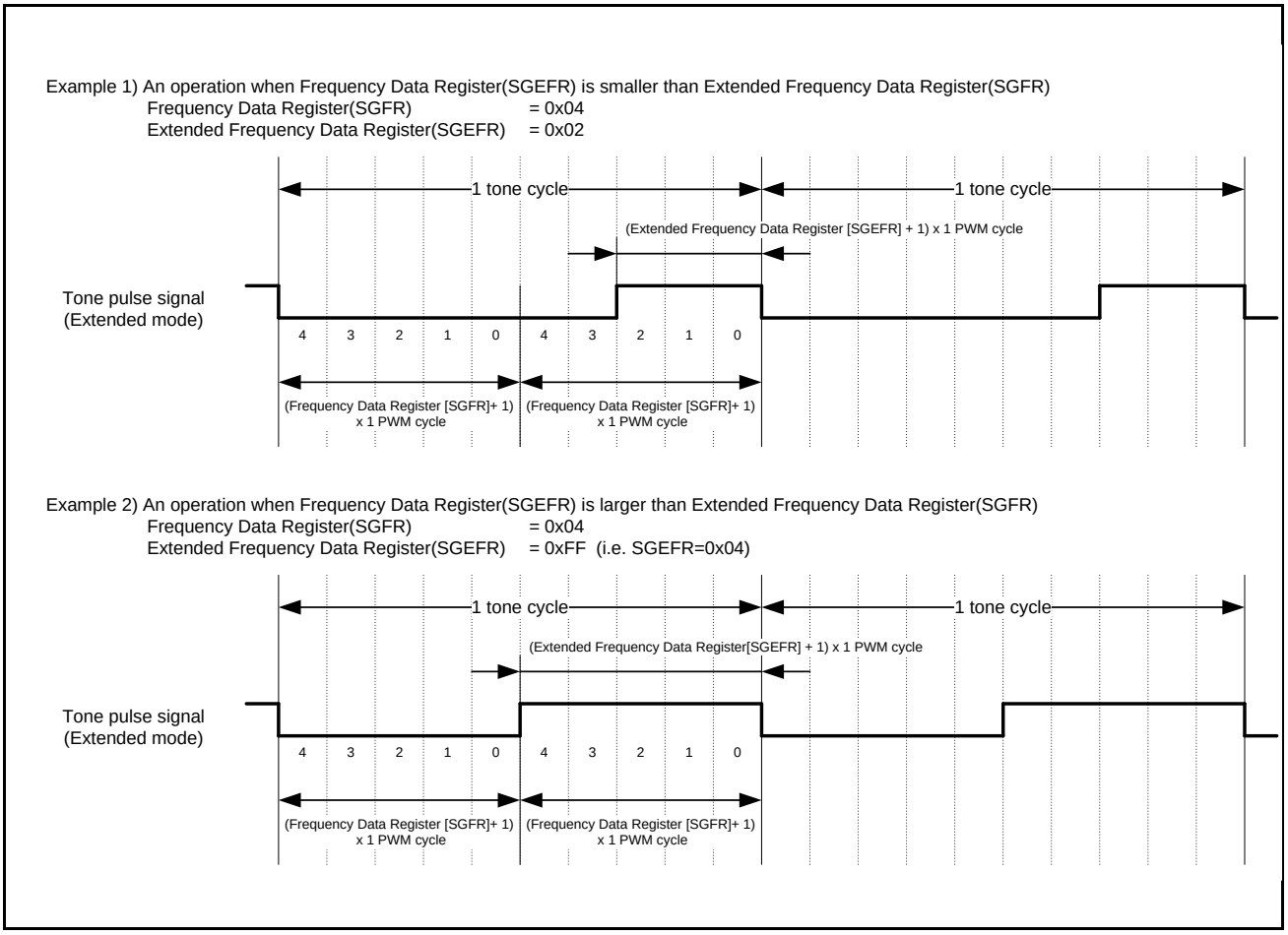
Note:

- *During operation, in the case of changing a register value, meet the following either conditions.*
  (1)Sound generator is in the stop state (SGCR.ST="0" and SGCR.BUSY="0").
  (2)The changing comes between an interrupt occurrence and falling edge of the first tone pulse.
  When it meets neither condition, the sound output cannot guarantee the expected duty ratio.

3.2.2. Generation of a Tone Pulse Signal in Extended Mode

When in Extended mode (SGCR.TMS="1"), a sound generator generates a tone pulse signal by the Frequency Data Register (SGFR) and the Extended Frequency Data Register (SGEFR).

Figure 18-6 Tone Pulse Signal in Extended Mode (SGCR.TMS="1")



A tone cycle of a tone pulse signal is controlled by the Frequency Data Register (SGFR) and the PWM cycle.

A tone cycle is "(Frequency Data Register [SGFR] + 1) x 1 PWM cycle x 2".

The duty ratio (high side) of a tone pulse signal of Extended mode (SGCR.TMS="1") is programmable up to 50%.

The high width of a tone pulse signal is controlled by the PWM cycle unit, with the value of Extended Frequency Data Register (SGEFR). The high width is "(Extended Frequency Data Register [SGEFR] + 1) x1 PWM cycle".

When the Tone output bit (SGCR.TONE) of the Sound Control Register (SGCR) is "0", the tone pulse signal is mixed with a PWM pulse, and is output from the SGO pin. And when the Tone output bit (SGCR.TONE) of the Sound Control Register (SGCR) is "1", the tone pulse signal is output without being mixed.

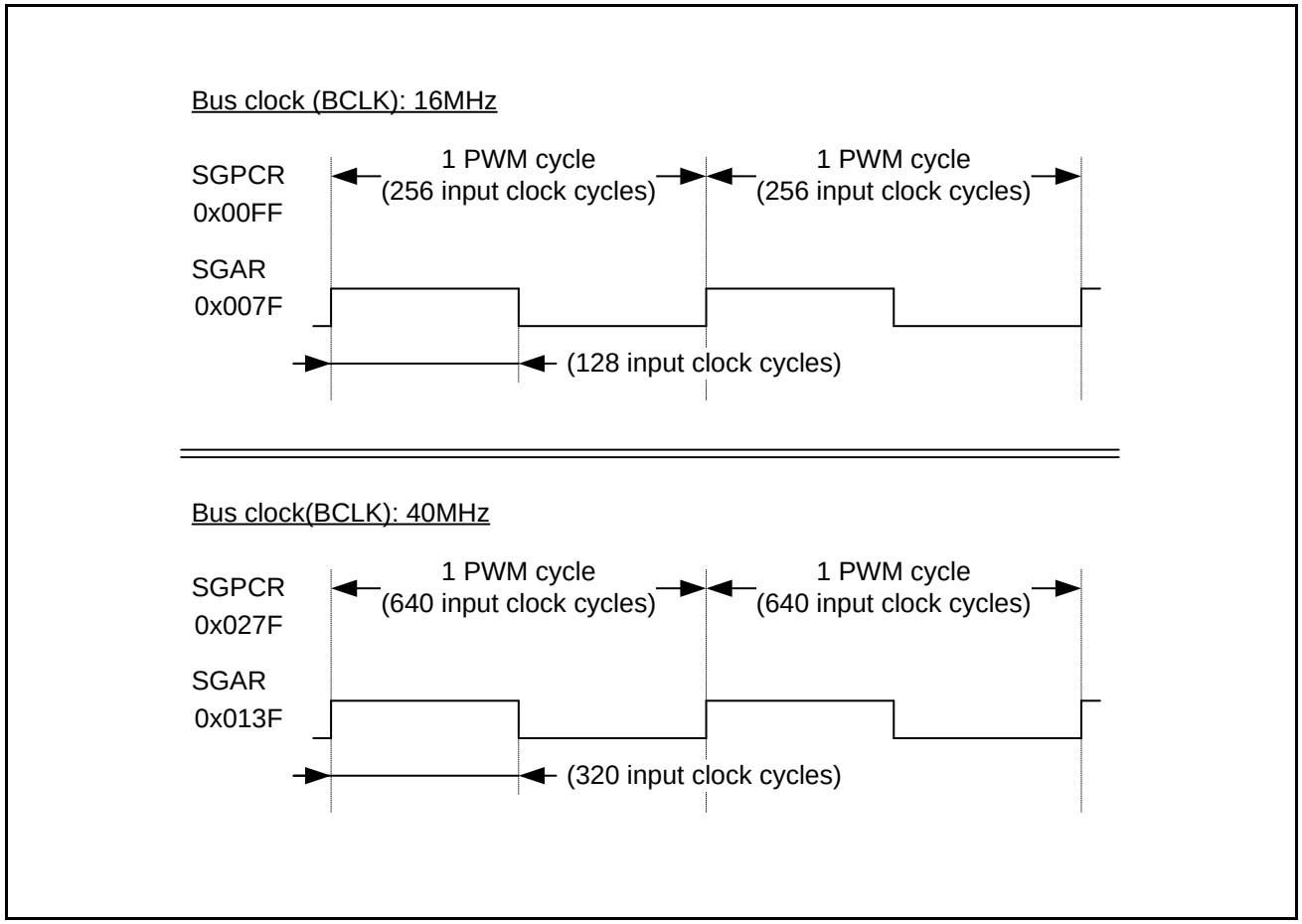
Notes:

- *When the Extended Frequency Data Register value (SGEFR) is larger than the Frequency Data Register value (SGFR) in Extended mode (SGCR.TMS="1"), the duty ratio is 50%.*
- *During operation, in the case of changing a register value, meet the following either conditions.*
  (1)Sound generator is in the stop state (SGCR.ST="0" and SGCR.BUSY="0").
  (2)The changing comes between an interrupt occurrence and falling edge of the first tone pulse.
  When it meets neither condition, the sound output cannot guarantee the expected duty ratio.

3.3. Relation between the PWM Cycle Data Register (SGPCR) and PWM Cycle

This section describes the relation between the PWM Cycle Data Register (SGPCR) and PWM cycle.

Figure 18-7 Relation between the PWM Cycle Data Register (SGPCR) and PWM Cycle



The length of a PWM cycle is programmable in the PWM Cycle Data Register (SGPCR). This length is based on the count of input clock, and defined as "PWM Cycle Data Register [SGPCR] + 1". The input clock is created by dividing the bus clock.

The PWM cycle is a reference clock for a tone pulse signal (or a mixed signal of the tone pulse signal and a PWM pulse signal), and a PWM pulse signal.

To generate the same sound output both by the bus clock at 16 MHz and 40 MHz, for example, it can be made by changing the values of the PWM Cycle Data Register (SGPCR) and the Amplitude Data Register (SGAR). Because the bus clock proportion of 16 MHz and 40 MHz is 1:2.5, the same sound can be made by setting the values of these two registers to 1:2.5.

3.4. Relation between the DMA Transfer Update Enable Register (SGDER) and DMA Settings

This section describes the relation between the DMA Transfer Update Enable Register (SGDER) and DMA settings.

3.4.1. The Number of DMA Transfers

DMA Transfer Update Enable Register (SGDER) determines the number of DMA transfer.

Table 18-2 The Relations between the Setting of the DMA Transfer Update Enable Register (SGDER) and the Number of DMA Transfer

No.	Setting of SGDER			The Number of DMA Transfers
	EFRE	ARE1, ARE0, FRE, NRE	TCRE, IDRE,PCRE1,PCRE0	
1	0	all"0"	all"0"	0
2	0	all"0"	except all"0"	1
3	0	except all"0"	all"0"	1
4	0	except all"0"	except all"0"	2
5	1	all"0"	all"0"	1
6	1	all"0"	except all"0"	2
7	1	except all"0"	all"0"	2
8	1	except all"0"	except all"0"	3

Note:

- *In Normal mode (SGCR.TMS="0"), the maximum number of DMA transfer is two. (i.e. SGDER.EFRE is fixed to "0".)*

3.4.2. DMA Transfer Size

The DMA transfer size (1 byte, 2 bytes or 4 bytes) depends on the setting of the DMA Transfer Update Enable Register (SGDER).

The greater value of the following three becomes the DMA transfer size:

The value of SGDER.EFRE,

The value of SGDER.ARE1, SGDER.ARE0, SGDER.FRE, and SGDER.NRE,

The value of SGDER.TCRE, SGDER.IDRE, SGDER.PCRE1, and SGDER.PCRE0

The transfer size of 3 bytes is regarded as 4 bytes.

3.4.3. Transfer Byte Position in the DMA Transfer Intermediate Register (SGDMAR)

The DMA transfer byte position in the "DMA Transfer Intermediate Register (SGDMAR)" depends on the setting of the "DMA Transfer Update Enable Register (SGDER)" and the DMA transfer size.

If the size of a DMA transfer is less than 4 bytes, the byte position of DMA transfer is left-aligned.

The table below shows the relations between
"the setting of the DMA Transfer Update Enable Register (SGDER)"
and
"the byte position in the DMA Transfer Intermediate Register (SGDMAR)".
The byte position in the DMA Transfer Intermediate Register (SGDMAR) corresponds to the selection of the following registers.

- Amplitude Data Register (SGAR [15:0])
- Frequency Data Register (SGFR [7:0])
- Tone Output Number Register (SGNR [7:0])

The transfer size #1 is calculated by the setting of {SGDER.ARE1, SGDER.ARE0, SGDER.FRE, and SGDER.NRE} in the "DMA Transfer Update Enable Register (SGDER)". When this transfer size #1 is not 4 bytes, the transfer byte position is left-aligned.

Table 18-3 The Relations between the Setting of the DMA Transfer Update Enable Register (SGDER) and the Transfer Byte Position of SGDMAR #1

No.	Setting of SGDER				Transfer Size #1 *1	Transfer Byte Position of SGDMAR			
	ARE1	ARE0	FRE	NRE		SGAR [15:8]	SGAR [7:0]	SGFR [7:0]	SGNR [7:0]
1	0	0	0	0	0	-	-	-	-
2	1	0	0	0	1	SGDMAR [31:24]	-	-	-
3	0	1	0	0	1	-	SGDMAR [31:24]	-	-
4	1	1	0	0	2	SGDMAR [31:24]	SGDMAR [23:16]	-	-
5	0	0	1	0	1	-	-	SGDMAR [31:24]	-
6	1	0	1	0	2	SGDMAR [31:24]	-	SGDMAR [23:16]	-
7	0	1	1	0	2	-	SGDMAR [31:24]	SGDMAR [23:16]	-
8	1	1	1	0	4	SGDMAR [31:24]	SGDMAR [23:16]	SGDMAR [15:8]	-
9	0	0	0	1	1	-	-	-	SGDMAR [31:24]
10	1	0	0	1	2	SGDMAR [31:24]	-	-	SGDMAR [23:16]
11	0	1	0	1	2	-	SGDMAR [31:24]	-	SGDMAR [23:16]
12	1	1	0	1	4	SGDMAR [31:24]	SGDMAR [23:16]	-	SGDMAR [15:8]
13	0	0	1	1	2	-	-	SGDMAR [31:24]	SGDMAR [23:16]
14	1	0	1	1	4	SGDMAR [31:24]	-	SGDMAR [23:16]	SGDMAR [15:8]
15	0	1	1	1	4	-	SGDMAR [31:24]	SGDMAR [23:16]	SGDMAR [15:8]
16	1	1	1	1	4	SGDMAR [31:24]	SGDMAR [23:16]	SGDMAR [15:8]	SGDMAR [7:0]

*1: The transfer size which is calculated by the setting of SGDER.ARE1, SGDER.ARE0, SGDER.FRE, and SGDER.NRE.

- : Do not care.

The table below shows the relations between
"the setting of the DMA Transfer Update Enable Register (SGDER)"
and
"the byte position in the DMA Transfer Intermediate Register (SGDMAR)".
The byte position in the DMA Transfer Intermediate Register (SGDMAR) corresponds to the selection of the following registers.
- Time Cycle Register (SGTCR[7:0])
- Increase and Decrease Data Register (SGIDR[7:0])
- PWM Cycle Data Register (SGPCR[15:0])

The transfer size #2 is calculated by the setting of {SGDER.TCRE, SGDER.IDRE, SGDER.PCRE1, and SGDER.PCRE0} in the DMA Transfer Update Enable Register (SGDER). When this transfer size #2 is not 4 bytes, the transfer byte position is left-aligned.

Table 18-4 The Relations between the Setting of the DMA Transfer Update Enable Register (SGDER) and Transfer Byte Position of SGDMAR #2

No.	Setting of SGDER				Transfer Size #2 *1	Transfer Byte Position of SGDMAR			
	TCRE	IDRE	PCRE1	PCRE0		SGTCR [7:0]	SGIDR [7:0]	SGPCR [15:8]	SGPCR [7:0]
1	0	0	0	0	0	-	-	-	-
2	1	0	0	0	1	SGDMAR [31:24]	-	-	-
3	0	1	0	0	1	-	SGDMAR [31:24]	-	-
4	1	1	0	0	2	SGDMAR [31:24]	SGDMAR [23:16]	-	-
5	0	0	1	0	1	-	-	SGDMAR [31:24]	-
6	1	0	1	0	2	SGDMAR [31:24]	-	SGDMAR [23:16]	-
7	0	1	1	0	2	-	SGDMAR [31:24]	SGDMAR [23:16]	-
8	1	1	1	0	4	SGDMAR [31:24]	SGDMAR [23:16]	SGDMAR [15:8]	-
9	0	0	0	1	1	-	-	-	SGDMAR [31:24]
10	1	0	0	1	2	SGDMAR [31:24]	-	-	SGDMAR [23:16]
11	0	1	0	1	2	-	SGDMAR [31:24]	-	SGDMAR [23:16]
12	1	1	0	1	4	SGDMAR [31:24]	SGDMAR [23:16]	-	SGDMAR [15:8]
13	0	0	1	1	2	-	-	SGDMAR [31:24]	SGDMAR [23:16]
14	1	0	1	1	4	SGDMAR [31:24]	-	SGDMAR [23:16]	SGDMAR [15:8]
15	0	1	1	1	4	-	SGDMAR [31:24]	SGDMAR [23:16]	SGDMAR [15:8]
16	1	1	1	1	4	SGDMAR [31:24]	SGDMAR [23:16]	SGDMAR [15:8]	SGDMAR [7:0]

*1: The transfer size which is calculated by the setting of SGDER.TCRE, SGDER.IDRE, SGDER.PCRE1, and SGDER.PCRE0.

- : Do not care.

The table below shows the relations between
"the setting of the DMA Transfer Update Enable Register (SGDER)"

and
"the byte position in the DMA Transfer Intermediate Register (SGDMAR)".
The byte position in the DMA Transfer Intermediate Register (SGDMAR) corresponds to the selection of the following register.

- Extended Frequency Data Register (SGEFR [7:0])

The transfer size #3 is calculated by the setting of {SGDER.EFRE} in the "DMA Transfer Update Enable Register (SGDER)". When this transfer size #3 is not 4 bytes, the transfer byte position is left-aligned.

Table 18-5 The Relations between the Setting of the DMA Transfer Update Enable Register (SGDER) and the Transfer Byte Position of SGDMAR #3

No.	Setting of SGDER				Transfer Size #3 *1	Transfer Byte Position of SGDMAR			
	-	-	-	EFRE		-	-	-	SGEFR [7:0]
1	0	0	0	0	0	-	-	-	-
2	0	0	0	1	1	-	-	-	SGDMAR [31:24]

*1: The transfer size which is calculated by the setting of SGDER.EFRE.

- : Do not care.

3.4.4. DMA Transfer Image

This section shows an example of DMA transfer image, under the setting of DMA Transfer Update Enable Register (SGDER):

[SGDER.EFRE]	="1"
[SGDER.ARE1, SGDER.ARE0, SGDER.FRE, SGDER.NRE]	="1001"
[SGDER.TCRE, SGDER.IDRE, SGDER.PCRE1, SGDER.PCRE0]	="0100"

Condition:

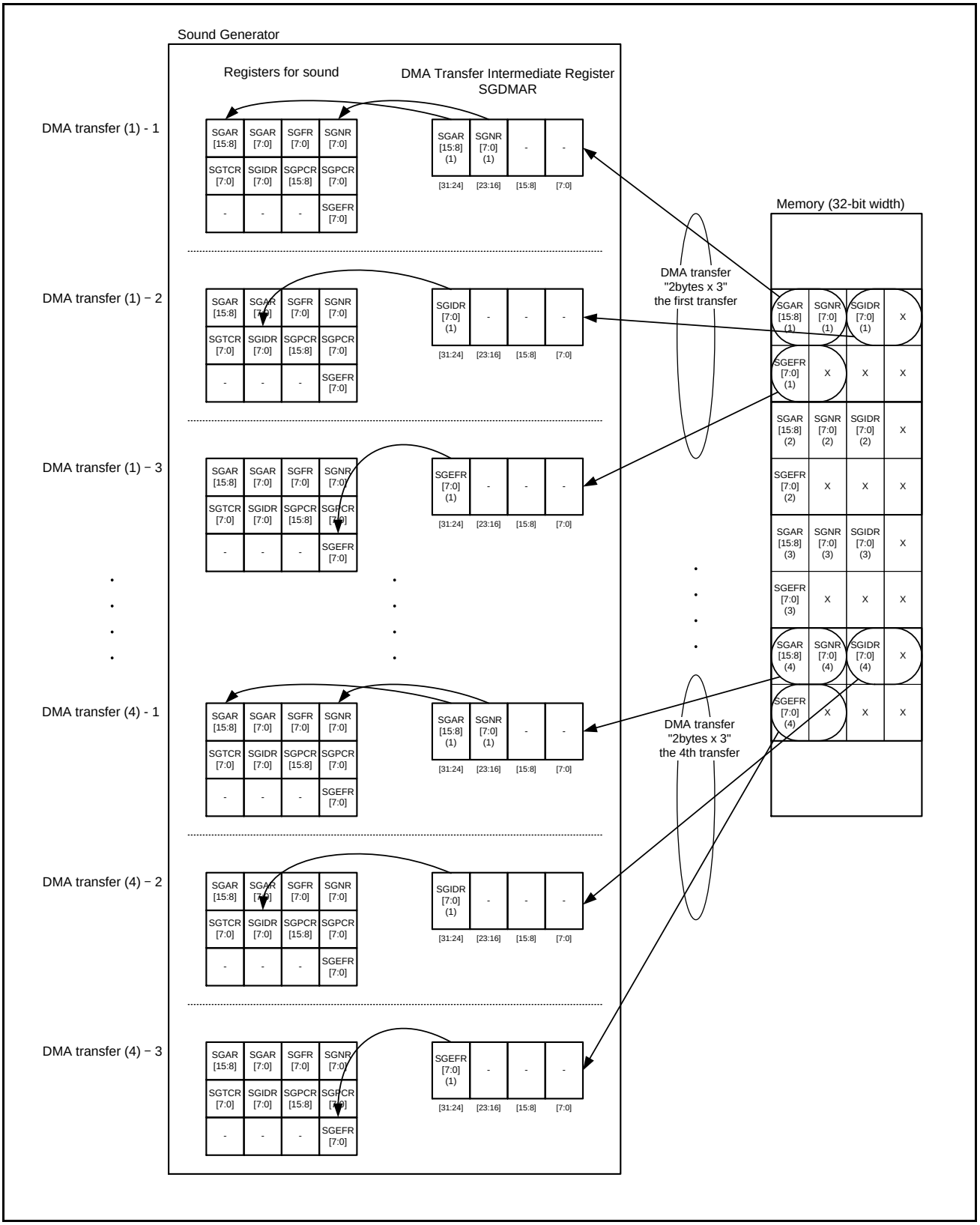
The number of times of DMA transfer : 3 times

DMA transfer size : 2 bytes

Transfer byte position of the DMA Transfer Intermediate Register (SGDMAR):

The first	SGDMAR[31:24]	← The amplitude data (upper byte) / SGAR[15:8]
	SGDMAR[23:16]	← The tone output number / SGNR[7:0]
	SGDMAR[15:0]	← Do not care
The second	SGDMAR[31:24]	← The increment and decrement data / SGIDR[7:0]
	SGDMAR[23:0]	← Do not care
The third	SGDMAR[31:24]	← The extended frequency data / SGEFR[7:0]
	SGDMAR[23:0]	← Do not care

Figure 18-8 DMA Transfer Image

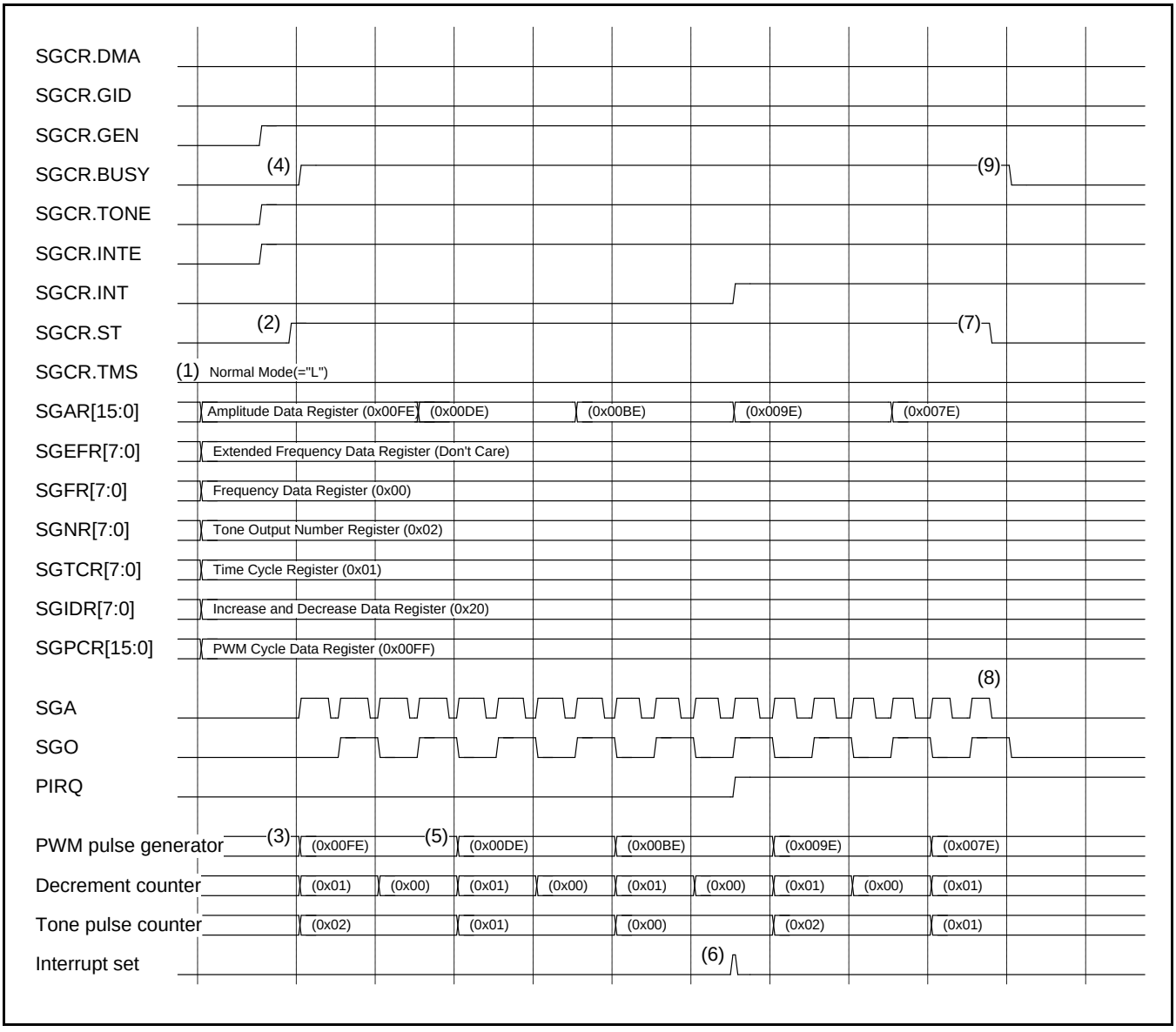


3.5. Sound Generator Operation

This section describes the Sound Generator operation as below

3.5.1. Sound Generator Operation in Normal Mode

Figure 18-9 Sound Generator Operation in Normal Mode (SGCR.TMS="0")



(1) The reload values are written to the Amplitude Data Register (SGAR), the Frequency Data Register (SGFR), the Tone Output Number Register (SGNR), and the Time Cycle Register (SGTCR) by software. Also the selection of increment and decrement of the amplitude is written to the Increase and Decrease Data Register (SGIDR) by software as well as the number of cycles in 1 PWM cycle to the PWM Cycle Data Register (SGPCR).

Moreover, set other information to the Sound Control Register (SGCR) to control the Sound Generator. Initialize the Interrupt status bit (SGCR.INT) and set the Interrupt enable bit (SGCR.INTE).

(2) Write "1" to the Start bit (SGCR.ST).

(3) By setting "1" to the Start bit (SGCR.ST), the Amplitude Data Register (SGAR) value is loaded into the PWM pulse generator, the Frequency Data Register (SGFR) value into the Frequency counter, the Tone Output Number Register (SGNR) value into the Tone pulse counter, the Time Cycle Register (SGTCR) value into the Decrement counter.

(4) The operation flag (SGCR.BUSY) is automatically set.

(5) Due to the operation that the Decrement counter counts the number of tone pulses until it reaches the reload value, the Amplitude Data Register (SGAR) value decreases according to the setting of the Automatic increase/decrease enable bit (SGCR.GEN) and the Increase/decrease setting bit (SGCR.GID).

(6) The Tone pulse counter counts the number of tone pulses. When the following conditions are satisfied, it sets the Interrupt status bit (SGCR.INT) and asserts the Interrupt request (PIRQ).
- Tone pulse counter is 0x00
- Decrement counter is 0x00
- At the rising edge of SGO

(7) Write "0" to the Start bit (SGCR.ST). The Sound Generator keeps operating until the Busy status bit (SGCR.BUSY) turns "0".

(8) The Sound Generator stops when the current tone cycle finishes.

(9) The Busy status bit (SGCR.BUSY) turns "0".

DMA Transfer Start Interrupt Setting Enable Bit (SGCR.DMA)

To assert the first Interrupt request (PIRQ) after starting (SGCR.ST="1"), there are two modes for the request. And the selection depends on the setting of the "DMA transfer start interrupt setting enable bit (SGCR.DMA)".

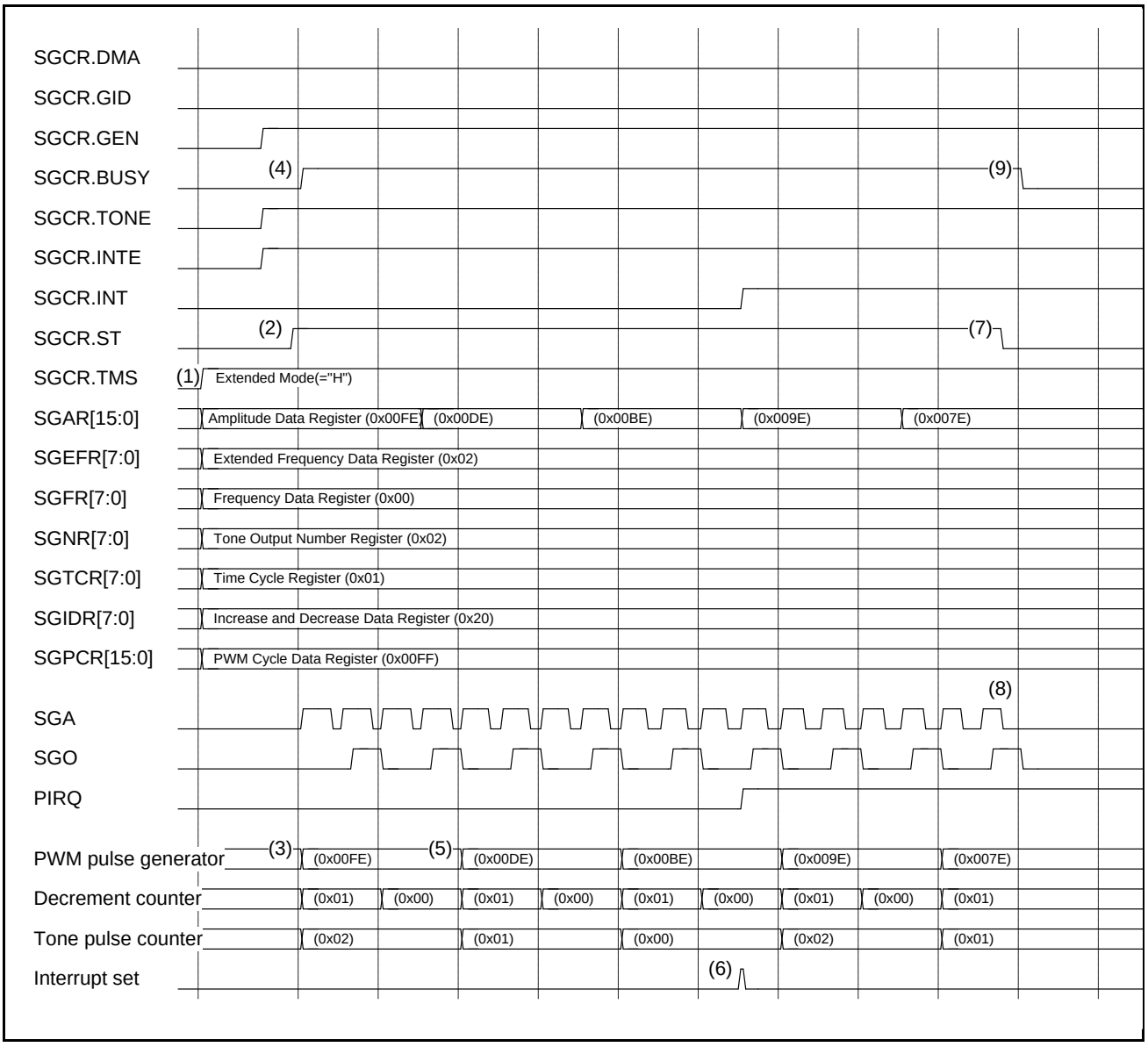
- Normal mode :
  When Sound Generator outputs the tone pulses, to the number programmed in the Time Cycle Register (SGTCR)
- DMA mode :
  Immediately after the setting of Start bit (writing "1" to SGCR.ST)
  (In this case, the Interrupt request (PIRQ) is regarded as the DMA transfer request.)

DMA Transfer

The following registers are set through the DMA Transfer Intermediate Register (SGDMAR).
- Amplitude Data Register (SGAR)
- Frequency Data Register (SGFR)
- Time Cycle Register (SGTCR)
- Tone Output Number Register (SGNR)
- Increase and Decrease Data Register (SGIDR)
- PWM Cycle Data Register (SGPCR)

3.5.2. Sound Generator Operation in Extended Mode

Figure 18-10 Sound Generator Operation in Extended Mode (SGCR.TMS="1")



(1) The reload values are written to the Amplitude Data Register (SGAR), the Frequency Data Register (SGFR), the Tone Output Number Register (SGNR), and the Time Cycle Register (SGTCR) by software. Comparison value of high width of a tone pulse signal is written in the Extended Frequency Data Register (SGEFR) by software. Also the selection of increment and decrement of the amplitude is written to the Increase and Decrease Data Register (SGIDR) by software as well as the number of cycles in 1 PWM cycle to the PWM Cycle Data Register (SGPCR).

Moreover, set other information to the Sound Control Register (SGCR) to control the Sound Generator. Initialize the Interrupt status bit (SGCR.INT) and set the Interrupt enable bit (SGCR.INTE).

(2) Write "1" to the Start bit (SGCR.ST).

(3) By setting "1" to the Start bit(SGCR.ST), the Amplitude Data Register (SGAR) value is loaded into the PWM pulse generator, the Frequency Data Register (SGFR) value into the Frequency counter, the Extended Frequency Data Register (SGEFR) value into the comparison register of high width of a tone pulse signal, the Tone Output Number Register (SGNR) value into the Tone pulse counter, the Time Cycle Register (SGTCR) value into the Decrement counter.

(4) The operation flag (SGCR.BUSY) is automatically set.

(5) Due to the operation that the Decrement counter counts the number of tone pulses until it reaches the reload value, the Amplitude Data Register (SGAR) value decreases according to the setting of the Automatic increase/decrease enable bit (SGCR.GEN) and the Increase/decrease setting bit (SGCR.GID).

(6) The Tone pulse counter counts the number of tone pulses. When the following conditions are satisfied, it sets the Interrupt status bit (SGCR.INT) and asserts the Interrupt request (PIRQ).
- Tone pulse counter is 0x00
- Decrement counter is 0x00
- At the rising edge of SGO

(7) Write "0" to the Start bit (SGCR.ST). The Sound Generator keeps operating until the Busy status bit (SGCR.BUSY) turns "0".

(8) The Sound Generator stops when the current tone cycle finishes.

(9) The Busy status bit (SGCR.BUSY) turns "0".

DMA Transfer Start Interrupt Setting Enable Bit (SGCR.DMA)

To assert the first Interrupt request (PIRQ) after starting (SGCR.ST="1"), there are two modes for the request. And the selection depends on the setting of the "DMA transfer start interrupt setting enable bit (SGCR.DMA)".

- Normal mode :
  When Sound Generator outputs the tone pulses, to the number programmed in the Time Cycle Register (SGTCR)
- DMA mode :
  Immediately after the setting of Start bit (writing "1" to SGCR.ST)
  (In this case, the Interrupt request (PIRQ) is regarded as the DMA transfer request.)

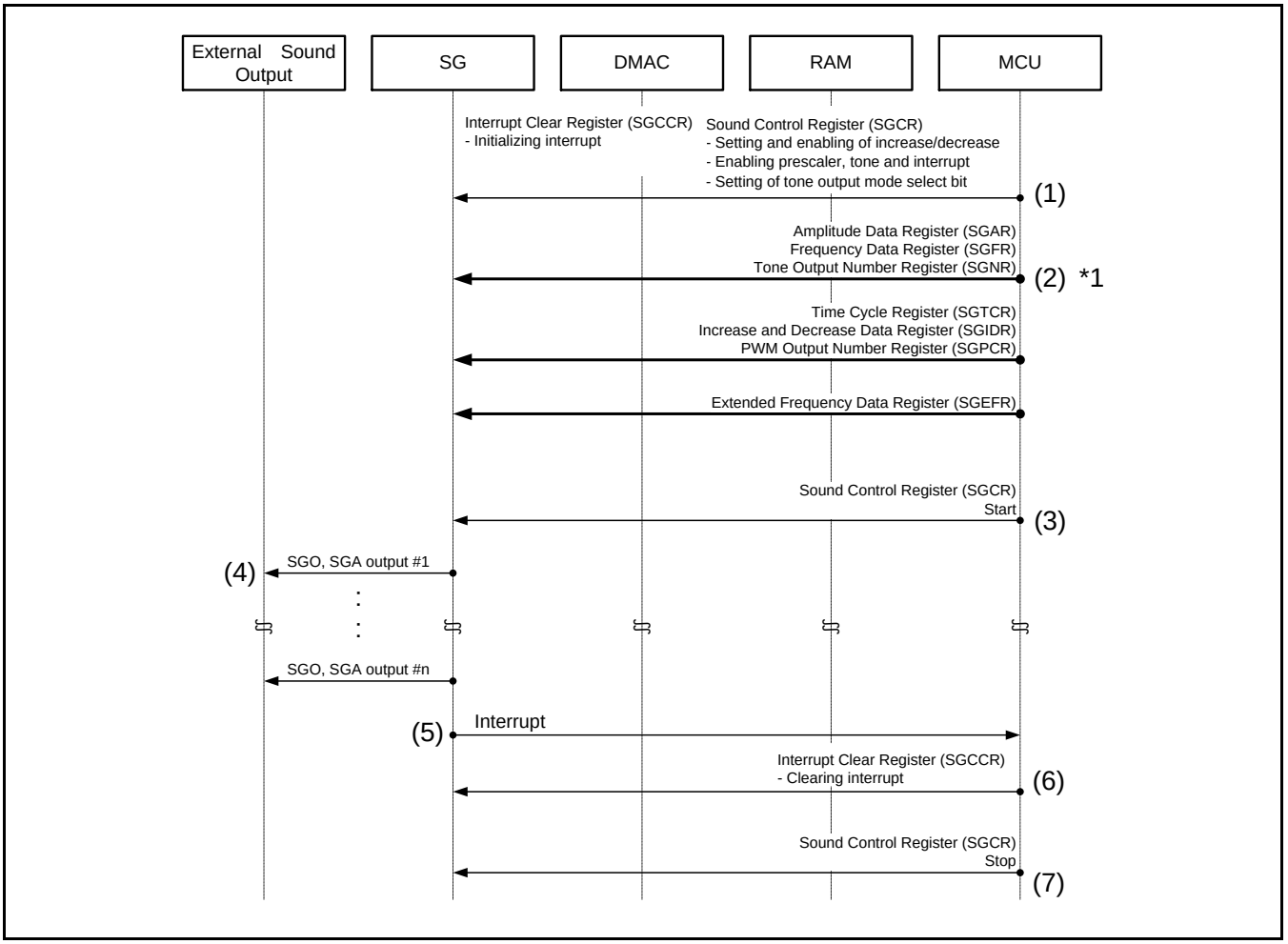
DMA Transfer

The following registers are set through the DMA Transfer Intermediate Register (SGDMAR).
- Amplitude Data Register (SGAR)
- Frequency Data Register (SGFR)
- Time Cycle Register (SGTCR)
- Tone Output Number Register (SGNR)
- Increase and Decrease Data Register (SGIDR)
- PWM Cycle Data Register (SGPCR)
- Extended Frequency Data Register (SGEFR)

3.5.3. Single Operation of Sound Generator by MCU

This section shows the single operation flow of the Sound Generator by MCU.

Figure 18-11 Single Operation of Sound Generator by MCU



(1) Set the information to control the Sound Generator to the Sound Control Register (SGCR) by software. Initialize the Interrupt status bit (SGCR.INT) and set the Interrupt enable bit (SGCR.INTE).

(2) Set registers "Amplitude Data Register (SGAR)", "Frequency Data Register (SGFR)", "Tone Output Number Register (SGNR)", "Time Cycle Register (SGTCR)", "Increase and Decrease Data Register (SGIDR)", "PWM Cycle Data Register (SGPCR)" and "Extended Frequency Data Register (SGEFR)" by software. (*1: Set only the necessary registers.)

(3) Write "1" to the Start bit (SGCR.ST).

(4) The outputs of SGO and SGA start.

(5) The Tone pulse counter counts the number of tone pulses. When the following conditions are satisfied, the interrupt is generated.
- Tone pulse counter is 0x00
- Decrement counter is 0x00
- At the rising edge of SGO

(6) MCU clears the interrupt.

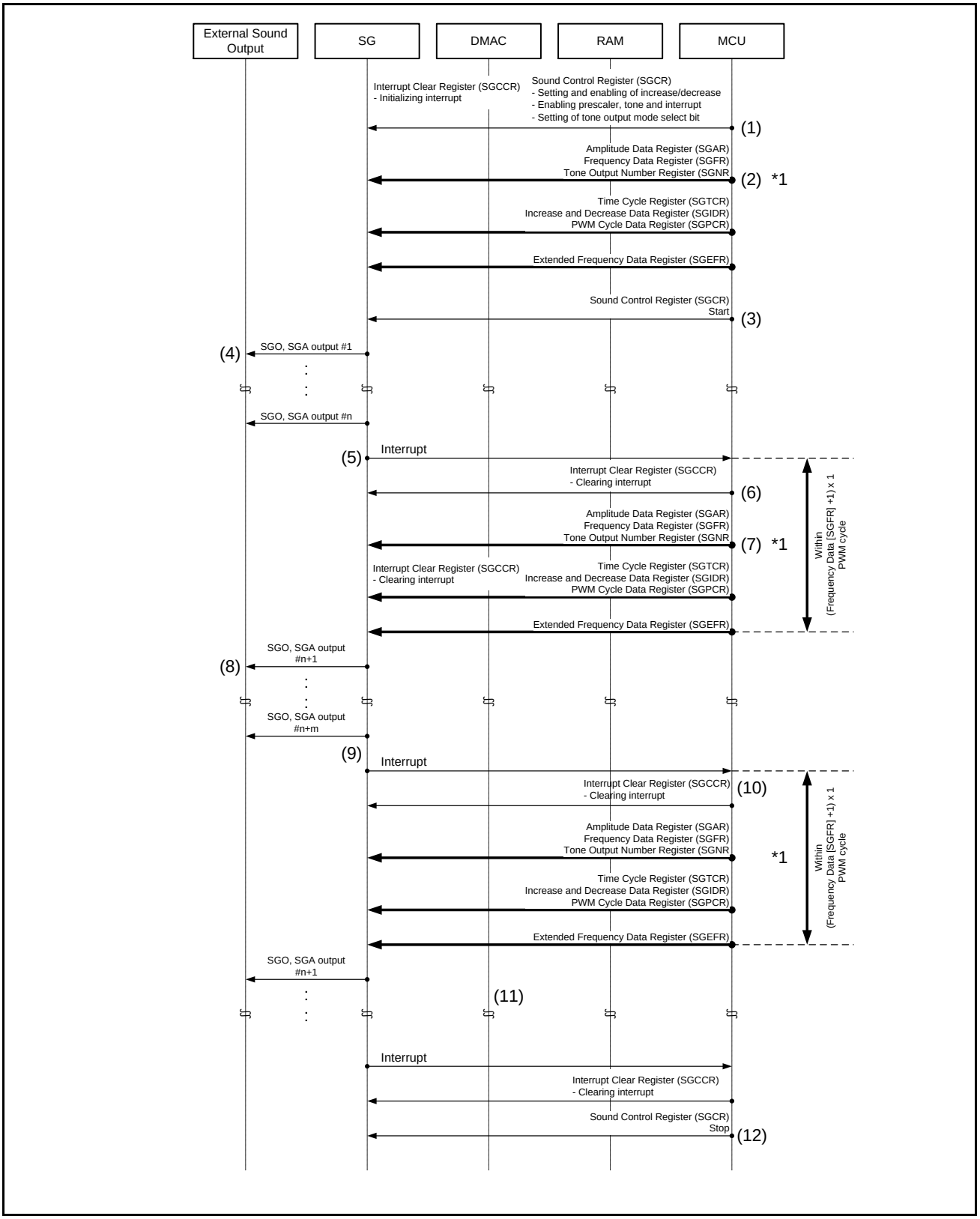
(7) Software writes "0" to the Start bit (SGCR.ST) to stop outputting the sound.

3.5.4. Continuous Operation of Sound Generator by MCU

This section shows the continuous operation flow of the Sound Generator by MCU.

The flow from (7) is different from that of the single operation.

Figure 18-12 Continuous Operation of Sound Generator by MCU



(1) Set the information to control the Sound Generator to the Sound Control Register (SGCR) by software. Initialize the Interrupt status bit (SGCR.INT) and set the Interrupt enable bit (SGCR.INTE).

(2) Set registers "Amplitude Data Register (SGAR)", "Frequency Data Register (SGFR)", "Tone Output Number Register (SGNR)", "Time Cycle Register (SGTCR)", "Increase and Decrease Data Register (SGIDR)", "PWM Cycle Data Register (SGPCR)" and "Extended Frequency Data Register (SGEFR)" by software. (*1: Set only the necessary registers.)

(3) Write "1" to the Start bit (SGCR.ST).

(4) The outputs of SGO and SGA start.

(5) The Tone pulse counter counts the number of tone pulses. When the following conditions are satisfied, the interrupt is generated.
- Tone pulse counter is 0x00
- Decrement counter is 0x00
- At the rising edge of SGO

(6) MCU clears the interrupt.

(7) Set registers "Amplitude Data Register (SGAR)", "Frequency Data Register (SGFR)", "Tone Output Number Register (SGNR)", "Time Cycle Register (SGTCR)", "Increase and Decrease Data Register (SGIDR)", "PWM Cycle Data Register (SGPCR)" and "Extended Frequency Data Register (SGEFR)" by software. (*1: Set only the necessary registers.)

(8) The Sound Generator keeps outputting SGO and SGA, according to the register settings above.

(9) The Tone pulse counter counts the number of tone pulses. When the following conditions are satisfied, the interrupt is generated.
- Tone pulse counter is 0x00
- Decrement counter is 0x00
- At the rising edge of SGO

(10) MCU clears the interrupt.

(11) Repeat the flow from 7 to 11 to continue outputting the sound.

(12) Software writes "0" to the Start bit (SGCR.ST) to stop outputting the sound.

Notes:
- *The software must finish the procedure from step 5 to 7, within the following time. The limit time = (Frequency Data Register [SGFR] + 1) x 1 PWM cycle*
- *In case of switching the selection of increase/decrease, it is needed to write "Increase/decrease setting bit (SGCR.GID)" and the "Automatic increase/decrease enable bit (SGCR.GEN)" in the Sound Control Register (SGCR) within above-mentioned limit time.*

3.5.5. Sound Generator Operation with DMA

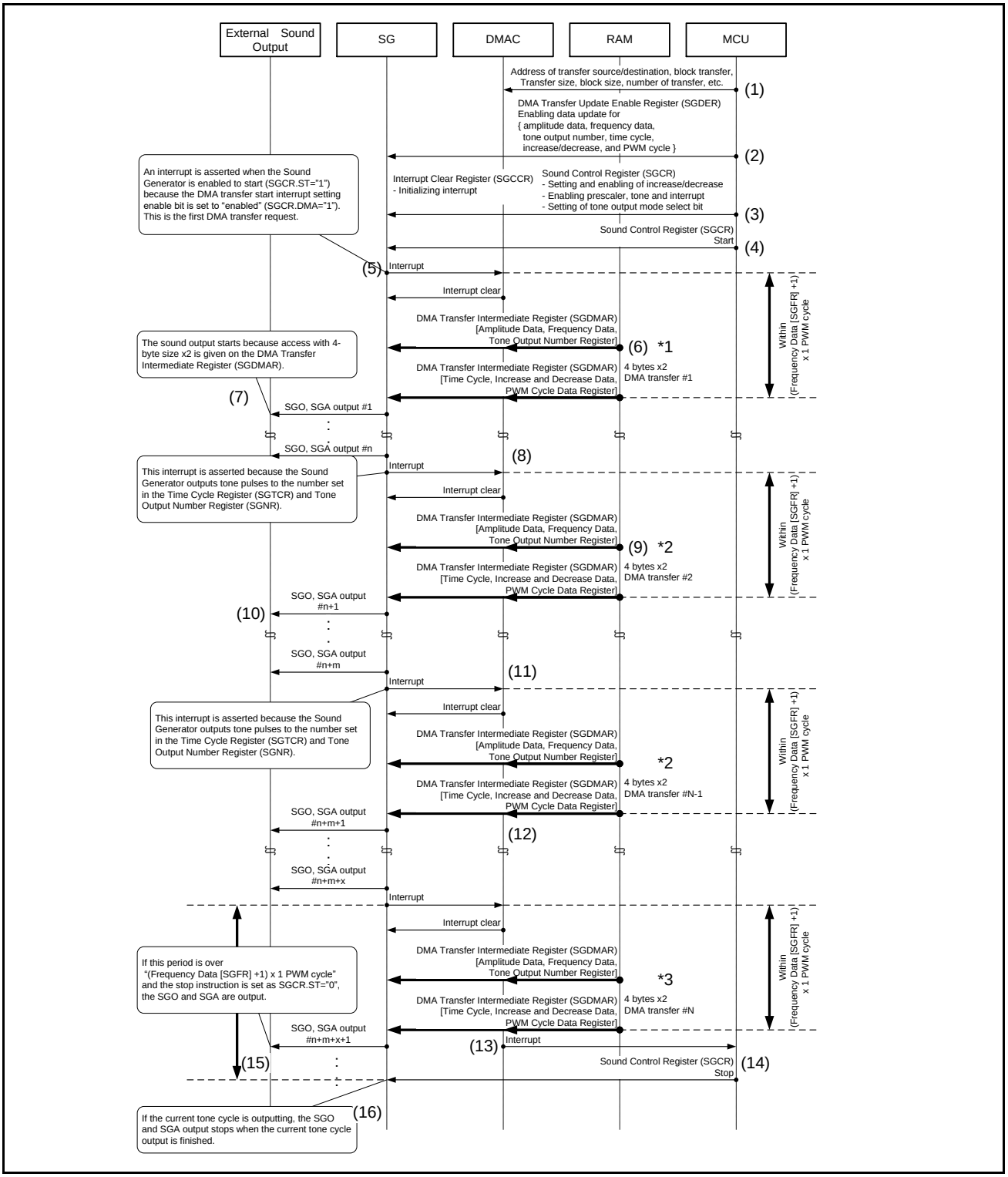
This section shows the flow of the Sound Generator operation with DMA.

DMAC writes a register related to DMA in the Sound Generator. The timing of the first interrupt with DMA is different from the flow by MCU. The DMAC writes functional registers in the Sound Generator, through the DMA Transfer Intermediate Register (SGDMAR) which works as a window register.

Note:
- *Software needs to set the Interrupt enable bit (SGCR.INTE="1") in order to use the Interrupt request (PIRQ) as a DMA transfer request.*

3.5.5.1. In Case of DMA Transfer with 4-Byte Size x 2 is Made N Times

Figure 18-13 Sound Generator Operation with DMA (when in DMA Transfer with 4-Byte Size x 2 is Made N Times)



(1) Software makes initial settings to DMAC which are needed in DMA transfer. The DMA transfer is based on a block data of "4-byte size x 2", and this block can be repeated N times. DMAC sets following registers to prepare for a DMA transfer, through the "DMA Transfer Intermediate Register (SGDMAR)".
- Amplitude Data Register (SGAR)
- Frequency Data Register (SGFR)
- Tone Output Number Register (SGNR)
- Time Cycle Register (SGTCR)
- Increase and Decrease Data Register (SGIDR)
- PWM Cycle Data Register (SGPCR)

The destination address of the DMA transfer is a fixed one on the "DMA Transfer Intermediate Register (SGDMAR)".

(2) Software configures the "DMA Transfer Update Enable Register (SGDER)" to enable the automatic update of the following registers during DMA transfer.
- Amplitude Data Register (SGAR)
- Frequency Data Register (SGFR)
- Tone Output Number Register (SGNR)
- Time Cycle Register (SGTCR)
- Increase and Decrease Data Register (SGIDR)
- PWM Cycle Data Register (SGPCR)

(3) Software initialize Interrupt status bit (SGCR.INT) by writing "1" to Interrupt status clear bit (SGCCR.INTC). Then, the software configures the Sound Control Register (SGCR) in the needed mode, and this must include the following bit operations.
- DMA transfer start interrupt setting enable bit (SGCR.DMA) to "1"(enabled)
- Interrupt enable bit (SGCR.INTE) to "1"(enabled)
- Tone output mode select bit (SGCR.TMS) to "0"(Normal mode)

(4) Write "1" to the Start bit (SGCR.ST).

(5) The interrupt occurs immediately after setting Start bit (SGCR.ST), since the Sound Generator is enabled on DMA transfer (SGCR.DMA="1"). An Interrupt request (PIRQ) is asserted, and this interrupt is used as a DMA transfer request.

(6) DMAC clears the interrupt, and write registers in the Sound Generator through the "DMA Transfer Intermediate Register (SGDMAR)". This operation configures the following registers by 2 steps.
[1st step]
- Amplitude Data Register (SGAR)
- Frequency Data Register (SGFR)
- Tone Output Number Register (SGNR)

[2nd step]
- Time Cycle Register (SGTCR)
- Increase and Decrease Data Register (SGIDR)
- PWM Cycle Data Register (SGPCR)

(*1: DMA block size must to be "4-byte size x 2" for the access to "DMA Transfer Intermediate Register (SGDMAR)")

(7) The outputs of SGO and SGA start, according to the register settings above.

(8) The Tone pulse counter counts the number of tone pulses. When the following conditions are satisfied, the interrupt is generated.
- Tone pulse counter is 0x00
- Decrement counter is 0x00
- At the rising edge of SGO

(9) DMAC clears the interrupt, and write registers in the Sound Generator through the "DMA Transfer Intermediate Register (SGDMAR)".
(*2: DMA block size must to be "4-byte size x 2" for the access to "DMA Transfer Intermediate Register (SGDMAR)")

(10) The Sound Generator keeps outputting SGO and SGA, according to the register settings above.

(11) The Tone pulse counter counts the number of tone pulses. When the following conditions are satisfied, the interrupt is generated.
- Tone pulse counter is 0x00
- Decrement counter is 0x00
- At the rising edge of SGO

DMAC doesn't assert an interrupt to MCU until the DMAC completes all DMA transfer (with 4-byte size x2, N times).

(12) Repeat the flow from 9 to 11 to continue outputting the sound.

(13) When DMAC completes all DMA transfer (with 4-byte size x2, N times), it asserts an interrupt to MCU.

(14) Software writes "0" to the Start bit (SGCR.ST) to stop outputting the sound.
(15) When above operation (14) was made within the following time, the Nth DMA transfer doesn't come to the output of SGO and SGA. The Sound Generator stops driving SGO and SGA just before outputting the data of Nth DMA transfer.
The limit time = (Frequency Data Register [SGFR] + 1) x 1 PWM cycle
(*3: The data of the Nth DMA transfer are written to the Sound Generator, however, they are not output. DMAC issues this Nth DMA transfer only to assert an interrupt toward the MCU.)

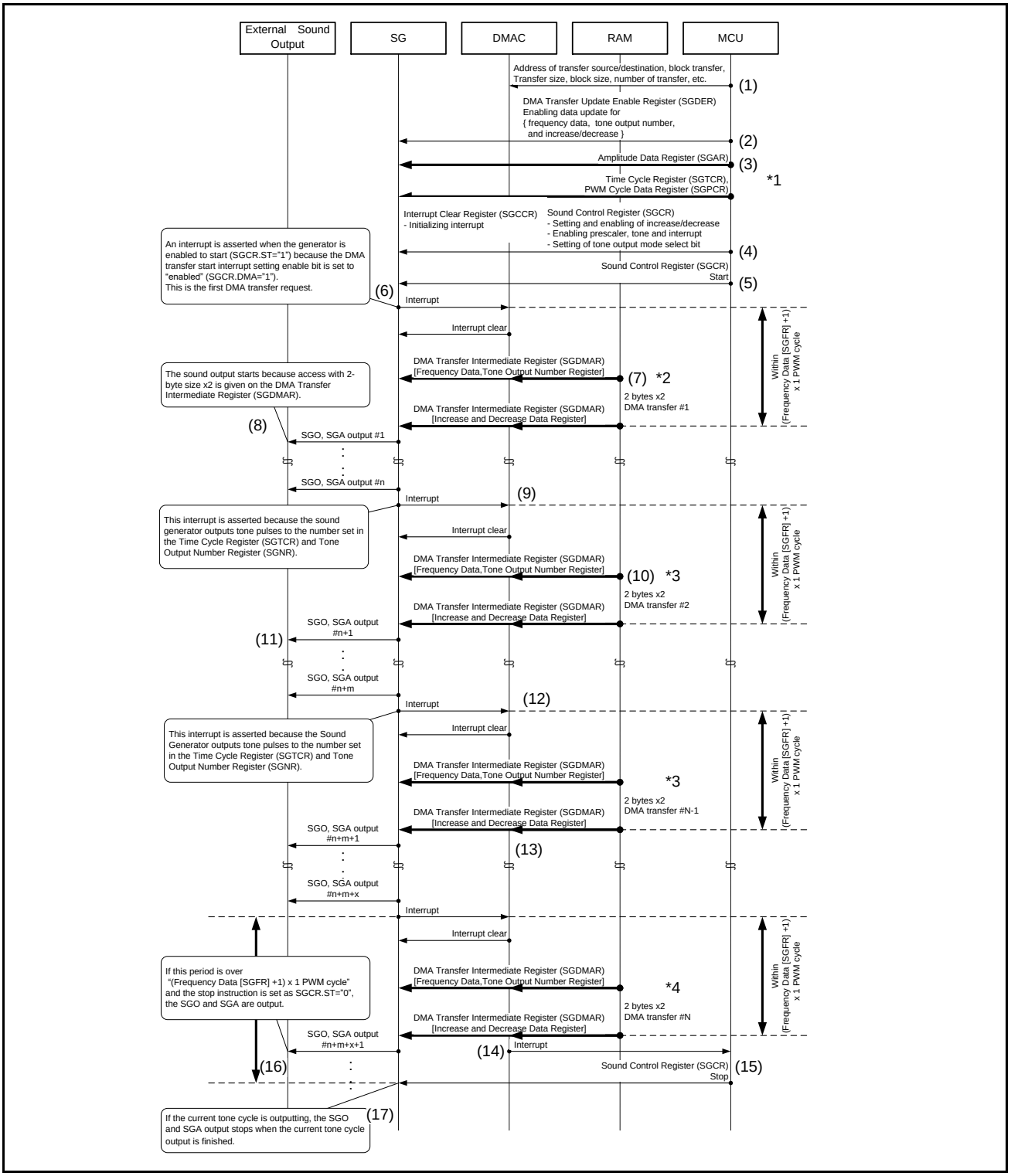
(16) If above (15) is not done within the limit time, the Sound Generator keeps driving SGO and SGA in order to output the data of the Nth DMA transfer. Then, the Sound Generator stops driving SGO and SGA after the end of all data.
(*3: The data of the Nth DMA transfer are written to the Sound Generator, and they are output to the end.)

Notes:

- *The DMAC must finish the sequence from step 5 to 6, within the following time.*
  The limit time = (Frequency Data Register [SGFR] + 1) x 1 PWM cycle
- *The DMA transfer error means the occurrence of delay in the sound data setting. It causes unsteady sound output. In that case, please fix the priority of DMA transfer in the system to finish all data transfer within the limit time.*

3.5.5.2. In Case of DMA Transfer with 2-Byte Size x 2 is Made N Times

Figure 18-14 Sound Generator Operation with DMA (When DMA Transfer with 2-Byte Size x 2 is Made N Times)



(1) Software makes initial settings to DMAC which are needed in DMA transfer. The DMA transfer is based on a block data of "2-byte size x 2", and this block can be repeated N times. DMAC sets following registers to prepare for a DMA transfer, through the "DMA Transfer Intermediate Register (SGDMAR)".
- Frequency Data Register (SGFR)
- Tone Output Number Register (SGNR)
- Increase and Decrease Data Register (SGIDR)
The destination address of the DMA transfer is a fixed one on the "DMA Transfer Intermediate Register (SGDMAR)".

(2) Software configures the "DMA Transfer Update Enable Register (SGDER)" to enable the automatic update of the following registers during DMA transfer.
- Frequency Data Register (SGFR)
- Tone Output Number Register (SGNR)
- Increase and Decrease Data Register (SGIDR)

(3) Software configures to the following registers by a DMA transfer that are not updated.
- Amplitude Data Register (SGAR)
- Time Cycle Register (SGTCR)
- PWM Cycle Data Register (SGPCR)
(*1: Set values to these registers which are not updated by the DMA Transfer Update Enable Register (SGDER).)

(4) Software initialize Interrupt status bit (SGCR.INT) by writing "1" to Interrupt status clear bit (SGCCR.INTC). Then, the software configures the Sound Control Register (SGCR) in the needed mode, and this must include the following bit operations.
- DMA transfer start interrupt setting enable bit (SGCR.DMA) to "1"(enabled)
- Interrupt enable bit (SGCR.INTE) to "1"(enabled)
- Tone output mode select bit (SGCR.TMS) to "0"(Normal mode)

(5) Write "1" to the Start bit (SGCR.ST).

(6) The interrupt occurs immediately after setting Start bit (SGCR.ST), since the Sound Generator is enabled on DMA transfer (SGCR.DMA="1"). An Interrupt request (PIRQ) is asserted, and this interrupt is used as a DMA transfer request.

(7) DMAC clears the interrupt, and write registers in the Sound Generator through the "DMA Transfer Intermediate Register (SGDMAR)". This operation configures the following registers by 2 steps.
[1st step]
- Frequency Data Register (SGFR)
- Tone Output Number Register (SGNR)
[2nd step]
- Increase and Decrease Data Register (SGIDR)
(*2: DMA block size must to be "2-byte size x 2" for the access to "DMA Transfer Intermediate Register (SGDMAR)")

(8) The outputs of SGO and SGA start, according to the register settings above.

(9) The Tone pulse counter counts the number of tone pulses. When the following conditions are satisfied, the interrupt is generated.
- Tone pulse counter is 0x00
- Decrement counter is 0x00
- At the rising edge of SGO

(10) DMAC clears the interrupt, and write registers in the Sound Generator through the "DMA Transfer Intermediate Register (SGDMAR)".
(*3: DMA block size must to be "2-byte size x 2" for the access to "DMA Transfer Intermediate Register (SGDMAR)")

(11) The Sound Generator keeps outputting SGO and SGA, according to the register settings above.

(12) The Tone pulse counter counts the number of tone pulses. When the following conditions are satisfied, the interrupt is generated.
- Tone pulse counter is 0x00
- Decrement counter is 0x00
- At the rising edge of SGO

DMAC doesn't assert an interrupt to MCU until the DMAC completes all DMA transfer (with 2-byte size x2, N times).

(13) Repeat the flow from 10 to 12 to continue outputting the sound.

(14) When DMAC completes all DMA transfer (with 2-byte size x2, N times), it asserts an interrupt to MCU.

(15) Software writes "0" to the Start bit (SGCR.ST) to stop outputting the sound.

(16) When above operation (15) was made within the following time, the Nth DMA transfer doesn't come to the output of SGO and SGA. The Sound Generator stops driving SGO and SGA just before outputting the data of Nth DMA transfer.
The limit time = (Frequency Data Register [SGFR] + 1) x 1 PWM cycle
(*4: The data of the Nth DMA transfer are written to the Sound Generator, however, they are not output. DMAC issues this Nth DMA transfer only to assert an interrupt toward the MCU.)

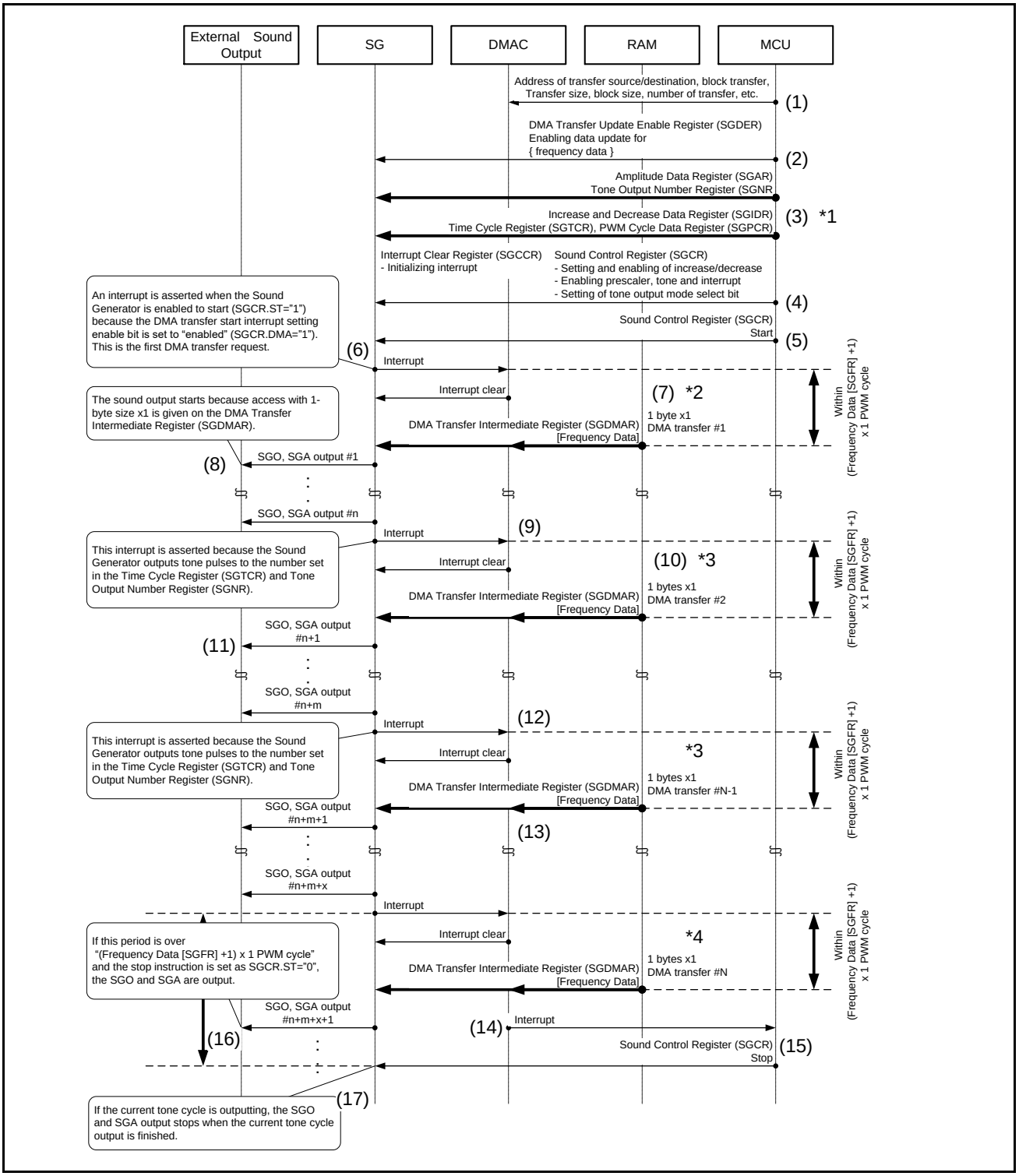
(17) If above (16) is not done within the limit time, the Sound Generator keeps driving SGO and SGA in order to output the data of the Nth DMA transfer. Then, the Sound Generator stops driving SGO and SGA after the end of all data.
(*4: The data of the Nth DMA transfer are written to the Sound Generator, and they are output to the end.)

Notes:

- *The DMAC must finish the sequence from step 6 to 7, within the following time.*
  The limit time = (Frequency Data Register [SGFR] + 1) x 1 PWM cycle
- *The DMA transfer error means the occurrence of delay in the sound data setting. It causes unsteady sound output. In that case, please fix the priority of DMA transfer in the system to finish all data transfer within the limit time.*

3.5.5.3. In Case of DMA Transfer with 1-Byte Size x 1 is Made N Times

Figure 18-15 Sound Generator Operation with DMA (when in DMA Transfer with 1-Byte Size x 1 is Made N Times)



(1) Software makes initial settings to DMAC which are needed in DMA transfer. The DMA transfer is based on a block data of "1-byte size x 1", and this block can be repeated N times. DMAC sets following registers to prepare for a DMA transfer, through the "DMA Transfer Intermediate Register (SGDMAR)".
- Frequency Data Register (SGFR)
The destination address of the DMA transfer is a fixed one on the "DMA Transfer Intermediate Register (SGDMAR)".

(2) Software configures the "DMA Transfer Update Enable Register (SGDER)" to enable the automatic update of the following registers during DMA transfer.
- Frequency Data Register (SGFR)

(3) Software configures to the following registers by a DMA transfer that are not updated.
- Amplitude Data Register (SGAR)
- Tone Output Number Register (SGNR)
- Time Cycle Register (SGTCR)
- Increase and Decrease Data Register (SGIDR)
- PWM Cycle Data Register (SGPCR)
(*1: Set values to these registers which are not updated by the DMA Transfer Update Enable Register (SGDER).)

(4) Software initialize Interrupt status bit (SGCR.INT) by writing "1" to Interrupt status clear bit (SGCCR.INTC). Then, the software configures the Sound Control Register (SGCR) in the needed mode, and this must include the following bit operations.
- DMA transfer start interrupt setting enable bit (SGCR.DMA) to "1"(enabled)
- Interrupt enable bit (SGCR.INTE) to "1"(enabled)
- Tone output mode select bit (SGCR.TMS) to "0"(Normal mode)

(5) Write "1" to the Start bit (SGCR.ST).

(6) The interrupt occurs immediately after setting Start bit (SGCR.ST), since the Sound Generator is enabled on DMA transfer (SGCR.DMA="1"). An Interrupt request (PIRQ) is asserted, and this interrupt is used as a DMA transfer request.

(7) DMAC clears the interrupt, and write registers in the Sound Generator through the "DMA Transfer Intermediate Register (SGDMAR)". This operation configures the following register.
- Frequency Data Register (SGFR)
(*2: DMA block size must to be "1-byte size x 1" for the access to "DMA Transfer Intermediate Register (SGDMAR)")

(8) The outputs of SGO and SGA start, according to the register settings above.

(9) The Tone pulse counter counts the number of tone pulses. When the following conditions are satisfied, the interrupt is generated.
- Tone pulse counter is 0x00
- Decrement counter is 0x00
- At the rising edge of SGO

(10) DMAC clears the interrupt, and write registers in the Sound Generator through the "DMA Transfer Intermediate Register (SGDMAR)".
(*3: DMA block size must to be "1-byte size x 1" for the access to "DMA Transfer Intermediate Register (SGDMAR)")

(11) The Sound Generator keeps outputting SGO and SGA, according to the register settings above.

(12) The Tone pulse counter counts the number of tone pulses. When the following conditions are satisfied, the interrupt is generated.
- Tone pulse counter is 0x00
- Decrement counter is 0x00
- At the rising edge of SGO
DMAC doesn't assert an interrupt to MCU until the DMAC completes all DMA transfer (with 1-byte size x1, N times).

(13) Repeat the flow from 10 to 12 to continue outputting the sound.

(14) When DMAC completes all DMA transfer (with 1-byte size x1, N times), it asserts an interrupt to MCU.

(15) Software writes "0" to the Start bit (SGCR.ST) to stop outputting the sound.

(16) When above operation (15) was made within the following time, the Nth DMA transfer doesn't come to the output of SGO and SGA. The Sound Generator stops driving SGO and SGA just before outputting the data of Nth DMA transfer.
The limit time = (Frequency Data Register [SGFR] + 1) x 1 PWM cycle
(*4: The data of the Nth DMA transfer are written to the Sound Generator, however, they are not output. DMAC issues this Nth DMA transfer only to assert an interrupt toward the MCU.)

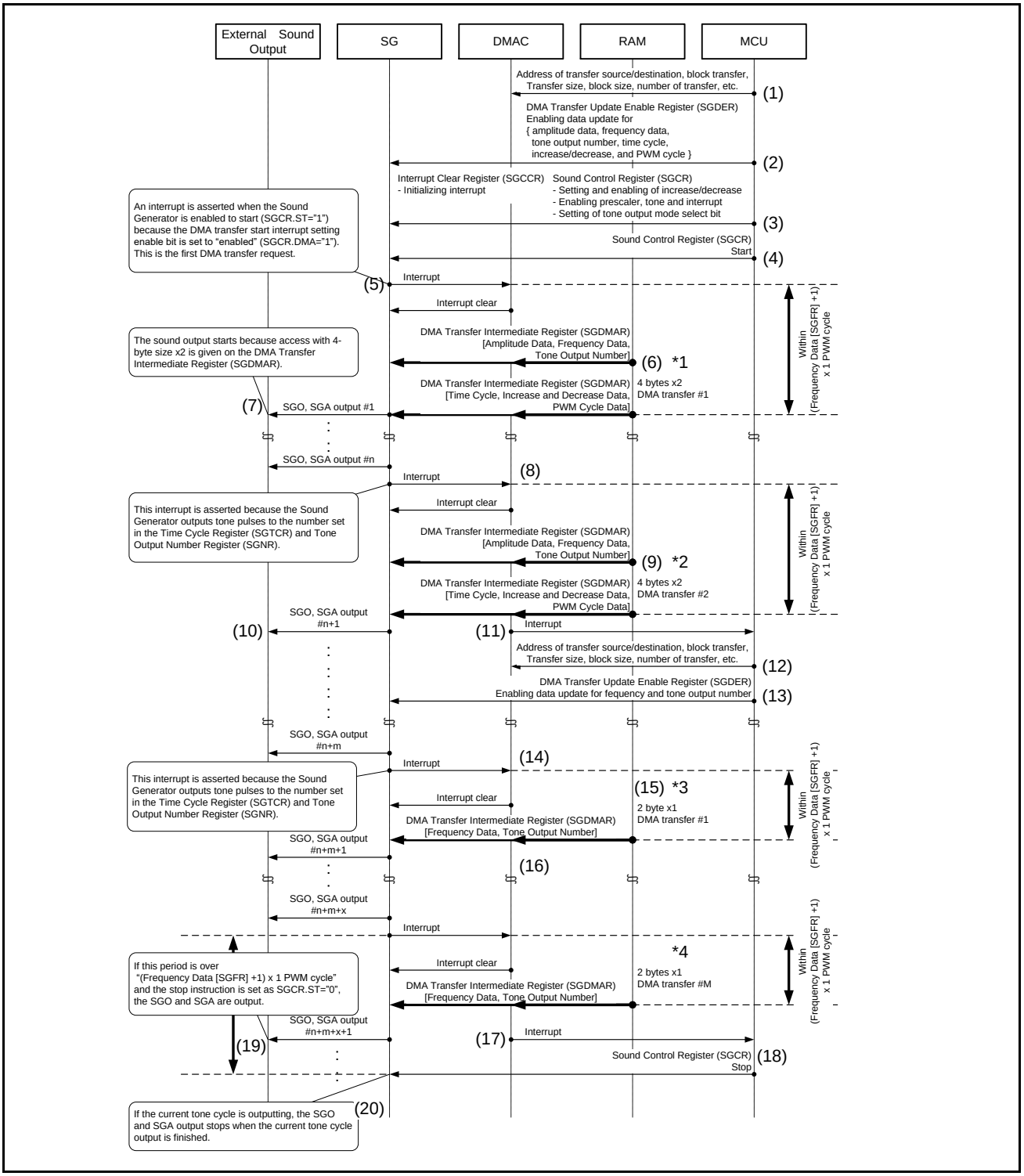
(17) If above (16) is not done within the limit time, the Sound Generator keeps driving SGO and SGA in order to output the data of the Nth DMA transfer. Then, the Sound Generator stops driving SGO and SGA after the end of all data.
(*4: The data of the Nth DMA transfer are written to the Sound Generator, and they are output to the end.)

Notes:

- *The DMAC must finish the sequence from step 6 to 7, within the following time.*
  The limit time = (Frequency Data Register [SGFR] + 1) x 1 PWM cycle
- *The DMA transfer error means the occurrence of delay in the sound data setting. It causes unsteady sound output. In that case, please fix the priority of DMA transfer in the system to finish all data transfer within the limit time.*

3.5.5.4. In Case of DMA Transfer with 4-Byte Size x 2 is Made N Times and DMA Transfer with 2-Byte Size x 1 is Made M Times (The Case the Transfer Byte Size is Changed while Sound Outputting)

Figure 18-16 Sound Generator Operation with DMA
(The Case the Transfer Byte Size is Changed while Sound is Being Output)



(1) Software makes initial settings to DMAC which are needed in DMA transfer. The DMA transfer is based on a block data of "4-byte size x 2", and this block can be repeated N times. DMAC sets following registers to prepare for a DMA transfer, through the "DMA Transfer Intermediate Register (SGDMAR)".
- Amplitude Data Register (SGAR)
- Frequency Data Register (SGFR)
- Tone Output Number Register (SGNR)
- Time Cycle Register (SGTCR)
- Increase and Decrease Data Register (SGIDR)
- PWM Cycle Data Register (SGPCR)

The destination address of the DMA transfer is a fixed one on the "DMA Transfer Intermediate Register (SGDMAR)".

(2) Software configures the "DMA Transfer Update Enable Register (SGDER)" to enable the automatic update of the following registers during DMA transfer.
- Amplitude Data Register (SGAR)
- Frequency Data Register (SGFR)
- Tone Output Number Register (SGNR)
- Time Cycle Register (SGTCR)
- Increase and Decrease Data Register (SGIDR)
- PWM Cycle Data Register (SGPCR)

(3) Software initialize Interrupt status bit (SGCR.INT) by writing "1" to Interrupt status clear bit (SGCCR.INTC). Then, the software configures the Sound Control Register (SGCR) in the needed mode, and this must include the following bit operations.
- DMA transfer start interrupt setting enable bit (SGCR.DMA) to "1"(enabled)
- Interrupt enable bit (SGCR.INTE) to "1"(enabled)
- Tone output mode select bit (SGCR.TMS) to "0"(Normal mode)

(4) Write "1" to the Start bit (SGCR.ST).

(5) The interrupt occurs immediately after setting Start bit (SGCR.ST), since the Sound Generator is enabled on DMA transfer (SGCR.DMA="1"). An Interrupt request (PIRQ) is asserted, and this interrupt is used as a DMA transfer request.

(6) DMAC clears the interrupt, and write registers in the Sound Generator through the "DMA Transfer Intermediate Register (SGDMAR)". This operation configures the following registers by 2 steps.
[1st step]
- Amplitude Data Register (SGAR)
- Frequency Data Register (SGFR)
- Tone Output Number Register (SGNR)

[2nd step]
- Time Cycle Register (SGTCR)
- Increase and Decrease Data Register (SGIDR)
- PWM Cycle Data Register (SGPCR)

(*1: DMA block size must to be "4-byte size x 2" for the access to "DMA Transfer Intermediate Register (SGDMAR)")

(7) The outputs of SGO and SGA start, according to the register settings above.

(8) The Tone pulse counter counts the number of tone pulses. When the following conditions are satisfied, the interrupt is generated.
- Tone pulse counter is 0x00
- Decrement counter is 0x00
- At the rising edge of SGO

(9) DMAC clears the interrupt, and write registers in the Sound Generator through the "DMA Transfer Intermediate Register (SGDMAR)".
(*2: DMA block size must to be "4-byte size x 2" for the access to "DMA Transfer Intermediate Register (SGDMAR)")

(10) The Sound Generator keeps outputting SGO and SGA, according to the register settings above.

(11) When DMAC completes all DMA transfer (with 4-byte size x2, N times), it asserts an interrupt to MCU.

(12) Software makes settings to those registers which are needed in DMA transfer. The DMA transfer is based on a block data of "2-byte size x 1", and this block can be repeated M times. DMAC sets following registers to prepare for a DMA transfer, through the "DMA Transfer Intermediate Register (SGDMAR)".

- Frequency Data Register (SGFR)
- Tone Output Number Register (SGNR)
The destination address of the DMA transfer is a fixed one on the "DMA Transfer Intermediate Register (SGDMAR)".

(13) Software configures the "DMA Transfer Update Enable Register (SGDER)" to enable the automatic update of the following registers during DMA transfer.
- Frequency Data Register (SGFR)
- Tone Output Number Register (SGNR)

(14) The Tone pulse counter counts the number of tone pulses. When the following conditions are satisfied, the interrupt is generated.
- Tone pulse counter is 0x00
- Decrement counter is 0x00
- At the rising edge of SGO

(15) DMAC clears the interrupt, and write registers in the Sound Generator through the "DMA Transfer Intermediate Register (SGDMAR)".
(*3: DMA block size must to be "2-byte size x 1" for the access to "DMA Transfer Intermediate Register (SGDMAR)")

(16) Repeat the flow from 14 to 15 to continue outputting the sound.

(17) When DMAC completes all DMA transfer (with 2-byte size x1, M times), it asserts an interrupt to MCU.

(18) Software writes "0" to the Start bit (SGCR.ST) to stop outputting the sound.

(19) When above operation (18) was made within the following time, the Mth DMA transfer doesn't come to the output of SGO and SGA. The Sound Generator stops driving SGO and SGA just before outputting the data of Nth DMA transfer.
The limit time = (Frequency Data Register [SGFR] + 1) x 1 PWM cycle
(*4: The data of the Mth DMA transfer are written to the Sound Generator, however, they are not output. DMAC issues this Mth DMA transfer only to assert an interrupt toward the MCU.)

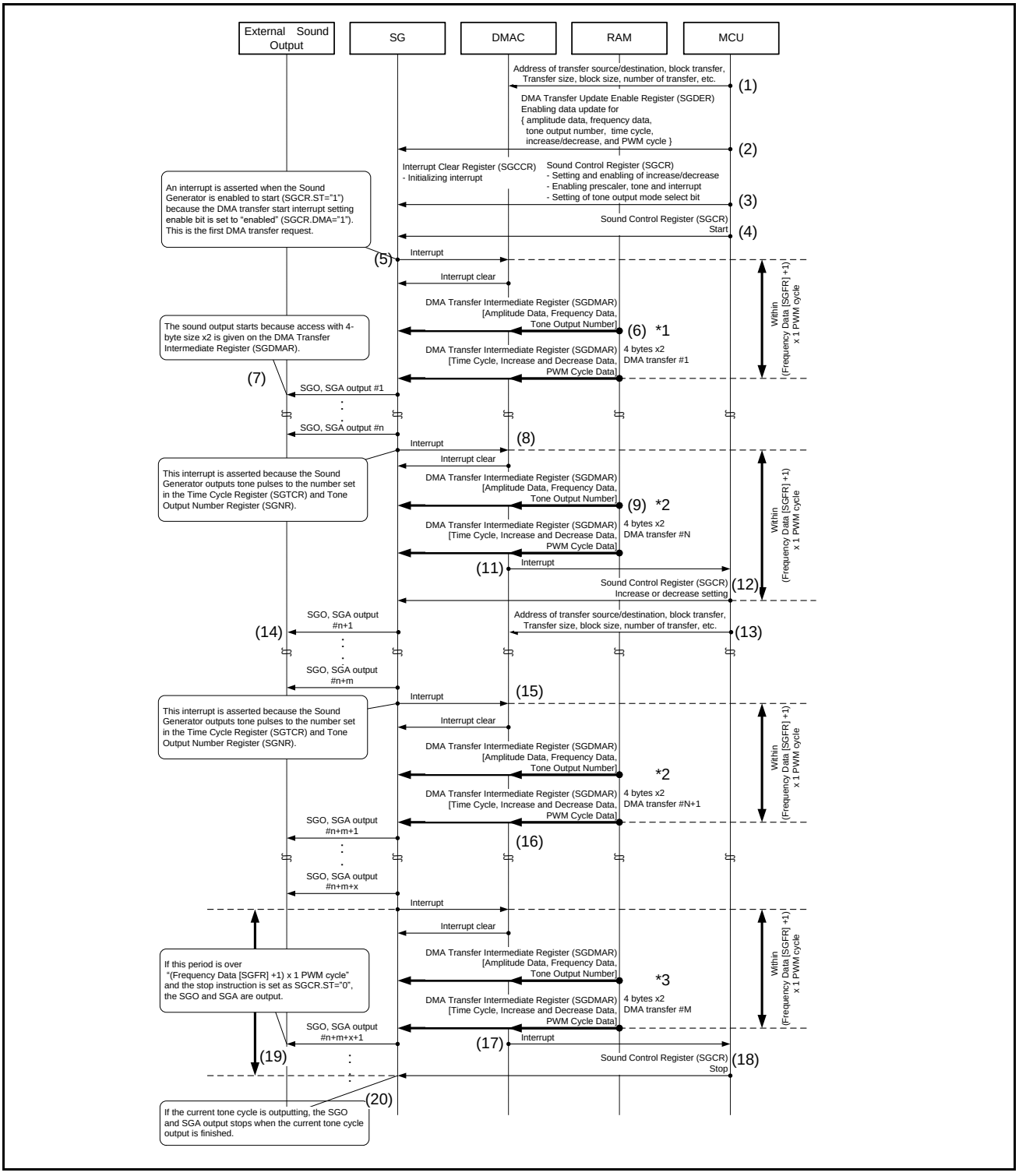
(20) If above (19) is not done within the limit time, the Sound Generator keeps driving SGO and SGA in order to output the data of the Mth DMA transfer. Then, the Sound Generator stops driving SGO and SGA after the end of all data.
(*4: The data of the Mth DMA transfer are written to the Sound Generator, and they are output to the end.)

Notes:

- *The DMAC must finish the sequence from step 5 to 6, within the following time.*
  The limit time = (Frequency Data Register [SGFR] + 1) x 1 PWM cycle
- *The DMA transfer error means the occurrence of delay in the sound data setting. It causes unsteady sound output. In that case, please fix the priority of DMA transfer in the system to finish all data transfer within the limit time.*

3.5.5.5. In Case of DMA Transfer with 4-Byte Size x 2 is Made N Times and DMA Transfer with 4-Byte Size x 2 is Made M Times (The Case the Transfer Byte Size and the Increase/Decrease Setting is Changed while Sound Outputting)

Figure 18-17 Sound Generator Operation with DMA
(The Case the Transfer Byte Size and Increase/Decrease Setting is Changed while Sound is Being Output)



(1) Software makes initial settings to DMAC which are needed in DMA transfer. The DMA transfer is based on a block data of "4-byte size x 2", and this block can be repeated N times. DMAC sets following registers to prepare for a DMA transfer, through the "DMA Transfer Intermediate Register (SGDMAR)".
- Amplitude Data Register (SGAR)
- Frequency Data Register (SGFR)
- Tone Output Number Register (SGNR)
- Time Cycle Register (SGTCR)
- Increase and Decrease Data Register (SGIDR)
- PWM Cycle Data Register (SGPCR)

The destination address of the DMA transfer is a fixed one on the "DMA Transfer Intermediate Register (SGDMAR)".

(2) Software configures the "DMA Transfer Update Enable Register (SGDER)" to enable the automatic update of the following registers during DMA transfer.
- Amplitude Data Register (SGAR)
- Frequency Data Register (SGFR)
- Tone Output Number Register (SGNR)
- Time Cycle Register (SGTCR)
- Increase and Decrease Data Register (SGIDR)
- PWM Cycle Data Register (SGPCR)

(3) Software initialize Interrupt status bit (SGCR.INT) by writing "1" to Interrupt status clear bit (SGCCR.INTC). Then, the software configures the Sound Control Register (SGCR) in the needed mode, and this must include the following bit operations.
- DMA transfer start interrupt setting enable bit (SGCR.DMA) to "1"(enabled)
- Interrupt enable bit (SGCR.INTE) to "1"(enabled)
- Tone output mode select bit (SGCR.TMS) to "0"(Normal mode)

(4) Write "1" to the Start bit (SGCR.ST).

(5) The interrupt occurs immediately after setting Start bit (SGCR.ST), since the Sound Generator is enabled on DMA transfer (SGCR.DMA="1"). An Interrupt request (PIRQ) is asserted, and this interrupt is used as a DMA transfer request.

(6) DMAC clears the interrupt, and write registers in the Sound Generator through the "DMA Transfer Intermediate Register (SGDMAR)". This operation configures the following registers by 2 steps.
[1st step]
- Amplitude Data Register (SGAR)
- Frequency Data Register (SGFR)
- Tone Output Number Register (SGNR)

[2nd step]
- Time Cycle Register (SGTCR)
- Increase and Decrease Data Register (SGIDR)
- PWM Cycle Data Register (SGPCR)

(*1: DMA block size must to be "4-byte size x 2" for the access to "DMA Transfer Intermediate Register (SGDMAR)")

(7) The outputs of SGO and SGA start, according to the register settings above.

(8) The Tone pulse counter counts the number of tone pulses. When the following conditions are satisfied, the interrupt is generated.
- Tone pulse counter is 0x00
- Decrement counter is 0x00
- At the rising edge of SGO

(9) DMAC clears the interrupt, and write registers in the Sound Generator through the "DMA Transfer Intermediate Register (SGDMAR)".
(*2: DMA block size must to be "4-byte size x 2" for the access to "DMA Transfer Intermediate Register (SGDMAR)")

(10) The Sound Generator keeps outputting SGO and SGA, according to the register settings above.

(11) When DMAC completes all DMA transfer (with 4-byte size x2, N times), it asserts an interrupt to MCU.

(12) Change the increase/decrease setting of the Sound Control Register (SGCR) by software (SGCR.GID).

(13) Software makes settings to the DMAC. The DMA transfer is based on a block data of "4-byte size x 2 cycle", and this block can be repeated M times.

(14) The Sound Generator keeps outputting SGO and SGA.

(15) The Tone pulse counter counts the number of tone pulses. When the following conditions are satisfied, the interrupt is generated.
- Tone pulse counter is 0x00
- Decrement counter is 0x00
- At the rising edge of SGO

Then, DMAC clears the interrupt, and write registers in the Sound Generator through the "DMA Transfer Intermediate Register (SGDMAR)".
(*2: DMA block size must to be "4-byte size x 2" for the access to "DMA Transfer Intermediate Register (SGDMAR)")

(16) Repeat the step 15 to continue outputting the sound.

(17) When DMAC completes all DMA transfer (with 4-byte size x2, M times), it asserts an interrupt to MCU.

(18) Software writes "0" to the Start bit (SGCR.ST) to stop outputting the sound.

(19) When above operation (18) was made within the following time, the Mth DMA transfer doesn't come to the output of SGO and SGA. The Sound Generator stops driving SGO and SGA just before outputting the data of Nth DMA transfer.
The limit time = (Frequency Data Register [SGFR] + 1) x 1 PWM cycle
(*3: The data of the Mth DMA transfer are written to the Sound Generator, however, they are not output. DMAC issues this Mth DMA transfer only to assert an interrupt toward the MCU.)

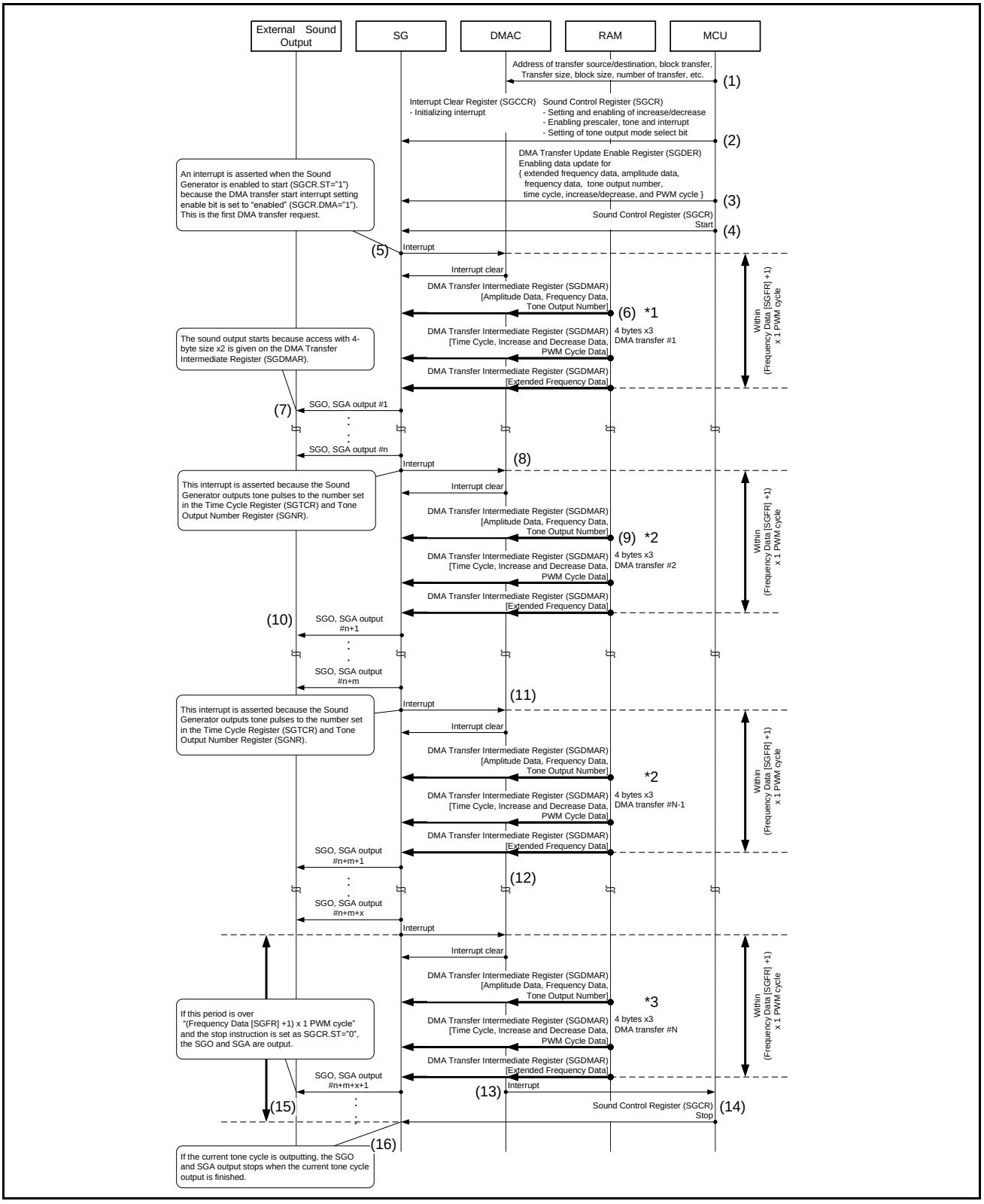
(20) If above (19) is not done within the limit time, the Sound Generator keeps driving SGO and SGA in order to output the data of the Mth DMA transfer. Then, the Sound Generator stops driving SGO and SGA after the end of all data.
(*3: The data of the Mth DMA transfer are written to the Sound Generator, and they are output to the end.)

Notes:

- *The DMAC must finish the sequence from step 5 to 6, within the following time.*
  The limit time = (Frequency Data Register [SGFR] + 1) x 1 PWM cycle
- *The DMA transfer error means the occurrence of delay in the sound data setting. It causes unsteady sound output. In that case, please fix the priority of DMA transfer in the system to finish all data transfer within the limit time.*
- *The DMAC must finish the sequence from step 8 to 12 (including the time for changing "the increase/decrease setting"), within the following time. The increase/decrease setting change comes into effect on the sound output from step 14, using the data of the Mth transfer with 4-byte size x 2.*
  The limit time = (Frequency Data Register [SGFR] + 1) x 1 PWM cycle

3.5.5.6. In Case of DMA Transfer with 4-Byte Size x 3 is Made N Times

Figure 18-18 Sound Generator Operation with DMA (when in DMA Transfer with 4-Byte Size x 3 is Made N Times)



(1) Software makes initial settings to DMAC which are needed in DMA transfer. The DMA transfer is based on a block data of "4-byte size x 3", and this block can be repeated N times. DMAC sets following registers to prepare for a DMA transfer, through the "DMA Transfer Intermediate Register (SGDMAR)".

- Amplitude Data Register (SGAR)
- Frequency Data Register (SGFR)
- Tone Output Number Register (SGNR)
- Time Cycle Register (SGTCR)
- Increase and Decrease Data Register (SGIDR)
- PWM Cycle Data Register (SGPCR)
- Extended Frequency Data Register (SGEFR)

The destination address of the DMA transfer is a fixed one on the "DMA Transfer Intermediate Register (SGDMAR)".

(2) Software initialize Interrupt status bit (SGCR.INT) by writing "1" to Interrupt status clear bit (SGCCR.INTC). Then, the software configures the Sound Control Register (SGCR) in the needed mode, and this must include the following bit operations.

- DMA transfer start interrupt setting enable bit (SGCR.DMA) to "1"(enabled)
- Interrupt enable bit (SGCR.INTE) to "1"(enabled)
- Tone output mode select bit (SGCR.TMS) to "1"(Extended mode)

(3) Software configures the "DMA Transfer Update Enable Register (SGDER)" to enable the automatic update of the following registers during DMA transfer.

- Amplitude Data Register (SGAR)
- Frequency Data Register (SGFR)
- Tone Output Number Register (SGNR)
- Time Cycle Register (SGTCR)
- Increase and Decrease Data Register (SGIDR)
- PWM Cycle Data Register (SGPCR)
- Extended Frequency Data Register (SGEFR)

(4) Write "1" to the Start bit (SGCR.ST).

(5) The interrupt occurs immediately after setting Start bit (SGCR.ST), since the Sound Generator is enabled on DMA transfer (SGCR.DMA="1"). An Interrupt request (PIRQ) is asserted, and this interrupt is used as a DMA transfer request.

(6) DMAC clears the interrupt, and write registers in the Sound Generator through the "DMA Transfer Intermediate Register (SGDMAR)". This operation configures the following registers by 3 steps.

[1st step]

- Amplitude Data Register (SGAR)
- Frequency Data Register (SGFR)
- Tone Output Number Register (SGNR)

[2nd step]

- Time Cycle Register (SGTCR)
- Increase and Decrease Data Register (SGIDR)
- PWM Cycle Data Register (SGPCR)

[3rd step]

- Extended Frequency Data Register (SGEFR)

(*1: DMA block size must to be "4-byte size x 3" for the access to "DMA Transfer Intermediate Register (SGDMAR)")

(7) The outputs of SGO and SGA start, according to the register settings above.

(8) The Tone pulse counter counts the number of tone pulses. When the following conditions are satisfied, the interrupt is generated.

- Tone pulse counter is 0x00

- Decrement counter is 0x00

- At the rising edge of SGO

(9) DMAC clears the interrupt, and write registers in the Sound Generator through the "DMA Transfer Intermediate Register (SGDMAR)".

(*2: DMA block size must to be "4-byte size x 3" for the access to "DMA Transfer Intermediate Register (SGDMAR)")

(10) The Sound Generator keeps outputting SGO and SGA, according to the register settings above.

(11) The Tone pulse counter counts the number of tone pulses. When the following conditions are satisfied, the interrupt is generated.

- Tone pulse counter is 0x00

- Decrement counter is 0x00

- At the rising edge of SGO

DMAC doesn't assert an interrupt to MCU until the DMAC completes all DMA transfer (with 4-byte size x3, N times).

(12) Repeat the flow from 9 to 11 to continue outputting the sound.

(13) When DMAC completes all DMA transfer (with 4-byte size x3, N times), it asserts an interrupt to MCU.

(14) Software writes "0" to the Start bit (SGCR.ST) to stop outputting the sound.

(15) When above operation (14) was made within the following time, the Nth DMA transfer doesn't come to the output of SGO and SGA. The Sound Generator stops driving SGO and SGA just before outputting the data of Nth DMA transfer.

The limit time = (Frequency Data Register [SGFR] + 1) x 1 PWM cycle

(*3: The data of the Nth DMA transfer are written to the Sound Generator, however, they are not output. DMAC issues this Nth DMA transfer only to assert an interrupt toward the MCU.)

(16) If above (15) is not done within the limit time, the Sound Generator keeps driving SGO and SGA in order to output the data of the Nth DMA transfer. Then, the Sound Generator stops driving SGO and SGA after the end of all data.

(*3: The data of the Nth DMA transfer are written to the Sound Generator, and they are output to the end.)

Notes:

- *The DMAC must finish the sequence from step 5 to 6, within the following time.*
  The limit time = (Frequency Data Register [SGFR] + 1) x 1 PWM cycle
- *The DMA transfer error means the occurrence of delay in the sound data setting. It causes unsteady sound output. In that case, please fix the priority of DMA transfer in the system to finish all data transfer within the limit time.*

3.6. Interrupt of Sound Generator

The interrupt from Sound Generator comes from the judgment of counting tone pulses.

Interrupt of Sound Generator

The interrupt control bit and its factor is shown in the following table.

Table 18-6 The Interrupt Bit of Sound Generator and Its Factor

	Interrupt bit of Sound Generator (singular)
Interrupt status bit	Register: Sound Control Register (SGCR) Bit: Interrupt status bit (INT: bit1)
Interrupt enable bit	Register: Sound Control Register (SGCR) Bit: Interrupt enable bit (INTE: bit2)
Interrupt factor	Tone pulse count is greater than or equal to ((Time Cycle Register [SGTCR] + 1) x (Tone Output Number Register [SGNR] + 1))

During the tone pulses are counted, when the count reaches the multiple of "Time Cycle Register [SGTCR] + 1" and "Tone Output Number Register [SGNR] + 1", an interrupt flag (SGCR.INT) is set to"1". When the interrupt is enabled by the setting of the Interrupt enable bit (SGCR.INTE), an Interrupt request (PIRQ) is subsequently asserted (to be "H").

4. Registers

This chapter shows the list of registers in Sound Generator.

Registers of Sound Generator

Table 18-7 The List of Registers of Sound Generator

Contracted Name	Register Name	Reference
SGDER	DMA Transfer Update Enable Register	4.1
SGCR	Sound Control Register	4.2
SGAR	Amplitude Data Register	4.3
SGFR	Frequency Data Register	4.4
SGNR	Tone Output Number Register	4.5
SGTCR	Time Cycle Register	4.6
SGIDR	Increase and Decrease Data Register	4.7
SGPCR	PWM Cycle Data Register	4.8
SGEFR	Extended Frequency Data Register	4.9
SGDMAR	DMA Transfer Intermediate Register	4.10
SGCCR	Interrupt Clear Register	4.11

The Meaning of the Register Bit Property Code

Table 18-8 The Meaning of the Register Bit Property Code

R	Read-Only Register. Only Read access is available, and Write access is ignored.
W	Write-Only Register. Only Write access is available, and Read access makes nothing.
R/W	Read/Write Register. Both Read access and Write access are available.
-	Both Read access and Write access make nothing.

■ **Register Offset Address Map**

OFFSET_ADDRESS	REGISTER_NAME				ACCESS_SIZE
	+3	+2	+1	+0	
0x00000000	Reserved	SGDER	SGCR		B, H, W
0x00000004	SGAR		SGFR	SGNR	B, H, W
0x00000008	SGTCR	SGIDR	SGPCR		B, H, W
0x0000000C	Reserved			SGEFR	B, H, W
0x00000010	SGDMAR				B, H, W
0x00000014	Reserved		SGCCR		H, W

4.1. DMA Transfer Update Enable Register (SGDER)

The DMA Transfer Update Enable Register (SGDER) is to enable the update of the registers (SGAR, SGFR, SGNR, SGTCR, SGIDR, SGPCR) which can be updated when in DMA transfer.

This setting is made for respective registers.

The Sound Generator identifies the register to be updated by DMA transfer according to the setting of this register.

It also recognizes the number of times of DMA transfer and the transfer byte size according to the setting of this register, as well as the valid byte position of the DMA Transfer Intermediate Register (SGDMAR).

bit	15	14	13	12	11	10	9	8
Field	Reserved							EFRE
Attribute	-							R/W
Initial value	0x00							0

bit	7	6	5	4	3	2	1	0
Field	ARE1	ARE0	FRE	NRE	TCRE	IDRE	PCRE1	PCRE0
Attribute	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial value	0	0	0	0	0	0	0	0

[bit15:9] Reserved : Reserved bits

Always write "0" to this bit. The read value is "0".

[bit8] EFRE : Extended frequency data update enable bit

This bit is to enable the update of the extended frequency data of the Extended Frequency Data Register (SGEFR) through the DMA Transfer Intermediate Register (SGDMAR) when in DMA transfer.

bit	Description
0	The update of the extended frequency data is disabled.
1	The update of the extended frequency data is enabled.

Notes:

- *When the Tone output mode select bit (SGCR.TMS) is in Extended mode (SGCR.TMS="1"), it is allowed to write the Extended frequency data update enable bit (SGDER.EFRE) and the Extended Frequency Data Register (SGEFR).*
- *The Tone output mode select bit (SGCR.TMS) and the Extended frequency data update enable bit (SGDER.EFRE) are allowed to be written at one time.*
- *When the Tone output mode select bit (SGCR.TMS) was changed from Extended mode (SGCR.TMS="1") to Normal mode (SGCR.TMS="0"), the Extended frequency data update enable bit (SGDER.EFRE) and the Extended Frequency Data Register (SGEFR) are cleared to the initial value.*

[bit7] ARE1 : Amplitude data (upper byte) update enable bit

This bit is to enable the update of the amplitude data (upper byte) of the Amplitude Data Register (SGAR) through the DMA Transfer Intermediate Register (SGDMAR) when in DMA transfer.

bit	Description
0	The update of the amplitude data (upper byte) is disabled.
1	The update of the amplitude data (upper byte) is enabled.

[bit6] ARE0 : Amplitude data (lower byte) update enable bit

This bit is to enable the update of the amplitude data (lower byte) of the Amplitude Data Register (SGAR) through the DMA Transfer Intermediate Register (SGDMAR) when in DMA transfer.

bit	Description
0	The update of the amplitude data (lower byte) is disabled.
1	The update of the amplitude data (lower byte) is enabled.

[bit5] FRE : Frequency data update enable bit

This bit is to enable the update of the frequency data of the Frequency Data Register (SGFR) through the DMA Transfer Intermediate Register (SGDMAR) when in DMA transfer.

bit	Description
0	The update of the frequency data is disabled.
1	The update of the frequency data is enabled.

[bit4] NRE : Tone output number update enable bit

This bit is to enable the update of the tone output number of the Tone Output Number Register (SGNR) through the DMA Transfer Intermediate Register (SGDMAR) when in DMA transfer.

bit	Description
0	The update of the tone output number is disabled.
1	The update of the tone output number is enabled.

[bit3] TCRE : Time cycle update enable bit

This bit is to enable the update of the time cycle of the Time Cycle Register (SGTCR) through the DMA Transfer Intermediate Register (SGDMAR) when in DMA transfer.

bit	Description
0	The update of the time cycle is disabled.
1	The update of the time cycle is enabled.

[bit2] IDRE : Increase and decrease data update enable bit

This bit is to enable the update of the increase and decrease data of the Increase and Decrease Data Register (SGIDR) through the DMA Transfer Intermediate Register (SGDMAR) when in DMA transfer.

bit	Description
0	The update of the increase and decrease data is disabled.
1	The update of the increase and decrease data is enabled.

[bit1] PCRE1 : PWM cycle data (upper byte) update enable bit

This bit is to enable the update of the PWM cycle data (upper byte) of the PWM Cycle Data Register (SGPCR) through the DMA Transfer Intermediate Register (SGDMAR) when in DMA transfer.

bit	Description
0	The update of the PWM cycle data (upper byte) is disabled.
1	The update of the PWM cycle data (upper byte) is enabled.

[bit0] PCRE0 : PWM cycle data (lower byte) update enable bit

This bit is to enable the update of the PWM cycle data (lower byte) of the PWM Cycle Data Register (SGPCR) through the DMA Transfer Intermediate Register (SGDMAR) when in DMA transfer.

bit	Description
0	The update of the PWM cycle data (lower byte) is disabled.
1	The update of the PWM cycle data (lower byte) is enabled.

4.2. Sound Control Register (SGCR)

The Sound Control Register (SGCR) is to control interrupts and operation status of the Sound Generator.

bit	15	14	13	12	11	10	9	8
Field	Reserved	SRST	DMA	GID	GEN	Reserved	BUSY	TMS
Attribute	-	W	R/W	R/W	R/W	-	R	R/W
Initial value	0	0	0	0	0	0	0	0

bit	7	6	5	4	3	2	1	0
Field	S1	S0	TONE	SGOOE	SGAOE	INTE	INT	ST
Attribute	R/W	R/W	R/W	R/W	R/W	R/W	R	R/W
Initial value	0	0	0	0	0	0	0	0

[bit15] Reserved : Reserved bit

Always write "0" to this bit. The read value is "0".

[bit14] SRST : Software reset bit

This bit is used to issue a software reset in the Sound Generator.

When in the read access, the read value is always "0". Writing "0" to this bit has no effect.

bit	Description
	Write
0	No effect.
1	Writing "1" issues a software reset.

[bit13] DMA : DMA transfer start interrupt setting enable bit

This bit is to enable DMA transfer start interrupt.

bit	Description
0	Writing "1" to the Start bit (SGCR.ST) does not make the Interrupt status bit (SGCR.INT) set.
1	Writing "1" to the Start bit (SGCR.ST) makes the Interrupt status bit (SGCR.INT) set.

Note:

- *Do not change this setting while operating (SGCR.ST="1" or SGCR.BUSY="1").*

[bit12] GID : Increase/decrease setting bit

This bit is for automatic increase/decrease setting of the sound amplitude, working with the Time Cycle Register (SGTCR), the Increase and Decrease Data Register (SGIDR), and the automatic increase/decrease enable bit (GEN).

This bit determines whether the value of Amplitude Data Register (SGAR) increases or decreases.

bit	Description
0	The value stored in the Amplitude Data Register (SGAR) decreases.
1	The value stored in the Amplitude Data Register (SGAR) increases.

Notes:

- *This bit is valid only when the automatic increase/decrease enable bit is set to "enabled"(SGCR.GEN="1").*
- *If this bit is changed during the operation, it is reflected at the time when the value of "Amplitude Data Register (SGAR)" is automatically updated.*

[bit11] GEN : Automatic increase/decrease enable bit

This bit is for automatic increase/decrease setting of the sound, working with the Time Cycle Register (SGTCR), the Increase and Decrease Data Register (SGIDR), and the Increase/decrease setting bit (GID).

bit	Description
0	The automatic increase or decrease is disabled.
1	The automatic increase or decrease is enabled.

Note:

- *When this bit turns from "enabled" to "disabled", the Amplitude Data Register (SGAR) holds the value at that time.*

[bit10] Reserved : Reserved bit

Always write "0" to this bit. The read value is "0".

[bit9] BUSY : Busy status bit

This bit is to indicate whether the Sound Generator is in operation or not.

This bit is only for read access, and the write access to this bit has no effect.

bit	Description
	Read
0	Sound Generator is inactive. This bit is cleared under any of following conditions. - Software reset is set (SGCR.SRST="1"). - Start bit(SGCR.ST) is cleared by writing "0". And the operation is completed after 1 tone cycle (current tone cycle) has finished.
1	Sound Generator is active. This bit is set when Start bit(SGCR.ST) is written "1".

[bit8] TMS: Tone output mode select bit

This bit selects output mode of tone pulse signal of SGO.

bit	Description
0	Normal mode (initial value) - Duty ratio of a tone pulse signal is fixed to 50%.
1	Extended mode - Duty ratio (high side) of a tone pulse signal is programmable up to 50%.

Notes:

- *Do not change this setting while operating (SGCR.ST="1" or SGCR.BUSY="1")*
- *When the Tone output mode select bit (SGCR.TMS) is in Extended mode (SGCR.TMS="1"), it is allowed to write the Extended frequency data update enable bit (SGDER.EFRE) and the Extended Frequency Data Register (SGEFR).*
- *The Tone output mode select bit (SGCR.TMS) and the Extended frequency data update enable bit (SGDER.EFRE) are allowed to be written at one time.*
- *When the Tone output mode select bit (SGCR.TMS) was changed from Extended mode (SGCR.TMS="1") to Normal mode (SGCR.TMS="0"), the Extended frequency data update enable bit (SGDER.EFRE) and the Extended Frequency Data Register (SGEFR) are cleared to the initial value.*

[bit7:6] S1, S0 : Operation clock select bits

The combination of these bits controls the internal clock division, to determine the bit rate on the output sound.

S1	S0	Description
0	0	The division ratio at the Prescaler is 1.
0	1	The division ratio at the Prescaler is 2.
1	0	The division ratio at the Prescaler is 4.
1	1	The division ratio at the Prescaler is 8..

[bit5] TONE : Tone output bit

This bit controls the type of SGO output signal.

bit	Description
0	SGO outputs the mixed signal of tone pulse and PWM pulse.
1	SGO outputs a simple square wave (tone pulse)

[bit4] SGOOE : SGO signal output enable bit

This bit controls whether to enable the output of SGO.

bit	Description
0	SGO output is disabled.
1	SGO output is enabled.

[bit3] SGAOE : SGA signal output enable bit

This bit controls whether to enable the output of SGA.

bit	Description
0	SGA output is disabled.
1	SGA output is enabled.

[bit2] INTE : Interrupt enable bit

This bit enables the interrupt request of the Sound Generator.

bit	Description
0	The interrupt by Interrupt status bit (SGCR.INT) is masked, and the status of Interrupt status bit (SGCR.INT) is not reflected to the output on PIRQ.
1	The interrupt by Interrupt status bit (SGCR.INT) is not masked, and the status of Interrupt status bit (SGCR.INT) is reflected to the output on PIRQ.

[bit1] INT : Interrupt status bit

This is an interrupt flag of the Sound Generator.

This bit is only for read access, and the write access to this bit has no effect.

bit	Description	
	Read	Write
0	The tone pulse counter hasn't detected the number of tone pulses set as Tone Output Number Register (SGNR) and Time Cycle Register (SGTCR). This bit is cleared to "0" under any of following conditions. - Software reset is set (SGCR.SRST="1"). - DMA transfer - Interrupt Clear bit is set (SGCCR.INTC="1")	No effect.

bit	Description	
	Read	Write
1	The tone pulse counter has detected the number of tone pulses set as Tone Output Number Register (SGNR) and Time Cycle Register (SGTCR). This bit is set to "1" under the following condition. - The number of tone pulse count is greater than or equal to ((Time Cycle Register [SGTCR] + 1) x (Tone Output Number Register [SGNR] + 1))	No effect.

[bit0] ST : Start bit

This bit is a start trigger of the Sound Generator.

bit	Description
0	Writing "0" to this bit stops Sound Generator.
1	Writing "1" to this bit starts Sound Generator.

Notes:

- *Whether the Sound Generator is stopped or not can be seen on the status bit BUSY.*
- *This bit is cleared also when a software reset is issued (SGCR.SRST = "1").*

4.3. Amplitude Data Register (SGAR)

The Amplitude Data Register (SGAR) stores the reload value for a PWM pulse generator.

The register value represents sound amplitude and is reloaded into the PWM pulse generator at the end of every tone cycle.

bit	15	14	13	12	11	10	9	8
Field	SGAR[15:8]							
Attribute	R/W							
Initial value	0x00							

bit	7	6	5	4	3	2	1	0
Field	SGAR[7:0]							
Attribute	R/W							
Initial value	0x00							

[bit15:0] SGAR : Amplitude data bits

These bits store the reload value for the PWM pulse generator,
and the value is written either in the following ways.
(1)Software writes a value to this register.
(2)DMAC writes a value through DMA Transfer Intermediate Register (SGDMAR).

When in decreasing the value of this register, 0x0000 is the lower limit and the value doesn't roll over to 0xFFFF from it.

When in increasing the value of this register, 0xFFFF is the upper limit and the value doesn't roll over to 0x0000 from it.

In any case, the Sound Generator keeps operating until the Start bit (SGCR.ST) is cleared.

Notes:

- *The number of clock cycles in high width of 1 PWM is equal to "SGAR + 1".*
- *During operation, in the case of changing a register value, meet the following either conditions.*
  (1)Sound generator is in the stop state (SGCR.ST="0" and SGCR.BUSY="0").
  (2)The changing comes between an interrupt occurrence and falling edge of the first tone pulse.
  (The limit time = (Frequency Data Register [SGFR] + 1) x 1 PWM cycle)
  When it meets neither condition, the sound output cannot guarantee the expected duty ratio.

4.4. Frequency Data Register (SGFR)

The Frequency Data Register (SGFR) stores the reload value for a Frequency counter.

This register setting is half cycle of a tone pulse signal.

bit	7	6	5	4	3	2	1	0
Field	SGFR[7:0]							
Attribute	R/W							
Initial value	0x00							

[bit7:0] SGFR : Frequency data bits

These bits store the reload value for the Frequency counter,
and the value is written either in the following ways.
(1)Software writes a value to this register.
(2)DMAC writes a value through DMA Transfer Intermediate Register (SGDMAR).

Notes:

- *The number of PWM pulses is equal to "SGFR + 1".*
- *During operation, in the case of changing a register value, meet the following either conditions.*
  (1)Sound generator is in the stop state (SGCR.ST="0" and SGCR.BUSY="0").
  (2)The changing comes between an interrupt occurrence and falling edge of the first tone pulse.
  (The limit time = (Frequency Data Register [SGFR] + 1) x 1 PWM cycle)
  When it meets neither condition, the sound output cannot guarantee the expected duty ratio.

4.5. Tone Output Number Register (SGNR)

The Tone Output Number Register (SGNR) stores the reload value for a Tone pulse counter.

The Tone pulse counter accumulates the number of tone pulses (or the number of sound increment/decrement). When the accumulated value reaches the reload value, it sets the Interrupt status bit (SGCR.INT). This operation is to reduce the frequency of interrupts.

bit	7	6	5	4	3	2	1	0
Field	SGNR[7:0]							
Attribute	R/W							
Initial value	0x00							

[bit7:0] SGNR : Tone output number bits

These bits store the reload value for the Tone pulse counter,
and the value is written either in the following ways.
(1)Software writes a value to this register.
(2)DMAC writes a value through DMA Transfer Intermediate Register (SGDMAR).

Note:

- *The number of pulses is equal to "SGNR + 1".*
- *During operation, in the case of changing a register value, meet the following either conditions.*
  (1)Sound generator is in the stop state (SGCR.ST="0" and SGCR.BUSY="0").
  (2)The changing comes between an interrupt occurrence and falling edge of the first tone pulse.
  (The limit time = (Frequency Data Register [SGFR] + 1) x 1 PWM cycle)
  When it meets neither condition, the sound output cannot guarantee the expected duty ratio.

4.6. Time Cycle Register (SGTCR)

The Time Cycle Register (SGTCR) stores the reload value for a Decrement counter.

bit	7	6	5	4	3	2	1	0
Field	SGTCR[7:0]							
Attribute	R/W							
Initial value	0x00							

[bit7:0] SGTCR : Time cycle bits

These bits store the reload value for the Decrement counter,
and the value is written either in the following ways.
(1)Software writes a value to this register.
(2)DMAC writes a value through DMA Transfer Intermediate Register (SGDMAR).

Notes:

- *The number of pulses is equal to "SGTCR + 1".*
- *When SGTCR is 0x00, the automatic increase or decrease operation on the Amplitude Data Register (SGAR) occurs at each tone cycle.*
- *During operation, in the case of changing a register value, meet the following either conditions.*
  (1)Sound generator is in the stop state (SGCR.ST="0" and SGCR.BUSY="0").
  (2)The changing comes between an interrupt occurrence and falling edge of the first tone pulse.
  (The limit time = (Frequency Data Register [SGFR] + 1) x 1 PWM cycle)
  When it meets neither condition, the sound output cannot guarantee the expected duty ratio.

4.7. Increase and Decrease Data Register (SGIDR)

The Increase and Decrease Data Register (SGIDR) stores the increment or decrement for the Amplitude Data Register (SGIDR).

bit	7	6	5	4	3	2	1	0
Field	SGIDR[7:0]							
Attribute	R/W							
Initial value	0x00							

[bit7:0] SGIDR : Increase and decrease data bits

These bits store the increment or decrement for the Amplitude Data Register (SGAR) , and the value is written either in the following ways.
(1)Software writes a value to this register.
(2)DMAC writes a value through DMA Transfer Intermediate Register (SGDMAR).

Note:

- *During operation, in the case of changing a register value, meet the following either conditions.*
  (1)Sound generator is in the stop state (SGCR.ST="0" and SGCR.BUSY="0").
  (2)The changing comes between an interrupt occurrence and falling edge of the first tone pulse.
  (The limit time = (Frequency Data Register [SGFR] + 1) x 1 PWM cycle)
  When it meets neither condition, the sound output cannot guarantee the expected duty ratio.

4.8. PWM Cycle Data Register (SGPCR)

The PWM Cycle Data Register (SGPCR) stores the number of cycles in 1 PWM cycle.

bit	15	14	13	12	11	10	9	8
Field	SGPCR[15:8]							
Attribute	R/W							
Initial value	0x00							

bit	7	6	5	4	3	2	1	0
Field	SGPCR[7:0]							
Attribute	R/W							
Initial value	0xFF							

[bit15:0] SGPCR : PWM cycle number data bits

These bits store the number of cycles in 1PWM cycle,
and the value is written either in the following ways.
(1)Software writes a value to this register.
(2)DMAC writes a value through DMA Transfer Intermediate Register (SGDMAR).

Notes:

- *The number of cycles in 1 PWM is equal to "SGPCR + 1".*
- *During operation, in the case of changing a register value, meet the following either conditions.*
  (1)Sound generator is in the stop state (SGCR.ST="0" and SGCR.BUSY="0").
  (2)The changing comes between an interrupt occurrence and falling edge of the first tone pulse.
  (The limit time = (Frequency Data Register [SGFR] + 1) x 1 PWM cycle)
  When it meets neither condition, the sound output cannot guarantee the expected duty ratio.

4.9. Extended Frequency Data Register (SGEFR)

The Extended Frequency Data Register (SGEFR) stores the comparison value of Frequency counter. This register setting is high width of a tone pulse signal in Extended mode (SGCR.TMS="1").

bit	7	6	5	4	3	2	1	0
Field	SGEFR[7:0]							
Attribute	R/W							
Initial value	0xFF							

[bit7:0] SGEFR : Extended Frequency data bits

These bits store the comparison value of Frequency counter,
and the value is written either in the following ways.
(1)Software writes a value to this register.
(2)DMAC writes a value through DMA Transfer Intermediate Register (SGDMAR).

Notes:

- *The number of PWM pulses is equal to "SGEFR + 1".*
- *When the Tone output mode select bit (SGCR.TMS) is in Extended mode (SGCR.TMS="1"), it is allowed to write the Extended frequency data update enable bit (SGDER.EFRE) and the Extended Frequency Data Register (SGEFR).*
- *The Tone output mode select bit (SGCR.TMS) and the Extended frequency data update enable bit (SGDER.EFRE) are allowed to be written at one time.*
- *When the Tone output mode select bit (SGCR.TMS) was changed from Extended mode (SGCR.TMS="1") to Normal mode (SGCR.TMS="0"), the Extended frequency data update enable bit (SGDER.EFRE) and the Extended Frequency Data Register (SGEFR) are cleared to the initial value.*
- *During operation, in the case of changing a register value, meet the following either conditions.*
  (1)Sound generator is in the stop state (SGCR.ST="0" and SGCR.BUSY="0").
  (2)The changing comes between an interrupt occurrence and falling edge of the first tone pulse. (The limit time = (Frequency Data Register [SGFR] + 1) x 1 PWM cycle)
  When it meets neither condition, the sound output cannot guarantee the expected duty ratio.

4.10. DMA Transfer Intermediate Register (SGDMAR)

The DMA Transfer Intermediate Register (SGDMAR) is available in DMA transfer, and works as a window to access several registers. The details are described in "3.4 Relation between the DMA Transfer Update Enable Register (SGDER) and DMA ".

bit	31	30	29	28	27	26	25	24
Field	SGDMAR[31:24]							
Attribute	W							
Initial value	0x00							

bit	23	22	21	20	19	18	17	16
Field	SGDMAR[23:16]							
Attribute	W							
Initial value	0x00							

bit	15	14	13	12	11	10	9	8
Field	SGDMAR[15:8]							
Attribute	W							
Initial value	0x00							

bit	7	6	5	4	3	2	1	0
Field	SGDMAR[7:0]							
Attribute	W							
Initial value	0x00							

[bit31:0] SGDMAR : DMA Transfer data bits

This register is used to write the following registers:
- Amplitude Data Register (SGAR)
- Frequency Data Register (SGFR)
- Tone Output Number Register (SGNR)
- Time Cycle Register (SGTCR)
- Increase and Decrease Data Register (SGIDR)
- PWM Cycle Data Register (SGPCR)
- Extended Frequency Data Register (SGEFR)

The details are described in "3.4 Relation between the DMA Transfer Update Enable Register (SGDER) and DMA ".

The read value is always 0x00000000.

Notes:

- *If the number of DMA transfer is greater than "the number determined by the DMA Transfer Update Enable Register (SGDER)", the value of this register is updated.*
- *When in reading corresponding registers (not through this register), please make sure the setting of "DMA Transfer Update Enable Register (SGDER)" is available for the target registers.*
- *As for the enabled setting register of the DMA Transfer Update Enable Register (SGDER), access is enabled to both the DMA Transfer Intermediate Register (SGDMAR) and individual registers (SGAR, SGFR, SGNR, SGTCR, SGIDR, SGPCR, SGEFR).*
- *During operation, in the case of changing a register value, meet the following either conditions.*
  (1)Sound generator is in the stop state (SGCR.ST="0" and SGCR.BUSY="0").
  (2)The changing comes between an interrupt occurrence and falling edge of the first tone pulse. (The limit time = (Frequency Data Register [SGFR] + 1) x 1 PWM cycle)
  When it meets neither condition, the sound output cannot guarantee the expected duty ratio.

4.11. Interrupt Clear Register (SGCCR)

The Interrupt Clear Register (SGCCR) clears the Interrupt status bit (SGCR.INT) in the Sound Control Register(SGCR).

bit	15	14	13	12	11	10	9	8
Field	Reserved							
Attribute	-							
Initial value	0x00							

bit	7	6	5	4	3	2	1	0
Field	Reserved						INTC	Reserved
Attribute	-						W	-
Initial value	0x00						0	0

Note:

- *Use 16- or 32-bit data access for the SGCCR register.*

[bit15:2] Reserved : Reserved bits

Always write "0" to these bits. The read value is "0".

[bit1] INTC : Interrupt status clear bit

When this bit is set to "1", Interrupt status bit (SGCR.INT) is cleared.

The read value is "0".

bit	Description
	Write
0	No effect.
1	Interrupt status bit (SGCR.INT) is cleared.

[bit0] Reserved : Reserved bit

Always write "0" to this bit. The read value is "0".

CHAPTER 19: Sound Waveform Generator

This chapter explains the sound waveform generator.

1. Overview
2. Configuration and Block Diagram
3. Operation of the Sound Waveform Generator
4. Registers

CODE: SOUNDWFG-S6J3300-E1

1. Overview

This section gives a brief overview of the sound waveform generator (SWFG).

Also refer to the chapter of "Sound System configuration" how to configure the SWFG.

The SWFG is a function that generates and outputs a Pulse Code Modulation (PCM) sound source using simple software settings.

Table 19-1 PCM Sound Source Software Settings

Feature	Description
Output frequency	64 types (scales) settable by software.
Output waveform	Sine wave, sawtooth wave, and square wave selectable by software.
Volume control (fade in and fade out functions)	Seamless fade in and fade out by software are supported immediately after generation starts and immediately before generation ends.
Sound time	A time from 1 ms to 4 seconds selectable by software.
Number of output channels	Simultaneous output of up to five different sound sources is supported.
Interrupt	Completion of sound source output generates an interrupt.
Consecutive setting	While a sound source is being generated, register settings for generating the next sound source can be configured.
Output data pattern	Generates a 16-bit PCM (Pulse Code Modulation) sound source XXXX internally by the SWFG, expands it to 32 bits, and outputs it (AHB master interface). The output data pattern can be selected by register settings. – XXXX_XXXX: Upper and lower 16 bits are the same data. – 0000_XXXX: Only lower 16 bits of data (Upper 16 bits are zero.) – XXXX_0000: Only upper 16 bits of data (Lower 16 bits are zero.)
Output sampling rate	48 kHz (fixed).
FIR filter	A Low Pass Filter is provided to eliminate sawtooth and square waveform overtones. The filtering cutoff frequency is selectable.

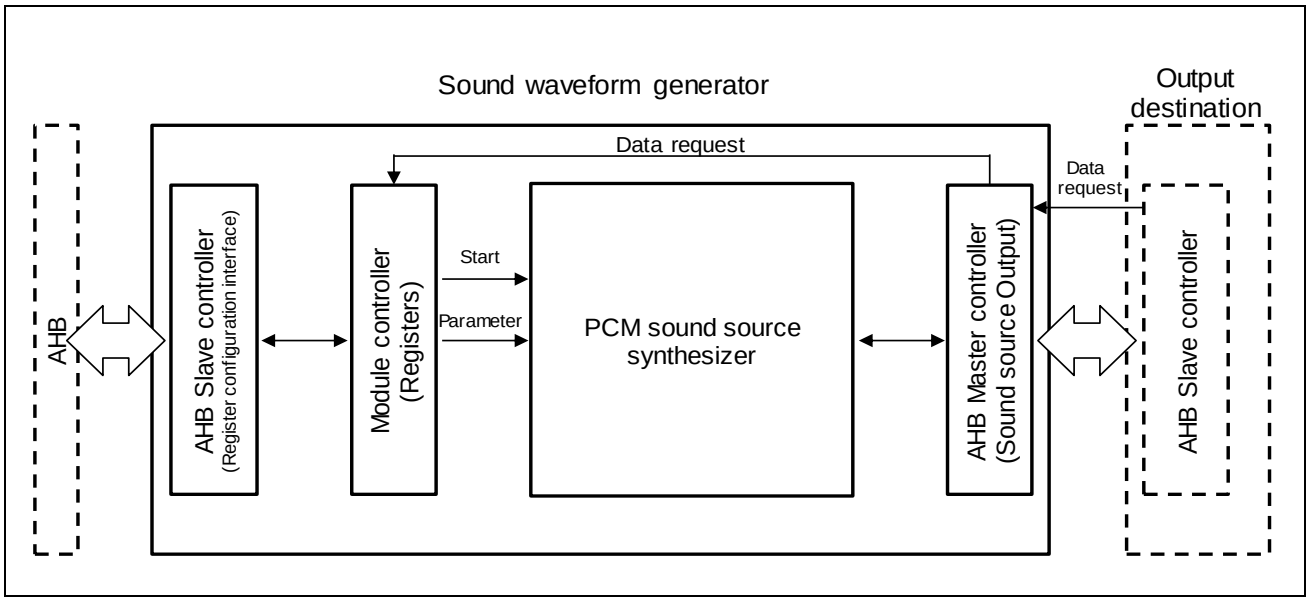
Note:

- *Theoretically, system noise tends not to be a factor in the case of a sine wave that contains single frequency components. Use of a sine wave is particularly recommended when sound source mixing or other processes are performed by other functions.*
- *Interrupts due to factors other than sound source output completion are also supported. For details, see 3.2.*
- *Continuous sound (continuation of the same sound or different sound) can be generated by configuring the settings of the next sound's parameters while a sound source is being generated. However, when repeatedly generating the same sound the same parameters settings need to be configured.*
- *Use of the Low Pass Filter can be disabled. When using the Low Pass Filter, the optimum cutoff frequency should be decided upon sufficient evaluation of sound quality on the system.*

2. Configuration and Block Diagram

This section shows a block diagram of the sound waveform generator.

Figure 19-1 Block Diagram of Sound Waveform Generator



3. Operation of the Sound Waveform Generator

3.1. Sound Source Specifications

This section describes the specifications of a sound source generated by SWFG.

Figure 19-2 Sound Source by SWFG

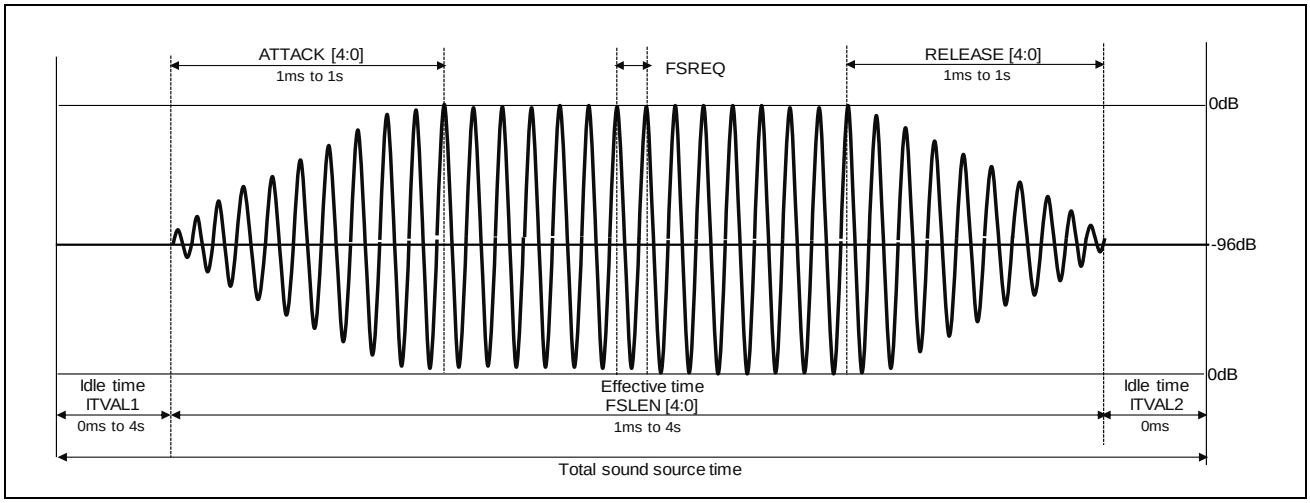


Table 19-2 Specifications of a Sound Source by SWFG

Element	Description	Relation
Start idle time	Silence generation time (ALL 0 data generation) immediately following sound source generation start.	4.8
End idle time	This function is not supported.	4.8
Waveform	Sound source waveform	4.7
Frequency	Sound source frequency	4.7
Run time	Time to generate a sound source of a specified frequency	4.7
Fade in time (ATTACK)	Fade in time (included in run time) immediately following run time start	4.9
Fade out time (RELEASE)	Fade out time immediately before run time end (included in run time)	4.9
Overall sound source time	Overall time of the specified sound source, which is the total of idle time and effective time	Figure 19-2

3.1.1. Idle Time

There is start idle time and end idle time.

Start idle time is silence generation time immediately following sound source generation start. End idle time is not supported.

As for start idle time, one of 32 idle times from 0 ms to 4 seconds is selectable by software.

Note:

- *Control of WGCHEN under 4.1 causes ALL 0 to be output at the point that SWFG operation is enabled, but this is not included in idle time. It should be noted that start idle time indicates the period from the start of sound source generation in accordance with control by WGCHSTART under 4.2, up to generation of ALL 0.*
- *Refer also to 3.1.*

3.1.2. Waveform

The waveform is the fundamental wave that makes up the sound source.

One of 3 waveforms (sine wave, sawtooth wave, square wave) is selectable by software. For details, see 4.7.

3.1.3. Frequency

Frequency is a sound source component that corresponds to the pitch of the sound.

One of 64 frequencies is selectable by software. For details, see 4.7.

3.1.4. Run Time

Run time is the time to generate a sound source of the specified frequency.

One of 31 run times from 1 ms to 4 seconds is selectable by software. For details, see 4.7.

Note:

- *Neither the start nor end idle times are included in the run time.*
- *Fade in time and fade out time are both included in the run time.*

3.1.5. Fade In Time (ATTACK)

Fade in time is time required for volume change from -96 dB to 0 dB immediately following run time start. Fade in time is included in run time.

One of 24 fade in times from 1 ms to 1 seconds is selectable by software. For details, see 4.9.

3.1.6. Fade Out Time (RELEASE)

Fade out time is time required for volume change from 0 dB to -96 dB immediately following run time end. Fade out time is included in the run time.

One of 24 fade out times from 1 ms to 1 second is selectable by software. For details, see 4.9.

Note:

- *Fade in time and fade out time are both included in the run time. Configuration of settings that result in fade in time, fade out time, and the total of these times being longer than the run time is prohibited.*
- *If the total of these times exceeds the run time, fade in will start from -96 dB and a switch will be made to fade out before 0 dB is reached.*
- *Each configuration should be under condition below.*
- *Attack ≤ FSLEN, Release ≤ FSLEN and Attack + Release ≤ FSLEN*

3.2. Interrupt

SWFG supports the interrupts described below.

- Sound source generation complete interrupt
  This interrupt is generated when sound source generation is complete. SWFG performs interrupt enable setting, interrupt clear control and status indication.
- SWFG AHB Master Interface bus error interrupt
  This interrupt is generated when an error response is received during sound source output to an output destination on the AHB Master Interface of SWFG. SWFG performs interrupt enable setting, interrupt clear control and status indication.

3.3. SWFG Sound Source Generation Start

The procedures for starting sound source generation is provided below. Each channel can be controlled independently.

1. Set the output destination address of the sound source. See WGCHADD1 through 3 under 4.3, 4.4, and 4.5.
2. Select the sound source output data pattern. See WGCHMONO under 4.6.
3. Configure interrupt enable/disable settings. See WGINTREN under 4.11.
4. Configure sound source specification settings. See WGCHxCTRL1 through 3(x=0 through 4) under 4.7, 4.8, and 4.9.
5. Performs startup control. For details, see WGCHEN under 4.1.
6. Start sound source generation. For details, see WGCHSTART under 4.2.

Note:

- *Steps 1 through 4 of the above procedure can be performed in any sequence.*
- *Enabling operation in step 5 of the above procedure causes SWFG to output ALL 0 data. After that, step 6 starts generation of the configured sound source. If SWFG is not running, the sound source is not generated when generation start is executed.*
- *Even if operation is enabled and sound source generation startup control is performed, it will not start without a data request from the SWFG output destination. The output destination needs to be set beforehand.*

3.4. SWFG Sound Source Generation Stop

SWFG operation that started sound source generation can be stopped on a channel-by-channel basis. See WGCHEN under 4.1 and WGCHCLR under 4.10.

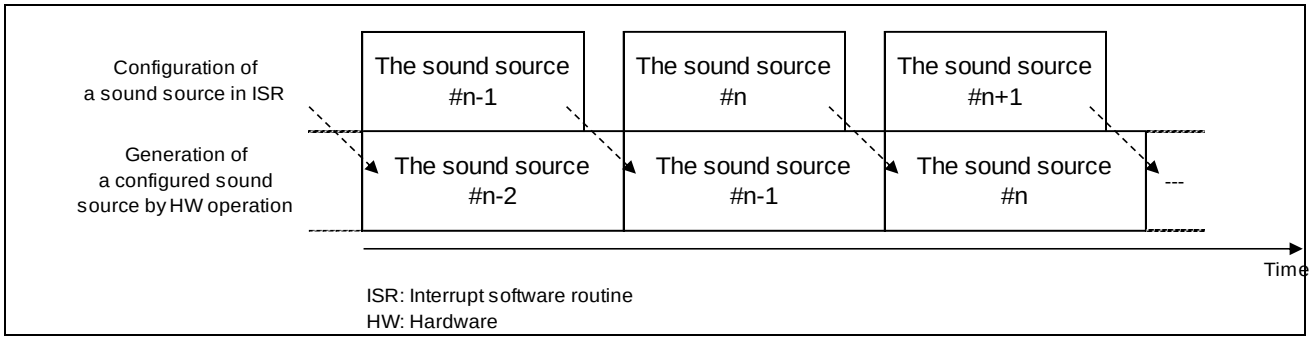
Note:

- *Sound source generation stopped by WGCHEN under 4.1 can be resumed by performing startup control on the same bit, which will resume from the point where generation was stopped. There is no sound source output, including ALL 0, from channels that are stopped by this control. The effect on sound source output destinations should be kept firmly in mind when stopping sound source generation.*
- *Sound source generation can also be stopped by WGCHCLR under 4.10. However, there is no control to restart sound source generation from the point it was stopped by WGCHCLR. ALL 0 is output from channels that are stopped by WGCHCLR.*

3.5. Continuous SWFG Sound Source Generation

Setting of the next sound source specification can be performed by starting sound source generation using the procedure under 3.1, even if a previous sound source generation operation is in progress. After setting is complete, sound source generation starts and the corresponding WGCHSTART bit is set to 1, so generation of the next sound source starts as soon as generation of the current sound source is complete.

Figure 19-3 Continuous SWFG Sound Source Generation



Note:

- *Setting the next sound's sound source specifications after the previous sound source generation is complete can cause an audible break in the sound. Consecutive sound generation that uses a sound source generation complete interrupt as shown in Figure 19-3 is recommended.*
- *Using fade in and fade out control between the previous generation and the current generation when consecutively generating sound sources can cause an audible break in the sound. When performing consecutive generation, application of fade in and fade out, control is recommended only when setting initial and final sound source specifications.*
- *The FLSEN should be configured as longer than RELEASE. Only when the final sound source of consecutive sound generation is generated, RELEASE only performs and FSLEN is ignored in generation time.*

3.6. Filter

3 type of digital filter can be selected and applied to the generated sound source to remove overtone affections of sawtooth and square wave-form. See 4.7 in detail.

Figure 19-4 Cutoff Frequency 8 kHz

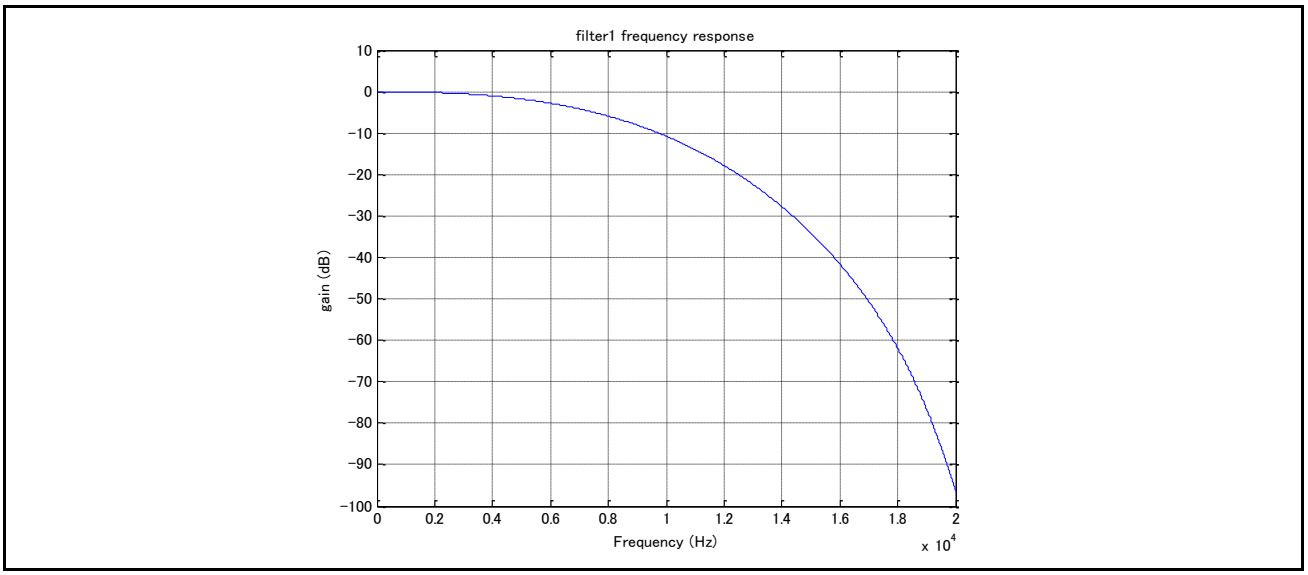


Figure 19-5 Cutoff Frequency 5 kHz

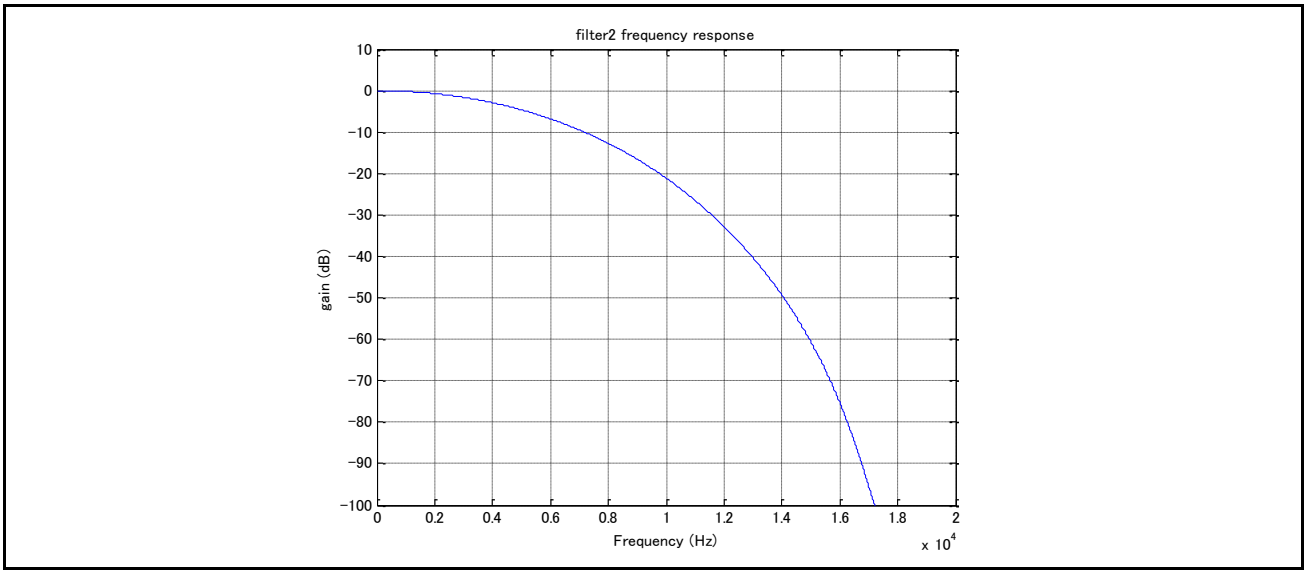
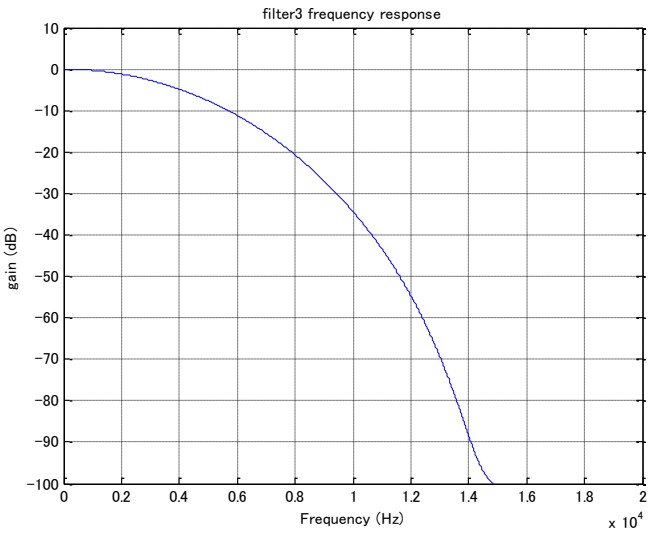


Figure 19-6 Cutoff Frequency 2 kHz



4. Registers

This section describes the registers of Sound Waveform Generator

4.1. Waveform Generator Channel Enable Register (WGCHEN)

REGISTER_NAME	Waveform Generator Channel Enable Register (WGCHEN)
OFFSET	0x000
ACCESS_SIZE	B H W
MULTIPLE	31:0
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,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	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,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	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,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	Reserved	Reserved	Reserved	CH4EN	CH3EN	CH2EN	CH1EN	CH0EN
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	Wp							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:5] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit4] CH4EN: Channel 4 operation enable/disable

[bit3] CH3EN: Channel 3 operation enable/disable

[bit2] CH2EN: Channel 2 operation enable/disable

[bit1] CH1EN: Channel 1 operation enable/disable

[bit0] CH0EN: Channel 0 operation enable/disable

CHnEN (n=0 to 4)	Description
0	Stop operation.

CHnEN (n=0 to 4)	Description
1	Enable operation.

Note:

- *When operation is stopped, SWFG stops sound source generation and output.*
- *There is no sound source output, including ALL 0, from channels that are stopped by this control. The effect on sound source output destinations should be kept firmly in mind when stopping sound source generation.*

4.2. Waveform Generator Channel Start Register (WGCHSTART)

REGISTER_NAME	Waveform Generator Channel Start Register (WGCHSTART)
OFFSET	0x004
ACCESS_SIZE	B H W
MULTIPLE	31:0
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,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	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,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	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,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	Reserved	Reserved	Reserved	CH4START	CH3START	CH2START	CH1START	CH0START
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	Wp							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:5] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit4] CH4START: Channel 4 sound source generation start

[bit3] CH3START: Channel 3 sound source generation start

[bit2] CH2START: Channel 2 sound source generation start

[bit1] CH1START: Channel 1 sound source generation start

[bit0] CH0START: Channel 0 sound source generation start

CHnSTART (n=0 to 4)	Description	
	Read	Write
0	Enable sound source generation.	Does not have any effect on operation.
1	Sound source generation start wait.	Start sound source generation.

Note:

- *When "1" is written to this bit, SWFG starts sound source generation in accordance with sound source specification settings.*
- *SWFG is when "1" is written to this bit while a previous sound source generation is in progress, consecutive sound source generation starts when generation of the previous sound source is complete.*
- *This bit is automatically cleared when the desired sound source generation starts.*
- *Writing "0" in this bit does not stop sound source generation. For information about stopping sound source generation, see 4.1.*
- *Sound source generation start wait cannot be canceled.*
- *Changing sound source specifications during sound source generation start standby can result in sound source generation starting while implementation of changes is in progress, resulting in unintended sound being produced. Set sound source specifications while sound source generation is enabled.*

4.3. Waveform Generator Channel Address1 Register (WGCHADD1)

REGISTER_NAME	Waveform Generator Channel Address1 Register (WGCHADD1)
OFFSET	0x010
ACCESS_SIZE	B H W
MULTIPLE	31:0
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved	Reserved	Reserved	CH1ADD[12:8]				
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	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	CH1ADD[7:0]							
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

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved	Reserved	Reserved	CH0ADD[12:8]				
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	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	CH0ADD[7:0]							
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

[bit31:29] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit28:16] CH1ADD: Channel 1 output destination address

Sets the Channel 1 output destination address as 13 bits.

Notes:

These bits can be set at the initialization. After that, it should not be changed.

[bit15:13] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit12:0] CH0ADD: Channel 0 output destination address

Sets the Channel 0 output destination address as 13 bits.

Notes:

These bits can be set at the initialization. After that, it should not be changed.

4.4. Waveform Generator Channel Address2 Register (WGCHADD2)

REGISTER_NAME	Waveform Generator Channel Address2 Register (WGCHADD2)
OFFSET	0x014
ACCESS_SIZE	B H W
MULTIPLE	31:0
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved	Reserved	Reserved	CH3ADD[12:8]				
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	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	CH3ADD[7:0]							
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

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved	Reserved	Reserved	CH2ADD[12:8]				
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	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	CH2ADD[7:0]							
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

[bit31:29] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit28:16] CH3ADD: Channel 3 output destination address

Sets the Channel 3 output destination address as 13 bits.

Notes:

These bits can be set at the initialization. After that, it should not be changed.

[bit15:13] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit12:0] CH2ADD: Channel 2 output destination address

Sets the Channel 2 output destination address as 13 bits.

Notes:

These bits can be set at the initialization. After that, it should not be changed.

4.5. Waveform Generator Channel Address3 Register (WGCHADD3)

REGISTER_NAME	Waveform Generator Channel Address3 Register (WGCHADD3)
OFFSET	0x018
ACCESS_SIZE	B H W
MULTIPLE	31:0
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,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	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,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	Reserved	Reserved	Reserved	CH4ADD[12:8]				
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	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	CH4ADD[7:0]							
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

[bit31:13] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit12:0] CH4ADD: Channel 4 output destination address

Sets the Channel 4 output destination address as 13 bits.

Notes:

These bits can be set at the initialization. After that, it should not be changed.

4.6. Waveform Generator Channel Monaural Register (WGCHMONO)

REGISTER_NAME	Waveform Generator Channel Monaural Register (WGCHMONO)
OFFSET	0x01C
ACCESS_SIZE	B H W
MULTIPLE	31:0
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,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	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,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	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	CH4MONO[1:0]	
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	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	CH3MONO[1:0]		CH2MONO[1:0]		CH1MONO[1:0]		CH0MONO[1:0]	
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

[bit31:10] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit9:8] CH4MONO Channel 4 output data pattern

[bit7:6] CH3MONO Channel 3 output data pattern

[bit5:4] CH2MONO Channel 2 output data pattern

[bit3:2] CH1MONO Channel 1 output data pattern

[bit1:0] CH0MONO Channel 0 output data pattern

CHnMONO [1:0] (n=0 to 4)	Description
00	Upper and lower 16 bits output the same data (XXXX_XXXX).
01	Only lower 16 bits of data are output (0000_XXXX).
10	Only upper 16 bits of data are output (XXXX_0000).
11	Setting prohibited

4.7. Waveform Generator Channel n Control Register1 (WGCHnCTRL1, n=0 to 4)

REGISTER_NAME	Waveform Generator Channel n Control Register (WGCHnCTRL1)
OFFSET	n=0:0x020, n=1:0x030, n=2:0x040, n=3:0x050, n=4:0x060,
ACCESS_SIZE	B H W
MULTIPLE	31:0
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,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	Reserved	Reserved	Reserved	FSLEN[4:0]				
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R/W	R/W	R/W	R/W	R/W
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	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	FSINF[1:0]	
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	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	Reserved	FSFREQ[5:0]					
ACCESS_TYPE	R0,W0	R0,W0	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

[bit31:21] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit20:16] FSLEN: Run time (Channel n = 0 to 4)

Specifies a setting of 1 ms to 4 seconds.

Bit[4:0]	Description
00000	1 ms
00001	1 ms
00010	2 ms
00011	3 ms
00100	4 ms
00101	5 ms
00110	7.5 ms
00111	10 ms
01000	15 ms
01001	20 ms

Bit[4:0]	Description
01010	25 ms
01011	30 ms
01100	40 ms
01101	50 ms
01110	75 ms
01111	100 ms
10000	125 ms
10001	150 ms
10010	200 ms
10011	250 ms
10100	300 ms
10101	400 ms
10110	500 ms
10111	750 ms
11000	1000 ms
11001	1250 ms
11010	1500 ms
11011	2000 ms
11100	2500 ms
11101	3000 ms
11110	3500 ms
11111	4000 ms

Note

When 00000 is written to these bits, the read value will be 00001 instead of 00000.

[bit15:10] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit9:8] FSINF: Waveform (Channel n = 0 to 4)

Bit[1:0]	Description
00	Sine wave
01	Sawtooth wave
10	Square wave
11	Setting prohibited

[bit7:6] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit5:0] FSFREQ: Frequency (64 piano scales) (Channel n = 0 to 4)

FSFREQ[5:0]	Frequency (Hz)	Musical scale
00	110.00	A2
01	116.54	A#2
02	123.47	B2
03	130.81	C3
04	138.59	C#3
05	146.83	D3
06	155.56	D#3
07	164.81	E3
08	174.61	F3
09	185.00	F#3
0A	196.00	G3
0B	207.65	G#3
0C	220.00	A3
0D	233.08	A#3
0E	246.94	B3
0F	261.63	C4
10	277.18	C#4
11	293.66	D4
12	311.13	D#4
13	329.63	E4
14	349.23	F4
15	369.99	F#4
16	392.00	G4
17	415.30	G#4
18	440.00	A4
19	466.16	A#4
1A	493.88	B4
1B	523.25	C5
1C	554.37	C#5
1D	587.33	D5
1E	622.25	D#5
1F	659.26	E5
20	698.46	F5
21	739.99	F#5
22	783.99	G5
23	830.61	G#5
24	880.00	A5
25	932.33	A#5
26	987.77	B5
27	1046.50	C6
28	1108.73	C#6
29	1174.66	D6

FSFREQ[5:0]	Frequency (Hz)	Musical scale
2A	1244.51	D#6
2B	1318.51	E6
2C	1396.91	F6
2D	1479.98	F#6
2E	1567.98	G6
2F	1661.22	G#6
30	1760.00	A6
31	1864.66	A#6
32	1975.53	B6
33	2093.00	C7
34	2217.46	C#7
35	2349.32	D7
36	2489.02	D#7
37	2637.02	E7
38	2793.83	F7
39	2959.96	F#7
3A	3135.96	G7
3B	3322.44	G#7
3C	3520.00	A7
3D	3729.31	A#7
3E	3951.07	B7
3F	4186.01	C8

4.8. Waveform Generator Channel n Control Register2 (WGCHnCTRL2, n=0 to 4)

REGISTER_NAME	Waveform Generator Channel n Control Register (WGCHnCTRL2)
OFFSET	n=0:0x024, n=1:0x034, n=2:0x044, n=3:0x054, n=4:0x064,
ACCESS_SIZE	B H W
MULTIPLE	31:0
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,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	Reserved	Reserved	Reserved	ITVAL2[4:0]				
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,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	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,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	Reserved	Reserved	Reserved	ITVAL1[4:0]				
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	Wp							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:21] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit20:16] ITVAL2: End idle time (Channel n = 0 to 4)

This function is not supported.

[bit15:5] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit4:0] ITVAL1: Start idle time (Channel n = 0 to 4)

Specifies a setting of 0 ms to 4 seconds.

ITVAL2 [4:0] ITVAL1 [4:0]	Description
00000	0 ms
00001	1 ms
00010	2 ms
00011	3 ms

ITVAL2 [4:0] ITVAL1 [4:0]	Description
00100	4 ms
00101	5 ms
00110	7.5 ms
00111	10 ms
01000	15 ms
01001	20 ms
01010	25 ms
01011	30 ms
01100	40 ms
01101	50 ms
01110	75 ms
01111	100 ms
10000	125 ms
10001	150 ms
10010	200 ms
10011	250 ms
10100	300 ms
10101	400 ms
10110	500 ms
10111	750 ms
11000	1000 ms
11001	1250 ms
11010	1500 ms
11011	2000 ms
11100	2500 ms
11101	3000 ms
11110	3500 ms
11111	4000 ms

4.9. Waveform Generator Channel n Control Register3 (WGCHnCTRL3, n=0 to 4)

REGISTER_NAME	Waveform Generator Channel n Control Register (WGCHnCTRL3)
OFFSET	n=0:0x028, n=1:0x038, n=2:0x048, n=3:0x058, n=4:0x068,
ACCESS_SIZE	B H W
MULTIPLE	31:0
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,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	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	FILTER[1:0]	
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	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	Reserved	Reserved	RELEASE[4:0]				
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	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	Reserved	Reserved	Reserved	ATTACK[4:0]				
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	Wp							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:18] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit17:16] FILTER : Low Pass Filter cutoff frequency (Channel n = 0 to 4)

Bit[1:0]	Description
00	Disable filter.
01	8 KHz
10	5 KHz
11	2 KHz

[bit15:13] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit12:8] RELEASE: Fade out time (Channel n = 0 to 4)

Specifies a setting of 1 ms to 1 second. Set these bits to 00000 when fade in control is not used.

[bit7:5] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit4:0] ATTACK: Fade in time (Channel n = 0 to 4)

Specifies a setting of 1 ms to 1 second. Set these bits to 00000 when fade out control is not used.

ATTACK[4:0] RELEASE[4:0]	Description
00000	Fade in/fade out control not used.
00001	1 ms
00010	2 ms
00011	3 ms
00100	4 ms
00101	5 ms
00110	7.5 ms
00111	10 ms
01000	15 ms
01001	20 ms
01010	25 ms
01011	30 ms
01100	40 ms
01101	50 ms
01110	75 ms
01111	100 ms
10000	125 ms
10001	150 ms
10010	200 ms
10011	250 ms
10100	300 ms
10101	400 ms
10110	500 ms
10111	750 ms
11000 to 11111	1000 ms

Note:

- *Exercise caution concerning use when performing consecutive sound source generation. See 3.1.5, 3.1.6.*
- *When using fade in control, consecutive fade in control cannot be used without fade out control. Also, when using fade out control, consecutive fade out control cannot be used without fade in control.*

4.10. Waveform Generator Channel Clear Register (WGCHCLR)

REGISTER_NAME	Waveform Generator Channel Clear Register (WGCHCLR)
OFFSET	0x070
ACCESS_SIZE	B H W
MULTIPLE	31:0
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,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	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,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	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,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	Reserved	Reserved	Reserved	CH4CL	CH3CL	CH2CL	CH1CL	CH0CL
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	Wp							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:5] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit4] CH4CL: Channel 4 sound source initialization

[bit3] CH3CL: Channel 3 sound source initialization

[bit2] CH2CL: Channel 2 sound source initialization

[bit1] CH1CL: Channel 1 sound source initialization

[bit0] CH0CL: Channel 0 sound source initialization

CHnCL (n=0 to 4)	Description	
	Read	Write
0	Not under initialization	Does not have any effect on operation.
1	Sound source specifications initialization wait	Sound source specifications initialization

Note:

- *When "1" is written to this bit, the set sound source specifications are initialized and sound source generation stops. However, this control does not initialize the source specification setting register but initialize WGCHSTART.*
- *Initializing sound source specifications and stopping generation automatically clears this bit.*
- *If sound source specifications are initialized, generation cannot be resumed from the point where it was stopped. To generate a source, sound source specifications need to be set again and sound source generation start control is required. See 3.1.*

4.11. Waveform Generator Interrupt Enable Register (WGINTREN)

REGISTER_NAME	Waveform Generator Interrupt Enable Register (WGINTREN)
OFFSET	0x080
ACCESS_SIZE	B H W
MULTIPLE	31:0
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,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	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,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	Reserved	Reserved	Reserved	CH4END	CH3END	CH2END	CH1END	CH0END
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	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	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	AHBERR
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R/W
PROT_TYPE	Wp							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:13] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit12] CH4END: Channel 4 sound source generation end interrupt enable

[bit11] CH3END: Channel 3 sound source generation end interrupt enable

[bit10] CH2END: Channel 2 sound source generation end interrupt enable

[bit9] CH1END: Channel 1 sound source generation end interrupt enable

[bit8] CH0END: Channel 0 sound source generation end interrupt enable

CHnEND (n=0 to 4)	Description
0	Disable interrupts.
1	Enable interrupts.

[bit7:1] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit0] AHBERR: AHB MASTER INTERFACE bus error interrupt enable

AHBERR	Description
0	Disable interrupts.
1	Enable interrupts.

4.12. Waveform Generator Interrupt State Register (WGINTRSTATE)

REGISTER_NAME	Waveform Generator Interrupt State Register (WGINTRSTATE)
OFFSET	0x084
ACCESS_SIZE	B H W
MULTIPLE	31:0
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,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	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,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	Reserved	Reserved	Reserved	CH4END	CH3END	CH2END	CH1END	CH0END
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R	R	R	R	R
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	AHBERR
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R
PROT_TYPE	Wp							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:13] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

This bit is read-only, and writing is prohibited.

Writing to this bit generates an SWFG AHB Slave interface access error.

[bit12] CH4END: Channel 4 sound source generation end interrupt status indication

[bit11] CH3END: Channel 3 sound source generation end interrupt status indication

[bit10] CH2END: Channel 2 sound source generation end interrupt status indication

[bit9] CH1END: Channel 1 sound source generation end interrupt status indication

[bit8] CH0END: Channel 0 sound source generation end interrupt status indication

CHnEND (n=0 to 4)	Description
0	No interrupt
1	Interrupt

Note:
- *This bit is read-only, and writing is prohibited.*
- *Writing to this bit generates an SWFG AHB Slave Interface access error.*

[bit7:1] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

This bit is read-only, and writing is prohibited.

Writing to this bit generates an SWFG AHB Slave interface access error.

[bit0] AHBERR: SWFG AHB MASTER INTERFACE bus error interrupt status indication

AHBERR	Description
0	No interrupt
1	Interrupt

Note:
- *This bit is read-only, and writing is prohibited.*
- *Writing to this bit generates an SWFG AHB Slave Interface access error interrupt.*

4.13. Waveform Generator Interrupt Clear Register (WGINTRCLR)

REGISTER_NAME	Waveform Generator Interrupt Clear Register (WGINTRCLR)
OFFSET	0x088
ACCESS_SIZE	B H W
MULTIPLE	31:0
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,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	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,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	Reserved	Reserved	Reserved	CH4END	CH3END	CH2END	CH1END	CH0END
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	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	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	AHBERR
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W
PROT_TYPE	Wp							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:13] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit12] CH4END: Channel 4 sound source generation end interrupt clear

[bit11] CH3END: Channel 3 sound source generation end interrupt clear

[bit10] CH2END: Channel 2 sound source generation end interrupt clear

[bit9] CH1END: Channel 1 sound source generation end interrupt clear

[bit8] CH0END: Channel 0 sound source generation end interrupt clear

CHnEND (n=0 to 4)	Description
0	Does not have any effect on operation.
1	Clear interrupts.

[bit7:1] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit0] AHBERR: SWFG AHB MASTER INTERFACE bus error interrupt clear

Explanation of AHBERR

AHBERR	Description
0	Does not have any effect on operation.
1	Clear interrupts.

Note:

- *If interrupt generation and interrupt clear occur simultaneously, interrupt generation is given priority.*

4.14. Waveform Generator AHB Bus Error Register (WGAHBERR)

REGISTER_NAME	Waveform Generator AHB Bus Error Register (WGAHBERR)
OFFSET	0x090
ACCESS_SIZE	B H W
MULTIPLE	31:0
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,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	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,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	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,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	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	AHBSERR	
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R	R
PROT_TYPE	Wp							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:2] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

This bit is read-only, and writing is prohibited.

Writing to this bit generates an SWFG AHB Slave interface access error.

[bit1:0] AHBSERR: SWFG AHB Slave Interface access error information indication

Bit[1:0]	Description
00	There is no error.
01	Address error
10	Write access to Read Only register
11	Access size error

Note:

- *This bit is read-only, and writing is prohibited.*
- *Writing to this bit generates an SWFG AHB Slave Interface access error.*

CHAPTER 20: Sound Mixer

This chapter explains the sound mixer.

1. Overview
2. Configuration and Block Diagram
3. Operation of the Sound Mixer
4. Registers
5. Appendix

CODE: SOUNDMIX-S6J3300-E1

1. Overview

This section gives a brief overview of the sound mixer.

Also refer to the chapter of "Sound System configuration" how to configure the sound mixer.

The sound mixer provides functions for mixing multiple input PCM (pulse code modulation) sound sources.

Table 20-1 Sound Mixer

Feature	Description
Input sound source	16-bit or 32-bit PCM sound source
Output sound source	32-bit PCM sound source
Number of input channels	Up to 10 channels of sound source input are supported.
Number of output channels	1ch
Input sampling rate	Mixing of sound sources with different sampling rates is supported. – WFG0 to 4 (five channels total): Input channels for fixed sampling rates: 48 kHz – PMIS0 to 4 (five channels total): Input channels for selectable sampling rates: 4, 8, 12, 24, 48, 96, and 44.1 kHz
Output sampling rate	48 kHz
Mixing System	Saturation calculation operation
FIR filter	Prevention of sound quality change by sampling rate conversion
Volume effects	The volume and the mute, fade in, and fade out effects of the input sound source and output sound source are controllable.

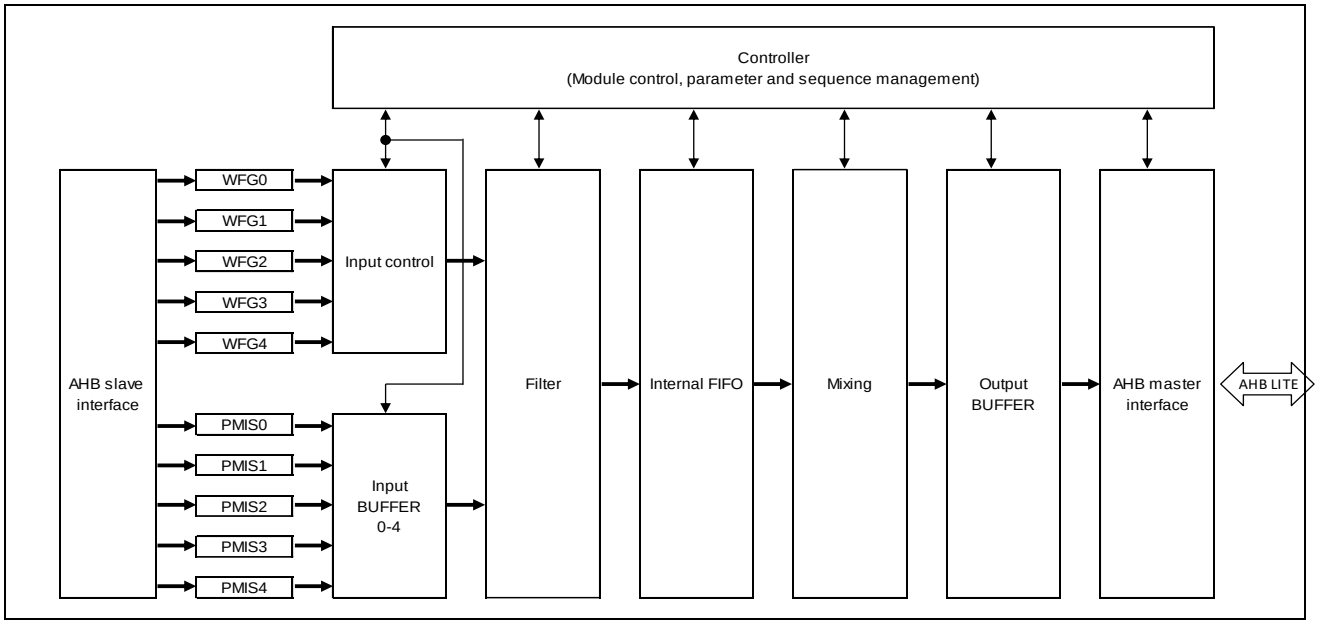
Note:

- *The input sound source is input in 32-bit units, regardless of the sound source format. A monaural sound source is input in 16-bit and 32-bit units.*
- *16-bit monaural sound source output is not supported.*
- *Sound waveform generator output may have a fixed connection to WFG0 to 4. It should be noted that there may be specification limits on specific models.*
- *The 44.1 kHz input sampling rate can be applied to only one of the input channels PMIS0 to 4. Operation when 44.1 kHz is applied to multiple channels cannot be guaranteed from the viewpoint of the mixing process load imposed by the sound mixer. It should be noted that there may limits on specific models concerning the selection of other frequencies.*
- *In the case of a sound source with a sampling rate that is not 48 kHz or 96 kHz, the sampling rate is converted internally by the mixer to 48 kHz. Noise removal by an FIR noise filter is performed before mixing to reduce change in sound quality due to distortion generated by this conversion. However, noise removal by FIR filtering is not performed in the case of down sampling of a 96 kHz sound source to 48 kHz. Because of this, there is the risk of sound quality being affected by aliasing generated at that time. The sampling rate must be determined with sufficient sound quality evaluation being implemented by the system.*
- *Mixed sound sources are calculated by the sound mixer's signed saturation calculation. Saturation calculation uses a pre-defined minimum value or maximum value when a calculation operation generates overflow. Mixing multiple high volume (large amplitude) sound sources creates the risk of sound quality being affected by saturation calculation rounding error. Sound mixing supports individual volume control for each channel, so adjustments can be made while evaluating the result on the system.*

2. Configuration and Block Diagram

This section shows a block diagram of the sound mixer.

Figure 20-1 Block Diagram of Sound Mixer



Note:

- *The mixing section is equipped with functions for use by volume effects (volume control, mute, fade in/fade out effects). Volume of each input channel and volume effects of output are applied before and after saturation calculation operations within each mixing operation.*
- *The AHB master outputs the mixed sound source to the address specified as the output destination.*
- *Every channel of WFG0 to 4 has an input buffer for a word (32 bit) in the part of "input control".*

3. Operation of the Sound Mixer

3.1. Basic Mixing Operation Procedure

3.1.1. Mixing Start

The mixing procedure started by sound mixing is shown below.

1. Disable input of the sound source. See MXCH under 4.1.
2. Set the output destination address and number of transfers of the mix sound source. See MXOCTRL under 4.2.
3. Configure sound source input channel PMIS0 through 4 sampling rate settings. See MXICTRL under 4.4.
4. Configure sound source input channel PMIS0 through 4 sound source process mode settings. See MXCHMONO under 4.5.
5. Initialize the sound source input channel PMIS0 through 4 input buffers. Also, initialize the mixed sound source output buffer. See MXBUFFCLR under 4.17.
6. Enable sound source input. See MXCH under 4.1.
7. Configure DMA transfer request and data transfer request interrupt threshold value settings for sound input channels PMIS0 through 4. See MXDRQCTRL under 4.3.
8. Configure interrupt enable/disable settings. See MXINTREN under 4.19.
9. Start sound source input.

Note:

- *The settings in steps 3, 4 and 7 are not required for sound source input channels WFG0 through 4. Also, other settings equivalent to these settings are not configured.*
- *Configure the input channels with the above settings in sequence from 1 through 9 in order to add new sound source input to an ongoing sound mixer operation and perform consecutive mixing. However, the output destination setting in step 2 is not required.*
- *The number of transfers of mixed sound sources from the sound mixer must match the number of output destination transfer requests. When the number of transfers on the sound mixer side is greater, the output destination will generate a DMA transfer error. When the number of transfer requests on the output destination side is greater, the output destination will not issue a DMA transfer request.*
- *If a sound source is not input on the actual device a sound source input is enabled, mixing is stopped for all channels (not only on the applicable channel).*
- *In the step 5, completion of the initialization is required before executing the next step. Completion can be confirmed by reading the applicable bit of the MXBUFFCLR register, because the bit is automatically cleared to 0 following initialization.*

3.1.2. Mixing Stop

Stopping mixing stops input of all channels. Removing a specific channel from mixing stops input of that channel. See MXCH under 4.1.

Notes:

If the DMAC is used for the sound source transmission, DMAEN bit of MXDRQCTRL set to 0 after the transmission is completed.

3.1.3. Fade In

- This setting sets the initial fade in time. See MXCHFADE1 to 5 under 4.10 to 4.14.
- Next, initialize the fade state. See MXFADECLR under 4.18.
- After that, enable fade in. See MXCHFADEEN under 4.16.

Note:

- *The fade in effect assumes that settings are configured before mixing and applied simultaneously with the start of mixing. However, they can also be applied while the mixing operation is in progress. In that case, the fade in process starts immediately.*
- *Be sure to initialize the fade state before fade in. Fade in following initialization will perform fade in control of the fade gain from -96 dB to 0 dB. However, this initialization operation is not necessarily required for fade in following fade out. This is because fade out causes gain to become -96 dB.*

3.1.4. Fade Out

Fade out can be applied only after fade in.

First, set the fade out time. See MXCHFADE1 to 5 under 4.10 to 4.14. After that, enable fade out. See MXCHFADEEN under 4.16.

Note:

- *The fade out effect assumes that settings are configured before the completion of mixing and applied simultaneously with the completion of mixing. However, they can also be applied while the mixing operation is in progress. In that case, the fade out process starts immediately.*
- *Be sure to initialize the fade state before fade out. Fade out following initialization will perform fade out control of the fade gain from 0 dB to -96 dB. However, this initialization operation is not necessarily required for fade out following fade in. This is because fade in causes gain to become 0 dB.*

3.2. Volume Effects

There are three volume effects: volume control, mute, and fade in/fade out.

These effects can be applied individually to sound sources being input/output.

In the case of an input sound source in particular, volume effect settings can be configured for each individual input channel (WFG0 to 4 and PMIS0 to 4).

Table 20-2 Volume Effects

Volume Effects	Description
Volume control	Volume is controlled in accordance with the gain set by software.
Mute	Mute outputs ALL 0 or ALL 1, regardless of the input/output sound source set software.
Fade in/fade out effects	Phased fade in and fade out are automatically controlled over the time set by software.

Notes:

- *Even if the minimum gain or mute is applied to the sound source, the mixer does not always output ALL 0 data. In that case, positive data become ALL 0, but negative data become ALL 1 (-1 in decimal). Due to this, slight sound may be heard in accordance with your output environment.*

3.2.1. Volume Control

Volume of the input/output sound source is controlled in accordance with the volume control gain set by software. This setting can be configured while operation is in progress, and settings are immediately reflected in operation.

For information about register settings, see 4.6, 4.7, and 4.8. For details about gain for volume control, see 5.1.

3.2.2. Mute

Outputs ALL 0 or ALL 1, regardless of the input/output sound source set software. This setting can be configured while operation is in progress, and settings are immediately reflected in operation.

For information about register settings, see 4.9.

3.2.3. Fade In/Fade Out Effects

Seamless fade in and fade out are controlled over the time set by software. The hardware internal fade in/out gain is changed automatically.

For information about the register setting procedure, see 3.1.3 and 3.1.4. For details about fade in and fade out time settings for volume control, see 5.2.

Note:

- *Volume control gain and fade in/gain are independent of each other. The product of both gains is applied as the overall volume effect.*
- *The fade in/out gain is 0 dB for both the fade in ending point and the fade out starting point. The gain of the overall volume effect is determined in accordance with the volume control gain set by software.*

3.3. Sound Source Input and Internal Mixing Process Mode

Input the input sound source to PMIS0 to 4 in 32-bit units, regardless of the sound source format. Also input using 32-bit units in the case of 16-bit monaural sound source input.

The sound source will be mixed internally by the mixer as a 32-bit stereo sound source. In other words, the upper 16 of the 32 bits is considered to be the left sound source, with the lower 16 bits are considered to be the right sound source, and this is mixed with the upper 16 bits and lower 16 bits of sound sources that are input from other channels.

For input of 16-bit sound sources, the n-th sound source is input as the upper 16 bits and the n+1 sound source is input as the lower 16 bits for 32-bit unit input. Note that n is a positive integer.

A processing mode can be selected from among those shown in Table 20-3 for monaural sound source internal processing.

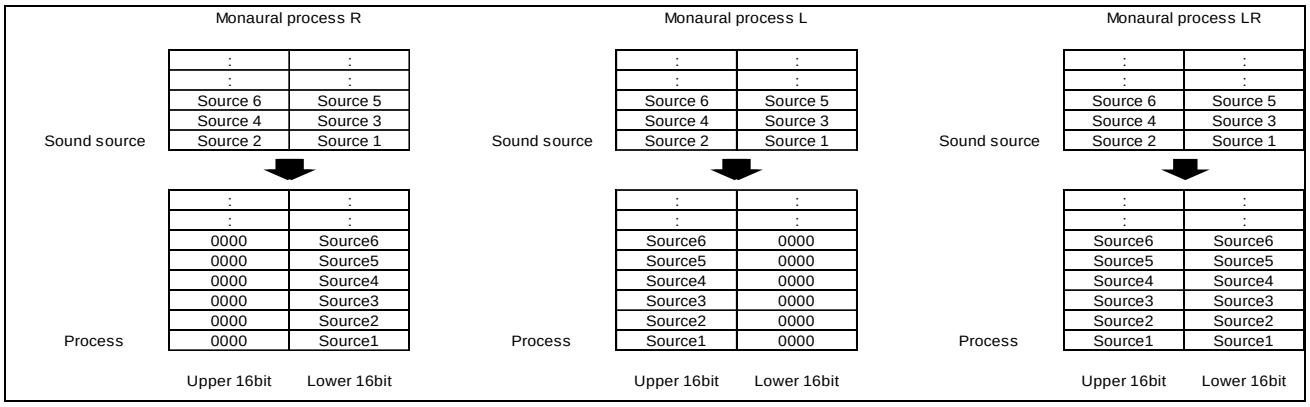
For information about configuring settings by software, see MXCHMONO under 4.5.

Table 20-3 Sound Source Internal Processing Mode

Processing Mode	Application	Internal Process Overview
Stereo processing	Stereo processing	Mixes input 32-bit stereo sound source.
Monaural processing R	Right-side mixing	Mixes by converting to 32-bit data (Upper 16 bits = ALL 0, Lower 16 bits = Monaural sound source).
Monaural processing L	Left-side mixing	Mixes by converting to 32-bit data ('Upper 16 bits = Monaural sound source, Lower 16 bits = ALL 0).
Monaural processing LR	Both side mixing (Left=Right)	Mixes by converting to 32-bit data (Upper 16 bits = Lower 16 bits = Monaural sound source).

An example of monaural processing is shown in Figure 20-2.

Figure 20-2 Example of Monaural Processing



Note:

- *A change in processing mode made while sound source is transferred to the sound source input channels PMIS0 through PMIS4 is prohibited.*
- *Monaural processing cannot be used on a 32-bit stereo sound source. Attempting to do so will result in improper sound.*

3.4. Interrupt

The sound mixer supports the interrupts described below.

- Data transfer request interrupt
  This interrupt is generated when a state is detected in which free space in the sound mixer PMIS0 to 4 input buffer 0 to 4 is greater than the FEST value, set in the MXDRQCTRL register, + 1. The interrupt is generated when the value is equal to or greater than the threshold value set by software. The sound mixer interrupt enable setting, interrupt clear control and status indication.
- Input FIFO overflow interrupt
  This interrupt is generated when an overflow is detected in the sound mixer PMIS0 to 4 input buffer 0 to 4. The sound mixer interrupt enable setting, interrupt clear control and status indication.
- DMA transfer error
  This interrupt is generated when a state is detected in which data input to the PMIS0 to 4 input buffer 0 to 4 is greater than the FEST value, set in the MXDRQCTRL register, + 1. The sound mixer interrupt enable setting, interrupt clear control and status indication.
- AHB Master Interface bus error interrupt
  This interrupt is generated when an error response is received during sound source output to an output destination on the AHB Master Interface of the sound mixer. The sound mixer interrupt enable setting, interrupt clear control and status indication.

3.5. Data Request Control

Mixer macros generate two types of data requests for WFG (WFGDATAREQ[4:0]) and for DMAC (MX_DMA_REQ[4:0]).

Data request protocol specifications

- Request assert conditions are described below (1 and 2).
- When ACK is received in response to a request, the request is negated.
- Data transfer is performed following request transfer.
- The data transfer volume is counted internally, and the sequence of transfer protocols is ended after the requested volume of data is complete.

  (The next sequential request cannot be asserted until the sequence of transfer protocols is complete.)

1. Data request for WFG
   There is a WFGDATAREQ[4:0] (for WFG) for each channel. WFGDATAREQ[0] corresponds to WFG Channel 1 while WFGDATAREQ[4] corresponds to WFG Channel 5. WFG is notified by a level signal when data receive is possible. HIGH level indicates that receive is possible.

2. Data request for DMAC
   Data request for DMAC (MX_DMA_REQ[4:0]) is a data request to DMAC that asserts the REQ signal of the applicable input channel when there is [threshold value + 1] of space in the input buffer. MX_DMA_REQ[0] is for PMIS Channel 0, while MX_DMA_REQ[4] is for PMIS Channel 4.

Notes:
If the DMAC is used for the sound source transmission, the corresponding DMAEN bit of MXDRQCTRL is set to 0 after the DMA-transmission is completed.

3.6. Data Output Control

Subordinate macro data transfer settings are regarded as DMAC INF.

A DMA request is received from a subordinate macro (DAC, PCMPWM, I2S), and [DATATN setting value + 1] data items are transferred to the macro specified by the MACRO field of the MXOCTRL register.

3.6.1. Function Details (Data Protocol Specifications)

1. Receive DMA request.
2. When [MXOCTRL register DATATN setting value + 1] or more of data is accumulated in the output buffer, send an ACK signal.
3. Following ACK transfer, send data. After data send is complete, end sequence of transfer protocols.
   (The next request cannot be accepted until the sequence of transfer protocols is complete.)

3.7. AHB Slave Interface

Mixer macros provide two AHB slave interface groups: for WFG interfacing and for CPU interfacing. AHB Slave interface specifications are shown in Table 20-4 and Table 20-5.

3.7.1. AHB Slave Interface (CPU/DMA)

Table 20-4 AHB Slave interface (CPU/DMA)

Item	Table_Header
Burst transfer	Mixer macros support burst transfer. However, all burst transfer is processed as a single transfer.
Protection control	not support
Response	OKAY and ERROR only
Access size	8,16,32 bit only
Width of address	13 bit

Note:

- *The following accesses cause the slave error response.*
- *1) When the access destination is an address that does not exist in the memory map*
- *2) The write access was done to the read only register.*
- *3) When the access size is not supported.*
- *(For example it accessed MXWFGnDADR/MXPMISnDADR with 8 bit or 16 bit) n=0 to 4*

3.7.2. WFG Dedicated AHB Slave Interface

Table 20-5 WFG AHB Slave interface

Item	Table_Header
Burst transfer	not support
Protection control	not support
Response	OKAY and ERROR only
Access size	8,16,32 bit only
Width of address	10 bit

Note:

- *The following accesses cause the slave error response.*
- *1) When the access destination is an address that does not exist in the memory map*
- *2) The write access was done to the read only register.*
- *3) When the access size is not supported.*
- *(For example it accessed MXWFGnDADR/MXPMISnDADR with 8 bit or 16 bit) n=0 to 4*

3.8. AHB Master Interface

Mixer macros provide an AHB master interface for interfacing with subordinate macros. AHB Master interface specifications are shown in Table 20-6.

Table 20-6 AHB Master interface

Item	Table_Header
Burst transfer	not support
Protection control	not support
Response	OKAY and ERROR only
Access size	8,16,32 bit only
Width of address	13 bit

Note:

- *When an ERROR response is received by transfer control from the CPU interface, an error response is transferred to the CPU interface.*
- *Information about the operation when an error response is received during mixer macro self-operation is stored in the information register and interrupt occurs.*

3.9. Input Buffer (FIFO)

Mixer macros provide a 5-channel input buffer for PCM/IS2 input. The input buffer is configured with one Dual Port RAM, and a one-channel 32-word area is configured by address segmentation.

The input buffer is accessed via the channel-use MXPMISnDADR0-15(n=0 to 4) register. Since access of the MXPMISnDADR0-15(n=0 to 4) register results in FIFO operation of the internal buffer, storage is in access sequence regardless of the access address. The configuration of the input buffer is shown in Table 20-7.

Table 20-7 Outline of input Buffer

Item	Description
Width of data	32 bit
Width of address	7 bit
Depth	160(32×5 ch)
Composition	Dual Port RAM

3.10. Sound Source Sampling Rate

3.10.1. Input Sampling Rate

Sound source input/output sampling rates are shown in Table 20-8.

Table 20-8 Input Sampling Rate

Input or Output	Frequency						
	96	48	24	12	8	4	44.1
WFG0 to 4	-	Supported	-	-	-	-	-
PMIS0 to 4	Supported	Supported	Supported	Supported	Supported	Supported	Supported
Mixer output	-	Supported	-	-	-	-	-

3.10.2. Sampling Rate Conversion and FIR Filter

The sound mixer uses saturation calculation to output a mixed sound source with a sampling rate of 48 kHz. When a sound source with a sampling rate that is not 48 kHz is input to PMIS0 to 4, the sound mixer converts the sampling rate.

Conversion is by up-sampling, along with processing by an FIR filter to dampen high frequency noise and prevent changes in sound quality. However, down-sampling is performed in the case of a 96 kHz sampling rate, without filtering.

A list of filter processes for input sampling is shown below.

Table 20-9 Filter Processes for Input Sampling

Sampling Rate (kHz) (Input Sound Source)	Sampling Rate Change	FIR Filter Cut-Off Frequency (kHz)
96	Perform sound sampling.	-
48	Conversion not performed.	-
24	Perform up-sampling and FIR filter to prevent changes in sound quality.	11.0
12		5.0
8		3.0
4		1.2
44.1		17.0

Note:

- *However, down-sampling is performed in the case of 96 kHz → 48 kHz, without filtering sampling rate, without filtering, which creates the risk of aliasing. Sufficient evaluation with the actual device is recommended.*
- *Filter characteristics have been selected based on Cypress rules.*
- *See 5.3 in detail.*

4. Registers

This section describes the registers of Sound Mixer

4.1. Mixer Channel Register (MXCH)

REGISTER_NAME	Mixer Channel Register (MXCH)
OFFSET	0x000
ACCESS_SIZE	B H W
MULTIPLE	31:0
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,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	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,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	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	PMIS4	PMIS3
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	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	PMIS2	PMIS1	PMIS0	WFG4	WFG3	WFG2	WFG1	WFG0
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

[bit31:10] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit9] PMIS4: PMIS4 input setting

[bit8] PMIS3: PMIS3 input setting

[bit7] PMIS2: PMIS2 input setting

[bit6] PMIS1: PMIS1 input setting

[bit5] PMIS0: PMIS0 input setting

PMISn (n=0 to 4)	Description
0	PMISn input invalid
1	PMISn input valid

Note:
PMISn bit must not be changed from 0 to 1 when DMAENCHn bit of MXDRQCTRL register is 1.

[bit4] WFG4: WFG4 input setting

[bit3] WFG3: WFG3 input setting

[bit2] WFG2: WFG2 input setting

[bit1] WFG1: WFG1 input setting

[bit0] WFG0: WFG0 input setting

WFGn (n=0 to 4)	Description
0	WFGn input invalid
1	WFGn input valid

Note:
- *Configuring invalid, the channel will be out of mixing operation not depending on the left data on buffer.*

4.2. Mixer Output Control Register (MXOCTRL)

REGISTER_NAME	Mixer Output Control Register (MXOCTRL)
OFFSET	0x004
ACCESS_SIZE	B H W
MULTIPLE	31:0
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,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	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,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	Reserved	Reserved	Reserved	Reserved	DATATN[3]	DATATN[2]	DATATN[1]	DATATN[0]
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	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	Reserved	Reserved	Reserved	Reserved	Reserved	MACRO[2]	MACRO[1]	MACRO[0]
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R/W	R/W	R/W
PROT_TYPE	Wp							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:12] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit11:8] DATATN[3:0]: Number of transfers to output destination

Bit[3:0]	Description
0000 to 1111	Transfers the [value written here + 1] number of data items to the output destination.

Note:

These bits can be set at the initialization. After that, it should not be changed.

[bit7:3] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit2:0] MACRO[2:0]: Output destination address

bit[2:0]	Description
2:0	Specifies [12:10] of the AHB bus address.

Note:

These bits can be set at the initialization. After that, it should not be changed.

4.3. Mixer Date Request Control Register (MXDRQCTRL)

REGISTER_NAME	Mixer Date Request Control Register (MXDRQCTRL)
OFFSET	0x008
ACCESS_SIZE	B H W
MULTIPLE	31:0
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved	Reserved	Reserved	DMAEN CH4	DMAEN CH3	DMAEN CH2	DMAE NCH1	DMAEN CH0
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	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	Reserved	Reserved	Reserved	Reserved	FESTCH4[3:0]			
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	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	FESTCH3[3:0]				FESTCH2[3:0]			
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

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	FESTCH1[3:0]				FESTCH0[3:0]			
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

[bit31:29] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit28] DMAENCH4: DMA transfer to PMIS4 request setting

[bit27] DMAENCH3: DMA transfer to PMIS3 request setting

[bit26] DMAENCH2: DMA transfer to PMIS2 request setting

[bit25] DMAENCH1: DMA transfer to PMIS1 request setting

[bit24] DMAENCH0: DMA transfer to PMIS0 request setting

DMAENCHn (n=0 to 4)	Description
0	Disable
1	Enable

[bit23:20] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit19:16] FESTCH4: Data transfer to PMIS4 request assert threshold setting

[bit15:12] FESTCH3: Data transfer to PMIS3 request assert threshold setting

[bit11:8] FESTCH2: Data transfer to PMIS2 request assert threshold setting

[bit7:4] FESTCH1: Data transfer to PMIS1 request assert threshold setting

[bit3:0] FESTCH0: Data transfer to PMIS0 request assert threshold setting

FESTCHn[3:0] (n=0 to 4)	Description
0000 to 1111	A data transfer request is asserted when the input buffer free space becomes [value written here + 1] word.

Note:

- *See Figure 20-2.*
- *When a 16-bit monaural sound source is input in 32-bit units into PMIS0 to 4, the FESTCHn upper limit setting value is 7. This is because twice the buffer capacity is needed to perform the internal process shown in 3.3 on the monaural sound source.*
- *FESTCHn bits must not be changed when DMAENCHn bit is 1.*

4.4. Mixer Input Control Register (MXICTRL)

REGISTER_NAME	Mixer Input Control Register (MXICTRL)
OFFSET	0x00C
ACCESS_SIZE	B H W
MULTIPLE	31:0
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,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	Reserved	Reserved	Reserved	Reserved	Reserved	PMIS4FREQ[2:0]		
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R/W	R/W	R/W
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	Reserved	PMIS3FREQ[2:0]			Reserved	PMIS2FREQ[2:0]		
ACCESS_TYPE	R0,W0	R/W	R/W	R/W	R0,W0	R/W	R/W	R/W
PROT_TYPE	Wp							
INITIAL_VALUE	0	0	0	1	0	0	0	1

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved	PMIS1FREQ[2:0]			Reserved	PMIS0FREQ[2:0]		
ACCESS_TYPE	R0,W0	R/W	R/W	R/W	R0,W0	R/W	R/W	R/W
PROT_TYPE	Wp							
INITIAL_VALUE	0	0	0	1	0	0	0	1

[bit31:19] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit18:16] PMIS4FREQ: Input sampling frequency to PMIS4

[bit15] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit14:12] PMIS3FREQ: Input sampling frequency to PMIS3

[bit11] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit10:8] PMIS2FREQ: Input sampling frequency to PMIS2

[bit7] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit6:4] PMIS1FREQ: Input sampling frequency to PMIS1

[bit3] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit2:0] PMIS0FREQ: Input sampling frequency to PMIS0

PMISnFREQ[2:0] (n=0 to 4)	Description
000	96 KHz
001	48 KHz
010	24 KHz
011	12 KHz
100	8 KHz
101	4 KHz
110	44.1 KHz
111	Reserved

Note:

- *Note that the initial value is 001 (48 kHz).*
- *The 44.1 kHz input sampling rate can be applied to only one of the input channels PMIS0 to 4. Operation when 44.1 kHz is applied to multiple channels cannot be guaranteed from the viewpoint of the mixing process load imposed by the sound mixer. It should be noted that there may limits on specific models concerning the selection of other frequencies.*

4.5. Mixer Channel Monaural Register (MXCHMONO)

REGISTER_NAME	Mixer Channel Monaural Register (MXCHMONO)
OFFSET	0x010
ACCESS_SIZE	B H W
MULTIPLE	31:0
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,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	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,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	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	PMIS4MONO[1:0]	
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	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	PMIS3MONO[1:0]		PMIS2MONO[1:0]		PMIS1MONO[1:0]		PMIS0MONO[1:0]	
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

[bit31:10] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit9:8] PMIS4MONO: PMIS4 sound source processing mode setting

[bit7:6] PMIS3MONO: PMIS3 sound source processing mode setting

[bit5:4] PMIS2MONO: PMIS2 sound source processing mode setting

[bit3:2] PMIS1MONO: PMIS1 sound source processing mode setting

[bit1:0] PMIS0MONO: PMIS0 sound source processing mode setting

PMISnMONO[1:0] (n=0 to 4)	Description
00	Stereo processing
01	Monaural process R (Upper 16 bits = ALL 0, Lower 16 bits = Monaural sound source).
10	Monaural process L (Upper 16 bits = Monaural sound source, Lower 16 bits = ALL 0).
11	Monaural process LR (Upper 16 bits = Lower 16 bits = Monaural sound source).

Note:
These bits must not be changed during sound source transmission of the channel.

4.6. Mixer Channel Volume1 Register (MXCHVOL1)

REGISTER_NAME	Mixer Channel Volume1 Register (MXCHVOL1)
OFFSET	0x020
ACCESS_SIZE	B H W
MULTIPLE	31:0
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	WFG3VOL[7:0]							
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	Wp							
INITIAL_VALUE	1	1	1	0	0	1	1	1

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	WFG2VOL[7:0]							
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	Wp							
INITIAL_VALUE	1	1	1	0	0	1	1	1

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	WFG1VOL[7:0]							
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	Wp							
INITIAL_VALUE	1	1	1	0	0	1	1	1

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	WFG0VOL[7:0]							
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	Wp							
INITIAL_VALUE	1	1	1	0	0	1	1	1

[bit31:24] WFG3VOL: Gain setting for WFG3 volume control

[bit23:16] WFG2VOL: Gain setting for WFG2 volume control

[bit15:8] WFG1VOL: Gain setting for WFG1 volume control

[bit7:0] WFG0VOL: Gain setting for WFG0 volume control

For details about Gain settings for volume control, see 5.1.

4.7. Mixer Channel Volume2 Register (MXCHVOL2)

REGISTER_NAME	Mixer Channel Volume2 Register (MXCHVOL2)
OFFSET	0x024
ACCESS_SIZE	B H W
MULTIPLE	31:0
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	PMIS2VOL[7:0]							
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	Wp							
INITIAL_VALUE	1	1	1	0	0	1	1	1

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	PMIS1VOL[7:0]							
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	Wp							
INITIAL_VALUE	1	1	1	0	0	1	1	1

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	PMIS0VOL[7:0]							
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	Wp							
INITIAL_VALUE	1	1	1	0	0	1	1	1

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	WFG4VOL[7:0]							
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	Wp							
INITIAL_VALUE	1	1	1	0	0	1	1	1

[bit31:24] PMIS2VOL: Gain setting for PMIS2VOL volume control

[bit23:16] PMIS1VOL: Gain setting for PMIS1VOL volume control

[bit15:8] PMIS0VOL: Gain setting for PMIS0VOL volume control

[bit7:0] WFG4VOL: Gain setting for WFG4 volume control

For details about Gain settings for volume control, see 5.1.

4.8. Mixer Channel Volume3 Register (MXCHVOL3)

REGISTER_NAME	Mixer Channel Volume3 Register (MXCHVOL3)
OFFSET	0x028
ACCESS_SIZE	B H W
MULTIPLE	31:0
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,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	MXDVOL[7:0]							
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	Wp							
INITIAL_VALUE	1	1	1	0	0	1	1	1

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	PMIS4VOL[7:0]							
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	Wp							
INITIAL_VALUE	1	1	1	0	0	1	1	1

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	PMIS3VOL[7:0]							
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	Wp							
INITIAL_VALUE	1	1	1	0	0	1	1	1

[bit31:24] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit23:16] MXDVOL: Gain setting for mixed sound source volume control

[bit15:8] PMIS4VOL: Gain setting for PMIS4VOL volume control

[bit7:0] PMIS3VOL: Gain setting for PMIS3VOL volume control

For details about Gain settings for volume control, see 5.1.

4.9. Mixer Channel Mute Register (MXCHMUTE)

REGISTER_NAME	Mixer Channel Mute Register (MXCHMUTE)
OFFSET	0x02C
ACCESS_SIZE	B H W
MULTIPLE	31:0
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,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	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,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	Reserved	Reserved	Reserved	Reserved	Reserved	MXD MUTE	PMIS4 MUTE	PMIS3 MUTE
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	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	PMIS2 MUTE	PMIS1 MUTE	PMIS0 MUTE	WFG4 MUTE	WFG3 MUTE	WFG2 MUTE	WFG1 MUTE	WFG0 MUTE
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

[bit31:11] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit10] MXDMUTE: Mixed sound source MUTE setting

Bit	Description
0	Disable mute.
1	Enable mute.

[bit9] PMIS4MUTE: PMIS4 input sound source MUTE setting

[bit8] PMIS3MUTE: PMIS3 input sound source MUTE setting

[bit7] PMIS2MUTE: PMIS2 input sound source MUTE setting

[bit6] PMIS1MUTE: PMIS1 input sound source MUTE setting

[bit5] PMIS0MUTE: PMIS0 input sound source MUTE setting

PMISnMUTE (n=0 to 4)	Description
0	Disable mute.
1	Enable mute.

[bit4] WFG4MUTE: WFG4 input sound source MUTE setting

[bit3] WFG3MUTE: WFG3 input sound source MUTE setting

[bit2] WFG2MUTE: WFG2 input sound source MUTE setting

[bit1] WFG1MUTE: WFG1 input sound source MUTE setting

[bit0] WFG0MUTE: WFG0 input sound source MUTE setting

WFGnMUTE (n=0 to 4)	Description
0	Disable mute.
1	Enable mute.

4.10. Mixer Channel Fade_in/out1 Register (MXCHFADE1)

REGISTER_NAME	Mixer Channel Fade_in/out1 Register (MXCHFADE1)
OFFSET	0x030
ACCESS_SIZE	B H W
MULTIPLE	31:0
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved	Reserved	Reserved	WFG1FADEOUT[4:0]				
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	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	Reserved	Reserved	Reserved	WFG1FADEIN[4:0]				
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	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	Reserved	Reserved	Reserved	WFG0FADEOUT[4:0]				
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	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	Reserved	Reserved	Reserved	WFG0FADEIN[4:0]				
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	Wp							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:29] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit28:24] WFG1FADEOUT: WFG1 fade out time setting

[bit23:21] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit20:16] WFG1FADEIN: WFG1 fade in time setting

[bit15:13] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit12:8] WFG0FADEOUT: WFG0 fade out time setting

[bit7:5] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit4:0] WFG0FADEIN: WFG0 fade in time setting

For details about fade in and fade out time settings for volume control, see 5.2.

4.11. Mixer Channel Fade_in/out2 Register (MXCHFADE2)

REGISTER_NAME	Mixer Channel Fade_in/out2 Register (MXCHFADE2)
OFFSET	0x034
ACCESS_SIZE	B H W
MULTIPLE	31:0
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved	Reserved	Reserved	WFG3FADEOUT[4:0]				
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	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	Reserved	Reserved	Reserved	WFG3FADEIN[4:0]				
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	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	Reserved	Reserved	Reserved	WFG2FADEOUT[4:0]				
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	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	Reserved	Reserved	Reserved	WFG2FADEIN[4:0]				
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	Wp							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:29] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit28:24] WFG3FADEOUT: WFG3 fade out time setting

[bit23:21] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit20:16] WFG3FADEIN: WFG3 fade in time setting

[bit15:13] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit12:8] WFG2FADEOUT: WFG2 fade out time setting

[bit7: 5] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit4:0] WFG2FADEIN: WFG2 fade in time setting

For details about fade in and fade out time settings for volume control, see 5.2.

4.12. Mixer Channel Fade_in/out3 Register (MXCHFADE3)

REGISTER_NAME	Mixer Channel Fade_in/out3 Register (MXCHFADE3)
OFFSET	0x038
ACCESS_SIZE	B H W
MULTIPLE	31:0
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved	Reserved	Reserved	PMIS0FADEOUT[4:0]				
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	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	Reserved	Reserved	Reserved	PMIS0FADEIN[4:0]				
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	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	Reserved	Reserved	Reserved	WFG4FADEOUT[4:0]				
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	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	Reserved	Reserved	Reserved	WFG4FADEIN[4:0]				
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	Wp							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:29] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit28:24] PMIS0FADEOUT: PMIS0 fade out time setting

[bit23:21] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit20:16] PMIS0FADEIN: PMIS0 fade in time setting

[bit15:13] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit12:8] WFG4FADEOUT: WFG4 fade out time setting

[bit7:5] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit4:0] WFG4FADEIN: WFG4 fade in time setting

For details about fade in and fade out time settings for volume control, see 5.2.

4.13. Mixer Channel Fade_in/out4 Register (MXCHFADE4)

REGISTER_NAME	Mixer Channel Fade_in/out4 Register (MXCHFADE4)
OFFSET	0x03C
ACCESS_SIZE	B H W
MULTIPLE	31:0
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved	Reserved	Reserved	PMIS2FADEOUT[4:0]				
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	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	Reserved	Reserved	Reserved	PMIS2FADEIN[4:0]				
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	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	Reserved	Reserved	Reserved	PMIS1FADEOUT[4:0]				
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	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	Reserved	Reserved	Reserved	PMIS1FADEIN[4:0]				
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	Wp							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:29] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit28:24] PMIS2FADEOUT: PMIS2 fade out time setting

[bit23:21] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit20:16] PMIS2FADEIN: PMIS2 fade in time setting

[bit15:13] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit12:8] PMIS1FADEOUT: PMIS1 fade out time setting

[bit7:5] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit4:0] PMIS1FADEIN: PMIS1 fade in time setting

For details about fade in and fade out time settings for volume control, see 5.2.

4.14. Mixer Channel Fade_in/out5 Register (MXCHFADE5)

REGISTER_NAME	Mixer Channel Fade_in/out5 Register (MXCHFADE5)
OFFSET	0x040
ACCESS_SIZE	B H W
MULTIPLE	31:0
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved	Reserved	Reserved	PMIS4FADEOUT[4:0]				
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	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	Reserved	Reserved	Reserved	PMIS4FADEIN[4:0]				
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	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	Reserved	Reserved	Reserved	PMIS3FADEOUT[4:0]				
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	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	Reserved	Reserved	Reserved	PMIS3FADEIN[4:0]				
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	Wp							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:29] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit28:24] PMIS4FADEOUT: PMIS4 fade out time setting

[bit23:21] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit20:16] PMIS4FADEIN: PMIS4 fade in time setting

[bit15:11] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit12:8] PMIS3FADEOUT: PMIS3 fade out time setting

[bit7:5] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit4:0] PMIS3FADEIN: PMIS3 fade in time setting

For details about fade in and fade out time settings for volume control, see 5.2.

4.15. Mixer Mixed Fade_in/out Register (MXMXDFADE)

REGISTER_NAME	Mixer Mixed Fade_in/out Register (MXMXDFADE)
OFFSET	0x044
ACCESS_SIZE	B H W
MULTIPLE	31:0
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,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	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,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	Reserved	Reserved	Reserved	MXDFADEOUT[4:0]				
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	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	Reserved	Reserved	Reserved	MXDFADEIN[4:0]				
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	Wp							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:13] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit12:8] MXDFADEOUT: Mixed sound source fade out time setting

[bit7:5] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit4:0] MXDFADEIN: Mixed sound source fade in time setting

For details about fade in and fade out time settings for volume control, see 5.2.

4.16. Mixer Channel Fade_in/out Enable Register (MXCHFADEEN)

REGISTER_NAME	Mixer Channel Fade_in/out Enable Register(MXCHFADEEN)
OFFSET	0x048
ACCESS_SIZE	B H W
MULTIPLE	31:0
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,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	Reserved	Reserved	MXDFADEEN[1:0]		PMIS4FADEEN[1:0]		PMIS3FADEEN[1:0]	
ACCESS_TYPE	R0,W0	R0,W0	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	PMIS2FADEEN[1:0]		PMIS1FADEEN[1:0]		PMIS0FADEEN[1:0]		WFG4FADEEN[1:0]	
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

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	WFG3FADEEN[1:0]		WFG2FADEEN[1:0]		WFG1FADEEN[1:0]		WFG0FADEEN[1:0]	
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

[bit31:22] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit21:20] MXDFADEEN: Mixed sound source fade in/out operation setting

Bit[1:0]	Description
00	No fade in/fade out
01	Setting prohibited
10	Enable fade in operation start.
11	Enable fade out operation start.

[bit19:18] PMIS4FADEEN: PMIS4 fade in/out operation setting

[bit17:16] PMIS3FADEEN: PMIS3 fade in/out operation setting

[bit15:14] PMIS2FADEEN: PMIS2 fade in/out operation setting

[bit13:12] PMIS1FADEEN: PMIS1 fade in/out operation setting

[bit11:10] PMIS0FADEEN: PMIS0 fade in/out operation setting

PMISnFADEEN[1:0] (n=0 to 4)	Description
00	No fade in/fade out
01	Setting prohibited
10	Enable fade in operation start.
11	Enable fade out operation start.

[bit9:8] WFG4FADEEN: WFG4 fade in/out operation setting

[bit7:6] WFG3FADEEN: WFG3 fade in/out operation setting

[bit5:4] WFG2FADEEN: WFG2 fade in/out operation setting

[bit3:2] WFG1FADEEN: WFG1 fade in/out operation setting

[bit1:0] WFG0FADEEN: WFG0 fade in/out operation setting

WFGnFADEEN[1:0] (n=0 to 4)	Description
00	No fade in/fade out
01	Setting prohibited
10	Enable fade in operation start.
11	Enable fade out operation start.

4.17. Mixer Buffer Clear Register (MXBUFFCLR)

REGISTER_NAME	Mixer Buffer Clear Register (MXBUFFCLR)
OFFSET	0x050
ACCESS_SIZE	B H W
MULTIPLE	31:0
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,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	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,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	Reserved	Reserved	Reserved	Reserved	Reserved	OUTBCLR	PMIS4 BCLR	PMIS3 BCLR
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	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	PMIS2 BCLR	PMIS1 BCLR	PMIS0 BCLR	WFG4 BCLR	WFG3 BCLR	WFG2 BCLR	WFG1 BCLR	WFG0 BCLR
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

[bit31:11] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit10] OUTBCLR: Output buffer initialization

Bit	Description	
	Read	Write
0	Not under initialization	Does not have any effect on operation.
1	Initialization wait	Initialize buffer (This bit automatically cleared to 0 following initialization.)

[bit9] PMIS4BCLR: PMIS4 input buffer initialization

[bit8] PMIS3BCLR: PMIS3 input buffer initialization

[bit7] PMIS2BCLR: PMIS2 input buffer initialization

[bit6] PMIS1BCLR: PMIS1 input buffer initialization

[bit5] PMIS0BCLR: PMIS0 input buffer initialization

PMISnBCLR (n=0 to 4)	Description	
	Read	Write
0	Not under initialization	Does not have any effect on operation.
1	Initialization wait	Initialize buffer (This bit automatically cleared to 0 following initialization.)

Note:

PMISn input buffer initialization must not be executed when DMAENCHn bit of MXDRQCTRL register is 1.

[bit4] WFG4BCLR: WFG4 input buffer initialization

[bit3] WFG3BCLR: WFG3 input buffer initialization

[bit2] WFG2BCLR: WFG2 input buffer initialization

[bit1] WFG1BCLR: WFG1 input buffer initialization

[bit0] WFG0BCLR: WFG0 input buffer initialization

WFGnBCLR (n=0 to 4)	Description	
	Read	Write
0	Not under initialization	Does not have any effect on operation.
1	Initialization wait	Initialize buffer (This bit automatically cleared to 0 following initialization.)

Note:

- *Every channel of WFG0 to 4 has an input buffer for a word (32 bit) in the part of "input control".*

4.18. Mixer FADE_in/out Clear Register (MXFADECLR)

REGISTER_NAME	Mixer FADE_in/out Clear Register (MXFADECLR)
OFFSET	0x054
ACCESS_SIZE	B H W
MULTIPLE	31:0
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,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	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,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	Reserved	Reserved	Reserved	Reserved	Reserved	MIXCLR	PMIS4 CLR	PMIS3 CLR
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	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	PMIS2 CLR	PMIS1 CLR	PMIS0 CLR	WFG4 CLR	WFG3 CLR	WFG2 CLR	WFG1 CLR	WFG0 CLR
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

[bit31:11] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit10] MIXCLR: Mixed data FADE state initialization

Bit	Description	
	Read	Write
0	Not under initialization	Does not have any effect on operation.
1	Initialization wait	Initialize fade state (This bit automatically cleared to 0 following initialization.)

[bit9] PMIS4CLR: PMIS4 FADE state initialization

[bit8] PMIS3CLR: PMIS3 FADE state initialization

[bit7] PMIS2CLR: PMIS2 FADE state initialization

[bit6] PMIS1CLR: PMIS1 FADE state initialization

[bit5] PMIS0CLR: PMIS0 FADE state initialization

PMISnCLR (n=0 to 4)	Description	
	Read	Write
0	Not under initialization	Does not have any effect on operation.
1	Initialization wait	Initialize fade state (This bit automatically cleared to 0 following initialization.)

[bit4] WFG4CLR: WFG4 FADE state initialization

[bit3] WFG3CLR: WFG3 FADE state initialization

[bit2] WFG2CLR: WFG2 FADE state initialization

[bit1] WFG1CLR: WFG1 FADE state initialization

[bit0] WFG0CLR: WFG0 FADE state initialization

WFGnCLR (n=0 to 4)	Description	
	Read	Write
0	Not under initialization	Does not have any effect on operation.
1	Initialization wait	Initialize fade state (This bit automatically cleared to 0 following initialization.)

Note:

- *"1" writing to initialize should be done after all the bits of MXFADECLR to be "0".*

4.19. Mixer Interrupt Enable Register (MXINTREN)

REGISTER_NAME	Mixer Interrupt Enable Register (MXINTREN)
OFFSET	0x060
ACCESS_SIZE	B H W
MULTIPLE	31:0
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	AHBERR	Reserved	Reserved	PMIS4 DMAERR	PMIS3 DMAERR	PMIS2 DMAERR	PMIS1 DMAERR	PMIS0 DMAERR
ACCESS_TYPE	R/W	R0,W0	R0,W0	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	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	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	Reserved	Reserved	Reserved	PMIS4 BUFOVFL	PMIS3 BUFOVFL	PMIS2 BUFOVFL	PMIS1 BUFOVFL	PMIS0 BUFOVFL
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	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	Reserved	Reserved	Reserved	PMIS4 BUFDREQ	PMIS3 BUFDREQ	PMIS2 BUFDREQ	PMIS1 BUFDREQ	PMIS0 BUFDREQ
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	Wp							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31] AHBERR: AHB Master interface bus error interrupt setting

Bit	Description
0	Disable interrupts.
1	Enable interrupts.

[bit30:29] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit28] PMIS4DMAERR: PMIS4DMA transfer error interrupt setting

[bit27] PMIS3DMAERR: PMIS3DMA transfer error interrupt setting

[bit26] PMIS2DMAERR: PMIS2DMA transfer error interrupt setting

[bit25] PMIS1DMAERR: PMIS1DMA transfer error interrupt setting

[bit24] PMIS0DMAERR: PMIS0DMA transfer error interrupt setting

PMISnDMAERR (n=0 to 4)	Description
0	Disable interrupts.
1	Enable interrupts.

[bit23:13] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit12] PMIS4BUFOVFL: PMIS4 input buffer overflow interrupt setting

[bit11] PMIS3BUFOVFL: PMIS3 input buffer overflow interrupt setting

[bit10] PMIS2BUFOVFL: PMIS2 input buffer overflow interrupt setting

[bit9] PMIS1BUFOVFL: PMIS1 input buffer overflow interrupt setting

[bit8] PMIS0BUFOVFL: PMIS0 input buffer overflow interrupt setting

PMISnBUFOVFL (n=0 to 4)	Description
0	Disable interrupts.
1	Enable interrupts.

[bit7:5] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit4] PMIS4BUFDREQ: PMIS4 data transfer request interrupt setting

[bit3] PMIS3BUFDREQ: PMIS3 data transfer request interrupt setting

[bit2] PMIS2BUFDREQ: PMIS2 data transfer request interrupt setting

[bit1] PMIS1BUFDREQ: PMIS1 data transfer request interrupt setting

[bit0] PMIS0BUFDREQ: PMIS0 data transfer request interrupt setting

PMISnBUFDREQ (n=0 to 4)	Description
0	Disable interrupts.
1	Enable interrupts.

Note:

- *When use of DMA request is disabled by the setting of MSDRQCTRL under 4.3, be sure to disable DMA transfer error interrupt. Failure to do so can result in generation of unexpected interrupts.*

4.20. Mixer Interrupt Status Register (MXINTRSTATE)

REGISTER_NAME	Mixer Interrupt Status Register (MXINTRSTATE)
OFFSET	0x064
ACCESS_SIZE	B H W
MULTIPLE	31:0
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	AHBERR	Reserved	Reserved	PMIS4 DMAERR	PMIS3 DMAERR	PMIS2 DMAERR	PMIS1 DMAERR	PMIS0 DMAERR
ACCESS_TYPE	R	R0,W0	R0,W0	R	R	R	R	R
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,W0	R0,W0	R0,W0	R	R	R	R	R
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	PMIS4 BUFOVFL	PMIS3 BUFOVFL	PMIS2 BUFOVFL	PMIS1 BUFOVFL	PMIS0 BUFOVFL
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R	R	R	R	R
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	PMIS4 BUFDREQ	PMIS3 BUFDREQ	PMIS2 BUFDREQ	PMIS1 BUFDREQ	PMIS0 BUFDREQ
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R	R	R	R	R
PROT_TYPE	Wp							
INITIAL_VALUE	0	0	0	1	1	1	1	1

[bit31] AHBERR: AHB master interface bus error indication

Bit	Description
0	No error/no transfer request
1	Error/transfer request present

Notes:

- *This bit is read-only, and writing is prohibited.*
- *Writing to this bit generates a sound mixer AHB Slave Interface access error.*

[bit30:29] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

This bit is read-only, and writing is prohibited.

Writing to this bit generates a sound mixer AHB Slave interface access error.

[bit28] PMIS4DMAERR: PMIS4 DMA transfer error indication

[bit27] PMIS3DMAERR: PMIS3 DAM transfer error indication

[bit26] PMIS2DMAERR: PMIS2DMA transfer error indication

[bit25] PMIS1DMAERR: PMIS1DMA transfer error indication

[bit24] PMIS0DMAERR: PMIS0DMA transfer error indication

PMISnDMAERR (n=0 to 4)	Description
0	No error/no transfer request
1	Error/transfer request present

Notes:
- *This bit is read-only, and writing is prohibited.*
- *Writing to this bit generates a sound mixer AHB Slave Interface access error.*

[bit23:13] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

This bit is read-only, and writing is prohibited.

Writing to this bit generates a sound mixer AHB Slave interface access error.

[bit12] PMIS4BUFOVFL: PMIS4 input buffer overflow error indication

[bit11] PMIS3BUFOVFL: PMIS3 input buffer overflow error indication

[bit10] PMIS2BUFOVFL: PIS2 input buffer overflow error indication

[bit9] PMIS1BUFOVFL: PMIS1 input buffer overflow error indication

[bit8] PMIS0BUFOVFL: PMIS0 input buffer overflow error indication

PMISnBUFOVFL (n=0 to 4)	Description
0	No error/no transfer request
1	Error/transfer request present

Notes:
- *This bit is read-only, and writing is prohibited.*
- *Writing to this bit generates a sound mixer AHB Slave Interface access error.*

[bit7:5] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

This bit is read-only, and writing is prohibited.

Writing to this bit generates a sound mixer AHB Slave interface access error.

[bit4] PMIS4BUFDREQ: PMIS4 data transfer request indication

[bit3] PMIS3BUFDREQ: PMIS3 data transfer request indication

[bit2] PMIS2BUFDREQ: PMIS2 data transfer request indication

[bit1] PMIS1BUFDREQ: PMIS1 data transfer request indication

[bit0] PMIS0BUFDREQ: PMIS0 data transfer request indication

PMISnBUFDREQ (n=0 to 4)	Description
0	No error/no transfer request
1	Error/transfer request present

Note:
- *This bit is read-only, and writing is prohibited.*
- *Writing to this bit generates a sound mixer AHB Slave Interface access error.*

4.21. Mixer Interrupt Clear Register (MXINTRCLR)

REGISTER_NAME	Mixer Interrupt Clear Register (MXINTRCLR)
OFFSET	0x068
ACCESS_SIZE	B H W
MULTIPLE	31:0
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	AHBERR	Reserved	Reserved	PMIS4 DMAERR	PMIS3 DMAERR	PMIS2 DMAERR	PMIS1 DMAERR	PMIS0 DMAERR
ACCESS_TYPE	RX,W	RX,W0	RX,W0	RX,W	RX,W	RX,W	RX,W	RX,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	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	RX,W0	RX,W0	RX,W0	RX,W	RX,W	RX,W	RX,W	RX,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	Reserved	Reserved	PMIS4 BUFOVFL	PMIS3 BUFOVFL	PMIS2 BUFOVFL	PMIS1 BUFOVFL	PMIS0 BUFOVFL
ACCESS_TYPE	RX,W0	RX,W0	RX,W0	RX,W	RX,W	RX,W	RX,W	RX,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	PMIS4 BUFDREQ	PMIS3 BUFDREQ	PMIS2 BUFDREQ	PMIS1 BUFDREQ	PMIS0 BUFDREQ
ACCESS_TYPE	RX,W0	RX,W0	RX,W0	RX,W	RX,W	RX,W	RX,W	RX,W
PROT_TYPE	Wp							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31] AHBERR: AHB master interface bus error interrupt clear

Bit	Description
0	Does not have any effect on operation.
1	Clear interrupts.

[bit30:29] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit28] PMIS4DMAERR: PMIS4 DMA transfer error interrupt clear

[bit27] PMIS3DMAERR: PMIS3 DMA transfer error interrupt clear

[bit26] PMIS2DMAERR: PMIS2DMA transfer error interrupt clear

[bit25] PMIS1DMAERR: PMIS1DMA transfer error interrupt clear

[bit24] PMIS0DMAERR: PMIS0DMA transfer error interrupt clear

PMISnDMAERR (n=0 to 4)	Description
0	Does not have any effect on operation.
1	Clear interrupts.

[bit23:13] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit12] PMIS4BUFOVFL: PMIS4 input buffer overflow interrupt clear

[bit11] PMIS3BUFOVFL: PMIS3 input buffer overflow interrupt clear

[bit10] PMIS2BUFOVFL: PMIS2 input buffer overflow interrupt clear

[bit9] PMIS1BUFOVFL: PMIS1 input buffer overflow interrupt clear

[bit8] PMIS0BUFOVFL: PMIS0 input buffer overflow interrupt clear

PMISnBUFOVFL (n=0 to 4)	Description
0	Does not have any effect on operation.
1	Clear interrupts.

[bit7:5] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit4] PMIS4BUFDREQ: PMIS4 data transfer request interrupt clear

[bit3] PMIS3BUFDREQ: PMIS3 data transfer request interrupt clear

[bit2] PMIS2BUFDREQ: PMIS2 data transfer request interrupt clear

[bit1] PMIS1BUFDREQ: PMIS1 data transfer request interrupt clear

[bit0] PMIS0BUFDREQ: PMIS0 data transfer request interrupt clear

PMISnBUFDREQ (n=0 to 4)	Description
0	Does not have any effect on operation.
1	Clear interrupts.

Note:

- *If interrupt generation and interrupt clear occur simultaneously, interrupt generation is given priority.*

4.22. Mixer Input Buffer Count1 Register (MXINBUFFCNT1)

REGISTER_NAME	Mixer Input Buffer Count1 Register (MXINBUFFCNT1)
OFFSET	0x070
ACCESS_SIZE	B H W
MULTIPLE	31:0
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved	Reserved	PMIS2CNT[5:0]					
ACCESS_TYPE	R0,W0	R0,W0	R	R	R	R	R	R
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	PMIS1CNT[5:0]					
ACCESS_TYPE	R0,W0	R0,W0	R	R	R	R	R	R
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	PMIS0CNT[5:0]					
ACCESS_TYPE	R0,W0	R0,W0	R	R	R	R	R	R
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	WFG4C	WFG3C	WFG2C	WFG1C	WFG0C
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R	R	R	R	R
PROT_TYPE	Wp							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:30] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

This bit is read-only, and writing is prohibited.

Writing to this bit generates a sound mixer AHB Slave interface access error.

[bit29:24] PMIS2CNT[5:0]: PMIS2 input buffer used volume indication

Bit[5:0]	Description
000000	Buffer is empty.
000001 to 111111	Used input buffer volume

Notes:

- *This bit is read-only, and writing is prohibited.*
- *Writing to this bit generates a sound mixer AHB Slave Interface access error.*

[bit23:22] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

This bit is read-only, and writing is prohibited.

Writing to this bit generates a sound mixer AHB Slave interface access error.

[bit21:16] PMIS1CNT[5:0]: PMIS1 input buffer used volume indication

Bit[5:0]	Description
000000	Buffer is empty.
000001 to 111111	Used input buffer volume

Notes:

- *This bit is read-only, and writing is prohibited.*
- *Writing to this bit generates a sound mixer AHB Slave Interface access error.*

[bit15:14] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

This bit is read-only, and writing is prohibited.

Writing to this bit generates a sound mixer AHB Slave interface access error.

[bit13:8] PMIS0CNT[5:0]: PMIS0 input buffer used volume indication

Bit[5:0]	Description
000000	Buffer is empty.
000001 to 111111	Used input buffer volume

Note:

- *This bit is read-only, and writing is prohibited.*
- *Writing to this bit generates a sound mixer AHB Slave Interface access error.*

[bit7:5] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

This bit is read-only, and writing is prohibited.

Writing to this bit generates a sound mixer AHB Slave interface access error.

[bit4] WFG4C: WFG4 input data present/not present indication

[bit3] WFG3C: WFG3 input data present/not present indication

[bit2] WFG2C: WFG2 input data present/not present indication

[bit1] WFG1C: WFG1 input data present/not present indication

[bit0] WFG0C: WFG0 input data present/not present indication

WFGnC (n=0 to 4)	Description
0	Without data
1	With data

Note:

- *This bit is read-only, and writing is prohibited.*
- *Writing to this bit generates a sound mixer AHB Slave Interface access error.*

4.23. Mixer Input Buffer Count2 Register (MXINBUFFCNT2)

REGISTER_NAME	Mixer Input Buffer Count2 Register (MXINBUFFCNT2)
OFFSET	0x074
ACCESS_SIZE	B H W
MULTIPLE	31:0
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,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	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,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	Reserved	Reserved	PMIS4CNT[5:0]					
ACCESS_TYPE	R0,W0	R0,W0	R	R	R	R	R	R
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	PMIS3CNT[5:0]					
ACCESS_TYPE	R0,W0	R0,W0	R	R	R	R	R	R
PROT_TYPE	Wp							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:14] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

This bit is read-only, and writing is prohibited.

Writing to this bit generates a sound mixer AHB Slave interface access error.

[bit13:8] PMIS4CNT[5:0]: Input buffer used volume indication

Bit[5:0]	Description
000000	Buffer is empty.
000001 to 111111	Used input buffer volume

Note:

- *This bit is read-only, and writing is prohibited.*
- *Writing to this bit generates a sound mixer AHB Slave Interface access error.*

[bit7:6] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

This bit is read-only, and writing is prohibited.

Writing to this bit generates a sound mixer AHB Slave interface access error.

[bit5:0] PMIS3CNT[5:0]: Input buffer used volume indication

Bit[5:0]	Description
000000	Buffer is empty.
000001 to 111111	Used input buffer volume

Note:

- *This bit is read-only, and writing is prohibited.*
- *Writing to this bit generates a sound mixer AHB Slave Interface access error.*

4.24. Mixer Channel Buffer Count Register (MXCHBUFFCNT)

REGISTER_NAME	Mixer Input Buffer Count Register (MXCHBUFFCNT)
OFFSET	0x078
ACCESS_SIZE	B H W
MULTIPLE	31:0
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved	Reserved	Reserved	Reserved	PMIS4CNT[3:0]			
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R	R	R	R
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	PMIS3CNT[3:0]				PMIS2CNT[3:0]			
ACCESS_TYPE	R	R	R	R	R	R	R	R
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	PMIS1CNT[3:0]				PMIS0CNT[3:0]			
ACCESS_TYPE	R	R	R	R	R	R	R	R
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	WFG4C	WFG3C	WFG2C	WFG1C	WFG0C
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R	R	R	R	R
PROT_TYPE	Wp							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:28] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

This bit is read-only, and writing is prohibited.

Writing to this bit generates a sound mixer AHB Slave interface access error.

[bit27:24] PMIS4CNT[3:0]: PMIS4 internal buffer used volume indication

[bit23:20] PMIS3CNT[3:0]: PMIS3 internal buffer used volume indication

[bit19:16] PMIS2CNT[3:0]: PMIS2 internal buffer used volume indication

[bit15:12] PMIS1CNT[3:0]: PMIS1 internal buffer used volume indication

[bit11:8] PMIS0CNT[3:0]: PMIS0 internal buffer used volume indication

PMISnCNT[3:0] (n=0 to 4)	Description
000000	Buffer is empty.
000001 to 111111	Internal buffer used volume display

Notes:
- *This bit is read-only, and writing is prohibited.*
- *Writing to this bit generates a sound mixer AHB Slave Interface access error.*

[bit7:5] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

This bit is read-only, and writing is prohibited.

Writing to this bit generates a sound mixer AHB Slave interface access error.

[bit4] WFG4C: WFG4 internal buffer used volume indication

[bit3] WFG3C: WFG3 internal buffer used volume indication

[bit2] WFG2C: WFG2 internal buffer used volume indication

[bit1] WFG1C: WFG1 internal buffer used volume indication

[bit0] WFG0C: WFG0 internal buffer used volume indication

WFGnC (n=0 to 4)	Description
0	Buffer is empty.
1	Internal buffer used

Note:
- *This bit is read-only, and writing is prohibited.*
- *Writing to this bit generates a sound mixer AHB Slave Interface access error.*

4.25. Mixer Output Buffer Count Register (MXOUTBUFFCNT)

REGISTER_NAME	Mixer Output Buffer Count Register (MXOUTBUFFCNT)
OFFSET	0x07C
ACCESS_SIZE	B H W
MULTIPLE	31:0
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,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	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,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	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,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	Reserved	Reserved	OUTCNT[5:0]					
ACCESS_TYPE	R0,W0	R0,W0	R	R	R	R	R	R
PROT_TYPE	Wp							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:6] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

This bit is read-only, and writing is prohibited.

Writing to this bit generates a sound mixer AHB Slave interface access error.

[bit5:0] OUTCNT[5:0]: Output buffer used volume indication

Bit[5:0]	Description
000000	Buffer is empty.
000001 to 111111	Output buffer used volume

Note:

- *This bit is read-only, and writing is prohibited.*
- *Writing to this bit generates a sound mixer AHB Slave Interface access error.*

4.26. Mixer AHB Bus Error Register (MXAHBERR)

REGISTER_NAME	Mixer AHB Bus Error Register (MXAHBERR)
OFFSET	0x080
ACCESS_SIZE	B H W
MULTIPLE	31:0
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,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	Reserved	Reserved	CNTREGERR[1:0]		PMIS4ERR[1:0]		PMIS3ERR[1:0]	
ACCESS_TYPE	R0,W0	R0,W0	R	R	R	R	R	R
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	PMIS2ERR[1:0]		PMIS1ERR[1:0]		PMIS0ERR[1:0]		WFG4ERR[1:0]	
ACCESS_TYPE	R	R	R	R	R	R	R	R
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	WFG3ERR[1:0]		WFG2ERR[1:0]		WFG1ERR[1:0]		WFG0ERR[1:0]	
ACCESS_TYPE	R	R	R	R	R	R	R	R
PROT_TYPE	Wp							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:22] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

This bit is read-only, and writing is prohibited.

Writing to this bit generates a sound mixer AHB Slave interface access error.

[bit21:20] CNTREGERR[1:0]: AHB Slave interface register access error information indication

Bit[1:0]	Description
00	There is no error.
01	Address error
10	Write access to Read Only register
11	Access size error

[bit19:18] PMIS4ERR[1:0]: AHB Slave interface PMIS4 transfer access error information indication

[bit17:16] PMIS3ERR[1:0]: AHB Slave interface PMIS3 transfer access error information indication

[bit15:14] PMIS2ERR[1:0]: AHB Slave interface PMIS2 transfer access error information indication

[bit13:12] PMIS1ERR[1:0]: AHB Slave interface PMIS1 transfer access error information indication

[bit11:10] PMIS0ERR[1:0]: AHB Slave interface PMIS0 transfer access error information indication

PMISnERR[1:0] (n=0 to 4)	Description
00	There is no error.
01	Address error
10	Write access to Read Only register
11	Access size error

[bit9:8] WFG4ERR[1:0]: AHB Slave interface WFG4 transfer access error information indication

[bit7:6] WFG3ERR[1:0]: AHB Slave interface WFG3 transfer access error information indication

[bit5:4] WFG2ERR[1:0]: AHB Slave interface WFG2 transfer access error information indication

[bit3:2] WFG1ERR[1:0]: AHB Slave interface WFG1 transfer access error information indication

[bit1:0] WFG0ERR[1:0]: AHB Slave interface WFG0 transfer access error information indication

WFGnERR[1:0] (n=0 to 4)	Description
00	There is no error.
01	Address error
10	Write access to Read Only register
11	Access size error

Note:
- *This bit is read-only, and writing is prohibited.*
- *Writing to this bit generates a Mixer AHB Slave Interface access error interrupt.*
- *Every address error during transmission to WFG0 through 4 is informed to WFG0ERR[1:0]. It is also informed to CNTREGERR[1:0] during transmission to PMIS0 through 4.*
- *Every write access error of Read Only register is informed to CNTREERR[1:0].*

4.27. Mixer WFGn Data Address Register (MXWFGnDADR, n=0 to 4)

REGISTER_NAME	Mixer WFGn Data Address Register (MXWFGnDADR, n=0 to 4)
OFFSET	0x100(n=0), 0x104(n=1), 0x108(n=2), 0x10C(n=3), 0x110(n=4)
ACCESS_SIZE	W
MULTIPLE	31:0
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	WFGnDADR[31:24]							
ACCESS_TYPE	W	W	W	W	W	W	W	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	WFGnDADR[23:16]							
ACCESS_TYPE	W	W	W	W	W	W	W	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	WFGnDADR[15:8]							
ACCESS_TYPE	W	W	W	W	W	W	W	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	WFGnDADR[7:0]							
ACCESS_TYPE	W	W	W	W	W	W	W	W
PROT_TYPE	Wp							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:0] WFGnDADR [31:0]: WFGn input data register

Note:

- *Because Sound waveform generator output has a fixed connection to this register, this register cannot be written to from the CPU. Writing generates an error.*

4.28. Mixer PCM/I2S n Data Address Register0-15 (MXPMISnDADR0-15, n=0 to 4)

REGISTER_NAME	Mixer PCM/I2S n Data Address Register (MXPMISnDADR, n=0 to 4)
OFFSET	0x200 to 0x23C(n=0), 0x240 to 0x27C(n=1), 0x280 to 0x2BC(n=2) 0x2C0 to 0x2FC(n=3), 0x300 to 0x33C(n=4)
ACCESS_SIZE	W
MULTIPLE	31:0
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	PMISnDADR0 to 15[31:24]							
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	PMISnDADR0 to 15[23:16]							
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	PMISnDADR0 to 15[15:8]							
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	PMISnDADR0 to 15[7:0]							
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] PMISnDADR0 to 15[31:0]: PMIS input data register

5. Appendix

5.1. Gain for Volume Control

Table 20-10 Gain for Volume Control

Bit[7:0]	Description	
	Gain (dB)	Magnification
00000000	-96.0	0.000016
00000001	-96.0	0.000016
00000010	-96.0	0.000016
00000011	-96.0	0.000016
00000100	-96.0	0.000016
00000101	-96.0	0.000016
00000110	-96.0	0.000016
00000111	-96.0	0.000016
00001000	-96.0	0.000016
00001001	-96.0	0.000016
00001010	-96.0	0.000016
00001011	-96.0	0.000016
00001100	-96.0	0.000016
00001101	-96.0	0.000016
00001110	-96.0	0.000016
00001111	-96.0	0.000016
00010000	-96.0	0.000016
00010001	-96.0	0.000016
00010010	-96.0	0.000016
00010011	-96.0	0.000016
00010100	-96.0	0.000016
00010101	-96.0	0.000016
00010110	-96.0	0.000016
00010111	-96.0	0.000016
00011000	-96.0	0.000016
00011001	-96.0	0.000016
00011010	-96.0	0.000016
00011011	-96.0	0.000016
00011100	-96.0	0.000016
00011101	-96.0	0.000016
00011110	-96.0	0.000016
00011111	-96.0	0.000016
00100000	-96.0	0.000016
00100001	-96.0	0.000016
00100010	-96.0	0.000016
00100011	-96.0	0.000016

Bit[7:0]	Description	
	Gain (dB)	Magnification
00100100	-96.0	0.000016
00100101	-96.0	0.000016
00100110	-96.0	0.000016
00100111	-96.0	0.000016
00101000	-95.5	0.000017
00101001	-95.0	0.000018
00101010	-94.5	0.000019
00101011	-94.0	0.000020
00101100	-93.5	0.000021
00101101	-93.0	0.000022
00101110	-92.5	0.000024
00101111	-92.0	0.000025
00110000	-91.5	0.000027
00110001	-91.0	0.000028
00110010	-90.5	0.000030
00110011	-90.0	0.000032
00110100	-89.5	0.000033
00110101	-89.0	0.000035
00110110	-88.5	0.000038
00110111	-88.0	0.000040
00111000	-87.5	0.000042
00111001	-87.0	0.000045
00111010	-86.5	0.000047
00111011	-86.0	0.000050
00111100	-85.5	0.000053
00111101	-85.0	0.000056
00111110	-84.5	0.000060
00111111	-84.0	0.000063
01000000	-83.5	0.000067
01000001	-83.0	0.000071
01000010	-82.5	0.000075
01000011	-82.0	0.000079
01000100	-81.5	0.000084
01000101	-81.0	0.000089
01000110	-80.5	0.000094
01000111	-80.0	0.000100
01001000	-79.5	0.000106
01001001	-79.0	0.000112
01001010	-78.5	0.000119
01001011	-78.0	0.000126

Bit[7:0]	Description	
	Gain (dB)	Magnification
01001100	-77.5	0.000133
01001101	-77.0	0.000141
01001110	-76.5	0.000150
01001111	-76.0	0.000158
01010000	-75.5	0.000168
01010001	-75.0	0.000178
01010010	-74.5	0.000188
01010011	-74.0	0.000200
01010100	-73.5	0.000211
01010101	-73.0	0.000224
01010110	-72.5	0.000237
01010111	-72.0	0.000251
01011000	-71.5	0.000266
01011001	-71.0	0.000282
01011010	-70.5	0.000299
01011011	-70.0	0.000316
01011100	-69.5	0.000335
01011101	-69.0	0.000355
01011110	-68.5	0.000376
01011111	-68.0	0.000398
01100000	-67.5	0.000422
01100001	-67.0	0.000447
01100010	-66.5	0.000473
01100011	-66.0	0.000501
01100100	-65.5	0.000531
01100101	-65.0	0.000562
01100110	-64.5	0.000596
01100111	-64.0	0.000631
01101000	-63.5	0.000668
01101001	-63.0	0.000708
01101010	-62.5	0.000750
01101011	-62.0	0.000794
01101100	-61.5	0.000841
01101101	-61.0	0.000891
01101110	-60.5	0.000944
01101111	-60.0	0.001000
01110000	-59.5	0.001059
01110001	-59.0	0.001122
01110010	-58.5	0.001189
01110011	-58.0	0.001259

Bit[7:0]	Description	
	Gain (dB)	Magnification
01110100	-57.5	0.001334
01110101	-57.0	0.001413
01110110	-56.5	0.001496
01110111	-56.0	0.001585
01111000	-55.5	0.001679
01111001	-55.0	0.001778
01111010	-54.5	0.001884
01111011	-54.0	0.001995
01111100	-53.5	0.002113
01111101	-53.0	0.002239
01111110	-52.5	0.002371
01111111	-52.0	0.002512
10000000	-51.5	0.002661
10000001	-51.0	0.002818
10000010	-50.5	0.002985
10000011	-50.0	0.003162
10000100	-49.5	0.003350
10000101	-49.0	0.003548
10000110	-48.5	0.003758
10000111	-48.0	0.003981
10001000	-47.5	0.004217
10001001	-47.0	0.004467
10001010	-46.5	0.004732
10001011	-46.0	0.005012
10001100	-45.5	0.005309
10001101	-45.0	0.005623
10001110	-44.5	0.005957
10001111	-44.0	0.006310
10010000	-43.5	0.006683
10010001	-43.0	0.007079
10010010	-42.5	0.007499
10010011	-42.0	0.007943
10010100	-41.5	0.008414
10010101	-41.0	0.008913
10010110	-40.5	0.009441
10010111	-40.0	0.010000
10011000	-39.5	0.010593
10011001	-39.0	0.011220
10011010	-38.5	0.011885
10011011	-38.0	0.012589

Bit[7:0]	Description	
	Gain (dB)	Magnification
10011100	-37.5	0.013335
10011101	-37.0	0.014125
10011110	-36.5	0.014962
10011111	-36.0	0.015849
10100000	-35.5	0.016788
10100001	-35.0	0.017783
10100010	-34.5	0.018836
10100011	-34.0	0.019953
10100100	-33.5	0.021135
10100101	-33.0	0.022387
10100110	-32.5	0.023714
10100111	-32.0	0.025119
10101000	-31.5	0.026607
10101001	-31.0	0.028184
10101010	-30.5	0.029854
10101011	-30.0	0.031623
10101100	-29.5	0.033497
10101101	-29.0	0.035481
10101110	-28.5	0.037584
10101111	-28.0	0.039811
10110000	-27.5	0.042170
10110001	-27.0	0.044668
10110010	-26.5	0.047315
10110011	-26.0	0.050119
10110100	-25.5	0.053088
10110101	-25.0	0.056234
10110110	-24.5	0.059566
10110111	-24.0	0.063096
10111000	-23.5	0.066834
10111001	-23.0	0.070795
10111010	-22.5	0.074989
10111011	-22.0	0.079433
10111100	-21.5	0.084140
10111101	-21.0	0.089125
10111110	-20.5	0.094406
10111111	-20.0	0.100000
11000000	-19.5	0.105925
11000001	-19.0	0.112202
11000010	-18.5	0.118850
11000011	-18.0	0.125893

Bit[7:0]	Description	
	Gain (dB)	Magnification
11000100	-17.5	0.133352
11000101	-17.0	0.141254
11000110	-16.5	0.149624
11000111	-16.0	0.158489
11001000	-15.5	0.167880
11001001	-15.0	0.177828
11001010	-14.5	0.188365
11001011	-14.0	0.199526
11001100	-13.5	0.211349
11001101	-13.0	0.223872
11001110	-12.5	0.237137
11001111	-12.0	0.251189
11010000	-11.5	0.266073
11010001	-11.0	0.281838
11010010	-10.5	0.298538
11010011	-10.0	0.316228
11010100	-9.5	0.334965
11010101	-9.0	0.354813
11010110	-8.5	0.375837
11010111	-8.0	0.398107
11011000	-7.5	0.421697
11011001	-7.0	0.446684
11011010	-6.5	0.473151
11011011	-6.0	0.501187
11011100	-5.5	0.530884
11011101	-5.0	0.562341
11011110	-4.5	0.595662
11011111	-4.0	0.630957
11100000	-3.5	0.668344
11100001	-3.0	0.707946
11100010	-2.5	0.749894
11100011	-2.0	0.794328
11100100	-1.5	0.841395
11100101	-1.0	0.891251
11100110	-0.5	0.944061
11100111	0.0	1.000000
11101000	0.5	1.059254
11101001	1.0	1.122018
11101010	1.5	1.188502
11101011	2.0	1.258925

Bit[7:0]	Description	
	Gain (dB)	Magnification
11101100	2.5	1.333521
11101101	3.0	1.412538
11101110	3.5	1.496236
11101111	4.0	1.584893
11110000	4.5	1.678804
11110001	5.0	1.778279
11110010	5.5	1.883649
11110011	6.0	1.995262
11110100	6.5	2.113489
11110101	7.0	2.238721
11110110	7.5	2.371374
11110111	8.0	2.511886
11111000	8.5	2.660725
11111001	9.0	2.818383
11111010	9.5	2.985383
11111011	10.0	3.162278
11111100	10.5	3.349654
11111101	11.0	3.548134
11111110	11.5	3.758374
11111111	12.0	3.981072

5.2. Fade In/Fade Out Time

Table 20-11 Fade In/Fade Out Time

Bit[4:0]	Description
00000	Fade in/fade out effect not used.
00001	20 ms (960 sample)
00010	40 ms (1920 sample)
00011	60 ms (2880 sample)
00100	80 ms (3840 sample)
00101	100 ms (4800 sample)
00110	200 ms (9600 sample)
00111	300 ms (14400 sample)
01000	400 ms (19200 sample)
01001	500 ms (24000 sample)
01010	600 ms (28800 sample)
01011	700 ms (33600 sample)
01100	800 ms (38400 sample)
01101	900 ms (43200 sample)
01110	1000 ms (48000 sample)
01111	1100 ms (52800 sample)
10000	1200 ms (57600 sample)
10001	1300 ms (62400 sample)
10010	1400 ms (67200 sample)
10011	1500 ms (72000 sample)
10100	1600 ms (76800 sample)
10101	1700 ms (81600 sample)
10110	1800 ms (86400 sample)
10111	1900 ms (91200 sample)
11000	2000 ms (96000 sample)
11001	2000 ms (96000 sample)
11010	2000 ms (96000 sample)
11011	2000 ms (96000 sample)
11100	2000 ms (96000 sample)
11101	2000 ms (96000 sample)
11110	2000 ms (96000 sample)
11111	2000 ms (96000 sample)

5.3. Digital Filter

5 type of digital filters are prepared and automatically selected and applied to change sampling rate. See 3.10.2 of the description of FIR digital filter.

No	Sampling frequency	Cutoff frequency	Remark
1	44 kHz	17 kHz	Figure 20-3
2	24 kHz	11 kHz	Figure 20-4
3	12 kHz	5 kHz	Figure 20-5
4	8 kHz	3 kHz	Figure 20-6
5	4 kHz	1,2 kHz	Figure 20-7

Figure 20-3 Filter No 1

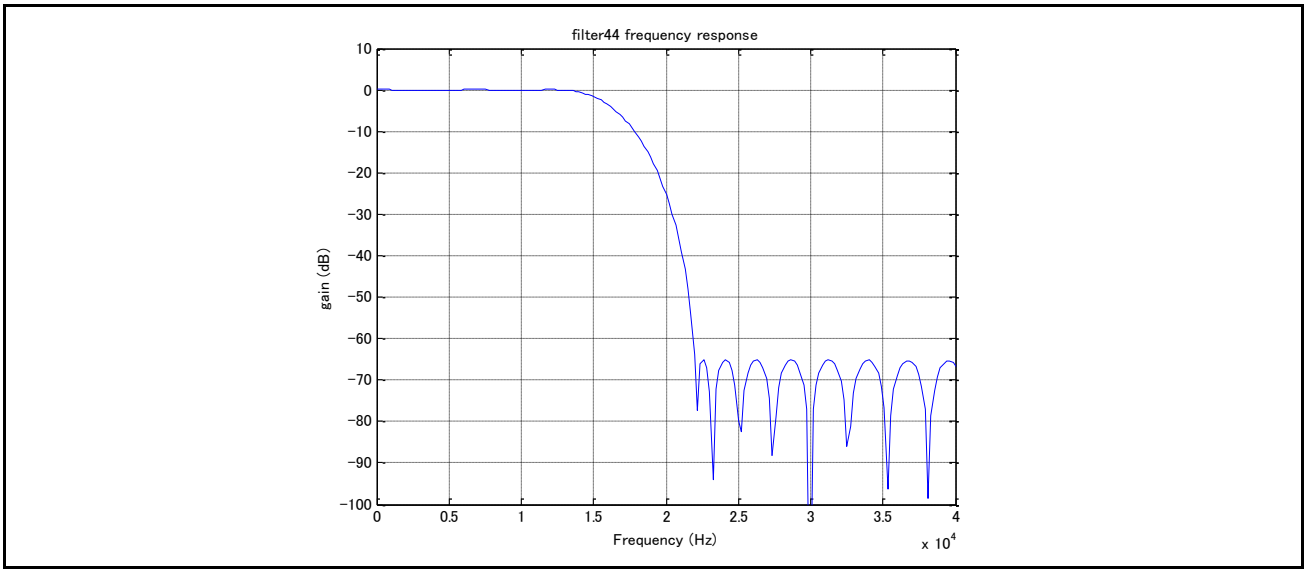


Figure 20-4 Filter No 2

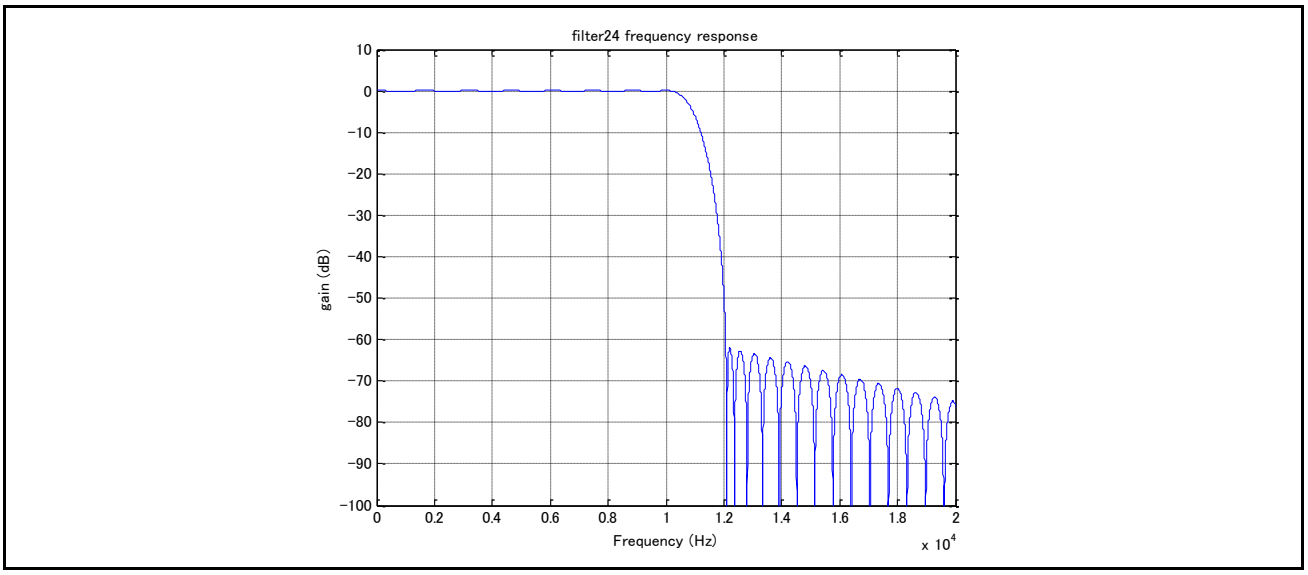


Figure 20-5 Filter No 3

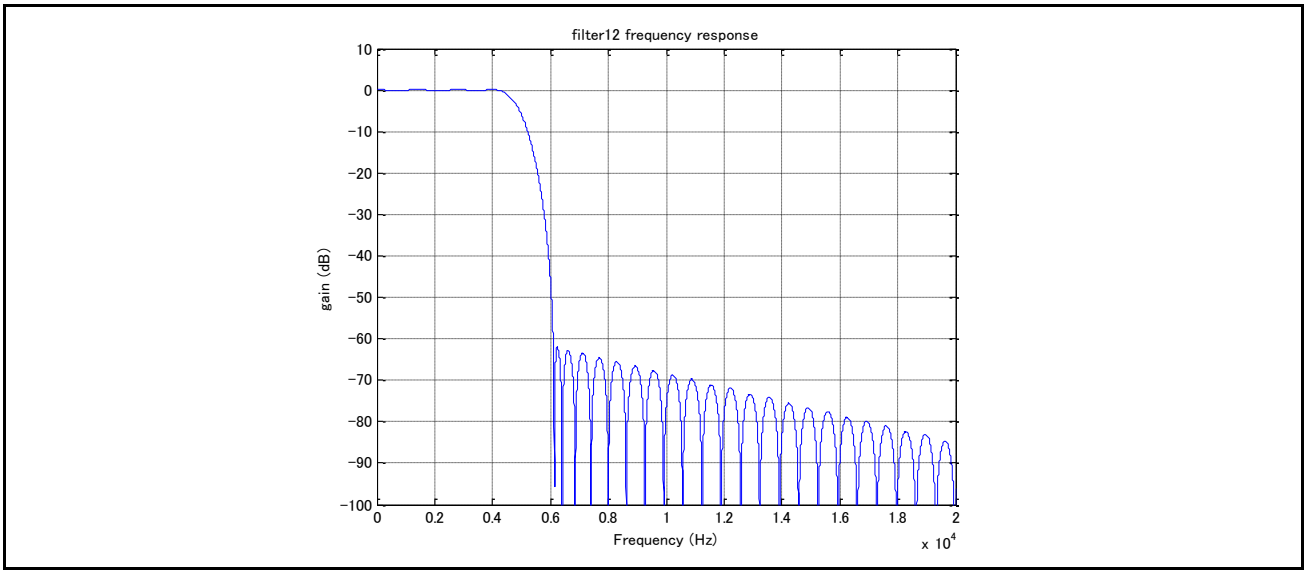


Figure 20-6 Filter No 4

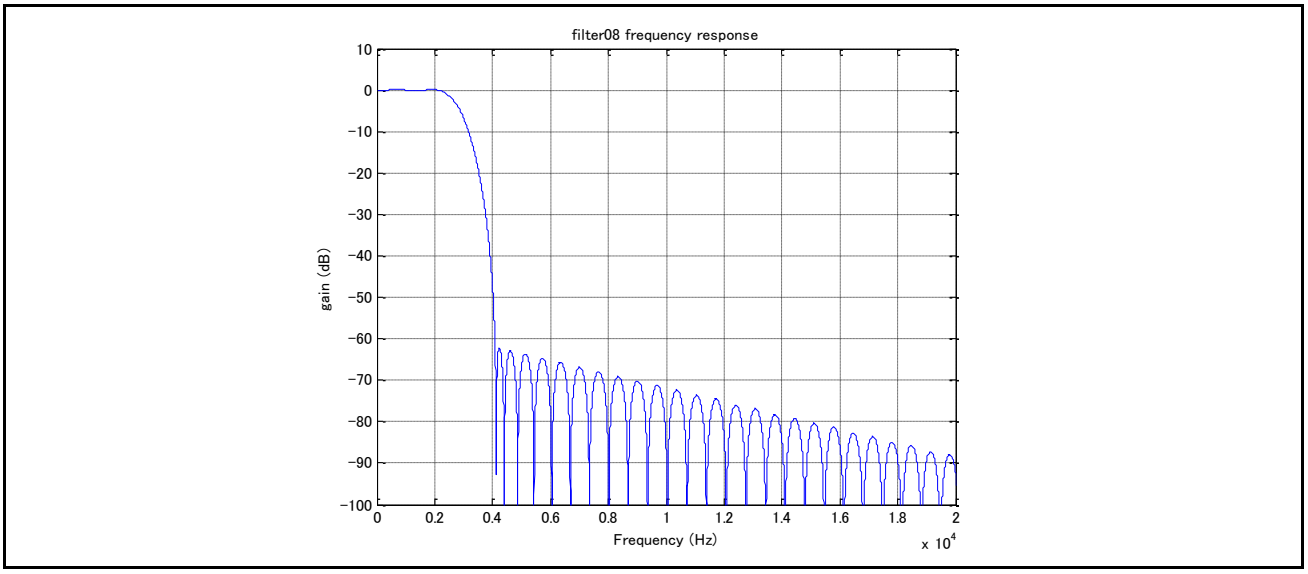
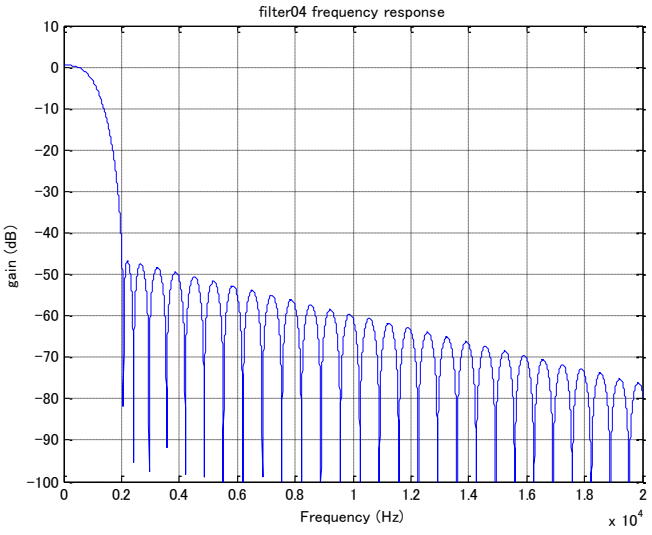


Figure 20-7 Filter No 5



CHAPTER 21: Ethernet MAC

This chapter explains the function and operation of the Ethernet MAC module.

1. Overview
2. Configuration and Block Diagram
3. Operation of the Ethernet MAC
4. Registers
5. Functional Limitations

CODE: ETHERNET-S6J3300-E1

1. Overview

This section gives a brief overview of **Ethernet MAC**.

Features of the Ethernet MAC

- Supports IEEE Std 802.1BA – Audio Video Bridging Systems
- Supports IEEE Std 802.1Qav – Forwarding and Queuing Enhancements for Time-Sensitive Streams
- Supports IEEE Std 802.1AS – Timing and Synchronization for Time-Sensitive Application in Bridged LAN's
- Supports IEEE Std 1588 – Precision Time Protocol
- 4 transmit and receive priority queues
- 16 screening type 1 registers to support up to 16 received priority queues
- 16 screening type 2 registers which combine matching of VLAN priority, EtherType, Compare A, Compare B, and Compare C comparisons
- full-duplex, 100 Mbit/s operation
- AMBA AXI Master Interface (32-bit address width, 64-bit data width)

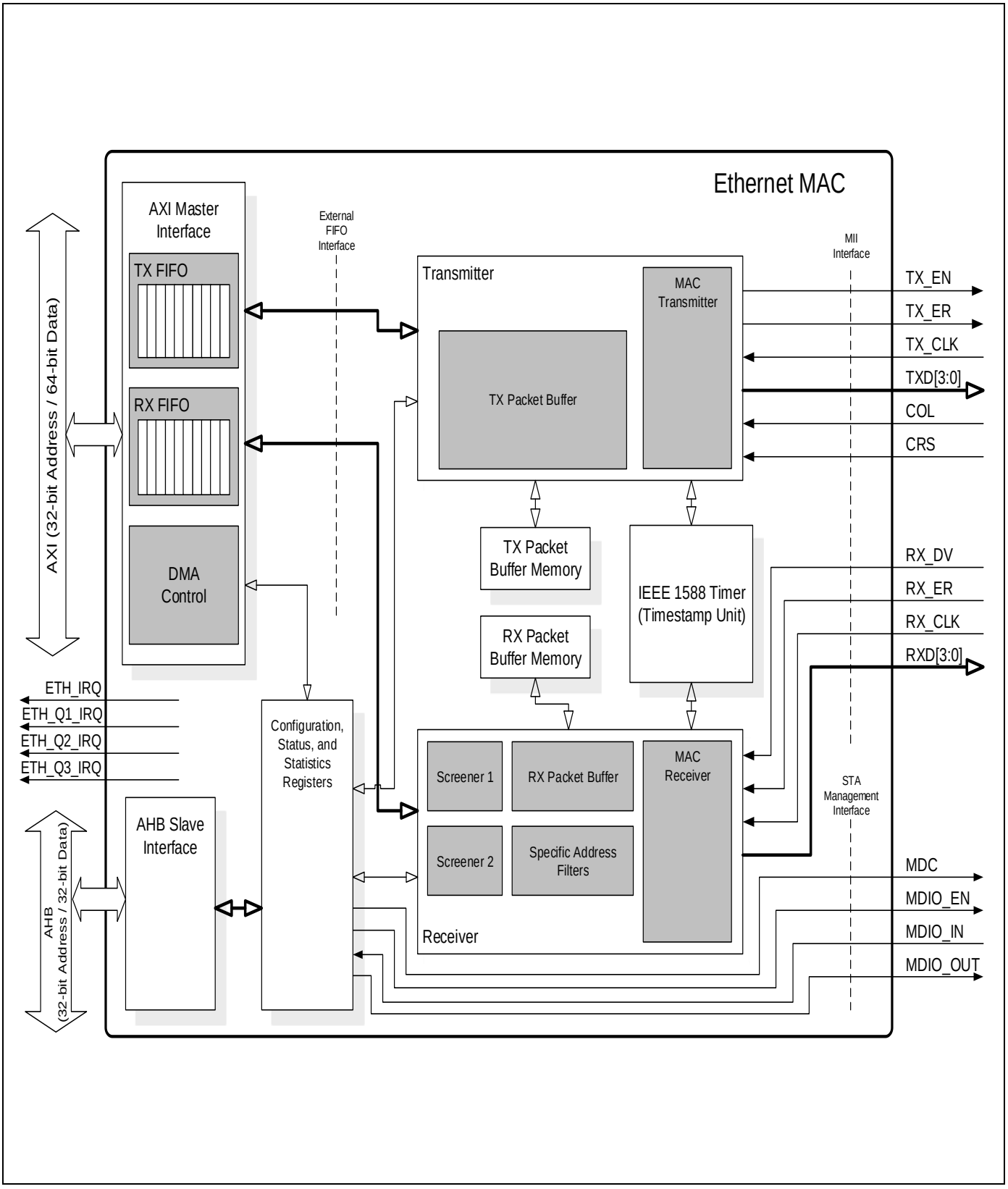
Acronyms and Abbreviations

Acronym	Description
BD	Buffer Descriptor
CBS	Credit Based Shaping
CFI	Canonical Format Indicator
EEE	Energy Efficient Ethernet (IEEE Std 802.3az)
EOF	End Of Frame
FCS	Frame Check Sequence
IP	Internet Protocol
IPG	Interpacket Gap
LAN	Local Area Network (IEEE Std 802)
LLDP	Link Layer Discovery Protocol (IEEE Std 802.1AB)
LPI	Low-Power Idle
MAC	Media Access Control (IEEE Std 802)
MDC	Management Data Clock
MII	Media Independent Interface
NSP	Non-Standard Preamble
PCS	Physical Control Sub-Layer
PFC	Priority based Flow Control (IEEE Std 802.1Qbb)
PHY	Physical sublayer
PPPoE	Point-to-Point Protocol over Ethernet
PTP	Precision Time Protocol (IEEE Std 1588)
RFC 791	DARPA Internet Program Protocol Specification
SerDes	Serialiser/Deserialiser
SFD	Start of Frame Delimiter
SGMII	Serial Gigabit Media Independent Interface
SNAP	Subnetwork Access Protocol
SOF	Start Of Frame
TCP	Transfer Control Protocol
TS	Timestamp
TSU	Timestamp Unit
GeUDP	User Datagram Protocol
VLAN	Virtual LAN (IEEE Std 802.1Q)
XFI	10-Gigabit/s Electrical Interface Specification

2. Configuration and Block Diagram

This section shows a block diagram of the Ethernet MAC.

Figure 21-1: Ethernet MAC Block Diagram



3. Operation of the Ethernet MAC

This section describes the operation of the Ethernet MAC.

3.1. Direct Memory Access Interface

The Ethernet MAC DMA is attached to the Ethernet MAC's External FIFO Interface to provide a scatter gather type capability for packet data storage.

The DMA is configured in packet buffering mode, where Dual-Port memories are used to buffer multiple frames. This allows using one of the programmable operation modes:

- Full Store and Forward
- Partial Store and Forward

In Full Store and Forward mode a packet will automatically be replayed directly from the packet buffer memory rather than having to re-fetch from system memory through AXI, when a collision occurs during transmission. In Full Store and Forward mode received erroneous packets are automatically dropped before they are send to system memory, thus reducing AXI activity.

Further key features are:

- Transmit TCP/IP checksum offload.
- Priority queuing.
- Manual RX packet flushes.
- RX packet flush when there is lack of resource.
- Burst padding at end of packet and end of buffer to maximize AXI efficiency.
- 64-bit addressing for Data Buffer start address within Buffer Descriptor entry
- TX/RX timestamp capture to Buffer Descriptor entry.

3.1.1. Using the AXI Interface

The Ethernet MAC's AXI master interface provides separate data and address connections for Reads and Writes, which allow simultaneous, bidirectional data transfers. The Ethernet MAC supports multiple outstanding transactions on both the AXI Read and Write address channels, up to the limit of 16. This means the issuing of Read and Write requests from the Ethernet MAC DMA (on AXI AR and AW channels) are decoupled from the AXI slave responses (on AXI R and B channels). The issuing of outstanding transactions is allowed to span multiple frames, maintaining high data transfer rates.

The Ethernet MAC will buffer the descriptor accesses locally to avoid the underlying DMA from pausing while descriptor transactions are completed by the High Performance Matrix.

- TX and RX descriptor reads are issued up-front and stored in a local buffer to feed the underlying DMA when required. This optimizes performance and avoids the need for the underlying DMA to pause while new descriptor fetches are sent to the system bus.
- TX and RX descriptor writes issued by the underlying DMA are buffered locally to avoid holding up the underlying DMA if and when the system delays the completion of descriptor writes. Note a descriptor write transaction is not considered complete until the write response (BRESP) associated with that transaction has arrived.

The maximum burst lengths the Ethernet MAC will use are programmable. Single accesses and bursts with up to 16 beats can be selected. With 64-bit data path and a burst length setting of 16, 128 Bytes transfers can be made with a single request. The burst length is controlled via the DMA Configuration register (ETHERNETn_dma_config[4:0]).

3.1.2. Partial Store and Forward Using Packet Buffer DMA

The Ethernet MAC can be programmed into a low latency mode, known as partial store and forward. This allows for a reduced latency as the full packet is not buffered before forwarding. This option is only available when not using multi buffer frames. This feature is enabled via the TX and RX Partial Store and

Forward registers (ETHERNETn_pbuf_txcutthru, ETHERNETn_pbuf_rxcutthru). When transmit partial store and forward mode is activated, the transmitter will only begin to forward the packet to the MAC when there is enough packet data stored in the TX Packet Buffer Memory. Likewise, when receive partial store and forward mode is activated, the receiver will only begin to forward the packet to the system memory when enough packet data is stored in the RX Packet Buffer Memory. The amount of packet data required to activate the forwarding process is programmable via watermark registers which are located at the same address as the partial store and forward enable bits. Note that the minimum operational value for the TX partial store and forward watermark is 20. There is no operational limit for the RX partial store and forward watermark. Enabling partial store and forward is a useful means to reduce latency, but there are performance implications. In essence, the packet buffer DMA will start behaving in a similar way to the internal FIFO DMA mode when partial store and forward is enabled. Information regarding this behavior is described in section Receive DMA Buffers. When priority queuing is enabled, each TX Packet Buffer Memory region allocated to a queue must be greater than the size of the maximum frame length to be transmitted.

3.1.3. Ethernet MAC DMA Transactions

The Ethernet MAC DMA uses separate transmit and receive lists of buffer descriptors, with each descriptor describing a buffer area in system memory. This allows Ethernet packets to be broken up and scattered around the system memory.

The Ethernet MAC DMA controller performs six types of operation on the AMBA bus. In order of priority these are:

1. receive buffer manager write/read
2. transmit buffer manager write/read
3. receive data DMA write
4. transmit data DMA read

All read operations are routed to the AXI Master Interface read channel and all write operations to the AXI Master Interface write channel. Both read and write channel operate simultaneously. Arbitration logic is used when multiple requests are active on the same channel (e.g. when the TX DMA requests a transmit data read at the same time the RX DMA requests a receive descriptor read). In these cases, the RX DMA is granted the bus before the TX DMA. However the vast majority of requests are either receive data writes or transmit data reads both of which can operate in parallel.

Transfer size is set to 64-bit words by default in the Network Configuration register (ETHERNETn_network_configuration) and burst length can be programmed in the range from single access up to 16 accesses per burst using the DMA Configuration register (ETHERNETn_dma_config).

3.1.4. Receive DMA Buffers

Received frames, optionally including FCS, are written to receive buffers located in system memory. The receive buffer depth (rx_buf_size[7:0]) is programmable in the range of 64 bytes to 16 Kbytes in the DMA Configuration register, with the default being 15.360 bytes.

The start location for each receive buffer is stored in system memory in a list of receive buffer descriptors at an address location pointed to by the receive buffer queue pointer. The base address of the receive buffer queue pointer must be configured by software using the receive buffer queue base address registers (ETHERNETn_receive_q_ptr, ETHERNETn_receive_q1_ptr to ETHERNETn_receive_q3_ptr).

The number of words in each buffer descriptor (BD) is dependent on the operating mode. Each BD word is defined as 32 bits. The first two words (Word 0 and Word 1) are used for all BD modes.

In Extended Buffer Descriptor mode (ETHERNETn_dma_config[28]:rx_bd_extended_mode_en = 1), two BD words (Word 2 and Word 3) are added for timestamp capture if timestamp capture mode is enabled (ETHERNETn_rx_bd_control[5:4]:rx_bd_ts_mode > 0_h). There are therefore either two or four BD words in each BD entry depending on the operating mode, and every BD entry will have the same number of words. To summarize:

- Every descriptor will be 64 bits wide when descriptor time capture mode is disabled.

- Every descriptor will be 128 bits wide when descriptor time capture mode is enabled.

The following description details the functionality of Word 0 and Word 1. Each list entry consists of the same first two words. The first (i.e. Word 0) contains the start location of the receive buffer and the second (i.e. Word 1) the receive status. If the length of a receive frame exceeds the Ethernet MAC DMA buffer length, the status word for the used buffer is written with zeroes except for the "start of frame" bit, which is always set for the first buffer in a frame. Bit zero of the address field is written to "1" to show the buffer has been used. The receive buffer manager then reads the location of the next receive buffer and fills that with the next part of the received frame data. Receive buffers are filled until the frame is complete and the final buffer descriptor entry table for details of the receive buffer descriptor list.

When using receive descriptor timestamp capture, bit 2 of Word 0 is used to indicate a valid timestamp has been captured in the BD. The use of bit 2 for this purpose also necessitates the data buffer being located on a 64-bit address boundary.

Each receive buffer start location is a word address. The start of the first buffer in a frame can be offset by up to seven bytes depending on the value written to bits 15 and 14 of the Network Configuration register (receive_buffer_offset[1:0]) and bit 2 of Word 0.

Table 21-1: Receive Buffer Byte Offset Configuration

Receive Buffer Offset Configuration Bit 2 of Word 0	Receive_Buffer_Offset[1]	Receive_Buffer_Offset[0]	Number of Bytes Offset
0	0	0	0
0	0	1	1
0	1	0	2
0	1	1	3
1	0	0	4
1	0	1	5
1	1	0	6
1	1	1	7

If the start location of the buffer is offset the available length of the first buffer is reduced by the corresponding number of bytes.

Table 21-2: Receive Buffer Descriptor Entry

Bits	Function
Word 0	
31:3	Address [31:3] of beginning of buffer
2	Address [2] of beginning of buffer or In Extended Buffer Descriptor Mode, indicates a valid timestamp in the BD entry.
1	Wrap - marks last descriptor in receive buffer descriptor list.
0	Ownership - needs to be "0" for the **Ethernet MAC** to write data to the receive buffer. The **Ethernet MAC** sets this to "1" once it has successfully written a frame to memory. Software has to clear this bit before the buffer can be used again.
Word 1	
31	Global all ones broadcast address detected.
30	Multicast hash match.
29	Unicast hash match.
28	External address match.
27	Unused.
26:25	Specific Address register match. Encoded as follows: 00 - Specific Address 1 register match (lowest priority) 01 - Specific Address 2 register match 10 - Specific Address 3 register match 11 - Specific Address 4 register match (highest priority) If more than one specific address is matched only one of them is indicated with priority 4 down to 1.

Bits	Function
24	This bit has a different meaning depending on whether RX checksum offloading is enabled (ETHERNETn_network_configuration[24] = 1). **With RX checksum offloading disabled:** Type ID register match found, bit 22 and bit 23 indicate which Type ID register causes the match. **With RX checksum offloading enabled:** 0 - The frame was not SNAP encoded and/or had a VLAN tag with the CFI bit set. 1 - The frame was SNAP encoded and had either no VLAN tag or a VLAN tag with the CFI bit not set.
23:22	This bit has a different meaning depending on whether RX checksum offloading is enabled. **With RX checksum offloading disabled:** Type ID register match. Encoded as follows: 00 - Type ID Match 1 register 01 - Type ID Match 2 register 10 - Type ID Match 3 register 11 - Type ID Match 4 register If more than one Type ID is matched only one of them is indicated with priority 4 down to 1. **With RX checksum offloading enabled:** 00 - Neither the IP header checksum nor the TCP/UDP checksum was checked. 01 - The IP header checksum was checked and was correct. Neither the TCP nor UCP checksum was checked. 10 - Both the IP header and TCP checksum were checked and were correct. 11 - Both the IP header and UDP checksum were checked and were correct.
21	VLAN tag detected - Type ID of 8100_h. For packets incorporating the stacked VLAN processing feature, this bit will be set if the second VLAN tag has a Type ID of 8100_h.
20	Priority tag detected - Type ID of 8100_h and null VLAN identifier. For packets incorporating the stacked VLAN processing feature, this bit will be set if the second VLAN tag has a Type ID of 8100_h and a null VLAN identifier.
19:17	VLAN priority - only valid if bit 21 is set. 000 - Priority 0 (lowest) BK Background 001 - Priority 1 BE Best Effort 010 - Priority 2 EE Excellent Effort 011 - Priority 3 CA Critical Applications 100 - Priority 4 VI Video, <100 ms latency and jitter 101 - Priority 5 VO Voice, <10 ms latency and jitter 110 - Priority 6 IC Internetwork Control 111 - Priority 7 (highest) NC Network Control
16	Canonical Format Indicator (CFI) bit - only valid if bit 21 is set.
15	End of Frame - when set the buffer contains the end of a frame. If End of Frame is not set, then the only valid status bit is Start of Frame (bit 14).
14	Start of Frame - when set the buffer contains the start of a frame. If both bits 15 and 14 are set, the buffer contains a whole frame.
13	This bit has a different meaning depending on whether jumbo frames and ignore FCS mode are enabled (ETHERNETn_network_configuration[3], ETHERNETn_network_configuration[26]). If neither mode is enabled this bit will be "0". **With jumbo frame mode enabled:** Additional bit for length of frame (bit 13), that is concatenated with bits [12:0] **With ignore FCS mode enabled and jumbo frames disabled:** This indicates per frame FCS status as follows: 0 - Frame had good FCS 1 - Frame had bad FCS, but was copied to memory as ignore FCS is enabled.
12:0	These bits represent the length of the received frame which may or may not include FCS depending on whether FCS discard mode is enabled (ETHERNETn_network_configuration[17] = 1). **With FCS discard mode disabled:** Least significant 12 bits for length of frame including FCS. If jumbo frames are enabled, these 12 bits are concatenated with bit 13 of the descriptor. **With FCS discard mode enabled:** Least significant 12 bits for length of frame excluding FCS. If jumbo frames are enabled, these 12 bits are concatenated with bit 13 of the descriptor.

When Descriptor Timestamp Capture mode is enabled, the following table identifies the added descriptor words.

Bit	Function
Word 2	
31:30	Timestamp seconds [1:0] (see Note1)
29:0	Timestamp nanoseconds [29:0] (see Note1)
Word 3	
31:4	Unused
3:0	Timestamp seconds [5:2] (see Note1)
	Note 1: The timestamp mode is controlled using the RX BD Control register (ETHERNETn_rx_bd_control).

To receive frames, the receive buffer descriptors must be initialized by writing an appropriate address to bits [31:2] (or [31:3] for timestamp capture mode) in the first word of each list entry. Bit 0 must be written with "0". Bit 1 is the wrap bit and indicates the last entry in the buffer descriptor list.

The start location of the receive buffer descriptor list must be written with the receive buffer queue base address before reception is enabled (ETHERNETn_network_control[2] = 1). Once reception is enabled, any writes to the RX Buffer Queue Base Address register are ignored. When read, it will return the current pointer position in the descriptor list, though this is only valid and stable when receive is disabled.

If the filter block indicates that a frame should be copied to memory, the receive data DMA operation starts writing data into the receive buffer. If an error occurs, the buffer is recovered.

A counter in the Ethernet MAC represents the receive buffer queue pointer and it is not visible through its register interface. The receive buffer queue pointer increments by two words after each buffer has been used. It re-initializes to the receive buffer queue base address if any descriptor has its wrap bit set.

As receive buffers are used, the receive buffer manager sets bit 0 of the first word of the descriptor to "1" indication the buffer has been used.

Software should search through the "Used" bits in the buffer descriptors to find out how many frames have been received, checking the Start of Frame and End of Frame bits.

When the Ethernet MAC DMA is configured in the partial store and forward mode, received frames are written out to the system memory as soon as enough frame data exists in the RX Packet Buffer Memory. This may mean several full buffers are used before some error conditions can be detected. If a receive error is detected the receive buffer currently being written will be recovered. Previous buffers will not be recovered. As an example, when receiving frames with CRC errors or excessive length, it is possible that a frame fragment might be stored in a sequence of receive buffers. Software can detect this by looking for Start of Frame bit set in a buffer following a buffer with no End of Frame bit set.

For a properly working 10/100 MBit/s Ethernet system there should be no excessive length frames or frames greater than 128 bytes with CRC errors. Collision fragments will be less than 128 bytes long, therefore it will be a rare occurrence to find a frame fragment in a receive buffer, when using the value of 128 bytes for the receive buffers size (ETHERNETn_dma_config:rx_buf_size[7:0]).

When in full store and forward mode only good received frames are written out of the Ethernet MAC DMA, so no fragments will exist in the buffers due to MAC Receiver errors. There is still the possibility of fragments due to Ethernet MAC DMA errors, for example used bit read on the second buffer of a multi-buffer frame.

If bit 0 of the receive buffer descriptor is already set when the receive buffer manager reads the location of the receive buffer, then the buffer has been already used and cannot be used again until software has processed the frame and cleared bit 0. In this case, the "buffer not available" bit in the Receive Status register is set and an interrupt triggered. The Receive Resource Errors statistics register is also incremented.

When the Ethernet MAC DMA is configured in the full store and forward mode, it can be selected whether received frames should be automatically discarded when no AXI buffer resource is available. This feature is selected via bit 24 of the DMA Configuration register (by default, the received frames are not automatically discarded). If this feature is off, then received packets will remain to be stored in the RX Packet Buffer Memory until system memory resource next becomes available. This may lead to an eventual packet buffer overflow if packets continue to be received when bit 0 (“Used” bit) of the receive buffer descriptor remains set. Note that after a “Used” bit has been read, the receive buffer manager will re-read the location of the receive buffer descriptor every time a new packet is received. When the

Ethernet MAC DMA is not configured in the full store and forward mode and a “Used” bit is read, the frame currently being received will be automatically discarded.

When the Ethernet MAC DMA is configured in the full store and forward mode, a receive over run condition occurs when the RX Packet Buffer Memory is full, or because an AXI error occurred. In all other modes, a receive over run condition occurs when either the AXI bus was not granted quickly enough, or because an AXI error occurred, or because a new frame has been detected by the receive block when the status update or write back for the previous frame has not yet finished. For a receive over run condition, the receive over run interrupt is asserted and the buffer currently being written is recovered. The next frame that is received whose address is recognized reuses the buffer.

When the Ethernet MAC DMA is configured for packet buffer mode, the upper bits of the data buffer address stored in bits [31:2] in the first word of each list entry can be dynamically altered in real-time without physically changing the system memory holding the list entry. This feature is useful if the destination has to be selected based on CPU usage or other flow control hardware. It is achieved using a MUX structure whereby it can be defined whether the upper 4 bits of the 32-bit data-buffer AXI address should come from the descriptor list entry or from a programmable register. Refer to the Receive DMA Data Buffer Address Mask register for further details. Note that any changes to this register will be ignored while the Ethernet MAC DMA is currently processing a receive packet. It will only affect the next full packet to be written to system memory.

3.1.5. Transmit DMA Buffers

Frames to transmit are stored in one or more transmit buffers. Transmit frames can be between 1 and 16.384 bytes long, so it is possible to transmit frames longer than the maximum length specified in the IEEE Std 802.3 standard. It should be noted that zero length buffers are allowed and that the maximum number of buffers permitted for each transmit frame is 128.

The start location for each transmit buffer is stored in system memory in a list of transmit buffer descriptors at a location pointed to by the transmit buffer queue pointer. The base address for this queue pointer must be configured by software using the TX Buffer Queue Base Address registers (ETHERNETn_transmit_q_ptr, ETHERNETn_transmit_q1_ptr to ETHERNETn_transmit_q3_ptr).

The number of words in each buffer descriptor (BD) is dependent on the operating mode. Each BD word is defined as 32 bits. The first two words (Word 0 and Word 1) are used for all BD modes.

In Extended Buffer Descriptor mode (ETHERNETn_dma_config[29]:tx_bd_extended_mode_en = 1), two BD words (Word 2 and Word 3) are added for timestamp capture mode if timestamp capture mode is enabled (ETHERNETn_tx_bd_control[5:4]:tx_bd_ts_mode > 0h). There are therefore either two or four BD words in each BD entry depending on the operating mode, and every BD entry will have the same number of words. To summarize:

- Every descriptor will be 64 bits wide when descriptor time capture mode is disabled.
- Every descriptor will be 128 bits wide when descriptor time capture mode is enabled.

The following description details the functionality of Word 0 and Word 1. Each list entry consists of the same first two words. The first (Word 0) is the byte address of the transmit buffer and the second (Word 1) contains the transmit control and status. For the packet buffer DMA, the start location for each AXI buffer is a byte address, the bottom bits of the address being used to offset the start of the data from the data-word boundary (i.e. bits 2, 1, and 0 are used to offset the address for 64-bit data paths).

Frames can be transmitted with or without automatic CRC generation. If CRC is automatically generated, pad will also be automatically generated to take frames to a minimum length of 64 bytes. When CRC is not automatically generated (as defined in Word 1 of the transmit buffer descriptor), the frame is assumed to be at least 64 bytes long and pad is not generated.

Table 21-3: Transmit Buffer Descriptor Entry

Bit	Function
Word 0	
31:0	Byte address of buffer
Word 1	
31	Used - must be "0" for the **Ethernet MAC** to read data to the transmit buffer. The **Ethernet MAC** sets this to "1" for the first buffer of a frame once it has been successfully transmitted. Software must clear this bit before the buffer can be used again.
30	Wrap - marks last descriptor in transmit buffer descriptor list. This can be set for any buffer within the frame.
29	Retry limit exceeded, transmit error detected
28	Unused
27	Transmit frame corruption due to AXI error - set if an error occurs whilst midway through reading through reading transmit frame from the AXI, including RRESP/BRESP errors and buffers exhausted mid frame (if the buffers run out during transmission of a frame then transmission stops, FCS shall be bad and `TX_ER` asserted). Also set if single frame is too large for the transmit packet buffer memory size.
26	Transmit error detected.
25:24	Reserved
23	For Extended Buffer Descriptor Mode this bit indicates a timestamp has been captured in the BD. Otherwise unused.

Bit	Function
22:20	Transmit IP/TCP/UDP checksum generation offload errors: 000 - No error 001 - The Packet was identified as a VLAN type, but the header was not fully complete, or had an error in it. 010 - The Packet was identified as a SNAP type, but the header was not fully complete, or had an error in it. 011 - The Packet was not of an IP type, or the IP packet was invalidly short, or the IP was not of type IPv4/IPv6 100 - The Packet was not identified as VLAN, SNAP or IP. 101 - Non supported packet fragmentation occurred. For IPv4 packets, the IP checksum was generated and inserted. 110 - Packet type detected was not TCP or UDP, TCP/UDP checksum was therefore not generated. For IPv4 packets, the IP checksum was generated and inserted. 111 - A premature end of packet was detected and the TCP/UDP checksum could not be generated.
19:17	Reserved
16	No CRC to be appended by MAC. When set this implies that the data in the buffers already contains a valid CRC and hence no CRC or padding is to be appended to the current frame by the MAC. This control bit must be set for the first buffer in a frame and will be ignored for the subsequent buffers of a frame. Note that this bit must be “0” when using the transmit IP/TCP/UDP checksum generation offload, otherwise checksum generation and substitution will not occur. Note this bit must also be “0” when TX Partial Store and Forward mode is active.
15	Last buffer, when “1” this bit will indicate the last buffer in the current frame has been reached.
14	Reserved
13:0	Length of buffer.

When Descriptor Timestamp Capture mode is enabled, the following table identifies the added descriptor words.

Bit	Function
Word 2	
31:30	Timestamp seconds [1:0] (see Note1)
29:0	Timestamp nanoseconds [29:0] (see Note1)
Word 3	
31:4	Unused
3:0	Timestamp seconds [5:2] (see Note1)
	Note 1: The timestamp mode is controlled using the TX BD Control register (ETHERNETn_tx_bd_control).

To transmit frames, the buffer descriptors must be initialized by writing an appropriate byte address to bits [31:0] in the first word (Word 0) of each descriptor list entry.

The second word (Word 1) of the transmit buffer descriptor is initialized with control information that indicates the length of the frame, whether or not the MAC is to append CRC and whether the buffer is the last buffer in the frame.

After transmission the status bits are written back to the second word of the first buffer along with the “Used” bit. Bit 31 is the “Used” bit which must be “0” when the control word is read if transmission is to take place. It is written to “1” once the frame has been transmitted. Bits [29:20] indicate various transmit error conditions. Bit 23 indicates a valid timestamp has been capture in the BD. Bit 30 is the “Wrap” bit which can be set for any buffer within a frame. If no wrap bit is encountered the queue pointer continues to increment.

The TX Buffer Queue Base Address register can only be updated whilst transmission is disabled or halted; otherwise any attempted write will be ignored. When transmission is halted the transmit buffer queue pointer will maintain its value. Therefore when transmission is restarted the next descriptor read from the queue will be from immediately after the last successfully transmitted frame. Whilst transmit is disabled (ETHERNETn_network_control[3] = 0), the transmit buffer queue pointer resets to point to the

address indicated by the TX Buffer Queue Base Address register. Note that disabling receive does not have the same effect on the receive buffer queue pointer.

Once the transmit queue is initialized, transmit is activated by writing to the transmit start bit (ETHERNETn_network_control[9]). Transmit is halted when a buffer descriptor with its “Used” bit set is read, a transmit error occurs, or by writing to the transmit halt bit of the Network Control register (ETHERNETn_network_control[10]). Transmission is suspended if a pause frame is received while the pause enable bit is set in the Network Configuration register (ETHERNETn_network_configuration[13]). Rewriting the start bit while transmission is active is allowed. This is implemented with a transmit_go variable which is readable in the Transmit Status register at bit location 3. The transmit_go variable is reset when:

1. Transmit is disabled
2. A buffer descriptor with its ownership bit set is read.
3. Bit 10, tx_halt_clk, of the Network Control register is written.
4. There is a transmit error such as too many retries or a transmit under run.

To set transmit_go write to bit 9, tx_start_clk, of the Network Control register. Transmit halt does not take effect until any ongoing transmit finishes.

If the Ethernet MAC DMA is configured for packet buffer partial store and forward mode and a collision occurs during transmission of a multi-buffer frame, transmission will automatically restart from the first buffer of the frame. For packet buffer mode, the entire contents of the frame are read into the TX Packet Buffer Memory, so the retry attempt will be replayed directly from the TX Packet Buffer Memory rather than having to re-fetch through AXI Master Interface.

If a “Used” bit is read mid way through transmission of a multi buffer frame this is treated as a transmit error. Transmission stops, `TX_ER` is asserted and the FCS will be bad.

If transmission stops due to a transmit error or a “Used” bit being read, transmission will restart from the first buffer descriptor of the frame being transmitted when the transmit start bit in the Network Control register (ETHERNETn_network_control[9]) is rewritten.

3.1.6. DMA Bursting

When performing data transfers, the burst length used can be programmed using bits [4:0] of the DMA Configuration register. Either single accesses (burst length = 1) of incrementing bursts of up to 16 can be used as appropriate.

When there is sufficient space and enough data to be transferred, the burst of programmed length will be used. If there is not enough data or space available, for example when at the end of a packet or buffer, burst lengths of less than the programmed burst length value will be issued. Single accesses will be used when a 4 Kbyte boundary will be crossed by the burst in order not to violate the AXI specification.

When the Ethernet MAC DMA is configured for packet buffer mode, an option to force the Ethernet MAC DMA to pad the remaining bursts at the end of a buffer or EOP to the programmed burst length value is available via bits 26 and 25 of the DMA Configuration register. Bit 26 will control the TX and bit 25 the RX. For RX, the data to burst is padded with "0"s up to the burst boundary defined by burst length. For TX, the extra data that is read is ignored by the Ethernet MAC DMA. This feature has been included for performance reasons when AHB/AXI slaves that are being accessed by the Ethernet MAC perform better when accessed using fixed length bursts. Note enabling this feature will not break the AHB 1 Kbyte or the AXI 4 Kbyte boundary rule.

The Ethernet MAC DMA will not terminate bursts premature if receive/transmit operation is disabled by writing to Network Control register bit 2/3.

3.1.7. DMA Packet Buffer

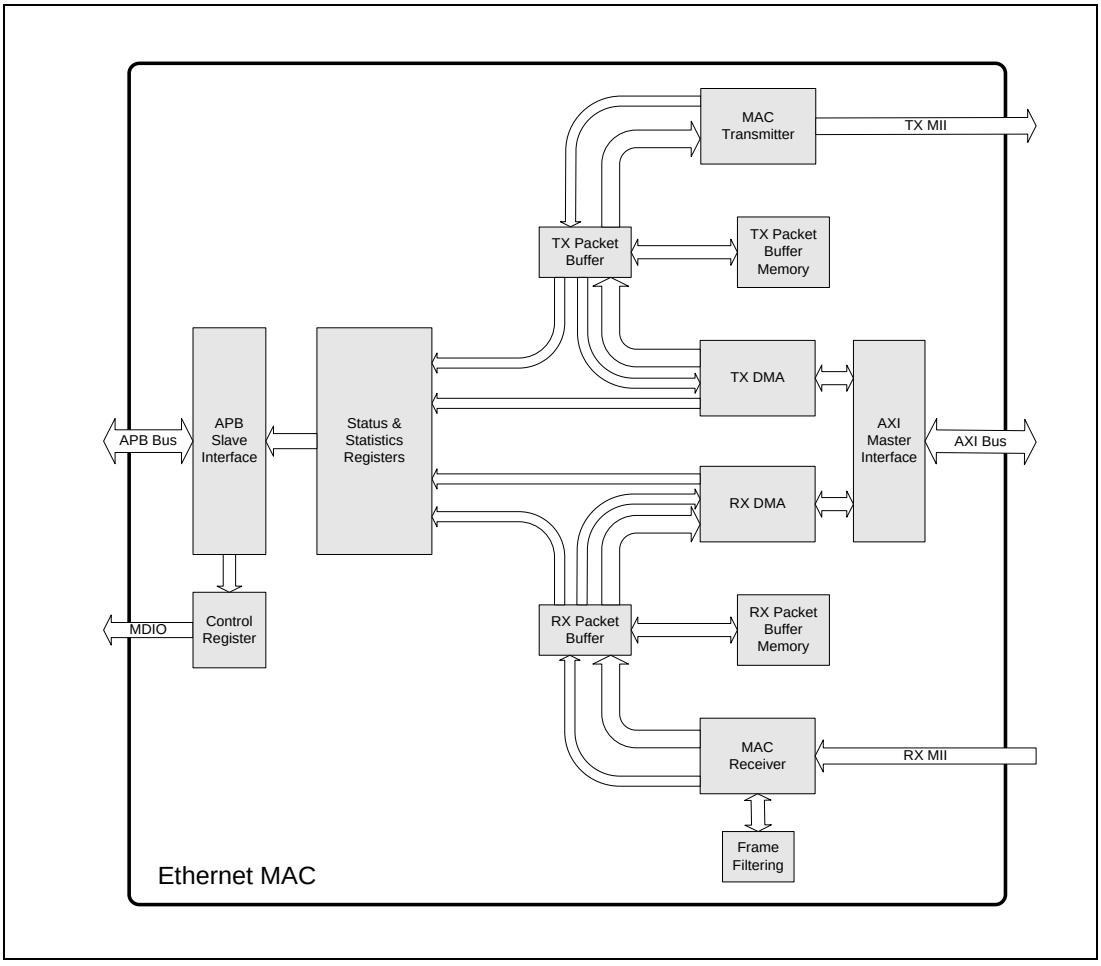
The packet buffer DMA mode allows multiple packets to be buffered in both transmit and receive directions and allows the Ethernet MAC DMA to withstand variable levels of access latencies on the AXI fabric. This mode offers the most efficient use of the AXI bandwidth.

As described earlier, the Ethernet MAC DMA can be programmed into a low latency mode known as partial store and forward. When the Ethernet MAC DMA is programmed in full store and forward mode full packets are buffered providing the opportunity to:

- Discard packets that are received with errors before they are partially written out of the **Ethernet MAC** DMA thus saving AXI bandwidth and driver processing overhead.
- Retry collided transmit frames from the buffer, thus saving AXI bus bandwidth.
- Implement transmit IP/TCP/UDP checksum generation offload.

The following figure illustrates the structure of the Ethernet MAC data paths.

Figure 21-2: Ethernet MAC Data Path Structure



In the transmit direction, the Ethernet MAC DMA will continue to fetch packet data up to a limit of 256 packets, or until the TX Packet Buffer Memory is full. The size of the TX Packet Buffer Memory is 16 Kbytes (4 Kbytes per queue). In the receive direction, if the RX Packet Buffer Memory becomes full, then an overflow will occur. An overflow will also occur if the limit of 256 packets is breached. The size of the RX Packet Buffer Memory is 4 Kbytes.

3.1.7.1. Transmit Packet Buffer

The Transmit Packet Buffer (TX Packet Buffer) will continue attempting to fetch frame data from the system memory until the TX Packet Buffer Memory is full, at which point it will attempt to maintain its full level.

To accommodate the status and statistics associated with each frame, 2 words per packet are reserved at the end of the packet data. If the packet was bad and requires to be dropped, the status and statistics are the only information held on that packet. Storing the status in the packet buffer is required in order to decouple the TX DMA interface of the buffer from the MAC Transmitter interface, to update the MAC status/statistics and to generate interrupts in the order in which the packets that they represent were fetched from the system memory.

If any errors occur on the AXI whilst reading the transmit frame then the fetching of packet data from memory is halted. The MAC Transmitter will continue to fetch packet data, thereby emptying the TX Packet Buffer Memory and allowing any good non-erroneous frames to be transmitted successfully. Once these have been fully transmitted, the status/statistics for the erroneous frame will be updated and software will be informed via an interrupt that an AXI error occurred. This way, the error is reported in the correct packet order.

The TX Packet Buffer will only attempt to read more frame data from the system memory when space is available in the TX Packet Buffer Memory. If space is not available it must wait until a packet fetched by the MAC Transmitter completes transmission and is subsequently removed from the TX Packet Buffer Memory. Note that if full store and forward mode is active and if a single frame is fetched that is too large for the TX Packet Buffer Memory, the frame is flushed and the TX DMA halted with an error status. This is because a complete frame must be written into the TX Packet Buffer Memory before transmission can begin.

In full store and forward mode, once the complete transmit frame is written into the TX Packet Buffer Memory, a trigger is sent across to the MAC Transmitter, which will then begin reading the frame from the TX Packet Buffer Memory. Since the whole frame is present and stable in the TX Packet Buffer Memory an underflow of the MAC Transmitter is not possible. The frame is kept in the TX Packet Buffer Memory until notification is received from the MAC Transmitter that the frame data has either been successfully transmitted or can no longer be re-transmitted. When this notification is received the frame is flushed from TX Packet Buffer Memory to make room for a new frame to be fetched from system memory.

In partial store and forward mode, a trigger is sent across to the MAC Transmitter as soon as sufficient packet data is available in the TX Packet Buffer Memory, which will then begin fetching the data from it. If, after this point, the MAC Transmitter is able to fetch data from the TX Packet Buffer Memory faster than the TX DMA can fill it, an underflow of the MAC Transmitter is possible. In this case, the transmission is terminated early, and the TX Packet Buffer Memory is flushed. Transmission can only be restarted by writing to the transmit start bit in the Network Control register (ETHERNETn_network_control[9]).

In full duplex mode, the frame is removed from the packet buffer on the fly.

3.1.7.2. Receive Packet Buffer

The Receive Packet Buffer (RX Packet Buffer) stores frames from the MAC Receiver along with their status and statistics. Frames with errors are flushed from the RX Packet Buffer Memory, whilst good frames are pushed onto the AXI Master Interface.

When programmed in full store and forward mode, if the frame has an error the frame data is immediately flushed from the RX Packet Buffer Memory allowing subsequent frames to utilize the freed up space. The status and statistics for bad frames are still used to update the **Ethernet MAC** registers.

To accommodate the status and statistics associated with each frame, up to 2 words (one being for descriptor timestamp capture when enabled) per packet are reserved at the end of the packet data. If the packet was bad and requires to be dropped, the status and statistics are the only information held on that packet.

The RX Packet Buffer will also detect a full condition such that an overflow condition can be detected. If this occurs subsequent packets will be dropped and an RX overflow interrupt is raised.

For full store and forward, the RX DMA will only begin packet fetches once the status and statistics for a frame are available. If the frame has a bad status due to a frame error, the status and statistics are

passed onto the **Ethernet MAC** registers. If the frame has a good status, the information is used to read the frame from the RX Packet Buffer Memory and burst onto the AXI Master Interface using the DMA buffer management protocol.

If partial store and forward mode is active, the RX DMA will begin fetching the packet data before the status is available. As soon as the status becomes available, the RX DMA will fetch this information before continuing to fetch the remainder of the frame.

3.1.8. Priority Queuing in the Ethernet MAC DMA

The Ethernet MAC DMA uses 4 transmit and 4 receive queues. Each queue has an independent list of buffer descriptors pointing to separate data streams.

In the transmit direction, higher priority queues are serviced before lower priority queues. This strict priority scheme requires the user to ensure that high priority traffic is constrained such that lower priority traffic will have the required bandwidth. The Ethernet MAC DMA will determine the next queue to service by initiating a sequence of buffer descriptor reads interrogating the ownership bits of each. The buffer descriptor corresponding to the highest priority queue is read first. If the ownership bit of this descriptor is “1”, then the Ethernet MAC DMA will progress to reading the 2nd highest priority queue’s descriptor. If that ownership bit read of this lower priority queue is “1”, then the Ethernet MAC DMA will read the 3rd highest priority queue’s descriptor, and so on. If all the descriptors return an ownership bit set, then a resource error has occurred, an interrupt is generated and transmission is automatically halted.

Transmission can only be restarted by setting the start bit (tx_start_clk) in the Network Control register (ETHERNETn_network_control[9]). The **Ethernet MAC** DMA will need to identify the highest available queue for transmit from when the start bit in the Network Control register is written to and the TX is in a halted state, or when the last word of any packet has been fetched from system memory.

The **Ethernet MAC** TX DMA will maximize the effectiveness of priority queuing by ensuring that high priority traffic be transmitted as early as possible after being fetched from system memory. High priority traffic will be pushed to the MAC layer depending on traffic shaping being enabled and the associated credit value for that queue, before any lower priority traffic that may pre-exist in the TX Packet Buffer. This is achieved by separating the TX Packet Buffer Memory into regions, 4 regions per queue. The size of each region determines the amount of memory space allocated per queue and is 1 Kbytes per region.

If a higher priority transmit queue contains relatively longer buffer, it can happen that a shorter frame from a lower priority queue is transferred before the higher prority queue. For each queue, there is an associated transmit buffer queue base address register. For the lowest priority queue (Queue 0), the TX Buffer Queue Base Address register is located at offset address 0x01C. For queues 1 to 3 the transmit buffer queue base address registers are located at sequential offset addresses starting at 0x440.

In the receive direction each data packet is written to system memory in the order that it is received. For each queue, there is an independent set of receive buffers for each queue. For each queue, there is an associated receive buffer queue base address register. For the lowest priority queue (Queue 0), the RX Buffer Queue Base Address register is located at offset address 0x018. For queues 1 to 3 the receive buffer queue base address registers are located at sequential offset addresses starting at 0x480. Every received packet will pass through a programmable screening algorithm which will allocate to that frame a particular queue to route it to. The interface to the screeners is via two banks of programmable registers, Screening Type 1 registers and Screening Type 2 registers.

Screening Type 1 registers allow routing received frames based on particular IP and UDP fields extracted from the received frames. Specifically these fields are DS (Differentiated Services field of IPv4 frames), TC (Traffic Class field of IPv6 frames) and/or the UDP destination port. These fields are compared against the values stored in each of the Screening Type 1 registers. If the result of this comparison is positive, then the received frame is routed to the priority queue specified in that Screening Type 1 register. Refer to Screening Type 1 register description for further programming details.

Screening Type 2 registers operate independently of Screening Type 1 registers and offer additional match capabilities, extending the capabilities into vendor specific protocols. The Type 2 screening allows a screen to be configured that is the combination of all or any of the following comparisons.

1. An enabled field VLAN Priority. A VLAN Priority match will be performed if the VLAN priority enable is set. The extracted priority field in the VLAN header is compared against 3 bits within the Screening Type 2 register itself.
2. An enabled field EtherType. The EtherType field inside the Screening Type 2 register maps to 1 of 8 EtherType Match registers. The extracted EtherType is compared against the EtherType 2 register designated by this EtherType field.
3. An enabled field Compare A.
4. An enabled field Compare B.
5. An enabled field Compare C.

Compare A, Compare B, and Compare C each have an Enable bit and Compare Register field. The Compare Register field is a pointer to a configured OFFSET, VALUE, and MASK. If enabled the compare is true if the data at the OFFSET into the frame, ANDed with the MASK value is equal to the COMPARE value. A 16-bit word comparison is done. The byte at the OFFSET number of bytes from the index start is compared through bits [15:8] of the configured VALUE and MASK. The OFFSET can be configured to be from 0 to 127 bytes from either the start of the frame, the byte following the EtherType field, the byte following the end of the IP header (IPv4 or IPv6) or the byte following the end of the TCP/UDP header. Note the logic to decode the IP header or the TCP/UDP header is reused from the TCP/UDP/IP checksum offload logic and therefore has the same restrictions on use (the main limitation is that IP fragmentation is not supported). Refer to Checksum Offload for IP, TCP, and UDP section for further details. The Compare Register field points to a single pool of 32 compare registers. Compare A, Compare B, and Compare C use a common set of compare registers.

Note Compare A, Compare B, and Compare C together allow matching against an arbitrary 48-bits of data and so can be used to match against a MAC address.

All enabled comparisons are ANDed together to form the overall Type 2 screener match. Refer to Screening Type 2 register descriptions for further programming details.

Each screener register is programmable vie the AHB Slave Interface. Although this is not recommended, it is possible that more than one screener register will be programmed to match against a single frame. If this happens there are 2 cases to consider.

1. If a received frame matches against multiple screeners of the same type, then the frame will route to the queue mapped by the screener located at the lowest numeric offset address. E.g. if Screening Type 2 Register 0 and Screening Type 2 Register 1 both match, then the frame will be routed to the queue identified in queue_number[3:0] of Screening Type 2 Register 0.
2. If a received frame matches against a Screening Type 2 and a Screening Type 1, then the Screening Type 1 will take precedence.

When a screener is matched, the received frame will be routed to a queue defined inside bits [3:0] of the screener register (ETHERNETn_screening_type_1_register_i or ETHERNETn_screening_type_2_register_i). Unmatched frames are routed to Queue 0.

The interrupt outputs from the **Ethernet MAC** match the number of supported priority queues. Only **Ethernet MAC** DMA related events are reported using the individual interrupt outputs, as the **Ethernet MAC** can relate these events to specific queues. All other events generated within the **Ethernet MAC** are reported in the interrupt associated with the lowest priority queue (Queue 0). For the lowest priority queue the Interrupt Status register is located at offset address 0x024. For all other priority queues, the Interrupt Status register is located at sequential offset addresses starting at 0x400.

3.2. MAC Transmitter Block

The MAC Transmitter operates in full duplex and transmits frames in accordance with the Ethernet IEEE Std 802.3 standard.

A small input buffer receives data through the External FIFO Interface (from the Ethernet MAC DMA module) which, depending on the data_bus_width control bits in the Network Configuration register, will extract data in 64-bit form. All subsequent processing prior to the final output is performed in bytes.

Transmit data can be output using the MII Interface.

Frame assembly starts by adding preamble and the start frame delimiter (SFD). Data is taken from the Transmit FIFO interface a word at a time. If necessary, padding is added to take the frame length to 60 bytes. CRC is calculated using an order 32 bit polynomial. This is inverted and appended to the end of the frame taking the frame length to a minimum of 64 bytes. If the “No CRC” bit (bit 16) is set in the

second word (Word 1) of the last buffer descriptor of a transmit frame neither pad nor CRC are appended. The “No CRC” bit can also be set through the FIFO interface.

In full duplex mode, frames are transmitted immediately. Back to back frames are transmitted at least 96 bit times apart to guarantee the Interpacket Gap.

If the collision signal is asserted during transmission, the transmitter will transmit a jam sequence of 32 bits taken from the data register and then retries transmission after the back off time has elapsed. If the collision occurs during either the preamble or SFD, then these fields will be completed prior to generation of the jam sequence.

The back off time is based on an XOR of the 10 least significant bits of the data coming from the transmit FIFO interface and a 10-bit pseudo random number generator. The number of bits used depends on the number of collisions seen. After the first collision 1 bit is used, then for the second collision 2 bits and so on up to the maximum of 10 bits. All 10 bits are used above ten collisions. An error will be indicated and no further attempts will be made if 16 consecutive attempts cause a collision. This operation is compliant with the description in Clause 4.2.3.2.5 of the IEEE Std 802.3 standard which refers to the truncated binary exponential back of algorithm.

In 10/100 MBit/s mode, both collisions and late collisions are treated identically and back off and retry will be performed up to 16 times.

In all modes of operation, if the TX DMA under runs, a bad CRC is automatically appended using the same mechanism as jam insertion and the TX_ER signal is asserted. For a properly configured system this should never happen and also it is impossible if configured to use the DMA with packet buffers, as the complete frame is buffered in TX Packet Buffer Memory.

By setting bit 28 in the Network Configuration register the IPG may be stretched beyond 96 bits depending on the length of the previously transmitted frame and the value written to the IPG Stretch register. Bits [7:0] of the IPG Stretch register multiplied with the previous frame length (including preamble) bits [15:8] (1 added so as not to get a divide by zero) divides the frame length to generate the IPG. The IPG Stretch register cannot be used to shrink the IPG below 96 bits.

If the back_pressure bit is set in the Network Control register (ETHERNETn_network_control[8]) the transmit block transmits 64 bits of data, which can consist of 16 nibbles of 1011_b or in bit rate mode 64 times 1b, whenever it sees an incoming frame to force a collision.

3.3. MAC Receiver Block

The MAC Receiver block checks for valid preamble, FCS, alignment and length, presents received frames to the External FIFO Interface and stores the frames destination address for use by the address checking block. If during frame reception, the frame is found to be too long, a bad frame indication is sent to the External FIFO Interface. The receiver logic ceases to send data to RX Packet Buffer Memory as soon as this condition occurs.

At End of Frame reception the MAC Receiver block indicates to the Ethernet MAC DMA whether the frame is good or bad. The Ethernet MAC DMA will recover the current receive buffer if the frame was bad.

Ethernet frames are normally stored in DMA memory or to the External FIFO Interface complete with the FCS. Setting the fcs_remove bit in the Network Configuration register (ETHERNETn_network_configuration[17]) causes frames to be stored without their corresponding FCS. The reported frame length field is reduced by four bytes to reflect this operation.

The MAC Receive block signals to the register block to increment the alignment, CRC (FCS), short frame, long frame, jabber or receive symbol errors when any of these exception conditions occur.

If bit 26 is set in the Network Configuration register, CRC errors will be ignored and CRC erroneous frames will not be discarded, though the Frame Check Sequence Errors statistics register will still be incremented. Additionally if configured to use the Ethernet MAC DMA and not enabled for jumbo frames mode, then bit 13 of the receive buffer descriptor Word 1 will be updated to indicate the FCS validity for the particular frame. This is useful for applications such as EtherCAT whereby individual frames with FCS errors must be identified.

Received frames can be checked for length field error by setting the length_field_error_frame_discard bit of the Network Configuration register (bit 16). When this bit is “1”, the receiver compares a frame’s measured length with the length field (bytes 13 and 14) extracted from the frame. The frame is discarded

if the measured length is shorter. This checking procedure is for received frames between 64 bytes and 1518 bytes in length.

Each discarded frame is counted in the 10-bit length field Frame Check Sequence Errors statistics register. Frames where the length field value is greater than or equal to 0600_h (1536) will not be checked.

3.4. Checksum Offload for IP, TCP, and UDP

The Ethernet MAC can be programmed to perform IP, TCP, and UDP checksum offloading in both receive and transmit directions which is enabled by setting bit 24 in the Network Configuration register for receive and bit 11 in the DMA Configuration register for transmit.

IPv4 packets contain a 16-bit checksum field, which is the 16-bit 1's complement of the 1's complement sum of all 16-bit words in the header. TCP and UDP packets contain a 16-bit checksum field, which is the 16-bit 1's complement of the 1's complement sum of all 16-bit words in the header, the data and a conceptual IP pseudo header.

To calculate these checksums in software requires each byte of the packet to be processed. For TCP and UDP this can use a large amount of processing power. Offloading the checksum calculation to hardware can result in significant performance improvements.

For IP, TCP, and UDP checksum offload to be useful, the operating system containing the protocol stack must be aware that this offload is available so that it can make use of the fact that the hardware can either generate or verify the checksum.

3.4.1. Receiver Checksum Offload

When receive checksum offloading is enabled in the Ethernet MAC, the IPv4 header checksum is checked as per RFC 791, where the packet meets the following criteria:

- If present, the VLAN header must be four octets long and the CFI bit must not be "1". (Also for receive one stacked VLAN is supported.)
- Encapsulation must be RFC 894 Ethernet Type Encoding or RFC 1042 SNAP Encoding or PPPoE Encoding.
- IPv4 packet.
- IP header is of valid length.
- IP options are supported.

The Ethernet MAC also checks the TCP checksum as per RFC 793, or UDP checksum as per RFC 768, if the following criteria are met:

- IPv4 or IPv6 packet.
- IP options and all IPv6 extension headers (i.e. hop-by-hop, routing and destination) are supported (except for fragmentation headers).
- Good IP header checksum (if IPv4).
- IP fragmentation is not supported. (If a packet is fragmented, then the checksum will not be checked)
- TCP or UDP packet.

When an IP, TCP, or UDP frame is received, the receive buffer descriptor gives an indication if the Ethernet MAC was able to verify the checksums. There is also an indication if the frame had SNAP encapsulation. These indication bits will replace the Type ID match indication bits when the receive checksum offload is enabled. For details of these indication bits refer to table **Receive Buffer Descriptor Entry**.

If any of the checksums are verified incorrect by the Ethernet MAC, the packet is discarded and the appropriate statistics counter incremented.

3.4.2. Transmitter Checksum Offload

The transmitter checksum offload is only available if the Ethernet MAC is configured to use the DMA in packet buffer mode and full store and forward mode is enabled. This is because the complete frame to be transmitted must be read into the TX Packet Buffer Memory before the checksum can be calculated and written back into the headers at the beginning of the frame.

Transmitter checksum offload is enabled by setting bit 11 in the DMA Configuration register. When enabled, it will monitor the frame as it is written into the TX Packet Buffer Memory to automatically detect the protocol of the frame. Protocol support is identical to the receiver checksum offload.

For transmit checksum generation and substitution to occur, the protocol of the frame must be recognized and the frame must be provided without the FCS field, by making sure that bit 16 of the transmit descriptor Word 1 is clear (VLAN tagged frames will be recognized but stacked VLAN tagged frames will not be recognized). If the frame data already had the FCS field, this would be corrupted by the substitution of the new checksum fields.

If these conditions are met, the transmit checksum offload engine will calculate the IP, TCP, and UDP checksums as appropriate. Once the full packet is completely written into TX Packet Buffer Memory, the checksums will be valid and the relevant memory locations will be updated for the new checksum fields as per standard IP/TCP and UDP packet structures.

If the transmitter checksum engine is prevented from generating the relevant checksums, bits [22:20] of the **Transmit Buffer Descriptor Entry** will be updated to identify the reason for the error. Note that the frame will still be transmitted but without the checksum substitution, as typically the reason that the substitution did not occur was that the protocol was not recognized.

3.5. MAC Filtering Block

The filter block determines which frames should be written to the External FIFO Interface and on to the Ethernet MAC DMA.

Whether a frame is passed depends on what is enabled in the Network Configuration register, the contents of the Specific Address, Type ID Match and Hash registers and the frame's destination address and type field.

Ethernet frames are transmitted a byte at a time, least significant bit first. The first six bytes of an Ethernet frame make up the destination address. The first bit of the destination address, which is the LSB of the first byte of the frame, is the group or individual bit. This is "1" for multicast addresses and "0" for unicast ones. The "all ones" address is the broadcast address and a special case of multicast.

The Ethernet MAC supports recognition of specific source or destination addresses. The number of specific source or destination address filters is four. Each specific address filter consists of two registers, Specific Address Bottom i register and Specific Address Top i register. Specific Address Bottom i register stores the first four bytes of the compares source or destination address. Specific Address Top i register contains the last two bytes of this address, a control bit to select between source or destination address filtering and a 6-bit byte mask field to allow masking certain bytes during the comparison. The first filter (Filter 1) is slightly different to all other filters in that there is no byte mask. Instead address comparison against individual bits of Specific Address 1 register can be masked using the unique Specific Address Mask 1 register. The addresses stored in all filters can be specific (unicast), group (multicast), local or universal.

The destination or source address of received frames is compared against the data stored in the Specific Address registers once they have been activated. The addresses are deactivated at reset or when their corresponding Specific Address Bottom i register is written. They are activated when the corresponding Specific Address Top i register is written. If a receive frame address matches an active address, the frame is written to the External FIFO Interface and on to DMA memory if used.

Frames may be filtered using the Type ID field for matching. Four Type ID Match registers exist and each can be enabled for matching by writing a "1" to the MSB (bit 31) of the respective register. When a frame is received, the matching is implemented as an OR function of the various types of match.

The content of each Type ID register (when enabled) is compared against the length/Type ID of the frame being received (e.g. bytes 13 and 14 in non-VLAN and non-SNAP encapsulated frames) and copied to system memory if a match is found. The encoded Type ID match bits (Word 0, bit 22 and bit

23) in the receive buffer descriptor status are set indication which Type ID Match register generated the match, if the receive checksum offload is disabled.

The reset state of the Type ID Match registers is “0”, hence each is initially disabled.

The following example illustrates the use of the address and Type ID Match registers for a MAC address of 21:43:65:87:A9:CB:

Preamble	55_h
SFD	$D5_h$
DA (Octet 0 – LSB)	21_h
DA (Octet 1)	43_h
DA (Octet 2)	65_h
DA (Octet 3)	87_h
DA (Octet 4)	$A9_h$
DA (Octet 5 - MSB)	CB_h
SA (Octet 0 - LSB)	1.)
SA (Octet 1)	1.)
SA (Octet 2)	1.)
SA (Octet 3)	1.)
SA (Octet 4)	1.)
SA (Octet 5 - MSB)	1.)
Type ID (MSB)	43_h
Type ID (LSB)	21_h

1. Contains the address of the transmitting device.

The sequence above shows the beginning of an Ethernet frame. Byte order of transmission is from top to bottom as shown. For a successful match to Specific Address 1, the following address match registers must be set up:

Specific Address 1 Bottom (0x088)	87654321_h
Specific Address 1 Top (0x08C)	$0000CBA9_h$

And for a successful match to the Type ID, the Type ID Match 1 register must be set up:

Type ID Match 1 (0x0A8)	80004321_h

3.5.1. Broadcast Address

Frames with the broadcast address of $FFFFFFFFFFFF_h$ are stored to memory only if the no_broadcast bit in the Network Configuration register is set to “0”.

3.5.2. Hash Addressing

The 64-bit Hash register (Hash Top/Hash Bottom) takes up two locations in the memory map. The least significant bits are stored in Hash Bottom register and the most significant bits in Hash Top register.

The unicast hash enable and the multicast hash enable bits in the Network Configuration register enable the reception of hash matched frames. The destination address is reduced to a 6-bit index into the 64-bit Hash register using the following hash function. The hash function is an XOR of every sixth bit of the destination address.

```
  hash_index[05] = da[05] ^ da[11] ^ da[17] ^ da[23] ^ da[29] ^ da[35] ^ da[41] ^
da[47]

  hash_index[04] = da[04] ^ da[10] ^ da[16] ^ da[22] ^ da[28] ^ da[34] ^ da[40] ^
da[46]

  hash_index[03] = da[03] ^ da[09] ^ da[15] ^ da[21] ^ da[27] ^ da[33] ^ da[39] ^
da[45]
```

```
hash_index[02] = da[02] ^ da[08] ^ da[14] ^ da[20] ^ da[26] ^ da[32] ^ da[38] ^
da[44]
hash_index[01] = da[01] ^ da[07] ^ da[13] ^ da[19] ^ da[25] ^ da[31] ^ da[37] ^
da[43]
hash_index[00] = da[00] ^ da[06] ^ da[12] ^ da[18] ^ da[24] ^ da[30] ^ da[36] ^
da[42]
```

`da[00]` represents the least significant bit of the first byte received, that is, the multicast/unicast indicator, and `da[47]` represents the most significant bit of the last byte received.

If the hash index points to a bit that is set in the Hash register then the frame will be matched according to whether the frame is multicast or unicast.

A multicast match will be signaled if the multicast hash enable bit is set, `da[00]` is “1” and the hash index points to a bit set in the Hash register.

A unicast match will be signaled if the unicast hash enable bit is set, `da[00]` is “1” and the hash index points to a bit set in the Hash register.

To receive all multicast frames, the Hash register must be set with all “1” and the multicast hash enable bit must be set to “1” in the Network Configuration register.

3.5.3. Copy All Frames (or Promiscuous Mode)

If the copy all frames bit is set in the Network Configuration register then all frames (except those that are too long, too short, have FCS errors or have RX_ER asserted during reception) will be copied to memory. Frames with FCS errors will be copied if bit 26 is set in the Network Configuration register.

3.5.4. Disable Copy of Pause Frames

Pause frames can be prevented from being written to memory by setting the disable copying of pause frames control bit 23 in the Network Configuration register. When set, pause frames are not copied to system memory regardless of the copy all frames bit, whether a hash match is found, a type ID match is identified or if a destination address match is found.

3.5.5. VLAN Support

An Ethernet encoded 802.1Q VLAN tag looks like this:

TPID (Tag Protocol Identifier) 16 bits	TCI (Tag Control Information) 16 bits
8100_h	First 3 bits priority, then CFI bit, last 12 bits VID

The VLAN tag is inserted at the 13th byte of the frame adding an extra four bytes to the frame. To support these extra four bytes, the Ethernet MAC can accept frame lengths up to 1536 bytes by setting bit 8 in the Network Configuration register.

If the VID (VLAN identifier) is null (000_h) this indicates a priority-tagged frame.

The following bits in the receive buffer descriptor status word give information about VLAN tagged frames:

- Bit 21 set if receive frame is VLAN tagged (i.e. Type ID of 8100h).
- Bit 20 set if receive frame is priority tagged (i.e. Type ID of 8100h and null VID). (If bit 20 is set bit 21 will be set also.)
- Bit 19, 18, and 17 set to priority if bit 21 is set.

- Bit 16 set to CFI if bit 21 is set.

The Ethernet MAC can be configured to reject all frames except VLAN tagged frames by setting the discard non-VLAN frames bit in the Network Configuration register.

3.6. IEEE Std 1588 and IEEE Std 802.1AS Support

IEEE Std 1588 is a standard for precision time synchronization in local area networks. It works with the exchange of special Precision Time Protocol (PTP) frames. The PTP messages can be transported over IEEE Std 802.3/Ethernet, over Internet Protocol Version 4 (IPv4) or over Internet Protocol Version 6 (IPv6) as described in the annex of IEEE Std 1588-2008.

Most IEEE Std 1588 functionality can be implemented in software but for greatest accuracy hardware assist is required to detect when PTP event messages pass the MII interface (clock timestamp point).

Synchronization between master and slave clocks is a two stage process.

First the offset between the master and slave clocks is corrected by the master sending a Sync frame to the slave with a follow up frame containing the exact time the Sync frame was sent. Hardware assist modules at the master and slave side detect exactly when the Sync frame was sent by the master and received by the slave. The slave then corrects its clock to match the master clock.

Second the transmission delay between the master and slave is corrected. The slave sends a delay request frame to the master which sends a delay response frame in reply. Hardware assist modules at the master and slave side detect exactly when the delay request frame was sent by the slave and received by the master. The slave will now have enough information to adjust its clock to account for delay. For example if the slave was assuming zero delay the actual delay will be half the difference between transmit and receive time of the delay request frame (assuming equal transmit and receive times) because the slave clock will be lagging the master clock by the delay time already.

For hardware assist it is necessary to timestamp when Sync and Delay_Req messages are sent and received. The timestamp is taken when the message timestamp point passes the clock timestamp point. For Ethernet the message timestamp point is the SFD and the clock timestamp point is MII interface. (The IEEE Std 1588 spec refers to Sync and Delay_Req messages as event messages as these require time stamping. Follow up, delay response and management messages do not require time stamping and are referred to as general messages.)

IEEE Std 1588 version 2 (IEEE Std 1588-2008) defines two additional PTP event messages. These are the peer delay request (Pdelay_Req) and peer delay response (Pdelay_Resp) messages. These messages are used to calculate the delay on a link. Nodes at both ends of a link send both types of frames (regardless of whether they contain a master or slave clock). The Pdelay_Resp massage contains the time at which a Pdelay_Req was received and is itself an event message. The time at which a Pdelay_Resp message is received is returned in a Pdelay_Resp_Follow_Up message.

IEEE Std 1588 version 2 introduces transparent clocks of which there are two kinds, peer-to-peer (P2P) and end-to-end (E2E). Transparent clocks measure the transit time of event messages through a bridge and amend a correction field within the message to allow for the transit time. P2P transparent clocks additionally correct for the delay in the receive path of the link using the information gathered from the peer delay frames. With P2P transparent clocks Delay_Req messages are not used to measure link delay. This simplifies the protocol and makes larger systems more stable.

The Ethernet MAC recognizes ten different encapsulations for PTP event messages:

1. IEEE Std 1588 version 1 (UDP/IPv4 multicast)
2. IEEE Std 1588 version 1 (UDP/IPv4 multicast with VLAN)
3. IEEE Std 1588 version 2 (UDP/IPv4 multicast)
4. IEEE Std 1588 version 2 (UDP/IPv4 multicast with VLAN)
5. IEEE Std 1588 version 2 (UDP/IPv4 unicast)
6. IEEE Std 1588 version 2 (UDP/IPv4 unicast with VLAN)
7. IEEE Std 1588 version 2 (UDP/IPv6 multicast)
8. IEEE Std 1588 version 2 (UDP/IPv6 multicast with VLAN)
9. IEEE Std 1588 version 2 (Ethernet multicast)
10. IEEE Std 1588 version 2 (Ethernet multicast with VLAN)

Note: IEEE Std 1588 version 1 (IEEE Std 1588-2002)

Unicast PTP frame recognition is enabled via bit 20 of the Network Control register. The unicast addresses themselves are programmable via the PTP Unicast IP Destination Address register for which there are two (ETHERNETn_rx_ptp_unicast and ETHERNETn_tx_ptp_unicast). The first holds the RX unicast IP destination address and the other the TX unicast destination address. The PTP Unicast IP Destination Address register should only be changed when the unicast PTP frame recognition is disabled.

Example of a Sync frame in the IEEE Std 1588 version 1 format:

Preamble/SFD	55555555555555D5$_h$
DA (Octets 0 – 5)	
SA (Octets 6 – 11)	
Type (Octets 12 – 13)	0800$_h$
IP stuff (Octets 14 – 22)	
UDP (Octet 23)	11$_h$
IP stuff (Octets 24 – 29)	
IP DA (Octets 30 - 32)	E00001$_h$
IP DA (Octet 33)	81$_h$ or 82$_h$ or 83$_h$ or 84$_h$
source IP port (Octets 34 – 35)	
dest IP port (Octets 36 – 37)	013F$_h$
other stuff (Octets 38 – 42)	
version PTP (Octet 43)	01$_h$
other stuff (Octets 44 – 73)	
control (Octet 74)	00$_h$
other stuff (Octets 75 – 168)	

Example of a Delay_Req frame in the IEEE Std 1588 version 1 format:

Preamble/SFD	55555555555555D5$_h$
DA (Octets 0 – 5)	
SA (Octets 6 – 11)	
Type (Octets 12 – 13)	0800$_h$
IP stuff (Octets 14 – 22)	
UDP (Octet 23)	11$_h$
IP stuff (Octets 24 – 29)	
IP DA (Octets 30 - 32)	E00001$_h$
IP DA (Octet 33)	81$_h$ or 82$_h$ or 83$_h$ or 84$_h$
source IP port (Octets 34 – 35)	
dest IP port (Octets 36 – 37)	013F$_h$
other stuff (Octets 38 – 42)	
version PTP (Octet 43)	01$_h$
other stuff (Octets 44 – 73)	
control (Octet 74)	01$_h$
other stuff (Octets 75 – 168)	

For IEEE Std 1588 version 1 messages Sync and Delay_Req frames are indicated by the Ethernet MAC if the frames type field indicates TCP/IP, UDP protocol is indicated, the destination IP address is 224.0.1.129/130/131/132, the destination UDP port is 319 and the control field is correct. The control field is 00$_h$ for Sync frames and 01$_h$ for Delay_Req frames.

For IEEE Std 1588 version 2 messages the type of frame is determined by looking at the message type field in the first byte of the PTP frame. Whether a frame is version 1 or version 2 can be determined by looking at the version PTP field in the second byte of version 1 and version 2 PTP frames.

In version 2 messages Sync frames have a message type value of 0$_h$, Delay_Req frames have 1$_h$, Pdelay_Req frames have 2$_h$ and Pdelay_Resp frames have 3$_h$.

Example of a Sync frame in the IEEE Std 1588 version 2 (UDP/IPv4) format:

Preamble/SFD	55555555555555D5$_h$
DA (Octets 0 – 5)	
SA (Octets 6 – 11)	
Type (Octets 12 – 13)	0800$_h$

IP stuff (Octets 14 – 22)	
UDP (Octet 23)	11_h
IP stuff (Octets 24 – 29)	
IP DA (Octets 30 - 33)	$E0000181_h$
source IP port (Octets 34 – 35)	
dest IP port (Octets 36 – 37)	$013F_h$
other stuff (Octets 38 – 41)	
message type (Octet 42)	00_h
version PTP (Octet 43)	02_h

Example of a Pdelay_Req frame in the IEEE Std 1588 version 2 (UDP/IPv4) format:

Preamble/SFD	$55555555555555D5_h$
DA (Octets 0 – 5)	
SA (Octets 6 – 11)	
Type (Octets 12 – 13)	0800_h
IP stuff (Octets 14 – 22)	
UDP (Octet 23)	11_h
IP stuff (Octets 24 – 29)	
IP DA (Octets 30 - 33)	$E000006B_h$
source IP port (Octets 34 – 35)	
dest IP port (Octets 36 – 37)	$013F_h$
other stuff (Octets 38 – 41)	
message type (Octet 42)	02_h
version PTP (Octet 43)	02_h

Example of a Sync frame in the IEEE Std 1588 version 2 (UDP/IPv6) format:

Preamble/SFD	$55555555555555D5_h$
DA (Octets 0 – 5)	
SA (Octets 6 – 11)	
Type (Octets 12 – 13)	$86DD_h$
IP stuff (Octets 14 – 19)	
UDP (Octet 20)	11_h
IP stuff (Octets 21 – 37)	
IP DA (Octets 38 - 53)	$FF0X000000000181_h$
source IP port (Octets 54 – 55)	
dest IP port (Octets 56 – 57)	$013F_h$
other stuff (Octets 58 – 61)	
message type (Octet 62)	00_h
other stuff (Octets 63 – 93)	
version PTP (Octet 94)	02_h

Example of a Pdelay_Req frame in the IEEE Std 1588 version 2 (UDP/IPv6) format:

Preamble/SFD	$55555555555555D5_h$
DA (Octets 0 – 5)	
SA (Octets 6 – 11)	
Type (Octets 12 – 13)	$86DD_h$
IP stuff (Octets 14 – 19)	
UDP (Octet 20)	11_h
IP stuff (Octets 21 – 37)	
IP DA (Octets 38 - 53)	$FF0200000000006B_h$
source IP port (Octets 54 – 55)	
dest IP port (Octets 56 – 57)	$013F_h$
other stuff (Octets 58 – 61)	
message type (Octet 62)	03_h
other stuff (Octets 63 – 93)	
version PTP (Octet 94)	02_h

Example of a Sync frame in the IEEE Std 1588 version 2 (Ethernet multicast) format. For the multicast address 011B19000000$_h$ Sync and Delay_Req frames are recognized depending on the message type field, 00h for Sync and 01h for Delay_Req:

Preamble/SFD	55555555555555D5$_h$
DA (Octets 0 – 5)	011B19000000$_h$
SA (Octets 6 – 11)	
Type (Octets 12 – 13)	88F7$_h$
message type (Octet 14)	00$_h$
version PTP (Octet 15)	02$_h$

Example of a Pdelay_Req frame in the IEEE Std 1588 version 2 (Ethernet multicast) format, these need a special multicast address so they can get through ports blocked by the spanning tree protocol. For the multicast address 0180C200000E$_h$ Sync, Pdelay_Req and Pdelay_Resp frames are recognized depending on the message type field, 00h for Sync, 02$_h$ for Pdelay_Req and 03$_h$ for Pdelay_Resp:

Preamble/SFD	55555555555555D5$_h$
DA (Octets 0 – 5)	0180C200000E$_h$
SA (Octets 6 – 11)	
Type (Octets 12 – 13)	88F7$_h$
message type (Octet 14)	02$_h$
version PTP (Octet 15)	02$_h$

The **Ethernet MAC** contains a timestamp unit (TSU) which consists of a timer and registers to capture the time at which PTP event frames cross the message timestamp point. The registers are accessible through the **Ethernet MAC**'s APB slave interface. An interrupt is issued when a capture register is updated.

3.6.1. Support for Time Stamping and Timestamp Accuracy

The MAC has the responsibility of sampling the TSU timer value when the TX or RX SOF event of the frame passes the MII boundary. This event is an existing signal synchronous to MAC TX/RX clock domains. The MAC uses the sampled timestamp to insert the timestamp into transmitted PTP Sync frames (if one step sync feature is enabled), or to pass to the Ethernet MAC's register block to capture the timestamp (TS) in registers, or to pass to the Ethernet MAC DMA to insert into TX or RX buffer descriptors. For each of these, the SOF event, which is captured in the TX and RX clock domains respectively, is synchronized to the TSU clock domain and the resulting signal is used to sample the TSU count value. This value will be kept stable for an entire frame, or specifically at least 64 TX/RX clock cycles, since the minimum frame size in Ethernet is 64 bytes and worst case is a transfer rate of 1 byte per cycle. It is used as the source for all the components within the Ethernet MAC that require the timestamp value. Since the SOF event had to pass a clock boundary, there is a degree of inaccuracy in the captured timestamp. The level of inaccuracy depends on the frequency of the TSU clock (tsu_clk). There will be no more than 1 clock cycle of inaccuracy.

In the best case, the SOF event (which is in the TX/RX clock domain) just meets the setup time of the TSU clock domain at the input to the first synchronization FlipFlop. The captured TS is N+2, but it really should be N+1. In the worst case, the captured TS is also N+2, but it really should be N.

3.6.2. Single Step Time Stamping

Support of one step clock for TX Sync frames can be enabled by setting bit 24 in the Network Control register. In this mode the timestamp field, within the IEEE Std 1588 version 2 Sync frame, is replaced by the TSU timestamp value at the time the Sync frame SOF passes the MII Interface To use single step time stamping, the sampled timestamp must be stable before the point at which Ethernet MAC requires to insert the timestamp. This can be guaranteed by enforcing a rule that TSU clock (tsu_clk) is greater than 1/8th the frequency of TX clock (TX_CLK) or RX clock (RX_CLK).

3.6.3. Timestamp Capture in Registers

There are four 80-bit timestamp status registers that capture the time at which PTP event frames are transmitted and received.

- {ETHERNETn_tsu_ptp_rx_msb_sec, ETHERNETn_tsu_ptp_rx_sec, ETHERNETn_tsu_ptp_rx_nsec}
- {ETHERNETn_tsu_ptp_tx_msb_sec, ETHERNETn_tsu_ptp_tx_sec, ETHERNETn_tsu_ptp_tx_nsec}
- {ETHERNETn_tsu_peer_rx_msb_sec, ETHERNETn_tsu_peer_rx_sec, ETHERNETn_tsu_peer_rx_nsec}
- {ETHERNETn_tsu_peer_tx_msb_sec, ETHERNETn_tsu_peer_tx_sec, ETHERNETn_tsu_peer_tx_nsec}

An interrupt is issued when these registers are updated.

3.6.4. Timestamp Capture in DMA Descriptors

The TX and/or RX timestamp can optionally be captured in an extended buffer descriptor when configured using bits [29:28] in the DMA Configuration register. The timestamp can be captured for a number of frame types (PTP event or PTP general, or all frames, or none as defined in TX BD Control/RX BD Control registers) and a bit within Buffer Descriptor Word 0/1 is used to indicate that the timestamp is present.

3.6.5. Controlling the Timestamp Unit

The timer is implemented as a 94-bit register with the upper 48 bits counting seconds, the next 30 bits counting nanoseconds and the lowest 16 bits counting sub-nanoseconds. The lower 46 bits roll over when they have counted to one second. An interrupt is generated when the seconds increment. The timer value can be read, written and adjusted through the APB Slave Interface. The timer is clocked with clock **CLK_HAPP1B0**.

There are two modes of operation to control the way the timer varies over time. These are:

Increment timer by a fixed value every clock (**CLK_HAPP1B0**). This is increment mode.

Increment timer by a fixed value for a fixed number of clocks, followed by an alternative increment value for a single clock. This is alternative increment mode.

3.6.5.1. Increment Mode

The amount by which the timer increments each clock cycle is controlled by the IEEE 1588 Timer Increment register. Bits [7:0] are the default increment value in nanoseconds and additional 16-bits of sub-nanoseconds resolution are available using the IEEE 1588 Timer Increment Sub Nanoseconds register. If the rest of the IEEE 1588 Timer Increment register is written with “0” the timer increments by the value in bits [7:0], plus the value in IEEE 1588 Timer Nanoseconds register, each clock cycle.

The IEEE 1588 Timer Nanoseconds register allows a resolution of approximately 15 femtoseconds (1 ns/65536).

3.6.5.2. Alternative Increment Mode

Bits [15:8] of the IEEE 1588 Timer Increment register are the alternative increment value in nanoseconds and bits [23:16] are the number of increments after which the alternative increment value is used. If bits [23:16] are 00_h then the alternative increment value will never be used.

The timer count value can be compared to a programmable comparison value. For the comparison the 48 bits of the seconds value and the upper 22 bits of the nanoseconds value are used. An interrupt can be issued when the timer count value and the comparison value in the IEEE 1588 Timer Comparison Value registers (ETHERNETn_tsu_msb_sec_cmp, ETHERNETn_tsu_sec_cmp, ETHERNETn_tsu_nsec_cmp) is equal. The interrupt can be enabled with bit 29 in the Interrupt Enable register.

IEEE Std 802.1AS is mostly a subset of IEEE Std 1588. There is one difference in that IEEE Std 802.1AS uses the Ethernet multicast address $0180C200000E_h$ for Sync frame recognition whereas IEEE

Std 1588 does not. Ethernet MAC is designed to recognize Sync frames with both IEEE Std 1588 and IEEE Std 802.1AS addresses and so can support their frame recognition simultaneously.

3.7. MAC IEEE Std 802.3 Pause Frame Support

The Ethernet MAC supports both hardware controlled pause of the transmitter upon reception of a pause frame and hardware generated pause frame transmission.

Note: See Clause 31, and Annex 31A and 31B of the IEEE Std 802.3 for a full description of pause operation.

The start of an IEEE Std 802.3 pause frame looks like this:

Destination Address	Source Address	Type (MAC Control Frame)	Pause Opcode	Pause Time
$0180C2000001_h$	6 bytes	8808_h	0001_h	2 bytes

3.7.1. IEEE Std 802.3 Pause Frame Reception

Bit 13 of the Network Configuration register is the pause enable control for reception. If this bit is set transmission will pause if a non zero pause quantum frame is received.

If a valid pause frame is received then the Receive Pause Quantum register is updated with the new frame's pause time regardless of whether a previous pause frame is active or not. An interrupt (either bit 12 or 13 of the Interrupt Status register) is triggered when a pause frame is received, but only if the interrupt has been enabled. Pause frames received with non-zero quanta are indicated through the interrupt bit 12 of the Interrupt Status register. Pause frames received with zero quanta are indicated on bit 13 of the Interrupt Status register.

Once the Receive Pause Quantum register (ETHERNETn_pause_time) is loaded and the frame currently being transmitted has been sent, no new frames are transmitted until the pause time reaches zero. The loading of a new pause time, and hence pausing of transmission, occurs since the Ethernet MAC is operating in full duplex mode. A valid pause frame is defined as having a destination address that matches either the address stored in Specific Address 1 register or if it matches the reserved address of $0180C2000001_h$. It must also have the MAC control frame Type ID of 8808_h and have the pause opcode of 0001_h.

Pause frames that have FCS or other errors will be treated as invalid and will be discarded. IEEE Std 802.3 pause frames that are received after Priority based Flow Control (PFC) has been negotiated will also be discarded. Valid pause frames received will increment the Pause Frames Received statistics register.

The Receive Pause Quantum register decrements every 512 bit times once transmission has stopped. For test purposes, the retry test bit can be set (bit 12 in the Network Configuration register) which causes the Receive Pause Quantum register to decrement every TX_CLK cycle once transmission has stopped.

The interrupt (bit 13 in the Interrupt Status register) is asserted whenever the Receive Pause Quantum register decrements to zero and it is enabled. This interrupt is also set when a zero quantum pause frame is received.

3.7.2. IEEE Std 802.3 Pause Frame Transmission

Automatic transmission of pause frames is supported through the transmit pause frame bits of the Network Control register. If either bit 11 or bit 12 of the Network Control register is written with "1", an IEEE Std 802.3 pause frame will be transmitted providing the MAC Transmitter is enabled (bit 3) in the Network Control register.

Pause frame transmission will happen immediately if transmit is inactive or if transmit is active between the current frame and the next frame due to be transmitted.

Transmitted pause frames comprise of the following:

- A destination address of $0180C2000001_h$
- A source address taken from Specific Address 1 register
- A Type ID of 8808_h (MAC control frame)
- A pause opcode of 0001_h
- A pause quantum register
- Fill of 00_h to take the frame to minimum frame length
- Valid FCS

The pause quantum used in the generated frame will depend on the trigger source for the frame as follows:

- If bit 11 is written with “1”, the pause quantum will be taken from the Transmit Pause Quantum register. The Transmit Pause Quantum register resets to a value of 0xFFFF giving maximum pause quantum as initial value.
- If bit 12 is written with “1”, the pause quantum will be zero.

After transmission, a pause frame transmitted interrupt will be generated (bit 14 of the Interrupt Status register) and the only statistics register that will be incremented will be the Pause Frames Transmitted register.

Pause frames can also be transmitted by the MAC using normal frame transmission methods.

3.8. MAC PFC Based Pause Frame Support

The Ethernet MAC supports PFC Priority Based Pause transmission and reception. Before PFC pause frames can be received, bit 16 of the Network Control register must be set to “1”.

Note: Refer to IEEE Std 802.1Qbb for a full description of priority based pause operation.

The start of a PFC pause frame looks like this:

Destination Address	Source Address	Type (MAC Control Frame)	Pause Opcode	Priority Enable Vector	Pause Time
$0180C2000001_h$	6 bytes	8808_h	0101_h	2 bytes	2 bytes

3.8.1. PFC Pause Frame Reception

The ability to receive and decode priority based pause frames is enabled by setting bit 16 of the Network Control register. When this bit is set, the Ethernet MAC will match either classic IEEE Std 802.3 pause frames or PFC priority based pause frames. Once a priority based pause frame has been received and matched, then from that moment on the Ethernet MAC will only match on priority based pause frames (this is IEEE Std 802.1Qbb requirement, known as PFC negotiation). Once priority based pause has been negotiated, any received IEEE Std 802.3x format pause frames will not be acted upon.

If a valid priority based pause frame is received then the Ethernet MAC will decode the frame and determine which, if any, of the 8 priorities require to be paused. Up to 8 pause time registers are then updated with the 8 pause times extracted from the frame regardless of whether a previous pause operation is active or not. An interrupt (either bit 12 or bit 13 of the Interrupt Status register) is triggered when a pause frame is received, but only if the interrupt has been enabled. Pause frames received with non-zero quanta are indicated through the interrupt bit 12 of the Interrupt Status register. Pause frames received with zero quanta are indicated on bit 13 of the Interrupt Status register. The loading of a new pause time occurs since the Ethernet MAC is operating in full duplex mode. A valid pause frame is defined as having a destination address that matches either the address stored in Specific Address 1 register of if it matches the reserved address of $0180C2000001_h$. It must also have the MAC control frame Type ID of 8808_h and have the pause opcode 0101_h.

Pause frames that have FCS or other errors will be treated as invalid and will be discarded. Valid pause frames received will increment the Pause Frames Received statistic register.

The Receive Pause Quantum register decrement every 512 bit times immediately, following the PFC frame reception. For test purposes, the retry test bit can be set (bit 12 in the Network Configuration

register) which causes the Receive Pause Quantum register to decrement every RX_CLK cycle once transmission has stopped.

The interrupt (bit 13 in the Interrupt Status register) is asserted whenever the Receive Pause Quantum register decrements to zero and it is enabled. This interrupt is also set when a zero quantum pause frame is received.

3.8.2. PFC Pause Frame Transmission

Automatic transmission of pause frames is supported through the transmit priority based pause frame bit of the Network Control register. If bit 17 of the Network Control register is written with “1”, a PFC pause frame will be transmitted providing the MAC Transmitter is enabled (bit 3) in the Network Control register. When bit 17 of the Network Control register is set to “1”, the fields of the priority base pause frame will be built using the values stored in the Transmit PFC Pause register.

Pause frame transmission will happen immediately if transmit is inactive or if transmit is active between the current frame and the next frame due to be transmitted.

Transmitted pause frames comprise of the following:

- A destination address of $0180C2000001_h$
- A source address taken from Specific Address 1 register
- A Type ID of 8808h (MAC control frame)
- A pause opcode of 0101h
- A priority enable vector taken from the Transmit PFC Pause register
- 8 pause quanta in 4 registers (ETHERNETn_tx_pause_quantum, ETHERNETn_tx_pause_quantum1, ETHERNETn_tx_pause_quantum2, ETHERNETn_tx_pause_quantum3)
- Fill of 00h to take the frame to minimum frame length
- Valid FCS

The pause quantum registers used in the generated frame will depend on the trigger source for the frame as follows:

- If bit 17 of the Network Control register is written with “1” then the priority enable vector of the priority based pause frame will be set equal to the value stored in the Transmit PFC Pause register bits [7:0]. For each entry equal to “0” in the Transmit PFC Pause register [15:8], the pause quantum field of the pause frame associated with that entry will be taken from the Transmit Pause Quantum register. For each entry equal to "1" in the Transmit PFC Pause register [15:8], the pause quantum associated with that entry will be "0". The Transmit Pause Quantum register resets to a value of $FFFF_h$ giving maximum pause quantum as initial value.
- The pause quantum registers are classed as static and these registers should be updated only when no PFC frame is transmitted.
- To use the eight priority pause quanta stored in the four Transmit Pause Quantum registers, set bit 24 to “1” in the Network Control register.

After transmission, a pause frame transmit interrupt will be issued (bit 14 of the Interrupt Status register) and the only statistics register that will be incremented will be the Pause Frames Transmitted register.

PFC Pause frames can also be transmitted by the MAC using normal frame transmission methods.

3.9. Energy Efficient Ethernet Support

IEEE Std 802.3az adds support for energy efficiency to Ethernet (EEE). These are the key features of IEEE Std 802.3az:

1. Allows a system's transmit path to enter a low power mode if there is nothing to transmit.
2. Allows a PHY to detect whether its link partner's transmit path is in low power mode, therefore allowing the system's receive path to enter low power mode.
3. Link remains up during lower power mode and no frames are dropped.
4. Asymmetric, one direction can be in low power mode while the other is transmitting normally.
5. LPI (Low Power idle) signaling is used to control entry and exit to and from low power modes.
6. LPI signaling can only take place if both sides have indicated support for it through auto-negotiation.

IEEE Std 802.3az operation:

1. Low-power control is done at MII (reconciliation sublayer).
2. As an architectural convenience in writing the 802.3az it is assumed that transmission is deferred by asserting carrier sense, in practice it will not be done this way. This system will know when it has nothing to transmit and only enter low power mode when it is not transmitting.
3. LPI should not be requested unless the link has been up for at least one second.
4. LPI is signaled on the MII transmit path by asserting 0x01 on TXD with TX_EN low and TX_ER high.
5. A PHY on seeing LPI requested on MII will send the sleep signal before going quiet. After going quiet it will periodically emit refresh signals.
6. The sleep, quiet and refresh periods are defined in Table 78-2 of IEEE Std 802.3az. E.g. for 1000BASE-X the sleep period (T_s) is 20 microseconds, the quiet period (T_q) is 2.5 milliseconds, and the refresh period (T_r) is 20 microseconds.
7. 1000BASE-X is required to go quiet after sleep is signaled. The easiest way to do this is to write to a control register to disable transmit in the SerDes.
8. SGMII and XFI are not part of IEEE Std 802.3az and should not go quiet after sleep is signaled.
9. LPI mode ends by transmitting normal idle for the wake time. There is a default time for this but it can be adjusted in software using the Link Layer Discovery Protocol (LLDP) described in Clause 79 of IEEE Std 802.3az.
10. LPI is indicated at the receive side when sleep and refresh signaling has been detected.

3.10. LPI Operation in Ethernet MAC

Auto-negotiation:

1. Indicate EEE capability using auto-negotiation.

For the transmit path:

1. If the link has been up for 1 second and there is nothing being transmitted, write to the LPI bit (bit 19) in the Network Control register.
2. If connected to 1000BASE-T PHY using SGMII or RGMII there is nothing more to do.
3. If connected to a backplane using a 1000BASE-KX PHY use firmware to periodically disable the SerDes transmit path. (Write to bit 1.160.0 for 1000BASE-KX.)
4. Wake up by clearing the LPI bit in the Network Control register.

For the receive path:

1. Wait for an interrupt to indicate that LPI has been received.
2. Disable relevant parts of the receive path if desired but keep the PCS and SerDes active.
3. Wait for an interrupt to indicate that regular idle has been received and then re-enable the receive path.

3.11. IEEE Std 802.1Qav Support – Credit Based Shaping

A credit based shaping (CBS) algorithm is available on the two highest priority queues (Queue 3 (highest priority), Queue 2) and is defined in IEEE Std 802.1Qav: Forwarding and Queuing Enhancements for Time-Sensitive Streams (FQTSS). This allows traffic on these queues to be limited and to allow other queues to transmit.

Traffic shaping is enabled via the CBS Control register. This enables a counter which stores the amount of transmit 'credit', measured in bytes that a particular queue has. A queue may only transmit if it has non-negative credit. If a queue has data to send, but is held off from doing as another queue is transmitting, then credit will accumulate in the credit counter at the rate defined in the CBS IdleSlope register for that queue (ETHERNETn_cbs_idleslope_q_a, ETHERNETn_cbs_idleslope_q_b). idleSlope is the rate of change of credit when waiting to transmit and must be less than the value of the portTransmitRate. When this queue is transmitting the credit counter is decremented at the rate of sendSlope which is defined as the portTransmitRate – idleSlope. A queue can accumulate negative credit when transmitting which will hold off any other transfers from that queue until credit returns to a non-negative value. No transfers are halted when a queue's credit becomes negative, it will accumulate negative credit until the transfer completes.

If both queues have positive credit, when the next queue to transfer is about to be selected, the queue with the most positive credit will be allowed to transfer first. The queue with the largest positive credit is the queue that had been prevented from transmitting for the longest time.

3.12. PHY Interface

The Ethernet MAC supports only MII as PHY interface.

3.13. Jumbo Frames

The jumbo frames enable bit in the Network Configuration register (bit 3) allows the Ethernet MAC in its initial state to receive jumbo frames up to 10240 bytes in size. This operation is not part of IEEE Std 802.3 specification and is normally disabled. When jumbo frames are enabled, frames received with a frame size greater than 10240 bytes are discarded.

The jumbo frames maximum length can be controlled using the Jumbo-Frame Maximum Length register.

Its initial value is 2800_h which corresponds to 10240 bytes.

Note: The Jumbo-Frame Maximum Length register must be set to greater 1536 bytes (600_h) when control bit receive_1536_byte_frames in the Network Configuration register (bit 8) is set to "1".

4. Registers

This section describes the registers of the Ethernet MAC module.

- The following registers are available for the Ethernet MAC:
  - Operation and configuration registers:

Network Control Register (ETHERNETn_network_control)
Network Configuration Register (ETHERNETn_network_configuration)
Network Status Register (ETHERNETn_network_status)
DMA Configuration Register (ETHERNETn_dma_config)
Transmit Status Register (ETHERNETn_transmit_status)
RX Buffer Queue Base Address Register (ETHERNETn_receive_q_ptr)
TX Buffer Queue Base Address Register (ETHERNETn_transmit_q_ptr)
Receive Status Register (ETHERNETn_receive_status)
Interrupt Status Register (ETHERNETn_int_status)
Interrupt Enable Register (ETHERNETn_int_enable)
Interrupt Disable Register (ETHERNETn_int_disable)
Interrupt Mask Register (ETHERNETn_int_mask)
PHY Maintenance Register (ETHERNETn_phy_management)
Receive Pause Quantum Register (ETHERNETn_pause_time)
Transmit Pause Quantum Register (ETHERNETn_tx_pause_quantum)
Transmit Pause Quantum 1 Register (ETHERNETn_tx_pause_quantum1)
Transmit Pause Quantum 2 Register (ETHERNETn_tx_pause_quantum2)
Transmit Pause Quantum 3 Register (ETHERNETn_tx_pause_quantum3)
TX Partial Store and Forward Register (ETHERNETn_pbuf_txcutthru)
RX Partial Store and Forward Register (ETHERNETn_pbuf_rxcutthru)
Jumbo-Frame Maximum Length Register (ETHERNETn_jumbo_max_length)
AXI Maximum Pipeline Register (ETHERNETn_axi_max_pipeline)
Hash Bottom Register (ETHERNETn_hash_bottom)
Hash Top Register (ETHERNETn_hash_top)
Specific Address Bottom 1 Register (ETHERNETn_spec_add_bottom_1)
Specific Address Top 1 Register (ETHERNETn_spec_add_top_1)
Specific Address Bottom 2 Register (ETHERNETn_spec_add_bottom_2)
Specific Address Top 2 Register (ETHERNETn_spec_add_top_2)
Specific Address Bottom 3 Register (ETHERNETn_spec_add_bottom_3)
Specific Address Top 3 Register (ETHERNETn_spec_add_top_3)
Specific Address Bottom 4 Register (ETHERNETn_spec_add_bottom_4)
Specific Address Top 4 Register (ETHERNETn_spec_add_top_4)
Type ID Match 1 Register (ETHERNETn_spec_type_1)
Type ID Match 2 Register (ETHERNETn_spec_type_2)
Type ID Match 3 Register (ETHERNETn_spec_type_3)
Type ID Match 4 Register (ETHERNETn_spec_type_4)
IPG Stretch Register (ETHERNETn_stretch_ratio)
Stacked VLAN Register (ETHERNETn_stacked_vlan)
Transmit PFC Pause Register (ETHERNETn_tx_pfc_pause)

Specific Address Mask 1 Bottom Register (ETHERNETn_mask_add1_bottom)

Specific Address Mask 1 Top Register (ETHERNETn_mask_add1_top

Receive DMA Data Buffer Address Mask Register (ETHERNETn_dma_addr_or_mask)

RX PTP Unicast IP Destination Address Register (ETHERNETn_rx_ptp_unicast)

TX PTP Unicast IP Destination Address Register (ETHERNETn_tx_ptp_unicast)

IEEE 1588 Timer Comparison Value Nanoseconds Register (ETHERNETn_tsu_nsec_cmp)

IEEE 1588 Timer Comparison Value Seconds Bottom Register (ETHERNETn_tsu_sec_cmp)

IEEE 1588 Timer Comparison Value Seconds Top Register (ETHERNETn_tsu_msb_sec_cmp)

PTP Event Frame Transmitted Seconds [47:32] Register (ETHERNETn_tsu_ptp_tx_msb_sec)

PTP Event Frame Received Seconds [47:32] Register (ETHERNETn_tsu_ptp_rx_msb_sec)

PTP Peer Event Frame Transmitted Seconds [47:32] Register (ETHERNETn_tsu_peer_tx_msb_sec)

PTP Peer Event Frame Received Seconds [47:32] Register (ETHERNETn_tsu_peer_rx_msb_sec)

Identification and Revision Register (ETHERNETn_revision_reg)

Statistics registers. These registers reset to zero on a read and stick at all ones when they count to their maximum value. They should be read frequently enough to prevent loss of data. The receive statistics registers are only incremented when receive enable bit (ETHERNETn_network_control[2]) is set in the Network Control register. The statistics registers optionally have a snapshot capability which, when exercised, will simultaneously store and clear the current values of all the statistics registers into a snapshot register set in order to allow a consistent set of statistics to be read by the processor. The snapshot is controlled using bit 13 of the Network Control register. The read snapshot control indicated by bit 14 of the Network Control register determines whether the processor reads the snapshot registers (logic 1) or the incrementing registers (logic 0). The default **Ethernet MAC** configuration does not support the snapshot capability. All the statistics registers are read only. For test purposes they may be written by setting bit 7 (Write Enable) in the Network Control register. Setting bit 6 (increment statistics) in the Network Control register causes all the statistics registers to increment by one, again for test purposes. Once a statistics register has been read, it is automatically cleared. The statistics register block contains the following registers:

Octets Transmitted Bottom Register (ETHERNETn_octets_txed_bottom)

Octets Transmitted Top Register (ETHERNETn_octets_txed_top)

Frames Transmitted Register (ETHERNETn_frames_txed_ok)

Broadcast Frames Transmitted Register (ETHERNETn_broadcast_txed)

Multicast Frames Transmitted Register (ETHERNETn_multicast_txed)

Pause Frames Transmitted Register (ETHERNETn_pause_frames_txed)

64 Byte Frames Transmitted Register (ETHERNETn_frames_txed_64)

65 To 127 Byte Frames Transmitted Register (ETHERNETn_frames_txed_65)

128 To 255 Byte Frames Transmitted Register (ETHERNETn_frames_txed_128)

256 To 511 Byte Frames Transmitted Register (ETHERNETn_frames_txed_256)

512 To 1023 Byte Frames Transmitted Register (ETHERNETn_frames_txed_512)

1024 To 1518 Byte Frames Transmitted Register (ETHERNETn_frames_txed_1024)

Greater Than 1518 Byte Frames Transmitted Register (ETHERNETn_frames_txed_1519)

Transmit Under Runs Register (ETHERNETn_tx_underruns)

Single Collision Frames Register (ETHERNETn_single_collisions)

Multiple Collision Frames Register (ETHERNETn_multiple_collisions)

Excessive Collisions Register (ETHERNETn_excessive_collisions)

Late Collisions Register (ETHERNETn_late_collisions)

Deferred Transmission Frames Register (ETHERNETn_deferred_frames)

Carrier Sense Errors Register (ETHERNETn_crs_errors)

Octets Receive Bottom Register (ETHERNETn_octets_rxed_bottom)

Octets Receive TOP Register (ETHERNETn_octets_rxed_top)
Frames Received Register (ETHERNETn_frames_rxed_ok)
Broadcast Frames Received Register (ETHERNETn_broadcast_rxed)
Multicast Frames Received Register (ETHERNETn_multicast_rxed)
Pause Frames Received Register (ETHERNETn_pause_frames_rxed)
64 Byte Frames Received Register (ETHERNETn_frames_rxed_64)
65 To 127 Byte Frames Received Register (ETHERNETn_frames_rxed_65)
128 To 255 Byte Frames Received Register (ETHERNETn_frames_rxed_128)
256 To 511 Byte Frames Received Register (ETHERNETn_frames_rxed_256)
512 To 1023 Byte Frames Received Register (ETHERNETn_frames_rxed_512)
1024 To 1518 Byte Frames Received Register (ETHERNETn_frames_rxed_1024)
Greater than 1518 Byte Frames Received Register (ETHERNETn_frames_rxed_1519)
Undersized Frames Received Register (ETHERNETn_undersized_frames)
Oversize Frames Received Register (ETHERNETn_excessive_rx_length)
Jabbers Received Register (ETHERNETn_rx_jabbers)
Frame Check Sequence Errors Register (ETHERNETn_fcs_errors)
Length Field Frame Errors Register (ETHERNETn_rx_length_errors)
Receive Symbol_Errors Register (ETHERNETn_rx_symbol_errors)
Alignment Errors Register (ETHERNETn_alignment_errors)
Receive Resource Errors Register (ETHERNETn_rx_resource_errors)
Receive Over Runs Register (ETHERNETn_rx_overruns)
IP Header Checksum Errors Register (ETHERNETn_rx_ip_ck_errors)
TCP Checksum Errors Register (ETHERNETn_rx_tcp_ck_errors)
UDP Checksum Errors Register (ETHERNETn_rx_udp_ck_errors)
Receive DMA Flushed Packets Register (ETHERNETn_auto_flushed_pkts)
------ end of statistics registers
IEEE 1588 Timer Increment Sub Nanoseconds Register (ETHERNETn_tsu_timer_incr_sub_nsec)
IEEE 1588 Timer Seconds [47:32] Register (ETHERNETn_tsu_timer_msb_sec)
IEEE 1588 Timer Seconds [31:0] Register (ETHERNETn_tsu_timer_sec)
IEEE 1588 Timer Nanoseconds Register (ETHERNETn_tsu_timer_nsec)
IEEE 1588 Timer Adjust Register (ETHERNETn_tsu_timer_adjust)
IEEE 1588 Timer Increment Register (ETHERNETn_tsu_timer_incr)
PTP Event Frame Transmitted Seconds [31:0] Register (ETHERNETn_tsu_ptp_tx_sec)
PTP Event Frame Transmitted Nanoseconds Register (ETHERNETn_tsu_ptp_tx_nsec)
PTP Event Frame Received Seconds [31:0] Register (ETHERNETn_tsu_ptp_rx_sec)
PTP Event Frame Received Nanoseconds Register (ETHERNETn_tsu_ptp_rx_nsec)
PTP Peer Event Frame Transmitted Seconds [31:0] Register (ETHERNETn_tsu_peer_tx_sec)
PTP Peer Event Frame Transmitted Nanoseconds Register (ETHERNETn_tsu_peer_tx_nsec)
PTP Peer Event Frame Received Seconds [31:0] Register (ETHERNETn_tsu_peer_rx_sec)
PTP Peer Event Frame Received Nanoseconds Register (ETHERNETn_tsu_peer_rx_nsec)
Receive LPI Transitions Register (ETHERNETn_rx_lpi)
Received LPI Time Register (ETHERNETn_rx_lpi_time)
Transmit LPI Transitions Register (ETHERNETn_tx_lpi)
Transmit LPI Time Register (ETHERNETn_tx_lpi_time)

Interrupt Status Queue 1 Status Register (ETHERNETn_int_status_q1)
Interrupt Status Queue 2 Status Register (ETHERNETn_int_status_q2)
Interrupt Status Queue 3 Status Register (ETHERNETn_int_status_q3)
TX Buffer Queue 1 Base Address Register (ETHERNETn_transmit_q1_ptr)
TX Buffer Queue 2 Base Address Register (ETHERNETn_transmit_q2_ptr)
TX Buffer Queue 3 Base Address Register (ETHERNETn_transmit_q3_ptr)
RX Buffer Queue 1 Base Address Register (ETHERNETn_receive_q1_ptr)
RX Buffer Queue 2 Base Address Register (ETHERNETn_receive_q2_ptr)
RX Buffer Queue 3 Base Address Register (ETHERNETn_receive_q3_ptr)
RX Buffer Size Queue 1 Register (ETHERNETn_rxbuf_size_q1)
RX Buffer Size Queue 2 Register (ETHERNETn_rxbuf_size_q2)
RX Buffer Size Queue 3 Register (ETHERNETn_rxbuf_size_q3)
CBS Control Register (ETHERNETn_cbs_control)
CBS IdleSlope Queue A Register (ETHERNETn_cbs_idleslope_q_a)
CBS IdleSlope Queue B Register (ETHERNETn_cbs_idleslope_q_b)
MSB Buffer Queue Base Address Register (ETHERNETn_msb_buff_q_base_addr_reg)
TX BD Control Register (ETHERNETn_tx_bd_control)
RX BD Control Register (ETHERNETn_rx_bd_control)
Screening Type 1 Register 0 (ETHERNETn_screening_type_1_register_0)
Screening Type 1 Register 1 (ETHERNETn_screening_type_1_register_1)
Screening Type 1 Register 2 (ETHERNETn_screening_type_1_register_2)
Screening Type 1 Register 3 (ETHERNETn_screening_type_1_register_3)
Screening Type 1 Register 4 (ETHERNETn_screening_type_1_register_4)
Screening Type 1 Register 5 (ETHERNETn_screening_type_1_register_5)
Screening Type 1 Register 6 (ETHERNETn_screening_type_1_register_6)
Screening Type 1 Register 7 (ETHERNETn_screening_type_1_register_7)
Screening Type 1 Register 8 (ETHERNETn_screening_type_1_register_8)
Screening Type 1 Register 9 (ETHERNETn_screening_type_1_register_9)
Screening Type 1 Register 10 (ETHERNETn_screening_type_1_register_10)
Screening Type 1 Register 11 (ETHERNETn_screening_type_1_register_11)
Screening Type 1 Register 12 (ETHERNETn_screening_type_1_register_12)
Screening Type 1 Register 13 (ETHERNETn_screening_type_1_register_13)
Screening Type 1 Register 14 (ETHERNETn_screening_type_1_register_14)
Screening Type 1 Register 15 (ETHERNETn_screening_type_1_register_15)
Screening Type 2 Register 0 (ETHERNETn_screening_type_2_register_0)
Screening Type 2 Register 1 (ETHERNETn_screening_type_2_register_1)
Screening Type 2 Register 2 (ETHERNETn_screening_type_2_register_2)
Screening Type 2 Register 3 (ETHERNETn_screening_type_2_register_3)
Screening Type 2 Register 4 (ETHERNETn_screening_type_2_register_4)
Screening Type 2 Register 5 (ETHERNETn_screening_type_2_register_5)
Screening Type 2 Register 6 (ETHERNETn_screening_type_2_register_6)
Screening Type 2 Register 7 (ETHERNETn_screening_type_2_register_7)
Screening Type 2 Register 8 (ETHERNETn_screening_type_2_register_8)
Screening Type 2 Register 9 (ETHERNETn_screening_type_2_register_9)

Screening Type 2 Register 10 (ETHERNETn_screening_type_2_register_10)
Screening Type 2 Register 11 (ETHERNETn_screening_type_2_register_11)
Screening Type 2 Register 12 (ETHERNETn_screening_type_2_register_12)
Screening Type 2 Register 13 (ETHERNETn_screening_type_2_register_13)
Screening Type 2 Register 14 (ETHERNETn_screening_type_2_register_ETHERNETn_screening_type_2_register_14)
Screening Type 2 Register 15 (ETHERNETn_screening_type_2_register_15)
Interrupt Enable Queue 1 Register (ETHERNETn_int_enable_q1)
Interrupt Enable Queue 2 Register (ETHERNETn_int_enable_q2)
Interrupt Enable Queue 3 Register (ETHERNETn_int_enable_q3)
Interrupt Disable Queue 1 Register (ETHERNETn_int_disable_q1)
Interrupt Disable Queue 2 Register (ETHERNETn_int_disable_q2)
Interrupt Disable Queue 3 Register (ETHERNETn_int_disable_q3)
Interrupt Mask Queue 1 Register (ETHERNETn_int_mask_q1)
Interrupt Mask Queue 2 Register (ETHERNETn_int_mask_q2)
Interrupt Mask Queue 3 Register (ETHERNETn_int_mask_q3)
EtherType 2 Register 0 (ETHERNETn_screening_type_2_ethertype_reg_0)
EtherType 2 Register 1 (ETHERNETn_screening_type_2_ethertype_reg_1)
EtherType 2 Register 2 (ETHERNETn_screening_type_2_ethertype_reg_2)
EtherType 2 Register 3 (ETHERNETn_screening_type_2_ethertype_reg_3)
EtherType 2 Register 4 (ETHERNETn_screening_type_2_ethertype_reg_4)
EtherType 2 Register 5 (ETHERNETn_screening_type_2_ethertype_reg_5)
EtherType 2 Register 6 (ETHERNETn_screening_type_2_ethertype_reg_6)
EtherType 2 Register 7 (ETHERNETn_screening_type_2_ethertype_reg_7)
Type2 Compare Word 0 0 Register (ETHERNETn_type2_compare_0_word_0)
Type2 Compare Word 1 0 Register (ETHERNETn_type2_compare_0_word_1)
Type2 Compare Word 0 1 Register (ETHERNETn_type2_compare_1_word_0)
Type2 Compare Word 1 1 Register (ETHERNETn_type2_compare_1_word_1)
Type2 Compare Word 0 2 Register (ETHERNETn_type2_compare_2_word_0)
Type2 Compare Word 1 2 Register (ETHERNETn_type2_compare_2_word_1)
Type2 Compare Word 0 3 Register (ETHERNETn_type2_compare_3_word_0)
Type2 Compare Word 1 3 Register (ETHERNETn_type2_compare_3_word_1)
Type2 Compare Word 0 4 Register (ETHERNETn_type2_compare_4_word_0)
Type2 Compare Word 1 4 Register (ETHERNETn_type2_compare_4_word_1)
Type2 Compare Word 0 5 Register (ETHERNETn_type2_compare_5_word_0)
Type2 Compare Word 1 5 Register (ETHERNETn_type2_compare_5_word_1)
Type2 Compare Word 0 6 Register (ETHERNETn_type2_compare_6_word_0)
Type2 Compare Word 1 6 Register (ETHERNETn_type2_compare_6_word_1)
Type2 Compare Word 0 7 Register (ETHERNETn_type2_compare_7_word_0)
Type2 Compare Word 1 7 Register (ETHERNETn_type2_compare_7_word_1)
Type2 Compare Word 0 8 Register (ETHERNETn_type2_compare_8_word_0)
Type2 Compare Word 1 8 Register (ETHERNETn_type2_compare_8_word_1)
Type2 Compare Word 0 9 Register (ETHERNETn_type2_compare_9_word_0)
Type2 Compare Word 1 9 Register (ETHERNETn_type2_compare_9_word_1)

Type2 Compare Word 0 10 Register (ETHERNETn_type2_compare_10_word_0)
Type2 Compare Word 1 10 Register (ETHERNETn_type2_compare_10_word_1)
Type2 Compare Word 0 11 Register (ETHERNETn_type2_compare_11_word_0)
Type2 Compare Word 1 11 Register (ETHERNETn_type2_compare_11_word_1)
Type2 Compare Word 0 12 Register (ETHERNETn_type2_compare_12_word_0)
Type2 Compare Word 1 12 Register (ETHERNETn_type2_compare_12_word_1)
Type2 Compare Word 0 13 Register (ETHERNETn_type2_compare_13_word_0)
Type2 Compare Word 1 13 Register (ETHERNETn_type2_compare_13_word_1)
Type2 Compare Word 0 14 Register (ETHERNETn_type2_compare_14_word_0)
Type2 Compare Word 1 14 Register (ETHERNETn_type2_compare_14_word_1)
Type2 Compare Word 0 15 Register (ETHERNETn_type2_compare_15_word_0)
Type2 Compare Word 1 15 Register (ETHERNETn_type2_compare_15_word_1)
Type2 Compare Word 0 16 Register (ETHERNETn_type2_compare_16_word_0)
Type2 Compare Word 1 16 Register (ETHERNETn_type2_compare_16_word_1)
Type2 Compare Word 0 17 Register (ETHERNETn_type2_compare_17_word_0)
Type2 Compare Word 1 17 Register (ETHERNETn_type2_compare_17_word_1)
Type2 Compare Word 0 18 Register (ETHERNETn_type2_compare_18_word_0)
Type2 Compare Word 1 18 Register (ETHERNETn_type2_compare_18_word_1)
Type2 Compare Word 0 19 Register (ETHERNETn_type2_compare_19_word_0)
Type2 Compare Word 1 19 Register (ETHERNETn_type2_compare_19_word_1)
Type2 Compare Word 0 20 Register (ETHERNETn_type2_compare_20_word_0)
Type2 Compare Word 1 20 Register (ETHERNETn_type2_compare_20_word_1)
Type2 Compare Word 0 21 Register (ETHERNETn_type2_compare_21_word_0)
Type2 Compare Word 1 21 Register (ETHERNETn_type2_compare_21_word_1)
Type2 Compare Word 0 22 Register (ETHERNETn_type2_compare_22_word_0)
Type2 Compare Word 1 22 Register (ETHERNETn_type2_compare_22_word_1)
Type2 Compare Word 0 23 Register (ETHERNETn_type2_compare_23_word_0)
Type2 Compare Word 1 23 Register (ETHERNETn_type2_compare_23_word_1)
Type2 Compare Word 0 24 Register (ETHERNETn_type2_compare_24_word_0)
Type2 Compare Word 1 24 Register (ETHERNETn_type2_compare_24_word_1)
Type2 Compare Word 0 25 Register (ETHERNETn_type2_compare_25_word_0)
Type2 Compare Word 1 25 Register (ETHERNETn_type2_compare_25_word_1)
Type2 Compare Word 0 26 Register (ETHERNETn_type2_compare_26_word_0)
Type2 Compare Word 1 26 Register (ETHERNETn_type2_compare_26_word_1)
Type2 Compare Word 0 27 Register (ETHERNETn_type2_compare_27_word_0)
Type2 Compare Word 1 27 Register (ETHERNETn_type2_compare_27_word_1)
Type2 Compare Word 0 28 Register (ETHERNETn_type2_compare_28_word_0)
Type2 Compare Word 1 28 Register (ETHERNETn_type2_compare_28_word_1)
Type2 Compare Word 0 29 Register (ETHERNETn_type2_compare_29_word_0)
Type2 Compare Word 1 29 Register (ETHERNETn_type2_compare_29_word_1)
Type2 Compare Word 0 30 Register (ETHERNETn_type2_compare_30_word_0)
Type2 Compare Word 1 30 Register (ETHERNETn_type2_compare_30_word_1)
Type2 Compare Word 0 31 Register (ETHERNETn_type2_compare_31_word_0)
Type2 Compare Word 1 31 Register (ETHERNETn_type2_compare_31_word_1)

4.1. Network Control Register (ETHERNETn_network_control)

Description of Network Control register is shown. The Network Control register contains general Ethernet MAC control functions for receiver and transmitter.

REGISTER_NAME	ETHERNETn_network_control
OFFSET	0x000
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved						pfc_ctrl	one_step_sync_mode
ACCESS_TYPE	R0,WX						R/W	R/W
PROT_TYPE	Wp							
INITIAL_VALUE	0x00						0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved	store_udp_offset	Reserved	ptp_unicast_ena	tx_lpi_en	flush_rx_pkt_clk	transmit_pfc_priority_based_pause_frame	pfc_enable
ACCESS_TYPE	R0,WX	R/W	R/W0	R/W	R/W	R0,W	R0,W1S	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	store_rx_ts	stats_read_snap	stats_take_snap	tx_pause_frame_zero	tx_pause_frame_req	tx_halt_clk	tx_start_clk	back_pressure
ACCESS_TYPE	R/W	R/W	R0,W	R0,W1S	R0,W1S	R0,W	R0,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	stats_write_en	inc_all_stats_regs	clear_all_stats_regs	man_port_en	enable_transmit	enable_receive	Reserved	
ACCESS_TYPE	R/W	R0,W	R0,W1S	R/W	R/W	R/W	R/W0	
PROT_TYPE	Wp							
INITIAL_VALUE	0	0	0	0	0	0	0x0	

[bit31:26] Reserved

Always read "0". Writing has no effect.

[bit25] pfc_ctrl: PFC control

Enable multiple PFC pause quanta, one per pause priority

Bit	Description
0	Disable
1	Enable

[bit24] one_step_sync_mode: IEEE 1588 Timer One Step Sync Mode

Replace timestamp field in the IEEE 1588 header for TX Sync Frames with current IEEE 1588 Timer value.

Bit	Description
0	Disable
1	Enable

[bit23] Reserved
Always read "0". Writing has no effect.

[bit22] store_udp_offset: Store UDP/TCP Offset
Store UDP/TCP offset to memory.

Bit	Description
0	Normal operation.
1	The upper 16-bits of the CRC of every received frame are replaced with the offset from start of frame to the beginning of the UDP or TCP header. The lower 16-bits of the CRC are replaced with zero and reserved for future use.

[bit21] Reserved
Always read write value. Always write “0”.

[bit20] ptp_unicast_ena: PTP Unicast Frames Enable
Enable the detection of PTP unicast frames.

Bit	Description
0	Disable
1	Enable

[bit19] tx_lpi_en: LPI Transmission Enable
Enable immediate transmission of LPI (low power idle).

Bit	Description
0	Disable
1	Enable

[bit18] flush_rx_pkt_clk: Flush Next Packet From External RX DPRAM

Bit	Description
0	No effect
1	Writing this bit to "1" will only have an effect if the DMA is currently not reading a packet stored in RX DPRAM to memory.

[bit17] transmit_pfc_priority_based_pause_frame:

Bit	Description
0	No effect
1	Transmit PFC priority based pause frame. Takes the values stored in the Transmit PFC Pause register. Cleared by hardware.

[bit16] pfc_enable: Enable PFC Priority Based Pause Reception

Bit	Description
0	Disable
1	Enable PFC negotiation and recognition of priority based pause frames..

[bit15] store_rx_ts: Store receive time stamp

Bit	Description
0	Normal operation
1	Setting this bit to one will cause the CRC of every received frame to be replaced with the value of the nanoseconds field of the IEEE 1588 Timer that was captured as the receive frame passed the message time stamp point.

[bit14] stats_read_snap: Read snapshot

Bit	Description
0	The current value of the statistics register will be read back.
1	The snapshot value of the statistics register will be read back.

[bit13] stats_take_snap: Take snapshot

Bit	Description
0	No effect
1	Record the current value of all statistics registers in the snapshot registers and clear the statistics registers. Cleared by hardware.

[bit12] tx_pause_frame_zero: Transmit zero quantum pause frame

Bit	Description
0	No effect
1	Transmit pause frame with zero quantum. Cleared by hardware.

[bit11] tx_pause_frame_req: Transmit pause frame

Bit	Description
0	No effect
1	Transmit pause frame. Cleared by hardware.

[bit10] tx_halt_clk: Transmit halt

Bit	Description
0	No effect
1	Halts transmission as soon as any ongoing frame transmission ends. Cleared by hardware.

[bit9] tx_start_clk: Transmit start

Bit	Description
0	No effect
1	Starts transmission. Cleared by hardware.

[bit8] back_pressure: Back pressure

Bit	Description
0	No effect
1	Will force collisions on all received frames in 10 M of 100 M half duplex mode.

[bit7] stats_write_en: Write enable for statistics register

Bit	Description
0	Normal operation.
1	Statistics register can be written for functional test purposes.

[bit6] inc_all_stats_regs: Increment statistics register

Bit	Description
0	No effect
1	Increment all statistics registers by 1 for functional test purposes. Cleared by hardware.

[bit5] clear_all_stats_regs: Clear statistics register

Bit	Description
0	No effect
1	Clears all statistics registers. Cleared by hardware.

[bit4] man_port_en: Management port enable

Bit	Description
0	Disable management ports. MDIO is forced to High-Z and MDC to "0".
1	Enable management ports.

[bit3] enable_transmit: Transmit enable

Bit	Description
0	Transmission will stop immediately, the transmit pipeline and control registers will be cleared and the transmit queue pointer registers (TX Buffer Queue Base Address, TX Buffer Queue 1 Base Address, TX Buffer Queue 2 Base Address, and TX Buffer Queue 3 Base Address) will be reset to point to the start of the transmit descriptor list.
1	Enable the Ethernet MAC to send data.

[bit2] enable_receive: Receive enable

Bit	Description
0	Frame reception will stop immediately and the receive pipeline will be cleared. The receive queue pointer register (ETHERNETn_receive_q_ptr) is unaffected.
1	Enable the Ethernet MAC to receive data.

[bit1:0] Reserved

Always read write value. Always write "0".

4.2. Network Configuration Register (ETHERNETn_network_configuration)

Description of Network Configuration register is shown.

REGISTER_NAME	ETHERNETn_network_configuration
OFFSET	0x004
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved	ignore_ipg_rx_er	nsp_change	ipg_stretch_enable	Reserved	ignore_rx_fcs	Reserved	receive_checksum_offload_enable
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W0	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	disable_copy_of_pause_frames	data_bus_width[1:0]		mdc_clock_division[2:0]			fcs_remove	length_field_error_frame_discard
ACCESS_TYPE	R/W	R/W		R/W			R/W	R/W
PROT_TYPE	Wp							
INITIAL_VALUE	0	0x1		0x2			0	0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	receive_buffer_offset[1:0]		pause_enable	retry_test	Reserved	Reserved	Reserved	receive_1536_byte_frames
ACCESS_TYPE	R/W		R/W	R/W	R/W	R/W0	R/W0	R/W
PROT_TYPE	Wp							
INITIAL_VALUE	0x0		0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	unicast_hash_enable	multicast_hash_enable	no_broadcast	copy_all_frames	jumbo_frames	discard_non_vlan_frames	full_duplex	speed
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] Reserved

Always read write value. Always write “0”.

[bit30] ignore_ipg_rx_er: Ignore IPG RX_ER

Bit	Description
0	Ethernet MAC operation is affected by input signal RX_ER when RX_DV is low.
1	Ethernet MAC operation is not affected by input signal RX_ER when RX_DV is low.

[bit29] nsp_change: Receive bad preamble

Bit	Description
0	Frames with non-standard preamble are rejected.
1	Frames with non-standard preamble are not rejected.

[bit28] ipg_stretch_enable: IPG stretch enable

Bit	Description
0	Disable
1	The transmit IPG can be increased above 96 bit times depending on the previous frame length using the IPG Stretch register.

[bit27] Reserved

Always read write value. Always write “0”.

[bit26] ignore_rx_fcs: Ignore RX FCS

Bit	Description
0	Normal operation.
1	Frames with FCS/CRC errors will not be rejected. FCS error statistics will still be collected for frames with bad FCS and FCS status will be recorded in frame's DMA descriptor.

[bit25] Reserved

Read always returns written value. Write always "0".

[bit24] receive_checksum_offload_enable: Receive checksum offload enable

Bit	Description
0	Disable
1	The receive checksum engine is enabled. Frames with bad IP, TCP or UDP checksums are discarded.

[bit23] disable_copy_of_pause_frames: Disable copy of pause frames

Bit	Description
0	Disable
1	Set to "1" to prevent valid pause frames from being copied to memory. When set, pause frames are not copied to memory regardless of the state of the copy_all_frames bit; whether a hash match is found or whether a Type ID match is identified. If a destination address match is found the pause frame will be copied to memory. Note that valid pause frames received will still increment pause statistics and pause the transmission of frames as required.

[bit22:21] data_bus_width: Data bus width

Bits	Description
00	Reserved
01	64-bit AMBA AXI data bus width
10	Reserved
11	Reserved

[bit20:18] mdc_clock_division: MDC clock division

Set according to CLK_HAPP1B0 speed. These three bits determine the number, CLK_HAPP1B0 will be divided by to generate MDC. MDC must not exceed 2.5 MHz (MDC is only active during MDIO read and write operations)

Bits	Description
000	divide by 8
001	divide by 16
010	divide by 32
011	divide by 48
100	divide by 64
101	divide by 96
110	divide by 128
111	divide by 224

[bit17] fcs_remove: FCS remove

Setting this bit will cause received frames to be written to memory without their frame check sequence (last 4 bytes). The frame length indicated will be reduced by four bytes in this mode.

Bit	Description
0	Disable
1	Enable

[bit16] length_field_error_frame_discard: Length field error frame discard

Setting this bit causes frames with a measured length shorter than the extracted length field (as indicated by bytes 13 and 14 in a non-VLAN tagged frame) to be discarded. This only applies to frames with a length field less than 0600_h (1536_d).

Bit	Description
0	Disable
1	Enable

[bit15:14] receive_buffer_offset: Receive buffer offset

Indicates the number of bytes by which the received data is offset from the start of the receive buffer.

Bits	Description
00	no offset
01	one byte offset
10	two byte offset
11	three byte offset

[bit13] pause_enable: Pause enable

Bit	Description
0	Disable
1	Transmission will pause if a non zero 802.3 classic pause frame is received and PFC has not been negotiated.

[bit12] retry_test: Retry test

Bit	Description
0	Normal operation.
1	Backoff between collisions will always be one slot time. Setting this bit to "1" helps test the "too many retries" condition. Also used in the pause frame tests to reduce the pause counter's decrement time from 512 bit times, to every `RX_CLK` cycle.

[bit11] Reserved
Always read write value. Always write “0”.

[bit10:9] Reserved
Read always returns written value. Write always "0".

[bit8] receive_1536_byte_frames: Receive 1536 byte frames

Bit	Description
0	Ethernet MAC rejects frames above 1518 bytes in length.
1	Ethernet MAC accepts frames up to 1536 bytes in length.

[bit7] unicast_hash_enable: Unicast hash enable

Bit	Description
0	Unicast frames will not be accepted.
1	Unicast frames will be accepted when the 6-bit hash function of the destination address points to a bit that is set in the Hash register

[bit6] multicast_hash_enable: Multicast hash enable

Bit	Description
0	Multicast frames will not be accepted.
1	Multicast frames will be accepted when the 6-bit hash function of the destination address points to a bit that is set in the Hash register

[bit5] no_broadcast: No broadcast

Bit	Description
0	Frames addressed to the broadcast address of all ones will be accepted.
1	Frames addressed to the broadcast address of all ones will not be accepted.

[bit4] copy_all_frames: Copy all frames

Bit	Description
0	Only Unicast, Multicast, and Broadcast frames if enabled or frames which match Specific Address 1 to 4 are accepted.
1	All valid frames will be accepted.

[bit3] jumbo_frames: Jumbo frames

Bit	Description
0	Reject jumbo frames.
1	Accept jumbo frames up to 10240 bytes.

[bit2] discard_non_vlan_frames: Discard non-VLAN frames

Bit	Description
0	Non VLAN tagged frames will be discarded.
1	Only VLAN tagged frames will be passed to the address matching logic.

[bit1] full_duplex: Full duplex

Bit	Description
0	Setting not allowed.
1	Full duplex mode. The transmit block ignores the state of state of collision and carrier sense and allows receive while transmitting.

[bit0] speed: Speed

Bit	Description
0	Setting not allowed.
1	100 Mbps operation.

4.3. Network StatusNetwork Status Register (ETHERNETn_network_status)

Description of Network Status register is shown. The register shows status information with respect to the PHY management interface.

REGISTER_NAME	ETHERNETn_network_status
OFFSET	0x008
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	lpi_indicate_clk	pfc_negotiate_clk	Reserved	Reserved	Reserved	man_done	mdio_in	Reserved
ACCESS_TYPE	R,WX	R,WX	R0,WX	R0,WX	R0,WX	R,WX	R,WX	R0,WX
PROT_TYPE	Wp							
INITIAL_VALUE	0	0	0	0	0	1	0	0

[bit31:8] Reserved
Always read "0". Writing has no effect

[bit7] lpi_indicate_clk: LPI indication
Low-power idle has been detected on receive. This bit is set when LPI is detected and reset when normal idle is detected. An interrupt is generated when the state of this bit is changed.

[bit6] pfc_negotiate_clk: PFC negotiated
This bit is set when PFC Priority Based Pause has been negotiated.

[bit5:3] Reserved
Always read "0". Writing has no effect

[bit2] man_done: Management done
When this bit is set it means PHY management logic is idle (i.e. has completed).

[bit1] mdio_in: MDIO_IN status
This bit returns the status of the mdio_in pin.

[bit0] Reserved
Always read "0". Writing has no effect

4.4. DMA Configuration Register (ETHERNETn_dma_config)

Description of DMA Configuration register is shown.

REGISTER_NAME	ETHERNETn_dma_config
OFFSET	0x010
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved	dma_addr_bus_width	tx_bd_extended_mode_en	rx_bd_extended_mode_en	Reserved	force_max_amba_burst_tx	force_max_amba_burst_rx	force_discard_on_err
ACCESS_TYPE	R0,WX	R/W	R/W	R/W	R0,WX	R/W	R/W0	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	rx_buf_size[7:0]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0xF0							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved				tx_pbuf_tcp_en	tx_pbuf_size	rx_pbuf_size[1:0]	
ACCESS_TYPE	R0,WX				R/W	R/W	R/W	
PROT_TYPE	Wp							
INITIAL_VALUE	0x0				0	1	0x3	

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved		Reserved	amba_burst_length[4:0]				
ACCESS_TYPE	R0,W0		R0,WX	R/W				
PROT_TYPE	Wp							
INITIAL_VALUE	0		0	0x04				

[bit31] Reserved

Always read "0". Writing has no effect.

[bit30] dma_addr_bus_width: DMA address bus width

Bit	Description
0	32-bit DMA address bus width.
1	Setting not allowed.

[bit29] tx_bd_extended_mode_en: Enable TX extended BD mode

See TX BD Control register (ETHERNETn_tx_bd_control) for description of feature.

Bit	Description
0	Disable
1	Enable

[bit28] rx_bd_extended_mode_en: Enable RX extended BD mode

See RX BD Control register (ETHERNETn_rx_bd_control) for description of feature.

Bit	Description
0	Disable
1	Enable

[bit27] Reserved

Always read "0". Writing has no effect.

[bit26] force_max_amba_burst_tx: Force maximum length bursts on TX

Bit	Description
0	Maximum length bursts are not forced.
1	Force maximum length bursts on TX. Force the TX DMA to always issue maximum length bursts on EOP (end of packet) or EOB (end of buffer) transfers as defined by bits [4:0] of this register, even when there is less than maximum burst data bytes to read. Residual data read is ignored. Does not apply on bursts that crosses 1 K boundary.

[bit25] force_max_amba_burst_rx: Force maximum length bursts on RX

Bit	Description
0	Maximum length bursts are not forced.
1	Force max length bursts on RX. Force the RX DMA to always issue maximum length bursts on EOP (end of packet) or EOB (end of buffer) transfers, even if there is less than maximum burst packet data required to be written. Any extra bytes of pad-data are set to 0x00. Does not apply on bursts that crosses 1 K boundary.

[bit24] force_discard_on_err: Auto discard RX packets during lack of resource

A write to this bit is ignored if the DMA is not configured in the packet buffer full store and forward mode.

Bit	Description
0	Received packets will remain to be stored in the packet buffer until AXI buffer resource becomes available.
1	The Ethernet MAC DMA will automatically discard receive packets from the RX Packet Buffer Memory when no AXI resource is available.

[bit23:16] rx_buf_size: DMA receive buffer size in system memory

The value defined by these bits determines the size of the buffer to use in system memory when writing received data. The value is defined in multiples of 64 bytes.

Bits	Description
0x00	Setting not allowed
0x01	64 Byte
0x02	128 Byte (2 * 64 Byte)
. . .	. . .
0x18	1536 Byte (24 * 64 Byte) one maximum length frame/buffer
. . .	. . .
0xA0	10240 Byte (160 * 64 Byte) one jumbo frame/buffer
. . .	. . .
0xFF	16320 Byte (255 * 64 Byte)

[bit15:12] Reserved

Always read "0". Writing has no effect.

[bit11] tx_pbuf_tcp_en: Transmitter IP, TCP and UDP checksum generation offload enable

Note: Not supported if DMA is configured in TX Partial Store and Forward Mode (i.e. ETHERNETn_pbuf_txcutthru[31] = 0).

Bit	Description
0	Transmit frame data is unaffected.
1	Transmitter checksum generation engine is enabled, to calculate and substitute checksums for transmit frames.

[bit10] tx_pbuf_size: Transmitter packet buffer memory size select

This bit selects the amount of memory used for the transmit packet buffer. The reset value of "1" represents the maximum available TX Packet Buffer Memory size of 16 Kbytes.

Bit	Description
0	Do not use top address bit (8 Kbytes)
1	Use full configured addressable space (16 Kbytes)

[bit9:8] rx_pbuf_size: Receiver packet buffer memory size select

These bits select the amount of memory used for the receive packet buffer. The reset value of "0x3" represents the maximum available RX Packet Buffer Memory size of 4 Kbytes.

Bits	Description
00	Do not use top three address bits (512 bytes)
01	Do not use top two address bits (1 Kbytes)
10	Do not use top address bit (2 Kbytes)
11	Use full configure addressable space (4 Kbytes)

[bit7:6] Reserved

Always read "0". Always write “0”.

[bit5] Reserved

Always read "0". Writing has no effect.

[bit4:0] amba_burst_length: AMBA burst length

Selects the attempted burst length used on the AMBA AXI when transferring frame data. Not used for DMA management operations and only used where space and data size allow and respecting AXI burst boundary rules.

Bits	Description
1xxxx	Attempt to use bursts with up to 16 data transfers
01xxx	Attempt to use bursts with up to 8 data transfers
001xx	Attempt to use bursts with up to 4 data transfers
0001x	Always use SINGLE data transfers
00001	Always use SINGLE data transfers
00000	The core will optimally decide the best AXI burst length with up to 16 data transfers if possible, whilst respecting the 4 KB boundary restriction.

4.5. Transmit Status Register (ETHERNETn_transmit_status)

Description of Transmit Status register is shown.

REGISTER_NAME	ETHERNETn_transmit_status
OFFSET	0x014
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved							resp_not_ok
ACCESS_TYPE	R0,WX							R,W1C
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	late_collision_occurred	transmit_under_run	transmit_complete	amba_error	transmit_go	retry_limit_exceeded	collision_occurred	used_bit_read
ACCESS_TYPE	R,W1C	R,W1C	R,W1C	R,W1C	R,WX	R,W1C	R,W1C	R,W1C
PROT_TYPE	Wp							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:9] Reserved

Always read "0". Writing has no effect

[bit8] resp_not_ok: Response not OK

Set to "1" when the Ethernet MAC DMA sees BRESP not OK. Cleared by writing a "1" to this bit.

[bit7] late_collision_occurred: Late collision occurred

Only set.if the condition occurs in gigabit mode, as retry is not attempted. Cleared by writing a one to this bit.

[bit6] transmit_under_run: Transmit under run

This bit is set if the transmitter was forced to terminate a frame that it had already began transmitting due to further data being unavailable. This bit is set if a transmitter status write back has not completed when another status write back is attempted. When using the DMA interface configured for internal FIFO mode, this bit is also set when the transmit DMA has written the SOP data into the FIFO and either the AHB bus was not granted in time for further data, or because an AHB not OK response was returned, or because a used bit was read. When using the DMA interface configured for packet buffer mode, this bit will never be set. When using the external FIFO interface, this bit is also set when the tx_r_underflow input is asserted during a frame transfer. Cleared by writing a 1.

[bit5] transmit_complete: Transmit complete

This bit is set when a frame has been transmitted. Cleared by writing “1” to this bit.

[bit4] amba_error: Transmit frame corruption due to AMBA (AXI) errors
Set if an error occurs whilst midway through reading transmit frame from system memory including RRESP or BRESP errors and buffers exhausted mid frame (if the buffers run out during transmission of a frame then transmission stops, FCS shall be bad and tx_er asserted). Also set in DMA packet buffer mode if single frame is too large for configured packet buffer memory size. Cleared by writing “1” to this bit.

[bit3] transmit_go: Transmit ongoing
While transmit is ongoing this bit reads as “1”. Writing has no effect.

[bit2] retry_limit_exceeded: Retry limit exceeded
Cleared by writing a one to this bit.

[bit1] collision_occurred: Collision occurred
Set by the assertion of collision. Cleared by writing a one to this bit. When operating in 10/100 mode, this status indicates either a collision or a late collision. In gigabit mode, this status is not set for a late collision.

[bit0] used_bit_read: Used bit read
Used bit read Set when a transmit buffer descriptor is read with its used bit set. Cleared by writing a one to this bit.

4.6. RX Buffer Queue Base Address Register (ETHERNETn_receive_q_ptr)

This register holds the start address of the receive buffer queue (receive buffers descriptor list). The RX Buffer Queue Base Address register must be initialized before receive is enabled through bit 2 of the Network Control register. Once reception is enabled, any write to the RX Buffer Queue Base Address register is ignored. Reading this register returns the location of the descriptor currently being accessed. This value increments as buffers are used. Software should not use this register for determining where to remove received frames from the queue as it constantly changes as new frames are received. Software should instead work its way through the buffer descriptor queue checking the used bits. In terms of AMBA AXI operation, the receive descriptors are read from memory using a single 32-bit AXI access. Since the datapath is 64-bit wide, the receive descriptors shall be aligned at 64-bit boundaries and each pair of 32-bit descriptors is written using a single 64-bit AXI access.

REGISTER_NAME	ETHERNETn_receive_q_ptr
OFFSET	0x018
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	dma_rx_q_ptr[31:24]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	dma_rx_q_ptr[23:16]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	dma_rx_q_ptr[15:8]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	dma_rx_q_ptr[7:2]						Reserved	
ACCESS_TYPE	R/W						R0,WX	
PROT_TYPE	Wp							
INITIAL_VALUE	0x00						0	

[bit31:2] dma_rx_q_ptr: Receive buffer queue base address

Start address of receive buffer queue in system memory.

[bit1:0] Reserved

Always read "0". Writing has no effect.

4.7. TX Buffer Queue Base Address Register (ETHERNETn_transmit_q_ptr)

This register holds the start address of the transmit buffer queue (transmit buffers descriptor list). The TX Buffer Queue Base Address register must be initialized before transmit is started through bit 9 of the Network Control register. Once transmission has started, any write to the TX Buffer Queue Base Address register is illegal and therefore ignored. Note that due to clock boundary synchronization, it takes a maximum of four clock cycles from the writing of the transmit start bit before the transmitter is active. Writing to the TX Buffer Queue Base Address register during this time may produce unpredictable results. Reading this register returns the location of the descriptor currently being accessed. Since the Ethernet MAC DMA handles two frames at once, this may not necessarily be pointing to the current frame being transmitted. In terms of AMBA AXI operation, the transmit descriptors are written to memory using a single 32-bit AXI access. Since the datapath is 64-bit wide, the transmit descriptors shall be aligned at 64-bit boundaries and each pair of 32-bit descriptors is read from memory using a single 64-bit AXI access.

REGISTER_NAME	ETHERNETn_transmit_q_ptr
OFFSET	0x01C
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	dma_tx_q_ptr[31:24]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	dma_tx_q_ptr[23:16]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	dma_tx_q_ptr[15:8]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	dma_tx_q_ptr[7:2]						Reserved	
ACCESS_TYPE	R/W						R0,WX	
PROT_TYPE	Wp							
INITIAL_VALUE	0x00						0x0	

[bit31:2] dma_tx_q_ptr: Transmit buffer queue base address

Start address of transmit buffer queue in system memory.

[bit1:0] Reserved

Always read "0". Writing has no effect.

4.8. Receive Status Register (ETHERNETn_receive_status)

Description of Receive Status register is shown.

REGISTER_NAME	ETHERNETn_receive_status
OFFSET	0x020
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved				resp_not_ok	receive_overrun	frame_received	buffer_not_available
ACCESS_TYPE	R0,WX				R,W1C	R,W1C	R,W1C	R,W1C
PROT_TYPE	Wp							
INITIAL_VALUE	0x0				0	0	0	0

[bit31:4] Reserved
Always read "0". Writing has no effect.

[bit3] resp_not_ok: BRESP not OK
Set when the DMA block sees BRESP not OK. This bit is cleared by writing "1" to it.

[bit2] receive_overrun: Receive over run
This bit is set if the RX Packet Buffer Memory overflows. For DMA operation the buffer will be recovered if an over run occurs. This bit is cleared by writing "1" to it.

[bit1] frame_received: Frame received
One or more frames have been received and placed in memory. This bit is cleared by writing "1" to it.

[bit0] buffer_not_available: Buffer not available
An attempt was made to get a new buffer and the pointer indicated that it was owned by the processor. The RX DMA will reread the pointer each time an end of frame is received until a valid pointer is found. This bit is set following each descriptor read attempt that fails, even if consecutive pointers are unsuccessful and software has in the mean time cleared the status flag. This bit is cleared by writing "1" to it.

4.9. Interrupt Status Register (ETHERNETn_int_status)

Description of Interrupt Status register is shown. If not configured for priority queuing, the Ethernet MAC generates a single interrupt. This register indicates the source of this interrupt. The corresponding bit in the Interrupt Mask register must be clear for a bit to be set. If any bit is set in this register the interrupt signal will be asserted. For test purposes each bit can be set or reset by writing to the Interrupt Mask register. The default configuration is shown below whereby every bit is reset to “0” on writing a "1" to the appropriate bit. Reading has no effect on the status of the bit.

REGISTER_NAME	ETHERNETn_int_status
OFFSET	0x024
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved		tsu_timer_comparison_interrupt	Reserved	receive_lpi_indication_status_bit_change	tsu_seconds_register_increment	ptp_pdelay_resp_frame_transmitted	ptp_pdelay_req_frame_transmitted
ACCESS_TYPE	R0,WX		R,W1C	R,WX	R,W1C	R,W1C	R,W1C	R,W1C
PROT_TYPE	Wp							
INITIAL_VALUE	0		0	0	0	0	0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	ptp_pdelay_resp_frame_received	ptp_pdelay_req_frame_received	ptp_sync_frame_transmitted	ptp_delay_req_frame_transmitted	ptp_sync_frame_received	ptp_delay_req_frame_received	Reserved	
ACCESS_TYPE	R,W1C	R,W1C	R,W1C	R,W1C	R,W1C	R,W1C	R,WX	
PROT_TYPE	Wp							
INITIAL_VALUE	0	0	0	0	0	0	0x0	

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved	pause_frame_transmitted	pause_time_elapsed	pause_frame_with_non_zero_pause_quantum_received	resp_not_ok	receive_overrun	Reserved	
ACCESS_TYPE	R,WX	R,W1C	R,W1C	R,W1C	R,W1C	R,W1C	R,WX	
PROT_TYPE	Wp							
INITIAL_VALUE	0	0	0	0	0	0	0x0	

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	transmit_complete	amba_error	retry_limit_exceeded	Reserved	tx_used_bit_read	rx_used_bit_read	receive_complete	management_frame_sent
ACCESS_TYPE	R,W1C	R,W1C	R,W1C	R,WX	R,W1C	R,W1C	R,W1C	R,W1C
PROT_TYPE	Wp							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:30] Reserved

Always read "0". Writing has no effect.

[bit29] tsu_timer_comparison_interrupt: IEEE 1588 Timer comparison interrupt

Bit	Description
0	Interrupt de-asserted.
1	Interrupt asserted. Indicates when IEEE 1588 Timer count value is equal to programmed value.

[bit28] Reserved

Writing has no effect. This bit will read “1” if bit 28 of Interrupt Mask register is set otherwise it will read “0”.

[bit27] receive_lpi_indication_status_bit_change: Receive LPI Indication Status Bit change

Bit	Description
0	Interrupt de-asserted.
1	Interrupt asserted. Indicates that Receive LPI Indication Status bit has changed.

[bit26] tsu_seconds_register_increment: IEEE 1588 Timer Seconds register increment

Bit	Description
0	Interrupt de-asserted.
1	Interrupt asserted. Indicates that IEEE 1588 Timer Seconds register was incremented.

[bit25] ptp_pdelay_resp_frame_transmitted: PTP Pdelay_Resp frame transmitted

Bit	Description
0	Interrupt de-asserted.
1	Interrupt asserted. Indicates a PTP Pdelay_Resp frame has been transmitted.

[bit24] ptp_pdelay_req_frame_transmitted: PTP Pdelay_Req frame transmitted

Bit	Description
0	Interrupt de-asserted.
1	Interrupt asserted. Indicates a PTP Pdelay_Req frame has been transmitted.

[bit23] ptp_pdelay_resp_frame_received: PTP Pdelay_Resp frame received

Bit	Description
0	Interrupt de-asserted.
1	Interrupt asserted. Indicates a PTP Pdelay_Resp frame has been received.

[bit22] ptp_pdelay_req_frame_received: PTP Pdelay_Req frame received

Bit	Description
0	Interrupt de-asserted.
1	Interrupt asserted. Indicates a PTP Pdelay_Req frame has been received.

[bit21] ptp_sync_frame_transmitted: PTP Sync frame transmitted

Bit	Description
0	Interrupt de-asserted.
1	Interrupt asserted. Indicates a PTP Sync frame has been transmitted.

[bit20] ptp_delay_req_frame_transmitted: PTP Delay_Req frame transmitted

Bit	Description
0	Interrupt de-asserted.
1	Interrupt asserted. Indicates a PTP Delay_Req frame has been transmitted.

[bit19] ptp_sync_frame_received: PTP Sync frame received

Bit	Description
0	Interrupt de-asserted.
1	Interrupt asserted. Indicates a PTP Sync frame has been received.

[bit18] ptp_delay_req_frame_received: PTP Delay_Req frame received

Bit	Description
0	Interrupt de-asserted.
1	Interrupt asserted. Indicates a PTP Delay_Req frame has been received.

[bit17:15] Reserved

Writing has no effect. These bits will read “111” if bits [17:15] of Interrupt Mask register are set otherwise it will read “000”.

[bit14] pause_frame_transmitted: Pause frame transmitted

Bit	Description
0	Interrupt de-asserted.
1	Interrupt asserted. Indicates a pause frame has been successfully transmitted after being initiated from the Network Control register or from the `tx_pause` control pin.

[bit13] pause_time_elapsed: Pause Time elapsed

Bit	Description
0	Interrupt de-asserted.
1	Interrupt asserted. Set when either the Receive Pause Quantum register decrements to zero, or when a valid pause frame is received with a zero pause quantum field.

[bit12] pause_frame_with_non_zero_pause_quantum_received: Pause frame with non-zero pause quantum received

Bit	Description
0	Interrupt de-asserted.
1	Interrupt asserted. Indicates a valid pause has been received that has a non-zero pause quantum field.

[bit11] resp_not_ok: BRESP not OK

Bit	Description
0	Interrupt de-asserted.
1	Interrupt asserted. Set when the DMA block sees BRESP not OK.

[bit10] receive_overrun: Receive over run

Bit	Description
0	Interrupt de-asserted.
1	Interrupt asserted. Set when the receive over run status bit (ETHERNETn_receive_status[2]) gets set.

[bit9:8] Reserved

Writing has no effect. These bits will read “11” if bits [9:8] of Interrupt Mask register are set otherwise it will read “00”.

[bit7] transmit_complete: Transmit complete

Bit	Description
0	Interrupt de-asserted.
1	Interrupt asserted. Set when a frame has been transmitted.

[bit6] amba_error: Transmit frame corruption due to AMBA AXI error

Bit	Description
0	Interrupt de-asserted.
1	Interrupt asserted. Set if an error occurs whilst midway through reading transmit frame from system memory, including BRESP errors and buffers exhausted mid frame (if the buffers run out during transmission of a frame then transmission stops, FCS shall be bad and TX_ER asserted). Also set in DMA packet buffer mode if single frame is too large for the transmit packet buffer memory size

[bit5] retry_limit_exceeded: Retry limit exceeded

Bit	Description
0	Interrupt de-asserted.
1	Interrupt asserted. Transmit error.

[bit4] Reserved

Writing has no effect. This bit will read “1” if bit 4 of Interrupt Mask register is set otherwise it will read “0”.

[bit3] tx_used_bit_read: TX used bit read

Bit	Description
0	Interrupt de-asserted.
1	Interrupt asserted. Set when a transmit buffer descriptor is read with its used bit set.

[bit2] rx_used_bit_read: RX used bit read

Bit	Description
0	Interrupt de-asserted.
1	Interrupt asserted. Set when a receive buffer descriptor is read with its used bit set.

[bit1] receive_complete: Receive complete

Bit	Description
0	Interrupt de-asserted.
1	Interrupt asserted. Set when a frame has been stored in memory.

[bit0] management_frame_sent: Management frame sent

Bit	Description
0	Interrupt de-asserted.
1	Interrupt asserted. Set when the PHY Maintenance register has completed its operation.

4.10. Interrupt Enable Register (ETHERNETn_int_enable)

At reset all interrupts are disabled. Writing a "1" to the relevant bit location enables that particular interrupt. This register is write only and when read will return "0".

REGISTER_NAME	ETHERNETn_int_enable
OFFSET	0x028
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved		enable_tsu_timer_comparison_interrupt	Reserved	enable_rx_lpi_indication_interrupt	enable_tsu_seconds_register_increment	enable_ptp_pdelay_resp_frame_transmitted	enable_ptp_pdelay_req_frame_transmitted
ACCESS_TYPE	R0,WX		R0,W1	R0,W0	R0,W1	R0,W1	R0,W1	R0,W1
PROT_TYPE	Wp							
INITIAL_VALUE	0x0		0	0	0	0	0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	enable_ptp_pdelay_resp_frame_received	enable_ptp_pdelay_req_frame_received	enable_ptp_sync_frame_transmitted	enable_ptp_delay_req_frame_transmitted	enable_ptp_sync_frame_received	enable_ptp_delay_req_frame_received	Reserved	
ACCESS_TYPE	R0,W1	R0,W1	R0,W1	R0,W1	R0,W1	R0,W1	R0,WX	
PROT_TYPE	Wp							
INITIAL_VALUE	0	0	0	0	0	0	0x0	

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	enable_external_interrupt	enable_pause_frame_transmitted	enable_pause_time_elapsed	enable_pause_frame_with_non_zero_pause_quantum_received	enable_resp_not_ok	enable_receive_overrun	Reserved	
ACCESS_TYPE	R0,W1	R0,W1	R0,W1	R0,W1	R0,W1	R0,W1	R0,WX	
PROT_TYPE	Wp							
INITIAL_VALUE	0	0	0	0	0	0	0x0	

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	enable_transmit_complete	enable_amba_error	enable_retry_limit_exceeded	Reserved	enable_tx_used_bit_read	enable_rx_used_bit_read	enable_receive_complete	enable_management_frame_sent
ACCESS_TYPE	R0,W1	R0,W1	R0,W1	R0,WX	R0,W1	R0,W1	R0,W1	R0,W1
PROT_TYPE	Wp							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:30] Reserved
Always read "0". Writing has no effect.

[bit29] enable_tsu_timer_comparison_interrupt: Enable IEEE 1588 Timer comparison interrupt
Always read "0". Writing "1" enables IEEE 1588 Timer comparison interrupt.

[bit28] Reserved
Always read "0". Always write "0".

[bit27] enable_rx_lpi_indication_interrupt: Enable RX LPI indication interrupt

[bit26] enable_tsu_seconds_register_increment: Enable IEEE 1588 Timer Seconds register increment
Always read "0". Writing "1" enables IEEE 1588 Timer Seconds register increment interrupt.

[bit25] enable_ptp_pdelay_resp_frame_transmitted: Enable PTP Pdelay_Resp frame transmitted
Always read "0". Writing "1" enables PTP Pdelay_Resp frame transmitted interrupt.

[bit24] enable_ptp_pdelay_req_frame_transmitted: Enable PTP Pdelay_Req frame transmitted
Always read "0". Writing "1" enables PTP Pdelay_Req frame transmitted interrupt.

[bit23] enable_ptp_pdelay_resp_frame_received: Enable PTP Pdelay_Resp frame received
Always read "0". Writing "1" enables PTP Pdelay_Resp frame received interrupt.

[bit22] enable_ptp_pdelay_req_frame_received: Enable PTP Pdelay_Req frame received
Always read "0". Writing "1" enables PTP Pdelay_Req frame received interrupt.

[bit21] enable_ptp_sync_frame_transmitted: Enable PTP Sync frame transmitted
Always read "0". Writing "1" enables PTP Sync frame transmitted interrupt.

[bit20] enable_ptp_delay_req_frame_transmitted: Enable PTP Delay_Req frame transmitted
Always read "0". Writing "1" enables PTP Delay_Req frame transmitted interrupt.

[bit19] enable_ptp_sync_frame_received: Enable PTP Sync frame received
Always read "0". Writing "1" enables PTP Sync frame received interrupt.

[bit18] enable_ptp_delay_req_frame_received: Enable PTP Delay_Req frame received
Always read "0". Writing "1" enables PTP Delay_Req frame received interrupt.

[bit17:16] Reserved
Always read "0". Writing has no effect.

[bit15] enable_external_interrupt: Enable External interrupt
Always read "0". Writing "1" enables External interrupt

[bit14] enable_pause_frame_transmitted: Enable Pause frame transmitted
Always read "0". Writing "1" enables Pause frame transmitted interrupt.

[bit13] enable_pause_time_elapsed: Enable Pause Time elapsed
Always read "0". Writing "1" enables Pause Time elapsed interrupt.

[bit12] enable_pause_frame_with_non_zero_pause_quantum_received: Enable Pause frame with non-zero pause quantum received
Always read "0". Writing "1" enables Pause frame with non-zero pause quantum received interrupt.

[bit11] enable_resp_not_ok: Enable BRESP not OK
Always read "0". Writing "1" enables BRESP not OK interrupt.

[bit10] enable_receive_overrun: Enable Receive over run
Always read "0". Writing "1" enables Receive over run interrupt.

[bit9:8] Reserved
Always read "0". Writing has no effect.

[bit7] enable_transmit_complete: Enable Transmit complete
Always read "0". Writing "1" enables Transmit complete interrupt.

[bit6] enable_amba_error: Enable Transmit frame corruption due to AMBA AXI error
Always read "0". Writing "1" enables Transmit frame corruption due to AMBA AXI error interrupt.

[bit5] enable_retry_limit_exceeded: Enable Retry limit exceeded
Always read "0". Writing "1" enables Retry limit exceeded interrupt.

[bit4] Reserved
Always read "0". Writing has no effect.

[bit3] enable_tx_used_bit_read: Enable TX used bit read
Always read "0". Writing "1" enables TX used bit read interrupt.

[bit2] enable_rx_used_bit_read: Enable RX used bit read
Always read "0". Writing "1" enables RX used bit read interrupt.

[bit1] enable_receive_complete: Enable Receive complete
Always read "0". Writing "1" enables Receive complete interrupt.

[bit0] enable_management_frame_sent: Enable Management frame sent
Always read "0". Writing "1" enables Management frame sent interrupt.

4.11. Interrupt Disable Register (ETHERNETn_int_disable)

Writing a "1" to the relevant bit location disables that particular interrupt. This register is write only and when read will return "0".

REGISTER_NAME	ETHERNETn_int_disable
OFFSET	0x02C
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved		disable_tsu_timer_comparison_interrupt	Reserved	disable_rx_lpi_indication_interrupt	disable_tsu_seconds_register_increment	disable_ptp_pdelay_resp_frame_transmitted	disable_ptp_pdelay_req_frame_transmitted
ACCESS_TYPE	R0,WX		R,W1	R0,W1	R,W1	R,W1	R,W1	R,W1
PROT_TYPE	Wp							
INITIAL_VALUE	0x0		0	0	0	0	0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	disable_ptp_pdelay_resp_frame_received	disable_ptp_pdelay_req_frame_received	disable_ptp_sync_frame_transmitted	disable_ptp_delay_req_frame_transmitted	disable_ptp_sync_frame_received	disable_ptp_delay_req_frame_received	Reserved	
ACCESS_TYPE	R,W1	R,W1	R,W1	R,W1	R,W1	R,W1	R0,WX	
PROT_TYPE	Wp							
INITIAL_VALUE	0	0	0	0	0	0	0x0	

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	disable_external_interrupt	disable_pause_frame_transmitted	disable_pause_time_elapsed	disable_pause_frame_with_non_zero_pause_quantum_received	disable_resp_not_ok	disable_receive_overrun	Reserved	
ACCESS_TYPE	R,W1	R,W1	R,W1	R,W1	R,W1	R,W1	R0,WX	
PROT_TYPE	Wp							
INITIAL_VALUE	0	0	0	0	0	0	0x0	

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	disable_transmit_complete	disable_amba_error	disable_retry_limit_exceeded	Reserved	disable_tx_used_bit_read	disable_rx_used_bit_read	disable_receive_complete	disable_management_frame_sent
ACCESS_TYPE	R,W1	R,W1	R,W1	R0,WX	R,W1	R,W1	R,W1	R,W1
PROT_TYPE	Wp							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:30] Reserved
Always read "0". Writing has no effect.

[bit29] disable_tsu_timer_comparison_interrupt: Disable IEEE 1588 Timer comparison interrupt
Writing "1" disables IEEE 1588 Timer comparison interrupt.

[bit28] Reserved
Always read "0". Always write “1”.

[bit27] disable_rx_lpi_indication_interrupt: Disable RX LPI indication interrupt

[bit26] disable_tsu_seconds_register_increment: Disable IEEE 1588 Timer Seconds register increment
Writing "1" disables IEEE 1588 Timer Seconds register increment interrupt.

[bit25] disable_ptp_pdelay_resp_frame_transmitted: Disable PTP Pdelay_Resp frame transmitted
Writing "1" disables PTP Pdelay_Resp frame transmitted interrupt.

[bit24] disable_ptp_pdelay_req_frame_transmitted: Disable PTP Pdelay_Req frame transmitted
Writing "1" disables PTP Pdelay_Req frame transmitted interrupt.

[bit23] disable_ptp_pdelay_resp_frame_received: Disable PTP Pdelay_Resp frame received
Writing "1" disables PTP Pdelay_Resp frame received interrupt.

[bit22] disable_ptp_pdelay_req_frame_received: Disable PTP Pdelay_Req frame received
Writing "1" disables PTP Pdelay_Req frame received interrupt.

[bit21] disable_ptp_sync_frame_transmitted: Disable PTP Sync frame transmitted
Writing "1" disables PTP Sync frame transmitted interrupt.

[bit20] disable_ptp_delay_req_frame_transmitted: Disable PTP Delay_Req frame transmitted
Writing "1" disables PTP Delay_Req frame transmitted interrupt.

[bit19] disable_ptp_sync_frame_received: Disable PTP Sync frame received
Writing "1" disables PTP Sync frame received interrupt.

[bit18] disable_ptp_delay_req_frame_received: Disable PTP Delay_Req frame received
Writing "1" disables PTP Delay_Req frame received interrupt.

[bit17:16] Reserved
Always read "0". Writing has no effect.

[bit15] disable_external_interrupt: Disable External interrupt
Writing "1" disables External interrupt

[bit14] disable_pause_frame_transmitted: Disable Pause frame transmitted
Writing "1" disables Pause frame transmitted interrupt.

[bit13] disable_pause_time_elapsed: Disable Pause Time elapsed
Writing "1" disables Pause Time elapsed interrupt.

[bit12] disable_pause_frame_with_non_zero_pause_quantum_received: Disable Pause frame with non-zero pause quantum received
Writing "1" disables Pause frame with non-zero pause quantum received interrupt.

[bit11] disable_resp_not_ok: Disable BRESP not OK
Writing "1" disables BRESP not OK interrupt.

[bit10] disable_receive_overrun: Disable Receive over run
Writing "1" disables Receive over run interrupt.

[bit9:8] Reserved
Always read "0". Writing has no effect.

[bit7] disable_transmit_complete: Disable Transmit complete
Writing "1" disables Transmit complete interrupt.

[bit6] disable_amba_error: Disable Transmit frame corruption due to AMBA AXI error
Writing "1" disables Transmit frame corruption due to AMBA AXI error interrupt.

[bit5] disable_retry_limit_exceeded: Disable Retry limit exceeded
Writing "1" disables Retry limit exceeded interrupt.

[bit4] Reserved
Always read "0". Writing has no effect.

[bit3] disable_tx_used_bit_read: Disable TX used bit read
Writing "1" disables TX used bit read interrupt.

[bit2] disable_rx_used_bit_read: Disable RX used bit read
Writing "1" disables RX used bit read interrupt.

[bit1] disable_receive_complete: Disable Receive complete
Writing "1" disables Receive complete interrupt.

[bit0] disable_management_frame_sent: Disable Management frame sent
Writing "1" disables Management frame sent interrupt.

4.12. Interrupt Mask Register (ETHERNETn_int_mask)

The Interrupt Mask register is a read only register indicating which interrupts are masked. All bits are set at reset and can be reset individually by writing to the Interrupt Enable register or set individually by writing to the Interrupt Disable register. Having separate address locations for enable and disable saves the need for performing a read modify write when updating the Interrupt Mask register. For test purposes there is a write-only function to this register that allows the bits in the Interrupt Status register to be set or cleared, regardless of the state of the Interrupt Mask register.

REGISTER_NAME	ETHERNETn_int_mask
OFFSET	0x030
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved		tsu_timer_comparison_mask	wol_event_received_mask	rx_lpi_indication_mask	tsu_seconds_register_increment_mask	ptp_pdelay_resp_frame_transmitted_mask	ptp_pdelay_req_frame_transmitted_mask
ACCESS_TYPE	R0,WX		R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	Wp							
INITIAL_VALUE	0x0		1	1	1	1	1	1

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	ptp_pdelay_resp_frame_received_mask	ptp_pdelay_req_frame_received_mask	ptp_sync_frame_transmitted_mask	ptp_delay_req_frame_transmitted_mask	ptp_sync_frame_received_mask	ptp_delay_req_frame_received_mask	pcs_link_partner_page_mask	pcs_auto_negotiation_complete_interrupt_mask
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	Wp							
INITIAL_VALUE	1	1	1	1	1	1	1	1

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	external_interrupt_mask	pause_frame_transmitted_interrupt_mask	pause_time_zero_interrupt_mask	pause_frame_with_non_zero_pause_quantum_interrupt_mask	resp_not_ok_interrupt_mask	receive_overrun_interrupt_mask	link_change_interrupt_mask	not_used
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R1,WX
PROT_TYPE	Wp							
INITIAL_VALUE	1	1	1	1	1	1	1	1

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	transmit_complete_interrupt_mask	amba_error_interrupt_mask	retry_limit_exceeded_or_late_collision_mask	transmit_buffer_under_run_interrupt_mask	transmit_used_bit_read_interrupt_mask	receive_used_bit_read_interrupt_mask	receive_complete_interrupt_mask	management_done_interrupt_mask
ACCESS_TYPE	R,WX	R,WX	R,WX	R1,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	Wp							
INITIAL_VALUE	1	1	1	1	1	1	1	1

[bit31:30] Reserved
Always read "0". Writing has no effect.

[bit29] tsu_timer_comparison_mask
Enable TSU timer comparison interrupt mask.

[bit28] wol_event_received_mask

A read of this register returns the value of the WOL event received mask. A write to this register directly affects the state of the corresponding bit in the interrupt status register, causing an interrupt to be generated if a 1 is written.

Bit	Description
0	Interrupt is enabled.
1	Interrupt is disabled.

[bit27] rx_lpi_indication_mask

A read of this register returns the value of the RX LPI indication mask. A write to this register directly affects the state of the corresponding bit in the Interrupt Status register, causing an interrupt to be generated if a "1" is written.

Bit	Description
0	Interrupt is enabled.
1	Interrupt is disabled.

[bit26] tsu_seconds_register_increment_mask: IEEE 1588 Timer Seconds register increment mask

A read of this register returns the value of the IEEE 1588 Timer Seconds register increment mask. A write to this register directly affects the state of the corresponding bit in the Interrupt Status register, causing an interrupt to be generated if a "1" is written.

Bit	Description
0	Interrupt is enabled.
1	Interrupt is disabled.

[bit25] ptp_pdelay_resp_frame_transmitted_mask: PTP Pdelay_Resp frame transmitted mask

A read of this register returns the value of the PTP Pdelay_Resp frame transmitted mask. A write to this register directly affects the state of the corresponding bit in the Interrupt Status register, causing an interrupt to be generated if a "1" is written.

Bit	Description
0	Interrupt is enabled.
1	Interrupt is disabled.

[bit24] ptp_pdelay_req_frame_transmitted_mask: PTP Pdelay_Req frame transmitted mask

A read of this register returns the value of the PTP Pdelay_Req frame transmitted mask. A write to this register directly affects the state of the corresponding bit in the Interrupt Status register, causing an interrupt to be generated if a "1" is written.

Bit	Description
0	Interrupt is enabled.
1	Interrupt is disabled.

[bit23] ptp_pdelay_resp_frame_received_mask: PTP Pdelay_Resp frame received mask

A read of this register returns the value of the PTP Pdelay_Resp frame received mask. A write to this register directly affects the state of the corresponding bit in the Interrupt Status register, causing an interrupt to be generated if a "1" is written.

Bit	Description
0	Interrupt is enabled.
1	Interrupt is disabled.

[bit22] ptp_pdelay_req_frame_received_mask: PTP Pdelay_Req frame received mask

A read of this register returns the value of the PTP Pdelay_Req frame received mask. A write to this register directly affects the state of the corresponding bit in the Interrupt Status register, causing an interrupt to be generated if a "1" is written.

Bit	Description
0	Interrupt is enabled.
1	Interrupt is disabled.

[bit21] ptp_sync_frame_transmitted_mask: PTP Sync frame transmitted mask

A read of this register returns the value of the PTP Sync frame transmitted mask. A write to this register directly affects the state of the corresponding bit in the Interrupt Status register, causing an interrupt to be generated if a "1" is written.

Bit	Description
0	Interrupt is enabled.
1	Interrupt is disabled.

[bit20] ptp_delay_req_frame_transmitted_mask: PTP Delay_Req frame transmitted mask

A read of this register returns the value of the PTP Delay_Req frame transmitted mask. A write to this register directly affects the state of the corresponding bit in the Interrupt Status register, causing an interrupt to be generated if a "1" is written.

Bit	Description
0	Interrupt is enabled.
1	Interrupt is disabled.

[bit19] ptp_sync_frame_received_mask: PTP Sync frame received mask

A read of this register returns the value of the PTP Sync frame received mask. A write to this register directly affects the state of the corresponding bit in the Interrupt Status register, causing an interrupt to be generated if a "1" is written.

Bit	Description
0	Interrupt is enabled.
1	Interrupt is disabled.

[bit18] ptp_delay_req_frame_received_mask: PTP Delay_Req frame received mask

A read of this register returns the value of the PTP Delay_Req frame received mask. A write to this register directly affects the state of the corresponding bit in the Interrupt Status register, causing an interrupt to be generated if a "1" is written.

Bit	Description
0	Interrupt is enabled.
1	Interrupt is disabled.

[bit17] pcs_link_partner_page_mask

A read of this register returns the value of the PCS link partner page mask. A write to this register directly affects the state of the corresponding bit in the interrupt status register, causing an interrupt to be generated if a 1 is written.

Bit	Description
0	Interrupt is enabled.
1	Interrupt is disabled.

[bit16] pcs_auto_negotiation_complete_interrupt_mask

A read of this register returns the value of the PCS auto-negotiation complete interrupt mask. A write to this register directly affects the state of the corresponding bit in the interrupt status register, causing an interrupt to be generated if a 1 is written.

Bit	Description
0	Interrupt is enabled.
1	Interrupt is disabled.

[bit15] external_interrupt_mask: External interrupt mask

A read of this register returns the value of the external interrupt mask. A write to this register directly affects the state of the corresponding bit in the Interrupt Status register, causing an interrupt to be generated if a "1" is written.

Bit	Description
0	Interrupt is enabled.
1	Interrupt is disabled.

[bit14] pause_frame_transmitted_interrupt_mask

pause frame transmitted interrupt mask. A write to this register directly affects the state of the corresponding bit in the interrupt status register, causing an interrupt to be generated if a 1 is written.

Bit	Description
0	Interrupt is enabled.
1	Interrupt is disabled.

[bit13] pause_time_zero_interrupt_mask

pause time zero interrupt mask. A write to this register directly affects the state of the corresponding bit in the interrupt status register, causing an interrupt to be generated if a 1 is written.

Bit	Description
0	Interrupt is enabled.
1	Interrupt is disabled.

[bit12] pause_frame_with_non_zero_pause_quantum_interrupt_mask

pause frame with non-zero pause quantum interrupt mask. A write to this register directly affects the state of the corresponding bit in the interrupt status register, causing an interrupt to be generated if a 1 is written.

Bit	Description
0	Interrupt is enabled.
1	Interrupt is disabled.

[bit11] resp_not_ok_interrupt_mask

bresp/hresp not OK interrupt mask. A write to this register directly affects the state of the corresponding bit in the interrupt status register, causing an interrupt to be generated if a 1 is written.

Bit	Description
0	Interrupt is enabled.
1	Interrupt is disabled.

[bit10] receive_overrun_interrupt_mask

receive overrun interrupt mask. A write to this register directly affects the state of the corresponding bit in the interrupt status register, causing an interrupt to be generated if a 1 is written.

Bit	Description
0	Interrupt is enabled.
1	Interrupt is disabled.

[bit9] link_change_interrupt_mask

A read of this register returns the value of the link change interrupt mask. A write to this register directly affects the state of the corresponding bit in the interrupt status register, causing an interrupt to be generated if a 1 is written.

Bit	Description
0	Interrupt is enabled.
1	Interrupt is disabled.

[bit8] not_used

Not used

[bit7] transmit_complete_interrupt_mask

transmit complete interrupt mask. A write to this register directly affects the state of the corresponding bit in the interrupt status register, causing an interrupt to be generated if a 1 is written.

Bit	Description
0	Interrupt is enabled.
1	Interrupt is disabled.

[bit6] amba_error_interrupt_mask

transmit frame corruption due to AMBA (AHB/AXI) error interrupt mask. A write to this register directly affects the state of the corresponding bit in the interrupt status register, causing an interrupt to be generated if a 1 is written.

Bit	Description
0	Interrupt is enabled.
1	Interrupt is disabled.

[bit5] retry_limit_exceeded_or_late_collision_mask

A read of this register returns the value of the retry limit exceeded or late collision (gigabit mode only) interrupt mask. A write to this register directly affects the state of the corresponding bit in the interrupt status register, causing an interrupt to be generated if a 1 is written.

Bit	Description
0	Interrupt is enabled.
1	Interrupt is disabled.

[bit4] transmit_buffer_under_run_interrupt_mask

transmit buffer under run interrupt mask. A write to this register directly affects the state of the corresponding bit in the interrupt status register, causing an interrupt to be generated if a 1 is written.

Bit	Description
0	Interrupt is enabled.
1	Interrupt is disabled.

[bit3] transmit_used_bit_read_interrupt_mask

transmit used bit read interrupt mask. A write to this register directly affects the state of the corresponding bit in the interrupt status register, causing an interrupt to be generated if a 1 is written.

Bit	Description
0	Interrupt is enabled.
1	Interrupt is disabled.

[bit2] receive_used_bit_read_interrupt_mask

receive used bit read interrupt mask. A write to this register directly affects the state of the corresponding bit in the interrupt status register, causing an interrupt to be generated if a 1 is written.

Bit	Description
0	Interrupt is enabled.
1	Interrupt is disabled.

[bit1] receive_complete_interrupt_mask

receive complete interrupt mask. A write to this register directly affects the state of the corresponding bit in the interrupt status register, causing an interrupt to be generated if a 1 is written.

Bit	Description
0	Interrupt is enabled.
1	Interrupt is disabled.

[bit0] management_done_interrupt_mask

management done interrupt mask. A write to this register directly affects the state of the corresponding bit in the interrupt status register, causing an interrupt to be generated if a 1 is written.

Bit	Description
0	Interrupt is enabled.
1	Interrupt is disabled.

4.13. PHY Maintenance Register (ETHERNETn_phy_management)

The PHY Maintenance register is implemented as a shift register. Writing to the register starts a shift operation which is signaled as complete when bit 2 is set in the Network Status register. It takes about 2000 clock cycles to complete, when MDC is set for clock divide by 32 in the Network Configuration register. An interrupt is generated upon completion. During this time, the MSB of the register is output on the MDIO pin and the LSB updated from the MDIO pin with each MDC cycle. This causes transmission of a PHY management frame on MDIO. See Section 22.2.4.5 of the IEEE Std 802.3 standard. Reading during the shift operation will return the current contents of the shift register. At the end of management operation, the bits will have shifted back to their original locations. For a read operation, the data bits will be updated with data read from the PHY. It is important to write the correct values to the register to ensure a valid PHY management frame is produced. The MDIO interface can read IEEE Std 802.3 clause 45 PHYs as well as clause 22 PHYs. To read clause 45 PHYs, bit 30 should be written with a "0" rather than a "1". For a description of MDC generation, see Network Configuration register.

REGISTER_NAME	ETHERNETn_phy_management
OFFSET	0x034
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	write0	write1	operation[1:0]		phy_address[4:1]			
ACCESS_TYPE	R/w0	R/W	R/W		R/W			
PROT_TYPE	Wp							
INITIAL_VALUE	0	0	0x0		0x0			

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	phy_address[0]	register_address[4:0]					write10[1:0]	
ACCESS_TYPE	R/W	R/W					R/W	
PROT_TYPE	Wp							
INITIAL_VALUE	0	0x00					0x0	

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	phy_write_read_data[15:8]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	phy_write_read_data[7:0]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31] write0: Write 0

Always read written value. Writing with "0".

[bit30] write1: Write 1

Must be written with "1" for a valid Clause 22 frame and with "0" for a valid Clause 45 frame.

[bit29:28] operation: Operation

Bits	Description
00	Reserved
01	Write operation.
10	Read operation.
11	Reserved

[bit27:23] phy_address: PHY address
PHY address.

[bit22:18] register_address: Register address
Specifies the register in the PHY to access.

[bit17:16] write10: Write 10
Always read written value, Write with "0x2".

[bit15:0] phy_write_read_data: PHY write/read data
For a write operation the value in this register is written to the PHY as data. For a read operation this register contains the data read from the PHY.

4.14. Receive Pause Quantum Register (ETHERNETn_pause_time)

Description of Receive Pause Quantum register is shown.

REGISTER_NAME	ETHERNETn_pause_time
OFFSET	0x038
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	quantum[15:8]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	quantum[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:16] Reserved

Always read "0". Writing has no effect.

[bit15:0] quantum: Received pause quantum

Stores the current value of the received pause quantum register which is decremented every 512 bit times.

4.15. Transmit Pause Quantum Register (ETHERNETn_tx_pause_quantum)

Description of Transmit Pause Quantum register is shown.

REGISTER_NAME	ETHERNETn_tx_pause_quantum
OFFSET	0x03C
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	quantum_p1[31:24]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0xFF							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	quantum_p1[23:16]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0xFF							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	quantum[15:8]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0xFF							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	quantum[7:0]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0xFF							

[bit31:16] quantum_p1: Transmit pause quantum - Priority 1

Write pause quantum value for pause frame transmission of priority 1.

[bit15:0] quantum: Transmit pause quantum

Write pause quantum value for pause frame transmission.

4.16. TX Partial Store and Forward Register (ETHERNETn_pbuf_txcutthru)

Description of TX Partial Store and Forward register is shown.

REGISTER_NAME	ETHERNETn_pbuf_txcutthru
OFFSET	0x040
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	dma_tx_cut thru	Reserved						
ACCESS_TYPE	R/W	R0,WX						
PROT_TYPE	Wp							
INITIAL_VALUE	0	0x00						

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved					dma_tx_cutthru_threshold[10:8]		
ACCESS_TYPE	R0,WX					R/W		
PROT_TYPE	Wp							
INITIAL_VALUE	0x00					0x7		

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	dma_tx_cutthru_threshold[7:0]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0xFF							

[bit31] dma_tx_cutthru: TX Partial Store and Forward

Bit	Description
0	Enable TX Full Store and Forward operation mode.
1	Enable TX Partial Store and Forward operation mode.

[bit30:11] Reserved

Always read "0". Writing has no effect.

[bit10:0] dma_tx_cutthru_threshold: TX partial store and forward threshold

Watermark value. This value must be >= 0x14

4.17. RX Partial Store and Forward Register (ETHERNETn_pbuf_rxcutthru)

Description of RX Partial Store and Forward register is shown.

REGISTER_NAME	ETHERNETn_pbuf_rxcutthru
OFFSET	0x044
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	dma_rx_cut thru	Reserved						
ACCESS_TYPE	R/W	R0,WX						
PROT_TYPE	Wp							
INITIAL_VALUE	0	0x00						

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved							dma_rx_cutthru_threshold[8]
ACCESS_TYPE	R0,WX							R/W
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							1

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	dma_rx_cutthru_threshold[7:0]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0xFF							

[bit31] dma_rx_cutthru: RX Partial Store and Forward

Bit	Description
0	Enable RX Full Store and Forward operation mode.
1	Enable RX Partial Store and Forward operation mode.

[bit30:9] Reserved

Always read "0". Writing has no effect.

[bit8:0] dma_rx_cutthru_threshold: RX partial store and forward threshold

Watermark value.

4.18. Jumbo-Frame Maximum Length Register (ETHERNETn_jumbo_max_length)

Description of Jumbo-Frame Maximum Length register is shown.

REGISTER_NAME	ETHERNETn_jumbo_max_length
OFFSET	0x048
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	jumbo_max_length[15:8]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x28							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	jumbo_max_length[7:0]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:16] Reserved

Always read "0". Writing has no effect.

[bit15:0] jumbo_max_length: Jumbo-Frame maximum length

Defines the maximum Jumbo-Frame size in Bytes. Reset value corresponds to 10240 Bytes.

4.19. AXI Maximum Pipeline Register (ETHERNETn_axi_max_pipeline)

Description of AXI Maximum Pipeline register is shown. Used to set the maximum amount of outstanding transactions on the AXI bus between AR/R channels and AW/W channels.

REGISTER_NAME	ETHERNETn_axi_max_pipeline
OFFSET	0x054
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	aw2w_max_pipeline[7:0]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x01							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	ar2r_max_pipeline[7:0]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x01							

[bit31:16] Reserved

Always read "0". Writing has no effect.

[bit15:8] aw2w_max_pipeline:

Defines the maximum number of outstanding AXI write requests that can be issued by the DMA via the AW channel.

Bits	Description
0x00	The maximum number of outstanding write requests is 1.
. . .	. . .
0x0F	The maximum number of outstanding write requests is 16.
0x10 – 0xFF	Setting reserved.

[bit7:0] ar2r_max_pipeline:

Defines the maximum number of outstanding AXI read requests that can be issued by the DMA via the AR channel.

Bits	Description
0x00	The maximum number of outstanding read requests is 1.
. . .	. . .
0x0F	The maximum number of outstanding read requests is 16.
0x10 – 0xFF	Setting reserved.

4.20. Hash Bottom Register (ETHERNETn_hash_bottom)

Description of Hash Bottom register is shown.

REGISTER_NAME	ETHERNETn_hash_bottom
OFFSET	0x080
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	address[31:24]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	address[23:16]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	address[15:8]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	address[7:0]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:0] address: Hash address [31:0]

The lower 32 bits of the hash address register.

4.21. Hash Top Register (ETHERNETn_hash_top)

Description of Hash Top register is shown.

REGISTER_NAME	ETHERNETn_hash_top
OFFSET	0x084
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	address[63:56]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	address[55:48]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	address[47:40]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	address[39:32]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:0] address: Hash address [63:32]

The upper 32 bits of the hash address register.

4.22. Specific Address Bottom i Register (ETHERNETn_spec_add_bottom_i) (i=1 to 4)

The address stored in the Specific Address i registers are deactivated at reset or when their corresponding Specific Address Bottom i register is written. They are activated when Specific Address Top i register is written.

REGISTER_NAME	ETHERNETn_spec_add_bottom_i
OFFSET	(0x088 + ((i-1)*0x8))
ACCESS_SIZE	W
MULTIPLE	1:4
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	address[31:24]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	address[23:16]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	address[15:8]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	address[7:0]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:0] address: Specific address i [31:0]

Least significant 32 bits of the destination address, which is bits [31:0]. Bit 0 indicates whether the address is multicast or unicast and corresponds to the least significant bit of the first byte received.

4.23. Specific Address Top i Register (ETHERNETn_spec_add_top_i) (i=1 to 4)

The address stored in the Specific Address i registers are deactivated at reset or when their corresponding Specific Address Bottom i register is written. They are activated when Specific Address Top i register is written.

REGISTER_NAME	ETHERNETn_spec_add_top_i
OFFSET	(0x08C + ((i-1)*0x8))
ACCESS_SIZE	W
MULTIPLE	1:4
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved		filter_byte_mask[5:0]					
ACCESS_TYPE	R0,WX		R/W					
PROT_TYPE	Wp							
INITIAL_VALUE	0x0		0x00					

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved							filter_type
ACCESS_TYPE	R0,WX							R/W
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	address[47:40]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	address[39:32]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:30] Reserved
Always read "0". Writing has no effect.

[bit29:24] filter_byte_mask: Filter byte mask
When high, the associated byte of the specific address will not be compared. Bit 24 controls whether the first byte received should be compared. Bit 29 controls whether the last byte received should be compared.

[bit23:17] Reserved
Always read "0". Writing has no effect.

[bit16] filter_type: Filter type
This control bit selects whether this filter should be comparing the MAC source address or the MAC destination address of the received Ethernet frame. When set to "0", the filter is a destination address filter. When set to "1", the filter is a source address filter.

[bit15:0] address: Specific address i [47:32]
The most significant bits of the destination/source address that is to be compared, that is bits [47:32].

4.24. Type ID Match i Register (ETHERNETn_spec_type_i) (i= 1 to 4)

Description of Type ID Match i register is shown.

REGISTER_NAME	ETHERNETn_spec_type_i
OFFSET	(0x0A8 + ((i-1)*0x4)
ACCESS_SIZE	W
MULTIPLE	1:4
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	enable_copy	Reserved						
ACCESS_TYPE	R/W	R0,WX						
PROT_TYPE	Wp							
INITIAL_VALUE	0	0x00						

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	match[15:8]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	match[7:0]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31] enable_copy: Enable copy

Enable copying of Type ID match, matched frames.

[bit30:16] Reserved

Always read "0". Writing has no effect.

[bit15:0] match: Type ID match

Type ID match. For use in comparisons with received frames Type ID/length field.

4.25. IPG Stretch Register (ETHERNETn_stretch_ratio)

Description of IPG Stretch register is shown.

REGISTER_NAME	ETHERNETn_stretch_ratio
OFFSET	0x0BC
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	ipg_stretch[15:8]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	ipg_stretch[7:0]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:16] Reserved

Always read "0". Writing has no effect.

[bit15:0] ipg_stretch: IPG Stretch

Bits [7:0] are multiplied with the previously transmitted frame length (including preamble) bits [15:8] +1 divide the frame length. If the resulting number is greater than 96 and bit 28 is set in the Network Configuration register then the resulting number is used for the transmit Interpacket Gap. 1 is added to bits [15:8] to prevent a divide by zero.

4.26. Stacked VLAN Register (ETHERNETn_stacked_vlan)

Description of Stacked VLAN register is shown.

REGISTER_NAME	ETHERNETn_stacked_vlan
OFFSET	0x0C0
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	enable_processing	Reserved						
ACCESS_TYPE	R/W	R0,WX						
PROT_TYPE	Wp							
INITIAL_VALUE	0	0x00						

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	match[15:8]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	match[7:0]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31] enable_processing: Enable processing

Enable stacked VLAN processing mode

[bit30:16] Reserved

Always read "0". Writing has no effect.

[bit15:0] match: User defined VLAN_TYPE field

When Stacked VLAN is enabled, the first VLAN tag in a received frame will only be accepted if the VLAN type field is equal to this user defined VLAN_TYPE OR equal to the standard VLAN type (0x8100). Note that the second VLAN tag of a Stacked VLAN packet will only be matched correctly if its VLAN_TYPE field equals 0x8100.

4.27. Transmit PFC Pause Register (ETHERNETn_tx_pfc_pause)

Description of Transmit PFC Pause register is shown.

REGISTER_NAME	ETHERNETn_tx_pfc_pause
OFFSET	0x0C4
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	vector[7:0]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	vector_enable[7:0]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:16] Reserved

Always read "0". Writing has no effect.

[bit15:8] vector: Priority vector pause size

If bit 17 of the Network Control register is written with a "1" then for each entry equal to "0" in the Transmit PFC Pause register [15:8], the PFC pause frame's pause quantum field associated with that entry will be taken from the Transmit Pause Quantum register. For each entry equal to "1" in the Transmit PFC Pause register [15:8], the pause quantum associated with that entry will be "0".

[bit7:0] vector_enable: Priority vector enable

If bit 17 of the Network Control register is written with a "1" then the priority enable vector of the PFC priority based pause frame will be set equal to the value stored in this register [7:0].

4.28. Specific Address Mask 1 Bottom Register (ETHERNETn_mask_add1_bottom)

Description of Specific Address Mask 1 Bottom register is shown.

REGISTER_NAME	ETHERNETn_mask_add1_bottom
OFFSET	0x0C8
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	address_mask[31:24]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	address_mask[23:16]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	address_mask[15:8]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	address_mask[7:0]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:0] address_mask: Specific Address Mask [31:0]

Setting a bit to "1" masks the corresponding bit in the Specific Address Bottom i register.

4.29. Specific Address Mask 1 Top Register (ETHERNETn_mask_add1_top)

Description of Specific Address Mask 1 Top register is shown.

REGISTER_NAME	ETHERNETn_mask_add1_top
OFFSET	0x0CC
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	address_mask[47:40]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	address_mask[39:32]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:16] Reserved

Always read "0". Writing has no effect.

[bit15:0] address_mask: Specific Address Mask [47:32]

Setting a bit to "1" masks the corresponding bit in the Specific Address Top i register.

4.30. RX PTP Unicast IP Destination Address Register (ETHERNETn_rx_ptp_unicast)

Description of RX PTP Unicast IP Destination Address register is shown.

REGISTER_NAME	ETHERNETn_rx_ptp_unicast
OFFSET	0x0D4
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	address[31:24]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	address[23:16]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	address[15:8]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	address[7:0]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:0] address: Unicast IP destination address

Used for detection of PTP frames on receive path.

4.31. TX PTP Unicast IP Destination Address Register (ETHERNETn_tx_ptp_unicast)

Description of TX PTP Unicast IP Destination Address register is shown.

REGISTER_NAME	ETHERNETn_tx_ptp_unicast
OFFSET	0x0D8
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	address[31:24]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	address[23:16]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	address[15:8]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	address[7:0]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:0] address: Unicast IP destination address

Used for detection of PTP frames on transmit path.

4.32. Receive DMA Data Buffer Address Mask Register (ETHERNETn_dma_addr_or_mask)

Description of Receive DMA Data Buffer Address Mask register is shown.

REGISTER_NAME	ETHERNETn_dma_addr_or_mask
OFFSET	0x0D0
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	mask_value[3:0]				Reserved			
ACCESS_TYPE	R/W				R0,WX			
PROT_TYPE	Wp							
INITIAL_VALUE	0x0				0x0			

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved				mask_enable[3:0]			
ACCESS_TYPE	R0,WX				R/W			
PROT_TYPE	Wp							
INITIAL_VALUE	0x0				0x0			

[bit31:28] mask_value: Data Buffer Address Mask Value

Values used to force bits [31:28] of the receive data buffer AXI address to a particular value when the associated enable bits stored in this register [3:0] are set. Any changes to this register will be ignored while the DMA is currently processing a receive packet. It will only affect the next full packet to be written to system memory.

[bit27:4] Reserved

Always read "0". Writing has no effect.

[bit3:0] mask_enable: Data Buffer Address Mask Enable

These bits are associated directly with bits [31:28].When bit 0 is set, the AXI address bit 28 used for accessing the receive data buffers will be forced to the value stored in bit 28 of this register. When bit 1 is set, the AXI address bit 29 used for accessing the receive data buffers will be forced to the value stored in bit 29 of this register. When bit 2 is set, the AXI address bit 30 used for accessing the receive data buffers will be forced to the value stored in bit 30 of this register. When bit 3 is set, the AXI address bit 31 used for accessing the receive data buffers will be forced to the value stored in bit 31 of this register. When these bits are clear, the associated value stored in bits [31:28] have no effect on the AXI address used for receive data buffer accesses. Any changes to this register will be ignored while the DMA is currently processing a receive packet. It will only affect the next full packet to be written to system memory.

4.33. IEEE 1588 Timer Comparison Value Nanoseconds Register (ETHERNETn_tsu_nsec_cmp)

Description of IEEE 1588 Timer Comparison Value Nanoseconds register is shown.

REGISTER_NAME	ETHERNETn_tsu_nsec_cmp
OFFSET	0x0DC
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved		comparison_value[21:16]					
ACCESS_TYPE	R0,WX		R/W					
PROT_TYPE	Wp							
INITIAL_VALUE	0x0		0x00					

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	comparison_value[15:8]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	comparison_value[7:0]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:22] Reserved

Always read "0". Writing has no effect.

[bit21:0] comparison_value: IEEE 1588 Timer comparison value nanoseconds [21:0]

The register value is compared to bits [45:24] of the IEEE 1588 Timer count value (upper 22 bits of nanosecond value).

4.34. IEEE 1588 Timer Comparison Value Seconds Bottom Register (ETHERNETn_tsu_sec_cmp)

Description of IEEE 1588 Timer Comparison Value Seconds Bottom register is shown.

REGISTER_NAME	ETHERNETn_tsu_sec_cmp
OFFSET	0x0E0
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	comparison_value[31:24]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	comparison_value[23:16]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	comparison_value[15:8]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	comparison_value[7:0]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:0] comparison_value: IEEE 1588 Timer comparison value seconds [31:0]

The register value is compared to bits [31:0] of the IEEE 1588 Timer count value.

4.35. IEEE 1588 Timer Comparison Value Seconds Top Register (ETHERNETn_tsu_msb_sec_cmp)

Description of IEEE 1588 Timer Comparison Value Seconds Top register is shown.

REGISTER_NAME	ETHERNETn_tsu_msb_sec_cmp
OFFSET	0x0E4
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	comparison_value[15:8]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	comparison_value[7:0]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:16] Reserved

Always read "0". Writing has no effect.

[bit15:0] comparison_value: IEEE 1588 Timer comparison value seconds [47:32]

The register value is compared to bits [47:32] of the IEEE 1588 Timer count value.

4.36. PTP Event Frame Transmitted Seconds [47:32] Register (ETHERNETn_tsu_ptp_tx_msb_sec)

Description of PTP Event Frame Transmitted Seconds [47:32] register is shown.

REGISTER_NAME	ETHERNETn_tsu_ptp_tx_msb_sec
OFFSET	0x0E8
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	timer_seconds[15:8]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	timer_seconds[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:16] Reserved

Always read "0". Writing has no effect.

[bit15:0] timer_seconds: PTP Event Frame Transmitted Seconds [47:32]

The register is updated with the value that the IEEE 1588 Timer Seconds register held when the SFD of a PTP transmit primary event crosses the MII interface. The actual update occurs when the Ethernet MAC recognizes the frame as a PTP sync or Delay_Req frame. An interrupt is issued when the register is updated.

4.37. PTP Event Frame Received Seconds [47:32] Register (ETHERNETn_tsu_ptp_rx_msb_sec)

Description of PTP Event Frame Received Seconds [47:32] register is shown.

REGISTER_NAME	ETHERNETn_tsu_ptp_rx_msb_sec
OFFSET	0x0EC
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	timer_seconds[15:8]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	timer_seconds[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:16] Reserved

Always read "0". Writing has no effect.

[bit15:0] timer_seconds: PTP Event Frame Received Seconds [47:32]

The register is updated with the value that the IEEE 1588 Timer Seconds register held when the SFD of a PTP receive primary event crosses the MII interface. The actual update occurs when the Ethernet MAC recognizes the frame as a PTP Sync or Delay_Req frame. An interrupt is issued when the register is updated.

4.38. PTP Peer Event Frame Transmitted Seconds [47:32] Register (ETHERNETn_tsu_peer_tx_msb_sec)

Description of PTP Peer Event Frame Transmitted Seconds [47:32] register is shown.

REGISTER_NAME	ETHERNETn_tsu_peer_tx_msb_sec
OFFSET	0x0F0
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	timer_seconds[15:8]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	timer_seconds[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:16] Reserved

Always read "0". Writing has no effect.

[bit15:0] timer_seconds: PTP Peer Event Frame Transmitted Seconds [47:32]

The register is updated with the value that the IEEE 1588 Timer Seconds register held when the SFD of a PTP transmit peer event crosses the MII interface. The actual update occurs when the Ethernet MAC recognizes the frame as a PTP Pdelay_Req or Pdelay_Resp frame. An interrupt is issued when the register is updated.

4.39. PTP Peer Event Frame Received Seconds [47:32] Register (ETHERNETn_tsu_peer_rx_msb_sec)

Description of PTP Peer Event Frame Received Seconds [47:32] register is shown.

REGISTER_NAME	ETHERNETn_tsu_peer_rx_msb_sec
OFFSET	0x0F4
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	timer_seconds[15:8]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	timer_seconds[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:16] Reserved

Always read "0". Writing has no effect.

[bit15:0] timer_seconds: PTP Peer Event Frame Received Seconds [47:32]

The register is updated with the value that the IEEE 1588 Timer Seconds register held when the SFD of a PTP transmit peer event crosses the MII interface. The actual update occurs when the Ethernet MAC recognizes the frame as a PTP Pdelay_Req or Pdelay_Resp frame. An interrupt is issued when the register is updated.

4.40. Identification and Revision Register (ETHERNETn_revision_reg)

Description of Identification and Revision register is shown.

REGISTER_NAME	ETHERNETn_revision_reg
OFFSET	0x0FC
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	module_identification_number[15:8]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	(product specification)							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	module_identification_number[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	(product specification)							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	module_revision[15:8]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	(product specification)							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	module_revision[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	(product specification)							

[bit31:16] module_identification_number[15:0]: Module identification number

Module identification number for the Ethernet MAC.

[bit15:0] module_revision[15:0]: Module revision number

Module revision number for the Ethernet MAC.

4.41. Octets Transmitted Bottom Register (ETHERNETn_octets_txed_bottom)

Description of Octets Transmitted [31:0] register is shown. When reading the Octets Transmitted registers, bits [31:0] (ETHERNETn_octets_txed_bottom) should be read prior to bits [47:32] (ETHERNETn_octets_txed_top) to ensure reliable operation.

REGISTER_NAME	ETHERNETn_octets_txed_bottom
OFFSET	0x100
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	count[31:24]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	count[23:16]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	count[15:8]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	count[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:0] count: Transmitted octets in frame without errors [31:0]

The number of octets transmitted in valid frames of any type. This counter is 48-bits, and is read through two registers (ETHERNETn_octets_txed_top and ETHERNETn_octets_txed_bottom). This count does not include octets from automatically generated pause frames.

4.42. Octets Transmitted Top Register (ETHERNETn_octets_txed_top)

Description of Octets Transmitted [47:32] register is shown. When reading the Octets Transmitted registers, bits [31:0] (ETHERNETn_octets_txed_bottom) should be read prior to bits [47:32] (ETHERNETn_octets_txed_top) to ensure reliable operation.

REGISTER_NAME	ETHERNETn_octets_txed_top
OFFSET	0x104
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	count[47:40]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	count[39:32]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:16] Reserved

Always read "0". Writing has no effect.

[bit15:0] count: Transmitted octets in frame without errors [47:32]

The number of octets transmitted in valid frames of any type. This counter is 48-bits, and is read through two registers (ETHERNETn_octets_txed_top and ETHERNETn_octets_txed_bottom). This count does not include octets from automatically generated pause frames.

4.43. Frames Transmitted Register (ETHERNETn_frames_txed_ok)

Description of Frames Transmitted register is shown.

REGISTER_NAME	ETHERNETn_frames_txed_ok
OFFSET	0x108
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	count[31:24]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	count[23:16]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	count[15:8]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	count[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:0] count: Frames transmitted without error

A 32-bit register counting the number of frames successfully transmitted, i.e. no under run and not too many retries. Excludes pause frames.

4.44. Broadcast Frames Transmitted Register (ETHERNETn_broadcast_txed)

Description of Broadcast Frames Transmitted register is shown.

REGISTER_NAME	ETHERNETn_broadcast_txed
OFFSET	0x10C
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	count[31:24]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	count[23:16]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	count[15:8]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	count[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:0] count: Broadcast frames transmitted without error

A 32-bit register counting the number of broadcast frames successfully transmitted without error, i.e. no under run and not too many retries. Excludes pause frames.

4.45. Multicast Frames Transmitted Register (ETHERNETn_multicast_txed)

Description of Multicast Frames Transmitted register is shown.

REGISTER_NAME	ETHERNETn_multicast_txed
OFFSET	0x110
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	count[31:24]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	count[23:16]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	count[15:8]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	count[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:0] count: Multicast frames transmitted without error

A 32-bit register counting the number of multicast frames successfully transmitted without error, i.e. no under run and not too many retries. Excludes pause frames.

4.46. Pause Frames Transmitted Register (ETHERNETn_pause_frames_txed)

Description of Pause Frames Transmitted register is shown.

REGISTER_NAME	ETHERNETn_pause_frames_txed
OFFSET	0x114
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	count[15:8]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	count[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:16] Reserved

Always read "0". Writing has no effect.

[bit15:0] count: Transmitted pause frames

A 16-bit register counting the number of pause frames transmitted. Only pause frames triggered by the register interface or through the external pause pins are counted as pause frames. Pause frames received through the External FIFO Interface are counted in the frames transmitted counter.

4.47. 64 Byte Frames Transmitted Register (ETHERNETn_frames_txed_64)

Description of 64 Byte Frames Transmitted register is shown.

REGISTER_NAME	ETHERNETn_frames_txed_64
OFFSET	0x118
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	count[31:24]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	count[23:16]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	count[15:8]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	count[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:0] count: 64 byte frames transmitted without error

A 32-bit register counting the number of 64 byte frames successfully transmitted without error, i.e. no under run and not too many retries.

4.48. 65 To 127 Byte Frames Transmitted Register (ETHERNETn_frames_txed_65)

Description of 65 To 127 Byte Frames Transmitted register is shown.

REGISTER_NAME	ETHERNETn_frames_txed_65
OFFSET	0x11C
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	count[31:24]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	count[23:16]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	count[15:8]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	count[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:0] count: 65 to 127 byte frames transmitted without error

A 32-bit register counting the number of 65 to 127 byte frames successfully transmitted without error, i.e. no under run and not too many retries.

4.49. 128 To 255 Byte Frames Transmitted Register (ETHERNETn_frames_txed_128)

Description of 128 To 255 Byte Frames Transmitted register is shown.

REGISTER_NAME	ETHERNETn_frames_txed_128
OFFSET	0x120
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	count[31:24]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	count[23:16]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	count[15:8]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	count[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:0] count: 128 to 255 byte frames transmitted without error

A 32-bit register counting the number of 128 to 255 byte frames successfully transmitted without error, i.e. no under run and not too many retries.

4.50. 256 To 511 Byte Frames Transmitted Register (ETHERNETn_frames_txed_256)

Description of 256 To 511 Byte Frames Transmitted register is shown.

REGISTER_NAME	ETHERNETn_frames_txed_256
OFFSET	0x124
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	count[31:24]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	count[23:16]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	count[15:8]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	count[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:0] count: 256 to 511 byte frames transmitted without error

A 32-bit register counting the number of 256 to 511 byte frames successfully transmitted without error, i.e. no under run and not too many retries.

4.51. 512 To 1023 Byte Frames Transmitted Register (ETHERNETn_frames_txed_512)

Description of 512 To 1023 Byte Frames Transmitted register is shown.

REGISTER_NAME	ETHERNETn_frames_txed_512
OFFSET	0x128
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	count[31:24]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	count[23:16]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	count[15:8]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	count[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:0] count: 512 to 1023 byte frames transmitted without error

A 32-bit register counting the number of 512 to 1023 byte frames successfully transmitted without error, i.e. no under run and not too many retries.

4.52. 1024 To 1518 Byte Frames Transmitted Register (ETHERNETn_frames_txed_1024)

Description of 1024 To 1518 Byte Frames Transmitted register is shown.

REGISTER_NAME	ETHERNETn_frames_txed_1024
OFFSET	0x12C
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	count[31:24]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	count[23:16]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	count[15:8]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	count[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:0] count: 1024 to 1518 byte frames transmitted without error

A 32-bit register counting the number of 1024 to 1518 byte frames successfully transmitted without error, i.e. no under run and not too many retries.

4.53. Greater Than 1518 Byte Frames Transmitted Register (ETHERNETn_frames_txed_1519)

Description of Greater Than 1518 Byte Frames Transmitted register is shown.

REGISTER_NAME	ETHERNETn_frames_txed_1519
OFFSET	0x130
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	count[31:24]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	count[23:16]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	count[15:8]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	count[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:0] count: Greater than 1518 Byte frames transmitted without error

A 32-bit register counting the number of 1518 or above byte frames successfully transmitted without error, i.e. no under run and not too many retries.

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	count[23:16]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	count[15:8]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	count[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:0] count: Greater than 1518 Byte frames transmitted without error

A 32-bit register counting the number of 1518 or above byte frames successfully transmitted without error, i.e. no under run and not too many retries.

4.54. Transmit Under Runs Register (ETHERNETn_tx_underruns)

Description of Transmit Under Runs register is shown.

REGISTER_NAME	ETHERNETn_tx_underruns
OFFSET	0x134
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved						count[9:8]	
ACCESS_TYPE	R0,WX						R,WX	
PROT_TYPE	Wp							
INITIAL_VALUE	0x00						0x0	

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	count[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:10] Reserved

Always read "0". Writing has no effect.

[bit9:0] count: Transmit under runs

A 10-bit register counting the number of frames not transmitted due to a transmit under run. If this register is incremented then no other statistics register is incremented.

4.55. Single Collision Frames Register (ETHERNETn_single_collisions)

Description of Single Collision Frames register is shown.

REGISTER_NAME	ETHERNETn_single_collisions
OFFSET	0x138
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved						count[17:16]	
ACCESS_TYPE	R0,WX						R,WX	
PROT_TYPE	Wp							
INITIAL_VALUE	0x00						0x0	

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	count[15:8]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	count[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:18] Reserved

Always read "0". Writing has no effect.

[bit17:0] count: Single collision frames

An 18-bit register counting the number of frames experiencing a single collision before being successfully transmitted, i.e. no under run.

4.56. Multiple Collision Frames Register (ETHERNETn_multiple_collisions)

Description of Multiple Collision Frames register is shown.

REGISTER_NAME	ETHERNETn_multiple_collisions
OFFSET	0x13C
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved						count[17:16]	
ACCESS_TYPE	R0,WX						R,WX	
PROT_TYPE	Wp							
INITIAL_VALUE	0x00						0x0	

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	count[15:8]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	count[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:18] Reserved

Always read "0". Writing has no effect.

[bit17:0] count: Multiple collision frames

An 18-bit register counting the number of frames experiencing between two and fifteen collisions prior to being successfully transmitted, i.e. no under run and not too many retries.

4.57. Excessive Collisions Register (ETHERNETn_excessive_collisions)

Description of Excessive Collisions register is shown.

REGISTER_NAME	ETHERNETn_excessive_collisions
OFFSET	0x140
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved						count[9:8]	
ACCESS_TYPE	R0,WX						R,WX	
PROT_TYPE	Wp							
INITIAL_VALUE	0x00						0x0	

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	count[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:10] Reserved

Always read "0". Writing has no effect.

[bit9:0] count: Excessive collision frames

A 10-bit register counting the number of frames that failed to be transmitted because they experienced 16 collisions.

4.58. Late Collisions Register (ETHERNETn_late_collisions)

Description of Late Collisions register is shown.

REGISTER_NAME	ETHERNETn_late_collisions
OFFSET	0x144
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved						count[9:8]	
ACCESS_TYPE	R0,WX						R,WX	
PROT_TYPE	Wp							
INITIAL_VALUE	0x00						0x0	

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	count[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:10] Reserved

Always read "0". Writing has no effect.

[bit9:0] count: Late collision frames

A 10-bit register counting the number of late collision occurring after the slot time (512 bits) has expired. In 10/100 MBit/s mode, late collisions are counted twice i.e. both as a collision and a late collision.

4.59. Deferred Transmission Frames Register (ETHERNETn_deferred_frames)

Description of Deferred Transmission Frames register is shown.

REGISTER_NAME	ETHERNETn_deferred_frames
OFFSET	0x148
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved						count[17:16]	
ACCESS_TYPE	R0,WX						R,WX	
PROT_TYPE	Wp							
INITIAL_VALUE	0x00						0x0	

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	count[15:8]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	count[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:18] Reserved

Always read "0". Writing has no effect.

[bit17:0] count: Deferred transmission frames

An 18-bit register counting the number of frames experiencing deferral due to carrier sense being active on their first attempt at transmission. Frames involved in any collision are not counted nor are frames that experienced a transmit under run.

4.60. Carrier Sense Errors Register (ETHERNETn_crs_errors)

Description of Carrier Sense Errors register is shown.

REGISTER_NAME	ETHERNETn_crs_errors
OFFSET	0x14C
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved						count[9:8]	
ACCESS_TYPE	R0,WX						R,WX	
PROT_TYPE	Wp							
INITIAL_VALUE	0x00						0x0	

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	count[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:10] Reserved

Always read "0". Writing has no effect.

[bit9:0] count: Carrier sense errors

A 10-bit register counting the number of frames transmitted where carrier sense was not seen during transmission or where carrier sense was de-asserted after being asserted in a transmit frame without collision (no under run). Only incremented in half duplex mode. The only effect of a carrier sense error is to increment this register. The behaviour of the other statistics registers is unaffected by the detection of a carrier sense error.

4.61. Octets Receive Bottom Register (ETHERNETn_octets_rxed_bottom)

Description of Octets Receive [31:0] register is shown. When reading the Octets Receive registers, bits [31:0] (ETHERNETn_octets_rxed_bottom) should be read prior to bits [47:32] (ETHERNETn_octets_rxed_top) to ensure reliable operation.

REGISTER_NAME	ETHERNETn_octets_rxed_bottom
OFFSET	0x150
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	count[31:24]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	count[23:16]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	count[15:8]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	count[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:0] count: Received octets in frame without errors [31:0]

The number of octets received in valid frames of any type. This counter is 48-bits, and is read through two registers (ETHERNETn_octets_rxed_top and ETHERNETn_octets_rxed_bottom). This count does not include octets from pause frames, and is only incremented if the frame is successfully filtered and copied to memory.

4.62. Octets Receive TOP Register (ETHERNETn_octets_rxed_top)

Description of Octets Receive [47:32] register is shown. When reading the Octets Receive registers, bits [31:0] (ETHERNETn_octets_rxed_bottom) should be read prior to bits [47:32] (ETHERNETn_octets_rxed_top) to ensure reliable operation.

REGISTER_NAME	ETHERNETn_octets_rxed_top
OFFSET	0x154
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	count[47:40]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	count[39:32]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:16] Reserved

Always read "0". Writing has no effect.

[bit15:0] count: Transmitted octets in frame without errors [47:32]

The number of octets received in valid frames of any type. This counter is 48-bits, and is read through two registers (ETHERNETn_octets_rxed_top and ETHERNETn_octets_rxed_bottom). This count does not include octets from pause frames, and is only incremented if the frame is successfully filtered and copied to memory.

4.63. Frames Received Register (ETHERNETn_frames_rxed_ok)

Description of Frames Received register is shown.

REGISTER_NAME	ETHERNETn_frames_rxed_ok
OFFSET	0x158
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	count[31:24]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	count[23:16]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	count[15:8]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	count[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:0] count: Frames received without error

A 32-bit register counting the number of frames successfully received. Excludes pause frames, and is only incremented if the frame is successfully filtered and copied to memory.

4.64. Broadcast Frames Received Register (ETHERNETn_broadcast_rxed)

Description of Broadcast Frames Received register is shown.

REGISTER_NAME	ETHERNETn_broadcast_rxed
OFFSET	0x15C
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	count[31:24]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	count[23:16]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	count[15:8]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	count[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:0] count: Broadcast frames received without error

A 32-bit register counting the number of broadcast frames successfully received without error. Excludes pause frames, and is only incremented if the frame is successfully filtered and copied to memory.

4.65. Multicast Frames Received Register (ETHERNETn_multicast_rxed)

Description of Multicast Frames Received register is shown.

REGISTER_NAME	ETHERNETn_multicast_rxed
OFFSET	0x160
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	count[31:24]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	count[23:16]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	count[15:8]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	count[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:0] count: Multicast frames received without error

A 32-bit register counting the number of multicast frames successfully received without error. Excludes pause frames, and is only incremented if the frame is successfully filtered and copied to memory.

4.66. Pause Frames Received Register (ETHERNETn_pause_frames_rxed)

Description of Pause Frames Received register is shown.

REGISTER_NAME	ETHERNETn_pause_frames_rxed
OFFSET	0x164
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	count[15:8]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	count[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:16] Reserved

Always read "0". Writing has no effect.

[bit15:0] count: Received pause frames

A 16-bit register counting the number of pause frames received without error.

4.67. 64 Byte Frames Received Register (ETHERNETn_frames_rxed_64)

Description of 64 Byte Frames Received register is shown.

REGISTER_NAME	ETHERNETn_frames_rxed_64
OFFSET	0x168
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	count[31:24]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	count[23:16]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	count[15:8]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	count[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:0] count: 64 Byte frames received without error

A 32-bit register counting the number of 64 byte frames successfully received without error. Excludes pause frames, and is only incremented if the frame is successfully filtered and copied to memory.

4.68. 65 To 127 Byte Frames Received Register (ETHERNETn_frames_rxed_65)

Description of 65 To 127 Byte Frames Received register is shown.

REGISTER_NAME	ETHERNETn_frames_rxed_65
OFFSET	0x16C
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	count[31:24]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	count[23:16]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	count[15:8]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	count[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:0] count: 65 to 127 Byte frames received without error

A 32-bit register counting the number of 65 to 127 byte frames successfully received without error. Excludes pause frames, and is only incremented if the frame is successfully filtered and copied to memory.

4.69. 128 To 255 Byte Frames Received Register (ETHERNETn_frames_rxed_128)

Description of 128 To 255 Byte Frames Received register is shown.

REGISTER_NAME	ETHERNETn_frames_rxed_128
OFFSET	0x170
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	count[31:24]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	count[23:16]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	count[15:8]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	count[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:0] count: 128 to 255 Byte frames received without error

A 32-bit register counting the number of 128 to 255 byte frames successfully received without error. Excludes pause frames, and is only incremented if the frame is successfully filtered and copied to memory.

4.70. 256 To 511 Byte Frames Received Register (ETHERNETn_frames_rxed_256)

Description of 256 To 511 Byte Frames Received register is shown.

REGISTER_NAME	ETHERNETn_frames_rxed_256
OFFSET	0x174
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	count[31:24]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	count[23:16]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	count[15:8]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	count[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:0] count: 256 to 511 Byte frames received without error

A 32-bit register counting the number of 256 to 511 byte frames successfully received without error. Excludes pause frames, and is only incremented if the frame is successfully filtered and copied to memory.

4.71. 512 To 1023 Byte Frames Received Register (ETHERNETn_frames_rxed_512)

Description of 512 To 1023 Byte Frames Received register is shown.

REGISTER_NAME	ETHERNETn_frames_rxed_512
OFFSET	0x178
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	count[31:24]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	count[23:16]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	count[15:8]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	count[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:0] count: 512 to 1023 Byte frames received without error

A 32-bit register counting the number of 512 to 1023 byte frames successfully received without error. Excludes pause frames, and is only incremented if the frame is successfully filtered and copied to memory.

4.72. 1024 To 1518 Byte Frames Received Register (ETHERNETn_frames_rxed_1024)

Description of 1024 To 1518 Byte Frames Received register is shown.

REGISTER_NAME	ETHERNETn_frames_rxed_1024
OFFSET	0x17C
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	count[31:24]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	count[23:16]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	count[15:8]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	count[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:0] count: 1024 to 1518 Byte frames received without error

A 32-bit register counting the number of 1024 to 1518 byte frames successfully received without error. Excludes pause frames, and is only incremented if the frame is successfully filtered and copied to memory.

4.73. Greater than 1518 Byte Frames Received Register (ETHERNETn_frames_rxed_1519)

Description of Greater than 1518 Byte Frames Received register is shown.

REGISTER_NAME	ETHERNETn_frames_rxed_1519
OFFSET	0x180
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	count[31:24]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	count[23:16]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	count[15:8]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	count[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:0] count: 1519 to maximum byte frames received without error

A 32-bit register counting the number of 1519 byte or above frames successfully received without error. Maximum frame size is determined by the Network Configuration register bit 8 (1536 maximum frame size) or bit 3 (jumbo frame size). Excludes pause frames, and is only incremented if the frame is successfully filtered and copied to memory.

4.74. Undersized Frames Received Register (ETHERNETn_undersized_frames)

Description of Undersized Frames Received register is shown.

REGISTER_NAME	ETHERNETn_undersized_frames
OFFSET	0x184
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved						count[9:8]	
ACCESS_TYPE	R0,WX						R,WX	
PROT_TYPE	Wp							
INITIAL_VALUE	0x00						0x0	

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	count[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:10] Reserved

Always read "0". Writing has no effect.

[bit9:0] count: Undersized frames received

A 10-bit register counting the number of frames received less than 64 bytes in length (10/100 MBit/s mode) that do not have either a CRC error or an alignment error.

4.75. Oversize Frames Received Register (ETHERNETn_excessive_rx_length)

Description of Oversize Frames Received register is shown.

REGISTER_NAME	ETHERNETn_excessive_rx_length
OFFSET	0x188
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved						count[9:8]	
ACCESS_TYPE	R0,WX						R,WX	
PROT_TYPE	Wp							
INITIAL_VALUE	0x00						0x0	

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	count[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:10] Reserved

Always read "0". Writing has no effect.

[bit9:0] count: Oversize frames received

A 10-bit register counting the number of frames received exceeding 1518 bytes (1536 bytes if bit 8 is set in Network Configuration register, 10240 bytes if bit 3 is set in the Network Configuration register) in length but do not have either a CRC error, an alignment error nor a receive symbol error.

4.76. Jabbers Received Register (ETHERNETn_rx_jabbers)

Description of Jabbers Received register is shown.

REGISTER_NAME	ETHERNETn_rx_jabbers
OFFSET	0x18C
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved						count[9:8]	
ACCESS_TYPE	R0,WX						R,WX	
PROT_TYPE	Wp							
INITIAL_VALUE	0x00						0x0	

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	count[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:10] Reserved

Always read "0". Writing has no effect.

[bit9:0] count: Jabbers received

A 10-bit register counting the number of frames received exceeding 1518 bytes (1536 if bit 8 set in Network Configuration register, 10240 bytes if bit 3 is set in the Network Configuration register) in length and have either a CRC error, an alignment error or a receive symbol error.

4.77. Frame Check Sequence Errors Register (ETHERNETn_fcs_errors)

Description of Frame Check Sequence Errors register is shown.

REGISTER_NAME	ETHERNETn_fcs_errors
OFFSET	0x190
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved						count[9:8]	
ACCESS_TYPE	R0,WX						R,WX	
PROT_TYPE	Wp							
INITIAL_VALUE	0x00						0x0	

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	count[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:10] Reserved

Always read "0". Writing has no effect.

[bit9:0] count: Frame check sequence errors

A 10-bit register counting frames that are an integral number of bytes, have bad CRC and are between 64 and 1518 bytes in length (1536 if bit 8 set in Network Configuration register, 10240 bytes if bit 3 is set in the Network Configuration register). This register is also incremented if a symbol error is detected and the frame is of valid length and has an integral number of bytes. This register is incremented for a frame with bad FCS, regardless of whether it is copied to memory due to ignore FCS mode being enabled in bit 26 of the Network Configuration register.

4.78. Length Field Frame Errors Register (ETHERNETn_rx_length_errors)

Description of Length Field Frame Errors register is shown.

REGISTER_NAME	ETHERNETn_rx_length_errors
OFFSET	0x194
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved						count[9:8]	
ACCESS_TYPE	R0,WX						R,WX	
PROT_TYPE	Wp							
INITIAL_VALUE	0x00						0x0	

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	count[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:10] Reserved

Always read "0". Writing has no effect.

[bit9:0] count: Length field frame errors

A 10-bit register counting frames received that have a measured length shorter than that extracted from the length field (bytes 13 and 14). This condition is only counted if the value of the length field is less than 0x0600 (1536)), the frame is not of excessive length and checking is enabled through bit 16 of the Network Configuration register.

4.79. Receive Symbol_Errors Register (ETHERNETn_rx_symbol_errors)

Description of Receive Symbol_Errors register is shown.

REGISTER_NAME	ETHERNETn_rx_symbol_errors
OFFSET	0x198
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved						count[9:8]	
ACCESS_TYPE	R0,WX						R,WX	
PROT_TYPE	Wp							
INITIAL_VALUE	0x00						0x0	

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	count[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:10] Reserved

Always read "0". Writing has no effect.

[bit9:0] count: Receive symbol errors

A 10-bit register counting the number of frames that had `RX_ER` asserted during reception. For 10/100 MBit/s mode symbol errors are counted regardless of frame length checks. Receive symbol errors will also be counted as an FCS or alignment error if the frame is between 64 and 1518 bytes (1536 bytes if bit 8 is set in the Network Configuration register, 10240 bytes if bit 3 is set in the Network Configuration register). If the frame is larger it will be recorded as a jabber error.

4.80. Alignment Errors Register (ETHERNETn_alignment_errors)

Description of Alignment Errors register is shown.

REGISTER_NAME	ETHERNETn_alignment_errors
OFFSET	0x19C
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved						count[9:8]	
ACCESS_TYPE	R0,WX						R,WX	
PROT_TYPE	Wp							
INITIAL_VALUE	0x00						0x0	

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	count[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:10] Reserved

Always read "0". Writing has no effect.

[bit9:0] count: Alignment errors

A 10-bit register counting the number of frames that are not an integral number of bytes long and have bad CRC when their length is truncated to an integral number of bytes and are between 64 and 1518 bytes in length (1536 if bit 8 set in Network Configuration register, 10240 bytes if bit 3 is set in the Network Configuration register). This register is also incremented if a symbol error is detected and the frame is of valid length and does not have an integral number of bytes.

4.81. Receive Resource Errors Register (ETHERNETn_rx_resource_errors)

Description of Receive Resource Errors register is shown.

REGISTER_NAME	ETHERNETn_rx_resource_errors
OFFSET	0x1A0
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved						count[17:16]	
ACCESS_TYPE	R0,WX						R,WX	
PROT_TYPE	Wp							
INITIAL_VALUE	0x00						0x0	

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	count[15:8]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	count[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:18] Reserved

Always read "0". Writing has no effect.

[bit17:0] count: Receive resource errors

An 18-bit register counting the number of frames that were successfully received by the Ethernet MAC (correct address matched frame and adequate slot time) but could not be copied to memory because no receive buffer was available. This occurs when the Ethernet MAC reads a buffer descriptor with its ownership (or used) bit set.

4.82. Receive Over Runs Register (ETHERNETn_rx_overruns)

Description Receive Over Runs register is shown.

REGISTER_NAME	ETHERNETn_rx_overruns
OFFSET	0x1A4
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved						count[9:8]	
ACCESS_TYPE	R0,WX						R,WX	
PROT_TYPE	Wp							
INITIAL_VALUE	0x00						0x0	

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	count[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:10] Reserved

Always read "0". Writing has no effect.

[bit9:0] count: Receive over runs

A 10-bit register counting the number of frames that are address recognized but were not copied to memory due to a receive over run.

4.83. IP Header Checksum Errors Register (ETHERNETn_rx_ip_ck_errors)

Description of IP Header Checksum Errors register is shown.

REGISTER_NAME	ETHERNETn_rx_ip_ck_errors
OFFSET	0x1A8
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	count[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:8] Reserved

Always read "0". Writing has no effect.

[bit7:0] count: IP header checksum errors

An 8-bit register counting the number of frames discarded due to an incorrect IP header checksum, but are between 64 and 1518 bytes (1536 bytes if bit 8 is set in the Network Configuration register or 10240 bytes if bit 3 is in the Network Configuration register) and do not have a CRC error, an alignment error, nor a symbol error.

4.84. TCP Checksum Errors Register (ETHERNETn_rx_tcp_ck_errors)

Description of TCP Checksum Errors register is shown.

REGISTER_NAME	ETHERNETn_rx_tcp_ck_errors
OFFSET	0x1AC
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	count[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:8] Reserved

Always read "0". Writing has no effect.

[bit7:0] count: TCP checksum errors

An 8-bit register counting the number of frames discarded due to an incorrect TCP checksum, but are between 64 and 1518 bytes (1536 bytes if bit 8 is set in the Network Configuration register or 10240 bytes if bit 3 is in the Network Configuration register) and do not have a CRC error, an alignment error, nor a symbol error.

4.85. UDP Checksum Errors Register (ETHERNETn_rx_udp_ck_errors)

Description of UDP Checksum Errors register is shown.

REGISTER_NAME	ETHERNETn_rx_udp_ck_errors
OFFSET	0x1B0
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	count[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:8] Reserved

Always read "0". Writing has no effect.

[bit7:0] count: UDP checksum errors

An 8-bit register counting the number of frames discarded due to an incorrect UDP checksum, but are between 64 and 1518 bytes (1536 bytes if bit 8 is set in the Network Configuration register or 10240 bytes if bit 3 is in the Network Configuration register) and do not have a CRC error, an alignment error, nor a symbol error.

4.86. Receive DMA Flushed Packets Register (ETHERNETn_auto_flushed_pkts)

Description of Receive DMA Flushed Packets register is shown.

REGISTER_NAME	ETHERNETn_auto_flushed_pkts
OFFSET	0x1B4
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	count[15:8]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	count[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:16] Reserved

Always read "0". Writing has no effect.

[bit15:0] count: Flushed RX packets counter

A 16-bit register counting the number of frames that have been flushed from the receive SRAM based RX Packet Buffer Memory due to one of the following reasons:

1. When partial store and forward mode is enabled or bit 24 of the DMA Configuration register is enabled, a packet is received while there is no AMBA AXI resource.
2. When partial store and forward mode is enabled and an AMBA AXI error is encountered while writing the packet data to system memory. When bit 18 of the Network Control register (software action to flush a packet from the head of the PBUF queue) is pulsed and the **Ethernet MAC** DMA is not currently busy.

4.87. IEEE 1588 Timer Increment Sub Nanoseconds Register (ETHERNETn_tsu_timer_incr_sub_nsec)

Description of IEEE 1588 Timer Increment Sub Nanoseconds register is shown.

REGISTER_NAME	ETHERNETn_tsu_timer_incr_sub_nsec
OFFSET	0x1BC
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	count[15:8]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	count[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:16] Reserved

Always read "0". Writing has no effect.

[bit15:0] count: IEEE 1588 Timer Increment [15:0]

Lower significant bits of IEEE 1588 Timer Increment [15:0] register giving a 24-bit timer_increment counter. These bits are the sub-ns value which the IEEE 1588 Timer will increment each clock cycle. Bit n = 2(n-16) nsec giving a resolution of approximately 15.2E-15 sec.

4.88. IEEE 1588 Timer Seconds [47:32] Register (ETHERNETn_tsu_timer_msb_sec)

Description of IEEE 1588 Timer Seconds [47:32] register is shown.

REGISTER_NAME	ETHERNETn_tsu_timer_msb_sec
OFFSET	0x1C0
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	timer[15:8]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	timer[7:0]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:16] Reserved

Always read "0". Writing has no effect.

[bit15:0] timer: IEEE 1588 Timer Seconds [47:32]

IEEE 1588 Timer value (s). Most significant 16 bits of seconds timer count. The register is writeable. The 48-bit counter increments by one when the IEEE 1588 Timer nanoseconds counter counts to one second. It may also be incremented or decremented when the IEEE 1588 Timer Adjust register is written (if decremented from zero the 48-bit combined count would roll back to 0xFFFFFFFFFFFF).
Note: The value of this register is used only when the lower 32-bit register is written to. This is to ensure a single update of the 48-bit seconds value.

4.89. IEEE 1588 Timer Seconds [31:0] Register (ETHERNETn_tsu_timer_sec)

Description of IEEE 1588 Timer Seconds [31:0] register is shown.

REGISTER_NAME	ETHERNETn_tsu_timer_sec
OFFSET	0x1D0
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	timer[31:24]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	timer[23:16]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	timer[15:8]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	timer[7:0]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:0] timer: IEEE 1588 Timer Seconds [31:0]

IEEE 1588 Timer value (s). Least significant 32 bits of seconds timer count. This register is writeable. The 48-bit counter increments by one when the IEEE 1588 Timer Nanoseconds counter counts to one second. It may also be incremented or decremented when the IEEE 1588 Timer Adjust register is written (if decremented from zero the 48-bit combined count would roll back to 0xFFFFFFFFFFFF).

4.90. IEEE 1588 Timer Nanoseconds Register (ETHERNETn_tsu_timer_nsec)

Description of IEEE 1588 Timer Nanoseconds register is shown.

REGISTER_NAME	ETHERNETn_tsu_timer_nsec
OFFSET	0x1D4
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved		timer[29:24]					
ACCESS_TYPE	R0,WX		R/W					
PROT_TYPE	Wp							
INITIAL_VALUE	0x0		0x00					

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	timer[23:16]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	timer[15:8]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	timer[7:0]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:30] Reserved

Always read "0". Writing has no effect.

[bit29:0] timer: IEEE 1588 Timer nanoseconds

This register is writeable. It can also be adjusted by writes to the IEEE 1588 Timer Adjust register. It increments by the value of the IEEE 1588 Timer Increment register each clock cycle (if this register is close to zero and a write to the IEEE 1588 Timer Adjust register causes a decrement the seconds register will be decremented if necessary and the nanoseconds register will roll back to 9999999xx(dec)).

4.91. IEEE 1588 Timer Adjust Register (ETHERNETn_tsu_timer_adjust)

Description of IEEE 1588 Timer Adjust register is shown.

REGISTER_NAME	ETHERNETn_tsu_timer_adjust
OFFSET	0x1D8
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	add_subtract	Reserved	increment_value[29:24]					
ACCESS_TYPE	R0,W	R0,WX	R0,W					
PROT_TYPE	Wp							
INITIAL_VALUE	0	0	0x00					

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	increment_value[23:16]							
ACCESS_TYPE	R0,W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	increment_value[15:8]							
ACCESS_TYPE	R0,W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	increment_value[7:0]							
ACCESS_TYPE	R0,W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31] add_subtract:
Write as "1" to subtract from the IEEE 1588 Timer. Write as "0" to add to it.

[bit30] Reserved
Always read "0". Writing has no effect.

[bit29:0] increment_value: IEEE 1588 Timer increment value
The number of nanoseconds to increment or decrement the IEEE 1588 Timer Nanoseconds register. If necessary the IEEE 1588 Timer Seconds register will be incremented or decremented.

4.92. IEEE 1588 Timer Increment Register (ETHERNETn_tsu_timer_incr)

Description of IEEE 1588 Timer Increment register is shown.

REGISTER_NAME	ETHERNETn_tsu_timer_incr
OFFSET	0x1DC
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	num_incs[7:0]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	alt_count[7:0]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	count[7:0]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:24] Reserved
Always read "0". Writing has no effect.

[bit23:16] num_incs: Number of incs before alt inc
The number of increments after which the alternative increment is used.

[bit15:8] alt_count: Alternative count of nanoseconds
Alternative count of nanoseconds by which the IEEE 1588 Timer Nanoseconds register will be incremented each clock cycle.

[bit7:0] count: Count of nanoseconds
A count of nanoseconds by which the IEEE 1588 Timer Nanoseconds register will be incremented each clock cycle.

4.93. PTP Event Frame Transmitted Seconds [31:0] Register (ETHERNETn_tsu_ptp_tx_sec)

Description of PTP Event Frame Transmitted Seconds [31:0] register is shown.

REGISTER_NAME	ETHERNETn_tsu_ptp_tx_sec
OFFSET	0x1E0
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	timer[31:24]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	timer[23:16]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	timer[15:8]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	timer[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:0] timer: PTP Event Frame Transmitted Seconds [31:0]

The register is updated with the value that the IEEE 1588 Timer Nanoseconds register held when the SFD of a PTP transmit primary event crosses the MII interface. The actual update occurs when the Ethernet MAC recognizes the frame as a PTP Sync or Delay_Req frame. An interrupt is issued when the register is updated.

4.94. PTP Event Frame Transmitted Nanoseconds Register (ETHERNETn_tsu_ptp_tx_nsec)

Description of PTP Event Frame Transmitted Nanoseconds register is shown.

REGISTER_NAME	ETHERNETn_tsu_ptp_tx_nsec
OFFSET	0x1E4
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved		timer[29:24]					
ACCESS_TYPE	R0,WX		R,WX					
PROT_TYPE	Wp							
INITIAL_VALUE	0x0		0x00					

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	timer[23:16]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	timer[15:8]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	timer[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:30] Reserved

Always read "0". Writing has no effect.

[bit29:0] timer: PTP Event Frame Transmitted Nanoseconds

The register is updated with the value that the IEEE 1588 Timer Nanoseconds register held when the SFD of a PTP transmit primary event crosses the MII interface. The actual update occurs when the Ethernet MAC recognizes the frame as a PTP Sync or Delay_Req frame. An interrupt is issued when the register is updated.

4.95. PTP Event Frame Received Seconds [31:0] Register (ETHERNETn_tsu_ptp_rx_sec)

Description of PTP Event Frame Received Seconds [31:0] register is shown.

REGISTER_NAME	ETHERNETn_tsu_ptp_rx_sec
OFFSET	0x1E8
ACCESS_SIZE	W
MULTIPLE	1
NUMERIC_TYPE	-
OTHER	-

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	timer[31:24]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	timer[23:16]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	timer[15:8]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	timer[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:0] timer: PTP Event Frame Received Seconds [31:0]

The register is updated with the value that the IEEE 1588 Timer Nanoseconds register held when the SFD of a PTP receive primary event crosses the MII interface. The actual update occurs when the Ethernet MAC recognizes the frame as a PTP Sync or Delay_Req frame. An interrupt is issued when the register is updated.

4.96. PTP Event Frame Received Nanoseconds Register (ETHERNETn_tsu_ptp_rx_nsec)

Description of PTP Event Frame Received Nanoseconds register is shown.

REGISTER_NAME	ETHERNETn_tsu_ptp_rx_nsec
OFFSET	0x1EC
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved		timer[29:24]					
ACCESS_TYPE	R0,WX		R,WX					
PROT_TYPE	Wp							
INITIAL_VALUE	0x0		0x00					

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	timer[23:16]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	timer[15:8]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	timer[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:30] Reserved

Always read "0". Writing has no effect.

[bit29:0] timer: PTP Event Frame Received Nanoseconds

The register is updated with the value that the IEEE 1588 Timer Nanoseconds register held when the SFD of a PTP receive primary event crosses the MII interface. The actual update occurs when the Ethernet MAC recognizes the frame as a PTP Sync or Delay_Req frame. An interrupt is issued when the register is updated.

4.97. PTP Peer Event Frame Transmitted Seconds [31:0] Register (ETHERNETn_tsu_peer_tx_sec)

Description of PTP Peer Event Frame Transmitted Seconds [31:0] register is shown.

REGISTER_NAME	ETHERNETn_tsu_peer_tx_sec
OFFSET	0x1F0
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	timer[31:24]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	timer[23:16]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	timer[15:8]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	timer[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:0] timer: PTP Peer Event Frame Transmitted Seconds [31:0]

The register is updated with the value that the IEEE 1588 Timer Nanoseconds register held when the SFD of a PTP transmit peer event crosses the MII interface. The actual update occurs when the Ethernet MAC recognizes the frame as a PTP Pdelay_Req or Pdelay_Resp frame. An interrupt is issued when the register is updated.

4.98. PTP Peer Event Frame Transmitted Nanoseconds Register (ETHERNETn_tsu_peer_tx_nsec)

Description of PTP Peer Event Frame Transmitted Nanoseconds register is shown.

REGISTER_NAME	ETHERNETn_tsu_peer_tx_nsec
OFFSET	0x1F4
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved		timer[29:24]					
ACCESS_TYPE	R0,WX		R,WX					
PROT_TYPE	Wp							
INITIAL_VALUE	0x0		0x00					

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	timer[23:16]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	timer[15:8]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	timer[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:30] Reserved

Always read "0". Writing has no effect.

[bit29:0] timer: PTP Peer Event Frame Transmitted Nanoseconds

The register is updated with the value that the IEEE 1588 Timer Nanoseconds register held when the SFD of a PTP transmit peer event crosses the MII interface. The actual update occurs when the Ethernet MAC recognizes the frame as a PTP Pdelay_Req or Pdelay_Resp frame. An interrupt is issued when the register is updated.

4.99. PTP Peer Event Frame Received Seconds [31:0] Register (ETHERNETn_tsu_peer_rx_sec)

Description of PTP Peer Event Frame Received Seconds [31:0] register is shown.

REGISTER_NAME	ETHERNETn_tsu_peer_rx_sec
OFFSET	0x1F8
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	timer[31:24]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	timer[23:16]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	timer[15:8]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	timer[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:0] timer: PTP Peer Event Frame Received Seconds [31:0]

The register is updated with the value that the IEEE 1588 Timer Seconds register held when the SFD of a PTP receive peer event crosses the MII interface. The actual update occurs when the Ethernet MAC recognizes the frame as a PTP Pdelay_Req or Pdelay_Resp frame. An interrupt is issued when the register is updated.

4.100. PTP Peer Event Frame Received Nanoseconds Register (ETHERNETn_tsu_peer_rx_nsec)

Description of PTP Peer Event Frame Received Nanoseconds register is shown.

REGISTER_NAME	ETHERNETn_tsu_peer_rx_nsec
OFFSET	0x1FC
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved		timer[29:24]					
ACCESS_TYPE	R0,WX		R,WX					
PROT_TYPE	Wp							
INITIAL_VALUE	0x0		0x00					

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	timer[23:16]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	timer[15:8]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	timer[7:0]							
ACCESS_TYPE	R,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:30] Reserved

Always read "0". Writing has no effect.

[bit29:0] timer: PTP Peer Event Frame Received Nanoseconds

The register is updated with the value that the IEEE 1588 Timer Nanoseconds register held when the SFD of a PTP receive peer event crosses the MII interface. The actual update occurs when the Ethernet MAC recognizes the frame as a PTP Pdelay_Req or Pdelay_Resp frame. An interrupt is issued when the register is updated.

4.101. Transmit Pause Quantum 1 Register (ETHERNETn_tx_pause_quantum1)

Description of Transmit Pause Quantum 1 register is shown.

REGISTER_NAME	ETHERNETn_tx_pause_quantum1
OFFSET	0x260
ACCESS_SIZE	W
MULTIPLE	1:3
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	quantum_p3[15:8]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0xFF							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	quantum_p3[7:0]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0xFF							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	quantum_p2[15:8]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0xFF							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	quantum_p2[7:0]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0xFF							

[bit31:16] quantum_p3: Transmit pause quantum prio 3

Written with the pause quantum value for pause frame transmission of priority 3.

[bit15:0] quantum_p2: Transmit pause quantum prio 2

Written with the pause quantum value for pause frame transmission of priority 2.

4.102. Transmit Pause Quantum 2 Register (ETHERNETn_tx_pause_quantum2)

Description of Transmit Pause Quantum 2 register is shown.

REGISTER_NAME	ETHERNETn_tx_pause_quantum2
OFFSET	0x264
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	quantum_p5[15:8]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0xFF							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	quantum_p5[7:0]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0xFF							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	quantum_p4[15:8]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0xFF							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	quantum_p4[7:0]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0xFF							

[bit31:16] quantum_p5: Transmit pause quantum prio 5

Written with the pause quantum value for pause frame transmission of priority 5.

[bit15:0] quantum_p4: Transmit pause quantum prio 4

Written with the pause quantum value for pause frame transmission of priority 4.

4.103. Transmit Pause Quantum 3 Register (ETHERNETn_tx_pause_quantum3)

Description of Transmit Pause Quantum 3 register is shown.

REGISTER_NAME	ETHERNETn_tx_pause_quantum3
OFFSET	0x268
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	quantum_p7[15:8]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0xFF							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	quantum_p7[7:0]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0xFF							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	quantum_p6[15:8]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0xFF							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	quantum_p6[7:0]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0xFF							

[bit31:16] quantum_p7: Transmit pause quantum prio 7

Written with the pause quantum value for pause frame transmission of priority 7.

[bit15:0] quantum_p6: Transmit pause quantum prio 6

Written with the pause quantum value for pause frame transmission of priority 6.

4.104. Receive LPI Transitions Register (ETHERNETn_rx_lpi)

Description of Receive LPI Transitions register is shown.

REGISTER_NAME	ETHERNETn_rx_lpi
OFFSET	0x270
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	count[15:8]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	count[7:0]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:16] Reserved

Always read "0". Writing has no effect.

[bit15:0] count: Count of RX LPI transitions

A count of the number of times there is a transition from receiving normal idle to receiving low power idle. Cleared on read.

4.105. Received LPI Time Register (ETHERNETn_rx_lpi_time)

Description of Received LPI Time register is shown.

REGISTER_NAME	ETHERNETn_rx_lpi_time
OFFSET	0x274
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	lpi_time[23:16]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	lpi_time[15:8]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	lpi_time[7:0]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:24] Reserved

Always read "0". Writing has no effect.

[bit23:0] lpi_time: Time in LPI

This register increments once every 16 clock cycles when the LPI Indication bit 7 is set in the Network Status register. Cleared on read.

4.106. Transmit LPI Transitions Register (ETHERNETn_tx_lpi)

Description of Transmit LPI Transitions register is shown.

REGISTER_NAME	ETHERNETn_tx_lpi
OFFSET	0x278
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	count[15:8]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	count[7:0]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:16] Reserved

Always read "0". Writing has no effect.

[bit15:0] count: Count of TX LPI transitions

A count of the number of times the Enable LPI transmission bit 19 goes from low to high in the Network Control register.

4.107. Transmit LPI Time Register (ETHERNETn_tx_lpi_time)

Description of Transmit LPI Time register is shown.

REGISTER_NAME	ETHERNETn_tx_lpi_time
OFFSET	0x27C
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	lpi_time[23:16]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	lpi_time[15:8]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	lpi_time[7:0]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:24] Reserved

Always read "0". Writing has no effect.

[bit23:0] lpi_time: Time in LPI

This register increments once every 16 clock cycles when the Enable LPI transmission bit 19 is set in the Network Control register. Cleared on read.

4.108. Interrupt Status Queue i Register (ETHERNETn_int_status_qi) (i=1 to 3)

Description of Interrupt Status Queue i register is shown.

REGISTER_NAME	ETHERNETn_int_status_qi
OFFSET	(0x400 + ((i-1)*0x4))
ACCESS_SIZE	W
MULTIPLE	1:3
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved				resp_not_ok	receive_overrun	Reserved	
ACCESS_TYPE	R0,WX				R/W	R/W	R0,WX	
PROT_TYPE	Wp							
INITIAL_VALUE	0x0				0	0	0x0	

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	transmit_complete	amba_error	retry_limit_exceeded_or_late_collision	Reserved		rx_used_bit_read	receive_complete	Reserved
ACCESS_TYPE	R/W	R/W	R/W	R0,WX		R/W	R/W	R0,WX
PROT_TYPE	Wp							
INITIAL_VALUE	0	0	0	0x0		0	0	0

[bit31:12] Reserved

Always read "0". Writing has no effect.

[bit11] resp_not_ok: BRESP not OK

Bit	Description
0	Interrupt de-asserted.
1	Interrupt asserted. Set when the DMA block sees BRESP not OK.

[bit10] receive_overrun: Receive over run

Bit	Description
0	Interrupt de-asserted.
1	Interrupt asserted. Set when the receive over run status bit (ETHERNETn_receive_status[2]) gets set.

[bit9:8] Reserved

Always read "0". Writing has no effect.

[bit7] transmit_complete: Transmit complete

Bit	Description
0	Interrupt de-asserted.
1	Interrupt asserted. Set when a frame has been transmitted.

[bit6] amba_error: Transmit frame corruption due to AMBA AXI error

Bit	Description
0	Interrupt de-asserted.
1	Interrupt asserted. Set if an error occurs whilst midway through reading transmit frame from system memory, including BRESP errors and buffers exhausted mid frame.

[bit5] retry_limit_exceeded_or_late_collision: Retry limit exceeded

Bit	Description
0	Interrupt de-asserted.
1	Interrupt asserted. Transmit error.

[bit4:3] Reserved

Always read "0". Writing has no effect.

[bit2] rx_used_bit_read: RX used bit read

Bit	Description
0	Interrupt de-asserted.
1	Interrupt asserted. Set when a receive buffer descriptor is read with its used bit set.

[bit1] receive_complete: Receive complete

Bit	Description
0	Interrupt de-asserted.
1	Interrupt asserted. Set when a frame has been stored in memory.

[bit0] Reserved

Always read "0". Writing has no effect.

4.109. TX Buffer Queue 1 Base Address Register (ETHERNETn_transmit_q1_ptr)

This register holds the start address of the transmit buffer queue 1 (transmit buffers descriptor list). The transmit buffer queue base address register must be initialized before transmit is started through bit 9 of the Network Control register. Once transmission has started, any write to the TX Buffer Queue 1 Base Address register is illegal and therefore ignored. Note that due to clock boundary synchronization, it takes a maximum of four clock cycles from the writing of the transmit start bit before the transmitter is active. Writing to the TX Buffer Queue 1 Base Address register during this time may produce unpredictable results. Reading this register returns the location of the descriptor currently being accessed. Since the DMA handles two frames at once, this may not necessarily be pointing to the current frame being transmitted. In terms of AMBA AXI operation, the transmit descriptors are written to memory using a single 32-bit AXI access. Since the datapath is 64-bit wide, the transmit descriptors shall be aligned at 64-bit boundaries and each pair of 32-bit descriptors is read from memory using a single 64-bit AXI access.

REGISTER_NAME	ETHERNETn_transmit_q1_ptr
OFFSET	0x440
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	dma_tx_q_ptr[31:24]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	dma_tx_q_ptr[23:16]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	dma_tx_q_ptr[15:8]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	dma_tx_q_ptr[7:2]						Reserved	
ACCESS_TYPE	R/W						R0,WX	
PROT_TYPE	Wp							
INITIAL_VALUE	0x00						0x0	

[bit31:2] dma_tx_q_ptr: Transmit buffer queue base address

Start address of transmit buffer queue.

[bit1:0] Reserved

Always read "0". Writing has no effect.

4.110. TX Buffer Queue 2 Base Address Register (ETHERNETn_transmit_q2_ptr)

This register holds the start address of the transmit buffer queue 2 (transmit buffers descriptor list). The transmit buffer queue base address register must be initialized before transmit is started through bit 9 of the Network Control register. Once transmission has started, any write to the TX Buffer Queue 2 Base Address register is illegal and therefore ignored. Note that due to clock boundary synchronization, it takes a maximum of four clock cycles from the writing of the transmit start bit before the transmitter is active. Writing to the TX Buffer Queue 2 Base Address register during this time may produce unpredictable results. Reading this register returns the location of the descriptor currently being accessed. Since the DMA handles two frames at once, this may not necessarily be pointing to the current frame being transmitted. In terms of AMBA AXI operation, the transmit descriptors are written to memory using a single 32-bit AXI access. Since the datapath is 64-bit wide, the transmit descriptors shall be aligned at 64-bit boundaries and each pair of 32-bit descriptors is read from memory using a single 64-bit AXI access.

REGISTER_NAME	ETHERNETn_transmit_q2_ptr
OFFSET	0x444
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	dma_tx_q_ptr[31:24]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	dma_tx_q_ptr[23:16]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	dma_tx_q_ptr[15:8]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	dma_tx_q_ptr[7:2]						Reserved	
ACCESS_TYPE	R/W						R0,WX	
PROT_TYPE	Wp							
INITIAL_VALUE	0x00						0x0	

[bit31:2] dma_tx_q_ptr: Transmit buffer queue base address

Start address of transmit buffer queue.

[bit1:0] Reserved

Always read "0". Writing has no effect.

4.111. TX Buffer Queue 3 Base Address Register (ETHERNETn_transmit_q3_ptr)

This register holds the start address of the transmit buffer queue 3 (transmit buffers descriptor list). The transmit buffer queue base address register must be initialized before transmit is started through bit 9 of the Network Control register. Once transmission has started, any write to the TX Buffer Queue 3 Base Address register is illegal and therefore ignored. Note that due to clock boundary synchronization, it takes a maximum of four clock cycles from the writing of the transmit start bit before the transmitter is active. Writing to the TX Buffer Queue 3 Base Address register during this time may produce unpredictable results. Reading this register returns the location of the descriptor currently being accessed. Since the DMA handles two frames at once, this may not necessarily be pointing to the current frame being transmitted. In terms of AMBA AXI operation, the transmit descriptors are written to memory using a single 32-bit AXI access. Since the datapath is 64-bit wide, the transmit descriptors shall be aligned at 64-bit boundaries and each pair of 32-bit descriptors is read from memory using a single 64-bit AXI access.

REGISTER_NAME	ETHERNETn_transmit_q3_ptr
OFFSET	0x448
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	dma_tx_q_ptr[31:24]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	dma_tx_q_ptr[23:16]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	dma_tx_q_ptr[15:8]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	dma_tx_q_ptr[7:2]						Reserved	
ACCESS_TYPE	R/W						R0,WX	
PROT_TYPE	Wp							
INITIAL_VALUE	0x00						0x0	

[bit31:2] dma_tx_q_ptr: Transmit buffer queue base address

Start address of transmit buffer queue.

[bit1:0] Reserved

Always read "0". Writing has no effect.

4.112. RX Buffer Queue 1 Base Address Register (ETHERNETn_receive_q1_ptr)

This register holds the start address of the receive buffer queue 1 (receive buffers descriptor list). The RX Buffer Queue 1 Base Address register must be initialized before receive is enabled through bit 2 of the Network Control register. Once reception is enabled, any write to the RX Buffer Queue 1 Base Address register is ignored. Reading this register returns the location of the descriptor currently being accessed. This value increments as buffers are used. Software should not use this register for determining where to remove received frames from the queue as it constantly changes as new frames are received. Software should instead work its way through the buffer descriptor queue checking the used bits. In terms of AMBA AXI operation, the receive descriptors are read from memory using a single 32-bit AXI access. Since the datapath is 64-bit wide, the receive descriptors shall be aligned at 64-bit boundaries and each pair of 32-bit descriptors is written using a single 64-bit AXI access.

REGISTER_NAME	ETHERNETn_receive_q1_ptr
OFFSET	0x480
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	dma_rx_q_ptr[31:24]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	dma_rx_q_ptr[23:16]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	dma_rx_q_ptr[15:8]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	dma_rx_q_ptr[7:2]						Reserved	
ACCESS_TYPE	R/W						R0,WX	
PROT_TYPE	Wp							
INITIAL_VALUE	0x00						0x0	

[bit31:2] dma_rx_q_ptr: Receive buffer queue base address

Start address of receive buffer queue.

[bit1:0] Reserved

Always read "0". Writing has no effect.

4.113. RX Buffer Queue 2 Base Address Register (ETHERNETn_receive_q2_ptr)

This register holds the start address of the receive buffer queue 2 (receive buffers descriptor list). The RX Buffer Queue 2 Base Address register must be initialized before receive is enabled through bit 2 of the Network Control register. Once reception is enabled, any write to the RX Buffer Queue 2 Base Address register is ignored. Reading this register returns the location of the descriptor currently being accessed. This value increments as buffers are used. Software should not use this register for determining where to remove received frames from the queue as it constantly changes as new frames are received. Software should instead work its way through the buffer descriptor queue checking the used bits. In terms of AMBA AXI operation, the receive descriptors are read from memory using a single 32-bit AXI access. Since the datapath is 64-bit wide, the receive descriptors shall be aligned at 64-bit boundaries and each pair of 32-bit descriptors is written using a single 64-bit AXI access.

REGISTER_NAME	ETHERNETn_receive_q2_ptr
OFFSET	0x484
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	dma_rx_q_ptr[31:24]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	dma_rx_q_ptr[23:16]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	dma_rx_q_ptr[15:8]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	dma_rx_q_ptr[7:2]						Reserved	
ACCESS_TYPE	R/W						R0,WX	
PROT_TYPE	Wp							
INITIAL_VALUE	0x00						0x0	

[bit31:2] dma_rx_q_ptr: Receive buffer queue base address

Start address of receive buffer queue.

[bit1:0] Reserved

Always read "0". Writing has no effect.

4.114. RX Buffer Queue 3 Base Address Register (ETHERNETn_receive_q3_ptr)

This register holds the start address of the receive buffer queue 3 (receive buffers descriptor list). The RX Buffer Queue 3 Base Address register must be initialized before receive is enabled through bit 2 of the Network Control register. Once reception is enabled, any write to the RX Buffer Queue 3 Base Address register is ignored. Reading this register returns the location of the descriptor currently being accessed. This value increments as buffers are used. Software should not use this register for determining where to remove received frames from the queue as it constantly changes as new frames are received. Software should instead work its way through the buffer descriptor queue checking the used bits. In terms of AMBA AXI operation, the receive descriptors are read from memory using a single 32-bit AXI access. Since the datapath is 64-bit wide, the receive descriptors shall be aligned at 64-bit boundaries and each pair of 32-bit descriptors is written using a single 64-bit AXI access.

REGISTER_NAME	ETHERNETn_receive_q3_ptr
OFFSET	0x488
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	dma_rx_q_ptr[31:24]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	dma_rx_q_ptr[23:16]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	dma_rx_q_ptr[15:8]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	dma_rx_q_ptr[7:2]						Reserved	
ACCESS_TYPE	R/W						R0,WX	
PROT_TYPE	Wp							
INITIAL_VALUE	0x00						0x0	

[bit31:2] dma_rx_q_ptr: Receive buffer queue base address

Start address of receive buffer queue.

[bit1:0] Reserved

Always read "0". Writing has no effect.

4.115. RX Buffer Size Queue i Register (ETHERNETn_rxbuf_size_qi) (i=1 to 3)

Description of RX Buffer Size Queue i register is shown.

REGISTER_NAME	ETHERNETn_rxbuf_size_qi
OFFSET	(0x4A0 + ((i-1)*0x4))
ACCESS_SIZE	W
MULTIPLE	1:3
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	dma_rx_q_buf_size[7:0]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0xF0							

[bit31:8] Reserved

Always read "0". Writing has no effect.

[bit7:0] dma_rx_q_buf_size: DMA receive buffer size in system memory

The value defined by these bits determines the size of the buffer to use in system memory when writing received data. The value is defined in multiples of 64 bytes.

Bits	Description
0x00	Setting not allowed
0x01	64 Byte
0x02	128 Byte (2 * 64 Byte)
. . .	. . .
0x18	1536 Byte (24 * 64 Byte) one maximum length frame/buffer
Bits	**Description**
. . .	. . .
0xA0	10240 Byte (160 * 64 Byte) one jumbo frame/buffer
. . .	. . .
0xF0	15360 Byte (240 * 64 Byte)

Bits	Description
. . .	. . .
0xFF	16320 Byte (255 * 64 Byte)

4.116. CBS Control Register (ETHERNETn_cbs_control)

Description of CBS Control register is shown. The idleSlope value is defined as the rate of change of credit when a packet is waiting to be sent. This must noch exceed the portTransmitRate which is dependent on the speed of operation, e.g. 100 Mb/s = 017D7840$_h$, 10 Mb/s = 002625A0$_h$. If 50 % of bandwidth was to be allocated to a particular queue in 1 Mb/s mode then the idleSlope value for that queue would be calculated as 017D7840$_h$/2. Note: Credit Based Shaping shall be disabled prior to updating the idleSlope values.

REGISTER_NAME	ETHERNETn_cbs_control
OFFSET	0x4BC
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved						cbs_enable_queue_b	cbs_enable_queue_a
ACCESS_TYPE	R0,WX						R/W	R/W
PROT_TYPE	Wp							
INITIAL_VALUE	0x00						0	0

[bit31:2] Reserved

Always read "0". Writing has no effect.

[bit1] cbs_enable_queue_b: Enable Credit Based Shaping on 2nd Highest Priority Queue

.

Bit	Description
0	CBS on Queue B disabled.
1	CBS on Queue B enabled.

[bit0] cbs_enable_queue_a: Enable Credit Based Shaping on Highest Priority Queue

Bit	Description
0	CBS on Queue A disabled.
1	CBS on Queue A enabled.

4.117. CBS IdleSlope Queue A Register (ETHERNETn_cbs_idleslope_q_a)

Description of CBS IdleSlope Queue A register is shown.

REGISTER_NAME	ETHERNETn_cbs_idleslope_q_a
OFFSET	0x4C0
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	idleslope_a[31:24]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	idleslope_a[23:16]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	idleslope_a[15:8]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	idleslope_a[7:0]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:0] idleslope_a: IdleSlope Value for Queue A

Contains the idleSlope value for queue A in bytes per second. Queue A is the queue with the highest priority, i.e. queue number 3.

4.118. CBS IdleSlope Queue B Register (ETHERNETn_cbs_idleslope_q_b)

Description of CBS IdleSlope Queue B register is shown.

REGISTER_NAME	ETHERNETn_cbs_idleslope_q_b
OFFSET	0x4C4
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	idleslope_b[31:24]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	idleslope_b[23:16]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	idleslope_b[15:8]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	idleslope_b[7:0]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:0] idleslope_b: IdleSlope Value for Queue B

Contains the idleSlope value for queue B in bytes per second. Queue B is the queue with the second highest priority, i.e. queue number 2.

4.119. MSB Buffer Queue Base Address Register (ETHERNETn_msb_buff_q_base_addr_reg)

Description of MSB Buffer Queue Base Address register is shown.

REGISTER_NAME	ETHERNETn_msb_buff_q_base_addr_reg
OFFSET	0x4C8
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	msb_buff_q_base_addr[31:24]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	msb_buff_q_base_addr[23:16]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	msb_buff_q_base_addr[15:8]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	msb_buff_q_base_addr[7:0]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:0] msb_buff_q_base_addr: Upper 32-bit Buffer Queue Base Address

4.120. TX BD Control Register (ETHERNETn_tx_bd_control)

Description of TX BD Control register is shown.

REGISTER_NAME	ETHERNETn_tx_bd_control
OFFSET	0x4CC
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved		tx_bd_ts_mode[1:0]		Reserved			
ACCESS_TYPE	R0,WX		R/W		R0,WX			
PROT_TYPE	Wp							
INITIAL_VALUE	0x0		0x0		0x0			

[bit31:6] Reserved

Always read "0". Writing has no effect.

[bit5:4] tx_bd_ts_mode: TX Descriptor Timestamp Insertion mode

Bits	Description
00	TS insertion disable
01	TS inserted for PTP Event Frames only
10	TS inserted for All PTP Frames only
11	TS insertion for All Frames

[bit3:0] Reserved

Always read "0". Writing has no effect.

4.121. RX BD Control Register (ETHERNETn_rx_bd_control)

Description of RX BD Control register is shown.

REGISTER_NAME	ETHERNETn_rx_bd_control
OFFSET	0x4D0
ACCESS_SIZE	W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved		rx_bd_ts_mode[1:0]		Reserved			
ACCESS_TYPE	R0,WX		R/W		R0,WX			
PROT_TYPE	Wp							
INITIAL_VALUE	0x0		0x0		0x0			

[bit31:6] Reserved

Always read "0". Writing has no effect.

[bit5:4] rx_bd_ts_mode: RX Descriptor Timestamp Insertion mode

Bits	Description
00	TS insertion disabled
01	TS inserted for PTP Event Frames only
10	TS inserted for All PTP Frames only
11	TS insertion for All Frames

[bit3:0] Reserved

Always read "0". Writing has no effect.

4.122. Screening Type 1 Register i (ETHERNETn_screening_type_1_register_i) (i=0 to 15)

Description of Screening Type 1 register is shown.

Screening Type 1 registers are used to allocate up to 16 priority queues to received frames based on certain IP or UDP fields of incoming frames. Firstly, when DS/TC match enable is set (bit 28), the DS (Differentiated Services) field of the received IPv4 header or TC field (Traffic Class) of IPv6 headers are matched against bits [11:4]. Secondly, when UDP port match enable is set (bit 29), the UDP Destination Port of the received UDP frame is matched against bits [27:12]. Both UDP and DS/TC matching can be enabled simultaneously or individually. If a match is successful, then the queue value programmed in bits [3:0] is allocated to the frame

REGISTER_NAME	ETHERNETn_screening_type_1_register_i
OFFSET	(0x500 + (i*0x4))
ACCESS_SIZE	W
MULTIPLE	0:15
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved		udp_port_match_enable	dstc_enable	udp_port_match[15:12]			
ACCESS_TYPE	R0,WX		R/W	R/W	R/W			
PROT_TYPE	Wp							
INITIAL_VALUE	0x0		0	0	0x0			

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	udp_port_match[11:4]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	udp_port_match[3:0]				dstc_match[7:4]			
ACCESS_TYPE	R/W				R/W			
PROT_TYPE	Wp							
INITIAL_VALUE	0x0				0x0			

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	dstc_match[3:0]				queue_number[3:0]			
ACCESS_TYPE	R/W				R/W			
PROT_TYPE	Wp							
INITIAL_VALUE	0x0				0x0			

[bit31:30] Reserved

Always read "0". Writing has no effect.

[bit29] udp_port_match_enable: UDP port match enable

Bit	Description
0	Disable
1	Enable

[bit28] dstc_enable: DS/TC enable

Bit	Description
0	Disable
1	Enable

[bit27:12] udp_port_match: UDP port match
UDP Destination Port of the received UDP frame is matched against this value.

[bit11:4] dstc_match: DS/TC match
DS (Differentiated Services) field of the received IPv4 header or TC field (traffic class) of IPv6 headers is matched against this value.

[bit3:0] queue_number: Queue number
Queue Number (0 - 15).

4.123. Screening Type 2 Register i (ETHERNETn_screening_type_2_register_i) (i=0 to 15)

Description of Screening Type 2 register is shown.

Screener Type 2 match registers allow a screen to be configured that is the combination of all or any of the following comparisons:

1. An enabled field VLAN Priority. A VLAN Priority match will be performed if the VLAN priority enable is set. The extracted priority field in the VLAN header is compared against 3 bits within the screener type 2 register itself.
2. An enabled field EtherType.
3. An enabled field Compare A.
4. An enabled field Compare B.
5. An enabled field Compare C.

REGISTER_NAME	ETHERNETn_screening_type_2_register_i
OFFSET	0x540 + (i*0x4)
ACCESS_SIZE	W
MULTIPLE	0:15
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved	compare_c_enable	compare_c[4:0]					compare_b_enable
ACCESS_TYPE	R0,WX	R/W	R/W					R/W
PROT_TYPE	Wp							
INITIAL_VALUE	0	0	0x00					0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	compare_b[4:0]					compare_a_enable	compare_a[4:3]	
ACCESS_TYPE	R/W					R/W	R/W	
PROT_TYPE	Wp							
INITIAL_VALUE	0x00					0	0x0	

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	compare_a[2:0]			ethertype_enable	index[2:0]			vlan_enable
ACCESS_TYPE	R/W			R/W	R/W			R/W
PROT_TYPE	Wp							
INITIAL_VALUE	0x0			0	0x0			0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved	vlan_priority[2:0]			queue_number[3:0]			
ACCESS_TYPE	R/W	R/W			R/W			
PROT_TYPE	Wp							
INITIAL_VALUE	0	0x0			0x0			

[bit31] Reserved

Always read "0". Writing has no effect.

[bit30] compare_c_enable: Compare C enable

Bit	Description
0	Disable
1	Enable

[bit29:25] compare_c: Compare C

Index to screener type 2 Compare register.

[bit24] compare_b_enable: Compare B enable

Bit	Description
0	Disable
1	Enable

[bit23:19] compare_b: Compare B
Index to screener type 2 Compare register.

[bit18] compare_a_enable: Compare A enable

Bit	Description
0	Disable
1	Enable

[bit17:13] compare_a: Compare A
Index to screener type 2 Compare register.

[bit12] ethertype_enable: EtherType enable

Bit	Description
0	Disable
1	Enable

[bit11:9] index: EtherType
Index to screener type 2 EtherType register.

[bit8] vlan_enable: VLAN enable

Bit	Description
0	Disable
1	Enable

[bit7] Reserved
Always read write value. Writing always “0”.

[bit6:4] vlan_priority: VLAN Priority
VLAN priority.

[bit3:0] queue_number: Queue number
Queue Number (0 - 15).

4.124. Interrupt Enable Queue i Register (ETHERNETn_int_enable_qi) (i=1 to 3)

At reset all interrupts are disabled. Writing a "1" to the relevant bit location enables that particular interrupt. This register is write only and when read will return "0".

REGISTER_NAME	ETHERNETn_int_enable_qi
OFFSET	(0x600 + ((i-1)*0x4))
ACCESS_SIZE	W
MULTIPLE	1:3
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved				enable_resp_not_ok_interrupt	enable_receive_overrun_interrupt	Reserved	
ACCESS_TYPE	R0,WX				R0,W1	R0,W1	R0,WX	
PROT_TYPE	Wp							
INITIAL_VALUE	0x0				0	0	0x0	

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	enable_transmit_complete_interrupt	enable_transmit_frame_corruption_due_to_amba_error_interrupt	enable_retry_limit_exceeded_or_late_collision_interrupt	Reserved		enable_rx_used_bit_read_interrupt	enable_receive_complete_interrupt	Reserved
ACCESS_TYPE	R0,W1	R0,W1	R0,W1	R0,WX		R0,W1	R0,W1	R0,WX
PROT_TYPE	Wp							
INITIAL_VALUE	0	0	0	0		0	0	0

[bit31:12] Reserved
Always read "0". Writing has no effect.

[bit11] enable_resp_not_ok_interrupt: Enable BRESP not OK interrupt
Writing "1" enables BRESP not OK interrupt.

[bit10] enable_receive_overrun_interrupt: Enable Receive over run interrupt
Writing "1" enables Receive over run interrupt.

[bit9:8] Reserved
Always read "0". Writing has no effect.

[bit7] enable_transmit_complete_interrupt: Enable Transmit complete interrupt
Writing "1" enables Transmit complete interrupt.

[bit6] enable_transmit_frame_corruption_due_to_amba_error_interrupt: Enable Transmit frame corruption due to AMBA AXI error interrupt

Writing "1" enables Transmit frame corruption due to AMBA AXI error interrupt.

[bit5] enable_retry_limit_exceeded_or_late_collision_interrupt: Enable Retry limit exceeded or late collision interrupt
Writing "1" enables Retry limit exceeded or late collision interrupt.

[bit4:3] Reserved
Always read "0". Writing has no effect.

[bit2] enable_rx_used_bit_read_interrupt: Enable RX used bit read interrupt
Writing "1" enables RX used bit read interrupt.

[bit1] enable_receive_complete_interrupt: Enable Receive complete interrupt
Writing "1" enables Receive complete interrupt.

[bit0] Reserved
Always read "0". Writing has no effect.

4.125. Interrupt Disable Queue i Register (ETHERNETn_int_disable_qi) (i=1 to 3)

Writing a "1" to the relevant bit location disables that particular interrupt. This register is write only and when read will return "0".

REGISTER_NAME	ETHERNETn_int_disable_qi
OFFSET	(0x620 + ((i-1)*0x4))
ACCESS_SIZE	W
MULTIPLE	1:3
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved				disable_resp_not_ok_interrupt	disable_receive_overrun_interrupt	Reserved	
ACCESS_TYPE	R0,WX				R0,W1	R0,W1	R0,WX	
PROT_TYPE	Wp							
INITIAL_VALUE	0x0				0	0	0x0	

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	disable_transmit_complete_interrupt	disable_transmit_frame_corruption_due_to_amba_error_interrupt	disable_retry_limit_exceeded_or_late_collision_interrupt	Reserved		disable_rx_used_bit_read_interrupt	disable_receive_complete_interrupt	Reserved
ACCESS_TYPE	R0,W1	R0,W1	R0,W1	R0,WX		R0,W1	R0,W1	R0,WX
PROT_TYPE	Wp							
INITIAL_VALUE	0	0	0	0		0	0	0

[bit31:12] Reserved
Always read "0". Writing has no effect.

[bit11] disable_resp_not_ok_interrupt: Disable BRESP not OK interrupt
Writing "1" disables BRESP not OK interrupt.

[bit10] disable_receive_overrun_interrupt: Disable Receive over run interrupt
Writing "1" disables Receive over run interrupt.

[bit9:8] Reserved
Always read "0". Writing has no effect.

[bit7] disable_transmit_complete_interrupt: Disable Transmit complete interrupt
Writing "1" disables Transmit complete interrupt.

[bit6] disable_transmit_frame_corruption_due_to_amba_error_interrupt: Disable Transmit frame corruption due to AMBA AXI error interrupt
Writing "1" disables Transmit frame corruption due to AMBA AXI error interrupt.

[bit5] disable_retry_limit_exceeded_or_late_collision_interrupt: Disable Retry limit exceeded or late collision interrupt
Writing "1" disables Retry limit exceeded or late collision interrupt.

[bit4:3] Reserved
Always read "0". Writing has no effect.

[bit2] disable_rx_used_bit_read_interrupt: Disable RX used bit read interrupt
Writing "1" disables RX used bit read interrupt.

[bit1] disable_receive_complete_interrupt: Disable Receive complete interrupt
Writing "1" disables Receive complete interrupt.

[bit0] Reserved
Always read "0". Writing has no effect.

4.126. Interrupt Mask Queue i Register (ETHERNETn_int_mask_qi) (i=1 to 3)

The Interrupt Mask Queue i register is a read only register indicating which interrupts are masked. All bits are set at reset and can be reset individually by writing to the Interrupt Enable Queue i register or set individually by writing to the Interrupt Disable Queue i register. Having separate address locations for enable and disable saves the need for performing a read modify write when updating the Interrupt Mask Queue i register. For test purposes there is a write-only function to this register that allows the bits in the Interrupt Status Queue i register to be set or cleared, regardless of the state of the mask register

REGISTER_NAME	ETHERNETn_int_mask_qi
OFFSET	(0x640 + ((i-1)*0x4))
ACCESS_SIZE	W
MULTIPLE	1:3
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved				resp_not_ok_interrupt_mask	receive_overrun_interrupt_mask	Reserved	
ACCESS_TYPE	R0,WX				R,W	R,W	R0,WX	
PROT_TYPE	Wp							
INITIAL_VALUE	0x0				1	1	0x0	

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	transmit_complete_interrupt_mask	transmit_frame_corruption_due_to_amba_error_interrupt_mask	retry_limit_exceeded_or_late_collision_interrupt_mask	Reserved		rx_used_bit_read_interrupt_mask	receive_complete_interrupt_mask	Reserved
ACCESS_TYPE	R,W	R,W	R,W	R0,WX		R,W	R,W	R0,WX
PROT_TYPE	Wp							
INITIAL_VALUE	1	1	1	0		1	1	0

[bit31:12] Reserved

Always read "0". Writing has no effect.

[bit11] resp_not_ok_interrupt_mask: BRESP not OK interrupt mask

A write to this register directly affects the state of the corresponding bit in the Interrupt Status Queue i register, causing an interrupt to be generated if a "1" is written.

Bit	Description
0	Interrupt disabled
1	Interrupt enabled

[bit10] receive_overrun_interrupt_mask: Receive over run interrupt mask

A write to this register directly affects the state of the corresponding bit in the Interrupt Status Queue i register, causing an interrupt to be generated if a "1" is written.

Bit	Description
0	Interrupt disabled
1	Interrupt enabled

[bit9:8] Reserved

Always read "0". Writing has no effect.

[bit7] transmit_complete_interrupt_mask: Transmit complete interrupt mask

A write to this register directly affects the state of the corresponding bit in the Interrupt Status Queue i register, causing an interrupt to be generated if a "1" is written.

Bit	Description
0	Interrupt disabled
1	Interrupt enabled

[bit6] transmit_frame_corruption_due_to_amba_error_interrupt_mask: Transmit frame corruption due to AMBA AXI error interrupt mask

A write to this register directly affects the state of the corresponding bit in the Interrupt Status Queue i register, causing an interrupt to be generated if a "1" is written.

Bit	Description
0	Interrupt disabled
1	Interrupt enabled

[bit5] retry_limit_exceeded_or_late_collision_interrupt_mask: Retry limit exceeded or late collision interrupt mask

A write to this register directly affects the state of the corresponding bit in the Interrupt Status Queue i register, causing an interrupt to be generated if a "1" is written.

Bit	Description
0	Interrupt disabled
1	Interrupt enabled

[bit4:3] Reserved

Always read "0". Writing has no effect.

[bit2] rx_used_bit_read_interrupt_mask: RX used bit read interrupt mask

A write to this register directly affects the state of the corresponding bit in the Interrupt Status Queue i register, causing an interrupt to be generated if a "1" is written.

Bit	Description
0	Interrupt disabled
1	Interrupt enabled

[bit1] receive_complete_interrupt_mask: Receive complete interrupt mask

A write to this register directly affects the state of the corresponding bit in the Interrupt Status Queue i register, causing an interrupt to be generated if a "1" is written.

Bit	Description
0	Interrupt disabled
1	Interrupt enabled

[bit0] Reserved

Always read "0". Writing has no effect.

4.127. EtherType 2 Register i (ETHERNETn_screening_type_2_ethertype_reg_i) (i=0 to 7)

Description of EtherType 2 register "i" is shown.

REGISTER_NAME	ETHERNETn_screening_type_2_ethertype_reg_i
OFFSET	0x6E0 + (i*0x4)
ACCESS_SIZE	W
MULTIPLE	0:7
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	compare_value[15:8]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	compare_value[7:0]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:16] Reserved
Always read "0". Writing has no effect.

[bit15:0] compare_value: Compare value
EtherType 2 compare value.

4.128. Type2 Compare Word 0 i Register (ETHERNETn_type2_compare_i_word_0) (i=0 to 31)

Description of Type2 Compare Word 0 "i" is shown.

REGISTER_NAME	ETHERNETn_type2_compare_i_word_0
OFFSET	0x700 + (i*0x8)
ACCESS_SIZE	W
MULTIPLE	0:31
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	compare_value[15:8]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	compare_value[7:0]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	mask_value[15:8]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	mask_value[7:0]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

[bit31:16] compare_value: Compare Value

2 byte Compare Value. The byte stored in bits [23:16] is compared against the 1st byte of the 2 bytes extracted from the frame. The byte stored in bits [31:24] is compared against the 2nd byte of the 2 bytes extracted from the frame.

[bit15:0] mask_value: Mask value

2 byte Mask Value.

4.129. Type2 Compare Word 1 i Register (ETHERNETn_type2_compare_i_word_1) (i=0 to 31)

Description of Type2 Compare Word 1 "i" is shown.

REGISTER_NAME	ETHERNETn_type2_compare_i_word_1
OFFSET	0x704 + (i*0x8)
ACCESS_SIZE	W
MULTIPLE	0:31
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved							compare_offset[1]
ACCESS_TYPE	R0,WX							R/W
PROT_TYPE	Wp							
INITIAL_VALUE	0x00							0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	compare_offset[0]	offset_value[6:0]						
ACCESS_TYPE	R/W	R/W						
PROT_TYPE	Wp							
INITIAL_VALUE	0	0x00						

[bit31:9] Reserved

Always read "0". Writing has no effect.

[bit8:7] compare_offset[1:0]: Compare offset

Compare byte offset.

Bits	Description
00	Offset from beginning of frame.
01	Offset from byte after EtherType.
10	Offset from byte following end of IP header.
11	Offset from byte following end of TCP/UDP header.

[bit6:0] offset_value[6:0]: Offset value

Offset value in bytes.

5. Functional Limitations

5.1. The interrupts on the pause frame reception

Interrupt bit13 (pause_time_elapsed) will never get set for PFC pause, though interrupt bit12 (pause_frame_with_non_zero_pause_quantum_received) will get set for the reception of pause frames including the PFC pause frames.
Therefore those interrupts are really useful for only the pause time monitoring by the standard legacy 802.3 pause frame reception.

5.2. Priority Flow Control - pausing the transmitting the frames

Transmission of frames will not be paused by the PFC pause frame reception.

5.3. Priority Queuing - screening using DS/TC field in IP header

When the frames with 0x00 at the IPv4 TOS field or at the IPv6 TC field are forwarded to the specific queue, the queue has to be the default queue (i.e. queue 0). Therefore there is no need of screeners to do such a forwarding.

5.4. Priority Queuing - screening using compare_offset = "10" or "11"

When only the byte matching is designated, screener type2 potentially produces a match for non-IP/TCP frames, since it doesn't check the type of frames. To specify the type of frames to match, the ethertype matching is needed to be enabled.

5.5. Enable/Disable the partial store and forward mode

Partial store and forward mode is a static programming option, and it should not be changed unless the ethernet link is down.

5.6. PTP - single step time stamping

For the single step time stamping, the data in the SYNC frames will be corrupted, if the format of the SYNC frame is of IEEE Std. 1588 version1 UDP/IPv4 multicast with VLAN tag.

The single step time stamping using the IEEE Std. 1588 version 1 frames is very rare use case. And this limitation can be avoided by either of following workarounds:

- Not to use the VLAN tags with the IEEE Std 1588 version 1 frames for the single step time stamping.
- Doing the two step time stamping if VLAN tagged IEEE Std version 1 frames will be used.

5.7. PTP/gPTP - The ptp_sync_frame_transmitted interrupt

The interrupt bit21 (ptp_sync_frame_transmitted) won't be generated for the transmission of the multicast SYNC frames satisfying all of the following conditions:

- Compliant for the IEEE Std. 1588 version 2.
- VLAN tagged.
- The destination address is 01:80:C2:00:00:0E.

This limitation won't be a issue for any real use case for the following reasons:

- In the IEEE Std. 1588 compliant system, the destination address 01:80:C2:00:00:0E is used by only the Pdelay Request frames and Pdelay Response frames. And the SYNC frames use the destination address 01:1B:19:00:00:00.
- In the IEEE Std. 802.1AS compliant system, The frames may not be VLAN tagged.

5.8. PTP/gPTP - TSU timer comparison with 0

When all of 3 TSU timer comparison value registers (located at the offset of 0x0DC, 0x0E0, and 0x0E4) are set to 0x0, the TSU timer comparison won't be executed. And, as a result, the interrupt request won't be generated.
The condition will be avoidable in the normal use cases.

CHAPTER 22: MEDIA Local Bus Interface (MediaLB)

This chapter explains the functions and operations of the MediaLB.

1. Overview
2. Configuration and Block Diagram
3. Operation of the MediaLB
4. Registers

CODE: MEDIA_LB-S6J3300-E1

1. Overview

The MediaLB comprises a MediaLB functional block, a AHB Master and Slave interface, and a 9 Kbit RAM for the local channel buffer. The MediaLB is an interface that implements the required functionality of a Media Local Bus Device for connecting the device with a MOST (Media Oriented System Transport, which is a multimedia-type in-vehicle LAN standard) network as outlined in reference MediaLB Specification by SMSC (TB0400AN3V0, rev 3.0 issued on February 2006). The MediaLB transfers data between the MediaLB Bus and the system memory, using the AHB Master Interface. In order to operate, MediaLB configuration registers must be set via the AHB Slave Interface. The features of the MediaLB module are listed in this section.

Features of the MediaLB

- Implements the Physical and Link Layer requirements outlined in MediaLB Specification rev 3.0
- Transmits commands and data when functioning as the transmitting device
- Receives data and transmits status responses when functioning as the receiving device
- Detection of the device lock or unlock from the MediaLB Frame.
- Handling of System Channel commands
- Supports 16 logical channels
- Each logical channel can be programmed as synchronous, asynchronous, isochronous and control channel type and as transmit or receive
- Loop back mode between the logical channel 0 (reception) and logical channel 1(transmission)
- Can operate in DMA mode (ping-pong buffering and circular buffering) and in IO mode
- Built-in dual-port RAM as local channel buffer for compensating the bus latency
- Supports 8-bit, 16-bit and 32-bit bus accesses to registers in MediaLB
- Configurable protection for read and write accesses to registers in MediaLB
- Programmable for 256Fs, 512Fs and 1024Fs transfer rates of operation at either 44.0 kHz, 48.0 kHz, or 48.1 kHz.
- 3-pin MediaLB Mode

Abbreviations

This section lists the terms and abbreviations used in this chapter.

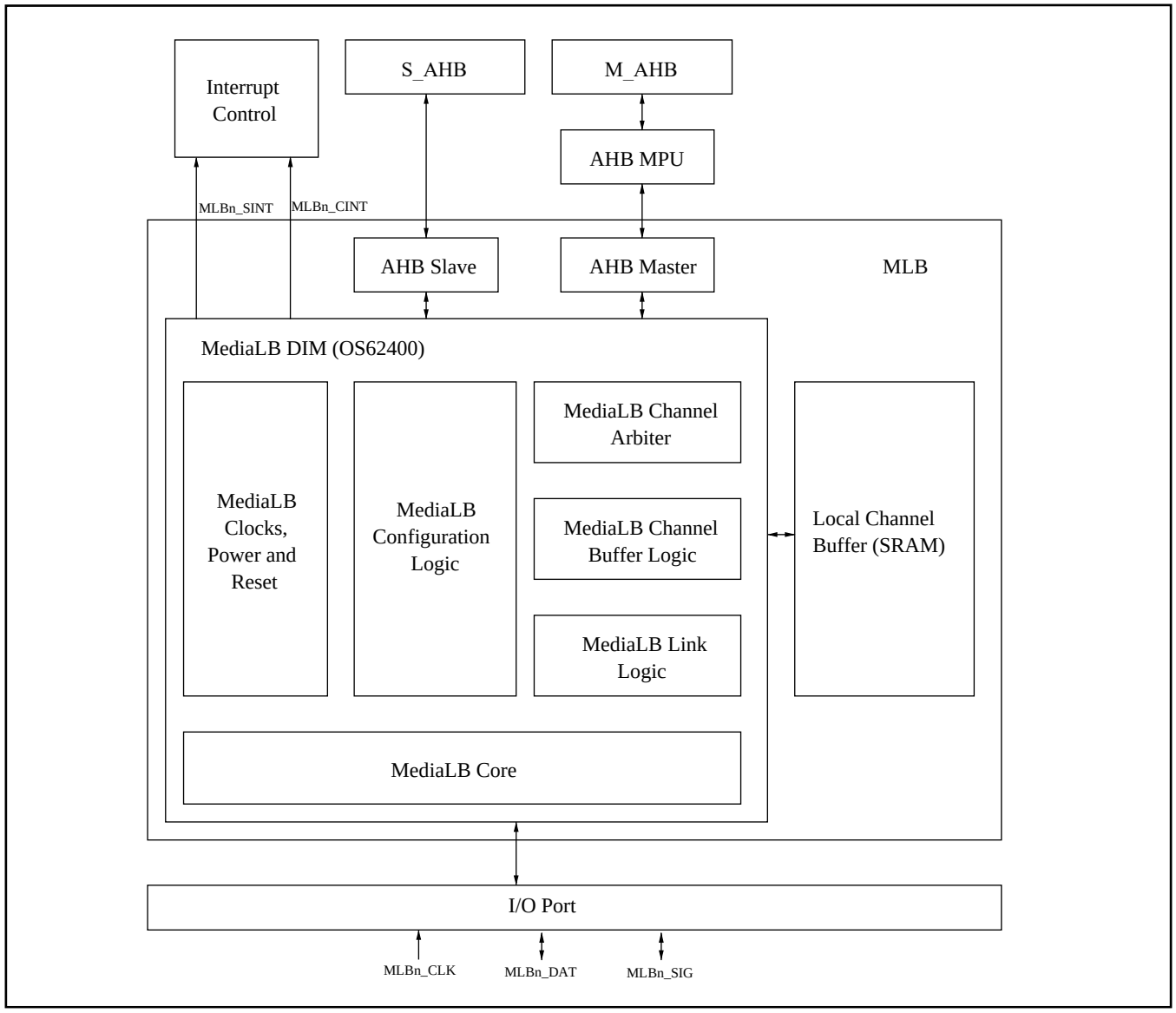
Table 22-1 Terms and Abbreviations

Term	Meaning
MediaLB	The function block that implements MediaLB (Media Local Bus), which is a local bus specification, for connecting IC devices with a MOST (Media Oriented System Transport), which is a multimedia-type in-vehicle LAN standard network.
Quadlet	Indicates a 4-byte unit.
Sync data	Data of stream signals, such as audio data.
Async data	Packet data.
Control data	Data of control signals.
Local channel buffer	The data storage area in the MediaLB
IO mode	The mode capable of exchanging transmission/reception data with the MediaLB from CPU.
DMA mode	The mode in which the MediaLB operates as the bus master of the AHB bus. Transmission/reception data is exchanged between the MediaLB and system memory via the AHB bus.
Current buffer	The data area on the AHB bus that is read/written in DMA mode.
Current buffer address	Indicates the address area that is output next on the AHB bus in DMA mode after processing the current buffer area. Set in the channel n next buffer configuration register (MLBn_CNBCRn).
Previous buffer	The data area processed before the current buffer in DMA mode.
Previous buffer address	The address area processed before the current buffer in DMA mode.
Ping-pong buffering	One of two data transfer methods in DMA mode. If the current buffer is filled, an interrupt occurs when moving to the next buffer and so the next address to be used next can be set again by software. Therefore, it is possible to transmit/receive data while changing the transfer destination data storage area one after another.
Circular buffering	One of two data transfer methods in DMA mode. Performs data transfers repeatedly for the data storage area indicated by the current buffer. No interrupt occurs at the completion of a transfer.
n	Indicates a logical channel number (n=0..15)

2. Configuration and Block Diagram

This section shows a block diagram of MediaLB.

Figure 22-1 MediaLB Block Diagram



MediaLB DIM

The function block of MediaLB is provided by SMSC IP OS62400 MediaLB Device Interface Macro. It implements the Physical and Link Layer requirements outlined in the MediaLB Specification by SMSC (TB0400AN3V0, rev 3.0 issued on February 2006).

Local Channel Buffer (SRAM)

A 9 kbit SRAM is used for buffering of logical channel data for compensating the bus latency. It stores up to 256 quadlets plus tag information (256 words x 36 bits).

1. AHB Master Interface

   The AHB Master Interface allows the MediaLB to access the system memory. It converts the MediaLB Host Bus Interface (HBI) protocol to AHB protocol.

2. AHB Slave Interface

   The AHB Master Interface allows the bus masters to access the MediaLB configuration and status registers. It converts the AHB protocol to MediaLB Peripheral Bus Interface (PBI).

Note:

- *In devices that use MediaLB, a minimum AHB Interface clock value is specified. Please refer to the Internal Clock Timing table in the device-specific datasheet.*
- *In addition, the system must be able to handle 6 MByte/sec on the bus. This means that, depending on the bus traffic as well as access to system RAM by other masters, higher AHB interface clock frequencies are recommended.*

MediaLB IO Mode

The IO mode dictates a particular method used for transferring data between the local channel buffer and the system memory. For channels configured to receive data from the MediaLB interface, system software is responsible for periodically unloading RX data from the local channel buffer. For channels configured to transmit data to the MediaLB Interface, system software is responsible for periodically loading the local channel buffer with TX data. The MLBn_CCBCRn and MLBn_CNBCRn registers, accessed through the AHB Slave interface, are used as a data receive buffer and a data transmit buffer, respectively.

For details on the IO Mode operation, please contact SMSC.

MediaLB DMA Mode

The DMA mode is a mode in which MediaLB accesses the system memory via the AHB Master bus. The system memory contains transmitted and received data, which can be read by software and the MediaLB.

When MediaLB is used in the DMA mode, data is transmitted and received via the system memory. When used in the DMA mode, MediaLB functions as a master of the AHB bus, reading and writing data from and to the system memory. When MediaLB accesses the AHB bus, Buffer Current Address BCA[15:0] in Channel n Current Buffer Configuration Register (MLBn_CCBCRn) for MediaLB is output as the address of the AHB bus. In the DMA mode, two types of buffering is supported: Ping-Pong and circular buffering.

In the Ping-Pong buffering, if the current buffer is filled, an interrupt occurs when moving to the next buffer and so the next address to be used next can be set again by software. Therefore, it is possible to transmit/receive data while changing the transfer destination data storage area one after another.

In the circular buffering, a buffer from a start address to an end address is used in a loop indefinitely by setting the start and end addresses in Next Buffer Configuration Register (MLBn_CNBCRn) until the software sets to “0” the RDY bit in Channel n Status Configuration Register (MLBn_CSCRn). This type of buffering should be used only for synch channels.

For details on the DMA Mode operation, please contact SMSC.

Local Channel Buffer

The local channel buffer comprises RAM of 256 words x 36 bits, where each word contains 4 quadlets plus tag information. The local channel buffer is in MediaLB and used by each logical channel to store transmitted and received data temporarily. To assign a RAM area, Local Channel n Buffer Configuration Register (MLBn_LCBCRn) is used to set the start address (SA bit) and a length (BD bit) of the RAM area. Furthermore, in the IO mode, an interrupt can be triggered to each logical channel by setting a threshold value (TH bit) using Local Channel n Buffer Configuration Register (MLBn_LCBCRn).

For details on the Local channel buffer operation, please contact SMSC.

Interrupts by MediaLB

This section describes interrupts that MediaLB has. MediaLB has two types of interrupts* Channel interrupts indicating the state of each channel buffer (Cint) and System interrupts indicating the state of the MediaLB system (Sint).

A channel interrupt is triggered by the status or error information retained by Channel n Status Configuration Register (MLBn_CSCRn). Channel interrupts are maskable on a channel basis via the MLBn_CECRn register. To know in which channel an interrupt has been triggered, Channel Interrupt Configuration Register (MLBn_CICR) needs to be read. Each bit in Channel Interrupt Configuration Register (MLBn_CICR) is cleared to 0 by clearing the factor that has triggered an interrupt to each channel or masking the interrupt using the mask bit. The root cause of the interrupt can be determined by reading the current and previous status fields in the MLBn_CSCRn register. The following tables describe the interrupt flags for channel interrupts and corresponding interrupt cause factors. The interrupt flag is changed to “1” when an interrupt cause is detected. There are bits that change the meaning of the interrupt flag, depending on the operation mode (IO/DMA mode).

Note:
*Data interrupts (mlb_dint[30:0]) provided by SMSC IP OS62400 are not supported. Reason is that data interrupts are provided for customers who want to attach an external DMA Controller directly to OS62400 and load/unload the local channel buffers in IO Mode.

Table 22-2 Interrupt Flags and Causes Common Across IO and DMA Mode.

Interrupt Flag	Register	Interrupt Source	Valid Channel	Interrupt Mask	Interrupt Flag Clear
STS[0]	MLBn_CSCRn	Current Buffer Protocol Error	All channels with reception setting, and async and control channels with transmission setting.	MLBn_CECRn: MASK[0]	Writing “1” to STS[0]
STS[1]		Current Buffer Break	Async and control channels	MLBn_CECRn: MASK[1]	Writing “1” to STS[1]
STS[4]		Buffer Error	Sync (both transmission/reception) and isochronous channels with reception setting.	MLBn_CECRn: MASK[4]	Writing “1” to STS[4]
STS[6]		Frame Sync Lost	Sync channels (both transmission/reception)	MLBn_CECRn: MASK[6]	Writing “1” to STS[6]

Table 22-3 Interrupt Flags and Causes Valid Only in IO Mode.

Interrupt Flag	Register	Interrupt Source	Valid Channel	Interrupt Mask	Interrupt Flag Clear
STS[2]	MLBn_CSCRn	Receive service requests	All channels enabled for receiving.	MLBn_CECRn: MASK[2]	Writing “1” to STS[2]
STS[3]		Transmit service request	All channels enabled for transmitting.	MLBn_CECRn: MASK[3]	Writing “1” to STS[3]
STS[8]		Received packet aborted	Async and control channels enabled for receiving	MLBn_CECEn: MASK[2]	Writing “1” to STS[8]

Table 22-4 Interrupt Flags and Causes Valid in DMA Mode Only

Interrupt Flag	Register	Interrupt Source	Valid Channel	Interrupt Mask	Interrupt Flag Clear
STS[2]	MLBn_CSCRn	Current buffer ends	All channels (both transmission/reception)	MLBn_CECRn: MASK[2]	Writing “1” to STS[2]
STS[3]		Current buffer starts	All channels (both transmission/reception)	MLBn_CECRn: MASK[3]	Writing “1” to STS[3]
STS[5]		Host bus error (received AHB bus error response)	All channels (both transmission/reception).	*	Writing “1” to STS[5]
STS[8]		Previous buffer protocol error	All channels enabled for receiving or async and control channel enabled for transmitting	MLBn_CECEn: MASK[2]	Writing “1” to STS[8]
STS[9]		Previous buffer break	All channels (both transmission/reception).	MLBn_CECRn: MASK[1]	Writing “1” to STS[9]
STS[10]		Previous buffer end	All channels (both transmission/reception)	MLBn_CECRn: MASK[2]	Writing “1” to STS[10]
STS[11]		Previous buffer start	All channels (both transmission/reception)	MLBn_CECRn: MASK[3]	Writing “1” to STS[11]

*Cannot be masked. Setting the cause factor flag to “1” triggers an interrupt.

Note:

- *A system interrupt is triggered when a system command, locking or unlocking is detected, as shown inTable 22-5. An interrupt flag is changed to "1" when an interrupt cause is detected. Each interrupt flag has a corresponding interrupt mask bit. Setting an interrupt mask to "1" enables masking.*

Table 22-5 Interrupt Flags and Interrupt Cause Factors for System Interrupts

Interrupt Source	Register	Interrupt Flag	Interrupt Mask	Interrupt Flag Clear
Detecting a reset (MlbRset(FEH))	MLBn_SSCR	SDR	MLBn_SMCR:SMR	Writing "1" to SDR
Detecting a network lock (MOST_Lock(F0H))		SDNL	MLBn_SMCR:SMNL	Writing "1" to SDNL
Detecting a network unlock (MOST_Unlock(E2H))		SDNU	MLBn_SMCR:SMNU	Writing "1" to SDNU
Detecting a channel scan (MlbScan(E4H))		SDCS	MLBn_SMCR:SMCS	Writing "1" to SDCS
Detecting a sub-command (MlbSubCmd(E6H))		SDSC	MLBn_SMCR:SMSC	Writing "1" to SDSC
Detecting a MediaLB lock		SDML	MLBn_SMCR:SMML	Writing "1" to SDML
Detecting a MediaLB unlock		SDMU	MLBn_SMCR:SMMU	Writing "1" to SDMU

Loop Back Mode

To facilitate debugging of transmission paths, MediaLB supports the loop back test mode.

Setting the LBM bit in Device Control Configuration Register (MLBn_DCCR) to "1" causes data to be received via Channel 0 and transmitted back via Channel 1.To use the loop back test mode, use the following steps to set the mode*:

1. Set the logical channel addresses for Channels 0 and 1. (Same address cannot be used for different channels.)
2. Set Channel 0 to any channel type. Enable "Receipt" and set "Enable channel".
3. Set Channel 1 to the same channel type as that of Channel 0, enable "Transmit" and set "Enable channel".
4. Set the loop back mode bit (MLBn_DCCR:LBM="1").

Note:

- **During the loop back mode, a protocol error or a break is prohibited for both Channels 0 and 1. During the loop back mode, the Next Buffer Ready bits for Channels 0 and 1 remains cleared (MLBn_CSCR0:RDY=MLBn_CSCR1:RDY="0"). During the loop back mode, little-endian mode must be disabled (MLBn_DCCR=MLE="0"). During the loop back mode, ishochronous packet lengths must be quadlet multiples.*

3. Operation of the MediaLB

This section describes the operation of MediaLB.

3.1. Notes on Using MediaLB

This section is the “Programmer’s Guide”, which lists the usage notes for programming the MediaLB module. It is recommended to contact SMSC for further details.

General Usage Notes

- Reserved bits return undefined values. The software programs shall be independent of the values read from the reserved register bits.
- If the MediaLB should be accessible in user mode, PPU should be configured accordingly.

Steps in Programming the MediaLB Module

- After the system reset, the software shall detect the Module ID number of MediaLB, by reading the MLBn_MID register. This would help it in identifying the attributes and capabilities supported by the MediaLB module.
- Then Software should configure MediaLB by setting appropriate registers following recommended Software Flowcharts. For further details, please contact SMSC.

3.2. List of References

This chapter lists the various references to other documents, used in this specification.

- OS62400 MediaLB Device Interface Macro Advanced Product Data Sheet (DS62400AP5) by SMSC, issued on October 2006.
- MediaLB Specification by SMSC (TB0400AN3V0, rev 3.0 issued on February 2006)

4. Registers

The MediaLB module contains various registers to configure its operation, to monitor its status and, when configured in IO Mode, to access the receive and transmit data buffers.

The MediaLB module is allocated 1 KB of MCU address space for mapping the Configuration and Status Registers (i.e. CSRs). The address area allocated to MediaLB and the Control and Status Registers in MediaLB are explained in this section.

The suffix ‘n’ in the register name indicates that the register is an instance ‘n’ of the module.

Registers of MediaLB

The following registers are available for each instance of MediaLB:

- MediaLBn Device Control Configuration Register (MLBn_DCCR)
- MediaLBn System Status Configuration Register (MLBn_SSCR)
- MediaLBn System Data Configuration Register (MLBn_SDCR)
- MediaLBn System Mask Configuration Register (MLBn_SMCR)
- MediaLBn Version Control Configuration Register (MLBn_VCCR)
- MediaLBn Synchronous Base Address Configuration Register (MLBn_SBCR)
- MediaLBn Asynchronous Base Address Configuration Register (MLBn_ABCR)
- MediaLBn Control Base Address Configuration Register (MLBn_CBCR)
- MediaLBn Isochronous Base Address Configuration Register (MLBn_IBCR)
- MediaLBn Channel Interrupt Configuration Register (MLBn_CICR)
- MediaLBn AHB Master Control Register (MLBn_AHBMCTL)
- MediaLBn Channel Entry Configuration Register (MLBn_CECR0 - MLBn_CECR15)
- MediaLBn Channel Status Configuration Register (MLBn_CSCR0 - MLBn_CSCR15)
- MediaLBn Channel Current Buffer Configuration Register (MLBn_CCBCR0 - MLBn_CCBCR15)
- MediaLBn Channel Next Buffer Configuration Register (MLBn_CNBCR0 - MLBn_CNBCR15)
- MediaLBn Local Channel Buffer Configuration Register (MLBn_LCBCR0 - MLBn_LCBCR15)
- MediaLBn Module Identification Register (MLBn_MID)

Arrangement of MediaLB Registers in Memory

Table 22-6 MediaLB Register Map

Offset	+3	+2	+1	+0
0x00000000	MLBn_DCCR 00000000 00000000 00000000 00000000			
0x00000004	MLBn_SSCR 00000000 00000000 00000000 00000000			
0x00000008	MLBn_SDCR 00000000 00000000 00000000 00000000			
0x0000000C	MLBn_SMCR 00000000 00000000 00000000 01100000			
0x00000010 - 0x00000018	reserved XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX			
0x0000001C	MLBn_VCCR 00000000 00000000 00000011 00000000			
0x00000020	MLBn_SBCR 00000000 00000000 00000000 00000000			
0x00000024	MLBn_ABCR 00000000 00000000 00000000 00000000			
0x00000028	MLBn_CBCR 00000000 00000000 00000000 00000000			
0x0000002C	MLBn_IBCR 00000000 00000000 00000000 00000000			
0x00000030	MLBn_CICR 00000000 00000000 00000000 00000000			
0x00000034 - 0x00000038	reserved XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX			
0x0000003C	MLBn_AHBMCTL 00000000 00000000 00000000 00000000			
0x00000040	MLBn_CECR0 00000000 00000000 00000000 00000000			
0x00000044	MLBn_CSCR0 10000000 00000000 00000000 00000000			
0x00000048	MLBn_CCBCR0 00000000 00000000 00000000 00000000			
0x0000004C	MLBn_CNBCR0 00000000 00000000 00000000 00000000			
0x00000050	MLBn_CECR1 00000000 00000000 00000000 00000000			
0x00000054	MLBn_CSCR1 10000000 00000000 00000000 00000000			
0x00000058	MLBn_CCBCR1 00000000 00000000 00000000 00000000			
0x0000005C	MLBn_CNBCR1 00000000 00000000 00000000 00000000			
0x00000060	MLBn_CECR2 00000000 00000000 00000000 00000000			
0x00000064	MLBn_CSCR2 10000000 00000000 00000000 00000000			
0x00000068	MLBn_CCBCR2 00000000 00000000 00000000 00000000			
0x0000006C	MLBn_CNBCR2 00000000 00000000 00000000 00000000			
0x00000070	MLBn_CECR3 00000000 00000000 00000000 00000000			
0x00000074	MLBn_CSCR3 10000000 00000000 00000000 00000000			
0x00000078	MLBn_CCBCR3 00000000 00000000 00000000 00000000			
0x0000007C	MLBn_CNBCR3 00000000 00000000 00000000 00000000			

Offset	+3	+2	+1	+0
0x00000080	MLBn_CECR4 00000000 00000000 00000000 00000000			
0x00000084	MLBn_CSCR4 10000000 00000000 00000000 00000000			
0x00000088	MLBn_CCBCR4 00000000 00000000 00000000 00000000			
0x0000008C	MLBn_CNBCR4 00000000 00000000 00000000 00000000			
0x00000090	MLBn_CECR5 00000000 00000000 00000000 00000000			
0x00000094	MLBn_CSCR5 10000000 00000000 00000000 00000000			
0x00000098	MLBn_CCBCR5 00000000 00000000 00000000 00000000			
0x0000009C	MLBn_CNBCR5 00000000 00000000 00000000 00000000			
0x000000A0	MLBn_CECR6 00000000 00000000 00000000 00000000			
0x000000A4	MLBn_CSCR6 10000000 00000000 00000000 00000000			
0x000000A8	MLBn_CCBCR6 00000000 00000000 00000000 00000000			
0x000000AC	MLBn_CNBCR6 00000000 00000000 00000000 00000000			
0x000000B0	MLBn_CECR7 00000000 00000000 00000000 00000000			
0x000000B4	MLBn_CSCR7 10000000 00000000 00000000 00000000			
0x000000B8	MLBn_CCBCR7 00000000 00000000 00000000 00000000			
0x000000BC	MLBn_CNBCR7 00000000 00000000 00000000 00000000			
0x000000C0	MLBn_CECR8 00000000 00000000 00000000 00000000			
0x000000C4	MLBn_CSCR8 10000000 00000000 00000000 00000000			
0x000000C8	MLBn_CCBCR8 00000000 00000000 00000000 00000000			
0x000000CC	MLBn_CNBCR8 00000000 00000000 00000000 00000000			
0x000000D0	MLBn_CECR9 00000000 00000000 00000000 00000000			
0x000000D4	MLBn_CSCR9 10000000 00000000 00000000 00000000			
0x000000D8	MLBn_CCBCR9 00000000 00000000 00000000 00000000			
0x000000DC	MLBn_CNBCR9 00000000 00000000 00000000 00000000			
0x000000E0	MLBn_CECR10 00000000 00000000 00000000 00000000			
0x000000E4	MLBn_CSCR10 10000000 00000000 00000000 00000000			
0x000000E8	MLBn_CCBCR10 00000000 00000000 00000000 00000000			
0x000000EC	MLBn_CNBCR10 00000000 00000000 00000000 00000000			
0x000000F0	MLBn_CECR11 00000000 00000000 00000000 00000000			
0x000000F4	MLBn_CSCR11 10000000 00000000 00000000 00000000			
0x000000F8	MLBn_CCBCR11 00000000 00000000 00000000 00000000			
0x000000FC	MLBn_CNBCR11 00000000 00000000 00000000 00000000			

Offset	+3	+2	+1	+0
0x00000100	MLBn_CECR12 00000000 00000000 00000000 00000000			
0x00000104	MLBn_CSCR12 10000000 00000000 00000000 00000000			
0x00000108	MLBn_CCBCR12 00000000 00000000 00000000 00000000			
0x0000010C	MLBn_CNBCR12 00000000 00000000 00000000 00000000			
0x00000110	MLBn_CECR13 00000000 00000000 00000000 00000000			
0x00000114	MLBn_CSCR13 10000000 00000000 00000000 00000000			
0x00000118	MLBn_CCBCR13 00000000 00000000 00000000 00000000			
0x0000011C	MLBn_CNBCR13 00000000 00000000 00000000 00000000			
0x00000120	MLBn_CECR14 00000000 00000000 00000000 00000000			
0x00000124	MLBn_CSCR14 10000000 00000000 00000000 00000000			
0x00000128	MLBn_CCBCR14 00000000 00000000 00000000 00000000			
0x0000012C	MLBn_CNBCR14 00000000 00000000 00000000 00000000			
0x00000130	MLBn_CECR15 00000000 00000000 00000000 00000000			
0x00000134	MLBn_CSCR15 10000000 00000000 00000000 00000000			
0x00000138	MLBn_CCBCR15 00000000 00000000 00000000 00000000			
0x0000013C	MLBn_CNBCR15 00000000 00000000 00000000 00000000			
0x00000140 - 0x0000027C	reserved XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX			
0x00000280	MLBn_LCBCR0 00000000 01000000 00000000 00000000			
0x00000284	MLBn_LCBCR1 00000000 01000000 00000000 00000000			
0x00000288	MLBn_LCBCR2 00000000 01000000 00000000 00000000			
0x0000028C	MLBn_LCBCR3 00000000 01000000 00000000 00000000			
0x00000290	MLBn_LCBCR4 00000000 01000000 00000000 00000000			
0x00000294	MLBn_LCBCR5 00000000 01000000 00000000 00000000			
0x00000298	MLBn_LCBCR6 00000000 01000000 00000000 00000000			
0x0000029C	MLBn_LCBCR7 00000000 01000000 00000000 00000000			
0x000002A0	MLBn_LCBCR8 00000000 01000000 00000000 00000000			
0x000002A4	MLBn_LCBCR9 00000000 01000000 00000000 00000000			
0x000002A8	MLBn_LCBCR10 00000000 01000000 00000000 00000000			
0x000002AC	MLBn_LCBCR11 00000000 01000000 00000000 00000000			
0x000002B0	MLBn_LCBCR12 00000000 01000000 00000000 00000000			
0x000002B4	MLBn_LCBCR13 00000000 01000000 00000000 00000000			
0x000002B8	MLBn_LCBCR14 00000000 01000000 00000000 00000000			

Offset	+3	+2	+1	+0
0x000002BC	MLBn_LCBCR15 00000000 01000000 00000000 00000000			
0x000002C0 - 0x000002F8	reserved XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX			
0x000002FC	MLBn_MID 00000000 00000000 00000000 00000000			

4.1. MediaLBn Device Control Configuration Register (MLBn_DCCR)

The MediaLBn Device Control Configuration Register (MLBn_DCCR) is used to control basic features of the MediaLB, such as clock rate, pinout, lock status, enable and device addressing.

REGISTER_NAME	YOURPERIn_CSRi
OFFSET	0 + i*4
ACCESS_SIZE	B H W
MULTIPLE	0:15
NUMERIC_TYPE	
OTHER	

MediaLBn Device Control Configuration Register (MLBn_DCCR)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	MDE	LBM	MCS[1]	MCS[0]	M5PS	MLK	MLE	MHRE
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R,WX	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	MRS	read0	read0	read0	read0	read0	read0	read0
ACCESS_TYPE	R/W	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	15	14	13	12	11	10	9	8
BIT_NAME	read0	read0	read0	read0	read0	read0	read0	read0
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	MDA[7]	MDA[6]	MDA[5]	MDA[4]	MDA[3]	MDA[2]	MDA[1]	MDA[0]
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] MDE : MediaLB Device Enable

Enables the operation of MediaLB interface.

Bit	Description
0	Disables MediaLB Interface.
1	Enables MediaLB Interface based on the other bits in the register.

[bit30] LBM : Loop-Back Mode Enable

Sets whether the loopback test for the MediaLB bus between even logical channel N (reception setting) and odd logical channel N+1 (transmission setting) is enabled or disabled.

Bit	Description
0	Normal operation. Disables Loop-back test mode.
1	Enables Loop-back test mode.

Note:

- *For more information on Loop-BackTest mode please contact SMSC support.*

[bit29:28] MCS[1:0] : MediaLB Clock Select

Sets the MediaLB transfer rate.

This field must be programmed by the system software to reflect the MLBn_MLBCLK speed.

Bit[1:0]	Description
00	256Fs - supports 8 quadlets per frame
01	512Fs - supports 16 quadlets per frame
10	1024Fs - supports 32 quadlets per frame
11	Reserved

It is prohibited to set this bit field to '11'

[bit27] M5PS : MediaLB 5-Pin Select

Selects MediaLB 5-pin or 3-pin configuration.

Bit	Description
0	3-pin MediaLB mode
1	5-pin MediaLB mode

Note:

- *This device does not support 5-pin MediaLB interface and therefore writing '1' to this bit is prohibited. Make sure the bit is set to '0'.*

[bit26] MLK : MediaLB Lock

Indicates whether MediaLB is in sync with the MediaLB frame (lock).

Bit	Description
0	Unlocked state
1	Locked state

When set, it indicates that the MediaLB Port is synchronized to the incoming MediaLB frame.

If MLK is clear (unlocked), MLK is set after FRAMESYNC is detected at the same position for three consecutive frames. If MLK is set (locked), MLK is cleared after not receiving FRAMESYNC at the expected time for two consecutive frames. While MLK is set, FRAMESYNC patterns occurring at locations other than the expected one are ignored.

Note:

- *When MLBn_DCCR:MRS is set to '1', this bit is cleared to '0'. When MLBn_DCCR:MDE is '0', the lock is not detected.*

[bit25] MLE : MediaLB Little Endian mode

This field determines how MediaLB data (quadlet-based) is stored in system memory.

Bit	Description
0	Big Endian mode
1	Little Endian mode

[bit24] MHRE : MediaLB Hardware Reset Enable

This bit enables the hardware to automatically reset the MediaLB physical and link layer logic upon reception of the reset command.

Resetting is preformed by the reset request from the Intelligent Network Interface Controller (INIC).

Resetting is performed when either the global (MLBn_SDCR:MSD = 0x00: all of MediaLB are targeted for reset) MlbReset (0xFE) or device-specific (MLBn_SDCR:MSD = DA: MediaLB designated by Device Address is targeted for reset) MlbReset(0xFE) is received from INIC.

Bit	Description
0	Disables resetting upon reception of reset command
1	Enables resetting upon reception of reset command

[bit23] MRS : MediaLB Software Reset

When set, this bit resets the MediaLB physical and link layer. It is cleared automatically by hardware after the reset execution is complete.

Bit	Description
0	Normal operation
1	Reset

Note:

- *Even when the MLBn_DCCR:MHRE bit is '1' and the reset operation by the receipt of the reset command is in progress, this bit is reset to '0' after the execution of the reset command.*

[bit22:16] read0 : -

[bit15:8] read0 : -

[bit7:0] MDA[7:0] : MediaLB Device Address

The MediaLB Device Address (MDA[8:1]) sets a unique Device Address (DA) for the MediaLB Device.

The device address is a 16-bit address allocated to identify the MediaLB device.

The DA is used by the system channel MlbScan and MlbReset commands.

The received DA (DA[15:0]) operates as the command target when DA[15:9] and DA[0] are '0' and DA[8:1] matches this bit field.

4.2. MediaLBn System Status Configuration Register (MLBn_SSCR)

The MediaLBn System Status Configuration Register (MLBn_SSCR) is a register to indicate the status of the MediaLB network. MLBn_SSCR is updated for each MediaLB frame.

REGISTER_NAME	YOURPERIn_CSRi
OFFSET	0 + i*4
ACCESS_SIZE	B H W
MULTIPLE	0:15
NUMERIC_TYPE	
OTHER	

MediaLBn System Status Configuration Register (MLBn_SSCR)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	read0	read0	read0	read0	read0	read0	read0	read0
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	read0	read0	read0	read0	read0	read0	read0	read0
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	15	14	13	12	11	10	9	8
BIT_NAME	read0	read0	read0	read0	read0	read0	read0	read0
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	SSRE	SDMU	SDML	SDSC	SDCS	SDNU	SDNL	SDR
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

[bit31:8] read0 : -

[bit7] SSRE : System service request enable

Bit	Description
0	System service request response disabled
1	System service request response enable

System software can set this bit to indicate that the MediaLB Device is present and needs service. When this bit is set to '1', RxStatus (RxStatus (DeviceServiceRequest(0x82)) is transmitted as a response to the MlbScan (0xE4) system command.
When RxStatus is transmitted, hardware clears this bit to '0'.

[bit6] SDMU : System Detects MediaLB Unlock

Bit	Description
0	MediaLB unlock not detected
1	MediaLB unlock detected

This bit is set to indicate that the MediaLB Device has unlocked from the MediaLB frame.
If not masked by the System Mask Configuration Register (MLBn_SMCR:SMMU), a system interrupt is generated when MediaLB unlock is detected.
Write '1' to clear this bit. Writing '0' has no effect. Once set, this bit holds until it is cleared by software.

Note:

- *During the locked state, this bit is set to '1' when the MLBn_DCCR:MRS bit is set to '1'.*

[bit5] SDML : System Detects MediaLB Lock

Bit	Description
0	MediaLB lock not detected
1	MediaLB lock detected

This bit is set to indicate that the MediaLB Device has locked to the MediaLB frame.
If not masked by the System Mask Configuration Register (MLBn_SMCR:SMML), a system interrupt is generated when the MediaLB lock is detected.
Write '1' to clear this bit. Writing '0' has no effect. Once set, this bit holds until it is cleared by software.

[bit4] SDSC : System Detects Subcommand

Bit	Description
0	Subcommand not detected
1	Subcommand detected

This bit is set to indicate that the MediaLB Device has received the MlbSubCmd (0xE6) System Command.
The user-defined software command is stored in the MLBn_SDCR register. The decoding of this command is left up to software.
If not masked by the System Mask Configuration Register (MLBn_SMCR:SMSC), a system interrupt is generated on detecting MlbSubCmd.
Write '1' to clear this bit. Writing '0' has no effect. Once set, this bit holds until it is cleared by software.

[bit3] SDCS : System Detects Channel Scan

Bit	Description
0	Channel scan is not detected
1	Channel scan is detected

This bit is set to indicate that the MediaLB Device has received the MlbScan (0xE4) System Command. The target DeviceAddress is stored in the MLBn_SDCR register. If not masked by the System Mask Configuration Register (MLBn_SMCR:SMCS), a system interrupt is generated on detection of the MlbScan command.
Write '1' to clear this bit. Writing '0' has no effect. Once set, this bit holds until it is cleared by software.

[bit2] SDNU : System Detects Network Unlock

Bit	Description
0	Network unlock not detected
1	Network unlock detected

This bit is set to indicate that the MediaLB Device has received the MOST_Unlock (0xE2) System Command.
If not masked by the System Mask Configuration Register (MLBn_SMCR:SMNU), a system interrupt is generated on detection of the MOST_Unlock command.
Write '1' to clear this bit. Writing '0' has no effect. Once set, this bit holds until it is cleared by software.

[bit1] SDNL : System Detects Network Lock

Bit	Description
0	Network lock not detected
1	Network lock detected

This bit is set to indicate that the MediaLB Device has received the MOST_Lock (0xE0) System Command.
If not masked by the System Mask Configuration Register (MLBn_SMCR:SMNL), a system interrupt is generated on detection of the MOST_Lock command.
Write '1' to clear this bit. Writing '0' has no effect. Once set, this bit holds until it is cleared by software

[bit0] SDR : System Detects Reset

Bit	Description
0	Reset not detected
1	Reset detected

This bit is set to indicate that the MediaLB Device has received the MlbReset (0xFE) System Command. The target DeviceAddress is stored in the MLBn_SDCR register.
If not masked by the System Mask Configuration Register (MLBn_SMCR:SMR), a system interrupt is generated on detection of the MlbReset command.
Write '1' to clear this bit. Writing '0' has no effect. Once set, this bit is sticky until cleared by software.

Note:

- *The MediaLBn System Status Configuration register (MLBn_SSCR) allows system software to monitor and control the status of the MediaLB network. MLBn_SSCR is updated once per frame by hardware during the MediaLB System Channel. Except for the bits associated with MediaLB lock and unlock (MLBn_SSCR.SDMU and MLBn_SSCR.SDML), the bits of the MLBn_SSCR register are not valid until the device is locked to the MediaLB interface. System software must service status events before the start of the next MediaLB frame to prevent the current frame status from being lost.*

4.3. MediaLBn System Data Configuration Register (MLBn_SDCR)

MediaLBn System Data Configuration Register (MLBn_SDCR) is a register to receive the system channel data for the MediaLB frame. SDCR is updated for each MediaLB frame. System software must read MLBn_SDCR before the start of the next MediaLB frame to prevent the current frame data from being lost.

REGISTER_NAME	YOURPERIn_CSRi
OFFSET	0 + i*4
ACCESS_SIZE	B H W
MULTIPLE	0:15
NUMERIC_TYPE	
OTHER	

MediaLBn System Data Configuration Register (MLBn_SDCR)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	MSD[31]	MSD[30]	MSD[29]	MSD[28]	MSD[27]	MSD[26]	MSD[25]	MSD[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

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	MSD[23]	MSD[22]	MSD[21]	MSD[20]	MSD[19]	MSD[18]	MSD[17]	MSD[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

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	MSD[15]	MSD[14]	MSD[13]	MSD[12]	MSD[11]	MSD[10]	MSD[9]	MSD[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

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	MSD[7]	MSD[6]	MSD[5]	MSD[4]	MSD[3]	MSD[2]	MSD[1]	MSD[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] MSD[31:0] : MediaLB System Data

This register is loaded with the data from MLBn_MLBDAT during the System Channel quadlet.
The System Data Configuration register (MLBn_SDCR) allows system software to receive control information from the MediaLB Controller.
MLBn_SDCR is updated once per frame by hardware during the MediaLB System Channel. System software must read MLBn_SDCR before the start of the next MediaLB frame to prevent the current frame data from being lost.

4.4. MediaLBn System Mask Configuration Register (MLBn_SMCR)

MediaLBn System Mask Configuration Register (MLBn_SMCR) sets the system interrupt mask

REGISTER_NAME	YOURPERIn_CSRi
OFFSET	0 + i*4
ACCESS_SIZE	B H W
MULTIPLE	0:15
NUMERIC_TYPE	
OTHER	

MediaLB System Mask Configuration Register (MLBn_SMCR)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	read0	read0	read0	read0	read0	read0	read0	read0
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	read0	read0	read0	read0	read0	read0	read0	read0
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	15	14	13	12	11	10	9	8
BIT_NAME	read0	read0	read0	read0	read0	read0	read0	read0
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	read0	SMMU	SMML	SMSC	SMCS	SMNU	SMNL	SMR
ACCESS_TYPE	R0,WX	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE								
INITIAL_VALUE	0	1	1	0	0	0	0	0

[bit31:8] read0 : -

[bit7] read0 : -

[bit6] SMMU : System Masks MediaLB Unlock

Sets whether system interrupt should be masked when unlock was detected from the MediaLB frame. Detection of unlock from the MediaLB frame is indicated by MLBn_SSCR:SDMU.

Bit	Description
0	Interrupt due to detection of unlock from the MediaLB frame is not masked.
1	Interrupt due to detection of unlock from the MediaLB frame is masked.

[bit5] SMML : System Masks MediaLB Lock

Sets whether the system interrupt should be masked when the lock was detected from the MediaLB frame. Detection of the lock from the MediaLB frame is indicated by MLBn_SSCR:SDML.

Bit	Description
0	Interrupt due to detection of lock with MediaLB frame is not masked.
1	Interrupt due to detection of lock with MediaLB frame is masked.

[bit4] SMSC : System Masks Subcommand

Sets whether system interrupt should be masked when the subcommand of the system command, MlbSubCmd(0xE6) is received. The receipt of a subcommand of the system command, MlbSubCmd(0xE6) is indicated by MLBn_SSCR:SDSC.

Bit	Description
0	Interrupt due to the receipt of the MlbSubCmd(0xE6) system command is not masked.
1	Interrupt due to the receipt of the MlbSubCmd(0xE6) system command is masked.

[bit3] SMCS : System Masks Channel Scan

Sets whether system interrupt should be masked when the system command, MlbScan(0xE4) is received. The receipt of system command, MlbScan(0xE4) is indicated by MLBn_SSCR:SDCS.

Bit	Description
0	Interrupt due to the receipt of the MlbScan(0xE4) system command is not masked.
1	Interrupt due to the receipt of the MlbScan(0xE4) system command is masked.

[bit2] SMNU : System Masks Network Unlock

Sets whether system interrupt should be masked when the system command, MOST_Unlock(0xE2) is received. The receipt of the system command, MOST_Unlock(0xE2) is indicated by MLBn_SSCR:SDNU.

Bit	Description
0	Interrupt due to the receipt of the MOST_Unlock(0xE2) system command is not masked.
1	Interrupt due to the receipt of the MOST_Unlock(0xE2) system command is masked.

[bit1] SMNL : System Masks Network Lock

Sets whether system interrupt should be masked when the system command, MOST_Lock(0xE0) is received. The receipt of system command, MOST_Lock(0xE0) is indicated by MLBn_SSCR:SDNL.

Bit	Description
0	Interrupt due to the receipt of the MOST_Lock(0xE0) system command is not masked
1	Interrupt due to the receipt of the MOST_Lock(0xE0) system command is masked.

[bit0] SMR : System Masks Reset

Sets whether system interrupt should be masked when the system command, MlbReset(0xFE) is received. The receipt of system command, MlbReset(0xFE)is indicated by MLBn_SSCR:SDR.

Bit	Description
0	Interrupt due to the receipt of the MlbReset(0xFE) system command is not masked.
1	Interrupt due to the receipt of the MlbReset(0xFE) system command is masked.

4.5. MediaLBn Version Control Configuration Register (MLBn_VCCR)

MediaLBn Version Control Configuration Register (MLBn_VCCR) is a register to indicate the MediaLB device version.

REGISTER_NAME	YOURPERIn_CSRi
OFFSET	0 + i*4
ACCESS_SIZE	B H W
MULTIPLE	0:15
NUMERIC_TYPE	
OTHER	

MediaLB Version Control Configuration Register (MLBn_VCCR)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	UMA[7]	UMA[6]	UMA[5]	UMA[4]	UMA[3]	UMA[2]	UMA[1]	UMA[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

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	UMI[7]	UMI[6]	UMI[5]	UMI[4]	UMI[3]	UMI[2]	UMI[1]	UMI[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

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	MMA[7]	MMA[6]	MMA[5]	MMA[4]	MMA[3]	MMA[2]	MMA[1]	MMA[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	1	1

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	MMI[7]	MMI[6]	MMI[5]	MMI[4]	MMI[3]	MMI[2]	MMI[1]	MMI[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

[bit31:24] UMA[7:0] : User Major Revision Code

User Major Revision Code.

[bit23:16] UMI[7:0] : User Minor Revision Code

User Minor Revision Code.

[bit15:8] MMA[7:0] : MediaLB Major Revision Code

MediaLB Major Revision Code. This field identifies the major revision of the MediaLB module (OS62400) on this device. This value is hard-coded by the vendor.

Note:

- *For OS62400 version code information, refer to the device datasheet.*

[bit7:0] MMI[7:0] : MediaLB Minor Revision Code

MediaLB Minor Revision Code. This field identifies the minor revision of the MediaLB module (OS62400) implemented on this device. This value is hard-coded by the vendor.

Note:

- *For OS62400 version code information, refer to the device datasheet.*

4.6. MediaLBn Synchronous Base Address Configuration Register (MLBn_SBCR)

MediaLBn Synchronous Base Address Configuration Register (MLBn_SBCR) defines the base address for synchronous receive/transmit system memory buffers.

REGISTER_NAME	YOURPERIn_CSRi
OFFSET	0 + i*4
ACCESS_SIZE	B H W
MULTIPLE	0:15
NUMERIC_TYPE	
OTHER	

MediaLB Synchronous Base Address Configuration Register (MLBn_SBCR)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	SRBA[15]	SRBA[14]	SRBA[13]	SRBA[12]	SRBA[11]	SRBA[10]	SRBA[9]	SRBA[8]
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	SRBA[7]	SRBA[6]	SRBA[5]	SRBA[4]	SRBA[3]	SRBA[2]	SRBA[1]	SRBA[0]
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	STBA[15]	STBA[14]	STBA[13]	STBA[12]	STBA[11]	STBA[10]	STBA[9]	STBA[8]
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	STBA[7]	STBA[6]	STBA[5]	STBA[4]	STBA[3]	STBA[2]	STBA[1]	STBA[0]
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] SRBA[15:0] : Upper Half of Synchronous Receive Base Address for DMA mode

These bits allow the system software to define the base address (SRBA[31:16]) for synchronous receive system memory buffers.
This base address is shared by all synchronous receive channels and defines the upper 16 bits of the 32-bit address for these channels.
This bit field is only used in DMA mode. In IO mode these bits are not used.

[bit15:0] STBA[15:0] : Upper Half of Synchronous Transmit Base Address for DMA mode

These bits allow the system software to define the base address (STBA[31:16]) for synchronous transmit system memory buffers in DMA mode.
This base address is shared by all synchronous transmit channels and defines the upper 16 bits of the 32-bit address for these channels.
This bit field is only used in DMA mode. In IO mode these bits are not used.

4.7. MediaLBn Asynchronous Base Address Configuration Register (MLBn_ABCR)

MediaLBn Asynchronous Base Address Configuration Register (MLBn_ABCR) defines the base address for asynchronous receive/transmit system memory buffers.

REGISTER_NAME	YOURPERIn_CSRi
OFFSET	0 + i*4
ACCESS_SIZE	B H W
MULTIPLE	0:15
NUMERIC_TYPE	
OTHER	

MediaLB Asynchronous Base Address Configuration Register (MLBn_ABCR)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	ARBA[15]	ARBA[14]	ARBA[13]	ARBA[12]	ARBA[11]	ARBA[10]	ARBA[9]	ARBA[8]
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	ARBA[7]	ARBA[6]	ARBA[5]	ARBA[4]	ARBA[3]	ARBA[2]	ARBA[1]	ARBA[0]
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	ATBA[15]	ATBA[14]	ATBA[13]	ATBA[12]	ATBA[11]	ATBA[10]	ATBA[9]	ATBA[8]
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	ATBA[7]	ATBA[6]	ATBA[5]	ATBA[4]	ATBA[3]	ATBA[2]	ATBA[1]	ATBA[0]
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] ARBA[15:0] : Upper Half of Asynchronous Receive Base Address for DMA mode

These bits allow system software to define the base address (ARBA[31:16]) for asynchronous receive system memory buffers.
This base address is shared by all asynchronous receive channels and defines the upper 16 bits of the 32-bit system bus address for these channels.
This bit field is only used in DMA mode. In IO mode these bits are not used.

[bit15:0] ATBA[15:0] : Upper Half of Asynchronous Transmit Base Address for DMA mode

These bits allow the system software to define the base address (ATBA[31:16]) for asynchronous transmit system memory buffers in DMA mode.
This base address is shared by all asynchronous transmit channels and defines the upper 16 bits of the 32-bit system bus address for these channels.
This bit field is only used in DMA mode. In IO mode these bits are not used.

4.8. MediaLBn Control Base Address Configuration Register (MLBn_CBCR)

MediaLBn Control Base Address Configuration Register (MLBn_CBCR) defines the base address for control receive/transmit system memory buffers.

REGISTER_NAME	YOURPERIn_CSRi
OFFSET	0 + i*4
ACCESS_SIZE	B H W
MULTIPLE	0:15
NUMERIC_TYPE	
OTHER	

MediaLB Control Base Address Configuration Register (MLBn_CBCR)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	CRBA[15]	CRBA[14]	CRBA[13]	CRBA[12]	CRBA[11]	CRBA[10]	CRBA[9]	CRBA[8]
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	CRBA[7]	CRBA[6]	CRBA[5]	CRBA[4]	CRBA[3]	CRBA[2]	CRBA[1]	CRBA[0]
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	CTBA[15]	CTBA[14]	CTBA[13]	CTBA[12]	CTBA[11]	CTBA[10]	CTBA[9]	CTBA[8]
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	CTBA[7]	CTBA[6]	CTBA[5]	CTBA[4]	CTBA[3]	CTBA[2]	CTBA[1]	CTBA[0]
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] CRBA[15:0] : Upper Half of Control Receive Base Address for DMA mode

These bits allow the system software to define the base address (CRBA[31:16]) for control receive system memory buffers.
This base address is shared by all control receive channels and defines the upper 16 bits of the 32-bit system bus address for these channels.
This bit field is only used in DMA mode. In IO mode these bits are not used.

[bit15:0] CTBA[15:0] : Upper Half of Control Transmit Base Address for DMA mode

These bits allow the system software to define the base address (CTBA[31:16]) for control transmit system memory buffers in DMA mode.
This base address is shared by all control transmit channels and defines the upper 16 bits of the 32-bit system bus address for these channels.
This bit field is only used in DMA mode. In IO mode these bits are not used.

4.9. MediaLBn Isochronous Base Address Configuration Register (MLBn_IBCR)

MediaLBn Isochronous Base Address Configuration Register (MLBn_IBCR) defines the base address for control receive/transmit system memory buffers.

REGISTER_NAME	YOURPERIn_CSRi
OFFSET	0 + i*4
ACCESS_SIZE	B H W
MULTIPLE	0:15
NUMERIC_TYPE	
OTHER	

MediaLB Isochronous Base Address Configuration Register (MLBn_IBCR)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	IRBA[15]	IRBA[14]	IRBA[13]	IRBA[12]	IRBA[11]	IRBA[10]	IRBA[9]	IRBA[8]
ACCESS_TYPE	RpWp	RpWp	RpWp	RpWp	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	IRBA[7]	IRBA[6]	IRBA[5]	IRBA[4]	IRBA[3]	IRBA[2]	IRBA[1]	IRBA[0]
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	ITBA[15]	ITBA[14]	ITBA[13]	ITBA[12]	ITBA[11]	ITBA[10]	ITBA[9]	ITBA[8]
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	ITBA[7]	ITBA[6]	ITBA[5]	ITBA[4]	ITBA[3]	ITBA[2]	ITBA[1]	ITBA[0]
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] IRBA[15:0] : Upper Half of Isochronous Receive Base Address for DMA mode

These bits allow the system software to define the base address (IRBA[31:16]) for isochronous receive system memory buffers.
This base address is shared by all isochronous receive channels and defines the upper 16 bits of the 32-bit system bus address for these channels.
This bit field is only used in DMA mode. In IO mode these bits are not used.

[bit15:0] ITBA[15:0] : Upper Half of Isochronous Transmit Base Address for DMA mode

These bits allow the system software to define the base address (ITBA[31:16]) for isochronous transmit system memory buffers in DMA mode.
This base address is shared by all isochronous transmit channels and defines the upper 16 bits of the 32-bit system bus address for these channels.
This bit field is only used in DMA mode. In IO mode these bits are not used.

4.10. MediaLBn Channel Interrupt Configuration Register (MLBn_CICR)

MediaLBn Channel Interrupt Configuration Register (MLBn_CICR) reflects the channel interrupt status. These bits are set by hardware when a channel interrupt is generated.

REGISTER_NAME	YOURPERIn_CSRi
OFFSET	0 + i*4
ACCESS_SIZE	B H W
MULTIPLE	0:15
NUMERIC_TYPE	
OTHER	

MediaLB Channel Interrupt Configuration Register (MLBn_CICR)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	read0	read0	read0	read0	read0	read0	read0	read0
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	read0	read0	read0	read0	read0	read0	read0	read0
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	15	14	13	12	11	10	9	8
BIT_NAME	CNSU[15]	CNSU[14]	CNSU[13]	CNSU[12]	CNSU[11]	CNSU[10]	CNSU[9]	CNSU[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

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	CNSU[7]	CNSU[6]	CNSU[5]	CNSU[4]	CNSU[3]	CNSU[2]	CNSU[1]	CNSU[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

[bit31:16] read0 : -

[bit15:0] CNSU[15:0] : Channel Status Update (for Channel 15 through 0)

CNSU[n] indicates whether there is an interrupt for Channel n.

Bit	Description
0	If CNSU[n] is '0' there is no interrupt for Channel n.
1	If CNSU[n] is '1' Channel n has an interrupt.

Note:

– *Writing to the MLBn_CICR register has no effect. To clear a particular bit in MLBn_CICR, software must clear all of the unmasked status bits in the corresponding MLBn_CSCRn register. For example, if MLBn_CNSU[4] is set, writing FFFFh to MLBn_CSCR4[15:0] releases the channel interrupt (MLBn_CINT) and clears MLBn_CICR[4].*

4.11. MediaLBn AHB Master Control Register (MLBn_AHBMCTL)

MediaLBn AHB Master Control Register (MLBn_AHBMCTL) controls the AHB Bus Request from MediaLB Master.

REGISTER_NAME	YOURPERIn_CSRi
OFFSET	0 + i*4
ACCESS_SIZE	B H W
MULTIPLE	0:15
NUMERIC_TYPE	
OTHER	

MediaLB AHB Master Control Register (MLBn_AHBMCTL)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	MAXTRANS[15]	MAXTRANS[14]	MAXTRANS[13]	MAXTRANS[12]	MAXTRANS[11]	MAXTRANS[10]	MAXTRANS[9]	MAXTRANS[8]
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	MAXTRANS[7]	MAXTRANS[6]	MAXTRANS[5]	MAXTRANS[4]	MAXTRANS[3]	MAXTRANS[2]	MAXTRANS[1]	MAXTRANS[0]
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	MCYCNONREQ[15]	MCYCNONREQ[14]	MCYCNONREQ[13]	MCYCNONREQ[12]	MCYCNONREQ[11]	MCYCNONREQ[10]	MCYCNONREQ[9]	MCYCNONREQ[8]
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	MCYCNONREQ[7]	MCYCNONREQ[6]	MCYCNONREQ[5]	MCYCNONREQ[4]	MCYCNONREQ[3]	MCYCNONREQ[2]	MCYCNONREQ[1]	MCYCNONREQ[0]
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] MAXTRANS[15:0]
Always write "0" to this register. Read value is "X".

[bit15:0] MCYCNONREQ[15:0]
Always write "0" to this register. Read value is "X".

4.12. MediaLBn Channel Entry Configuration Register (MLBn_CECR0 - MLBn_CECR15)

MediaLBn Channel Entry Configuration Register (MLBn_CECRn) defines basic attributes for a given logical channel, such as the channel enable, channel type, channel direction, and channel address. There are 16 such registers corresponding to 16 logical MediaLB channels. The definition of the bit fields in the MLBn_CECRn register depends on the selected channel type.

REGISTER_NAME	YOURPERIn_CSRi
OFFSET	0 + i*4
ACCESS_SIZE	B H W
MULTIPLE	0:15
NUMERIC_TYPE	
OTHER	

MediaLB Channel Entry Configuration Register (MLBn_CECR0 - MLBn_CECR15)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	CE	TR	CT[1]	CT[0]	FCE	MDS[1]	MDS[0]	read0
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	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	MASK[7]	MASK[6]	MASK[5]	MASK[4]	MASK[3]	MASK[2]	MASK[1]	MASK[0]
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	IPL[7]	IPL[6]	IPL[5]	IPL[4]	IPL[3]	IPL[2]	IPL[1]	IPL[0]
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	CA[7]	CA[6]	CA[5]	CA[4]	CA[3]	CA[2]	CA[1]	CA[0]
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] CE : Channel Enable

Enables channels

Bit	Description
0	Channel Disabled
1	Channel enabled

[bit30] TR : Channel Transmit Select
Sets whether the channel is for transmission or reception.

Bit	Description
0	Receive
1	Transmit

[bit29:28] CT[1:0] : Channel Type Select
Sets the channel type selection.

Bit[1:0]	Description
00	Synchronous
01	Isochronous
10	Asynchronous
11	Control

[bit27] FCE : Flow Control Enable
For Isochronous channel, when set allows an isochronous receive channel to generate the ReceiverBusy (0x10) response.

Bit	Description
0	Generation of ReceiverBusy(0x10) response prohibited
1	Generation of ReceiverBusy(0x10) response allowed

For Asynchronous and control channel, this bit (PCE) sets whether the reception packet counter should be enabled. This bit is valid for asynchronous and control receive channels in IO Mode only.

Bit	Description
0	Reception packet counter disable
1	Reception packet counter enable

For Synchronous channel this bit (FSE) sets whether the frame synchronization for the streaming channel should be enabled.

Bit	Description
0	Frame synchronization disable
1	Frame synchronization enable

[bit26:25] MDS[1:0] : Channel Mode Select
Sets the channel mode.

Bit[1:0]	Description
00	DMA mode enable (Ping-pong buffering)
01	DMA mode enable (Circular buffering)
10	IO mode enable
11	Reserved. It is prohibited to set MDS[1:0] = '11'

Set either DMA mode or IO mode for the channels to be used. All channels must be set to either DMA mode or IO mode. It is prohibited to set the IO mode and DMA mode in a mixed manner.

[bit24] read0 : -

[bit23:16] MASK[7:0] : Channel interrupt mask
MASK[7] - Reserved bit. Read value is '0'. Write always '0' to this bit.

MASK[6] - Sets whether channel interrupt due to the frame sync lost should be masked. The status bit targeted for masking is CSCRn:STS[6].

Bit	Description
0	Lost frame synchronization interrupt is not masked
1	Lost frame synchronization interrupt is masked

MASK[5] - Reserved bit. Read value is '0'. Write always '0' to this bit.

MASK[4] - Sets whether the channel interrupt due to a buffer error should be masked. The status bit targeted for masking is CSCRn:STS[4].

Bit	Description
0	Buffer error channel interrupt is not masked
1	Buffer error channel interrupt is masked

MASK[3] - This bit has different interpretation depending on DMA-mode or IO-mode and sets the mask for channel interrupts. The status bit targeted for masking is CSCRn:STS[3].
For DMA mode, it masks the buffer start interrupt.
For IO mode, it masks the transmission service request interrupt.

Bit	Description
0	Channel interrupt is not masked
1	Channel interrupt is masked

MASK[2] - This bit has different interpretation depending on DMA-mode or IO-mode and sets the mask for channel interrupts. The status bit targeted for masking is CSCRn:STS[2].
For DMA mode, it masks the buffer end interrupt.
For IO mode, it masks the reception service request interrupt or reception packet abort interrupt.

Bit	Description
0	Channel interrupt is not masked
1	Channel interrupt is masked

MASK[1] - Sets whether the channel interrupt due to the detection of break should be masked. The status bit targeted for masking is CSCRn:STS[1].

Bit	Description
0	Break detection channel interrupt is not masked
1	Break detection channel interrupt is masked

MASK[0] - Sets whether the channel interrupt due a protocol error should be masked. The status bit targeted for masking is CSCRn:STS[0].

Bit	Description
0	Protocol error channel interrupt is not masked
1	Protocol error channel interrupt is masked

[bit15:8] IPL[7:0] : Packet Length

- For Isochronous channels:

For Isochronous transmit channels these bits define the number of packet bytes. The smallest isochronous packet size per frame is 5 bytes (IPL[7:0] >= 5).
For Isochronous receive channels, software must program IPL[7:2] to indicate the expected number of bytes per packet, where as IPL[1:0] always equals '00'. A packet length of 8 is indicated by IPL=0x08, a packet length of 12 is indicated by IPL=0x0C, a packet length of 252 is indicated by IPL=0xFC. However, a packet length of 256 bytes is represented as IPL[7:0]=0x00.

- For Synchronous channel:

IPL[7]/FSCD sets frame synchronization disable.

Bit	Description
0	Frame synchronization channel is not disable
1	Frame synchronization channel disabled

IPL[6:5] - These bits are not used.
IPL[4:0]/FSPC[4:0] sets the number of frame synchronous physical channels. In other words, it defines the number of physical channels that match with the channel address (CECRn:CA[8:1]) among the synchronous channels contained in one frame.

- For Asynchronous/Control channel:

IPL[7:5] - These bits are not used for Asynchronous/Control channel.
IPL[4:0]/PCTH[4:0] sets the packet count threshold. This bit field sets the number of packets to receive before generating a receive packet-count service request.
When the number of received packets has reached this setting, a reception service request is generated. Also, when the local channel buffer has become full, a reception service request is generated.
When using this bit, it is recommended that the software sets LCBCRn:TH[6:0]=0x00. Set "000" for IPL[7:5]. PCTH[4:0] are valid only in the IO mode.

[bit7:0] CA[7:0] : Channel Address

Sets the channel address of the logical channel.
These bits determine the ChannelAddress (CA[8:1) associated with this logical channel. This value is matched against the ChannelAddress of each received physical channel from the MediaLB Controller. There is a ChannelAddress match if and only if the ChannelAddress recovered from the MediaLB input, MLBn_MLBSIG, equals the ChannelAddress defined by:

CA[15:0] = {7'h00, CA[8:1], 1'b0}

4.13. MediaLBn Channel Status Configuration Register (MLBn_CSCR0 - MLBn_CSCR15)

MediaLBn Channel Status Configuration Register (MLBn_CSCRn) reflects the status of the current buffer and previous buffer for the logical channel n. There are 16 such registers corresponding to 16 logical MediaLB channels. The definitions of the bit fields in the MLBn_CSCRn register depends on the selected channel type.

REGISTER_NAME	YOURPERIn_CSRi
OFFSET	0 + i*4
ACCESS_SIZE	B H W
MULTIPLE	0:15
NUMERIC_TYPE	
OTHER	

MediaLB Channel Status Configuration Register (MLBn_CSCR0 - MLBn_CSCR15)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	BM	BF	read0	read0	read0	read0	read0	read0
ACCESS_TYPE	R,WX	R,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX
PROT_TYPE								
INITIAL_VALUE	1	0	0	0	0	0	0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	read0	read0	read0	read0	IVB[1]	IVB[0]	GB	RDY
ACCESS_TYPE	R0,WX	R0,WX	R0,WX	R0,WX	R,WX	R,WX	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	STS[3]	STS[2]	STS[1]	STS[0]	STS	STS	STS	STS
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	STS	STS	STS	STS	STS	STS	STS	STS
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] BM : Buffer Empty

This bit indicates that the local channel buffer is empty.

Bit	Description
0	Local channel buffer not empty
1	Local channel buffer empty

Note:

- *This bit is set and cleared by hardware.*

[bit30] BF : Buffer Full
This bit indicates that the local channel buffer is full.

Bit	Description
0	Local channel buffer not full
1	Local channel buffer full

Note:
- *This bit is set and cleared by hardware.*

[bit29:20] read0 : -

[bit19:18] IVB[1:0] : Isochronous Valid Bytes
These bits are loaded by hardware with the number of valid bytes in the last packet of a broken Isochronous receive channel. Used in conjunction with CCBCRn.BCA, IVB[1:0] can be used by software to determine the final valid byte of the local channel buffer.

Note:
- *Valid for local channel buffers configured for isochronous RX data.*

Bit[1:0]	Description
00	Final valid byte = (CCBCRn.BCA - 5)
01	Final valid byte = (CCBCRn.BCA - 4).
10	Final valid byte = (CCBCRn.BCA - 3)
11	Final valid byte = (CCBCRn.BCA - 2)

[bit17] GB : Generate Break
This bit has different interpretation depending on the channel configuration.

For Synchronous channel, this bit is not used.

For Isochronous channel, the function of this bit is to Generate the Isochronous Receive Break (GIRB). When set, this bit causes the hardware to terminate the current packet, flush the local channel buffer, clear the RDY bit, and load CSCRn:IVB[1:0]. This bit is set by system software and cleared by hardware. It is valid for local channel buffers configured for isochronous RX data.

For Asynchronous and Control Channel, this bit (GB) enables break generation. When the local channel buffer is configured for transmitting data, the setting of this bit causes the hardware to send the AsyncBreak (0x26) or ControlBreak (0x36) command and stop the transfer. When the local channel buffer is configured for receiving data, the setting of this bit causes hardware to send the MediaLB RxStatus ReceiverBreak (0x70) and stop the transfer. This bit is set by system software and cleared by hardware.

[bit16] RDY : Next Buffer Ready
This bit has a different interpretation depending on DMA-mode or IO-mode.

In IO Mode this bit is not used.

In DMA Mode, system software should set this bit when all the registers, data and program memory variables are setup and ready to transmit or receive data in DMA Mode.
For transmitting data, the system memory buffer should also be filled. For DMA Mode using ping-pong buffering, hardware clears this bit after the buffer begins to be processed.
In DMA Mode using circular buffering, the software should clear this bit only when buffer processing needs to halted.

[bit15:12] STS : STS[15-12] Reserved
Read value is '0'. Write always '0' to these bits.

[bit11] STS : STS[11]
This bit has a different interpretation depending on DMA-mode or IO-mode.

In DMA mode this is the Previous Buffer Start bit. When set, this bit indicates the first quadlet of the Previous Buffer has been successfully transmitted or received. The setting of this bit generates a maskable channel interrupt to system software. This bit is valid for all channel types.
Write '1' to clear this bit. Writing '0' has no effect.
Once set, this bit holds until it is cleared by software.

In IO Mode, this bit is not used.

[bit10] STS : STS[10]
This bit has a different interpretation depending on DMA-mode or IO-mode.

In DMA mode this is the Previous Buffer Done bit. When set, this bit indicates the last quadlet of the Previous Buffer has been successfully transmitted or received. The setting of this bit generates a maskable channel interrupt to system software. This bit is valid for all channel types.
Write '1' to clear this bit. Writing '0' has no effect.
Once set, this bit holds until it is cleared by software.

In IO Mode, this bit is not used.

[bit9] STS : STS[9]
This bit has different interpretation depending on DMA-mode or IO-mode.

In DMA mode this is the Previous Buffer Detect Break bit. When set, this bit indicates that either a transmit channel has detected a receiver break response, ReceiverBreak (0x70), or a receive channel has detected a transmitter break command, ControlBreak (0x36) or AsyncBreak (0x26), while processing the Previous Buffer.
The setting of this bit generates a maskable channel interrupt to system software. This bit is valid for all channel types.
Write '1' to clear this bit. Writing '0' has no effect.
Once set, this bit holds until it is cleared by software.

In IO Mode this is Receive Packet Start bit. When set, this bit indicates that an RX channel has detected a transmitter packet start command; ControlStart (0x30) or AsyncStart (0x20). This status bit can be used by system software to detect when it has reached the end of an aborted packet. This bit is valid for asynchronous and control RX channels only.

Note:

- *In IO Mode, the STS[9] bit cannot be programmed to generate a channel interrupt. System software must poll this bit following a Receive Packet Abort (see MLBn_CSCR:STS[8]). As each quadlet of the broken packet is popped from the local channel buffer, software must check to see if the next quadlet is the start of a new packet. When software detects a Receive Packet Start, it can start processing valid data.*

Write '1' to clear this bit. Writing '0' has no effect.
Once set, this bit holds until it is cleared by software.

[bit8] STS : STS[8]
This bit has different interpretation depending on DMA-mode or IO-mode.

In DMA mode this is Previous Buffer Protocol Error bit. When set, this bit indicates that either a transmit channel has detected an RxStatus of ReceiverProtocolError (0x72), a receive channel has detected an invalid command for this channel type, or an additional AsyncStart (0x20) or ControlStart (0x30) command has been received while in the middle of a packet.
The setting of this bit generates a maskable channel interrupt to system software. This bit is valid for all receive channels and valid for only asynchronous and control transmit channels.

In IO mode this is Receive Packet Abort bit. When set, this bit indicates that a receive channel has detected an aborted packet. Received packets are aborted if the receiver generates a break response, ReceiverBreak (0x70), or detects a transmitter packet break command; ControlBreak (0x36) or AsyncBreak (0x26).

This bit can also indicate the receive channel has detected a transmit command protocol error. The setting of this bit generates a maskable channel interrupt to system software. This interrupt can be used by system software to detect when it has encountered the beginning of an aborted packet. This bit is valid for asynchronous and control receive channels only.
Write '1' to clear this bit. Writing '0' has no effect.
Once set, this bit holds until it is cleared by softwar

[bit7] STS : STS[7] - Reserved
Read value is '0'. Always write '0' to this bits.

[bit6] STS : STS[6] - Lost Frame Synchronization bit
When set, this bit indicates that the logical channel has lost synchronization with the MediaLB frame. The setting of this bit generates a maskable channel interrupt to system software. This bit is valid for synchronous channels only.
Write '1' to clear this bit. Writing '0' has no effect.
Once set, this bit holds until it is cleared by software.

[bit5] STS : STS[5]
This bit has a different interpretation depending on DMA-mode or IO-mode.

In DMA mode this is Host Bus Error bit. When set, this bit indicates that an HBI bus error has been detected. The setting of this bit generates a non-maskable channel interrupt to system software.
Write '1' to clear this bit. Writing '0' has no effect.
Once set, this bit holds until it is cleared by software.

In IO Mode, this bit is not used.

[bit4] STS : STS[4] : Buffer Error bit
When set, this bit indicates that either a transmit channel has detected a buffer underflow (e.g., attempted to pop data from an empty buffer), or a receive channel has detected a buffer overflow (e.g., attempted to push data onto a full buffer).
The setting of this bit generates a maskable channel interrupt to the system software. This bit is valid for synchronous receive/transmit and isochronous receive (MLBn_CECRn:FCE = 0) channels only.
Write '1' to clear this bit. Writing '0' has no effect.
Once set, this bit holds until it is cleared by software.

[bit3] STS : STS[3]
This bit has different interpretation depending on DMA-mode or IO-mode.

In DMA mode this is Current Buffer Start bit. When set, this bit indicates that the DMA controller has started processing the Current Buffer. This bit is set after the contents of MLBn_CNBCRn have been loaded into MLBn_CCBCRn, the MLBn_CSCRn:RDY bit has been cleared (for ping-pong buffering), and hardware is available to accept the next buffer.
The setting of this bit generates a maskable channel interrupt to system software. This bit is valid for all channel types.

In IO mode this is a Transmit Service Request bit. When set, it indicates that a transmit channel requests service from the system software. Transmit service requests are issued if the number of valid quadlets in the local channel buffer is less than or equal to MLBn_LCBCRn:TH[6:0].
The setting of this bit generates a maskable channel interrupt to the system software. This bit is valid for all channel types.
Write '1' to clear this bit. Writing '0' has no effect.
Once set, this bit holds until it is cleared by software.

[bit2] STS : STS[2]
This bit has different interpretation depending on DMA-mode or IO-mode.

In DMA mode this is Current Buffer Done bit. When set, this bit indicates that the last quadlet from the last packet (in the Current Buffer) has been successfully transmitted or received. The setting of this bit generates a maskable channel interrupt to system software. This bit is valid for all channel types.

In IO mode this bit is the Receive Service Request bit. When set, this bit indicates that a receive channel is requesting service from system software. Receive service requests are issued if the number of free quadlets in the local channel buffer is less than or equal to MLBn_LCBCRn:TH[6:0]. The setting of this bit generates a maskable channel interrupt to system software. This bit is valid for all channel types.
Write '1' to clear this bit. Writing '0' has no effect.
Once set, this bit holds until it is cleared by software.

[bit1] STS : STS[1] : Current Buffer Detect Break bit.
When set, this bit indicates that either a transmit channel has detected a receiver break response, ReceiverBreak (0x70), or a receive channel has detected a transmitter break command, ControlBreak (0x36) or AsyncBreak (0x26), while processing the Current Buffer. The setting of this bit generates a maskable channel interrupt to system software. This bit is valid for asynchronous and control channels only.
Write '1' to clear this bit. Writing '0' has no effect.
Once set, this bit holds until it is cleared by software.

[bit0] STS : STS[0] : Current Buffer Protocol Error bit
This bit indicates that either a transmit channel has detected an RxStatus of ReceiverProtocolError (0x72), a receive channel has detected an invalid command for a given channel type, or an additional ControlStart (0x30) or AsyncStart (0x20) command has been received while in the middle of a packet. The setting of this bit generates a maskable channel interrupt to the system software. This bit is valid for all RX channel types and valid for only asynchronous and control TX channels.
Write '1' to clear this bit. Writing '0' has no effect.
Once set, this bit holds until it is cleared by software.

4.14. MediaLBn Channel Current Buffer Configuration Register (MLBn_CCBCR0 - MLBn_CCBCR15)

MediaLBn Channel n Current Buffer Configuration Register (MLBn_CCBCRn) indicates the address pointer and buffer length of the Current Buffer in system memory for the logical channel n when configured for DMA Mode. When configured for IO Mode it is receive data buffer. There are 16 such registers corresponding to 16 logical MediaLB channels. The definition of the bit fields in the MLBn_CCBCRn register depends on the selected channel type.

REGISTER_NAME	YOURPERIn_CSRi
OFFSET	0 + i*4
ACCESS_SIZE	B H W
MULTIPLE	0:15
NUMERIC_TYPE	
OTHER	

MediaLB Channel Current Buffer Configuration Register (MLBn_CCBCR0 -

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	BCA[15]	BCA[14]	BCA[13]	BCA[12]	BCA[11]	BCA[10]	BCA[9]	BCA[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

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	BCA[7]	BCA[6]	BCA[5]	BCA[4]	BCA[3]	BCA[2]	BCA[1]	BCA[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

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	BFA[15]	BFA[14]	BFA[13]	BFA[12]	BFA[11]	BFA[10]	BFA[9]	BFA[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

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	BFA[7]	BFA[6]	BFA[5]	BFA[4]	BFA[3]	BFA[2]	BFA[1]	BFA[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

[bit31:16] BCA[15:0] : Buffer Current Address

This bit field has different interpretations for DMA mode and IO mode.

In DMA mode, the BCA field defines a 16-bit address pointer, which identifies the lower half of the beginning address of the Current Buffer in system memory. The BCA[15:2] bits are loaded from MLBn_CNBCRn.BSA[15:2] when the Next Buffer is ready for processing. This Current Buffer address pointer, except when associated with isochronous channels, should always be quadlet aligned (i.e., set BCA[1:0] to '00' for Synchronous, Asynchronous and Control channels). During the processing of the Current Buffer, the BCA field marks which quadlet of the buffer is currently being processed. The upper half of the beginning address of the Current Buffer is system memory is defined by MLBn_SBCR.SRBA, MLBn_ABCR.ARBA, MLBn_CBCR.CRBA, or MLBn_IBCR.IRBA when MLBn_CECRn.TR is clear; MLBn_SBCR.STBA, MLBn_ABCR.ATBA, MLBn_CBCR.CTBA, or MLBn_IBCR.ITBA when MLBn_CECRn.TR is set, dependant on the value of MLBn_CECRn.CT[1:0].

In IO mode, this bit field defines the Receive Data Buffer bits - RDB[31:16].

This field contains the upper half of the next quadlet of receive data when the logical channel is configured as receive channel.

[bit15:0] BFA[15:0] : Buffer Final Address

This bit field has different interpretations for DMA mode and IO mode.

In DMA mode, the BFA field defines a 16-bit address pointer, which identifies the lower half of the ending address of the Current Buffer in system memory. The BFA[15:2] bits are loaded from MLBn_CNBCRn.BEA[15:2] when the Next Buffer is read for processing. This Current Buffer address pointer, except when associated with isochronous channels, should always be quadlet aligned (i.e., BFA[1:0] equals '00' for Synchronous, Asynchronous and Control channels). During the processing of the Current Buffer, the point at which the BCA field becomes equal to (or greater than) the BFA field indicates that the processing of the Current Buffer will end upon successful completion of the current quadlet (for isochronous and synchronous channels) or upon successful completion of the current packet (for asynchronous and control channels). It is the responsibility of system software to ensure the system memory buffers (for RX asynchronous and control channels) can accommodate overflow in the size of the largest packet supported. Additionally, single-packet buffering can be used by simply programming MLBn_CNBCRn.BSA[15:2] = MLBn_CNBCRn.BEA[15:2].

The upper half of the ending address of the Current Buffer in system memory is defined by MLBn_SBCR.SRBA, MLBn_ABCR.ARBA, MLBn_CBCR.CRBA, and MLBn_IBCR.IRBA when MLBn_CECRn.TR is clear; MLBn_SBCR.STBA, MLBn_ABCR.ATBA, MLBn_CBCR.CTBA, or MLBn_IBCR.ITBA when MLBn_CECRn.TR is set, dependant on the value of MLBn_CECRn.CT[1:0].

In IO mode, this bit field defines the Receive Data Buffer bits - RDB[15:0].
This field contains the lower half of the next quadlet of receive data when the logical channel is configured as receive channel.

4.15. MediaLBn Channel Next Buffer Configuration Register (MLBn_CNBCR0 - MLBn_CNBCR15)

The Channel n Next Buffer Configuration Register (MLBn_CNBCRn) configures the start and end addresses of the Next Buffer in system memory for the logical channel when configured for DMA Mode. When configured for IO Mode, the MLBn_CNBCRn registers is the Transmit Data Buffer. There are 16 such registers corresponding to 16 logical MediaLB channels. The definition of the bit fields in the MLBn_CNBCRn register depend on the selected channel type.

REGISTER_NAME	YOURPERIn_CSRi
OFFSET	0 + i*4
ACCESS_SIZE	B H W
MULTIPLE	0:15
NUMERIC_TYPE	
OTHER	

MediaLBn Channel Next Buffer Configuration Register (MLBn_CNBCR0 - MLBn_CNBCR15)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	BSA[15]	BSA[14]	BSA[13]	BSA[12]	BSA[11]	BSA[10]	BSA[9]	BSA[8]
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	BSA[7]	BSA[6]	BSA[5]	BSA[4]	BSA[3]	BSA[2]	BSA[1]	BSA[0]
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	BEA[15]	BEA[14]	BEA[13]	BEA[12]	BEA[11]	BEA[10]	BEA[9]	BEA[8]
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	BEA[7]	BEA[6]	BEA[5]	BEA[4]	BEA[3]	BEA[2]	BEA[1]	BEA[0]
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] BSA[15:0] : Buffer Start Address bits

This bit field has different interpretations for DMA mode and IO mode.

In DMA mode, the BSA field defines a 16-bit address pointer, which identifies the lower half of the beginning address of the Next Buffer in system memory. Once system software detects MLBn_CSCRn.RDY has been cleared by hardware (for ping-pong buffering), the beginning address of the Next Buffer may be loaded into BSA[15:2]. System software should then set MLBn_CSCRn.RDY. Once processing of the Current Buffer for the logical channel is complete, the BSA[15:2] field is loaded into the MLBn_CCBCRn.BCA[15:2] field and processing of the next buffer can begin. This Next Buffer address pointer must always be quadlet aligned (e.g., BSA[1:0] must be written as '00').
The upper half of the beginning address of the Next Buffer in system memory is defined by MLBn_SBCR.SRBA, MLBn_ABCR.ARBA, MLBn_CBCR.CRBA, or MLBn_IBCR.IRBA when MLBn_CECRn.TR is clear; MLBn_SBCR.STBA, MLBn_ABCR.ATBA, MLBn_CBCR.CTBA, or MLBn_IBCR.ITBA when MLBn_CECRn.TR is set, dependant on the value of MLBn_CECRn.CT[1:0].

Note:

- *For BSA[1:0] bits in DMA mode, read value is 'X'. Write always '0' to these bits.*
  In IO mode, this bit field defines the Transmit Data Buffer bits - TDB[31:16].
  This field contains the upper half of the next quadlet of transmit data when the logical channel is configured as transmit channel.

[bit15:0] BEA[15:0] : Buffer End Address bits

This bit field has different interpretations for DMA mode and IO mode.

In DMA mode, the BEA field defines a 16-bit address pointer, which identifies the lower half of the ending address of the Next Buffer in system memory. Once system software detects MLBn_CSCRn.RDY has been cleared by hardware (for ping-pong buffering), the ending address of the Next Buffer may be loaded into BEA[15:2]. System software should then set MLBn_CSCRn.RDY. Once processing of the Current Buffer for the logical channel is complete, the BEA[15:2] field is loaded into the MLBn_CCBCRn.BFA[15:2] field and processing of the next buffer can begin. The BEA[15:2] bits are loaded into MLBn_CCBCRn.BFA[15:2] when the Current Buffer is finished being processed. This Next Buffer address pointer, except when associated with isochronous channels, should always be quadlet aligned (i.e., set BEA[1:0] to '00' for Synchronous, Asynchronous and Control channels).
The upper half of the ending address of the Next Buffer in system memory is defined by MLBn_SBCR.SRBA, MLBn_ABCR.ARBA, MLBn_CBCR.CRBA, and MLBn_IBCR.IRBA when MLBn_CECRn.TR is clear; MLBn_SBCR.STBA, MLBn_ABCR.ATBA, MLBn_CBCR.CTBA, or MLBn_IBCR.ITBA when MLBn_CECRn.TR is set, dependant on the value of MLBn_CECRn.CT[1:0].

In IO mode, this bit field defines the Transmit Data Buffer bits - TDB[15:0].
This field contains the lower half of the next quadlet of transmit data when the logical channel is configured as transmit channel.

4.16. MediaLBn Local Channel Buffer Configuration Register (MLBn_LCBCR0 - MLBn_LCBCR15)

MediaLBn Local Channel n Buffer Configuration Register (MLBn_LCBCRn) configures the allocation of the local channel buffer area on the RAM for the local channel buffer. There are 16 such registers corresponding to 16 logical MediaLB channels. This register should only be written while the logical channel is disabled. Writing to this register while the corresponding logical channel is enabled may result in unexpected behavior.

REGISTER_NAME	YOURPERIn_CSRi
OFFSET	0 + i*4
ACCESS_SIZE	B H W
MULTIPLE	0:15
NUMERIC_TYPE	
OTHER	

MediaLB Local Channel Buffer Configuration Register (MLBn_LCBCR0 - MLBn_LCBCR15)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	read0	read0	read0	TH[6]	TH[5]	TH[4]	TH[3]	TH[2]
ACCESS_TYPE	R0,WX	R0,WX	R0,WX	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	TH[1]	TH[0]	read0	read0	read0	BD[5]	BD[4]	BD[3]
ACCESS_TYPE	R/W	R/W	R0,WX	R0,WX	R0,WX	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	BD[2]	BD[1]	BD[0]	read0	read0	read0	read0	read0
ACCESS_TYPE	R/W	R/W	R/W	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	read0	read0	SA[5]	SA[4]	SA[3]	SA[2]	SA[1]	SA[0]
ACCESS_TYPE	R0,WX	R0,WX	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE								
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:29] read0 : -

[bit28:22] TH[6:0] : Buffer Threshold

TH[6:0] sets the threshold of the local channel buffer. The buffer threshold is set in units of 2 quadlets. This bit is used to determine whether a transmission/reception service request should be issued.

Bit[6:0]	Description
0x00	When TH[6:0] is 0x00, an reception service request is generated if the buffer is full and a transmission service request is generated when the buffer is empty.
0x01	Threshold = 2 quadlets
0x02	Threshold = 4 quadlets
...	...
0x7F	Threshold = 254 quadlets

[bit21:19] read0 : -

[bit18:13] BD[5:0] : Buffer Depth

BD[5:0] sets the depth of the local channel buffer. The depth is set in units of 4 quadlets.

Bit[5:0]	Description
0x00	Depth= 4 quadlets
0x01	Depth= 8 quadlets
0x02	Depth= 12 quadlets
...	...
0x3F	Depth= 256 quadlets

[bit12:6] read0 : -

[bit5:0] SA[5:0] : Buffer Start Address

SA[5:0] sets the start address of the local channel buffer.

Bit[5:0]	Description
0x00	RAM Start Address offset of 0 quadlets
0x01	RAM Start Address offset of 4 quadlets
0x02	RAM Start Address offset of 8 quadlets
...	...
0x3F	RAM Start Address offset of 252 quadlets

Note:

- *The initial value of SA[5:0] is n*4, where n is the channel number (0,1,2 ... 15). The initial value of SA[5:0] for Channel 0 is '0x0', for Channel 1 is '0x4', for Channel 2 is '0x8' and so on.*

4.17. MediaLBn Module Identification Register (MLBn_MID)

This register identifies the particular version of the MediaLB hardware used in the device.

REGISTER_NAME	YOURPERIn_CSRi
OFFSET	0 + i*4
ACCESS_SIZE	B H W
MULTIPLE	0:15
NUMERIC_TYPE	
OTHER	

MediaLB Module Identification Register (MLBn_MID)

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	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	MID[23]	MID[22]	MID[21]	MID[20]	MID[19]	MID[18]	MID[17]	MID[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

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	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	MID[7]	MID[6]	MID[5]	MID[4]	MID[3]	MID[2]	MID[1]	MID[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] MID[31:0] : Module ID

The MediaLB module implemented in the device may vary from device to device. This register identifies the particular version of the hardware used in the device. This register helps in developing software to the hardware version implemented in the device.

Note:

- *Please refer to the device specific data sheet for the MID value.*

CHAPTER 23: Stereo Audio DAC

This chapter presents the Stereo Audio Digital to Analog Converter(DAC).

1. Overview
2. Configuration and Block Diagram
3. Operation
4. Registers

CODE: ADAC-S6J3300-E1

1. Overview

This section provides a general description of the Stereo Audio DAC

Also refer to the chapter of "Sound System configuration" how to configure the Stereo Audio DAC.

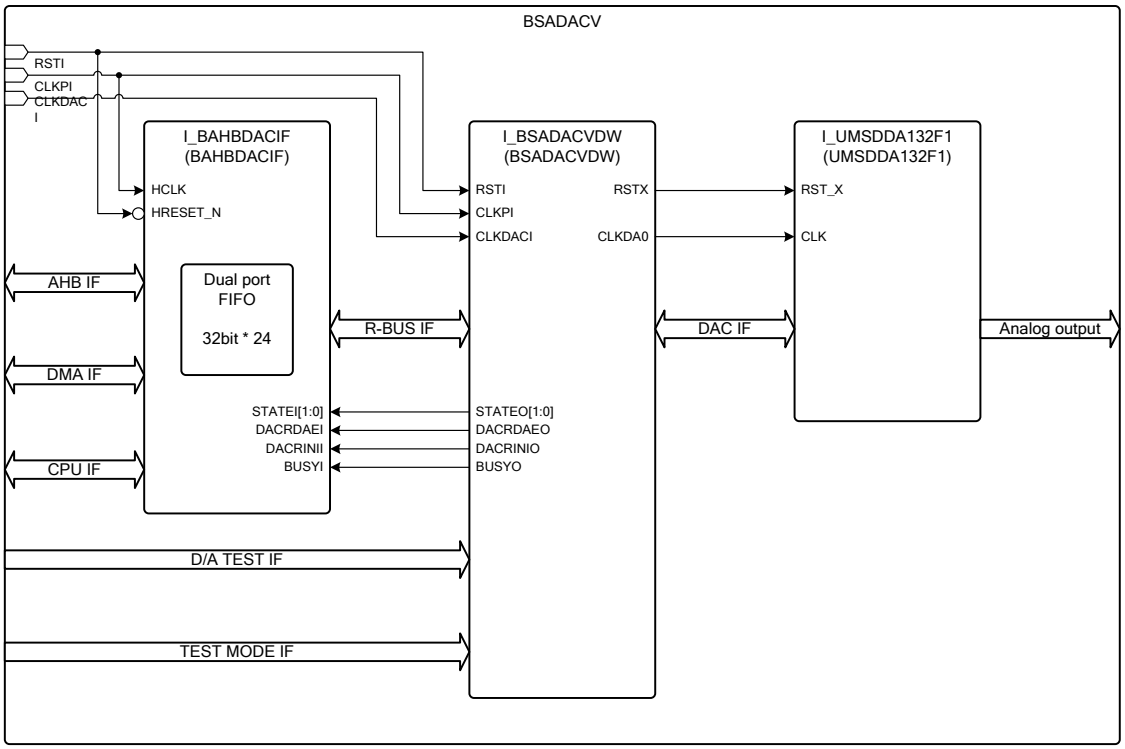
The Stereo Audio DAC is a dual-channel, audio digital-to-analog converter (DAC) that employs a delta-sigma modulator, with a sampling rate of 8-48 kHz, and an oversampling rate of 64-512. The Stereo Audio DAC consists of a digital input interface and an analog DAC, and features a software power-down function for lower power consumption.

The audio data format of the digital interface is a 32 bit word - the upper (lower) 16 bits are used as the data for the left (right) 24 bit channel of the DAC in a left justified manner.

2. Configuration and Block Diagram

This section contains the functional block diagram of the Stereo Audio DAC.

Figure 23-1 Block Diagram



3. Operation

This section describes how to use the Stereo Audio DAC

The Stereo Audio DAC is configured to be used at a sampling frequency of 8-48 KHz, an oversampling rate of 64-512, and an operating frequency of 256fs or 512fs.

3.1. Initialization

Reset operation : reset the Stereo Audio DAC after powering up

DMA Interface Unused:

The sequence for powering up the analog DAC is as follows.

1) Configure the OSR and DACCLK bit of the DAOSR register.
2) Write a "1" into the INIT bit and DAE bit of the DACR register. This will release the power down state of the analog DAC and allow it to power up.
3) Wait for at least 300 sampling cycles (1/fs) before proceeding to the next step (step 4)
4) Write a "0" into the INIT bit of the DACR register. This will enable the output data pins of the digital interface thus providing data to the analog DAC.
5) Configure the PCL/PCR bit of the DPCR register.
6) Configure the FEST bit of the DACTRL register.
7) Writing the '7' into the INTREN register if you want to use DATA_REQ_IRQ, UDRN_IRQ, the OVFL_IRQ. As a result, DATA_REQ_IRQ is asserted.

DMA Interface Used:

The sequence for powering up the analog DAC is as follows.

1) Configure the OSR and DACCLK bit of the DAOSR register.
2) Write a "1" into the INIT bit and DAE bit of the DACR register. This will release the power down state of the analog DAC and allow it to power up.
3) Wait for at least 300 sampling cycles (1/fs) before proceeding to the next step (step 4)
4) Write a "0" into the INIT bit of the DACR register. This will enable the output data pins of the digital interface thus providing data to the analog DAC.
5) Configure the PCL/PCR bit of the DPCR register.
6) Configure the FEST bit of the DACTRL register. Write a "1" into the DMAEN bit of the DACTRL register. As a result, DAC_DMA_REQ is asserted.
7) Writing the 'F' into the INTREN register if you want to use DMA_ERR_IRQ, DATA_REQ_IRQ, UDRN_IRQ, OVFL_IRQ. As a result, DATA_REQ_IRQ is asserted.

Note;

- *The write access to DAOSR/DACR register is effective at DABUSY_DABUSY=0. Write access to DAOSR / DACR register is discarded in the case of DABUSY_DABUSY=1.*
- *The data output pins of the analog DAC may take up to 715 ms to stabilize after the DAE bit has been set*
- *The Stereo Audio DAC will generate a pop noise when a transition from high-to-low or low-to-high occurs on the DAE bit of the DACR register during power up or power down. Add external circuitry to suppress such pop noises if necessary.*
- *Do not change the DPCR register dynamically.*

3.2. Using DMA Transfers to Update DADR Register

DMA mode must be enabled by setting the DACTRL_DMAEN=1.

The Stereo Audio DAC module will then assert the DAC_DMA_REQ output when the number of free entries in the FIFO buffer exceeds the value given by FEST. DAC_DMA_REQ stays active until the DMA acknowledges the request by asserting DAC_DMA_ACK.

There is an internal down counter for this purpose which is reloaded with the value of FEST+1 when DAC_DMA_REQ is asserted. It is decremented with each word written to the FIFO buffer. When the counter reaches zero the transfer is complete and another DMA transfer may be requested. It is an error if the DMA writes data to the FIFO while the down counter is zero.

Note:

- *The block size configured in the DMA must match DACTRL_FEST+1*

3.3. Output Data Values of the Analog DAC

The data output pins of the analog DAC will be set to 0 V when disabled.

When enabled, with a voltage reference at 0.5 × AVCC(DAC), these pins will generate an output voltage with peak-to-peak values of 0.609 × AVCC(DAC). The table below shows the theoretical output voltage values under typical conditions.

Table 23-1 Output Data Values of Analog DAC

DADR[31:16] or DADR[15:0] MSB LSB	DAC Left (or Right) data output pin voltage
0000H	(0.5 + 0.3365 × (0/32768)) × AVCC(DAC)
0001H	(0.5 + 0.3365 × (1/32768)) × AVCC(DAC)
0002H	(0.5 + 0.3365 × (2/32768)) × AVCC(DAC)
…	
7FFEH	(0.5 + 0.3365 × (32766/32768)) × AVCC(DAC)
7FFFH	(0.5 + 0.3365 × (32767/32768)) × AVCC(DAC)
8000H	(0.5 + 0.3365 × (-32768/32768)) × AVCC(DAC)
8001H	(0.5 + 0.3365 × (-32767/32768)) × AVCC(DAC)
…	
FFFDH	(0.5 + 0.3365 × (-3/32768)) × AVCC(DAC)
FFFEH	(0.5 + 0.3365 × (-2/32768)) × AVCC(DAC)
FFFFH	(0.5 + 0.3365 × (-1/32768)) × AVCC(DAC)

Note:

- *The Stereo Audio DAC does not feature integrated output voltage buffers.*

3.4. Switching Clock Frequencies of the Analog DAC

The sequence for changing the clock frequency of the analog DAC is as follows.

1) Write a "1" into the INIT bit of the DACR register. This will cause the data output pins of the digital interface to output a "00000000h" to the analog DAC.

2) Wait for at least 300 sampling cycles (1/fs) before proceeding to the next step (step 3)

3) Configure the DACCLK and OSR bits of the DAOSR register. The value of the INIT bit must be "1" throughout this procedure.

4) Wait for at least 300 sampling cycles (1/fs) before proceeding to the next step (step 5)

5) Write a "0" into the INIT bit of the DACR register. This will enable the output data pins of the digital interface thus providing data to the analog DAC.

3.5. Minimum Writing Interval Between the DAOSR and DACR Registers

The write access to the DAOSR/DACR register mounts the asynchronous circuit from CLKPI to CLKDA. DABUSY field indicates that the write access to DAOSR/DACR register has not been completed. The write access to the DAOSR/DACR register is effective only at DABUSY=0. Write access to DAOSR/DACR register is discarded in the case of DABUSY=1.

This rule applies to the following cases:

- ・Write into the DOSR register → Write into the DAOSR register
- ・Write into the DOSR register → Write into the DACR register
- ・Write into the DCR register → Write into the DOSR register
- ・Write into the DCR register → Write into the DCR register

3.6. Interrupt

The Stereo Audio DAC module interrupts are controlled by three registers: the Interrupt Enable Register (INTREN), the Interrupt Status Register (INTRSTAT) and the Interrupt Clear Register (INTRCLR). After reset all interrupts are disabled. If an interrupt is to be used, it must be first enabled. The current status of an interrupt may be checked at any time in the Interrupt Status Register. The Interrupt Status Register contents are independent of the enable status of the interrupts. I.e. the Interrupt Status Bits are not masked by the Interrupt Enable Register.

If an interrupt has occurred, it can be reset by the Interrupt Clear Register. Writing a logic 1 to a bit in the Interrupt Clear Register clears the corresponding interrupt line as well as the interrupt status bit.

1. Data Request Interrupt

   This interrupt indicates there is at least space for another FEST + 1 data samples in the FIFO buffer. I.e. the Stereo Audio DAC module asserts the interrupt request when a read from the FIFO buffer frees up another samples slot, so that there are FEST + 1free entries in total.

2. FIFO Buffer Overflow Error Interrupt

   This interrupt indicates the CPU has tried to write another data sample to the FIFO buffer when it was already completely filled.

3. FIFO Buffer Under-Run Error Interrupt

   This interrupt indicates an under-run of the FIFO buffer for data samples. I.e. the analog DAC has tried to read a data sample from the FIFO buffer when it was empty.

4. DMA Block Error Interrupt

   This interrupt indicates an error case when the DMA tries to transfer more data to the FIFO buffer than configured by DACTRL:FEST + 1.

3.7. AHB Slave Interface

AHB slave interface controls the read/write access to the FIFO and register access. Table 23-2 shows the access type of the corresponding AHB.

Table 23-2 AHB Slave Interface

Item	support
Burst transfer	Stereo Audio DAC supports the burst transfer. However, the burst transfer is processed as a single transfer.
Protection control	not support
response	OKAY and ERROR only
Access size	8,16,32 bit only
Width of address	10 bit

The following accesses cause the slave error response.

1) When the access destination is an address that does not exist in the memory map.

2) The write access was done to the read only register.

3) When the access size is a size not supported. (For example, it accessed DADR with 8 bit or 16 bit.)

3.8. FIFO

FIFO for data is built into the Stereo Audio DAC module. Table 23-3 shows the specifications of the FIFO. It accesses FIFO through the DADR0-15 register. It becomes possible to access FIFO by accessing either of register of DADR0-15. The access to FIFO is not related to the address of DADR. The access to FIFO is done in order of the access regardless of the address of DADR0-15.

Table 23-3 Outline of FIFO

Item	Content
Width of data	32 bit
Width of address	5 bit
Depth	24
Composition	Register

3.9. Frequency of CLKPI and CLKDA

Please set the clock that meets the following requirements to CLKPI (bus clock) and sampling clock CLKDA (CLKDACI is divides)

CLKPI ≧ CLKDA / 17

3.10. Sampling Frequency and Oversampling Setting of DA Converter

The input of analogDAC is decided by setting DAOSR_OSR and DAOSR_DACCLK. Table 23-4 and Table 23-5 show the example of the set value to DAOSR_OSR and DAOSR_DACCLK.

Table 23-4 shows System Clock and over sampling ratio.

Table 23-5 shows Setting of dividing frequency.

Table 23-4 System Clock and Over Sampling Ratio

Sampling frequency (fs)[kHz]	pin setting OSR[1:0]	OSR	system clock frequency
8, 11.025, 12	11	512	512fs
16, 22.05, 24	10	256	256fs
32, 44.1, 48	01	128	256fs

Note:

- *384fs is not supported.*
- *OSR[1:0]=00 is not supported.*
- *OSR[1:0] is OSR field of the DAOSR register.*

Table 23-5 Setting of Dividing Frequency

Fs	256fs				512fs			
	CLKDAO	DACCLK	CLKDACI		CLKDAO	DACCLK	CLKDACI	
8 kHz	2.048 MHz	1/1	2.048	MHz	4.096 MHz	1/1	4.096	MHz
		1/2	4.096	MHz		1/2	8.192	MHz
		1/8	16.384	MHz		1/8	32.768	MHz
		1/10	20.48	MHz		1/10	40.96	MHz
		1/16	32.768	MHz		1/16	65.536	MHz
11.025 kHz	2.8224 MHz	1/1	2.8224	MHz	5.6448 MHz	1/1	5.6448	MHz
		1/2	5.6448	MHz		1/2	11.2896	MHz
		1/8	22.5792	MHz		1/8	45.1584	MHz
		1/10	28.224	MHz		1/10	56.448	MHz
		1/16	45.1584	MHz		1/16	90.3168	MHz
12 kHz	3.072 MHz	1/1	3.072	MHz	6.144 MHz	1/1	6.144	MHz
		1/2	6.144	MHz		1/2	12.288	MHz
		1/8	24.576	MHz		1/8	49.152	MHz
		1/10	30.72	MHz		1/10	61.44	MHz
		1/16	49.152	MHz		1/16	98.304	MHz
12.8 kHz	3.2768 MHz	1/1	3.2768	MHz	6.5536 MHz	1/1	6.5536	MHz
		1/2	6.5536	MHz		1/2	13.1072	MHz
		1/8	26.2144	MHz		1/8	52.4288	MHz
		1/10	32.768	MHz		1/10	65.536	MHz
		1/16	52.4288	MHz		1/16	104.8576	MHz
16 kHz	4.096 MHz	1/1	4.096	MHz	8.192 MHz	1/1	8.192	MHz
		1/2	8.192	MHz		1/2	16.384	MHz
		1/8	32.768	MHz		1/8	65.536	MHz
		1/10	40.96	MHz		1/10	81.92	MHz
		1/16	65.536	MHz		1/16	131.072	MHz
22.05 kHz	5.6448 MHz	1/1	5.6448	MHz	11.2896 MHz	1/1	11.2896	MHz
		1/2	11.2896	MHz		1/2	22.5792	MHz
		1/8	45.1584	MHz		1/8	90.3168	MHz
		1/10	56.448	MHz		1/10	112.896	MHz
		1/16	90.3168	MHz		1/16	180.6336	MHz

Fs	256fs				512fs			
	CLKDAO	DACCLK	CLKDACI		CLKDAO	DACCLK	CLKDACI	
24 kHz	6.144 MHz	1/1	6.144	MHz	12.288 MHz	1/1	12.288	MHz
		1/2	12.288	MHz		1/2	24.576	MHz
		1/8	49.152	MHz		1/8	98.304	MHz
		1/10	61.44	MHz		1/10	122.88	MHz
		1/16	98.304	MHz		1/16	196.608	MHz
32 kHz	8.192 MHz	1/1	8.192	MHz	16.384 MHz	1/1	16.384	MHz
		1/2	16.384	MHz		1/2	32.768	MHz
		1/8	65.536	MHz		1/8	131.072	MHz
		1/10	81.92	MHz		1/10	163.84	MHz
		1/16	131.072	MHz		1/16	262.144	MHz
44.1 kHz	11.2896 MHz	1/1	11.2896	MHz	22.5792 MHz	1/1	22.5792	MHz
		1/2	22.5792	MHz		1/2	45.1584	MHz
		1/8	90.3168	MHz		1/8	180.6336	MHz
		1/10	112.896	MHz		1/10	225.792	MHz
		1/16	180.6336	MHz		1/16	361.2672	MHz
48 kHz	12.288 MHz	1/1	12.288	MHz	24.576 MHz	1/1	24.576	MHz
		1/2	24.576	MHz		1/2	49.152	MHz
		1/8	98.304	MHz		1/8	196.608	MHz
		1/10	122.88	MHz		1/10	245.76	MHz
		1/16	196.608	MHz		1/16	393.216	MHz

Note:
- *DACCLK is DACCLK field of the DAOSR register.*
- *The exact value cannot be configured. But the deviation or mismatch can be minimized.*

3.11. Output Stop Control of Analog DAC

The procedure for stopping the output data of analog DAC is below.

1) Write a "1" into the INIT bit of the DACR register.
2) FIFO is reset by writing "1" in FLUSH bit of the DAFLUSH register.
   FIFO is reset by writing "1" in FLUSH bit, and the DAC_DATA_REQ_IRQ and DAC_DMA_REQ interruption are issued.
3) Write a "0" into the INIT bit of the DACR register. As a result, the output of analog DAC is restarted.
   When FIFO FLUSH is not used, it restarts by using the data stored in FIFO. However, the first data that was stored in the FIFO are discarded.

3.12. Bridge-Tied Load (BTL)

The Stereo Audio DAC mounts the method of connection by Bridge-tied load(BTL). When BTL is used, output L and R of Audio DAC become the signal outputs with the phase difference of 180° .

It is necessary to reverse the polarity of the upper 16 bits (LDATA) by the DPCR register setting when BTL is applied or to reverse the polarity of the lower 16 bits (RDATA). Concretely, the phase of the input signal is reversed by writing 01b or 10b in the DPCR register.

Table 23-6 AHB DATA (Input DATA)Format

31 16	15 0
The upper 16 bits (LDATA)	The lower 16 bits (RDATA)
Left-Channel	Right-Channel

Note:

- *BTL assumes the case when input data 32 bit to Audio DAC is monaural (31:16 and 15:0 are equal). The effect that monaural BTL expects is not achieved though it operates even when 32 bit stereo data of L ≠R is input.*
- *Do not set the value of the DPCR register to 00b or 11b.*
  Because the amplitude is counterbalanced when connecting it with one speaker

4. Registers

The Stereo Audio DAC has the following set of registers, which are used to program the modes of operation

4.1. DAC Over Sampling Register(DAOSR)

This DAOSR Register is used to configure the oversampling clock frequency of the Analog DAC.

REGISTER_NAME	DAC Over Sampling Register(DAOSR)
OFFSET	0x000
ACCESS_SIZE	B H W
MULTIPLE	0
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,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	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,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	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,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	Reserved	DACCLK2	DACCLK1	DACCLK0	Reserved	Reserved	OSR1	OSR0
ACCESS_TYPE	R0,W0	R/W	R/W	R/W	R0,W0	R0,W0	R/W	R/W
PROT_TYPE	Wp							
INITIAL_VALUE	0	0	0	0	0	0	1	0

[bit31:7] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit6:4] DACCLK[2:0]:Analog DAC clock setting

Explanation of DACCLK[2:0]

bit[2:0]	Description
000	Divide by 8: Conversion clock = CLKDACI/8
001	Divide by 10: Conversion clock = CLKDACI/10
010	Divide by 16: Conversion clock = CLKDACI/16
011	Divide by 2: Conversion clock = CLKDACI/2
100	Divide by 1: Conversion clock = CLKDACI
other	Not allowed. (Conversion clock = CLKDACI/10)

[bit3:2] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit1:0] OSR[1:0]:Over sampling

Explanation of OSR[1:0]

bit[1:0]	Description
00	Over sampling ratio : 64
01	Over sampling ratio : 128
10	Over sampling ratio : 256
11	Over sampling ratio : 512

4.2. DAC Configuration (DACR)

This DACR Register is used to configure the power down and send data of the Analog DAC.

REGISTER_NAME	DAC Configuration (DACR)
OFFSET	0x004
ACCESS_SIZE	B H W
MULTIPLE	0
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,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	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,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	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	INIT
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	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	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	DAE
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R/W
PROT_TYPE	Wp							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:9] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit8] INIT: Initialize

Explanation of INIT

bit	Description
0	The digital interface will output the DADR register value to the analog DAC (default)
1	The digital interface logic will be initialized and output a 00000000h value to the analog DAC

[bit7:1] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit0] DAE: DAC enable

Explanation of DAC enable

bit	Description
0	Disable operation. The analog DAC output pins will output 0.0 V. (default)
1	Normal operation. The analog DAC output pins will be enabled.

4.3. DAC BUSY Register(DABUSY)

It is displayed not to be able to do the write access in the DAOSR/DACR register.

REGISTER_NAME	DAC BUSY Register(DABUSY)
OFFSET	0x008
ACCESS_SIZE	B H W
MULTIPLE	0
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,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	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,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	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,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	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	DABUSY
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R
PROT_TYPE	Wp							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:1] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit0] DABUSY: DAOSR/DACR BUSY

Explanation of DAOSR/DACR

bit	Description
0	The write access to the DAOSR/DACR register is possible.
1	The write access to the DAOSR/DACR register is impossible.

Note:

- *The write access to the DAOSR/DACR register uses the asynchronous circuit from CLKPI to CLKDA. DABUSY field indicates that the write access to DAOSR / DACR register has not been completed. Write access to DAOSR / DACR register is discarded in the case of DABUSY=1.*

4.4. DAC Control Register(DACTRL)

This DACTRL register performs a FIFO threshold setting and DMA setting.

REGISTER_NAME	DAC Control Register(DACTRL)
OFFSET	0x00C
ACCESS_SIZE	B H W
MULTIPLE	0
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,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	Reserved	Reserved	Reserved	FEST4	FEST3	FEST2	FEST1	FEST0
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	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	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	DMAEN
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	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	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0
PROT_TYPE	Wp							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:21] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit20:16] FEST[4:0]:FIFO Empty threshold

Explanation of FEST[4:0]

bit[4:0]	Description
00000 to 111111	This field defines the number of empty entries in the FIFO buffer, which triggers a DMA request and/or a data request interrupt when it is exceeded..

Note:

- *FIFO buffer is 32 bit * 24word. If this field is set to a value from 0 to 23, DAC_DMA_REQ and DAC_DATA_REQ_IRQ is generated when the number of empty FIFO buffer entries exceeds this value. If this field is set to a value greater than 23, DAC_DMA_REQ and DAC_DATA_REQ_IRQ are never generated.*
- *These bits must not be changed during the DMA transfers.*

[bit15:9] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit8] DMAEN: DMA enable

Explanation of DMA enable

bit	Description
0	The DMA interface is disabled (default)
1	The DMA interface is enabled

Note:

This bit must not be changed during the DMA transfers.

[bit7:0] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

4.5. DAC FLUSH Register(DAFLUSH)

This DAFLUSH Register is the initialization of DATA FIFO.

REGISTER_NAME	DAC FLUSH Register(DAFLUSH)
OFFSET	0x010
ACCESS_SIZE	B H W
MULTIPLE	0
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,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	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,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	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,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	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	FLUSH
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W1
PROT_TYPE	Wp							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:1] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit0] FLUSH: FIFO FLUSH

Explanation of FIFO FLUSH

bit	Description
0	No change
1	Enable FIFO FLUSH Mode

Note:

- *The write access to this register is effective at DACR_INIT=1*

4.6. Interrupt Enable Register(INTREN)

This INTREN Register is used to enable or disable individual interrupts of the Stereo Audio DAC module.

REGISTER_NAME	Interrupt Enable Register(INTREN)
OFFSET	0x014
ACCESS_SIZE	B H W
MULTIPLE	0
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,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	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,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	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,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	Reserved	Reserved	Reserved	Reserved	DMA_ERR	UDRN	OVFL	DREQ
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R/W	R/W	R/W	R/W
PROT_TYPE	Wp							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:4] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit3] DMA_ERR: DMA Block Error

Explanation of DMA_ERR

bit	Description
0	The DMA block error interrupt is disabled. (default)
1	The DMA block error interrupt is enabled

[bit2] UDRN:FIFO Under-Run Error

Explanation of UDRN

bit	Description
0	The FIFO under-run error interrupt is disabled (default)
1	The FIFO under-run error interrupt is enabled

[bit1] OVFL: FIFO Overflow Error

Explanation of OVFL

bit	Description
0	The FIFO overflow error interrupt is disabled (default)
1	The FIFO overflow error interrupt is enabled

[bit0] DREQ: Date Request

Explanation of DREQ

bit	Description
0	The FIFO data request interrupt is disabled (default)
1	The FIFO data request interrupt is enabled

4.7. Interrupt State Register (INTRSTAT)

This INTRSTAT Register reflects the status of the individual interrupt sources of the BSADACV module. The bits in this registers are ‘sticky’, i.e. once they have been set from the hardware, they remain set until they are reset via a write access to the INTRCLR register.

REGISTER_NAME	Interrupt State Register(INTRSTAT)
OFFSET	0x018
ACCESS_SIZE	B H W
MULTIPLE	0
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,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	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,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	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,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	Reserved	Reserved	Reserved	Reserved	DMA_ERR	UDRN	OVFL	DREQ
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R	R	R	R
PROT_TYPE	Wp							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:4] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit3] DMA_ERR: DMA Block Error

Explanation of DMA_ERR

bit	Description
0	No error
1	A DMA block error has occurred, i.e. there were more data transfers to the FIFO buffer than specified by CONTROL:FEST + 1

[bit2] UDRN:FIFO Under-Run Error

Explanation of UDRN

bit	Description
0	No error
1	A FIFO buffer under-run has occurred, i.e. the digital DAC block tried to read another data sample, but the FIFO buffer is empty.

[bit1] OVFL: FIFO Overflow Error

Explanation of OVFL

bit	Description
0	No error
1	A FIFO buffer overflow has occurred, i.e. there was a write access to the FIFO buffer when there was no more space available

[bit0] DREQ: Date Request

Explanation of DREQ

bit	Description
0	No data request: There are less than CONTROL:FEST + 1 empty entries available in the FIFO buffer
1	Data request: there are CONTROL:FEST+1 or more empty entries available in the FIFO buffer

Note:

- *DREQ bit is set to "0" by a reset input. However, DREQ bit is set to "1" by releasing the reset. Because FIFO is empty after the reset is released*

4.8. Interrupt Clear Register (INTCTR)

This INTRCLR Register is used to clear individual interrupts of the BSADACV module.

REGISTER_NAME	Interrupt Clear Register(INTCTR)
OFFSET	0x01C
ACCESS_SIZE	B H W
MULTIPLE	0
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,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	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,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	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,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	Reserved	Reserved	Reserved	Reserved	DMA_ERR	UDRN	OVFL	DREQ
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W	R0,W	R0,W	R0,W
PROT_TYPE	Wp							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:4] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit3] DMA_ERR: DMA Block Error

Explanation of DMA_ERR

bit	Description
0	Leave the DMA block error interrupt unchanged
1	Writing a ‘1’ to this bit clears the DMA block error interrupt. This bit is always read as ‘0’.

[bit2] UDRN:FIFO Under-Run Error

Explanation of UDRN

bit	Description
0	Leave the FIFO under-run error interrupt unchanged
1	Writing a ‘1’ to this bit clears the FIFO under-run error interrupt. This bit is always read as ‘0’.

[bit1] OVFL: FIFO Overflow Error

Explanation of OVFL

bit	Description
0	Leave the FIFO overflow error interrupt unchanged
1	Writing a ‘1’ to this bit clears the FIFO overflow error interrupt. This bit is always read as ‘0’.

[bit0] DREQ: Date Request

Explanation of DREQ

bit	Description
0	Leave the data request interrupt unchanged
1	Writing a ‘1’ to this bit clears the data request interrupt. This bit is always read as ‘0’.

Note:

- *DREQ bit is set to "0" by a reset input. However, DREQ bit is set to "1" by releasing the reset. Because FIFO is empty after the reset is released*

4.9. DAC Polarity Configuration Register (DPCR)

This DPCR Register is used to configuration polarity conversion of LDATA and RDATA.

REGISTER_NAME	DAC Polarity Configuration Register (DPCR)
OFFSET	0x020
ACCESS_SIZE	B H W
MULTIPLE	0
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,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	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,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	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,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	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	PCL	PCR
ACCESS_TYPE	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R0,W0	R/W	R/W
PROT_TYPE	Wp							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:2] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit1] PCL: The upper 16 bits (Left-DATA) Polarity conversion

Explanation of PCL

bit	Description
0	No influence
1	LDATA Polarity conversion

[bit0] PCR: The lower 16 bits (Right-DATA) polarity conversion

Explanation of PCR

bit	Description
0	Leave the FIFO under-run error interrupt unchanged
1	RDATA Polarity conversion

4.10. Data 0..15 Register (DADR)

The DADR register (DAC Data Register) is the digital audio data interface to the analog DAC.

REGISTER_NAME	Data 0..15 Register (DADRi)
OFFSET	0x040+i*4
ACCESS_SIZE	W
MULTIPLE	0:15
NUMERIC_TYPE	
OTHER	

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	DADR31	DADR30	DADR29	DADR28	DADR27	DADR26	DADR25	DADR24
ACCESS_TYPE	RX/W	RX/W	RX/W	RX/W	RX/W	RX/W	RX/W	RX/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	DADR23	DADR22	DADR21	DADR20	DADR19	DADR18	DADR17	DADR16
ACCESS_TYPE	RX/W	RX/W	RX/W	RX/W	RX/W	RX/W	RX/W	RX/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	DADR15	DADR14	DADR13	DADR12	DADR11	DADR10	DADR9	DADR8
ACCESS_TYPE	RX/W	RX/W	RX/W	RX/W	RX/W	RX/W	RX/W	RX/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	DADR7	DADR6	DADR5	DADR4	DADR3	DADR2	DADR1	DADR0
ACCESS_TYPE	RX/W	RX/W	RX/W	RX/W	RX/W	RX/W	RX/W	RX/W
PROT_TYPE	Wp							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:0] DADR: DAC Data Register

Explanation of DADR

bit	Description
00000000 to 11111111	The DADR register is the digital audio data input to the analog DAC. The digital audio data format must be in 32 bit words. The upper 16 bits will be used as the input to the 24 bit left channel of the analog DAC, left justified. The lower 16 bits will be used as the input to the 24 bit right channel of the analog DAC, left justified. All data must be in binary twos complement format.

Note:

- *The DADR register accepts 32 bit word writes only.*
- *Please set the same value as DADR[31:16] and DADR[15:0] when outputting it by monaural.*

CHAPTER 24: Inter IC Sound (I2S)

This chapter explains the functions and operations of the serial audio interface that is the Inter IC Sound (I2S)..

1. Overview
2. Configuration and Block Diagram
3. Operations of the I2S
4. Registers

CODE: I2S-S6J3300-E1

1. Overview

This section describes the features and the block diagram of the I2S module.

Also refer to the chapter of "Sound System Configuration" on this manual how to configure the I2S which is connected to the Sound Mixer.

I2S module is a full duplex, synchronous serial audio interface for multichannel specification. It can be configured to various frame formats by register setting.

This module can be set to operate as master and slave. In the master mode, clock (SCK) and frame synchronous signal (WS) are output to the external slave. In the slave mode, they are input from the external master.

During the master mode, SCK clock can be output by dividing external clock or internal bus clock (it is selectable by register). Frame synchronous signal can be generated by free-running or burst mode (generated only when there is transmission data).

This module has transmission/reception FIFOs, and their depths depend on the mode:

In transmission only mode, there is a 132-word 32-bit transmission FIFO. In reception only mode, there is a 132-word 32-bit reception FIFO. This module can also be configured in simultaneous mode. Simultaneous mode operates with a 66-word 32-bit transmission FIFO and a 66-word 32-bit reception FIFO.

Internal transfer between transmission and reception FIFO and internal system memory can be performed by DMA, interrupt, and polling.

Features of I2S

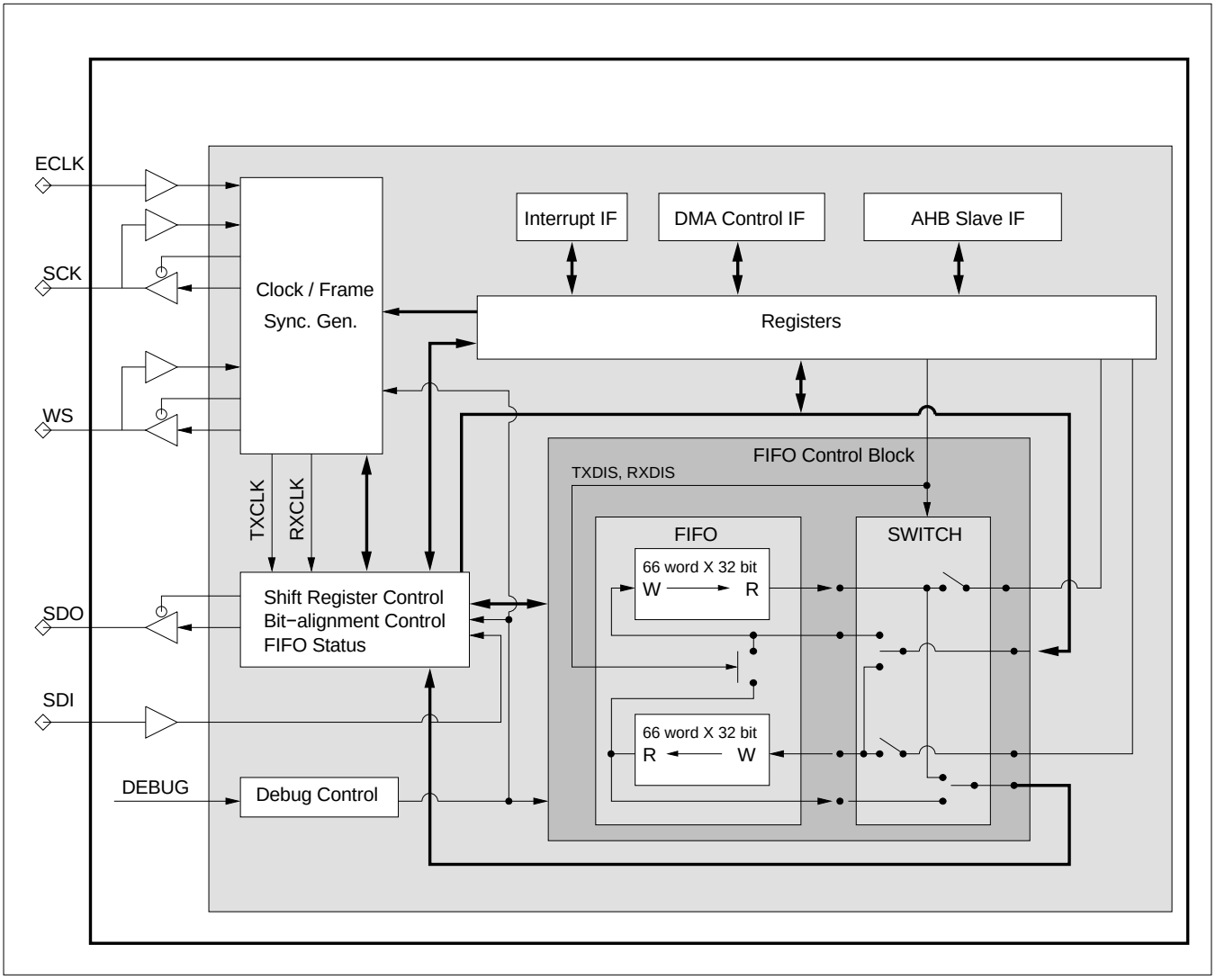
I2S interface has the following features:

- Programmable master/slave operations
- Support of transmission only, reception only and simultaneous transmission/reception modes
- Selecting 1 sub frame and 2 sub frame constructions
- Setting up to 32 channels to each sub frame
- Individually setting number of channel in each sub frame
- Individually setting channel length of each sub frame (number channel bit)
- Individually setting word length in channel of each sub frame
- Setting valid/invalid of each channel in each sub frame. Data is not sent or received to invalid channel
- Setting word length from 7 to 32 bits
- Programming frequency of frame synchronous signal
- Setting up to 3072 bits in 1 frame
- Programming width of frame synchronous signal (1 bit or 1 channel length)
- Programming phase of frame synchronous signal (0-bit or 1-bit delay)
- Setting polarity of frame synchronous signal
- Setting polarity of serial bit clock
- Programming sampling point of received data (center or at the end of received data)
- Selecting clock frequency source of serial bit clock in the master mode (internal bus clock and external clock)
- Setting clock frequency dividing ratio in the master mode
- Frequency of SCK = (frequency of internal bus clock or external clock) / (2 x I2Sn_CNTREG:CKRT[5:0])
- Frequency dividing ratio is settable within 0-126 in multiple of 2 (when the ratio is '0', frequency dividing source is bypassed)
- Data transfer to system memory by DMA, interrupt, and polling
- Debug support

2. Configuration and Block Diagram

This section describes the block diagram of I2S.

Figure 24-1 Block Diagram of I2S



- Four wire interface is used for full duplex data transfer (separate line for serial data input and serial data output)
- Frame and clock lines are bi-directional. These lines are output when module is configured in master mode and act as input line in slave mode
- DMA controller is used for DMA access

Clocking of I2S

- The supply clock of I2S can be internal bus clock or external (ECLK) clock source. This clock is then pre-scaled to required frequency through I2S control register bits I2Sn_CNTREG:CKRT[5:0]
- Frame frequency can be adjusted using I2Sn_CNTREG:OVHD[9:0] bits

3. Operations of the I2S

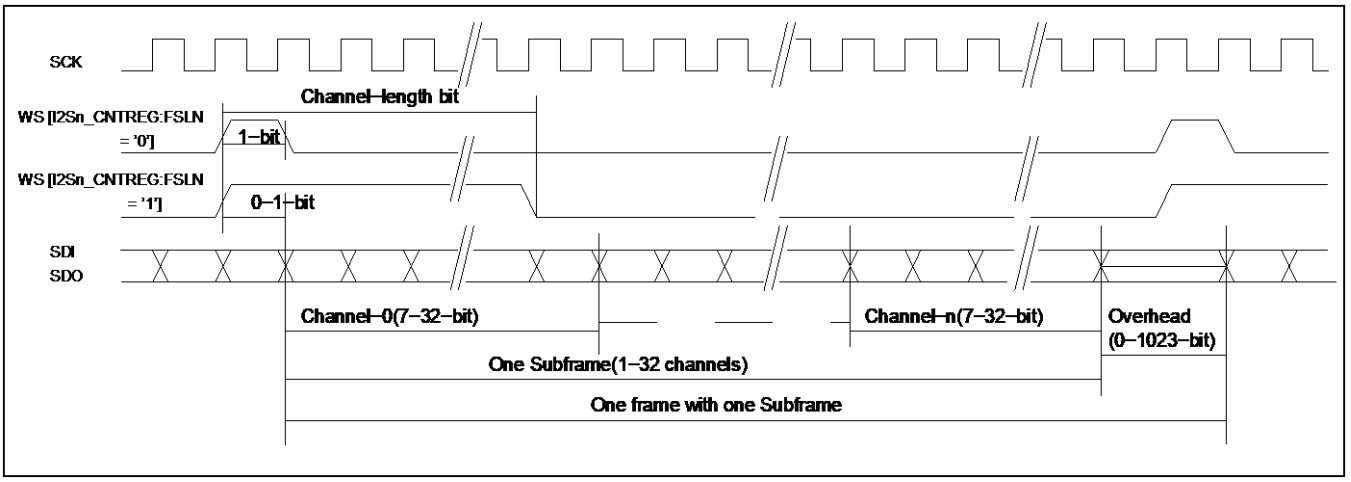
This section describes the operation of I2S.

3.1. I2S Frame Construction

I2S supports frame format of multiple channel construction. Frame can be configured to 1 or 2 sub frames. Number of each frame's channel and word length can be set individually.

1 Sub Frame Construction

Figure 24-2 1 Sub Frame Construction

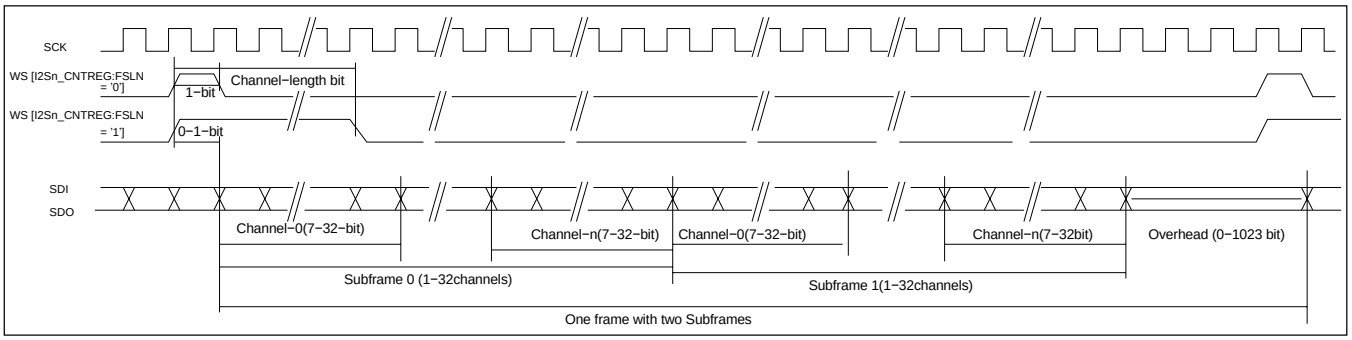


Description

- When I2Sn_CNTREG:SBFN bit is '0', frame becomes 1 sub frame composite
- Number of channels of 1-sub frame is determined by I2Sn_MCR0REG:S0CHN
- Up to 32 channels can be set
- Each channel bit length (word length) is determined by I2Sn_MCR0REG:S0WDL
- Sub frame channel starts from 0th. Each channel is set to valid/invalid with the corresponding bit of I2Sn_MCR1REG register. Transmission/reception of data is not performed to invalid channel
- Dummy bit can be inserted behind sub frame by setting I2Sn_CNTREG:OVHD. 0 to 1023 bits are insertable
- Polarity of WS is set with I2Sn_CNTREG:FSPL
- Pulse width of WS can be set to 1 bit or 1 channel length by setting I2Sn_CNTREG:FSLN
- Frame sync phase of WS can be set to '0' or '1' clock through I2Sn_CNTREG:FSPH
- In this construction, settings of I2Sn_MCR0REG:S1CHN and I2Sn_MCR2REG:S1WDL are ignored

2 Sub Frame Construction

Figure 24-3 2 Sub Frame Construction



Description

- When I2Sn_CNTREG:SBFN bit is '1', frame becomes 2 sub frame composition
- Set number of channel of sub frame 0 to I2Sn_MCR0REG:S0CHN, and set number of sub frame 1 channel to I2Sn_MCR0REG:S1CHN. Up to 32 channels can be set
- Channel bit length (word length) of sub frame 0 is determined by I2Sn_MCR0REG:S0WDL. For sub frame 1, they are determined by I2Sn_MCR0REG:S1WDL
- Sub frame channel starts from 0th. Each channel of sub frame 0 is set to valid/invalid with the corresponding bit of I2Sn_MCR1REG register, and corresponding bit of I2Sn_MCR2REG register for sub frame 1
- Transmission/reception of data is not performed to invalid channel
- Dummy bit can be inserted behind sub frame 1 by setting I2Sn_CNTREG:OVHD. 0 to 1023 bits are insertable
- Polarity of WS is set to I2Sn_CNTREG:FSPL bit
- Pulse width of WS can be set to 1 bit or 1 channel length by setting I2Sn_CNTREG:FSLN bit, channel length setting of 1 channel is determined by the channel length of sub frame '0'
- Frame sync phase of WS can be set to '0' or '1' clock through I2Sn_CNTREG:FSPH bit

3.2. I2S Configuration and Operation Modes

Transmission Only Mode

Table 24-1 Transmission Only Mode

Transfer Setting	Operation	Master Mode (I2Sn_CNTREG:MSMD = '1')	Slave Mode (I2Sn_CNTREG:MSMD = '0')
Transmission only I2Sn_CNTREG: TXDIS = '0' I2Sn_CNTREG: RXDIS = '1'	Start	**Free-Running Mode (I2Sn_CNTREG:FRUN = '1'):** After I2Sn_OPRREG:START bit becomes ’1’ and I2Sn_OPRREG:TXENB bit is ’1’, frame synchronous signal starts to output when transmission FIFO is not empty. From the second time, frame synchronous signal is output with the frame rate determined by the register setting. If transmission FIFO is empty, empty frame is output at the same time of frame synchronous signal output. Serial data of the empty frame can be set to ’0’ or ’1’ by the register setting. **Burst Mode (I2Sn_CNTREG:FRUN = '0'):** If transmission FIFO is not empty, I2Sn_OPRREG:START bit is ’1’ and I2Sn_OPRREG:TXENB bit is ’1’, frame synchronous signal is output. After completion of one frame output, transmission FIFO status is always confirmed. If transmission FIFO is not empty, frame synchronous signal is output to perform frame transmission.	**Free-Running Mode (I2Sn_CNTREG:FRUN = '1'):** The frame synchronous signal is input at the frame rate determined by the register setting. If transmission FIFO is empty when the frame synchronous signal is input when I2Sn_OPRREG:START bit is ’1’ and I2Sn_OPRREG:TXENB bit is ’1’, empty frame is output. Serial data of the empty frame can be set to '0' or '1' by the register setting. **Burst Mode (I2Sn_CNTREG:FRUN = '0'):** When I2Sn_OPRREG:START bit is ’1’ and I2Sn_OPRREG:TXENB bit is ’1’, one frame is output every time the frame synchronous signal is input. When transmission FIFO is empty at the time of frame synchronous signal input, empty frame is output.
Transmission only I2Sn_CNTREG: TXDIS = '0' I2Sn_CNTREG: RXDIS = '1'	Stop	At the time of stop, transmission FIFO becomes empty when there’s no data transfer from internal memory to I2S transmission FIFO. **To Maintain I2Sn_OPRREG:START Bit to '1':** I2Sn_OPRREG:TXENB = ’1’: When ’1’ is written to I2Sn_OPRREG:TXENB, synchronous signal is output in the free-running mode. When transmission FIFO becomes empty, empty frame is output. In burst mode, frame synchronous signal is not output, and empty frame bits are output to serial data bus. I2Sn_OPRREG:TXENB = ’0’: When '0' is written to I2Sn_OPRREG:TXENB, transmission FIFO becomes empty. In the free-running mode, frame synchronous signal continues outputting and serial bus becomes high impedance state. In the burst mode, frame synchronous signal is not output and serial data bus becomes high impedance state. **To Make I2Sn_OPRREG:START Bit '0':** When '0' is written to I2Sn_OPRREG:START bit, then transmission FIFO becomes empty. Clock supply to the serial control part is stopped regardless of I2Sn_OPRREG:TXENB setting. Serial Output Clock and Frame synchronous signal output is stopped. Serial data bus becomes high impedance state.	**To Maintain I2Sn_OPRREG:START Bit to '1':** I2Sn_OPRREG:TXENB = ’1’: Empty frame data is output to serial bus. I2Sn_OPRREG:TXENB = ’0’: When ’0’ is written to I2Sn_OPRREG:TXENB, transmission FIFO becomes empty, and data present in the transmission FIFO at the time ’0’ was written to I2Sn_OPRREG:TXENB is not transmitted. Writing to transmission FIFO and detection of the frame synchronous signal are stopped. Serial data bus becomes high impedance state. **To Make I2Sn_OPRREG:START Bit '0':** When ’0’ is written to I2Sn_OPRREG:START bit, transmission FIFO becomes empty. Writing to transmission FIFO and detection of frame synchronous signal are stopped regardless of I2Sn_OPRREG:TXENB setting and serial bus becomes high impedance state.

Transfer Setting	Operation	Master Mode (I2Sn_CNTREG:MSMD = '1')	Slave Mode (I2Sn_CNTREG:MSMD = '0')
	Abnormality	When reading from transmission FIFO occurs while it is empty, empty frame is output. For the setting conditions of I2Sn_STATUS:TXUDR0 and I2Sn_STATUS:TXUDR1, refer to their bit descriptions. When writing to transmission FIFO occurs while it is full, set I2Sn_STATUS:TXOVR to ’1’.	When reading from transmission FIFO occurs while it is empty, empty frame is output. For the setting conditions of I2Sn_STATUS:TXUDR0 and I2Sn_STATUS:TXUDR1, refer to their bit descriptions. However I2Sn_STATUS:TXUDR0/1 are not set to ’1’ for the 1st output frame after the bits become I2Sn_OPRREG:START = ’1’ and I2Sn_OPRREG:TXENB = ’1’. When writing to transmission FIFO occurs while it is full, I2Sn_STATUS:TXOVR is set to ’1’. If the frame synchronous signal is not input with the defined frame rate in the free-running mode, I2Sn_STATUS:FERR is set to ’1’. If the next frame synchronous signal is input before completing 1 frame transmission in the burst mode, I2Sn_STATUS:FERR is set to ’1’.

Reception Only Mode

Table 24-2 Reception Only Mode

Transfer Setting	Operation	Master Mode (I2Sn_CNTREG:MSMD = '1')	Slave Mode (I2Sn_CNTREG:MSMD = '0')
Reception only I2Sn_CNTREG: TXDIS = '1' I2Sn_CNTREG: RXDIS = '0'	Start	**Free-Running Mode (I2Sn_CNTREG:FRUN = '1'):** Frame synchronous signal starts to output after I2Sn_OPRREG:START bit becomes '1' and I2Sn_OPRREG:RXENB bit is '1' when reception FIFO is not full. From the second time, frame synchronous signal with the frame rate determined by the register setting is output. **Burst Mode (I2Sn_CNTREG:FRUN = '0'):** When I2Sn_OPRREG:START bit is ’1’ and I2Sn_OPRREG:RXENB bit is ’1’, frame synchronous signal is output to receive frame if reception FIFO is not full. If the FIFO is full, the signal is not output.	**Free-Running Mode (I2Sn_CNTREG:FRUN = '1'):** When I2Sn_OPRREG:START bit is '1' and I2Sn_OPRREG:RXENB bit is '1', input frame synchronous signal with the frame rate determined by the register setting. Frame should be received every time the signal is input. **Burst Mode (I2Sn_CNTREG:FRUN = '0'):** When I2Sn_OPRREG:START bit is ’1’ and I2Sn_OPRREG:RXENB bit is ’1’, frame reception is performed every time frame synchronous signal is input. The signal is input with less speed than the frame rate in the free-running mode.
	Stop	At the time of stop, frame is not imported from serial bus even though reception FIFO is empty. **To Maintain I2Sn_OPRREG:START Bit to '1':** When ’0’ is written to I2Sn_OPRREG:RXENB, reception FIFO becomes empty. Although frame synchronous signal is kept outputting in the free-running mode, frame is not received. In the burst mode, frame is not received and the signal is not output. **To Make I2Sn_OPRREG:START Bit '0':** When ’0’ is written to I2Sn_OPRREG:START, reception FIFO becomes empty. Clock supply to the serial control part stops regardless of I2Sn_OPRREG:RXENB setting, and SCK supply to the external part is stopped as well.	**To Maintain I2Sn_OPRREG:START Bit to '1':** Reception FIFO becomes empty by writing '0' to I2Sn_OPRREG:RXENB. The input frame synchronous signal is ignored, and frames are not received. **To Make I2Sn_OPRREG:START Bit '0':** When ’0’ is written to the I2Sn_OPRREG:START bit, the rececption FIFO becomes empty. The input frame synchronous signal is ignored regardless of I2Sn_OPRREG:RXENB setting, and frames are not received.
	Abnormality	When writing to reception FIFO occurs while it is full, I2Sn_STATUS:RXOVR is set to ’1’. I2Sn_STATUS:RXUDR bit is set to ’1’ when read access to reception FIFO occurs while it is empty.	When writing to reception FIFO occurs while it is full, I2Sn_STATUS:RXOVR is set to ’1’. When read access to reception FIFO occurs while it is empty, I2Sn_STATUS:RXUDR is set to ’1’. **Free-Running Mode:** If frame synchronous signal is not input with the frame rate defined by the register setting, I2Sn_STATUS:FERR bit is set to ’1’.

Simultaneous Transfer Mode

Table 24-3 Simultaneous Transfer Mode

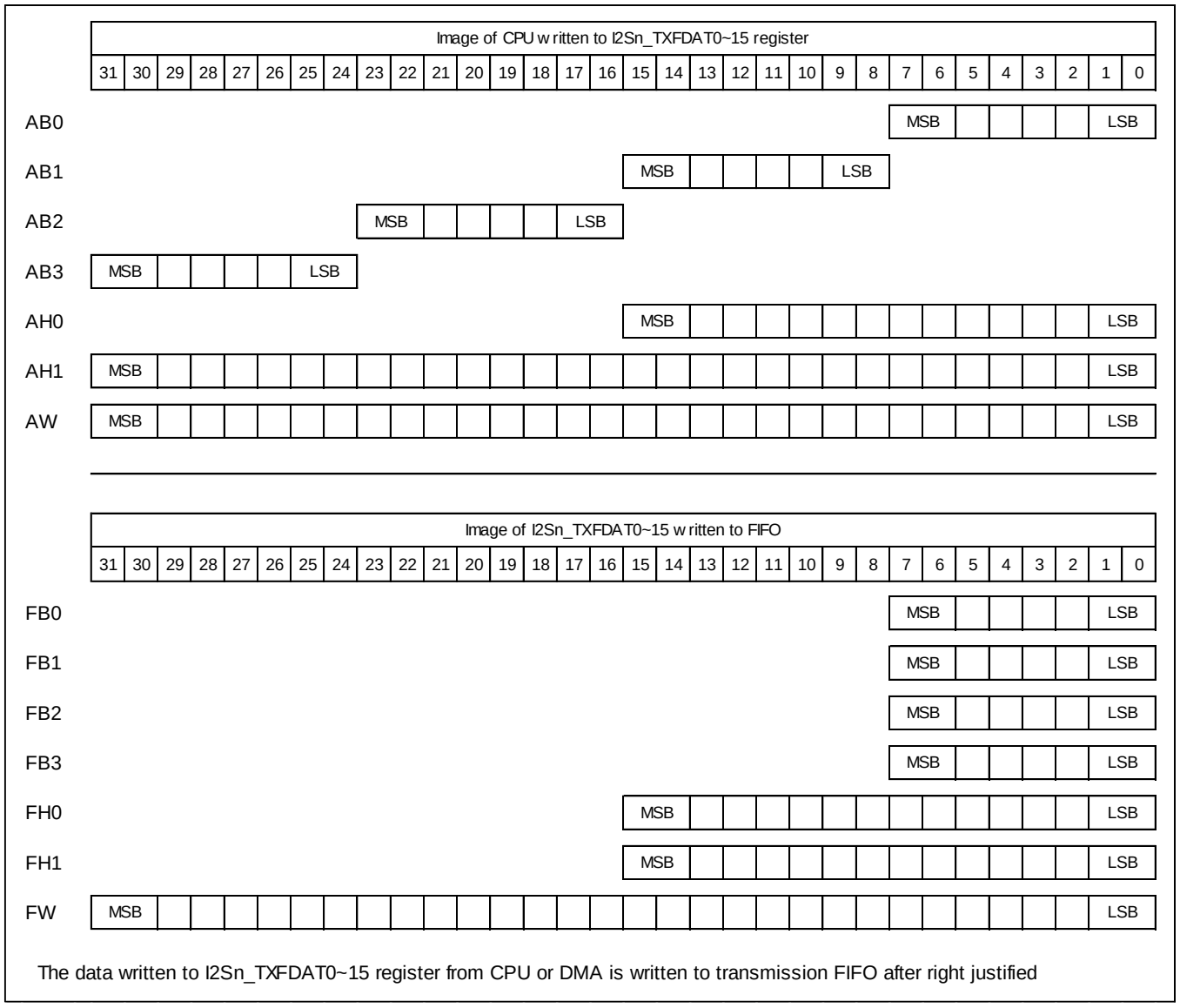
Transfer Setting	Operation	Master Mode (I2Sn_CNTREG:MSMD = '1')	Slave Mode (I2Sn_CNTREG:MSMD = '0')
Simultaneous Transfer I2Sn_CNTREG: TXDIS = ‘0’ I2Sn_CNTREG: RXDIS = ‘0’	Start	**Free-Running Mode (I2Sn_CNTREG:FRUN = '1'):** I2Sn_OPRREG:START = ’1’, I2Sn_OPRREG:TXENB = ’1’, and I2Sn_OPRREG:RXENB = ’0’: The same operation as transmission only mode. I2Sn_OPRREG:START = ’1’, I2Sn_OPRREG:TXENB = ’0’, and I2Sn_OPRREG:RXENB = ’1’: The same operation as reception only mode. I2Sn_OPRREG:START = ’1’, I2Sn_OPRREG:TXENB = ’1’, and I2Sn_OPRREG:RXENB = ’1’: Frame synchronous signal is output from the state that transmission FIFO is not empty and reception FIFO is not full. Then frame synchronous signal is output with the frame rate defined by the register setting. At the same time, empty frame is output if transmission FIFO is empty. Empty frame’s serial data can be set to ’0’ or ’1’ by register setting. Every time frame synchronous signal is output, frame is received. **Burst Mode (I2Sn_CNTREG:FRUN = '0'):** I2Sn_OPRREG:START = ’1’, I2Sn_OPRREG:TXENB = ’1’, and I2Sn_OPRREG:RXENB = ’0’: The same operation as transmission only mode. I2Sn_OPRREG:START = ’1’, I2Sn_OPRREG:TXENB = ’0’, and I2Sn_OPRREG:RXENB = ’1’: The same operation as reception only mode. I2Sn_OPRREG:START = ’1’, I2Sn_OPRREG:TXENB = ’1’, and I2Sn_OPRREG:RXENB = ’1’: Frame synchronous signal is output from the state that transmission FIFO is not empty and reception FIFO is not full. After completion of one frame output or at idle state, transmission/reception FIFO status is always confirmed. If transmission FIFO is not empty and reception FIFO is not full, frame synchronous signal is output to perform frame transmission/reception.	**Free-Running Mode (I2Sn_CNTREG:FRUN = '1'):** I2Sn_OPRREG:START = ’1’, I2Sn_OPRREG:TXENB = ’1’, and I2Sn_OPRREG:RXENB = ’0’: The same operation as transmission only mode. I2Sn_OPRREG:START = ’1’, I2Sn_OPRREG:TXENB = ’0’, and I2Sn_OPRREG:RXENB = ’1’: The same operation as reception only mode. I2Sn_OPRREG:START = ’1’, I2Sn_OPRREG:TXENB = ’1’, and I2Sn_OPRREG:RXENB = ’1’: Frame synchronous signal is input with the frame rate defined by the register setting. At the same time, empty frame is output if transmission FIFO is empty. The serial data can be set to ’0’ or ’1’ by the register setting I2Sn_CNTREG:MSKB. Every time frame synchronous signal is input, frame is received. **Burst Mode (I2Sn_CNTREG:FRUN = '0'):** Every time frame synchronous signal is input when I2Sn_OPRREG:START bit is ’1’, transmission and reception for one frame is performed. When the frame synchronous signal is input, empty frame is output if transmission FIFO is empty.

Transfer Setting	Operation	Master Mode (I2Sn_CNTREG:MSMD = '1')	Slave Mode (I2Sn_CNTREG:MSMD = '0')
Simultaneous Transfer I2Sn_CNTREG: TXDIS = ‘0’ I2Sn_CNTREG: RXDIS = ‘0’	Stop	Stop operation has following states: Transmission stop: Transmission FIFO becomes empty when data is not transferred from internal memory to I2S transmission FIFO. Reception stop: Data does not need to be transferred from I2S reception FIFO to internal memory. **To Maintain I2Sn_OPRREG:START Bit to '1':** In free-running mode frame synchronous signal is output. In the burst mode, when transmission FIFO becomes empty, frame synchronous output is stopped. Transmission stop: I2Sn_OPRREG:TXENB = '1': Empty frame bit is output when transmission FIFO becomes empty. I2Sn_OPRREG:TXENB = '0': Transmission FIFO becomes empty and transmission serial data bus becomes high impedance. Writing to transmission FIFO stops. Reception stop: When ’0’ is written to I2Sn_OPRREG:RXENB, reception FIFO becomes empty and frame reception operation stops. **To Make I2Sn_OPRREG:START Bit '0':** When ’0’ is written to I2Sn_OPRREG:START, transmission/reception FIFO becomes empty. The clock supply to the internal serial control part stops regardless of I2Sn_OPRREG:TXENB and I2Sn_OPRREG:RXENB status. SCK output to the external part and frame synchronous signal output is also stopped.	**To Maintain I2Sn_OPRREG:START Bit to '1':** Transmission stop: When I2Sn_OPRREG:TXENB = ’1’, empty frame bits are output after transmission FIFO becomes empty. When ’0’ is written to I2Sn_OPRREG:TXENB, transmisison FIFO becomes empty and transmission serial data bus becomes high impedance. Data present in the transmission FIFO at the time ’0’ was written to I2Sn_OPRREG:TXENB is not transmitted. While I2Sn_OPRREG:TXENB = ’0’, data is not written to transmission FIFO. Reception stop: When ’0’ is written to I2Sn_OPRREG:RXENB, reception FIFO becomes empty and frame reception operation stops. **To Make I2Sn_OPRREG:START Bit '0':** When ’0’ is written to I2Sn_OPRREG:START, transmission/reception FIFO becomes empty. Transmission/reception is stopped regardless of I2Sn_OPRREG:TXENB and I2Sn_OPRREG:RXENB status.
	Abnormality	When reading from transmission FIFO occurs while it is empty, empty frame bit is output. For the setting conditions of I2Sn_STATUS:TXUDR0 and I2Sn_STATUS:TXUDR1, refer to their bit descriptions. When writing to transmission FIFO occurs while it is full, I2Sn_STATUS:TXOVR is set to ’1’. When read access occurs to reception FIFO while it is empty, I2Sn_STATUS:RXUDR is set to ’1’. When writing to reception FIFO occurs while it is full, I2Sn_STATUS:RXOVR is set to ’1’.	When reading from transmission FIFO occurs while it is empty, empty frame bit is output. For the setting conditions of I2Sn_STATUS:TXUDR0 and I2Sn_STATUS:TXUDR1, refer to their bit descriptions. When writing to transmission FIFO occurs while it is full, I2Sn_STATUS:TXOVR is set to ’1’. When read access occurs to reception FIFO while it is empty, I2Sn_STATUS:RXUDR is set to ’1’. When writing to reception FIFO occurs while it is full, I2Sn_STATUS:RXOVR is set to ’1’. If the frame synchronous signal is not input with the defined frame rate in the free-running mode, I2Sn_STATUS:FERR is set to ’1’. If the following frame synchronous signal is input before completing one frame transmission in the burst mode, I2Sn_STATUS:FERR is set to ’1’.

3.3. Bit Alignment

Transmission Word Alignment

Figure 24-4 Transmission Word Line Chart



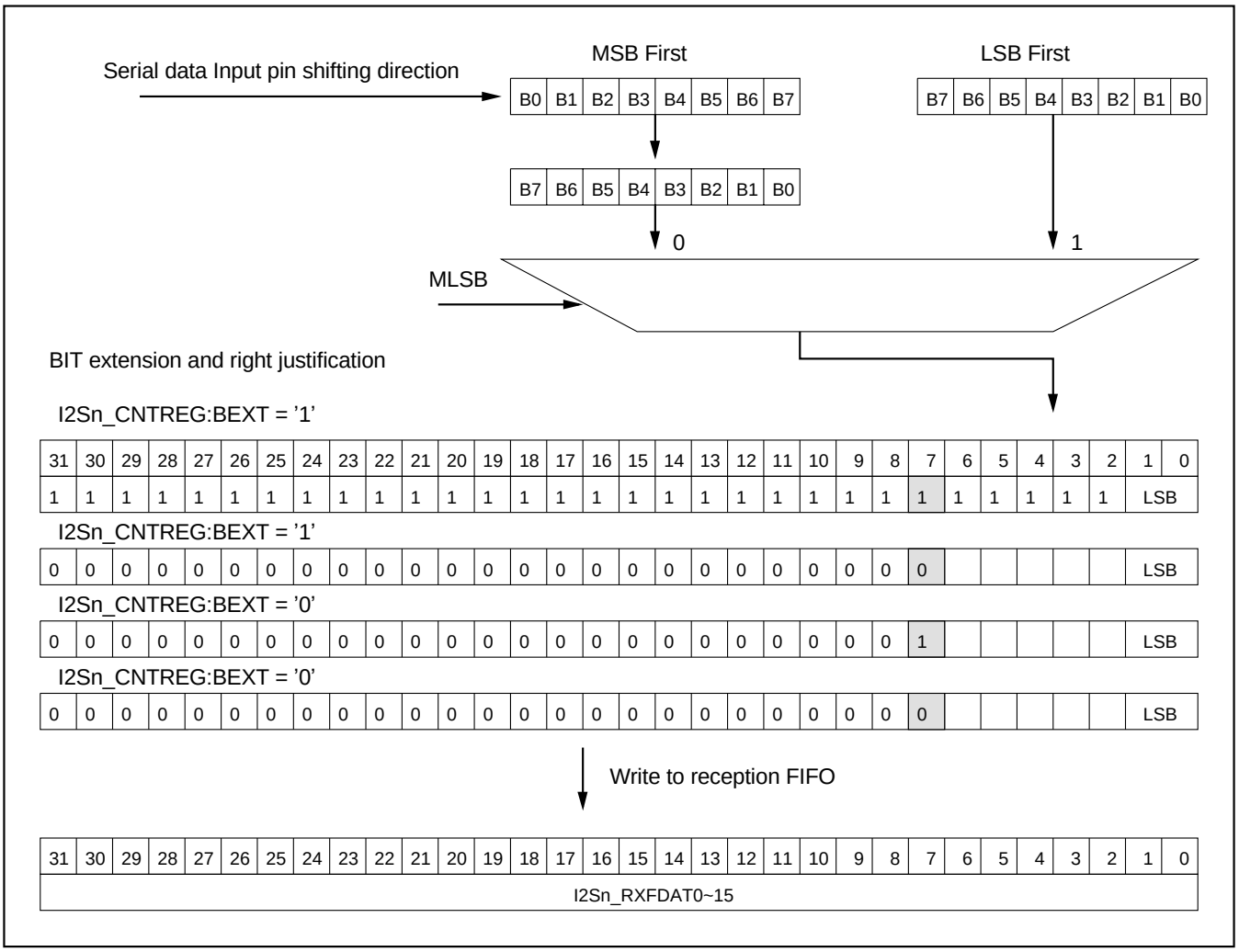
When transmission is performed with serial bus, word is sent MSB first when I2Sn_CNTREG:MSLB is '0' and LSB first when the value is '1'. When channel length (set to I2Sn_MCR0REG:S0CHL and I2Sn_MCR0REG:S1CHL) is longer than the word length (set to I2Sn_MCR0REG:S0WDL and I2Sn_MCR0REG:S1WDL), remaining bits in the channel become I2Sn_CNTREG:MSKB. Setting the channel length to shorter than the word length is prohibited.

Notes:

- *AB0, AB1, AB2, AB3, AH0, AH1, and AW on the above chart indicate byte 0, byte 1, byte 2, byte 3, half word 0, half word 1, and word at write accessing to I2Sn_TXFDAT0 to 15 on AHB bus*
- *Each FB0, FB1, FB2, FB3, FH0, FH1, and FW indicate AB0, AB1, AB2, AB3, AH0, AH1, and AW at writing to transmission FIFO after they are right justified*

Reception Word Alignment

Figure 24-5 Reception Word Line Chart



This chart shows word line example of when word length is 8.

The word received from serial bus is always written to reception FIFO after being right justified.

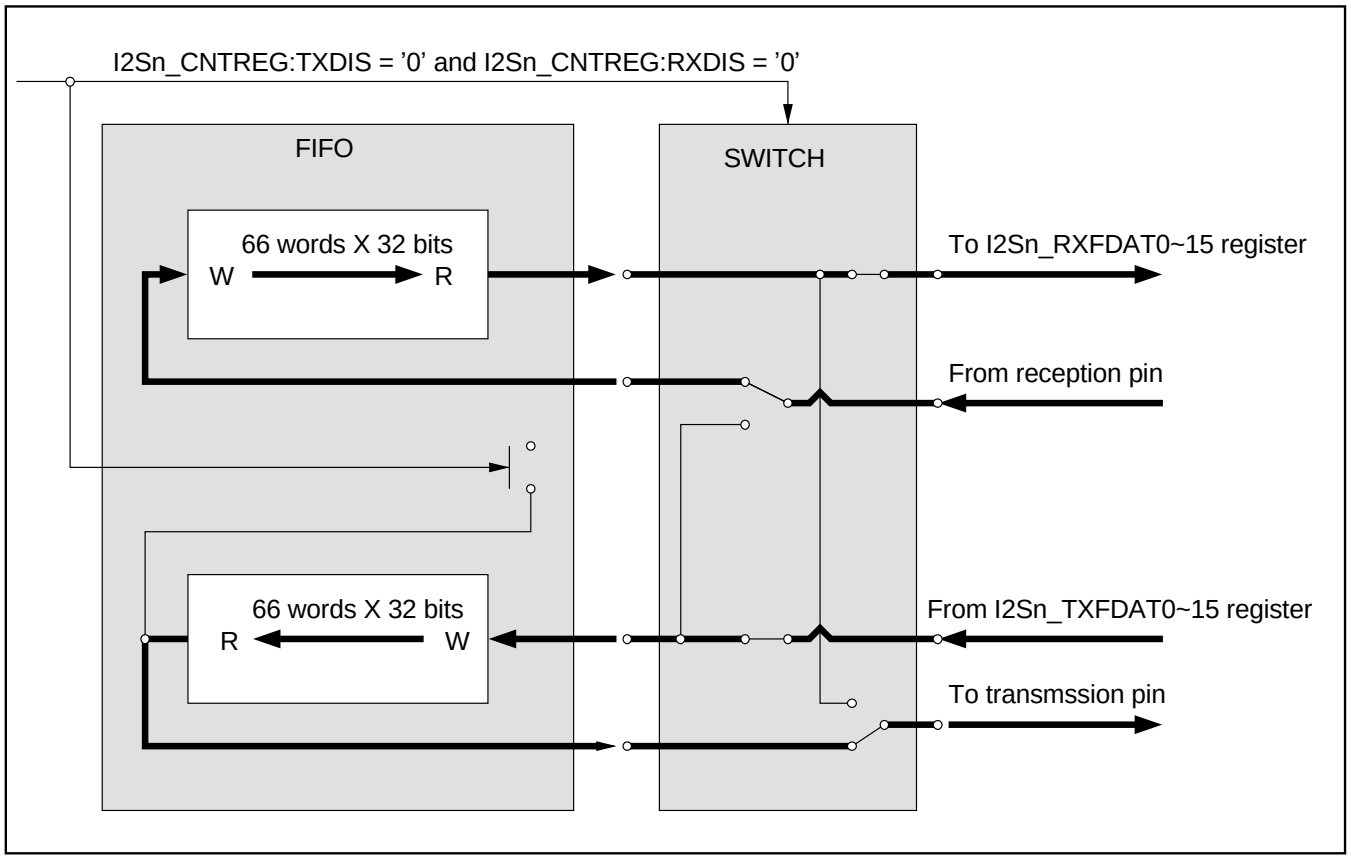
Therefore, read access should be performed from AHB bus to I2Sn_RXFDAT0 to 15 in order to read as follows:

Word length:

- 8 or less: byte 0
- 9- 16: half word 0
- 17-32: all words

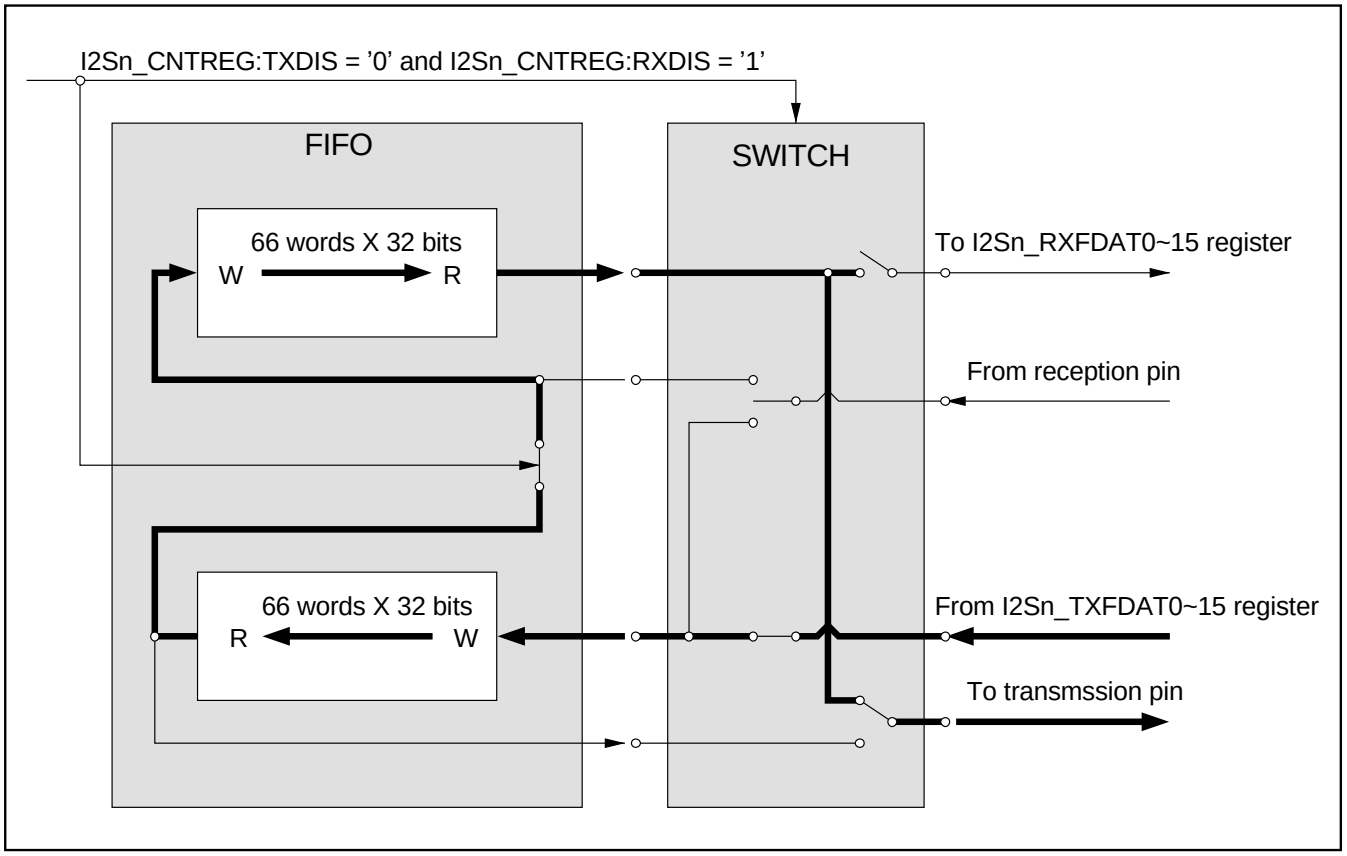
3.4. FIFO Construction

Figure 24-6 Simultaneous Transfer Mode Data Flow



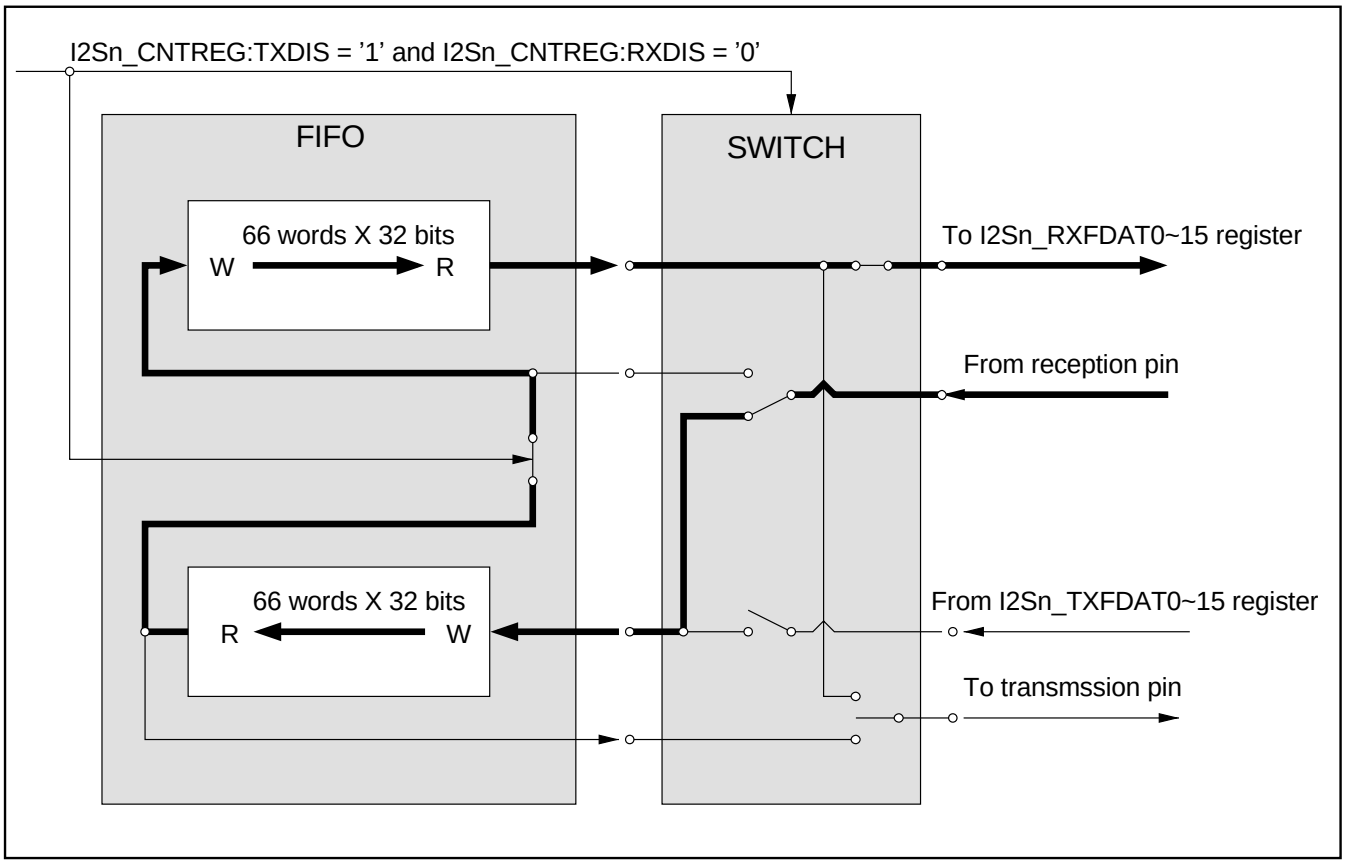
When I2Sn_CNTREG:TXDIS = '0' and I2Sn_CNTREG:RXDIS = '0', the mode is set to simultaneous transfer mode which operates with 66-word x 32-bit transmission and reception FIFOs.

Figure 24-7 Transmission Only Mode Data Flow



When I2Sn_CNTREG:TXDIS = ’0’ and I2Sn_CNTREG:RXDIS = ’1’, the mode is set to transmission only mode which operates with a 132-word x 32-bit transmission FIFO, and reception is not performed.

Figure 24-8 Reception Only Mode Data Flow



When I2Sn_CNTREG:TXDIS = '1' and I2Sn_CNTREG:RXDIS = '0', the mode is set to reception only mode which operates with a 132-word x 32-bit reception FIFO, and transmission is not performed.

3.5. Caution Summary

- I2Sn_MCR0REG:S0WDL, I2Sn_MCR0REG:S0CHL 1 to 6 bits are prohibited
- I2Sn_MCR0REG:S1WDL, I2Sn_MCR0REG:S1CHL 1 to 6 bits are prohibited
- When channel length (set to I2Sn_MCR0REG:S0CHL and I2Sn_MCR0REG:S1CHL) is longer than the word length (set to I2Sn_MCR0REG:S0WDL and I2Sn_MCR0REG:S1WDL), remaining bits in the channel become I2Sn_CNTREG:MSKB. Setting the channel length to shorter than the word length is prohibited
- Pulse width of one channel (I2Sn_CNTREG:FSLN = '1') is prohibited when frame length is set to one channel by I2Sn_MCR0REG:S0CHN = 0 and I2Sn_CNTREG:SBFN = '0'
- Rewrite to I2Sn_CNTREG, I2Sn_MCR0REG, I2Sn_MCR1REG, and I2Sn_MCR2REG is prohibited after I2Sn_OPRREG:START is set
- Rewrite to I2Sn_CNTREG, I2Sn_MCR0REG, I2Sn_MCR1REG, and I2Sn_MCR2REG is prohibited while DBGE is set to ’1’ and the processor is in debug state

4. Registers

This section describes the registers of the I2S in detail.

The suffix 'n' in the register name indicates that the register is in instance 'n' of the module.

Registers of I2S

The following registers are available for each instance of I2S:

- Reception FIFO Data Register (I2Sn_RXFDAT0 to 15)
- Transmission FIFO Data Register (I2Sn_TXFDAT0 to 15)
- Control Register (I2Sn_CNTREG)
- Channel Control Register 0 (I2Sn_MCR0REG)
- Channel Control Register 1 (I2Sn_MCR1REG)
- Channel Control Register 2 (I2Sn_MCR2REG)
- Operation Control Register (I2Sn_OPRREG)
- Software Reset Register (I2Sn_SRST)
- Interrupt Control Register (I2Sn_INTCNT)
- Status Register (I2Sn_STATUS)
- DMA Activate Register (I2Sn_DMAACT)
- Debug Register (I2Sn_DEBUG)
- Module ID Register (I2Sn_MIDREG)

Memory Layout of I2S Registers

Offset	Register Name				Access Size
	+3	+2	+1	+0	
0x00000000	I2Sn_RXFDAT0 00000000 00000000 00000000 00000000				B,H,W
0x00000004	I2Sn_RXFDAT1 00000000 00000000 00000000 00000000				B,H,W
0x00000008	I2Sn_RXFDAT2 00000000 00000000 00000000 00000000				B,H,W
0x0000000C	I2Sn_RXFDAT3 00000000 00000000 00000000 00000000				B,H,W
0x00000010	I2Sn_RXFDAT4 00000000 00000000 00000000 00000000				B,H,W
0x00000014	I2Sn_RXFDAT5 00000000 00000000 00000000 00000000				B,H,W
0x00000018	I2Sn_RXFDAT6 00000000 00000000 00000000 00000000				B,H,W
0x0000001C	I2Sn_RXFDAT7 00000000 00000000 00000000 00000000				B,H,W
0x00000020	I2Sn_RXFDAT8 00000000 00000000 00000000 00000000				B,H,W
0x00000024	I2Sn_RXFDAT9 00000000 00000000 00000000 00000000				B,H,W
0x00000028	I2Sn_RXFDAT10 00000000 00000000 00000000 00000000				B,H,W
0x0000002C	I2Sn_RXFDAT11 00000000 00000000 00000000 00000000				B,H,W
0x00000030	I2Sn_RXFDAT12 00000000 00000000 00000000 00000000				B,H,W
0x00000034	I2Sn_RXFDAT13 00000000 00000000 00000000 00000000				B,H,W
0x00000038	I2Sn_RXFDAT14 00000000 00000000 00000000 00000000				B,H,W
0x0000003C	I2Sn_RXFDAT15 00000000 00000000 00000000 00000000				B,H,W
0x00000040	I2Sn_TXFDAT0 00000000 00000000 00000000 00000000				B,H,W
0x00000044	I2Sn_TXFDAT1 00000000 00000000 00000000 00000000				B,H,W
0x00000048	I2Sn_TXFDAT2 00000000 00000000 00000000 00000000				B,H,W
0x0000004C	I2Sn_TXFDAT3 00000000 00000000 00000000 00000000				B,H,W
0x00000050	I2Sn_TXFDAT4 00000000 00000000 00000000 00000000				B,H,W
0x00000054	I2Sn_TXFDAT5 00000000 00000000 00000000 00000000				B,H,W
0x00000058	I2Sn_TXFDAT6 00000000 00000000 00000000 00000000				B,H,W
0x0000005C	I2Sn_TXFDAT7 00000000 00000000 00000000 00000000				B,H,W
0x00000060	I2Sn_TXFDAT8 00000000 00000000 00000000 00000000				B,H,W
0x00000064	I2Sn_TXFDAT9 00000000 00000000 00000000 00000000				B,H,W
0x00000068	I2Sn_TXFDAT10 00000000 00000000 00000000 00000000				B,H,W
0x0000006C	I2Sn_TXFDAT11 00000000 00000000 00000000 00000000				B,H,W
0x00000070	I2Sn_TXFDAT12 00000000 00000000 00000000 00000000				B,H,W
0x00000074	I2Sn_TXFDAT13 00000000 00000000 00000000 00000000				B,H,W

Offset	Register Name				Access Size
	+3	+2	+1	+0	
0x00000078	I2Sn_TXFDAT14 00000000 00000000 00000000 00000000				B,H,W
0x0000007C	I2Sn_TXFDAT15 00000000 00000000 00000000 00000000				B,H,W
0x00000080	I2Sn_CNTREG 00000000 00000000 00000000 01100000				B,H,W
0x00000084	I2Sn_MCR0REG 00000000 00000000 00000000 00000000				B,H,W
0x00000088	I2Sn_MCR1REG 00000000 00000000 00000000 00000000				B,H,W
0x0000008C	I2Sn_MCR2REG 00000000 00000000 00000000 00000000				B,H,W
0x00000090	I2Sn_OPRREG 00000000 00000000 00000000 00000000				B,H,W
0x00000094	I2Sn_SRST 00000000 00000000 00000000 00000000				B,H,W
0x00000098	I2Sn_INTCNT 01111111 00111111 00000000 00000000				B,H,W
0x0000009C	I2Sn_STATUS 00000000 00000000 00000000 00000000				B,H,W
0x000000A0	I2Sn_DMAACT 00000000 00000000 00000000 00000000				B,H,W
0x000000A4	I2Sn_DEBUG 00000000 00000000 00000000 00000000				B,H,W
0x000000A8	I2Sn_MIDREG 00000000 00000000 00000000 00000001				B,H,W
0x000000AC	reserved XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				-

4.1. Reception FIFO Data Register (I2Sn_RXFDAT0 to 15)

These registers are reception FIFO registers that can maintain up to 66 words (simultaneous transfer mode) or 132 words (reception only mode). There are 16 such registers, all consecutively placed in the register map. This is to support AHB burst transfers. Read access to any of the I2Sn_RXFDAT0 to 15 registers returns the word from reception FIFO. Each of these 16 registers are identical in function. Only one register (i.e. I2Sn_RXFDAT0) is explained here.

REGISTER_NAME	I2Sn_RXFDATi
OFFSET	0x0 + i*0x4
ACCESS_SIZE	B H W
MULTIPLE	0:15
NUMERIC_TYPE	
OTHER	

Reception FIFO Data Register 0 (I2Sn_RXFDAT0)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	RXDATA[31]	RXDATA[30]	RXDATA[29]	RXDATA[28]	RXDATA[27]	RXDATA[26]	RXDATA[25]	RXDATA[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

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	RXDATA[23]	RXDATA[22]	RXDATA[21]	RXDATA[20]	RXDATA[19]	RXDATA[18]	RXDATA[17]	RXDATA[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

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	RXDATA[15]	RXDATA[14]	RXDATA[13]	RXDATA[12]	RXDATA[11]	RXDATA[10]	RXDATA[9]	RXDATA[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

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	RXDATA[7]	RXDATA[6]	RXDATA[5]	RXDATA[4]	RXDATA[3]	RXDATA[2]	RXDATA[1]	RXDATA[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] RXDATA : Receive Data

The word received from serial bus is written to reception FIFO.

When frame is 1 sub frame construction and word length set to I2Sn_MCR0REG:S0WDL is 32 bits or less (16 bits when I2Sn_CNTREG:RHLL register is '1'), it is written to reception FIFO after higher order bit is extended.

When frame is 2 sub frame construction and word length set to I2Sn_MCR0REG:S0WDL is 32 bits or less (16 bits when I2Sn_CNTREG:RHLL register is '1'), reception data of sub frame '0' is written to reception FIFO after higher order bit is extended.

For the case that word length set to I2Sn_MCR0REG:S1WDL is 32 bits or less, reception data of sub frame 1 is written to reception FIFO after higher order bit is extended.

When I2Sn_CNTREG:BEXT is '1', it is extended with MSB of reception word (sign extension). For the case that the value is '0', it is enhanced by '0'.

Reading this register returns a word of data from the Rx FIFO location pointed by the Rx FIFO read pointer. After a read access to this register, the Rx FIFO read pointer is incremented, provided that the read cycle was initiated by the AHB master other than the Debug Access Port (DAP) controller. If the DAP controller reads this register, the Rx FIFO read pointer is not incremented.

When I2Sn_STATUS:RXNUM is 0x00000000, invalid data is read. An Rx FIFO underflow error (I2Sn_STATUS:RXUDR) flag is set, if the read cycle was initiated by the AHB master other than the DAP controller. If the DAP controller reads this register while the Rx FIFO is empty, the I2Sn_STATUS:RXUDR flag is not set.

4.2. Transmission FIFO Data Register (I2Sn_TXFDAT0 to 15)

These registers are transmission FIFO registers that can maintain up to 66 words (simultaneous transfer mode) or 132 words (transmission mode only). There are 16 such registers, all consecutively placed in the register map. This is to support AHB burst transfers. A write access to any of I2Sn_TXFDAT0 to 15 register writes a word of data in the transmission FIFO. Each of these 16 registers are identical in function. Only one register (i.e. I2Sn_TXFDAT0) is explained here.

REGISTER_NAME	I2Sn_TXFDATi
OFFSET	0x40 + i*0x4
ACCESS_SIZE	B H W
MULTIPLE	0:15
NUMERIC_TYPE	
OTHER	

Transmission FIFO Data Register 0 (I2Sn_TXFDAT0)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	TXDATA[31]	TXDATA[30]	TXDATA[29]	TXDATA[28]	TXDATA[27]	TXDATA[26]	TXDATA[25]	TXDATA[24]
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	TXDATA[23]	TXDATA[22]	TXDATA[21]	TXDATA[20]	TXDATA[19]	TXDATA[18]	TXDATA[17]	TXDATA[16]
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	TXDATA[15]	TXDATA[14]	TXDATA[13]	TXDATA[12]	TXDATA[11]	TXDATA[10]	TXDATA[9]	TXDATA[8]
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	TXDATA[7]	TXDATA[6]	TXDATA[5]	TXDATA[4]	TXDATA[3]	TXDATA[2]	TXDATA[1]	TXDATA[0]
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] TXDATA : Transmit Data

Word to be transmitted can be written as long as transmission FIFO is not full.

Write access can be performed regardless of shift register's operation status. Write access to full transmission FIFO is ignored and I2Sn_STATUS:TXOVR flag is set. Although writing data is accessed in word, half-word, and byte access, actual number of bits to be transmitted is determined by I2Sn_MCR0REG:S0WDL and I2Sn_MCR0REG:S1WDL (when frame is 2 sub frame).

The data read from I2Sn_TXFDAT0 returns '0'.

4.3. Control Register (I2Sn_CNTREG)

The Control Register (I2Sn_CNTREG) sets the module configuration.

REGISTER_NAME	I2Sn_CNTREG
OFFSET	0x80
ACCESS_SIZE	B H W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

Control Register (I2Sn_CNTREG)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	CKRT[5]	CKRT[4]	CKRT[3]	CKRT[2]	CKRT[1]	CKRT[0]	OVHD[9]	OVHD[8]
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	OVHD[7]	OVHD[6]	OVHD[5]	OVHD[4]	OVHD[3]	OVHD[2]	OVHD[1]	OVHD[0]
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	read0	MSKB	MSMD	SBFN	RHLL	ECKM	BEXT	FRUN
ACCESS_TYPE	R0,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

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	MSLB	TXDIS	RXDIS	SMPL	CPOL	FSPH	FSLN	FSPL
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE								
INITIAL_VALUE	0	1	1	0	0	0	0	0

[bit31:26] CKRT : Clock Divider

This sets output clock frequency dividing ratio at master operation.

Internal bus clock is divided at I2Sn_CNTREG:ECKM = '0', and external clock (ECLK) is divided at I2Sn_CNTREG:ECKM = '1'. Only even number of the ratio is supported and output clock's duty becomes 50%.

bit	Description
000000	Output clock is not divided
000001	Output clock is divided by 2
000010	Output clock is divided by 4
000011	Output clock is divided by 6
	...
	...
111110	Output clock is divided by 124
111111	Output clock is divided by 126

[bit25:16] OVHD : Frame Rate Control

Frame rate can be adjusted by inserting overhead (OVHD) bits following the valid data of the frame. The overhead section of the transmission frame enters the state of high impedance. Up to 1023 overhead bits can be inserted at the end of the frame.

The value set to OVHD becomes the number of insertion bits.

The following expressions are formed for OVHD and frame synchronous signal cycle.

1 sub frame construction:

- OVHD = Frame synchronous signal cycle/SCK cycle - (I2Sn_MCR0REG:S0CHL + 1) * (I2Sn_MCR0REG:S0CHN + 1)

2 sub frame construction:

- OVHD = Frame synchronous signal cycle/SCK cycle - (I2Sn_MCR0REG:S0CHL + 1) * (I2Sn_MCR0REG:S0CHN + 1) - (I2Sn_MCR0REG:S1CHL + 1) * (I2Sn_MCR0REG:S1CHN + 1)

[bit15] read0 : -

[bit14] MSKB : Serial Output Data in case of Invalid/Empty Frame Transmission

For master operation (I2Sn_CNTREG:MSMD = '1'), free-running mode (I2Sn_CNTREG:FRUN = '0'), and I2Sn_OPRREG:TXENB = '1':

- When transmission FIFO is empty at frame synchronous signal output, MSKB is output to all valid channels of its transmission frame.

For slave operation (I2Sn_CNTREG:MSMD = '0') and I2Sn_OPRREG:TXENB = '1':

- When transmission FIFO is empty at frame synchronous signal reception, MSKB is output to all valid channels of its transmission frame.

For the case that transmission word length is shorter than the channel length, MSKB is driven to the rest of bit in transmission channel (channel length - word length).

[bit13] MSMD : Master and Slave Mode Select

Master and Slave modes are set.

bit	Description
0	Slave operation
1	Master operation

[bit12] SBFN : Sub Frame Construction

Sub frame construction (number of sub frame) of the frame is specified.

bit	Description
0	1 Sub frame construction (only sub frame 0)
1	2 Sub frame construction (sub frame 0 and sub frame 1). Frame starts from 0th sub frame

[bit11] RHLL : Word Construction

Word construction of FIFO – 1 word (32 bits) or 2 half words (16 bits) – is specified.

It is considered to be used at protocol, such as MSB-justified.

bit	Description
0	32-bit FIFO word is handled as 1 word
1	32-bit FIFO word is handled as 2 half words at serial bus with dividing 16 bits each to low order and high order. They are transferred by serial bus in order of low order, high order, low order, and high order. At reception, 2 consecutive half words from serial bus are handled as low order and high order, and they are put in 1 word (32 bits) to write to reception FIFO

[bit10] ECKM : Clock Selector

Clock frequency dividing is selected in master mode.

bit	Description
0	Internal bus clock is divided and output
1	External clock (ECLK)is divided and output

[bit9] BEXT : Bit Extension

When reception word length is shorter than the word length of FIFO (32 bits when RHLL is '0' and 16 bits when RHLL is '1'), extension mode of upper bit (word length of FIFO-reception word length) should be set.

bit	Description
0	Extended by '0'
1	Extended by sign bit (if MSB of word/half word is '1', then it is extended by '1', if MSB is '0' then it is extended by '0')

[bit8] FRUN : Output Mode of Frame Synchronous Signal

bit	Description
0	Burst mode When I2Sn_OPRREG:START bit is '1', frame synchronous signal is output according to I2Sn_OPRREG:TXENB, I2Sn_OPRREG:RXENB, and transmission/reception FIFO conditions. When I2Sn_OPRREG:START bit is '0', frame synchronous signal is not output.
1	Free-running mode When I2Sn_OPRREG:START register is '1', frame synchronous signal proceeds free-running with the set frame rate. When I2Sn_OPRREG:START bit is '0', frame synchronous signal is not output.

[bit7] MSLB : Shifting Order

Word bit's shift order is set.

bit	Description
0	Shift starts from MSB of the word
1	Shift starts from LSB of the word

[bit6] TXDIS : Transmitter Disable

Transmitting function is enabled or disabled.

bit	Description
0	Transmitting function is enabled
1	Transmitting function is disabled

[bit5] RXDIS : Receiver Disable

Receiving function is enabled or disabled.

bit	Description
0	Receiving function is enabled
1	Receiving function is disabled

[bit4] SMPL : Sampling Point

Sampling point of the data is specified.

bit	Description
0	Sampling at the center of reception data
1	Sampling at the end of reception data

[bit3] CPOL : Clock Polarity

SCK polarity which drives/samples serial data is specified.

bit	Description
0	Data is driven at rising edge of SCK, and sampled at falling edge
1	Data is driven at falling edge of SCK, and sampled at rising edge

[bit2] FSPH : Frame Sync Phase

Phase is specified to WS frame data.

bit	Description
0	WS becomes valid '1' clock before the first bit of frame data
1	WS becomes valid at the same time as the first bit of frame data

[bit1] FSLN : Frame Sync Pulse Width

Pulse width of WS is specified.

bit	Description
0	Pulse width is 1 cycle/SCK long (1-bit)
1	Pulse width is 1 channel long (1 channel)

Pulse width of one channel FSLN = '1' is prohibited when frame length is set to one channel by I2Sn_MCR0REG:S0CHN = 0x0 and I2Sn_CNTREG:SBFN = '0'.

[bit0] FSPL : Frame Sync Polarity

Polarity of WS is set.

bit	Description
0	Frame synchronous signal becomes valid when WS is '1'
1	Frame synchronous signal becomes valid when WS is '0'

4.4. Channel Control Register 0 (I2Sn_MCR0REG)

The Channel Control Register 0 (I2Sn_MCR0REG) controls the frame construction.

REGISTER_NAME	I2Sn_MCR0REG
OFFSET	0x84
ACCESS_SIZE	B H W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

Channel Control Register 0 (I2Sn_MCR0REG)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	read0	S1CHN[4]	S1CHN[3]	S1CHN[2]	S1CHN[1]	S1CHN[0]	S1CHL[4]	S1CHL[3]
ACCESS_TYPE	R0,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

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	S1CHL[2]	S1CHL[1]	S1CHL[0]	S1WDL[4]	S1WDL[3]	S1WDL[2]	S1WDL[1]	S1WDL[0]
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	read0	S0CHN[4]	S0CHN[3]	S0CHN[2]	S0CHN[1]	S0CHN[0]	S0CHL[4]	S0CHL[3]
ACCESS_TYPE	R0,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

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	S0CHL[2]	S0CHL[1]	S0CHL[0]	S0WDL[4]	S0WDL[3]	S0WDL[2]	S0WDL[1]	S0WDL[0]
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] read0 : -

[bit30:26] S1CHN : Sub Frame 1 Channel Numbers

Number of channels of sub frame 1 is set.

This is valid only when the frame is 2 sub frame construction (I2Sn_CNTREG:SBFN is '1') and is invalid when the frame is 1 sub frame construction (I2Sn_CNTREG:SBFN is '0').

Up to 32 channels can be specified, and S1CHN needs to be set to 'number of channel - 1'.

Setting examples are shown below.

bit	Description
00000	Sub frame 1 becomes 1 channel construction
00001	Sub frame 1 becomes 2 channel construction
00010	Sub frame 1 becomes 3 channel construction
	...
	...
11110	Sub frame 1 becomes 31 channel construction
11111	Sub frame 1 becomes 32 channel construction

[bit25:21] S1CHL : Sub Frame 1 Channel Length

Channel length of the channel constructing sub frame 1 (bit length of channel) is set.

7 to 32 bits of channel length are available but 1 to 6 bits are prohibited. S1CHL needs to be set to 'channel length - 1'

Setting examples are shown below.

bit	Description
00000-00101	Setting is prohibited
00110	Sub frame 1 channel length is 7 bits
00111	Sub frame 1 channel length is 8 bits
	...
	...
11110	Sub frame 1 channel length is 31 bits
11111	Sub frame 1 channel length is 32 bits

Channel length can be set to 32 bits or less regardless of I2Sn_CNTREG:RHLL.

[bit20:16] S1WDL : Sub Frame 1 Word Length

word length of the channel constructing sub frame 1 (bit length of word) is set.

7 to 32 bits of word length are available but 1 to 6 bits are prohibited. S1WDL needs to be set to 'word length - 1'

S1WDL needs to be set less than or equal to the value set in I2Sn_MCR0REG:S1CHL.

Setting examples are shown below.

bit	Description
00000-00101	Setting is prohibited
00110	Sub frame 1 word length is 7 bits
00111	Sub frame 1 word length is 8 bits
	...
	...
11110	Sub frame 1 word length is 31 bits
11111	Sub frame 1 word length is 32 bits

Notes:

- *1. If I2Sn_CNTREG:RHLL is '1', set word length to 16 bits or less.*
- *2. If I2Sn_CNTREG:RHLL is '0', set word length to 32 bits or less.*

S1WDL is valid only in 2 sub frame construction (I2Sn_CNTREG:SBFN is '1') and is invalid in 1 sub frame construction (I2Sn_CNTREG:SBFN is '0').

[bit15] read0 : -

[bit14:10] S0CHN : Sub Frame 0 Channel Numbers

Number of channels of sub frame 0 is set up to 32 channels.

S0CHN needs to be set to 'number of channel - 1'.

Setting examples are shown below.

bit	Description
00000	Sub frame 0 becomes 1 channel construction
00001	Sub frame 0 becomes 2 channel construction
00010	Sub frame 0 becomes 3 channel construction
	...
	...
11110	Sub frame 0 becomes 31 channel construction
11111	Sub frame 0 becomes 32 channel construction

[bit9:5] S0CHL : Sub Frame 0 Channel Length

Channel length of the channel constructing sub frame 0 (bit length of channel) is set.

7 to 32 bits of channel length are available but 1 to 6 bits are prohibited. S0CHL needs to be set to 'channel length - 1'.

Setting examples are shown below.

bit	Description
00000-00101	Setting is prohibited
00110	Sub frame 0 channel length is 7 bits
00111	Sub frame 0 channel length is 8 bits
	...
	...
11110	Sub frame 0 channel length is 31 bits
11111	Sub frame 0 channel length is 32 bits

Channel length can be set to 32 bits or less regardless of I2Sn_CNTREG:RHLL

[bit4:0] S0WDL: Sub Frame 1 Word Length

word length of the channel constructing sub frame 0 (bit length of channel) is set.

7 to 32 bits of word length are available but 1 to 6 bits are prohibited. S0WDL needs to be set to 'word length - 1'.

S0WDL needs to be set less than or equal to the value set in I2Sn_MCR0REG:S0CHL.

Setting examples are shown below.

bit	Description
00000-00101	Setting is prohibited
00110	Sub frame 0 word length is 7 bits
00111	Sub frame 0 word length is 8 bits
	...
	...
11110	Sub frame 0 word length is 31 bits
11111	Sub frame 0 word length is 32 bits

Notes:

- *1. If I2Sn_CNTREG:RHLL is '1', set word length to 16 bits or less.*
- *2. If I2Sn_CNTREG:RHLL is '0', set word length to 32 bits or less.*

4.5. Channel Control Register 1 (I2Sn_MCR1REG)

The Channel Control Register 1 (I2Sn_MCR1REG) controls enable and disable functions to each channel of sub frame 0.

REGISTER_NAME	I2Sn_MCR1REG
OFFSET	0x88
ACCESS_SIZE	B H W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

Channel Control Register 1 (I2Sn_MCR1REG)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	S0CH[31]	S0CH[30]	S0CH[29]	S0CH[28]	S0CH[27]	S0CH[26]	S0CH[25]	S0CH[24]
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	S0CH[23]	S0CH[22]	S0CH[21]	S0CH[20]	S0CH[19]	S0CH[18]	S0CH[17]	S0CH[16]
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	S0CH[15]	S0CH[14]	S0CH[13]	S0CH[12]	S0CH[11]	S0CH[10]	S0CH[9]	S0CH[8]
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	S0CH[7]	S0CH[6]	S0CH[5]	S0CH[4]	S0CH[3]	S0CH[2]	S0CH[1]	S0CH[0]
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] S0CH : Sub Frame 0 Channel Enable

Each bit enables/disables corresponding channel of sub frame 0 (e.g. S0CH[0] bit controls 0th channel of sub frame 0, S0CH[31] bit controls 31st channel of sub frame 0).

bit	Description
0	The corresponding channel is disabled Transmission/reception is not performed to the disabled channel.
1	The corresponding channel is enabled Transmission/reception is performed to the enabled channel.

4.6. Channel Control Register 2 (I2Sn_MCR2REG)

The Channel Control Register 2 (I2Sn_MCR2REG) controls enable and disable functions to each channel of sub frame 1.

REGISTER_NAME	I2Sn_MCR2REG
OFFSET	0x8C
ACCESS_SIZE	B H W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

Channel Control Register 2 (I2Sn_MCR2REG)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	S1CH[31]	S1CH[30]	S1CH[29]	S1CH[28]	S1CH[27]	S1CH[26]	S1CH[25]	S1CH[24]
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	S1CH[23]	S1CH[22]	S1CH[21]	S1CH[20]	S1CH[19]	S1CH[18]	S1CH[17]	S1CH[16]
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	S1CH[15]	S1CH[14]	S1CH[13]	S1CH[12]	S1CH[11]	S1CH[10]	S1CH[9]	S1CH[8]
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	S1CH[7]	S1CH[6]	S1CH[5]	S1CH[4]	S1CH[3]	S1CH[2]	S1CH[1]	S1CH[0]
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] S1CH : Sub Frame 1 Channel Enable

Each bit enables/disables corresponding channel of sub frame 1 (e.g. S1CH[0] bit controls 0th channel of sub frame 1), S1CH[31] bit controls 31st channel of sub frame 1. When frame is 1 sub frame construction (I2Sn_CNTREG:SBFN is '0'), this is invalid.

bit	Description
0	The corresponding channel is disabled Transmission/reception is not performed to the disabled channel.
1	The corresponding channel is enabled Transmission/reception is performed to the enabled channel.

4.7. Operation Control Register (I2Sn_OPRREG)

The Operation Control Register (I2Sn_OPRREG) controls receive and transmit operation.

REGISTER_NAME	I2Sn_OPRREG
OFFSET	0x90
ACCESS_SIZE	B H W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

Operation Control Register (I2Sn_OPRREG)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	read0	read0	read0	read0	read0	read0	read0	RXENB
ACCESS_TYPE	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	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	read0	read0	read0	read0	read0	read0	read0	TXENB
ACCESS_TYPE	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	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	read0	read0	read0	read0	read0	read0	read0	read0
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	read0	read0	read0	read0	read0	read0	read0	START
ACCESS_TYPE	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R/W
PROT_TYPE								
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:25] read0 : -

[bit24] RXENB : Receive Enable

Enable/disable functions of receiving operation is set.

bit	Description
0	Receiving operation is disabled Reception FIFO becomes empty with writing '0' to this bit. When RXENB is '0', the data received from serial reception bus is not written to reception FIFO. DMA reception channel stops during DMA transfer.
1	Receiving operation is enabled

[bit23:17] read0 : -

[bit16] TXENB : Transmit Enable

Enable/disable functions of transmitting operation is set.

bit	Description
0	Transmit operation is disabled Transmit FIFO becomes empty with writing '0' to this bit. When TXENB is '0', the data written to Transmission FIFO Data Registers from CPU or DMA is not written to transmission FIFO. DMA transmit channel stops during DMA transfer.
1	Transmit operation is enabled

[bit15:8] read0 : -

[bit7:1] read0 : -

[bit0] START : I2S Enable

I2S is enabled/disabled.

bit	Description
0	I2S is stopped and internal transmission/reception FIFO becomes empty by writing '0' to this bit
1	I2S is operable.

When START is '1', it is prohibited to rewrite I2Sn_CNTREG, I2Sn_MCR0REG, I2Sn_MCR1REG, and I2Sn_MCR2REG registers.

4.8. Software Reset Register (I2Sn_SRST)

This register is to control software reset.

REGISTER_NAME	I2Sn_SRST
OFFSET	0x94
ACCESS_SIZE	B H W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

Software Reset Register (I2Sn_SRST)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	read0	read0	read0	read0	read0	read0	read0	read0
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	read0	read0	read0	read0	read0	read0	read0	read0
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	read0	read0	read0	read0	read0	read0	read0	read0
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	read0	read0	read0	read0	read0	read0	read0	SRST
ACCESS_TYPE	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R/W
PROT_TYPE								
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:8] read0 : -

[bit7:1] read0 : -

[bit0] SRST : Software Reset

Software reset is performed by writing '1'.

I2Sn_STATUS register and each internal state machine enter initial state by software reset and transmission/reception FIFO becomes empty.

There is no influence to registers other than I2Sn_STATUS, I2Sn_INTCNT, and I2Sn_DMAACT registers.

When read value is '0' after writing '1', it indicates software reset is completed. '1' indicates software reset is in process.

4.9. Interrupt Control Register (I2Sn_INTCNT)

This register is to enable or disable interrupt function.

REGISTER_NAME	I2Sn_INTCNT
OFFSET	0x98
ACCESS_SIZE	B H W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

Interrupt Control Register (I2Sn_INTCNT)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	read0	TXUD1M	TBERM	FERRM	TXUD0M	TXOVM	TXFDM	TXFIM
ACCESS_TYPE	R0,WX	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE								
INITIAL_VALUE	0	1	1	1	1	1	1	1

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	read0	read0	RBERM	RXUDM	RXOVM	EOPM	RXFDM	RXFIM
ACCESS_TYPE	R0,WX	R0,WX	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE								
INITIAL_VALUE	0	0	1	1	1	1	1	1

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	read0	read0	read0	read0	TFTH[3]	TFTH[2]	TFTH[1]	TFTH[0]
ACCESS_TYPE	R0,WX	R0,WX	R0,WX	R0,WX	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	read0	read0	RPTMR[1]	RPTMR[0]	RFTH[3]	RFTH[2]	RFTH[1]	RFTH[0]
ACCESS_TYPE	R0,WX	R0,WX	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] read0 : -

[bit30] TXUD1M : Tx FIFO Underflow Interrupt Mask

This is transmission FIFO underflow interrupt mask bit.

It becomes '1' by software reset.

bit	Description
0	Interrupt to CPU by I2Sn_STATUS:TXUDR1 is not masked
1	Interrupt to CPU by I2Sn_STATUS:TXUDR1 is masked

[bit29] TBERM : Tx Block Size Error Interrupt Mask

This is interrupt mask bit of block size error of transmission channel.

It becomes '1' by software reset.

bit	Description
0	Interrupt to CPU by I2Sn_STATUS:TBERR is not masked
1	Interrupt to CPU by I2Sn_STATUS:TBERR is masked

[bit28] FERRM : Frame Error Interrupt Mask

This is frame error interrupt mask bit.

It becomes '1' by software reset.

bit	Description
0	Interrupt to CPU by I2Sn_STATUS:FERR is not masked
1	Interrupt to CPU by I2Sn_STATUS:FERR is masked

[bit27] TXUD0M : Tx FIFO Underflow Interrupt Mask

This is transmission FIFO underflow interrupt mask bit.

It becomes '1' by software reset.

bit	Description
0	Interrupt to CPU by I2Sn_STATUS:TXUDR0 is not masked
1	Interrupt to CPU by I2Sn_STATUS:TXUDR0 is masked

[bit26] TXOVM : Tx FIFO Overflow Interrupt Mask

This is transmission FIFO overflow interrupt mask bit.

It becomes '1' by software reset.

bit	Description
0	Interrupt to CPU by I2Sn_STATUS:TXOVR is not masked
1	Interrupt to CPU by I2Sn_STATUS:TXOVR is masked

[bit25] TXFDM : Tx DMA Mask

This is transmission DMA request mask register bit.

It becomes '1' by software reset.

bit	Description
0	DMA transfer is requested when empty space in transmission FIFO is more than threshold value
1	DMA transfer is not requested even when empty space in transmission FIFO is more than threshold value

[bit24] TXFIM : Tx FIFO Interrupt Mask

This is transmission FIFO interrupt mask bit.

It becomes '1' by software reset.

bit	Description
0	Interrupt to CPU by I2Sn_STATUS:TXFI is not masked
1	Interrupt to CPU by I2Sn_STATUS:TXFI is masked

[bit23:22] read0 : -

[bit21] RBERM : Rx Block Size Error Interrupt Mask

This is interrupt mask bit of reception channel block size error.

It becomes '1' by software reset.

bit	Description
0	Interrupt to CPU by I2Sn_STATUS:RBERR is not masked
1	Interrupt to CPU by I2Sn_STATUS:RBERR is masked

[bit20] RXUDM : Rx FIFO Underflow Interrupt Mask

This is reception FIFO underflow interrupt mask bit.

It becomes '1' by software reset.

bit	Description
0	Interrupt to CPU by I2Sn_STATUS:RXUDR is not masked
1	Interrupt to CPU by I2Sn_STATUS:RXUDR is masked

[bit19] RXOVM : Rx FIFO Overflow Interrupt Mask

This is interrupt mask bit of reception FIFO overflow.

It becomes '1' by software reset.

bit	Description
0	Interrupt to CPU by I2Sn_STATUS:RXOVR is not masked
1	Interrupt to CPU by I2Sn_STATUS:RXOVR is masked

[bit18] EOPM : EOPI Interrupt Mask

This is interrupt mask bit by EOPI of status register.

It becomes '1' by software reset.

bit	Description
0	Interrupt to CPU by I2Sn_STATUS:EOPI is not masked
1	Interrupt to CPU by I2Sn_STATUS:EOPI is masked

[bit17] RXFDM : Rx FIFO DMA Mask

This is reception DMA request mask bit.

It becomes '1' by software reset.

bit	Description
0	DMA transfer is requested when reception data written to reception FIFO is more than threshold value, previous Rx DMA block transfer has completed and there is no Rx block size error.
1	DMA transfer is not requested

[bit16] RXFIM : Rx FIFO Interrupt Mask

This is reception FIFO interrupt mask bit.

It becomes '1' by software reset.

bit	Description
0	Interrupt to CPU by I2Sn_STATUS:RXFI is not masked
1	Interrupt to CPU by I2Sn_STATUS:RXFI is masked

[bit15:12] read0 : -

[bit11:8] TFTH : Tx FIFO Threshold

Threshold value of transmission FIFO is set.

Empty space of transmission FIFO is more than threshold value and I2Sn_INTCNT: Interrupt to CPU occurs.

Empty space of transmission FIFO is more than threshold value, previous Tx DMA block transfer has completed, there is no Tx block size error I2Sn_INTCNT:TXFDM is '0': DMA is requested to DMAC.

Note:

- *These bits must not be changed during DMA transfers*

[bit7:6] read0 : -

[bit5:4] RPTMR : Rx Completion Timer

This is packet reception completion timer setting bit which sets time-out value of the internal reception completion timer.

Reception FIFO is not empty and number of its data is smaller than or equal to threshold value: the timer always counts up.

Reception FIFO is empty or the data value is more than threshold value: the timer is cleared.

When the timer times out, I2Sn_STATUS:EOPI bit is set to '1'.

The timer becomes '00' by software reset.

bit	Description
00	0 (the timer is not in operation)
01	54000 Internal bus clock cycles
10	108000 Internal bus clock cycles
11	216000 Internal bus clock cycles

[bit3:0] RFTH : Rx FIFO Threshold

Threshold value of reception FIFO is set.

Number of receive words in reception FIFO is more than threshold value and I2Sn_INTCNT: Interrupt to CPU occurs.

Number of receive words in reception FIFO is more than threshold value, previous Rx DMA block transfer has completed, there is no Rx block size errorand I2Sn_INTCNT:RXFDM is '0': DMA is requested to DMAC.

Note:

- *These bits must not be changed during DMA transfers*

4.10. Status Register (I2Sn_STATUS)

This register gives status information.

REGISTER_NAME	I2Sn_STATUS
OFFSET	0x9C
ACCESS_SIZE	B H W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

Status Register (I2Sn_STATUS)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	TBERR	RBERR	FERR	TXUDR1	TXUDR0	TXOVR	RXUDR	RXOVR
ACCESS_TYPE	R,WX	R,WX	R,W1	R,W1	R,W1	R,W1	R,W1	R,W1
PROT_TYPE								
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	read0	read0	read0	read0	EOPI	BSY	TXFI	RXFI
ACCESS_TYPE	R0,WX	R0,WX	R0,WX	R0,WX	R,W1	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	TXNUM[7]	TXNUM[6]	TXNUM[5]	TXNUM[4]	TXNUM[3]	TXNUM[2]	TXNUM[1]	TXNUM[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

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	RXNUM[7]	RXNUM[6]	RXNUM[5]	RXNUM[4]	RXNUM[3]	RXNUM[2]	RXNUM[1]	RXNUM[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] TBERR : Tx Block Size Error

When I2Sn_DMAACT:TDMACT is '1' and block transfer through DMA transmission channel is more than I2Sn_INTCNT:TFTH + 1, this bit is set to 1 and the DMA transmission channel is stopped.

When TBERR is '1' and I2Sn_INTCNT:TBERM is '0', interrupt to CPU occurs.

This bit becomes '0' by software reset.

[30] RBERR : Rx Block Size Error

When I2Sn_DMAACT:RDMACT is '1' and block transfer through DMA reception channel is more than I2Sn_INTCNT:RFTH + 1, this bit is set to 1 and the DMA reception channel is stopped.

When RBERR is '1' and I2Sn_INTCNT:RBERM is '0', interrupt to CPU occurs.

This bit becomes '0' by software reset.

[bit29] FERR : Frame Error

Occurrence of frame error is indicated. This bit is set to '1' in the following cases:

- Frame synchronous signal cannot be received with the set frame rate in the free-running mode (I2Sn_CNTREG:FRUN = '0') and the slave mode (I2Sn_CNTREG:MSMD = '0').
- The next frame synchronous signal is received during frame transmission/reception in the slave mode (I2Sn_CNTREG:MSMD), not free-running mode (I2Sn_CNTREG:FRUN = 1).

When FERR is '1' and I2Sn_INTCNT:FERRM is '0', interrupt to CPU occurs.

Writing '1' from CPU clears the value to '0'.

This bit becomes '0' by software reset.

[bit28] TXUDR1 : Tx FIFO Underflow Error

When transmission FIFO underflows at the start of frame, the value is set to '1'.

When TXUDR1 is '1' and I2Sn_INTCNT:TXUD1M is '0', interrupt to the CPU occurs.

Writing '1' from CPU clears the value to '0'.

This bit becomes '0' by software reset.

Note:

- *When the transmission FIFO underflows at the start of frame, there will be no more attempts to read the transmission FIFO during the rest of the frame, so if the transmission FIFO is not written after it has underflown, I2Sn_STATUS:TXUDR0 is not set to '1'.*

[bit27] TXUDR0 : Tx FIFO Underflow Error

When transmission FIFO underflows during frame transmission (from 2nd bit word to the last frame of the word), the value is set to '1'.

When TXUDR0 is '1' and I2Sn_INTCNT:TXUD0M is '0', interrupt to the CPU occurs.

Writing '1' from CPU clears the value to '0'.

This bit becomes '0' by software reset.

Note:

- *When the transmission FIFO underflows during frame transmission and is still empty at the start of the next frame, I2Sn_STATUS:TXUDR1 is set to '1' too.*

[bit26] TXOVR : Tx FIFO Overflow Error

When transmission FIFO overflows, the value is set to '1' indicating transmission data is written in the condition that transmission FIFO is full. The value '1' indicates 1 word or more of transmission data is ignored.

When TXOVR is '1' and I2Sn_INTCNT:TXOVM is '0', interrupt to CPU occurs.

Writing '1' from CPU clears the value to '0'.

This bit becomes '0' by software reset.

[bit25] RXUDR : Rx FIFO Underflow Error

When reception FIFO underflows, the value is set to '1' indicating read access is carried out to reception FIFO in the condition that reception FIFO is empty.

When RXUDR is '1' and I2Sn_INTCNT:RXUDM is '0', interrupt to the CPU occurs. Writing '1' from CPU clears the value to '0'.

This bit becomes '0' by software reset.

Note:

- *This flag is not set when the DAP controller reads the Reception FIFO Data Registers while the reception FIFO is empty.*

[bit24] RXOVR : Rx FIFO Overflow Error

When reception FIFO overflows, the value is set to '1' indicating reception is carried out in the condition that reception FIFO is full. The value '1' indicates 1 word or more of reception data is ignored.

When RXOVR is '1' and I2Sn_INTCNT:RXOVM is '0', interrupt to CPU occurs.

Writing '1' from CPU clears the value to '0'.

This bit becomes '0' by software reset.

[bit23:20] read0 : -

[bit19] EOPI : Interrupt Flag for Rx Timer

This is an interrupt flag that is triggered when an internal reception timer times out. The reception timer is enabled when following conditions are met at the same time:

I2Sn_CNTREG:RXDIS is set to '0'.

I2Sn_OPRREG:START bit is set to '1' and I2Sn_OPRREG:RXENB = '1'.

After the reset, operation starts with the 1st word reception.

The count value is automatically cleared if reception FIFO data is more than threshold or it becomes empty. When the reception FIFO is not empty and the number of data is less than or equal to the threshold, the reception timer is incremented with each internal bus clock cycle.

EOPI is set to '1' when reception FIFO is non-empty and reception timer count value reaches timeout set by I2Sn_INTCNT:RPTMR.

When EOPI is '1' and I2Sn_INTCNT:EOPM is '0', interrupt to CPU occurs.

Writing '1' from CPU clears the value to '0'.

This bit becomes '0' by software reset.

[bit18] BSY : Serial Tx Busy

Serial transmission control part is in busy state. This bit is not affected by software reset.

bit	Description
0	Serial transmission control part is in idle state
1	Serial transmission control part is in busy state

[bit17] TXFI : Tx FIFO Empty

When number of empty space of transmission FIFO is greater than the threshold set in I2Sn_INTCNT:TFTH, this bit is set to '1'.

This bit is '1' and I2Sn_INTCNT: interrupt to CPU occurs.

When number of empty slot of transmission FIFO becomes smaller or equal to the threshold by writing to Transmission FIFO Data Registers from CPU or DMAC, this bit is cleared automatically to '0'.

The value also become '0' when I2Sn_OPRREG:START bit is '0' or I2Sn_OPRREG:TXENB bit is '0'.

If software reset is performed at I2Sn_OPRREG:START bit = '1' and I2Sn_OPRREG:TXENB bit = '1', the value becomes '0' during software reset then changes to '1' after the process.

[bit16] RXFI : Rx FIFO Full

When number of reception FIFO data becomes more than the threshold set in I2Sn_INTCNT:RFTH, this bit is set to '1'.

This bit is '1' and I2Sn_INTCNT: interrupt to CPU occurs.

When number of data in reception FIFO becomes smaller or equal to the threshold by reading Reception FIFO DATA Registers from CPU or DMAC, this bit is automatically cleared to '0'.

This bit becomes '0' by software reset.

[bit15:8] TXNUM : Number of Tx FIFO Data

The number of data in transmission FIFO is indicated.

This field is incremented by write access to Transmission FIFO Data Registers and decremented by serial word transmission.

Max. value of 66 can be displayed in the simultaneous transfer mode and value of 132 in the transmission only mode.

This field becomes '00000000' by software reset.

[bit7:0] RXNUM : Number of Data in Rx FIFO

The number of data in reception FIFO is indicated.

This field is incremented by word reception from serial bus and decremented by read access to Reception FIFO Data Registers.

Maximum value of 66 can be displayed in the simultaneous transfer mode and value of 132 in the reception mode.

This field becomes '00000000' by software reset.

4.11. DMA Activate Register (I2Sn_DMAACT)

This register is to enable or disable DMA control function.

REGISTER_NAME	I2Sn_DMAACT
OFFSET	0xA0
ACCESS_SIZE	B H W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

DMA Activate Register (I2Sn_DMAACT)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	read0	read0	read0	read0	read0	read0	read0	read0
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	read0	read0	read0	read0	read0	read0	read0	TDMACT
ACCESS_TYPE	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	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	read0	read0	read0	read0	read0	read0	read0	read0
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	read0	read0	read0	read0	read0	read0	read0	RDMACT
ACCESS_TYPE	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R/W
PROT_TYPE								
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:24] read0 : -

[bit23:17] read0 : -

[bit16] TDMACT : Tx DMA Control

The DMA transmission channel is activated.

After transmission channel starts, software should write '1' to TDMACT for transmit DMA channel to be active. When TDMACT is '0', transfer request of transmission channel block is not sent to DMAC.

Writing '0' from CPU clears the value to '0'.

This bit becomes '0' by software reset.

bit	Description
0	DMA transmission channel is disabled
1	DMA transmission channel is enabled

Clearing TDMACT also clears write transmission request.

[bit15:8] read0 : -

[bit7:1] read0 : -

[bit0] RDMACT : Rx DMA Control

The DMA reception channel is activated.

After reception channel starts, software should write '1' to RDMACT for receive DMA channel to be active. When RDMACT is '0', transfer request of reception channel block is not sent to DMAC.

Writing '0' from CPU clears the value to '0'.

This bit becomes '0' by software reset.

bit	Description
0	DMA reception channel is disabled
1	DMA reception channel is enabled

Clearing RDMACT also clears reception transfer request.

4.12. Debug Register (I2Sn_DEBUG)

This register is to enable or disable debug function.

REGISTER_NAME	I2Sn_DEBUG
OFFSET	0xA4
ACCESS_SIZE	B H W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

Debug Register (I2Sn_DEBUG)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	read0	read0	read0	read0	read0	read0	read0	read0
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	read0	read0	read0	read0	read0	read0	read0	read0
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	15	14	13	12	11	10	9	8
BIT_NAME	read0	read0	read0	read0	read0	read0	read0	read0
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	read0	read0	read0	read0	read0	read0	read0	DBGE
ACCESS_TYPE	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R/W
PROT_TYPE								
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:8] read0 : -

[bit7:1] read0 : -

[bit0] DBGE : Debug Enable (DBGE)

This bit is used to enable/disable debug mode for I2S.

bit	Description
0	Debug mode disabled
1	Debug mode enabled

This bit takes effect only in master mode (i.e. I2Sn_CNTREG:MSMD = '1').

When DBGE is set to '1' and the processor is in debug state and I2S is working as a master, then the serial interface is halted by stopping the activity on the SCK output. The activity on the serial clock resumes either when the processor leaves debug state or DBGE is set to '0'.

4.13. Module ID Register (I2Sn_MIDREG)

This register implements unique module identification number. Refer to the device datasheet for the module identification number of I2S module in the device.

REGISTER_NAME	I2Sn_MIDREG
OFFSET	0xA8
ACCESS_SIZE	B H W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

Module ID Register (I2Sn_MIDREG)

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	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	MID[23]	MID[22]	MID[21]	MID[20]	MID[19]	MID[18]	MID[17]	MID[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

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	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	MID[7]	MID[6]	MID[5]	MID[4]	MID[3]	MID[2]	MID[1]	MID[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	1

[bit31:0] MID : Module ID

This read-only register gives the unique module identification number of I2S module. The unique Module ID number identifies the version of the I2S module used in the MCU. Refer to the device specific datasheet for the module identification number of its I2S.

CHAPTER 25: Programmable CRC

This chapter explains the function and operation of the Programmable CRC.

1. Overview
2. Configuration and Block Diagram
3. Operation of the Programmable CRC
4. Registers

CODE: PRGCRC-S6J3300-E1

1. Overview

This section describes the features and the block diagram of the Programmable CRC.

Features of Programmable CRC

The Programmable CRC is a software configurable module with serial CRC calculation logic as hardware implementation. The serial CRC logic works on modulo-2 arithmetic for calculation of checksum. The CRC module can detect errors in data blocks by calculating a checksum.

- Programmable 8-, 16-, 24-, or 32-bit input data width
- Programmable polynomial value (polynomial degree from 2 to 32)
- Programmable initial seed value
- Programmable final checksum XOR value
- Interrupt and DMA trigger capability
- Configurable input/output bit reflection and byte swapping. This facilitates the different settings of common CRC standards and the handling of data organized in little or big endian format
- Supports block/multiple data transfers (more than 32-bit)

Areas of Application

- Data security/integrity
- Communication protocols

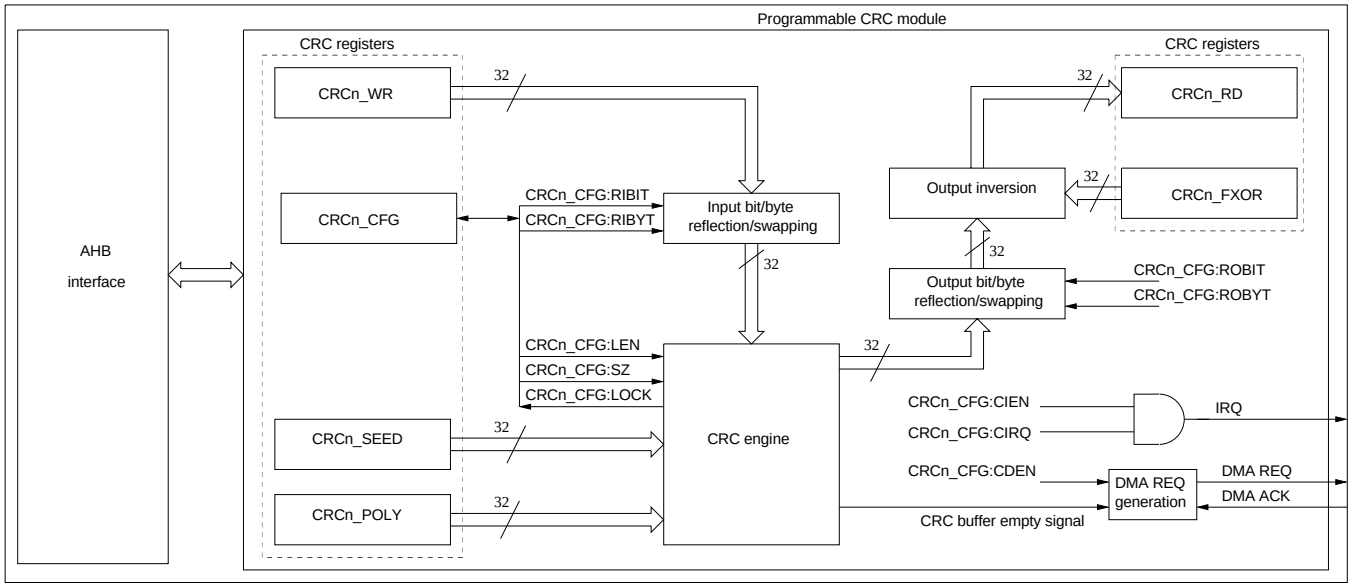
Programmable CRC module can be configured to widely used common CRC standards, some of them are listed below:

- CRC-32-IEEE 802.3
- CRC-16-CCITT
- CRC-8-CCITT
- CRC-5-USB
- CRC-XMODEM
- 12-bit CRC
- 10-bit CRC
- 8-bit CRC

2. Configuration and Block Diagram

This section shows a block diagram of Programmable CRC

Figure 25-1 Block Diagram of Programmable CRC



3. Operation of the Programmable CRC

This section describes operation of Programmable CRC in detail.

For more details on flowcharts for CRC operation see Section 3.1, on CRC calculation flow see Section 3.2, and on an example for CRC calculation see Section 3.3.

3.1. CRC Operation Flowcharts

The flowcharts Figure 25-2, Figure 25-3, and Figure 25-4 show the steps to configure CRC registers and to perform a CRC calculation.

Figure 25-2 Polling Based CRC Operation

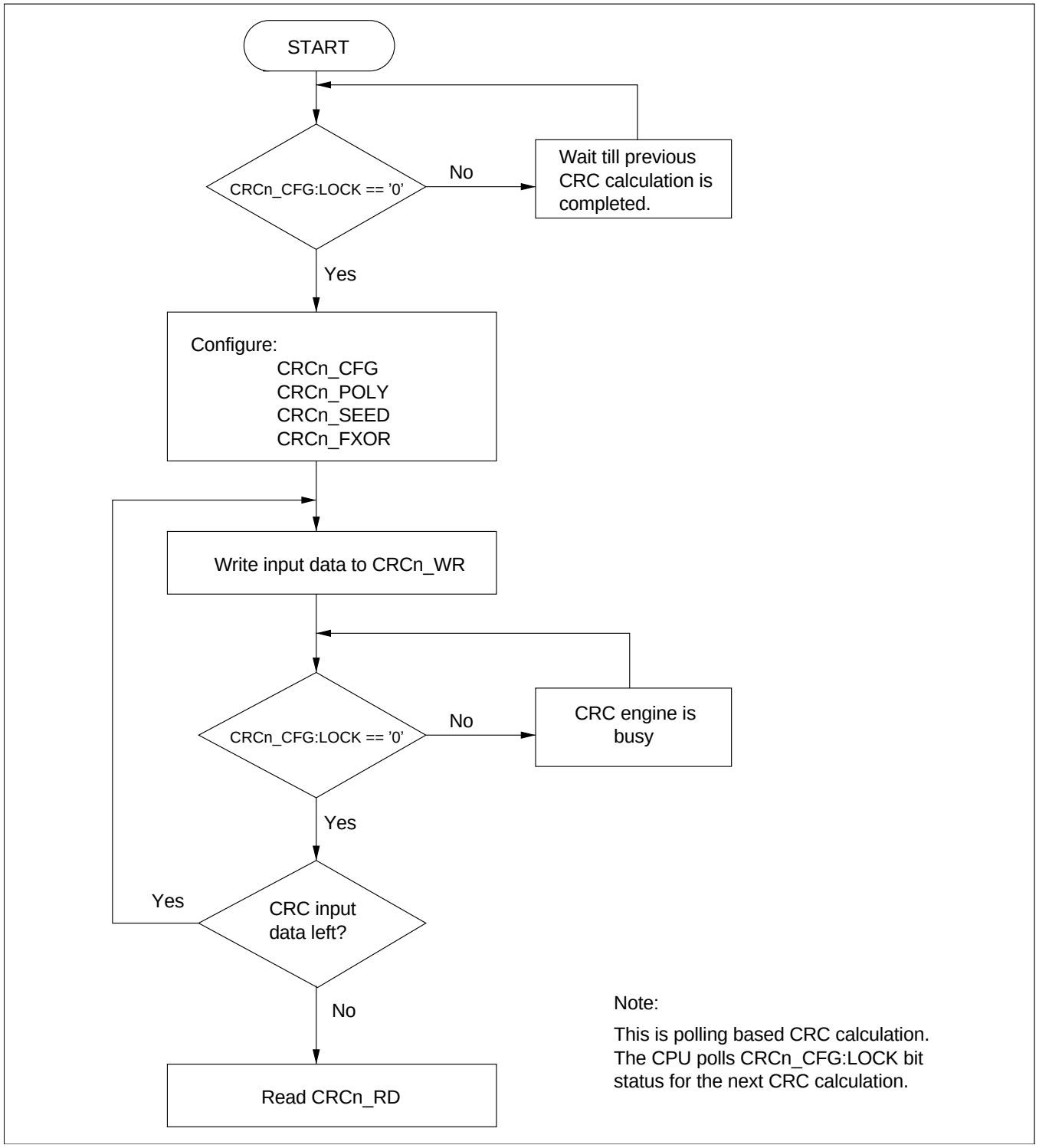


Figure 25-3 CRC Operation with IRQ

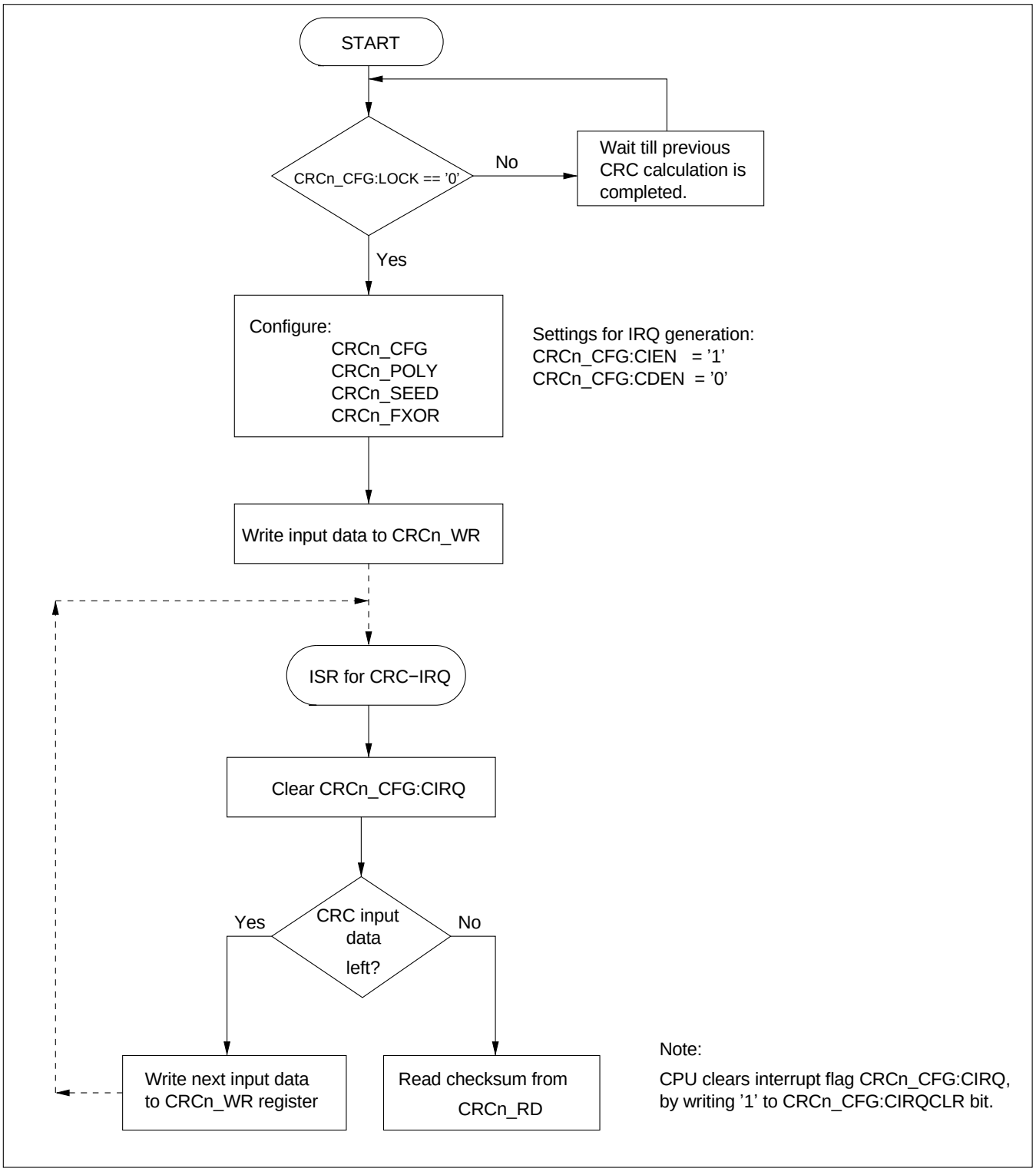
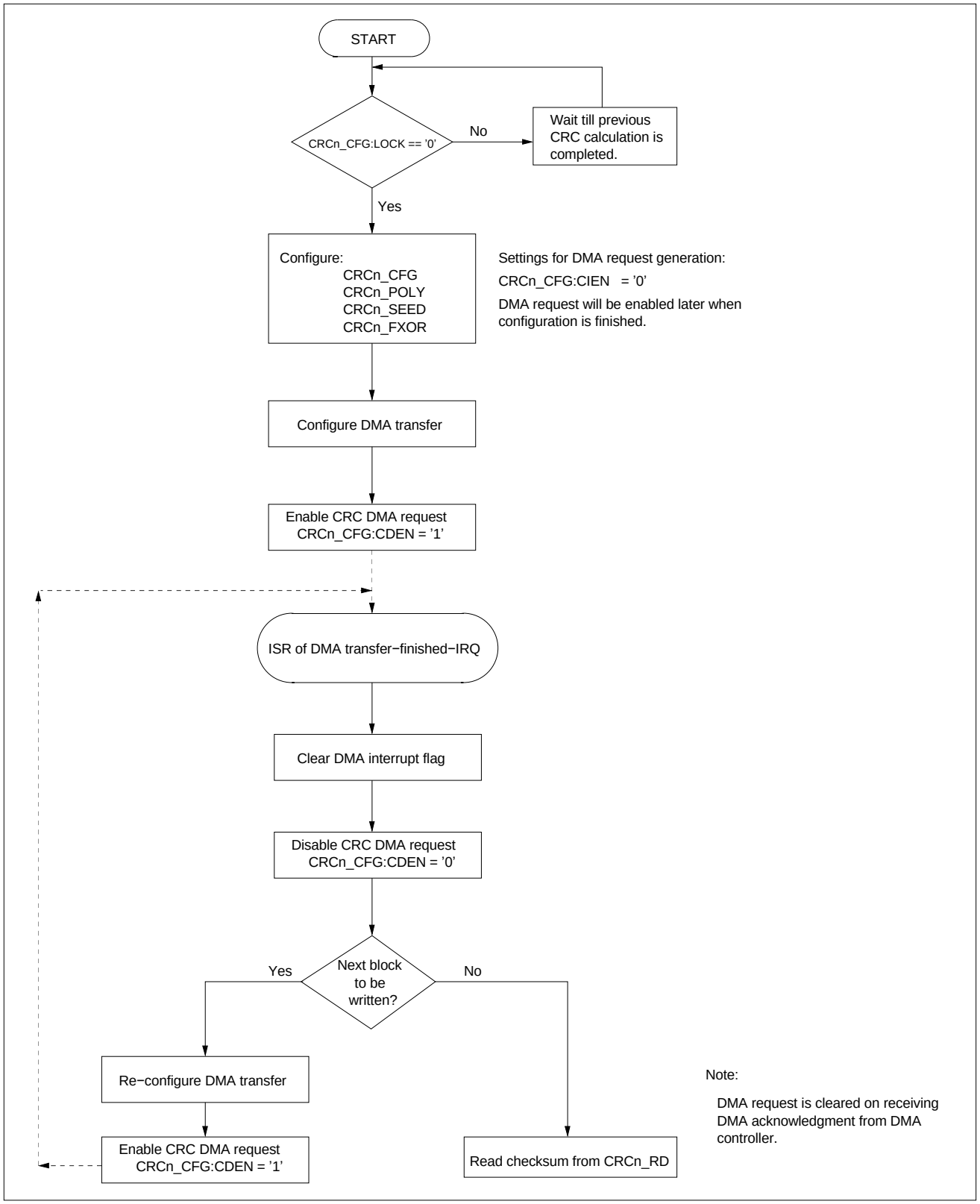
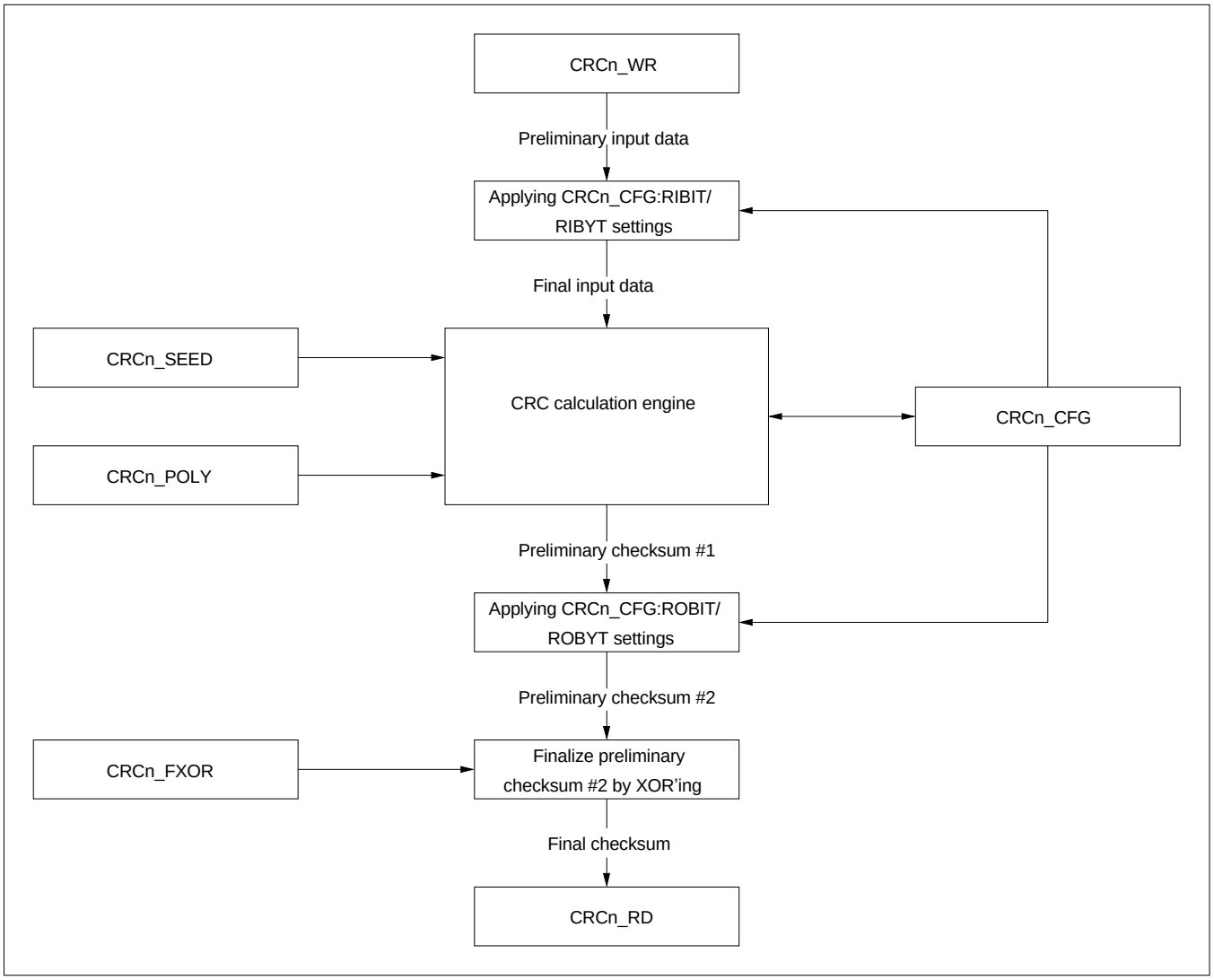


Figure 25-4 CRC Operation with DMA Request



3.2. CRC Input Data and Checksum Calculation Flow

Figure 25-5 Block Diagram of CRC Input Data and Checksum Calculation Flow



1. The input data for which CRC is to be calculated is written to CRCn_WR register. This is the 'preliminary input data'.
2. The 'preliminary input data' bytes can be swapped/reflected bit-wise using CRCn_CFG:RIBIT and/or byte-wise using CRCn_CFG:RIBYT before they enter the CRC engine. The settings are shown below:

   'preliminary input data' in CRCn_WR register:
   A7----A0 B7----B0 C7----C0 D7----D0
   If the input data size is less than 32-bit (SZ < '11'), then the remaining bits (8-,16-, or 24-bit) of the data are considered as don't care (X) as shown in below table.

Table 25-1 Preliminary Input Data Bit-Wise and/or Byte-Wise Reflection/Swapping

RIBYT	RIBIT	SZ	Final Input Data for CRC Engine			
			+3	+2	+1	+0
‘0’	‘0’	‘00’	XXXX XXXX	XXXX XXXX	XXXX XXXX	D7------- D0
		‘01’	XXXX XXXX	XXXX XXXX	C7--------C0	D7--------D0
		‘10’	XXXX XXXX	B7--------B0	C7--------C0	D7--------D0
		‘11’	A7--------A0	B7--------B0	C7--------C0	D7--------D0
	‘1’	‘00’	XXXX XXXX	XXXX XXXX	XXXX XXXX	D0--------D7
		‘01’	XXXX XXXX	XXXX XXXX	C0--------C7	D0--------D7
		‘10’	XXXX XXXX	B0--------B7	C0--------C7	D0--------D7
		‘11’	A0---------A7	B0--------B7	C0--------C7	D0--------D7
‘1’	‘0’	‘00’	XXXX XXXX	XXXX XXXX	XXXX XXXX	D7--------D0
		‘01’	XXXX XXXX	XXXX XXXX	D7--------D0	C7--------C0
		‘10’	XXXX XXXX	D7--------D0	C7--------C0	B7--------B0
		‘11’	D7---------D0	C7--------C0	B7--------B0	A7--------A0
	‘1’	‘00’	XXXX XXXX	XXXX XXXX	XXXX XXXX	D0--------D7
		‘01’	XXXX XXXX	XXXX XXXX	D0--------D7	C0--------C7
		‘10’	XXXX XXXX	D0--------D7	C0--------C7	B0--------B7
		‘11’	D0---------D7	C0--------C7	B0--------B7	A0--------A7

3. The 'preliminary input data' after applying the settings of CRCn_CFG:RIBIT/RIBYT results in the 'final input data', which is sent to the CRC engine for checksum calculation.
4. The CRCn_SEED register provides the initial value to the CRC engine. The required polynomial is provided by CRCn_POLY register. The CRC engine starts its operation once CRCn_WR register is written with the input data.
5. CRC engine performance: The performance of CRC engine for CRC checksum calculation is based on the input data size and number of clock cycles (bus clock) required to complete a calculation. The shows number of clock cycles required to get final checksum at CRCn_RD register with respect to input data size.

Table 25-2 Clock Cycles Requirement for Checksum Calculation

Input Data Size	Number of Clock Cycles Required for Final Checksum at CRCn_RD
8-bit	Input data size (8-bit) + 2 = 10 clock cycles.
16-bit	Input data size (16-bit) + 2 = 18 clock cycles.
24-bit	Input data size (24-bit) + 2 = 26 clock cycles.
32-bit	Input data size (32-bit) + 2 = 34 clock cycles.

6. The 'preliminary checksum #1' bytes can be swapped/reflected bit-wise using CRCn_CFG:ROBIT and/or byte-wise using CRCn_CFG:ROBYT. shows at which positions the checksum bits of 'preliminary checksum #1' S[(LEN-1):0] will be located in 'preliminary checksum #2' after CRCn_CFG:ROBIT/ROBYT settings have been applied. Only some examples of different CRCn_CFG:LEN configurations are shown.

Note:

- *Only some examples for CRCn_CFG:LEN are shown.*

Table 25-3 Preliminary Checksum #1 Bit-Wise and/or Byte-Wise Reflection/Swapping

ROBYT	ROBIT	LEN	Preliminary Checksum #2				
			+3	+2	+1	+0	Action
‘0’	‘0’	32	S31---S24	S23---S16	S15---S8	S7---S0	No swapping/reflection. The checksum is aligned with the polynomial degree/length.
		21	‘0000 0000’	‘000 S20---S16’	S15---S8	S7---S0	No swapping/reflection. The checksum is aligned with the polynomial degree/length. The bits S21 to S31 are ’0’.
		16	‘0000 0000’	‘0000 0000’	S15---S8	S7---S0	No swapping/reflection. The checksum is aligned with the polynomial degree/length. The bits S16 to S31 are ’0’.
		3	‘0000 0000’	‘0000 0000’	‘0000 0000’	‘00000 S2--S0’	No swapping/reflection. The checksum is aligned with the polynomial degree/length. The bits S3 to S31 are ’0’.
	‘1’	32	S24---S31	S16---S23	S8---S15	S0---S7	Byte aligned checksum reflection.
		21	‘0000 0000’	‘S16---S20 000’	S8---S15	S0---S7	Bit reflection. The checksum is byte aligned. Bit S21-S23 and S24-S31 are ‘0’.
		16	‘0000 0000’	‘0000 0000’	S8---S15	S0---S7	Bit reflection. The checksum is byte aligned. Bit S16-S31 are ‘0’.
		3	‘0000 0000’	‘0000 0000’	‘0000 0000’	‘S0---S2 00000’	Bit reflection. The checksum is byte aligned. Bit S3-S7 and S8-S31 are ‘0’.
‘1’	‘0’	32	S7---S0	S15---S8	S23---S16	S31---S24	Byte aligned checksum swapping.
		21	‘0000 0000’	S7---S0	S15---S8	‘000 S20---S16’	Byte swapping. The checksum is byte aligned. Bit S21-S23 and S24-S31 are ‘0’.
		16	‘0000 0000’	‘0000 0000’	S7---S0	S15---S8	Byte aligned checksum swapping. Bit S16-S31 are ‘0’.
		3	‘0000 0000’	‘0000 0000’	‘0000 0000’	‘00000 S2_S0’	No byte swapping. Bit S3-S7 and S8-S31 are ‘0’.
	‘1’	32	S0---S7	S8---S15	S16---S23	S24---S31	Bit reflection and byte swapping aligned with polynomial degree/length.
		21	‘0000 0000’	‘000 S0---S4’	S5---S12	S13---S20	Bit reflection and byte swapping. The checksum is aligned with polynomial length/degree. Bit S21-S23 and S24-S31 are ‘0’.
		16	‘0000 0000’	‘0000 0000’	S0---S7	S8---S15	Bit reflection and byte swapping. The checksum is aligned with polynomial length/degree. Bit S16-S31 are ‘0’.
		3	‘0000 0000’	‘0000 0000’	‘0000 0000’	‘00000 S0---S2’	Bit reflection and byte swapping. The checksum is aligned with polynomial length/degree. Bit S3-S7 and S8-S31 are ‘0’.

7. The checksum after applying settings of CRCn_CFG:ROBIT/ROBYT is 'preliminary checksum #2'.
8. The 'preliminary checksum #2' is XOR’ed with the contents of CRCn_FXOR register to get the 'final checksum'.
9. The 'final checksum' gets available at CRCn_RD register.

3.3. CRC Calculation Example

Consider the following values for calculating 8-bit CRC checksum value.

Input data = 0x0F (Hex)

Polynomial = $x^8 + x^2 + x + 1$

Seed = 0xFF (Hex)

Final XOR = 0x00 (Hex)

The coefficients of the polynomial are arranged in .

Table 25-4 Coefficients of the Polynomial

x8	x7	x6	x5	x4	x3	x2	x1	x0
1	0	0	0	0	0	1	1	1

The highest order coefficient x8 provides the degree of the CRC polynomial and the checksum length, respectively. It must not be set to '1' while configuring CRCn_POLY register, instead CRCn_CFG:LEN should be configured (here it is CRCn_CFG:LEN = 8). Therefore, the value of the polynomial that is written to CRCn_POLY register in accordance with above coefficients is 0x07 (Hex).

The input/output bit reflection is disabled in this example.

The Programmable CRC registers should be configured as follows for the given values:

The CRC configuration register is configured by considering 8-bit input data size and 8-bit polynomial/checksum length as follows.

CRCn_CFG = 0x00080000 (Hex)

CRCn_POLY = 0x00000007 (Hex)

CRCn_SEED = 0x000000FF (Hex)

CRCn_FXOR = 0x00000000 (Hex)

CRCn_WR = 0x0000000F (Hex)

The final result of CRC checksum calculation is 0xDE (Hex), which gets available after 10 clock cycles (once CRCn_WR is written) in the CRCn_RD register.

If another input data is given to the CRC module, then 'preliminary checksum #1' (0xDE) is used as the initial seed value.

If the new CRC calculation should start from the seed value instead of from the last CRC result, then the CRCn_SEED register needs to be re-written (even if it is the same seed value as before).

4. Registers

All Programmable CRC registers are explained in this section.

The suffix 'n' in the register name indicates that the register is an instance 'n' of the module.

Registers of Programmable CRC

The following registers are available for each instance of Programmable CRC:

- CRC Polynomial Register (CRCn_POLY)
- CRC Seed Register (CRCn_SEED)
- CRC Final XOR Register (CRCn_FXOR)
- CRC Configuration Register (CRCn_CFG)
- CRC Write Register (CRCn_WR)
- CRC Read Register (CRCn_RD)

Memory Layout of Programmable CRC Registers

Figure 25-6 Memory Layout of Programmable CRC Registers

Offset	REGISTER_NAME				ACCESS_SIZE
	+3	+2	+1	+0	
0x00000000	CRCn_POLY 00000100 11000001 00011101 10110111				B, H, W
0x00000004	CRCn_SEED 11111111 11111111 11111111 11111111				B, H, W
0x00000008	CRCn_FXOR 11111111 11111111 11111111 11111111				B, H, W
0x0000000C	CRCn_CFG 00000000 11100000 00000000 00000000				B, H, W
0x00000010	CRCn_WR 00000000 00000000 00000000 00000000				B, H, W
0x00000014	CRCn_RD 00000000 00000000 00000000 00000000				B, H, W

4.1. CRC Polynomial Register (CRCn_POLY)

The CRC Polynomial Register (CRCn_POLY) defines the polynomial value for the CRC checksum calculation.

REGISTER_NAME	YOURPERIn_CSRi
OFFSET	0 + i*4
ACCESS_SIZE	B H W
MULTIPLE	0:15
NUMERIC_TYPE	
OTHER	

CRC Polynomial Register (CRCn_POLY)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	POLY[31]	POLY[30]	POLY[29]	POLY[28]	POLY[27]	POLY[26]	POLY[25]	POLY[24]
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	Rp/Wp							
INITIAL_VALUE	0	0	0	0	0	1	0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	POLY[23]	POLY[22]	POLY[21]	POLY[20]	POLY[19]	POLY[18]	POLY[17]	POLY[16]
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	Rp/Wp							
INITIAL_VALUE	1	1	0	0	0	0	0	1

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	POLY[15]	POLY[14]	POLY[13]	POLY[12]	POLY[11]	POLY[10]	POLY[9]	POLY[8]
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	Rp/Wp							
INITIAL_VALUE	0	0	0	1	1	1	0	1

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	POLY[7]	POLY[6]	POLY[5]	POLY[4]	POLY[3]	POLY[2]	POLY[1]	POLY[0]
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	Rp/Wp							
INITIAL_VALUE	1	0	1	1	0	1	1	1

[bit31:0] POLY[31:0] : CRC Polynomial

The CRCn_POLY contains the CRC polynomial. The degree of the polynomial must be between 2 to 32. Initial value is the common CRC-32 polynomial 0x04C11DB7 (x32 + x26 + x23 + x22 + x16 + x12 + x11 + x10 + x8 + x7 + x5 + x4 + x2 + x + 1). The highest degree of coefficient should be used to configure polynomial/checksum length (CRCn_CFG:LEN).

Note:

- *If the polynomial length as defined by CRCn_CFG:LEN is less than 32-bit, then the upper bits [31:LEN] must be written to '0' by the programmer. The highest order degree must not be set to '1' while configuring CRCn_POLY register, as it is implicitly defined by CRCn_CFG:LEN.*

4.2. CRC Seed Register (CRCn_SEED)

The CRC Seed Register (CRCn_SEED) defines the initial value for the CRC checksum calculation.

REGISTER_NAME	YOURPERIn_CSRi
OFFSET	0 + i*4
ACCESS_SIZE	B H W
MULTIPLE	0:15
NUMERIC_TYPE	
OTHER	

CRC Seed Register (CRCn_SEED)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	SEED[31]	SEED[30]	SEED[29]	SEED[28]	SEED[27]	SEED[26]	SEED[25]	SEED[24]
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	Rp/Wp							
INITIAL_VALUE	1	1	1	1	1	1	1	1

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	SEED[23]	SEED[22]	SEED[21]	SEED[20]	SEED[19]	SEED[18]	SEED[17]	SEED[16]
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	Rp/Wp							
INITIAL_VALUE	1	1	1	1	1	1	1	1

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	SEED[15]	SEED[14]	SEED[13]	SEED[12]	SEED[11]	SEED[10]	SEED[9]	SEED[8]
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	Rp/Wp							
INITIAL_VALUE	1	1	1	1	1	1	1	1

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	SEED[7]	SEED[6]	SEED[5]	SEED[4]	SEED[3]	SEED[2]	SEED[1]	SEED[0]
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	Rp/Wp							
INITIAL_VALUE	1	1	1	1	1	1	1	1

[bit31:0] SEED[31:0] : CRC SEED

CRCn_SEED contains the initial value for checksum calculation. If the seed value is not initialized for the next operation (even if the seed value is identical to the previous operation), then calculation is continued from the last state with the current checksum value as initial value. Initial value for seed register is 0xFFFFFFFF.

Note:

- *The CRCn_SEED register should be configured with respect to the polynomial length (CRCn_CFG:LEN). If the polynomial length is less than 32-bit, then the upper bits [31:LEN] must be written to '0' by the programmer.*

4.3. CRC Final XOR Register (CRCn_FXOR)

The CRC Final XOR register (CRCn_FXOR) contains the values to be XOR'ed with the preliminary checksum to finalize the CRC calculation

REGISTER_NAME	YOURPERIn_CSRi
OFFSET	0 + i*4
ACCESS_SIZE	B H W
MULTIPLE	0:15
NUMERIC_TYPE	
OTHER	

CRC Final XOR Register (CRCn_FXOR)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	FXOR[31]	FXOR[30]	FXOR[29]	FXOR[28]	FXOR[27]	FXOR[26]	FXOR[25]	FXOR[24]
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	Rp/Wp							
INITIAL_VALUE	1	1	1	1	1	1	1	1

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	FXOR[23]	FXOR[22]	FXOR[21]	FXOR[20]	FXOR[19]	FXOR[18]	FXOR[17]	FXOR[16]
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	Rp/Wp							
INITIAL_VALUE	1	1	1	1	1	1	1	1

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	FXOR[15]	FXOR[14]	FXOR[13]	FXOR[12]	FXOR[11]	FXOR[10]	FXOR[9]	FXOR[8]
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	Rp/Wp							
INITIAL_VALUE	1	1	1	1	1	1	1	1

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	FXOR[7]	FXOR[6]	FXOR[5]	FXOR[4]	FXOR[3]	FXOR[2]	FXOR[1]	FXOR[0]
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	Rp/Wp							
INITIAL_VALUE	1	1	1	1	1	1	1	1

[bit31:0] FXOR [31:0] : CRC XOR Data

The contents of CRCn_FXOR register are XOR'ed with the preliminary checksum data (after CRCn_CFG:ROBIT/ROBYT settings have been applied) and then the final checksum is moved to CRCn_RD register. Initial value for final XOR register is 0xFFFFFFFF.

Notes:

- *The bits of this register affect the corresponding bits of the CRCn_RD register. Therefore, the bits not belonging to the checksum should be written to '0'. For the position of the checksum bits depending on the used output bit/byte reflection refer to Table 25-3.*

4.4. CRC Configuration Register (CRCn_CFG)

The CRC Configuration Register (CRCn_CFG) is used to set the operation mode of the CRC module. CRCn_CFG register describes the polynomial/checksum length, input data size, and input/output bit/byte reflection. It indicates the status of CRC operation. Interrupt and DMA requests are also configured using CRCn_CFG register.

REGISTER_NAME	YOURPERIn_CSRi
OFFSET	0 + i*4
ACCESS_SIZE	B H W
MULTIPLE	0:15
NUMERIC_TYPE	
OTHER	

CRC Configuration Register (CRCn_CFG)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	read0	read0	read0	LOCK	read0	CDEN	CIEN	CIRQ
ACCESS_TYPE	R0,WX	R0,WX	R0,WX	R,WX	R0,WX	R/W	R/W	R,WX
PROT_TYPE	Rp/Wp							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	SZ[1]	SZ[0]	LEN[5]	LEN[4]	LEN[3]	LEN[2]	LEN[1]	LEN[0]
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	Rp/Wp							
INITIAL_VALUE	1	1	1	0	0	0	0	0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	read0	read0	read0	read0	RIBIT	RIBYT	ROBIT	ROBYT
ACCESS_TYPE	R0,WX	R0,WX	R0,WX	R0,WX	R/W	R/W	R/W	R/W
PROT_TYPE	Rp/Wp							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	read0	read0	read0	read0	read0	read0	read0	CIRQCLR
ACCESS_TYPE	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,W
PROT_TYPE	Rp/Wp							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:29] read0 : -

[bit28] LOCK : CRC Engine Status bit

This bit indicates the status of the CRC engine.

bit	Description
0	CRC engine is ready, new data can be written to CRC registers
1	CRC engine is busy and writing to CRC registers is not possible. If the data is written to the CRC registers when LOCK bit is '1', then an error response is generated

[bit27] read0 : -

[bit26] CDEN : DMA Request Enable bit

This bit enables/disables the DMA request

bit	Description
0	Disable the DMA request
1	Enable the DMA request

DMA request is generated when CRC is in buffer empty state and CRCn_CFG:CDEN is set. DMA ISR should clear this bit after final transfer. Clearing this bit by CPU at any other time might lead to unwanted behavior.

[bit25] CIEN : CRC Interrupt Enable bit to CPU

This bit enables/disables the interrupt request.

bit	Description
0	Disable the interrupt request
1	Enable the interrupt request

The IRQ is triggered when CRCn_CFG:CIEN bit is enabled and CRC interrupt flag (CRCn_CFG:CIRQ) is set.

[bit24] CIRQ : CRC Interrupt Flag

This bit indicates the interrupt status of CRC.

bit	Description
0	No interrupt request Note: CPU clears interrupt flag by writing CRCn_CFG:CIRQCLR bit to '1':
1	Interrupt request

Checksum has been calculated by CRC engine and is available in CRCn_RD register.

[bit23:22] SZ[1:0] : CRC Input Data Size Configuration bits

These bits are used to configure the input data size as follows

bit[1:0]	Description
00	8-bit
01	16-bit
10	24-bit
11	32-bit

[bit21:16] LEN[5:0] : CRC Polynomial/Checksum Length Configuration bits

These bits are used to configure the length (degree) of CRC polynomial/checksum as follows:

bit[5:0]	Description
100000	32
011111	31
.....	.....
000010	2

Note:

- *The following settings are not supported:*
- *CRCn_CFG:LEN > 32*
- *CRCn_CFG:LEN < 2*

[bit15:12] read0 : -

[bit11] RIBIT : Reflect Input Bits

bit	Description
0	Disable input bit reflection
1	Enable input bit reflection

When the input data in the CRCn_WR register is passed to the CRC engine, the bit ordering of each byte within input data is reversed. For more details refer to Section 'Operation of the Programmable CRC'.

[bit10] RIBYT : Reflect Input Bytes

bit	Description
0	Disable input byte reflection (swapping)
1	Enable input byte reflection (swapping)

When the input data in the CRCn_WR register is passed to the CRC engine, the byte ordering of input data is reversed. Only the bytes of the configured input data size CRCn_CFG:SZ are affected. For more details refer to Section "Operation of the Programmable CRC'.

Note:

- *For 8-bit input data, this setting has no effect.*

[bit9] ROBIT : Reflect Output Bits

bit	Description
0	Disable output bit reflection
1	Enable output bit reflection

The bit ordering of each byte within checksum is reversed, before passing the checksum to final XOR'ing stage. For more details refer to Section "Operation of the Programmable CRC'.

[bit8] ROBYT : Reflect Output Bytes

bit	Description
0	Disable output byte reflection (swapping)
1	Enable output byte reflection (swapping)

The byte ordering of checksum data is reversed, before passing the byte aligned polynomial length of checksum data to final XOR'ing stage. For more details refer to Section "'Operation of the Programmable CRC".

Note:

- *For the checksum data less than or equal to 8 bits, this setting has no effect.*

[bit7:1] read0 : -

[bit0] CIRQCLR : Interrupt Clear

This bit clears the CRC interrupt flag.

bit	Description
0	Write '0' is ignored, reading this bit always returns '0'
1	Clear CRC interrupt flag (CRCn_CFG:CIRQ)

4.5. CRC Write Register (CRCn_WR)

The input data for the CRC checksum calculation must be written to the CRC Write Register (CRCn_WR).

REGISTER_NAME	YOURPERIn_CSRi
OFFSET	0 + i*4
ACCESS_SIZE	B H W
MULTIPLE	0:15
NUMERIC_TYPE	
OTHER	

CRC Write Register (CRCn_WR)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	CRCWR[31]	CRCWR[30]	CRCWR[29]	CRCWR[28]	CRCWR[27]	CRCWR[26]	CRCWR[25]	CRCWR[24]
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	Rp/Wp							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	CRCWR[23]	CRCWR[22]	CRCWR[21]	CRCWR[20]	CRCWR[19]	CRCWR[18]	CRCWR[17]	CRCWR[16]
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	Rp/Wp							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	CRCWR[15]	CRCWR[14]	CRCWR[13]	CRCWR[12]	CRCWR[11]	CRCWR[10]	CRCWR[9]	CRCWR[8]
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	Rp/Wp							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	CRCWR[7]	CRCWR[6]	CRCWR[5]	CRCWR[4]	CRCWR[3]	CRCWR[2]	CRCWR[1]	CRCWR[0]
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	Rp/Wp							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:0] CRCWR[31:0] : CRC Write Register

The CRCn_WR register contains the input data, for which the CRC checksum is to be calculated. Writing to this register starts the CRC calculation process. After pre-processing (bit/byte reflection/swapping) the contents of the CRCn_WR register are passed to the CRC engine (the contents of CRCn_SEED register are also provided). There it is divided by the content of CRCn_POLY register in modulo-2 arithmetic to get the final checksum after post-processing (bit/byte reflection/swapping and XOR'ing).

Note:

- *The size of input data is configured by CRCn_CFG:SZ, where 8, 16, 24, and 32 bits are only supported as data size. If the input data size is less than 32-bit (i.e. 8, 16, or 24 bits), then the invalid/unused bits are considered as don't care (X).*

4.6. CRC Read Register (CRCn_RD)

The CRC Read Register (CRCn_RD) contains the final checksum of the data written to the CRCn_WR register.

REGISTER_NAME	YOURPERIn_CSRi
OFFSET	0 + i*4
ACCESS_SIZE	B H W
MULTIPLE	0:15
NUMERIC_TYPE	
OTHER	

CRC Read Register (CRCn_RD)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	CRCRD[31]	CRCRD[30]	CRCRD[29]	CRCRD[28]	CRCRD[27]	CRCRD[26]	CRCRD[25]	CRCRD[24]
ACCESS_TYPE	R,W	R,W	R,W	R,W	R,W	R,W	R,W	R,W
PROT_TYPE	Rp/Wp							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	CRCRD[23]	CRCRD[22]	CRCRD[21]	CRCRD[20]	CRCRD[19]	CRCRD[18]	CRCRD[17]	CRCRD[16]
ACCESS_TYPE	R,W	R,W	R,W	R,W	R,W	R,W	R,W	R,W
PROT_TYPE	Rp/Wp							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	CRCRD[15]	CRCRD[14]	CRCRD[13]	CRCRD[12]	CRCRD[11]	CRCRD[10]	CRCRD[9]	CRCRD[8]
ACCESS_TYPE	R,W	R,W	R,W	R,W	R,W	R,W	R,W	R,W
PROT_TYPE	Rp/Wp							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	CRCRD[7]	CRCRD[6]	CRCRD[5]	CRCRD[4]	CRCRD[3]	CRCRD[2]	CRCRD[1]	CRCRD[0]
ACCESS_TYPE	R,W	R,W	R,W	R,W	R,W	R,W	R,W	R,W
PROT_TYPE	Rp/Wp							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:0] CRCRD[31:0]: CRC Read Data

The CRC Read Register contains the result (final checksum) of the CRC calculation after post-processing (applying CRCn_CFG:ROBIT, CRCn_CFG:ROBYT, and CRCn_FXOR settings). Writing any value on CRCn_RD register changes its content and it does not affect CRC calculation.

Note:

- *The polynomial length (CRCn_CFG:LEN) provides the length of the final checksum. If the polynomial length is less than 32-bit, then bits not belonging to the checksum are '0' (in case the invalid bits of the CRCn_FXOR register are not programmed to '0', then these bits in CRCn_RD register might also be '1'). The bit/byte reflection settings (CRCn_CFG:ROBIT/ROBYT) can influence the checksum in CRCn_RD register. For the position of the checksum bits refer to Table 25-3.*

CHAPTER 26: PCMPWM

This chapter explains the function and operation of the PCMPWM module

1. Overview
2. Configuration and Block Diagram
3. Operation of the PCMPWM
4. Registers

CODE: PCMPWM-S6J3300-E1

1. Overview

The PCMPWM module converts Pulse Code Modulated (PCM) data samples to Pulse Width Modulated (PWM) signals. The intended purpose of the module is to provide simple audio output capabilities.

Also refer to the chapter of "Sound System Configuration" on this manual how to configure the PCMPWM.

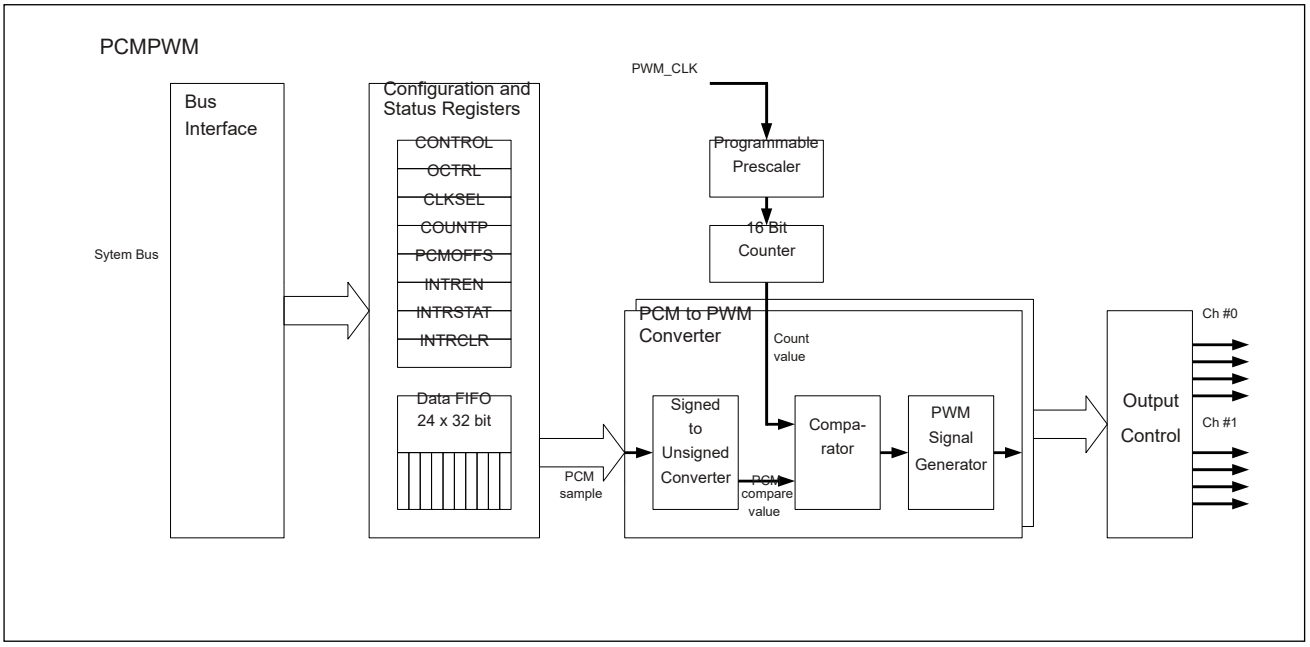
Features of the PCMPWM

- Two separate channels for stereo audio output
- Optional mono audio output mode
- A maximum output resolution of 16 bits
- An output resolution of 12 bits at a sampling frequency of 24.4 kHz (at 100 MHz PWM clock)
- An output resolution of 11 bits at s sampling frequency of 48.8 kHz (at 100 MHz PWM clock)
- Three modes of operation for different speaker interfacing:
  - Low-pass filter output mode
  - Simplified H-bridge output mode
  - Full H-bridge output mode
- Support for the DMA transfer of PCM data samples using DMA block transfer mode
- FIFO input buffer for PCM samples, with a depth 24
- Programmable clock divider for PWM cycle time
- Optional output of silence signals in debug mode and normal mode

2. Configuration and Block Diagram

2.1. Block Diagram

Figure 26-1 Block Diagram of the PCMPWM Module



Bus Interface
The Bus Interface connects the PCMPWM module's register file to the system bus.

Configuration and Status Registers
This block holds all the module's user accessible registers.

Data FIFO
This block represents a buffer for PCM samples and a boundary between the system bus clock and the PWM clock.

Clock Prescaler and 16-Bit Counter
These two blocks represent the free running counter and its prescaler used for the PCM to PWM conversion.

PCM to PWM Converters
These blocks represent the actual PCM to PWM conversion units. Each channel uses its own converter block.

Output Control
This block controls the PWM signal outputs. It e.g. defines the signals' output polarity and keeps unused outputs at their inactive levels.

2.2. Configuration of the PCMPWM Module

PCMPWM Enable

After reset the PCMPWM module is disabled. All PWM outputs are held at their inactive values. To start the PCM to PWM conversion and generate the PWM output signals, all configuration registers should be initialized and the global enable bit PCMPWMi_CONTROL:EN set. Once the module has been enabled, it must be ensured that the PCM data samples are provided to the FIFO buffer at the required rate. This may be done by either the CPU or the DMA unit.

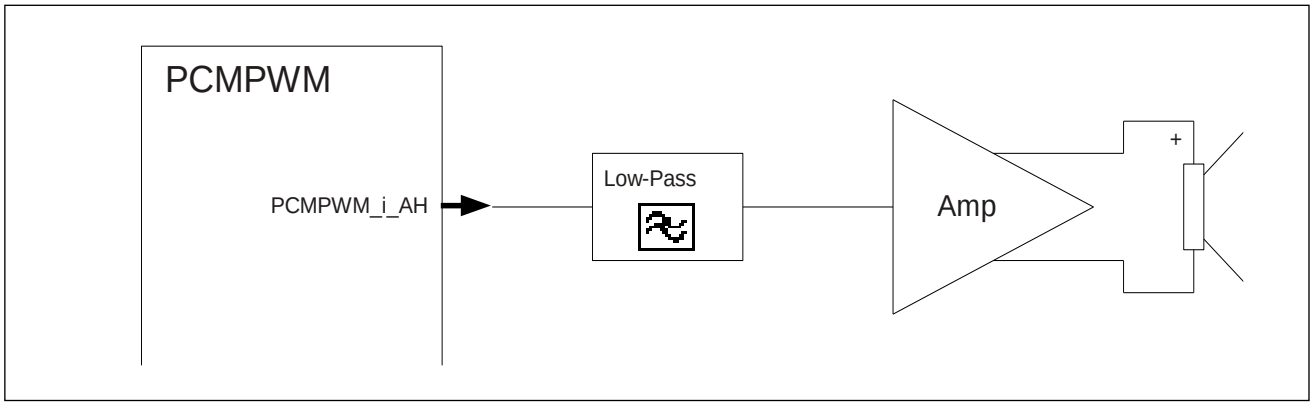
Operation Modes

The PCMPWM module supports three general modes of operation. Each mode is dedicated to a specific type of connection to the speaker. The operation mode is configured in the MODE bits of the CONTROL register (PCMPWMi_CONTROL:MODE). The following paragraphs describe these modes in detail.

1. Low-Pass Filter Mode

   In the low-pass filter mode only a single output is used. The output drives an audio amplifier with a low-pass filter connected in-between. Figure 26-2 depicts the corresponding circuit.

Figure 26-2 Simplified Schematic Illustration of Low Pass Filter Mode Output Circuit

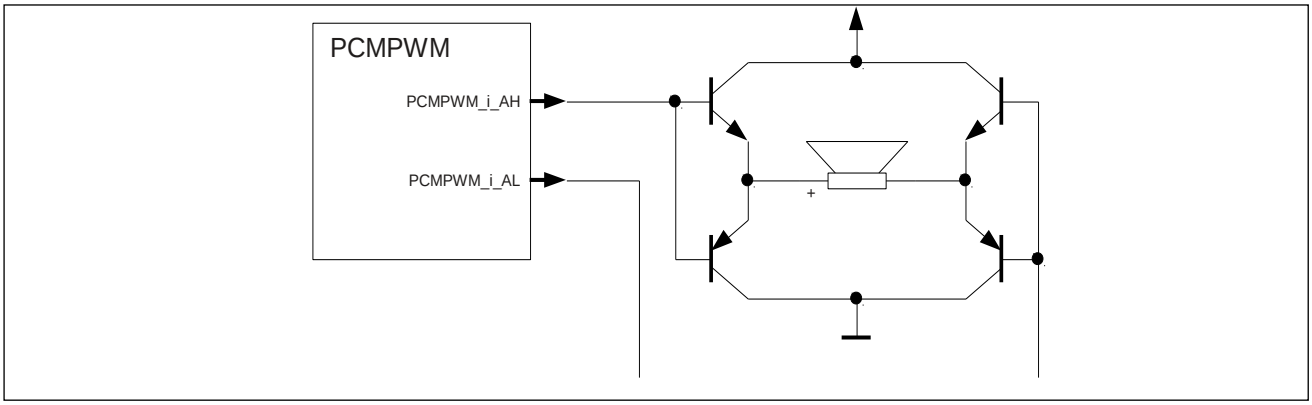


For positive PCM data samples an output signal with a duty cycle of more than 50% is generated. For negative PCM data samples, the signal has a duty cycle of less than 50%. A PCM data sample of zero generates an exact 50% duty cycle output signal. A 0% and a 100% duty cycle may be reached as well if the PWM count period and the PCM conversion offset are configured appropriately (registers PCMPWMi_COUNTP and PCMPWMi_PCMOFFS).

2. Simplified H-Bridge Mode

   In the simplified H-bridge mode two outputs are used to drive a speaker. A pair of complementary emitter followers is connected to each output. Figure 26-3 depicts the circuit for simplified H-bridge mode.

Figure 26-3 Simplified Schematic Illustration of Simplified H-Bridge Mode Output Circuit

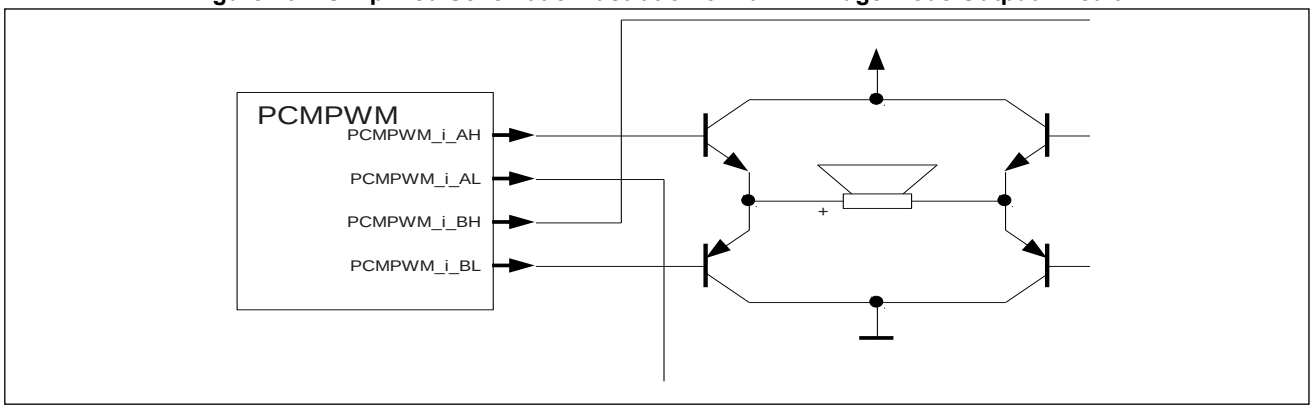


In simplified H-bridge mode, the signal at the PCMPWM_i_AL output is the inverted PCMPWM_i_AH signal.

3. Full H-Bridge Mode

   In the full H-bridge mode four outputs per speaker are used. All transistors of the H-bridge have individual driver outputs. Figure 26-4 depicts the circuit for full H-bridge mode.

Figure 26-4 Simplified Schematic Illustration of Full H-Bridge Mode Output Circuit

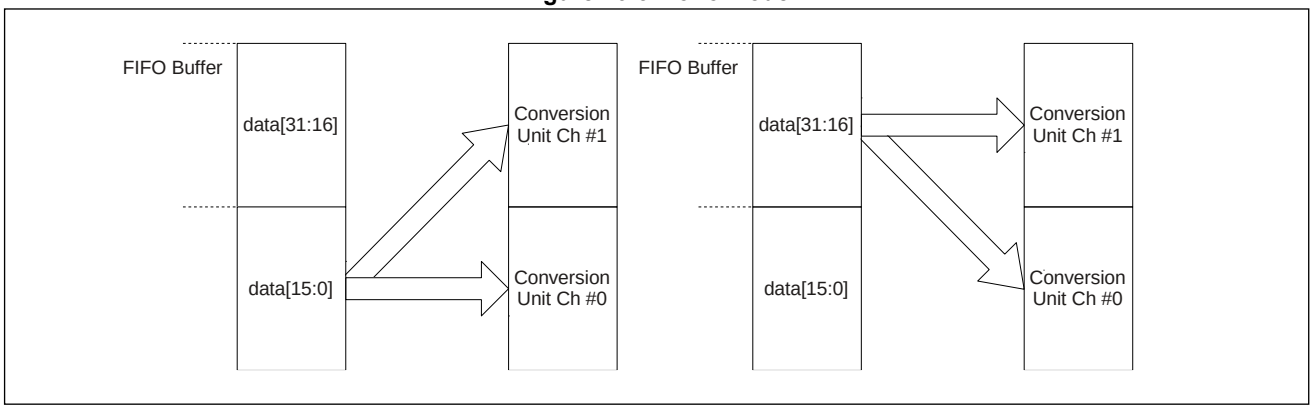


In full H-bridge mode the PCMPWM_i_AL signal is again the inverted PCMPWM_i_AH signal whereas the signal at the PCMPWM_i_BL output is the inverted PCMPWM_i_BH signal. The PCMPWM_i_Ax output pair is activated for positive PCM data samples. The PCMPWM_i_Bx output pair is activated for negative PCM data samples. If the PCM data sample is zero, no branch of the H-bridge is activated. The PCMPWM_i_Ax and PCMPWM_i_Bx branches are never active at the same time.

Mono or Stereo Mode

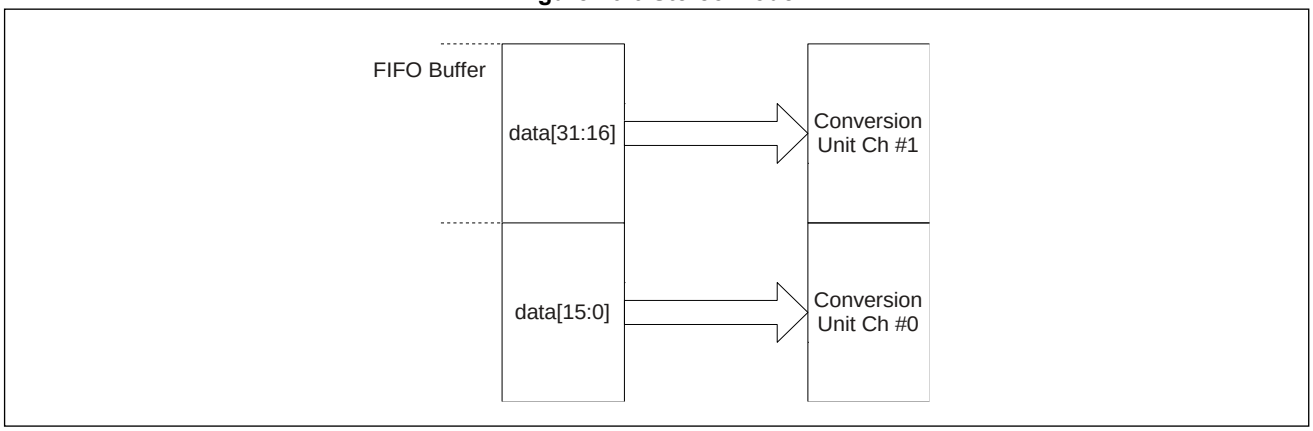
The PCMPWM module supports two modes of driving the two output channels – mono and stereo. In mono mode a single stream of PCM data samples is used to drive both output channels. I.e. both conversion units use the same PCM sample data values. Mono mode alternates between the lower (used first) and the upper 16 bits of the PCM data sample word. Thus the effective depth of the FIFO buffer increases to 48. Figure 26-5 depicts the data transfer from the FIFO buffer to the PCM to PWM conversion units in mono mode.

Figure 26-5 Mono Mode



In stereo mode each PCM sample consists of a pair of values for the two channels. They are written to the module at the same time but the values for both channels are independent of each other. The lower 16 bits of the sample word are used for channel #0 and the upper 16 bits are used for channel #1. Mono or stereo mode is configured with the PCMPWMi_CONTROL:STEREO bit. Figure 26-6 depicts the data transfer from the FIFO buffer to the PCM to PWM conversion units in stereo mode.

Figure 26-6 Stereo Mode



Single or Double Output Mode

The PCMPWM module supports two ways of processing the PCM data samples. In single output mode each PCM data sample is read from the FIFO buffer and then processed by the conversion unit. In double output mode each PCM data sample is processed twice. I.e. the same PCM data sample is used for two PWM cycles. The double output mode enables doubling of the PWM output frequency at the same PCM sample data rate.

The single/double output mode is independent of the mono/stereo mode setting.

Behavior in Debug Mode

The PCMPWM module supports debug mode. If enabled, the module stops converting PCM data samples after the one in progress and instead simply outputs a silent signal on both channels. The silent signal corresponds to a continuous stream of PCM data samples with the value of zero. In this mode no PCM samples are read from the FIFO buffer. However, new sample data may still be written to the FIFO buffer.

If the module's support for debug mode is not enabled, the PCMPWM module continues with regular PCM data sample conversion and PWM signal output.

DMA Mode

The PCM data samples may be transferred to the PCMPWM module via DMA. For this the DMA mode must be enabled by setting the PCMPWMi_CONTROL:DMAEN bit. The transfer works fully autonomously without any CPU intervention. The transfer takes place in chunks. The respective maximum number of 32-bit words per transfer is defined by PCMPWMi_CONTROL:FEST + 1. If the number of free entries in the FIFO buffer exceeds the number specified in PCMPWMi_CONTROL:FEST, a DMA transfer is requested by the PCMPWM module. The DMA then may fill up the full number of entries given by PCMPWMi_CONTROL:FEST + 1.

To set up the PCMPWM module for a DMA data transfer, the DMA unit must be configured for block transfer mode. The block size configured in the DMA must match PCMPWMi_CONTROL:FEST + 1. The PCMPWM module will then assert the PCMPWM_i_DMA_REQ output when the number of free entries in the FIFO buffer exceeds the value given by FEST. PCMPWM_i_DMA_REQ stays active until the DMA acknowledges the request by asserting PCMPWM_i_DMA_REQ_ACK. After the DMA has transferred FEST + 1 PCM data samples to the FIFO buffer, the PCMPWM will again assert the PCMPWM_i_DMA_REQ line again when the required number of free FIFO buffer entries is available. The PCMPWM module keeps track of the number of transferred data words by itself. There is an internal down counter for this purpose which is reloaded with the value of FEST + 1 when PCMPWM_i_DMA_REQ is asserted. It is decremented with each word written to the FIFO buffer. When the counter reaches zero the transfer is complete and another DMA transfer may be requested. It is an error if the DMA writes data to the FIFO while the down counter is zero. This is condition is flagged as a DMA block error in the interrupt status register.

Silence Mode

By setting the PCMPWMi_CONTROL:SILENCE trigger, the module is set to silence mode. When this mode is triggered, the register holding the PCM sample to be converted is cleared to 0 and the reading from the FIFO is stopped. Also the FIFO is flushed. The FIFO can then be filled again via DMA or the CPU. As soon as the number of empty entries in the FIFO is equal or less than PCMPWMi_CONTROL:FEST, reading from the FIFO is resumed.

At begin of operation when the PCMPWM is enabled, the module is in silence mode until the required number of FIFO entries are written. This prevents possible FIFO under-runs directly after the start of operation.

2.3. Output Configuration

The two output channels of the PCMPWM module may be individually enabled and disabled. In addition, the output polarity of each channel may be selected. The enable bits are PCMPWMi_OCTRL:EN0 and PCMPWMi_OCTRL:EN1. The output polarity is programmed with PCMPWMi_OCTRL:LEVL0 and PCMPWMi_OCTRL:LEVL1.

If a channel's output is disabled, all the corresponding signals stay at their inactive values. The inactive value depends on the programmed output polarity. If a channel is programmed to an operation mode which does not use all outputs, the unused outputs also stay at their inactive values. Table 26-1 shows the inactive values of all outputs depending on the selected polarity.

Table 26-1 Inactive Output Values

PCMPWMi_OCTRL:LEVL0/LEVL1	PCMPWM_i_AH/BH	PCMPWM_i_AL/BL
0	1	0
1	0	1

3. Operation of the PCMPWM

This section describes the operation of the PCMPWM module.

3.1. Description of the PCM to PWM Conversion Process

The actual conversion of the PCM data samples to a PWM signal is done in the PCM to PWM converter as shown in Figure 26-1. The first step is a conversion of the signed PCM sample to an unsigned PCM compare value. How this is done depends on the operation mode. In low-pass filter mode and simplified H-bridge mode the step is a simple addition. The value to be added is configured in the PCMPWMi_PCMOFFS register. In full H-bridge mode the PCM compare value is the absolute value of the PCM data sample. Table 26-2 summarizes how the PCM compare value is computed in each operation mode.

Table 26-2 Inactive Output Values

Operation Mode	Computation
Low Pass Filter	PCM sample + PCMOFFS
Simplified H-Bridge	PCM sample + PCMOFFS
Full H-Bridge	|PCM sample|

The second step of the PCM to PWM conversion process comprises a free running counter and two comparators. The counter counts up from zero to a programmable maximum value. The maximum value is given by the PCMPWMi_COUNTP register. Whenever the counter has reached its maximum value, it wraps around to zero. The counter value is continuously compared against two values. The first value is zero and the second is the PCM compare value. The outputs of the two comparators are fed into the actual PWM signal generation. Depending on the operation mode this final block generates the actual PWM signals. Table 26-3 summarizes the logic for generating the PWM signals from the two comparator outputs. Please note the signal levels in the table are for the output polarity with PCMPWMi_OCTRL:LEVLx set to logic 1.

Table 26-3 Generation of PWM Output Signals

Event		Low-Pass Filter Mode	Simplified H-Bridge Mode		Full H-Bridge Mode			
		AH	AH	AL	AH	AL	BH	BL
Counter = 0	PCM sample positive	↑	↑	↓	↑	↓	0	1
	PCM sample = 0				0	1	0	1
	PCM sample negative				0	1	↑	↓
Counter = PCM compare value	PCM sample positive	↓	↓	↑	↓	↑	0	1
	PCM sample = 0				0	1	0	1
	PCM sample negative				0	1	↓	↑

Note:

- *In case the PCM compare value is zero, the event given for equality with the PCM compare value takes precedence.*

Figure 26-7 to Figure 26-10 depict some exemplary conversion cycles for all three operation modes. For low-pass filter mode and simplified H-bridge mode, PCMPWMi_COUNTP:COUNTP is set to 0xFFFF and PCMPWMi_PCMOFFS:PCMOFFS is set to 0x8000. For full H-bridge mode PCMPWMi_COUNTP:COUNTP is set to 0x7FFF. The diagrams show the value of the free running counter periodically ascending from 0 to its maximum value as well as the PCM samples and the corresponding PCM compare value.

Figure 26-7 PCM to PWM Conversion in Low-Pass Filter Mode

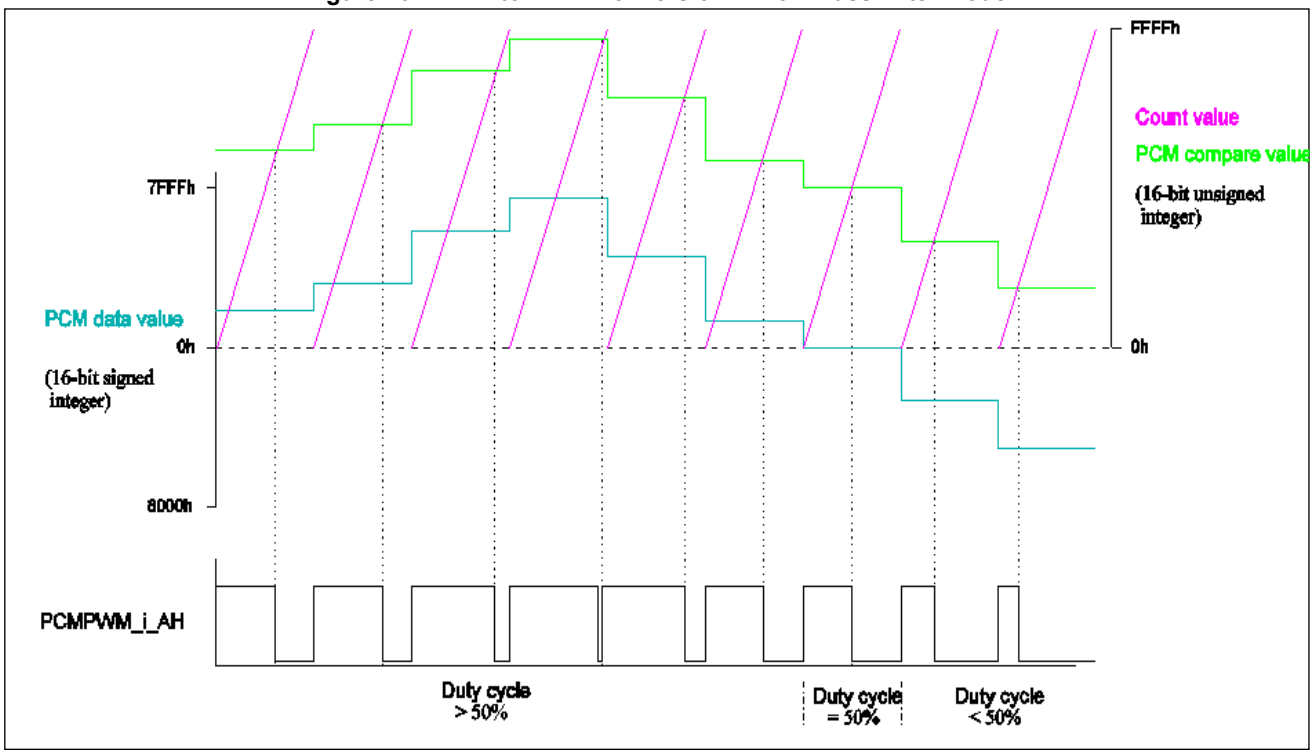


Figure 26-8 PCM to PWM Conversion in Simplified H-Bridge Mode

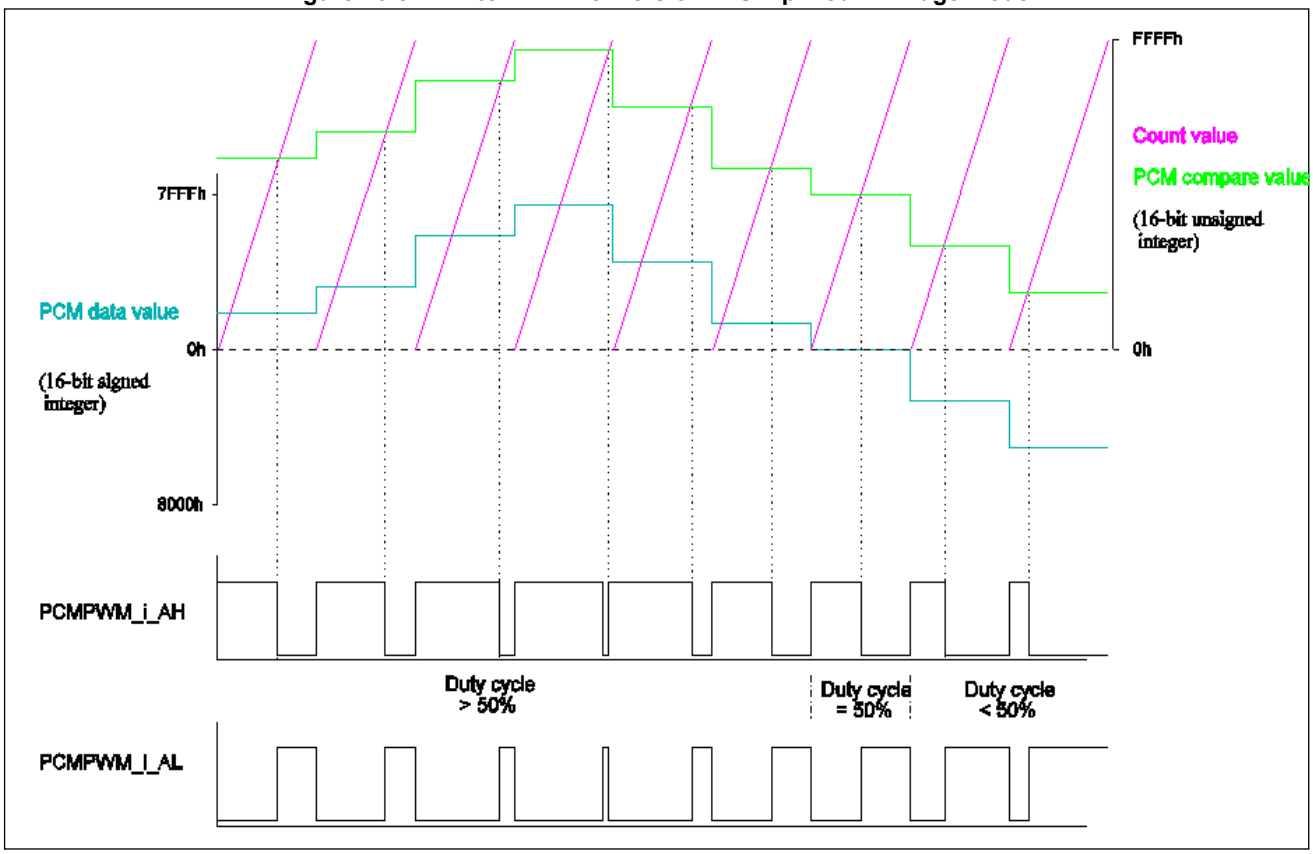


Figure 26-9 PCM to PWM Conversion in Full H-Bridge Mode

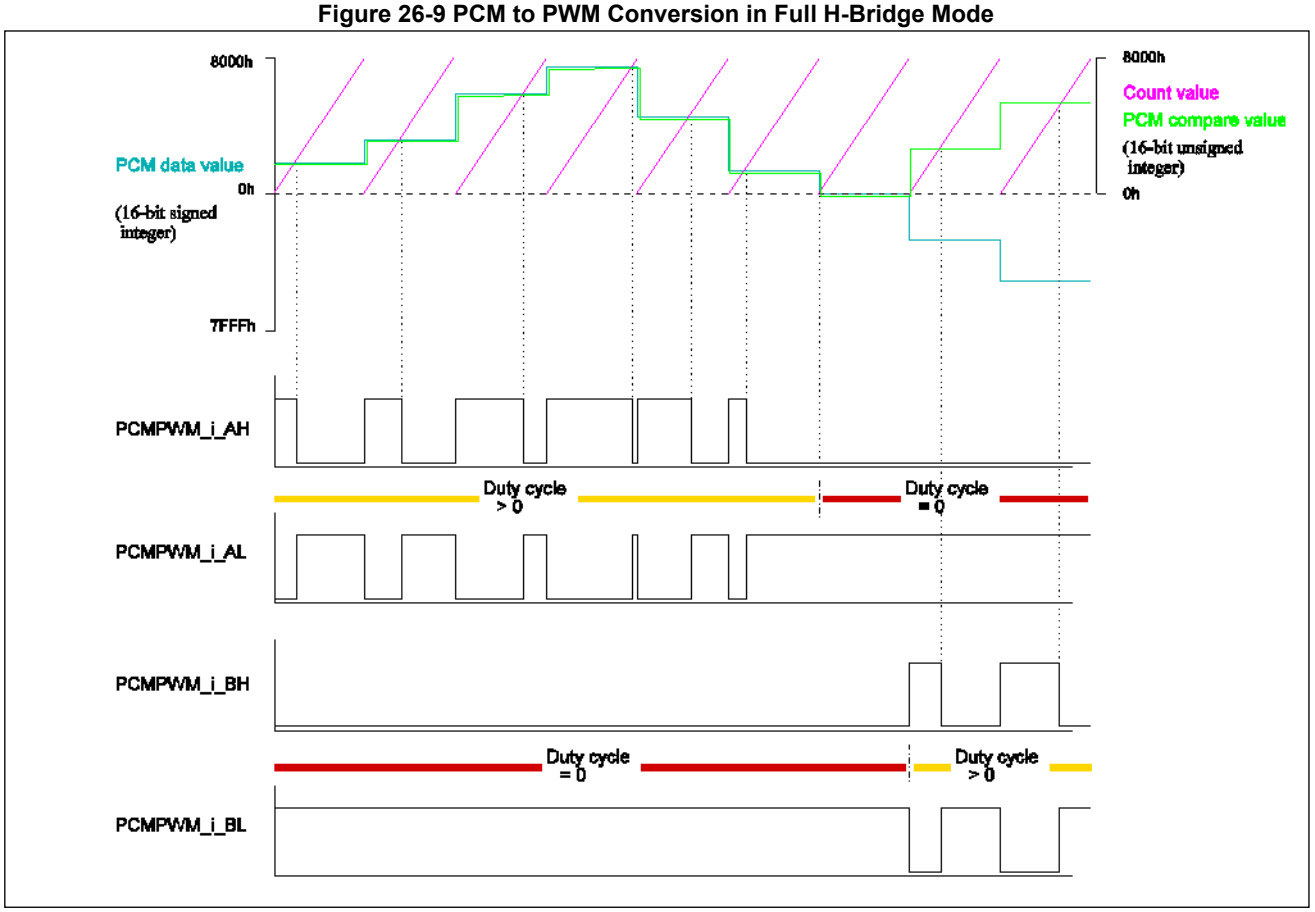
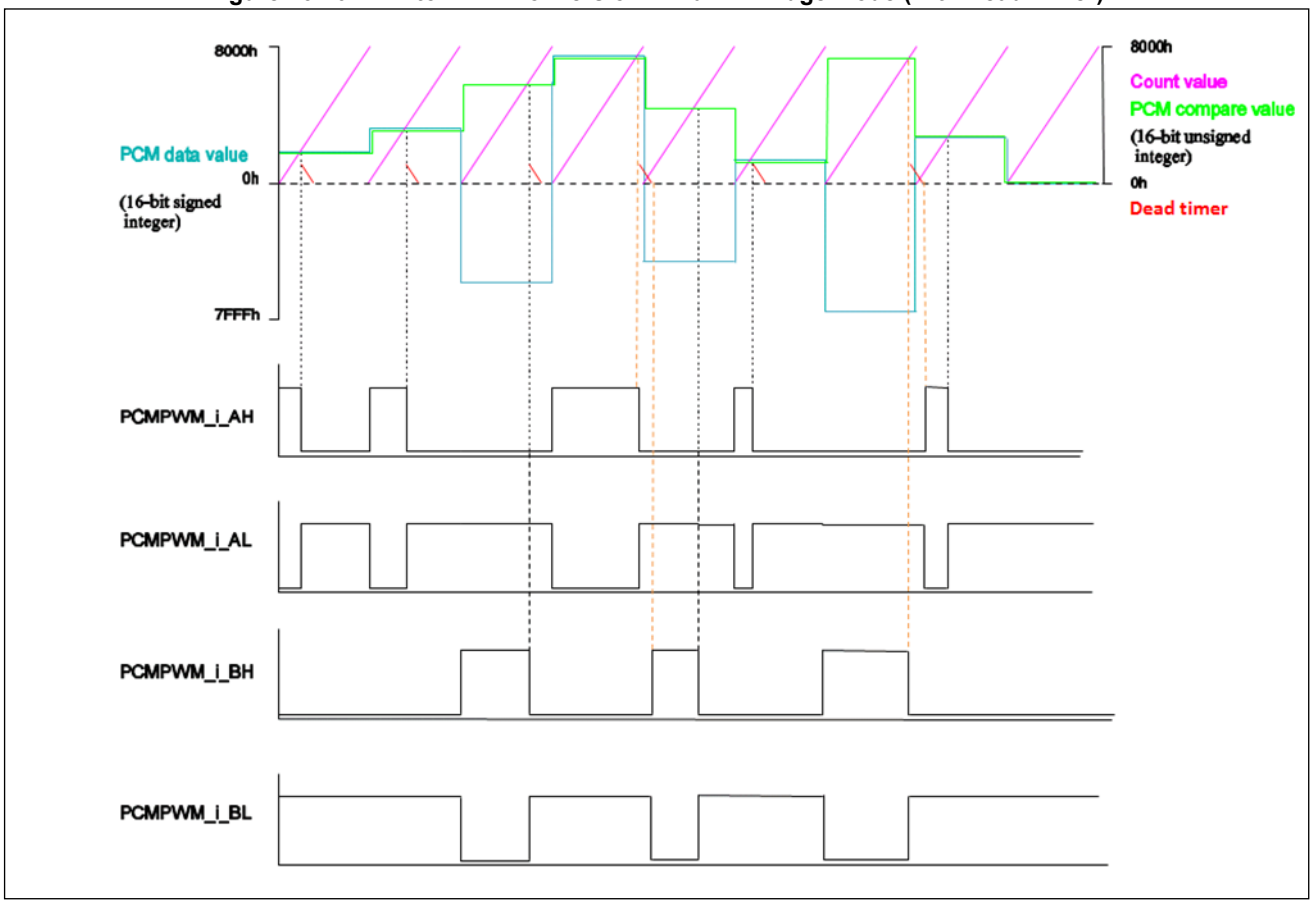


Figure 26-10 PCM to PWM Conversion in Full H-Bridge Mode (with Dead Timer)



3.2. PWM Cycle Time Configuration

The free running counter used in the PCM to PWM conversion process determines the cycle time of the generated PWM signal. The PWM cycle time is equal to the counter's full count period. So, the cycle time computes as follows:

$$T_{PWM} = \frac{COUNTP + 1}{f_{counter}}$$

The countp parameter is configured in the PCMPWMi_COUNTP register. The count frequency of the counter may be configured in the PCMPWMi_CLKSEL register. It determines the counter clock by dividing the PWM clock PWM_CLK by 1, 2, 4 or 8. So, the final formula for the PWM cycle time results to:

$$T_{PWM} = \frac{(COUNTP + 1) \times 2^{CLKSEL}}{f_{PWM_CLK}}$$

3.3. PCM Data Sample Input

The PCM to PWM conversion block reads the PCM data samples from a FIFO buffer. To ensure continuous operation of the PCM to PWM conversion, it is essential to provide the FIFO buffer with the same sustained rate of PCM data samples. The required frequency for the data samples corresponds to the reciprocal of the PWM cycle time:

$$f_{sample} = \frac{f_{PWM_CLK}}{(COUNTP + 1) \times 2^{CLKSEL}}$$

The data samples may be provided by the CPU or may be transferred via DMA. For a CPU based transfer there is support for requesting data by interrupt. Alternatively the CPU may poll the interrupt status register to determine, if there is free space in the FIFO buffer. Regardless of how the buffer FIFO is supplied with data samples, the PCMPWMi_CONTROL:FEST field specifies, at which number of free FIFO buffer entries the data transfer is requested. If the number of free entries exceeds that specified in the FEST field, the PCMPWMi_INTRSTAT:DREQ status bit is set and an interrupt request may be asserted or another DMA data transfer may be initiated.

If the PCM to PWM conversion tries to read a PCM data sample from the FIFO buffer and finds it empty, it uses the PCM value instead. I.e. the module outputs the last PWM pattern again, if no PCM data is supplied. To detect this error case of FIFO buffer under-run, an interrupt request is asserted.

Please note the above formula limits the resolution of the PCM to PWM conversion at a given f_{PWM_CLK} and f_{sample}. E.g. at a clock frequency of 80 MHz, a sample frequency of 19.5 kHz, and CLKSEL set to zero, the counter may count to 4103 only. This corresponds to a resolution of about 12 bits. Table 26-4 shows some examples for the configuration of the PCMPWM module and the resulting sample frequencies and resolutions.

Table 26-4 Configuration Examples

f_{PWM_CLK}	CLKSEL	COUNTP	f_{sample}	Resolution
80 MHz	0	4,095	19.53 kHz	12 bit
80 MHz	0	65,535	1.22 kHz	16 bit
80 MHz	2	255	78.13 kHz	8 bit
100 MHz	1	1,023	48.83 kHz	10 bit

3.4. Interrupts

The PCMPWM module interrupts are controlled by three registers: the Interrupt Enable Register (PCMPWMi_INTREN), the Interrupt Status Register (PCMPWMi_INTRSTAT) and the Interrupt Clear Register (PCMPWMi_INTRCLR). After reset all interrupts are disabled. If an interrupt is to be used, it must be first enabled. The current status of an interrupt may be checked at any time in the Interrupt Status Register. The Interrupt Status Register contents are independent of the enable status of the interrupts. I.e. the Interrupt Status Bits are not masked by the Interrupt Enable Register.

If an interrupt has occurred, it can be reset by the Interrupt Clear Register. Writing a logic 1 to a bit in the Interrupt Clear Register clears the corresponding interrupt line as well as the interrupt status bit.

1. DMA Block Error Interrupt
This interrupt indicates an error case when the DMA tries to transfer more data to the FIFO buffer than configured by PCMPWMi_CONTROL:FEST + 1.
2. FIFO Buffer Under-Run Error Interrupt
This interrupt indicates an under-run of the FIFO buffer for PCM data samples. I.e. the PCM to PWM conversion has tried to read a PCM data sample from the FIFO buffer when it was empty. Instead of a new PCM data sample the PCM to PWM conversion uses the last PCM value instead.
3. FIFO Buffer Overflow Error Interrupt
This interrupt indicates the CPU has tried to write another PCM data sample to the FIFO buffer when it was already completely filled.
4. Data Request Interrupt
This interrupt indicates there is at least space for another FEST + 1 PCM data samples in the FIFO buffer. I.e. the PCMPWM module asserts the interrupt request when a read from the FIFO buffer frees up another PCM samples slot, so that there are FEST + 1free entries in total.

3.5. Dead Timer Operation

In full H-Bridge mode, the 16-bit counter is compared in each count period CP with the value from the function minimum(duty, cycle) (this is the falling edge or the end of the PWM cycle). If the two values match, a down counter loaded with PCMPWMi_CONTROL:DTVAL starts to decrement with every clock cycle of the 16-bit counter. As long as the down counter is not zero, the opposite phase of the PWM outputs is masked with inactive values. Opposite phase in this case means:

- If the PCM data sample of count period CP is positive, then output signals PCMPWM_i_BL/BH are masked, although this doesn’t happen in usual setup.
- If the PCM data sample of count period CP is negative, then output signals PCMPWM_i_AL/AH are masked, although this doesn’t happen in usual setup.

4. Registers

This section describes the registers of the PCMPWM module.

Note:

- *The suffix 'i' in the register name indicates that the register is in instance 'i' of the module..*

The following registers are available for each instance of the PCMPWM:

PCMPWM Control Register (PCMPWMi_CONTROL)
PCMPWM Output Control Register (PCMPWMi_OCTRL)
PCMPWM Clock Select Register (PCMPWMi_CLKSEL)
PCMPWM Count Period Register (PCMPWMi_COUNTP)
PCM Offset Register (PCMPWMi_PCMOFFS)
PCM Interrupt Enable Register (PCMPWMi_INTREN)
PCM Interrupt Status Register (PCMPWMi_INTRSTAT)
PCM Interrupt Clear Register (PCMPWMi_INTRCLR)
PCM Data Register 0..15 (PCMPWMi_DATA)

4.1. PCMPWM Control Register (PCMPWMi_CONTROL)

This Global Control Register is used to configure the PCMPWM module's basic mode of operation.

REGISTER_NAME	PCMPWMi_CONTROL
OFFSET	0x00000000
ACCESS_SIZE	B, H, W
MULTIPLE	0
NUMERIC_TYPE	-
OTHER	-

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	DTVAL[7:0]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved			FEST[4:0]				
ACCESS_TYPE	R0,WX			R/W				
PROT_TYPE	Wp							
INITIAL_VALUE	0			0				

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved		DOUBLE	MODE[1:0]		STEREO	DBGEN	DMAEN
ACCESS_TYPE	R0,WX		R/W	R/W		R/W	R/W	R/W
PROT_TYPE	Wp							
INITIAL_VALUE	0		0	0		0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved						SILENCE	EN
ACCESS_TYPE	R0,WX						R0,W1	R/W
PROT_TYPE	Wp							
INITIAL_VALUE	0						0	0

[bit31:24] DTVAL[7:0] : Dead Timer Value

This field defines the number of PWM_CLK cycles after the trailing edge for which the opposite phase of the PWM outputs in full H-Bridge Mode are masked with inactive value.

Setting of this field to a value greater than or equal to PCMPWMi_COUNTP:COUNTP is not allowed.

[bit23:21] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit20:16] FEST[4:0] : FIFO Empty Space Threshold

This field defines the number of empty entries in the FIFO buffer, which triggers a DMA request and/or a data request interrupt when it is exceeded. If this field is set to a value from 0 to 23 a data request is generated when the number of empty FIFO buffer entries exceeds this value. If this field is set to a value greater than 23, a data request is never generated.

Note:

- *These bits must not be changed during DMA transfers.*

[bit15:14] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit13] DOUBLE : Double Mode Enable

This bit enables operation in single or double mode.

DOUBLE	Description
0	The module operates in single output mode, i.e. each PCM data sample is processed once by the conversion unit
1	The module operates in double output mode, i.e. each PCM data sample is processed twice by the conversion unit

[bit12:11] MODE[1:0] : Operation Mode

This field defines the module's mode of operation.

MODE[1:0]	Description
00	The module is configured for low-pass filter mode
01	The module is configured for simplified H-bridge mode
10	The module is configured for full H-bridge mode
11	Not allowed

[bit10] STEREO : Stereo Mode Enable

This bit enables operation in mono or stereo mode.

STEREO	Description
0	The module operates in mono mode, i.e. both channels output the same data
1	The module operates in stereo mode, i.e. each channel outputs its individual data

[bit9] DBGEN : Debug Mode Enable

This bit controls the module's operation when the CPU is in debug state.

DBGEN	Description
0	Disabled - The module continues regular operation when the CPU enters debug state
1	The module generates silence when the CPU is in debug state; no data is fetched from the FIFO buffer

[bit8] DMAEN : DMA Mode Enable

This bit controls the module's DMA interface.

DMAEN	Description
0	The DMA interface is disabled
1	The DMA interface is enabled

[bit7:2] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit1] SILENCE : Silence Module Trigger

This bit triggers the Silence Mode of the PCMPWM module. Reading of this bit returns always 0. Writing 1 enables the Silence Mode, writing 0 has no effect. Since this bit is in the same byte as the EN bit and writing to SILENCE has only sense if the module is enabled, EN = 1 needs to be written at the same time as SILENCE is written.

SILENCE	Description
0	No change
1	Enable Silence Mode

[bit0] EN : PCMPWM Module Enable

This bit globally enables or disables the PCMPWM module. This bit must not be set before the module's configuration has been completed. After the module has been enabled via this bit and the PCM sample data FIFO buffer has been prefilled, the PCM to PWM conversion starts immediately. If the FIFO buffer is empty when the module gets enabled, silence is driven on the PWM outputs and the PCM to PWM conversion starts as soon as there are equal or less than PCMPWMi_CONTROL:FEST empty spaces in the FIFO. The conversion counter is always restarted from zero when the module gets enabled via this bit.

If the module's configuration is changed while this bit is set, i.e. the PCM to PWM conversion is ongoing; the changes take effect immediately and may lead to audible noise. Depending on the type of changes the reprogramming may also cause misbehavior on the module's other hardware interfaces. E.g. the DMA transfer protocol may be violated and thus a DMA error condition might be caused.

EN	Description
0	PCMPWM is disabled; if the module is switched off via this bit, the PWM output stops immediately and the outputs are set to their inactive values; no interrupts and DMA transfers can be requested
1	PCMPWM is enabled

4.2. PCMPWM Output Control Register (PCMPWMi_OCTRL)

This Output Control Register is used to configure the individual output channels of the PCMPWM module.

REGISTER_NAME	PCMPWMi_OCTRL
OFFSET	0x00000004
ACCESS_SIZE	B, H, W
MULTIPLE	0
NUMERIC_TYPE	-
OTHER	-

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved						LEVL1	LEVL0
ACCESS_TYPE	R0,WX						R/W	R/W
PROT_TYPE	Wp							
INITIAL_VALUE	0						1	1

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0							

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	Wp							
INITIAL_VALUE	0						0	0

[bit31:18] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit17] LEVL1 : Output Level Select Channel #1

This bit selects the output polarity of channel #1.

LEVL1	Description
0	PCMPWM_1_AH/BH are low-active and PCMPWM_1_AL/BL are high-active
1	PCMPWM_1_AH/BH are high-active and PCMPWM_1_AL/BL are low-active

[bit16] LEVL0 : Output Level Select Channel #0

This bit selects the output polarity of channel #0.

LEVL0	Description
0	PCMPWM_0_AH/BH are low-active and PCMPWM_0_AL/BL are high-active
1	PCMPWM_0_AH/BH are high-active and PCMPWM_0_AL/BL are low-active

[bit15:2] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit1] EN1 : Output Enable Channel #1

This bit enables or disables PWM signal generation on channel #1.

EN1	Description
0	PCMPWM_1_AH/BH/AL/BL are set to the inactive value
1	The PWM signal is enabled on channel #1

[bit0] EN0 : Output Enable Channel #0

This bit enables or disables PWM signal generation on channel #0.

EN0	Description
0	PCMPWM_0_AH/BH/AL/BL are set to the inactive value
1	The PWM signal is enabled on channel #0

4.3. PCMPWM Clock Select Register (PCMPWMi_CLKSEL)

This Clock Select Register is used to configure the PCMPWM conversion counter clock frequency.

REGISTER_NAME	PCMPWMi_CLKSEL
OFFSET	0x00000008
ACCESS_SIZE	B, H, W
MULTIPLE	0
NUMERIC_TYPE	-
OTHER	-

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved						CLK_SEL[1:0]	
ACCESS_TYPE	R0,WX						R/W	
PROT_TYPE	Wp							
INITIAL_VALUE	0						0	

[bit31:2] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit1:0] CLK_SEL : Clock Select

This field selects the PCMPWM conversion counter clock.

CLK_SEL[1:0]	Description
00	Divide by 1: Conversion counter clock = PWM_CLK
01	Divide by 2: Conversion counter clock = PWM_CLK/2
10	Divide by 4: Conversion counter clock = PWM_CLK/4
11	Divide by 8: Conversion counter clock = PWM_CLK/8

4.4. PCMPWM Count Period Register (PCMPWMi_COUNTP)

This Count Period Register is used to configure the PCMPWM conversion counter period.

REGISTER_NAME	PCMPWMi_COUNTP
OFFSET	0x0000000C
ACCESS_SIZE	B, H, W
MULTIPLE	0
NUMERIC_TYPE	-
OTHER	-

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	COUNTP[15:8]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	1							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	COUNTP[7:0]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	1							

[bit31:16] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit15:0] COUNTP : Count Period

This field defines the value of the PCMPWM conversion counter after which it continues to count from 0.

Setting of this field to 0 is not allowed.

4.5. PCM Offset Register (PCMPWMi_PCMOFFS)

This PCM Offset Register is used to configure the offset value used in the PCM signed to unsigned conversion.

REGISTER_NAME	PCMPWMi_PCMOFFS
OFFSET	0x00000010
ACCESS_SIZE	B, H, W
MULTIPLE	0
NUMERIC_TYPE	-
OTHER	-

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	PCM_OFFS[15:8]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	PCM_OFFS[7:0]							
ACCESS_TYPE	R/W							
PROT_TYPE	Wp							
INITIAL_VALUE	0							

[bit31:16] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit15:0] PCM_OFFS : PCM Offset

This field defines the offset value used in the PCM signed to unsigned conversion in low-pass filter and simplified H-bridge mode.

4.6. PCM Interrupt Enable Register (PCMPWMi_INTREN)

This PCM Interrupt Enable Register is used to enable or disable individual interrupts of the PCMPWM module.

REGISTER_NAME	PCMPWMi_INTREN
OFFSET	0x00000014
ACCESS_SIZE	B, H, W
MULTIPLE	0
NUMERIC_TYPE	-
OTHER	-

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved				DMA_ERR	UDRN	OVFL	DREQ
ACCESS_TYPE	R0,WX				R/W	R/W	R/W	R/W
PROT_TYPE	Wp							
INITIAL_VALUE	0				0	0	0	0

[bit31:4] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit3] DMA_ERR : DMA Block Error

This bit enables or disables the DMA block error interrupt.

DMA_ERR	Description
0	The DMA block error interrupt is disabled
1	The DMA block error interrupt is enabled

[bit2] UDRN : FIFO Under-Run Error

This bit enables or disables the FIFO under-run error interrupt.

UDRN	Description
0	The FIFO under-run error interrupt is disabled
1	The FIFO under-run error interrupt is enabled

[bit1] OVFL : FIFO Overflow Error

This bit enables or disables the FIFO overflow error interrupt.

OVFL	Description
0	The FIFO overflow error interrupt is disabled
1	The FIFO overflow error interrupt is enabled

[bit0] DREQ : Data Request

This bit enables or disables the data request interrupt.

DREQ	Description
0	The FIFO data request interrupt is disabled
1	The FIFO data request interrupt is enabled

4.7. PCM Interrupt Status Register (PCMPWMi_INTRSTAT)

This PCM Interrupt Status Register reflects the status of the individual interrupt sources of the PCMPWM module. The bits in this registers are ‘sticky’, i.e. once they have been set from the hardware, they remain set until they are reset via a write access to the PCMPWMi_INTRCLR register.

REGISTER_NAME	PCMPWMi_INTRSTAT
OFFSET	0x00000018
ACCESS_SIZE	B, H, W
MULTIPLE	0
NUMERIC_TYPE	-
OTHER	-

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WN							
PROT_TYPE	WN							
INITIAL_VALUE	0							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WN							
PROT_TYPE	WN							
INITIAL_VALUE	0							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WN							
PROT_TYPE	WN							
INITIAL_VALUE	0							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved				DMA_ERR	UDRN	OVFL	DREQ
ACCESS_TYPE	R0,WN				R,WN	R,WN	R,WN	R,WN
PROT_TYPE	WN							
INITIAL_VALUE	0				0	0	0	0

[bit31:4] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit3] DMA_ERR : DMA Block Error

This status bit reflects if a DMA block error has occurred.

DMA_ERR	Description
0	No error
1	A DMA block error has occurred, i.e. there were more data transfers to the FIFO buffer than specified by PCMPWMi_CONTROL:FEST + 1

[bit2] UDRN : FIFO Under-Run Error

This status bit reflects if a FIFO buffer under-run has occurred.

UDRN	Description
0	No error
1	A FIFO buffer under-run has occurred, i.e. the PCM to PWM conversion block tried to read another PCM data sample, but the FIFO buffer is empty

[bit1] OVFL : FIFO Overflow Error

This status bit reflects if a FIFO buffer overflow has occurred.

OVFL	Description
0	No error
1	A FIFO buffer overflow has occurred, i.e. there was a write access to the FIFO buffer when there was no more space available

[bit0] DREQ : Data Request

This status bit reflects if there is space for another n PCM data samples in the FIFO buffer. The number n is defined by PCMPWMi_CONTROL:FEST + 1.

DREQ	Description
0	No data request: There are less than PCMPWMi_CONTROL:FEST + 1empty entries available in the FIFO buffer
1	Data request: There are PCMPWMi_CONTROL:FEST + 1 or more empty entries available in the FIFO buffer

4.8. PCM Interrupt Clear Register (PCMPWMi_INTRCLR)

This PCM Interrupt Clear Register is used to clear individual interrupts of the PCMPWM module.

REGISTER_NAME	PCMPWMi_INTRCLR
OFFSET	0x0000001C
ACCESS_SIZE	B, H, W
MULTIPLE	0
NUMERIC_TYPE	-
OTHER	-

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved							
ACCESS_TYPE	R0,WX							
PROT_TYPE	Wp							
INITIAL_VALUE	0							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved				DMA_ERR	UDRN	OVFL	DREQ
ACCESS_TYPE	R0,WX				R0/W	R0/W	R0/W	R0/W
PROT_TYPE	Wp							
INITIAL_VALUE	0				0	0	0	0

[bit31:4] Reserved

This bit is reserved.

Always write "0" to this bit. The read value is "0".

[bit3] DMA_ERR : DMA Block Error

When this bit is written as ‘1’, the DMA block error interrupt is cleared.

DMA_ERR	Description
0	Leave the DMA block error interrupt unchanged
1	Writing a ‘1’ to this bit clears the DMA block error interrupt. This bit is always read as ‘0’.

[bit2] UDRN : FIFO Under-Run Error

When this bit is written as ‘1’, the FIFO under-run error interrupt is cleared.

UDRN	Description
0	Leave the FIFO under-run error interrupt unchanged
1	Writing a ‘1’ to this bit clears the FIFO under-run error interrupt. This bit is always read as ‘0’.

[bit1] OVFL : FIFO Overflow Error

When this bit is written as ‘1’, the FIFO overflow error interrupt is cleared.

OVFL	Description
0	Leave the FIFO overflow error interrupt unchanged
1	Writing a ‘1’ to this bit clears the FIFO overflow error interrupt. This bit is always read as ‘0’.

[bit0] DREQ : Data Request

When this bit is written as ‘1’, the data request interrupt is cleared.

DREQ	Description
0	Leave the data request interrupt unchanged
1	Writing a ‘1’ to this bit clears the data request interrupt. This bit is always read as ‘0’.

4.9. PCM Data Register 0..15 (PCMPWMi_DATA)

This PCM Data Register used to write PCM data samples to the PCMPWM module.

REGISTER_NAME	PCMPWMi_DATA
OFFSET	0x00000040
ACCESS_SIZE	W
MULTIPLE	0:15
NUMERIC_TYPE	-
OTHER	-

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	DATA1[15:8]							
ACCESS_TYPE	RN,W							
PROT_TYPE	RN							
INITIAL_VALUE	-							

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	DATA1[7:0]							
ACCESS_TYPE	RN,W							
PROT_TYPE	RN							
INITIAL_VALUE	-							

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	DATA0[15:8]							
ACCESS_TYPE	RN,W							
PROT_TYPE	RN							
INITIAL_VALUE	-							

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	DATA0[7:0]							
ACCESS_TYPE	RN,W							
PROT_TYPE	RN							
INITIAL_VALUE	-							

[bit31:16] DATA1 : Data Input Channel #1

This field holds the PCM data for channel #1. In mono mode the data is used for channel #0 as well.

[bit15:0] DATA0 : Data Input Channel #0

This field holds the PCM data for channel #0. In mono mode the data is used for channel #1 as well.

CHAPTER 27: HyperBus Interface

This chapter explains the functionality and operation of the HyperBus interface (HYPERBUSI).

1. Overview
2. Block Diagram
3. Operation of the HYPERBUSI
4. Registers

CODE: HYPBUS-S6J3300-E1

1. Overview

The HYPERBUSI module provides function and operation for interfacing to the HyperBus memory devices. The HyperBus achieves both high speed read/write throughput by double data rate interface and low pin counts. The HYPERBUSI module's features are described in this section.

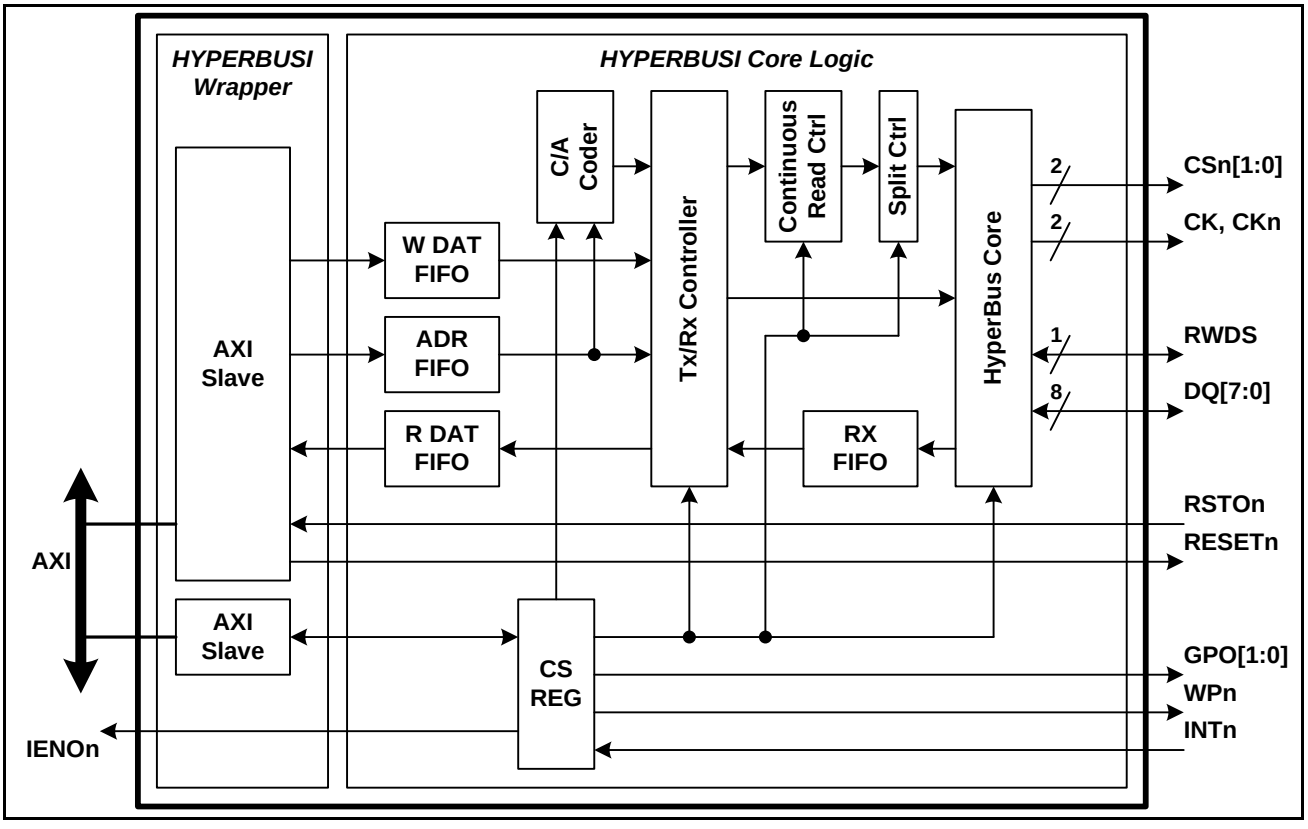
Features of the HyperBus Interface

- Supports operational frequency up to 166 MHz
- Achieves maximum 333 MB/s data throughput by 8 bits bus and few timing signals only
- Supports the double data rate interface
- Supports maximum 4 GByte address space
- Supports two slave devices
- Supports XiP operation by dynamic wrapped burst request

2. Block Diagram

This section shows a block diagram of HYPERBUSI module.

Figure 27-1 Block Diagram of HYPERBUSI



3. Operation of the HYPERBUSI

This section describes the operation of HYPERBUSI module.

3.1. HyperBus Core

This section describes physical interface of HYPERBUSI module. The HyperBus core generates HyperBus timing from control signals.

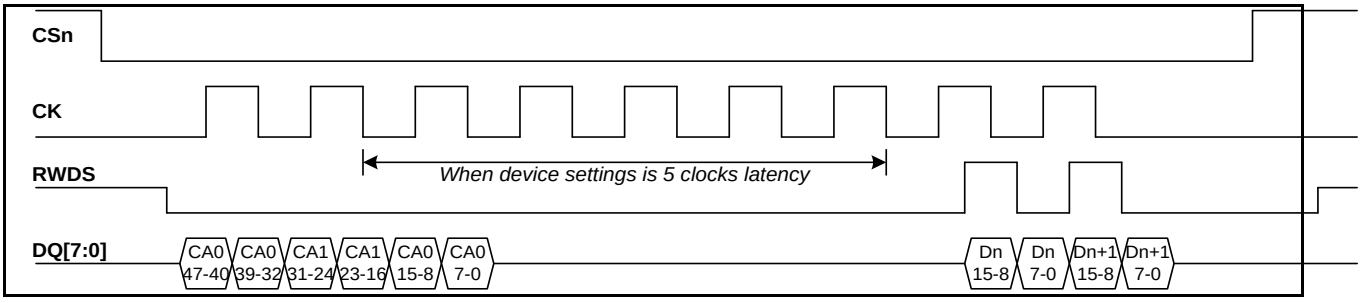
3.1.1. Read Operation

This section describes the read operation of HYPERBUSI. HYPERBUSI module asserts CSn, CK, and DQ. Then, it outputs 3 words command/address, and reads 1 or more data by the timing of RWDS.

If both RX FIFO and R DAT FIFO become full, HyperBus Core ends the access. Then, when only RX FIFO becomes empty, HyperBus Core re-initiates the access. At this time, R DAT FIFO is transferring read data continuously to AXI bus.

The HYPERBUSI puts the transaction request from AXI bus, such as the burst type and the burst length, to the C/A cycle on HyperBus as is.

Figure 27-2 Read Waveform



3.1.2. Write Operation

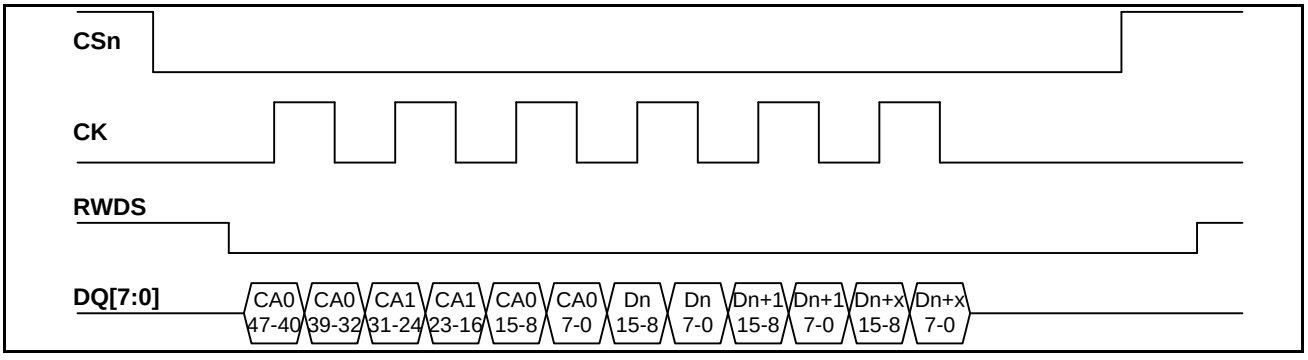
This section describes the write operation of HYPERBUSI for HyperFlash. HYPERBUSI asserts CSn, CK, and DQ. Then, it outputs 3 words command/address, and writes 1 or more data. This write operation is used when the DEVTYPE of HYPERBUSI_MCRn is chosen as HyperFlash.

The HYPERBUSI module puts the transaction request from AXI bus, such as the burst type and the burst length, to the C/A cycle on HyperBus as is.

Notes:

- *Since HyperFlash supports only the write operation in every 2 bytes, invalid data (0xFF) is written as insufficient one byte if only one byte is written.*
- *When using HyperFlash, the burst write of HyperFlash is supported only when CK frequency is 50 MHz or less. Therefore, when the CK frequency of HyperFlash is faster than 50 MHz, the write data size in an AXI transaction should be only 2 bytes.*

Figure 27-3 Write Waveform

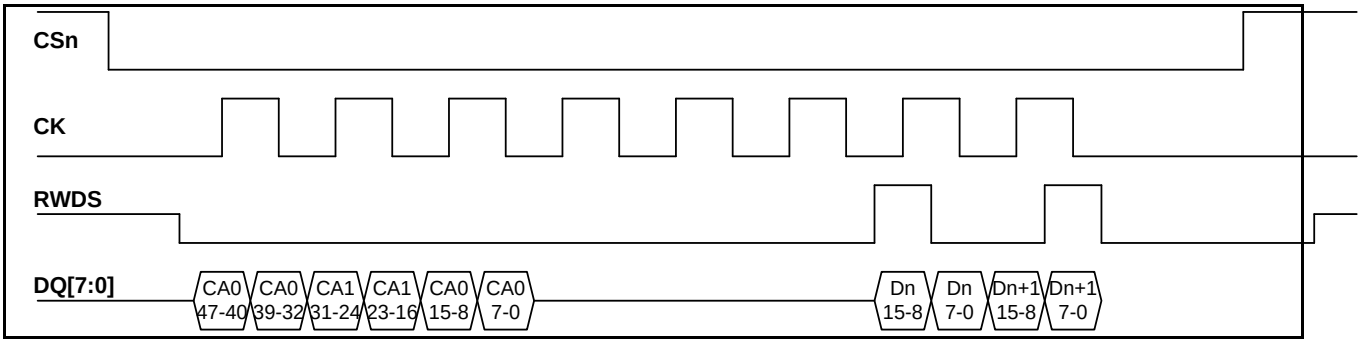


3.1.3. Write Operation with Byte Mask

This section describes the write operation of HYPERBUSI with byte mask for HyperRAM. HYPERBUSI asserts CSn, CK, and DQ. Then, it outputs 3 words command/address, and writes 1 or more data after latency cycles. This write operation is used when the DEVTYPE of HYPERBUSI_MCRn is chosen as HyperRAM. The HYPERBUSI module outputs RWDS signal as the byte mask during write operation. The invalid write data is masked by high of RWDS.

The HYPERBUSI module puts the transaction request from AXI bus, such as the burst type and the burst length, to the C/A cycle on HyperBus as is. Therefore, RWDS is determined by the access address, data size, and strobe signal on AXI bus.

Figure 27-4 Write Waveform with Byte Mask



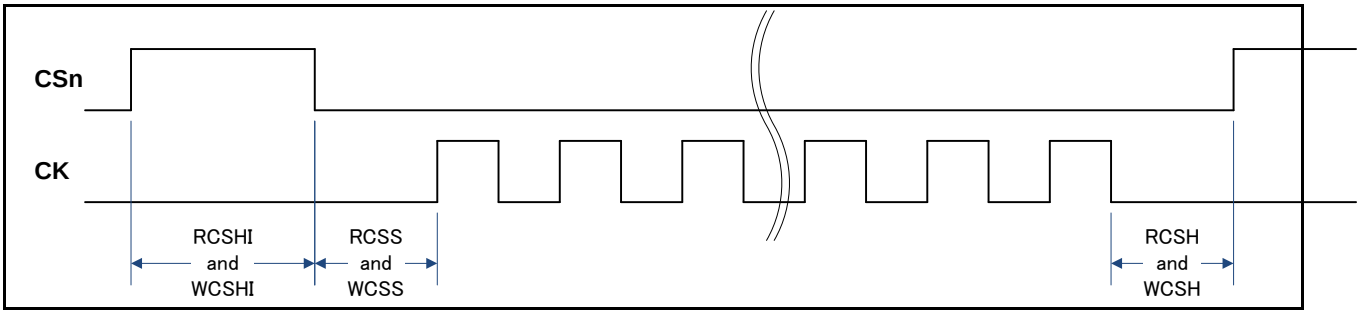
Note:

- *In Figure 27-4, Dn 15-8 and Dn+1 7-0 are invalid data.*

3.1.4. Timing Adjustment

This section describes the HyperBus timing adjustment. HyperBus core has timing adjustment circuit and timing adjustment is controlled by HYPERBUSI_MTRn. RCSHI, RCSS, and RCSH in HYPERBUSI_MTRn are used for the read timing adjustment, and WCSHI, WCSS, and WCSH in HYPERBUSI_MTRn are used for the write timing adjustment. Please refer to section 4.6 Memory Timing Register (HYPERBUSIn_MTR0 to 1).

Figure 27-5 Timing Adjustment



3.2. Tx/Rx Controller

This section describes the Tx/Rx Controller of HYPERBUSI module. The Tx/Rx Controller performs Tx/Rx operation control by state machine, and flow control of data between AXI bus and HyperBus. In addition, when next access address is accepted by multiple outstanding address on AXI bus, HYPERBUSI passes the transaction to the Continuous Read Ctrl if the address of subsequent access is continuous address to the present access.

3.2.1. Asymmetry Cache System Support

This section describes the asymmetry cache system support. The Tx/Rx Controller supports the system which has a different request size of the wrap burst from asymmetry cache system by having wrap size setting of HyperBus memory. This function is optional. Please refer to section 4.5 Memory Configuration Register (HYPERBUSIn_MCR0 to 1) for more detailed information.

The operation of the Tx/Rx Controller for asymmetry cache system support is below.

- The Tx/Rx Controller accepts the wrap read transaction, and compares the required wrap size with the WRAPSIZE in HYPERBUSI_MCRn.
- When the wrap size is the same, the Tx/Rx Controller requires the wrap burst read to the HyperBus.
- When the wrap size differs, the Tx/Rx Controller emulates wrap burst read by requesting two continuous burst read to the HyperBus.

3.2.2. Continuous Read Ctrl

This section describes the Continuous Read Ctrl of HYPERBUSI module. When the address of subsequent access is continuous address to the present access, Continuous Read Ctrl merges the subsequent access to present access in order to improve the performance by removing the C/A cycle and initial latency cycle.

The Continuous Read Ctrl merges some read transactions by the following condition.

- Access to HyperFlash.
- Between the sequential read transactions which have INCR as the type of the burst and have the continuous address on AXI bus are merged.
- Between the read transaction with WRAP and the subsequent read transaction with INCR which have an address following the final address of wrap boundary are merged. This function is optional. Please refer to section 4.5 Memory Configuration Register (HYPERBUSIn_MCR0 to 1) for more detailed information.

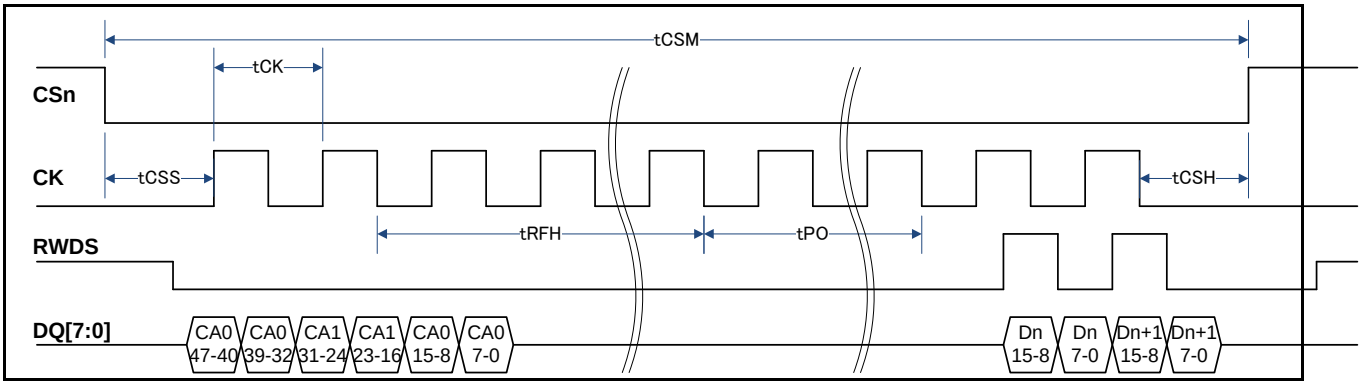
3.2.3. Split Ctrl

This section describes the Split Ctrl of HYPERBUSI module. The HyperRAM has Chip Select Maximum Low Time (tCSM) regulation and the Split Ctrl splits up a transaction in order to satisfy tCSM regulation. This function is controlled by MAXEN and MAXLEN bit in the Memory Configuration Register. Please refer to section 4.5 Memory Configuration Register (HYPERBUSIn_MCR0 to 1) for more detailed information. The Split Ctrl splits up a transaction by access data size (MAXLEN), therefore MAXLEN should be calculated from tCSM as follows.

MAXLEN ≦ ((tCSM - tCSS - 1.5 tCK - tRFH - tPO - tCSH) / tCK) - 1

tCSM : Chip Select Maximum LOW Time. It is AC timing requirement of HyperRAM.
tCK : HyperBus CK Period.
tCSS : This parameter is decided by RCSS and WCSS in HYPERBUSIn_MTR0 to 1.
tRFH : This parameter is decided by LTCY in HYPERBUSIn_MTR0 to 1.
tPO : This parameter is decided by LTCY in HYPERBUSIn_MTR0 to 1.
tCSH : This parameter is decided by RCSH and WCSH in HYPERBUSIn_MTR0 to 1.

Figure 27-6 AC Timing for MAXLEN Calculation



3.3. C/A Coder

The HYPERBUSI module outputs 6 bytes of command/address information to define the transaction. The C/A Coder builds a command/address information by the request from AXI bus and the register settings.

Table 27-1 C/A Format of HYPERBUSI

C/A Bit	Name	Assignment
47	R/W#	0: Write, 1: Read
46	Target	CRT in HYPERBUSI_MCRn
45	Burst Type	0: WRAP, 1: INCR
44	RFU	0
43 - 16	Page Address	Address [31:4]
15	RFU	0
14 - 13	RFU	3
12 - 3	RFU	0
2 - 0	Column Address	Address [3:1]

3.4. FIFO

This section describes the FIFO for processing transaction request.

3.4.1. ADR FIFO

ADR FIFO is composed of 46 bits width x 16 depth. This FIFO stores the control data of AXI bus transaction which includes address, length, burst type and R/W flag.

3.4.2. W DAT FIFO

The W DAT FIFO is composed of 40 bits width x 128 depth. This FIFO is used to store the write data, valid information, and strobe information written from the AXI bus, and in order to receive the burst write transaction without a wait on AXI bus.

3.4.3. R DAT FIFO and RX FIFO

The RX FIFO is composed of 20 bits width x 256 depth. This FIFO is used to store the data which reads from HyperBus, and stored data is outputted according to timing of CK on HyperBus to R DAT FIFO.

The R DAT FIFO is composed of 40 bits width x 128 depth. This FIFO is used to store the data read from RX FIFO and the error detection information, and data stored is outputted according to timing of ACLK on AXI bus. This error information is transferred to the master device by RRESP signals on AXI bus.

The FIFO for receiving data is divided into RX FIFO and R DAT FIFO in order to absorb the timing difference between AXI and HyperBus.

3.5. CS REG (Control Status Register)

The CS REG holds the configuration registers, which are used to control of the HYPERBUSI module. Please refer to section 4 Registers for more detailed information about the CS REG.

And, CS REG controls and outputs the WPn, IENOn, and GPO signals and inputs the INTn signal for controlling IENOn.

3.5.1. WPn Signal

WPn signal is allocated to external interface in order to connect WP# of HyperBus memory. WPn signal is controlled by HYPERBUSI_WPR. Please refer to section 4.8 Write Protection Register (HYPERBUSIn_WPR).

3.5.2. INTn / IENOn Signal

INTn signal is allocated to external interface in order to connect INT# of HyperBus memory. IENOn signal is connected internal interrupt controller and is controlled by HYPERBUSI_IEN and INTn signal. Please refer to section 4.2 Interrupt Enable Register (HYPERBUSIn_IEN) and 4.3 Interrupt Status Register (HYPERBUSIn_ISR).

3.5.3. GPO Signal

GPO signals are general purpose output signal and those signals are used for internal control or external control depending on the implemented system. GPO signals are controlled by HYPERBUSI_GPOR. Please refer to section 4.7 General Purpose Output Register (HYPERBUSIn_GPOR).

The Figure 27-7 shows an example of internal control. In the example, GPO was used for choosing HYPERBUSI module.

The Figure 27-8 shows an example of external control. In the example, GPO was used for resetting HyperBus memory by software in addition to Hardware Reset from AXI interface.

Figure 27-7 Internal Control Example by GPO

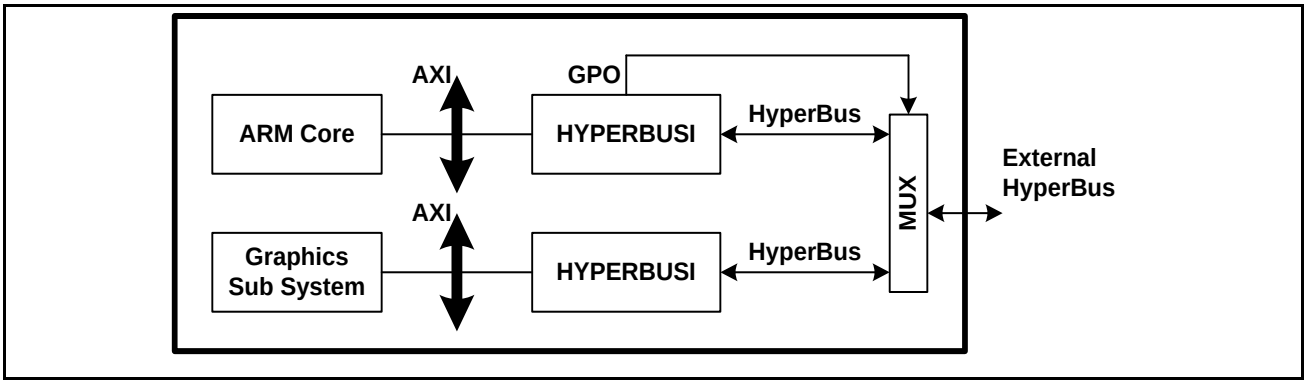
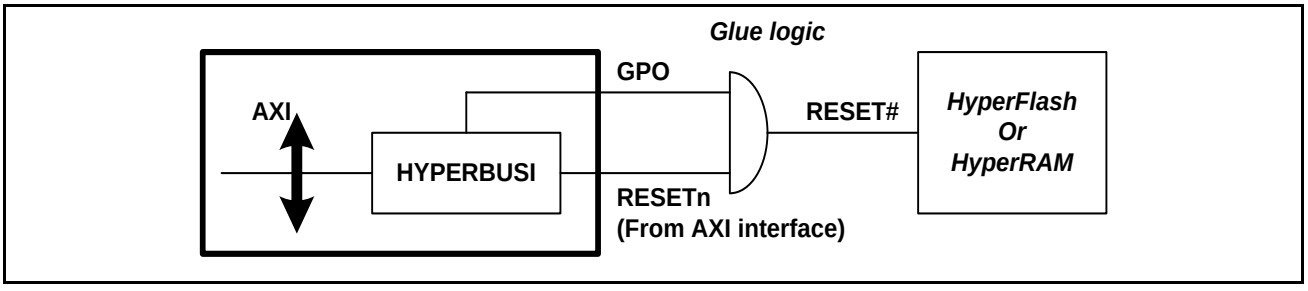


Figure 27-8 External Control Example by GPO



3.6. AXI Slave Interface

HYPERBUSI module has two AXI slave interface for memory spaces access and CS REG access.

And, AXI slave interface outputs the RESETn signal and inputs the RSTOn signal for controlling memory access.

3.6.1. RSTOn

RSTOn signal is allocated to external interface in order to connect RSTO# of HyperBus memory. RSTOn signal is used in order to determine whether AXI transaction is acceptable. When RSTOn is low and AXI transaction comes, HYPERBUSI module replies the SLVERR response.

3.6.2. RESETn

RESETn signal is allocated to external interface in order to connect RESET# of HyperBus memory. When ARESETN signal on AXI interface is low, RESETn signal is low.

4. Registers

This section describes the registers of the HYPERBUSI in detail.

The suffix 'n' in the register name indicates that the register is an instance 'n' of the module.

■ Registers of the HYPERBUSI

The following registers are available for each instance of the HYPERBUSI.

- Controller Status Register (HYPERBUSIn_CSR)
- Interrupt Enable Register (HYPERBUSIn_IEN)
- Interrupt Status Register (HYPERBUSIn_ISR)
- Memory Base Address Register 0..1 (HYPERBUSIn_MBR)
- Memory Configuration Register 0..1 (HYPERBUSIn_MCR)
- Memory Timing Register 0..1 (HYPERBUSIn_MTR)
- General Purpose Output Register (HYPERBUSIn_GPOR)
- Write Protection Register (HYPERBUSIn_WPR)
- Test Register (HYPERBUSIn_TEST)

PERIPHERAL_NAME	HYPERBUSI
ADDRESS_WIDTH	32
ENDIANESS	Little
DOCPREFIX	HYPERBUSI
MODULE_ID	
PPU_NAME	PPU_HYPERBUSI
BASEADDRESS	
ADDR_BLOCK_SIZE	60
USAGE	Register

4.1. Controller Status Register (HYPERBUSIn_CSR)

HYPERBUSI_CSR indicates the internal status of HYPERBUSI module.

REGISTER_NAME	HYPERBUSIn_CSR
OFFSET	0x00
ACCESS_SIZE	B H W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

Table 27-2 HYPERBUSI Controller Status Register Description

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	RFU	RFU	RFU	RFU	RFU	WRSTOERR	WTRSERR	WDECERR
ACCESS_TYPE	R0,WX	R0,WX	R0,WX	R0,WX	R0,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	RFU	RFU	RFU	RFU	RFU	RFU	RFU	WACT
ACCESS_TYPE	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,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	RFU	RFU	RFU	RFU	RDSSTALL	RRSTOERR	RTRSERR	RDECERR
ACCESS_TYPE	R0,WX	R0,WX	R0,WX	R0,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	RFU	RFU	RFU	RFU	RFU	RFU	RFU	RACT
ACCESS_TYPE	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R,WX
PROT_TYPE	-	-	-	-	-	-	-	-
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:27], [bit23:17], [bit15:12], [bit7:1] RFU : Reserved for Future Use

When writing this register, this bit should be set to 0 for future compatibility. When reading this register, this bit is 0 for future compatibility.

[bit26] WRSTOERR : RSTO Error in Write Transaction

This bit indicates whether HyperBus memory is under reset state in the latest write transaction.

Bit[i]	Description
0	Normal operation
1	HyperBus memory is under reset (AXI SLVERR occurs)

[bit25] WTRSERR : Transaction Error in Write Transaction

This bit indicates whether AXI protocol is acceptable by HYPERBUSI module in the latest write transaction.

Bit[i]	Description
0	Normal operation
1	Protocol is not supported (AXI SLVERR occurs)

[bit24] WDECERR : Decode Error in Write Transaction

This bit indicates whether access address is acceptable in the latest write transaction.

Bit[i]	Description
0	Normal operation
1	Access address is not reachable (AXI DECERR occurs)

[bit16] WACT : Write Transaction Active

This bit indicates whether write transaction is in progress.

Bit[i]	Description
0	Write transaction is idle
1	Write transaction is active

[bit11] RDSSTALL : RDS Stall Error in Read Transaction

This bit indicates whether read data error occurs in the latest read transaction.

Bit[i]	Description
0	Normal operation
1	Detect read data error (AXI SLVERR occurs)

[bit10] RRSTOERR : RSTO Error in Read Transaction

This bit indicates whether HyperBus memory is under reset state in the latest read transaction.

Bit[i]	Description
0	Normal operation
1	HyperBus memory is under reset (AXI SLVERR occurs)

[bit9] RTRSERR : Transaction Error in Read Transaction

This bit indicates whether AXI protocol is acceptable by HYPERBUSI module in the latest read transaction.

Bit[i]	Description
0	Normal operation
1	Protocol is not supported (AXI SLVERR occurs)

[bit8] RDECERR : Decode Error in Read Transaction

This bit indicates whether access address is acceptable in the latest read transaction.

Bit[i]	Description
0	Normal operation
1	Access address is not reachable (AXI DECERR occurs)

[bit0] RACT : Read Transaction Active

This bit indicates whether read transaction is in progress.

Bit[i]	Description
0	Read transaction is idle
1	Read transaction is active

4.2. Interrupt Enable Register (HYPERBUSIn_IEN)

. HYPERBUSIn_IEN sets up the condition of interrupt signal output (IENOn)

REGISTER_NAME	HYPERBUSIn_IEN
OFFSET	0x04
ACCESS_SIZE	B H W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

Table 27-3 HYPERBUSI Interrupt Enable Register Description

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	INTP	RFU	RFU	RFU	RFU	RFU	RFU	RFU
ACCESS_TYPE	R/W	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	RFU	RFU	RFU	RFU	RFU	RFU	RFU	RFU
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	15	14	13	12	11	10	9	8
BIT_NAME	RFU	RFU	RFU	RFU	RFU	RFU	RFU	RFU
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	RFU	RFU	RFU	RFU	RFU	RFU	RFU	RPCINTE
ACCESS_TYPE	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R/W
PROT_TYPE	-	-	-	-	-	-	-	-
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31] INTP : Interrupt Polarity

This bit is used to choose the polarity of IENOn signal.

Bit[i]	Description
0	IENOn signal is active low
1	IENOn signal is active high

[bit30:1] RFU : Reserved for Future Use

When writing this register, this bit should be set to 0 for future compatibility. When reading this register, this bit is 0 for future compatibility.

[bit0] RPCINTE : HyperBus Memory Interrupt Enable

This bit enables the interrupt from the HyperBus memory by INTn signal.

Bit[i]	Description
0	Disable
1	Enable

4.3. Interrupt Status Register (HYPERBUSIn_ISR)

HYPERBUSI_ISR indicates the interrupt request source.

REGISTER_NAME	HYPERBUSIn_ISR
OFFSET	0x08
ACCESS_SIZE	B H W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

Table 27-4 HYPERBUSI Interrupt Status Register Description

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved	RFU	RFU	RFU	RFU	RFU	RFU	RFU
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	RFU	RFU	RFU	RFU	RFU	RFU	RFU	RFU
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	15	14	13	12	11	10	9	8
BIT_NAME	RFU	RFU	RFU	RFU	RFU	RFU	RFU	RFU
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	RFU	RFU	RFU	RFU	RFU	RFU	RFU	RPCINTS
ACCESS_TYPE	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:1] RFU : Reserved for Future Use

When writing this register, this bit should be set to 0 for future compatibility. When reading this register, this bit is 0 for future compatibility.

[bit0] RPCINTS : HyperBus memory Interrupt Status

This bit indicates the interrupt from HyperBus memory by INTn signal.

While the INT# signal of the HyperBus memory is asserted, this bit is also asserted. In order to clear this bit, the interrupt of HyperBus memory must be cleared.

Bit[i]	Description
0	No interrupt
1	Interrupt

4.4. Memory Base Address Register (HYPERBUSIn_MBR0 to 1)

The base address for each memory space is set up.

REGISTER_NAME	HYPERBUSIn_MBRi
OFFSET	0x10 + i*4
ACCESS_SIZE	B H W
MULTIPLE	0:1
NUMERIC_TYPE	
OTHER	

Table 27-5 HYPERBUSI Memory Base Address Register Description

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	A[31]	A[30]	A[29]	A[28]	A[27]	A[26]	A[25]	A[24]
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	A[23]	A[22]	A[21]	A[20]	A[19]	A[18]	A[17]	A[16]
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	15	14	13	12	11	10	9	8
BIT_NAME	A[15]	A[14]	A[13]	A[12]	A[11]	A[10]	A[9]	A[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	A[7]	A[6]	A[5]	A[4]	A[3]	A[2]	A[1]	A[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] A[31:0] : Memory Base Address

The base address of addressable region to each memory is set up. Since register can be set in 16 M bytes boundary, lower 24 bit is fixed to 0.

4.5. Memory Configuration Register (HYPERBUSIn_MCR0 to 1)

The memory configuration for each memory space is set up.

REGISTER_NAME	HYPERBUSIn_MCRi
OFFSET	0x20 + i*4
ACCESS_SIZE	B H W
MULTIPLE	0:1
NUMERIC_TYPE	
OTHER	

Table 27-6 HYPERBUSI Memory Configuration Register Description

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	MAXEN	RFU	RFU	RFU	RFU	MAXLEN[8]	MAXLEN[7]	MAXLEN[6]
ACCESS_TYPE	R/W	R0,WX	R0,WX	R0,WX	R0,WX	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	MAXLEN[5]	MAXLEN[4]	MAXLEN[3]	MAXLEN[2]	MAXLEN[1]	MAXLEN[0]	CRMO	ACS
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	RFU	RFU	RFU	RFU	RFU	RFU	RFU	RFU
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	RFU	RFU	CRT	DEVTYPE	RFU	RFU	WRAPSIZE[1]	WRAPSIZE[0]
ACCESS_TYPE	R0,WX	R0,WX	R/W	R/W	R0,WX	R0,WX	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	1	1

[bit31] MAXEN : Maximum Length Enable

This bit enables maximum access data length control by Split Ctrl for restricting HyperBus CS# assertion time by MAXLEN bit. Please refer to section 3.2.3 Split Ctrl for more detailed information.

bit	Description
0	Disable maximum access data length control
1	Enable maximum access data length control

[bit30:27], [bit15:6], [bit3:2] RFU : Reserved for Future Use

When writing this register, this bit should be set to 0 for future compatibility. When reading this register, this bit is 0 for future compatibility.

[bit26:18] MAXLEN[8:0] : Maximum Length

This bit is set as maximum access data length in order to restrict HyperBus CS# assertion time. These bits are ignored when MAXEN bit is 0. Please refer to section 3.2.3 Split Ctrl for more detailed information.

bit[8:0]	Description
000000000	2 Byte (1 HyperBus CK)
000000001	4 Byte (2 HyperBus CK)
000000010	6 Byte (3 HyperBus CK)
: :	: :
111111111	1024 Byte (512 HyperBus CK)

[bit17] CRMO : Continuous Read Merging Option

This bit is set when the wrap transaction and subsequent continuous transaction can be merged. Please confirm whether it is corresponding HyperFlash memory before enabling this function.

bit	Description
0	No merging WRAP and INCR transaction
1	Merging WRAP and INCR transaction

[bit16] ACS : Asymmetry Cache System Support

This bit is set when the different wrap size (cache size) is required by multi-core in system. This function should be disabled if the HyperBus memory itself supports the asymmetry cache system.

bit	Description
0	No asymmetry cache system support
1	Asymmetry cache system support

[bit5] CRT : Configuration Register Target

This bit indicates whether access is to memory space or register space. This bit is mapped to CA[46] bit in command/address cycle to HyperBus memory.

bit	Description
0	Memory space
1	Configuration register space

[bit4] DEVTYPE : Device Type

This bit is set as a device type of connected memory.

bit	Description
0	HyperFlash memory
1	HyperRAM memory

[bit1:0] WRAPSIZE[1:0] : Wrapped Burst Size

This bit is set as wrap burst length which was set to HyperBus memory. This bit is ignored when the ACS bit ([bit16]) is 0. When the ACS bit is 1, this bit should be set the same as wrap size of configuration register in HyperBus memory.

bit[1:0]	Description
00	Reserved
01	64 Bytes
10	16 Bytes
11	32 Bytes

4.6. Memory Timing Register (HYPERBUSIn_MTR0 to 1)

The memory timing for each memory space is set up.

REGISTER_NAME	HYPERBUSIn_MTRi
OFFSET	0x30 + i*4
ACCESS_SIZE	B H W
MULTIPLE	0:1
NUMERIC_TYPE	
OTHER	

Table 27-7 HYPERBUSI Memory Timing Register Description

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	RCSHI[3]	RCSHI[2]	RCSHI[1]	RCSHI[0]	WCSHI[3]	WCSHI[2]	WCSHI[1]	WCSHI[0]
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	RCSS[3]	RCSS[2]	RCSS[1]	RCSS[0]	WCSS[3]	WCSS[2]	WCSS[1]	WCSS[0]
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	RCSH[3]	RCSH[2]	RCSH[1]	RCSH[0]	WCSH[3]	WCSH[2]	WCSH[1]	WCSH[0]
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	RFU	RFU	RFU	RFU	LTCY[3]	LTCY[2]	LTCY[1]	LTCY[0]
ACCESS_TYPE	R0,WX	R0,WX	R0,WX	R0,WX	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	1

[bit31:28] RCSHI[3:0] : Read Chip Select High Between Operations

Before the read access, this bit inserts the CK cycles to the chip select high period.

bit[3:0]	Description
0000	1.5 CK
0001	2.5 CK
0010	3.5 CK
:	:
1111	16.5 CK

[bit27:24] WCSHI[3:0] : Write Chip Select High Between Operations

Before the write access, this bit inserts the CK cycles to the chip select high period.

bit[3:0]	Description
0000	1.5 CK
0001	2.5 CK
0010	3.5 CK
:	:
1111	16.5 CK

[bit23:20] RCSS[3:0] : Read Chip Select Setup to next CK Rising Edge

In the read access, this bit inserts the CK cycles, between the falling edge of chip select and the rising edge of first CK.

bit[3:0]	Description
0000	1 CK
0001	2 CK
0010	3 CK
:	:
1111	16 CK

[bit19:16] WCSS[3:0] : Write Chip Select Setup To Next CK Rising Edge

In the write access, this bit inserts the CK cycles, between the falling edge of chip select and the rising edge of first CK.

bit[3:0]	Description
0000	1 CK
0001	2 CK
0010	3 CK
:	:
1111	16 CK

[bit15:12] RCSH[3:0] : Read Chip Select Hold After CK falling Edge

In the read access, this bit inserts the CK cycles, between the falling edge of last CK and the rising edge of chip select.

bit[3:0]	Description
0000	1 CK
0001	2 CK
0010	3 CK
:	:
1111	16 CK

[bit11:8] WCSH[3:0] : Write Chip Select Hold After CK falling Edge

In the write access, this bit inserts the CK cycles, between the falling edge of last CK and the rising edge of chip select.

bit[3:0]	Description
0000	1 CK
0001	2 CK
0010	3 CK
:	:
:	:
1111	16 CK

[bit7:4] RFU : Reserved for Future Use

When writing this register, this bit should be set to 0 for future compatibility. When reading this register, this bit is 0 for future compatibility.

[bit3:0] LTCY[3:0] : Latency Cycle for HyperRAM mode

When DEVTYPE of HYPERBUSI_MCRn is set to HyperRAM mode, this bit should be set to the same value as the read latency in configuration register of HyperBus memory. This bit is ignored when the DEVTYPE of HYPERBUSI_MCRn is chosen to HyperFlash mode.

bit[3:0]	Description
0000	5 CK Latency
0001	6 CK Latency
0010	Reserved
:	:
:	:
1110	Reserved
1111	4 CK Latency

4.7. General Purpose Output Register (HYPERBUSIn_GPOR)

HYPERBUSI_GPOR is used in order to control the GPIO signals polarity.

REGISTER_NAME	HYPERBUSIn_GPOR
OFFSET	0x40
ACCESS_SIZE	B H W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

Table 27-8 HYPERBUSI General Purpose Output Register Description

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	RFU	RFU	RFU	RFU	RFU	RFU	RFU	RFU
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	RFU	RFU	RFU	RFU	RFU	RFU	RFU	RFU
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	15	14	13	12	11	10	9	8
BIT_NAME	RFU	RFU	RFU	RFU	RFU	RFU	RFU	RFU
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	RFU	RFU	RFU	RFU	RFU	RFU	GPO[1]	GPO[0]
ACCESS_TYPE	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:2] RFU : Reserved for Future Use

When writing this register, this bit should be set to 0 for future compatibility. When reading this register, this bit is 0 for future compatibility.

[bit1:0] GPO[1:0] : General Purpose Output Interface

The signal polarity that is outputted to the general purpose output interface is set up.

Bit[i]	Description
0	Output signal polarity is low
1	Output signal polarity is high

4.8. Write Protection Register (HYPERBUSIn_WPR)

HYPERBUSI_WPR is used in order to control the WPn signal polarity.

REGISTER_NAME	HYPERBUSIn_WPR
OFFSET	0x44
ACCESS_SIZE	B H W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

Table 27-9 HYPERBUSI Write Protection Register Description

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	RFU	RFU	RFU	RFU	RFU	RFU	RFU	RFU
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	RFU	RFU	RFU	RFU	RFU	RFU	RFU	RFU
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	15	14	13	12	11	10	9	8
BIT_NAME	RFU	RFU	RFU	RFU	RFU	RFU	RFU	RFU
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	RFU	RFU	RFU	RFU	RFU	RFU	RFU	WP
ACCESS_TYPE	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:1] RFU : Reserved for Future Use

When writing this register, this bit should be set to 0 for future compatibility. When reading this register, this bit is 0 for future compatibility.

[bit0] WP : Write Protection

This bit is used to control WPn signal.

Bit[i]	Description
0	Not protection (WPn signal is high)
1	Protection (WPn signal is low)

4.9. Test Register (HYPERBUSIn_TEST)

This register is internal device test purpose. When writing this register, it should be set to 0.

REGISTER_NAME	HYPERBUSIn_TEST
OFFSET	0x48
ACCESS_SIZE	B H W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

Table 27-10 HYPERBUSI Test Register Description

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	RFU	RFU	RFU	RFU	RFU	RFU	RFU	RFU
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	RFU	RFU	RFU	RFU	RFU	RFU	RFU	RFU
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	15	14	13	12	11	10	9	8
BIT_NAME	RFU	RFU	RFU	RFU	RFU	RFU	RFU	RFU
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	RFU	RFU	RFU	RFU	RFU	RFU	RFU	RVD
ACCESS_TYPE	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:1] RFU : Reserved for Future Use

When writing this register, this bit should be set to 0 for future compatibility. When reading this register, this bit is 0 for future compatibility.

[bit0] RVD : Reserved bit

This bit is reserved. It should be set to 0.

CHAPTER 28: LCD Controller

This chapter explains the LCD controller.

1. Overview
2. Features
3. Configuration
4. Operation
5. Setting
6. Registers
7. Q&A
8. Sample Program
9. Notes

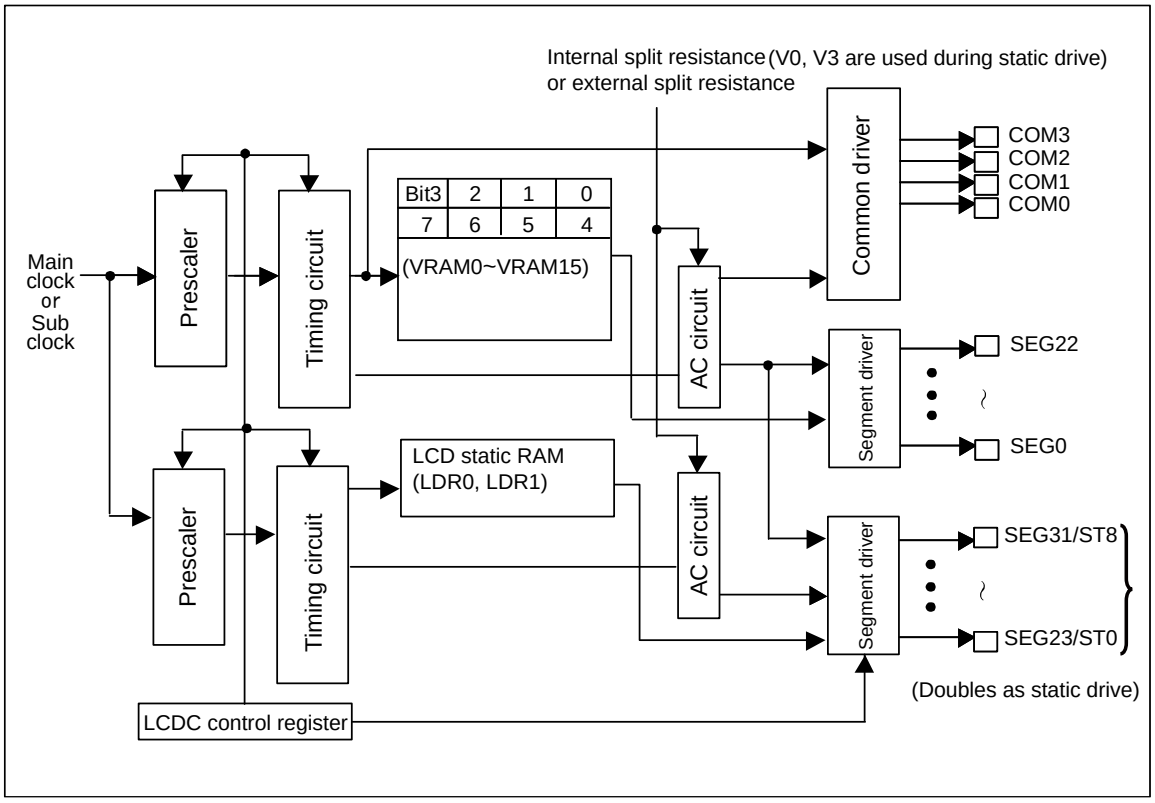
CODE:LCDC-S6J3300-E1

1. Overview

This section explains the overview of the LCD controller.

The LCD enables the selection of duty from 1/2, 1/3, and 1/4, and up to 128-element display.

Furthermore, it enables up to 8-element display as static LCD output.



2. Features

This section explains the features of the LCD controller.

<Duty drive>

- Number: 1 (4-common × 32-segment)
- Display: Up to 128-element (on 1/4 duty)
- Duty: Selection from three types (1/2, 1/3, 1/4)
- Bias: Selection from 1/2 and 1/3

Table 28-1Combination of Bias, Duty, and Common Output

Bias	1/2 Duty Output Mode	1/3 Duty Output Mode	1/4 Duty Output Mode
1/2 bias	○	×	×
1/3 bias	×	○	○

○ : Recommended mode
× : Prohibited

- Frame cycle: Selectable from four types (For a clock, a main clock or sub clock can be used.)
- Driver: Built-in (Internal division resistor), or an external division resistor can be connected to V0 to V3 pins.
- Data memory: Built-in 16-byte data memory for display
- No display selection: Enabled
- Pin: COM0 to COM3, SEG0 to SEG31, V0, V1, V2, V3 pins are used for general-purpose ports also and switching is enabled.
  SEG23 to SEG31 pins are used as static drive pins (ST0 to ST8) also and switching is enabled.
  VCC can be selected instead of V3.
- During duty driving, up to 4 common outputs (COM0 to COM3) and 32 segment outputs (SEG0 to SEG31) can be used.
- Others: The external division resistor also can block a current when LCD is stopped.

< Static drive >

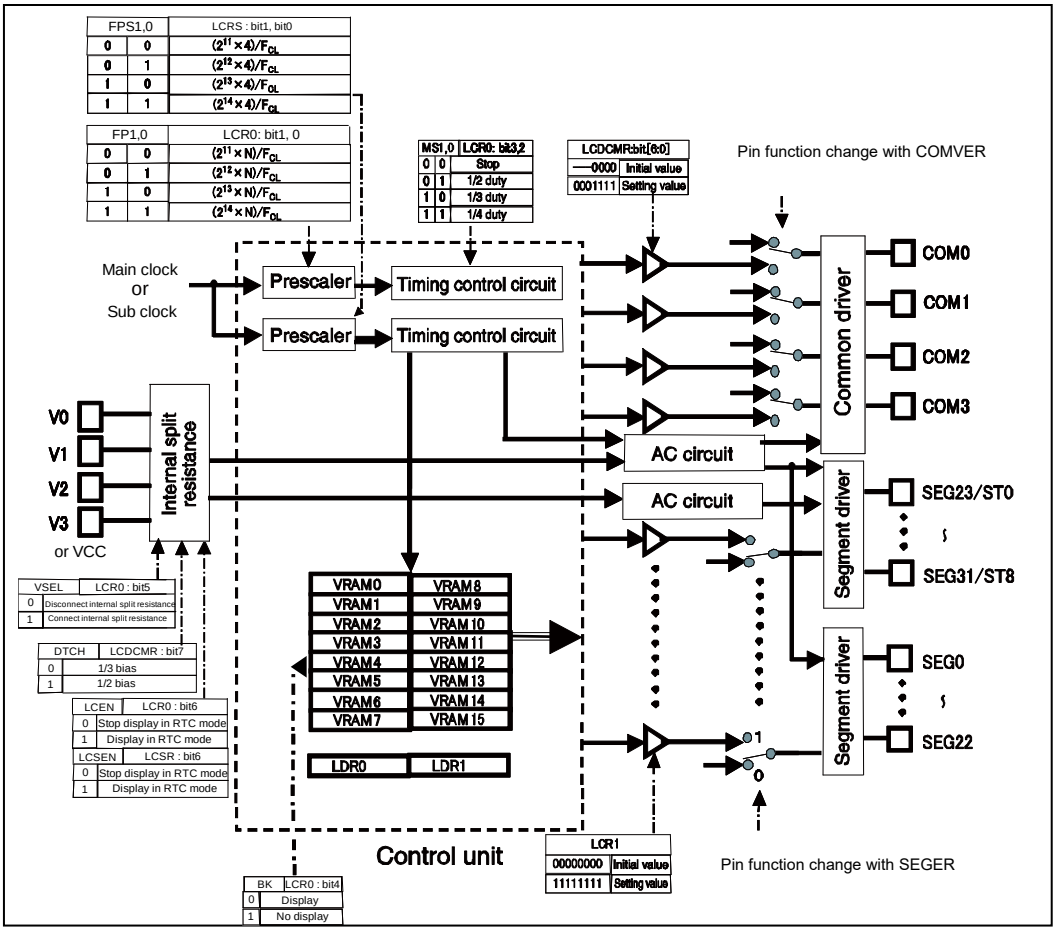
- Number: 1 (1-common × 8-segment)
- Display: Up to 8 element
- Frame cycle: Selectable from four types (For a clock, a main clock or sub clock can be used.)
- Pin: ST0 to ST8 pins are used also as general-purpose ports and duty drive pins (SEG23 to SEG31) and switching is enabled.
- During static driving, up to 1 static common output and 8 static segment outputs (ST0 to ST8) can be used.

3. Configuration

This section shows the configuration of the LCD controller.

- Configuration Diagram of LCD Controller
- Figure 28-1 shows the configuration diagram of the LCD controller.

Figure 28-1 Configuration Diagram



4. Operation

The following sections explain the operation of the LCD controller.

4.1. Operation of LCD Cintroller/Driver (LCDC)
4.2. 1/2 Duty Outpuy Waveform
4.3. 1/3 Duty Outpuy Waveform
4.4. 1/4 Duty Outpuy Waveform
4.5. Static Drive Output Waveform

4.1. Operation of LCD Controller/Driver (LCDC)

The operation of LCD controller/driver (LCDC) is shown below.

(1)Data memory for display

<Duty drive>

Set a value to the data memory for display (VRAM) in advance.

<Static drive>

Set a value to the data memory for display (LDR0, LDR1) in advance.

(2)Write necessary settings to each register.

(3)Output pin

<Duty drive>

When a clock for frame cycle generation oscillates, waveforms that drive the LCD are output to the common/segment output pins (COM0 to COM3, SEG0 to SEG31).

The content of the VRAM is read automatically in synchronization with the timing of the common signal and output from the segment output pin.

(When the bit is set to "1", a selection waveform is output from the segment output pin.

When the bit is set to "0", a non-selection waveform is output from the segment output pin.)

Non-selection level waveforms are output from the COM2 and COM3 pins in 1/2 duty display mode and from the COM3 pin in 1/3 duty display mode.

<Static drive>

When a clock for frame cycle generation oscillates, waveforms that drive the LCD are output to the common/segment output pins (ST0 to ST8).

(4)Output waveform

<Duty drive>

Output waveforms drive in 2 frames alternating waveform according to the duty setting.

<Static drive>

Output waveforms drive in 1 frame alternating waveform.

(5)Operation in the PSS timer mode (main oscillation operation/ sub oscillation operation)

<Duty drive>

For operation enable (LCEN = "1"), the LCD is displayed.

Please do not stop the clock oscillation which selected by the CSS bit during the transition to PSS mode.

LCD display stop when power shutdown.

<Static drive>

For operation enable (LCSEN = "1"), the LCD is displayed.

Please do not stop the clock oscillation which selected by the SCSS bit during the transition to PSS mode.

LCD display stop when power shutdown

(6)Blanking function

<Duty drive>

Light of the LCD can be turned off with selection of non-display (BK = "1") of blanking.

However, non-selection waveforms are output.

(7)LCD stop state

<Duty drive>

When display operation of the LCD is stopped (MS [1:0] = "00"), both the common/segment output pins become "L" level.

<Static drive>

When display operation of the LCD is stopped (LCS [3:0] = "0000"), both the common/segment output pins become "L" level.

When the LCD with static drive (ST0 to ST8) is switched from display to non-display, set LDR0:ST8 = "0" and LDR1: ST [7:0] = "00000000" without change of the static drive selection port (LCS [3:0]).

The same potential pulses are output from the static drive pins (ST0 to ST8).

4.2. 1/2 Duty Output Waveform

1/2 duty output waveform is shown below.

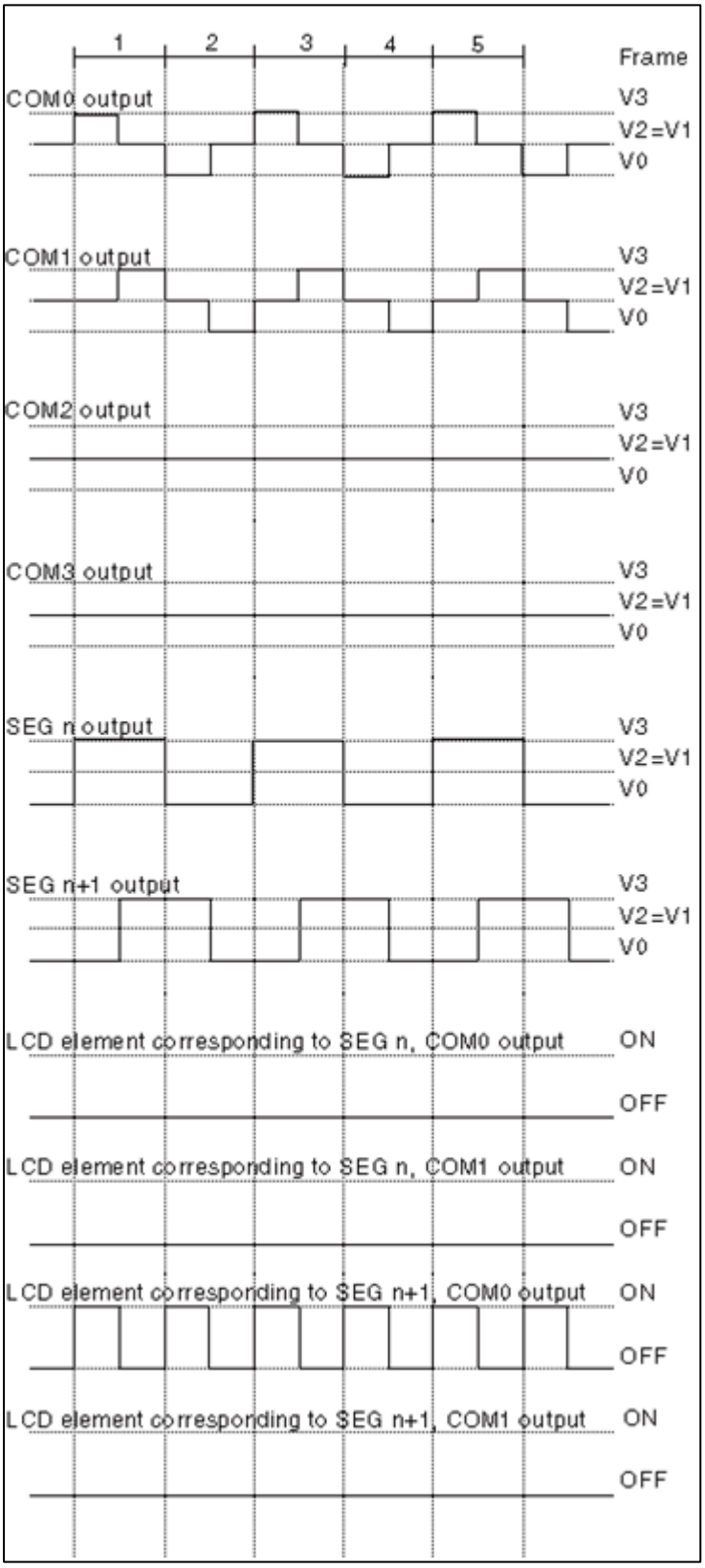
Only COM0 output and COM1 output are used for LCD display. COM2 output and COM3 output are not used.

1/2 Bias Output Waveform Case

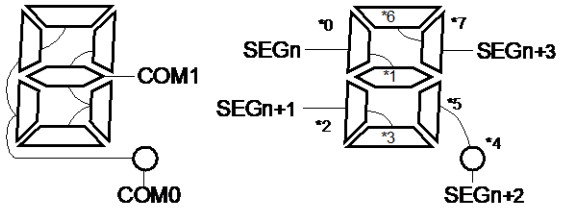
Liquid crystal elements that have maximum difference in potential between the common output and segment output light up.

■ Example of Content of Data Memory for Display

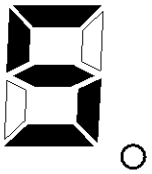
Segment	Content of Data Memory for Display			
	COM3 Output	COM2 Output	COM1 Output	COM0 Output
SEG n output	-	-	0	0
SEG n+1 output	-	-	0	1


1
2
3
4
5
Frame
COM0 output
V3
V2=V1
V0
COM1 output
V3
V2=V1
V0
COM2 output
V3
V2=V1
V0
COM3 output
V3
V2=V1
V0
SEG n output
V3
V2=V1
V0
SEG n+1 output
V3
V2=V1
V0
LCD element corresponding to SEG n, COM0 output
ON
OFF
LCD element corresponding to SEG n, COM1 output
ON
OFF
LCD element corresponding to SEG n+1, COM0 output
ON
OFF
LCD element corresponding to SEG n+1, COM1 output
ON
OFF

■ LCD panel connection case and display data case (1/2 duty drive method)



Example for displaying 5:

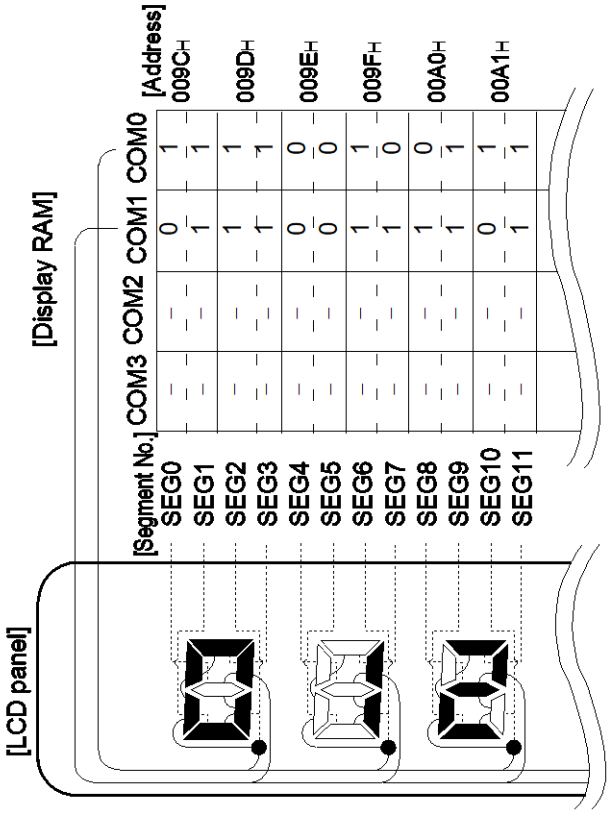


Address	COM3	COM2	COM1	COM0	
nH	bit3	bit2	bit1*1	bit0*0	SEGn
	bit7	bit6	bit5*3	bit4*2	SEGn+1
n+1H	bit3	bit2	bit1*5	bit0*4	SEGn+2
	bit7	bit6	bit5*7	bit4*6	SEGn+3

*0 to *7 : Indicates correspondence with display RAM
bit2, 3, 6, 7 are unused

Address	COM3	COM2	COM1	COM0	
3950H	–	–	1	1	SEG0
	–	–	1	0	SEG1
3951H	–	–	1	0	SEG2
	–	–	0	1	SEG3

0 : OFF
1 : ON



LCD display	Example of data correspondence from 0 to 9							
	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0	–	–	1	1	–	–	0	1
	–	–	1	1	–	–	1	1
1	–	–	0	0	–	–	0	0
	–	–	1	0	–	–	1	1
2	–	–	1	1	–	–	1	0
	–		1	1			0	1
3	–	–	1	0	–	–	1	0
	–	–	1	1	–	–	1	1
4	–	–	0	0	–	–	1	1
	–	–	1	0	–	–	1	1
5	–	–	1	0	–	–	1	1
	–	–	0	1	–	–	1	1
6	–	–	1	1	–	–	1	1
	–	–	0	1	–	–	1	1
7	–	–	0	0	–	–	0	1
	–	–	1	1	–	–	1	1
8	–	–	1	1	–	–	1	1
	–	–	1	1	–	–	1	1
9	–	–	1	0	–	–	1	1
	–	–	1	1	–	–	1	1

4.3. 1/3 Duty Output Waveform

1/3 duty output waveform is shown below.

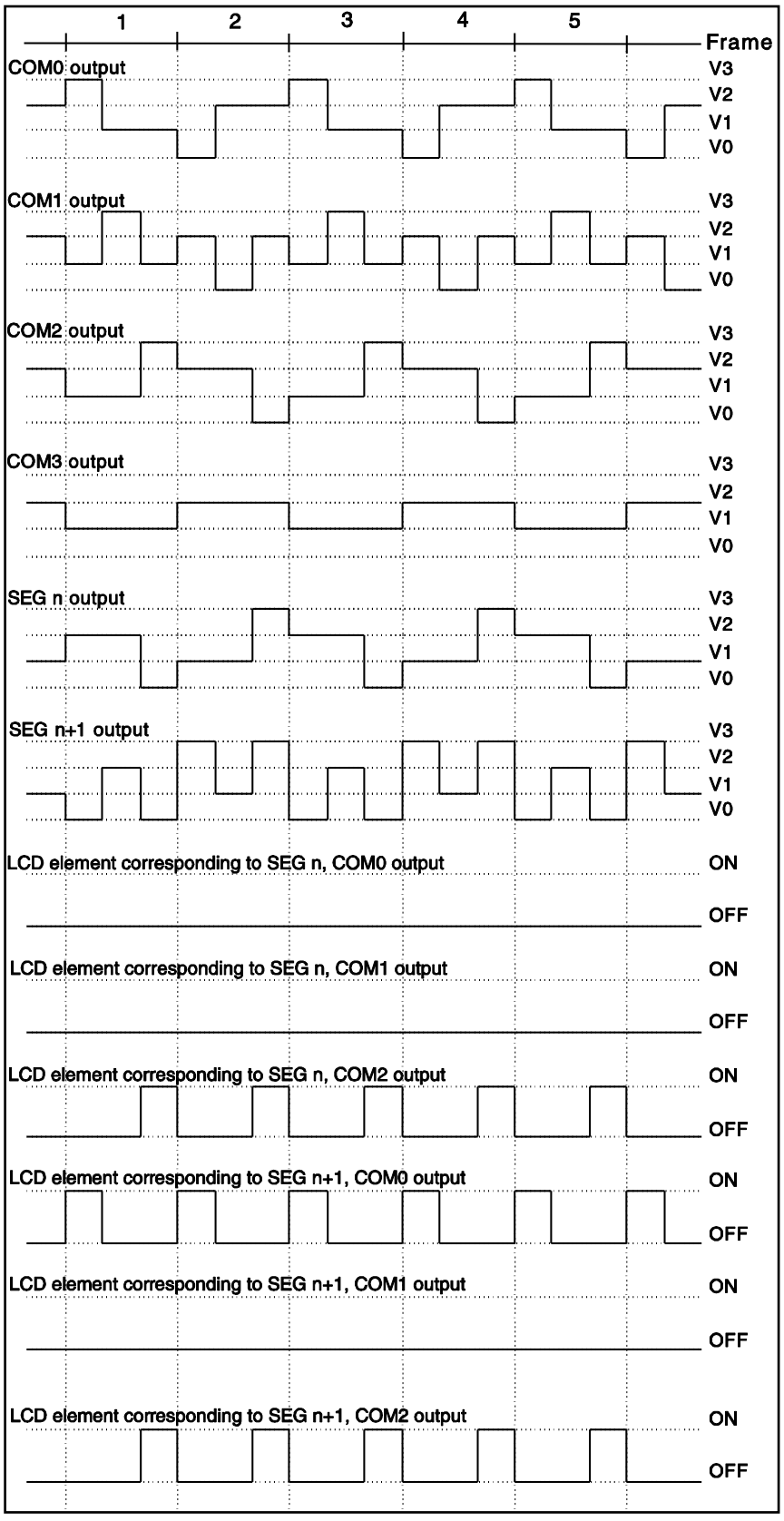
For 1/3 duty output mode, COM0 output, COM1 output, and COM2 output are used for LCD display. COM3 output is not used.

1/3 Bias Output Waveform Case

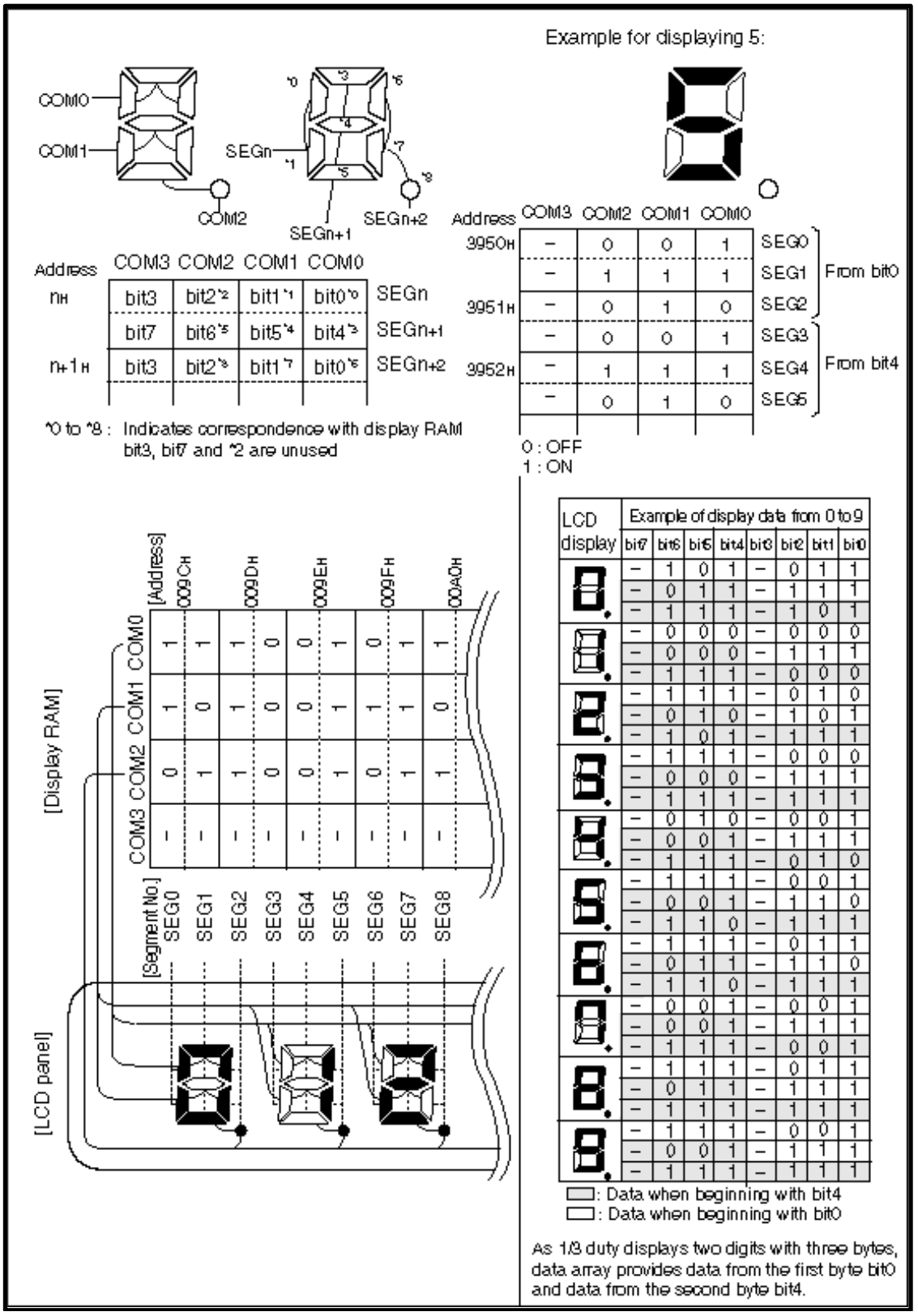
Liquid crystal elements that have maximum difference in potential between the common output and segment output light up.

■ Example of Content of Data Memory for Display

Segment	Content of Data Memory for Display			
	COM3 Output	COM2 Output	COM1 Output	COM0 Output
SEG n output	-	1	0	0
SEG n+1 output	-	1	0	1


1
2
3
4
5
Frame
COM0 output
COM1 output
COM2 output
COM3 output
SEG n output
SEG n+1 output
V3
V2
V1
V0
LCD element corresponding to SEG n, COM0 output
LCD element corresponding to SEG n, COM1 output
LCD element corresponding to SEG n, COM2 output
LCD element corresponding to SEG n+1, COM0 output
LCD element corresponding to SEG n+1, COM1 output
LCD element corresponding to SEG n+1, COM2 output
ON
OFF

■ LCD panel connection case and display data case (1/3 duty drive method)



4.4. 1/4 Duty Output Waveform

1/4 duty output waveform is shown below.

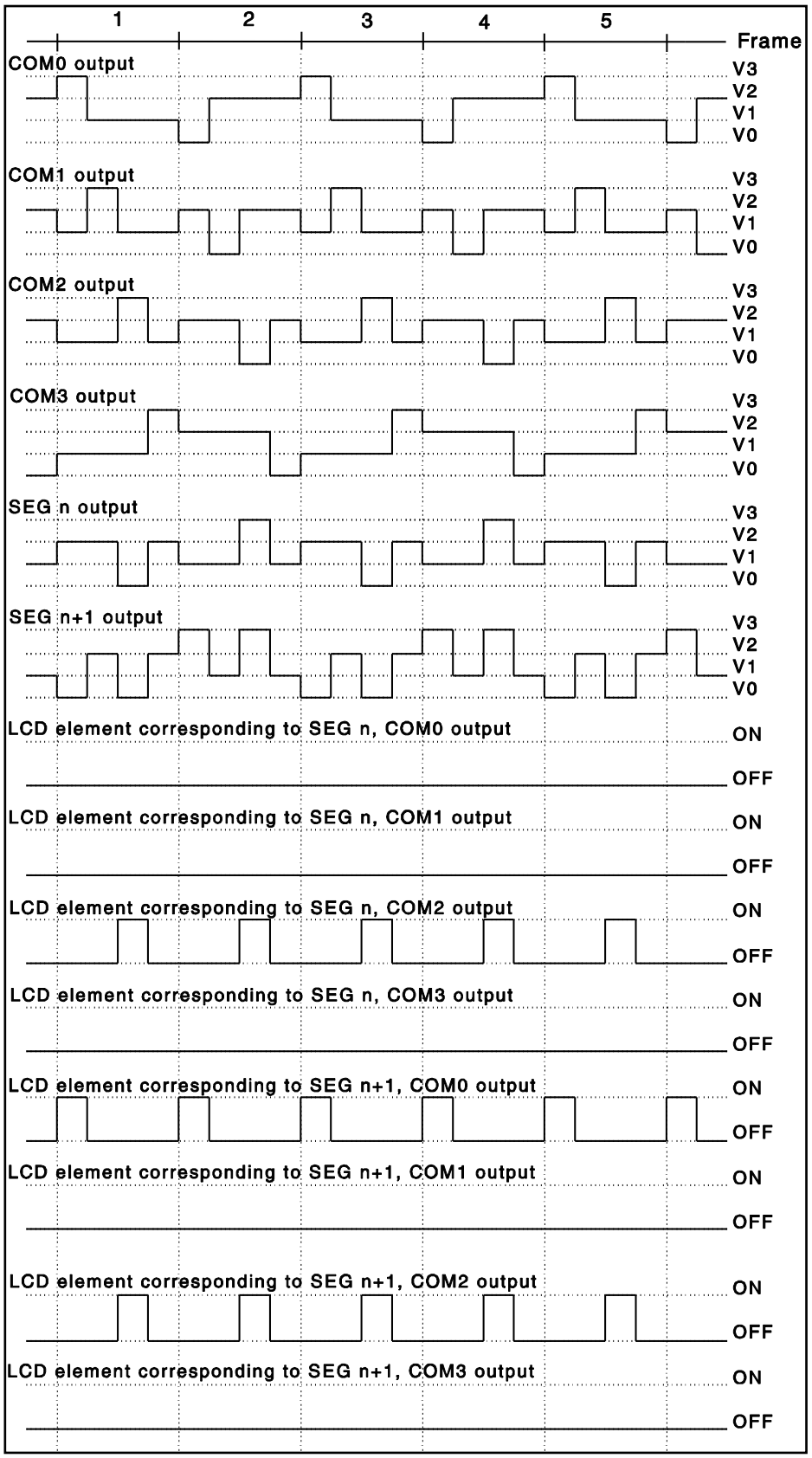
For 1/4 duty output mode, all of COM0 output, COM1 output, COM2 output, and COM3 output are used for LCD display.

1/4 Bias Output Waveform Case

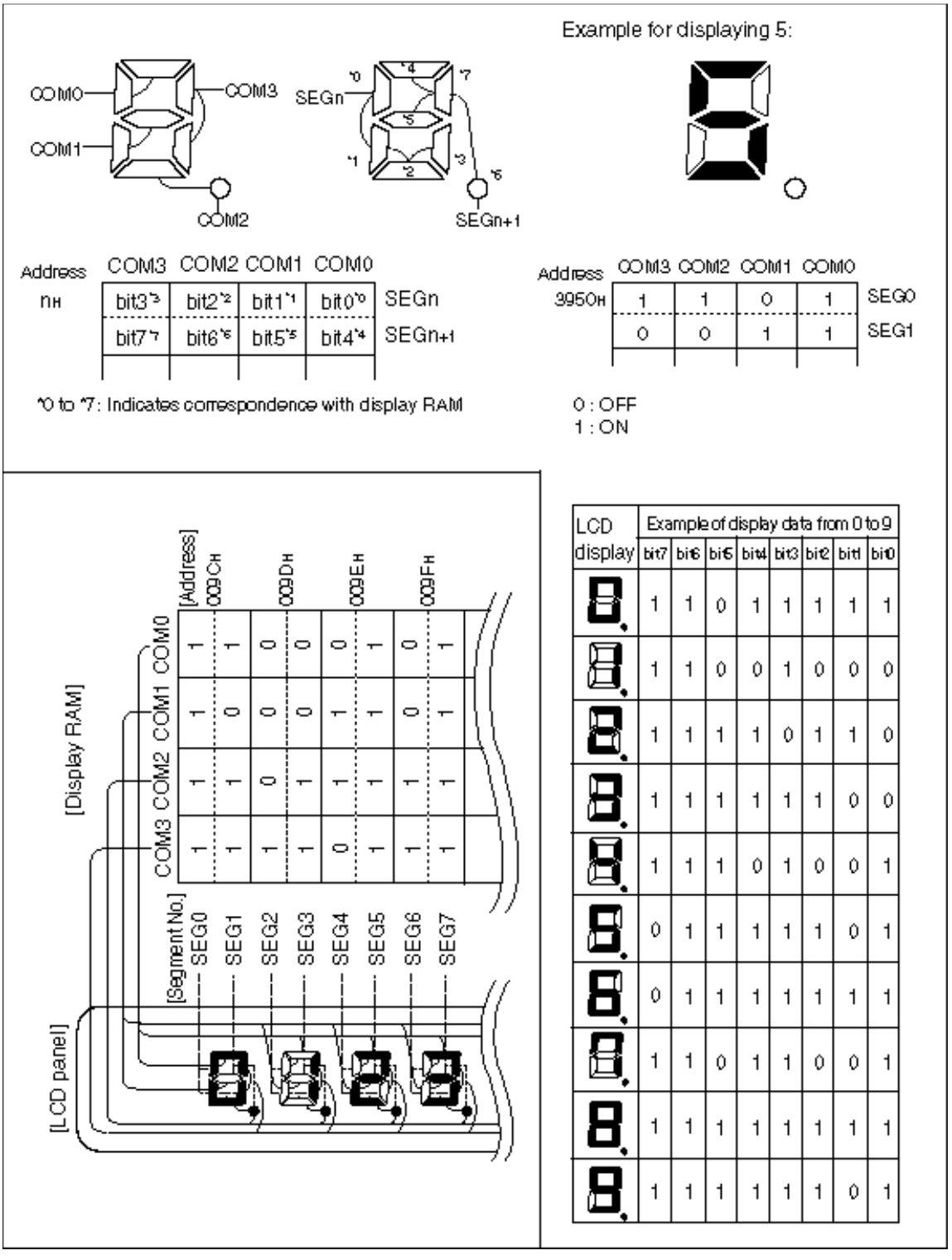
For LCD, liquid crystal elements that have maximum difference in potential between the common output and segment output light up.

■ Example of Content of Data Memory for Display

Segment	Content of Data Memory for Display			
	COM3 Output	COM2 Output	COM1 Output	COM0 Output
SEG n output	0	1	0	0
SEG n+1 output	0	1	0	1


1
2
3
4
5
Frame
COM0 output
COM1 output
COM2 output
COM3 output
SEG n output
SEG n+1 output
V3
V2
V1
V0
LCD element corresponding to SEG n, COM0 output
LCD element corresponding to SEG n, COM1 output
LCD element corresponding to SEG n, COM2 output
LCD element corresponding to SEG n, COM3 output
LCD element corresponding to SEG n+1, COM0 output
LCD element corresponding to SEG n+1, COM1 output
LCD element corresponding to SEG n+1, COM2 output
LCD element corresponding to SEG n+1, COM3 output
ON
OFF

■ LCD panel connection case and display data case (1/4 duty drive method)



LCD display	Example of display data from 0 to 9							
	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0	1	1	0	1	1	1	1	1
1	1	1	0	0	1	0	0	0
2	1	1	1	1	0	1	1	0
3	1	1	1	1	1	1	0	0
4	1	1	1	0	1	0	0	1
5	0	1	1	1	1	1	0	1
6	0	1	1	1	1	1	1	1
7	1	1	0	1	1	0	0	1
8	1	1	1	1	1	1	1	1
9	1	1	1	1	1	1	0	1

4.5. Static Drive Output Waveform

Static drive output waveform is shown below.

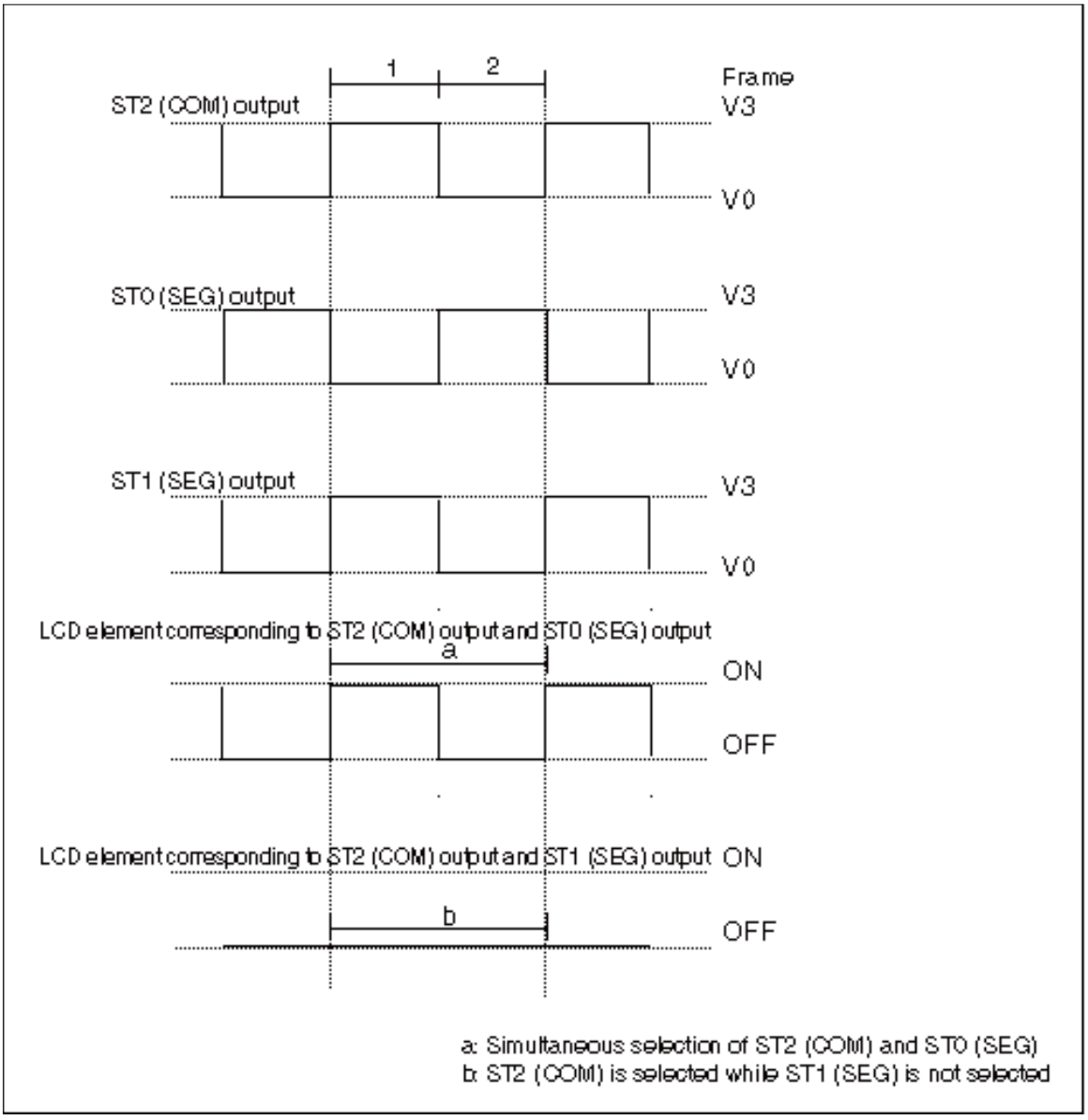
For the static drive output mode, the common/segment output pins (ST0 to ST8) are used for LCD display.

Static Drive Output Waveform Case

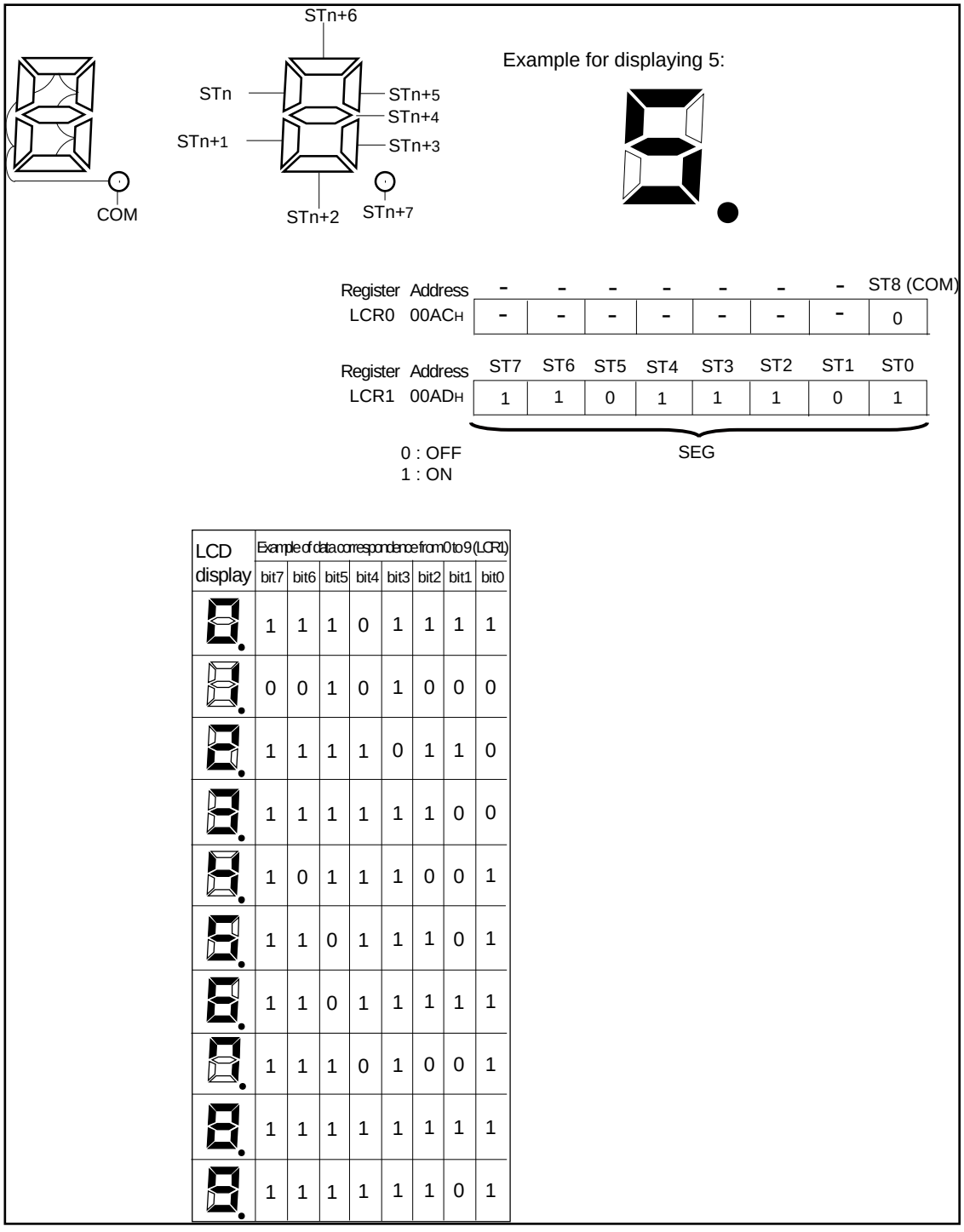
For LCD, liquid crystal elements that have maximum difference in potential between the common output and segment output light up.

- Example of Content of Data Memory for Display (Example of static SEG output to the output pins ST0 to ST1 and static COM output to ST2)

Content of Data Memory for Display (LDR0, LDR1)		
ST2 Output LDR0[2]	**ST1 Output LDR0[1]**	**ST0 Output LDR0[0]**
0	0	1


1
2
Frame
ST2 (COM) output
V3
V0
ST0 (SEG) output
V3
V0
ST1 (SEG) output
V3
V0
LCD element corresponding to ST2 (COM) output and ST0 (SEG) output
a
ON
OFF
LCD element corresponding to ST2 (COM) output and ST1 (SEG) output
ON
b
OFF
a: Simultaneous selection of ST2 (COM) and ST0 (SEG)
b: ST2 (COM) is selected while ST1 (SEG) is not selected

■ LCD panel connection case and display data case (Static drive method)



LCD display	Example of data correspondence from 0 to 9 (LCR1)							
	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0.	1	1	1	0	1	1	1	1
1.	0	0	1	0	1	0	0	0
2.	1	1	1	1	0	1	1	0
3.	1	1	1	1	1	1	0	0
4.	1	0	1	1	1	0	0	1
5.	1	1	0	1	1	1	0	1
6.	1	1	0	1	1	1	1	1
7.	1	1	1	0	1	0	0	1
8.	1	1	1	1	1	1	1	1
9.	1	1	1	1	1	1	0	1

5. Setting

This section shows the setting of the LCD controller.

<Duty drive>

Setting Necessary for Use of LCD

Setting	Set Register	Set Method
Preliminary setting	LCD control register 1 (LCR1) Common pin switching register (LCDCMR)	See 6.3, 6.4
Division resistor setting	LCD control register 0 (LCR0)	See 7.9, 7.11
Port setting	Segment output register (SEGER) Common output V pin control register(COMVER)	See 6.8, 6.9
Display data setting	Data memory for display (VRAM)	See 7.2
Frame cycle setting	LCD control register 0 (LCR0)	See 7.3
Duty selection (activation)		See 7.5
Display selection		See 7.7

Setting Necessary for Cancellation of LCD Display

Setting	Set Register	Set Method
Non-display selection	LCD control register 0 (LCR0)	See 7.7

Setting Necessary for LCD Operation Stop

Setting	Set Register	Set Method
LCD operation stop	LCD control register 0 (LCR0)	See 7.6

Setting Necessary for LCD Display during PSS Timer Mode (Main Oscillation Operation/ Sub Oscillation Operation)

Setting	Set Register	Set Method
Display selection during PSS timer mode(main oscillation operation/ sub oscillation operation)	LCD control register 0 (LCR0)	See 7.8
Transition to PSS timer mode(main oscillation operation/ sub oscillation operation)	See chapter of "LOW-POWER CONSUMPTION".	-

< Static drive >

Setting Necessary for Use of LCD

Setting	Set Register	Set Method
Preliminary setting	LCD control register 1 (LCR1) Common pin switching register (LCDCMR)	See 6.3, 6.4
Port setting	Set the pin to LCDC output. See chapter of "I/O PORTS".	See 6.8, 6.9
	LCD control register (LCRS)	See 6.5
Display data setting	Data memory for display (LDR0,LDR1)	See 6.6
Frame cycle setting	LCD control register (LCRS)	See 7.3

Setting Necessary for LCD Operation Stop

Setting	Set Register	Set Method
LCD operation stop	LCD control register (LCRS)	See 7.6

Setting Necessary for LCD Display during PSS Timer Mode (Main Oscillation Operation/ Sub Oscillation Operation)

Setting	Set Register	Set Method
Display selection during PSS timer mode(main oscillation operation/ sub oscillation operation)	LCD control register (LCRS)	See 7.8
Transition to PSS timer mode(main oscillation operation/ sub oscillation operation)	See chapter of "LOW-POWER CONSUMPTION".	-

6. Registers

This section explains the registers of the LCD controller.

■ Register map

Table 28-2 Registers of the LCD Controller

Register Abbreviation	Register Name	Reference
LCR0	LCD control register 0	6.1
VRAMn(n=0 to15)	Data memory for display	6.2
LCR1	LCDC control register 1	6.3
LCDCMR	Common pin switching register	6.4
LCRS	LCDC Static control register	6.5
LDR	Static LCD display data register	6.6

Table 28-3 Registers of LCD Port-Related

Register Abbreviation	Register Name	Reference
LCD_KEYCDR	key code register	6.7
SEGER	Segment output Register	6.8
COMVER	Common output V pin control register	6.9

■ **Register Offset Address Map (LCDC)**

OFFSET_ADDRESS	REGISTER_NAME				ACCESS_SIZE
	+3	+2	+1	+0	
0x00000000	LCDCMR	LCRS	LCR0	LCR1	B, H, W
0x00000004	VRAM0	VRAM1	VRAM2	VRAM3	B, H, W
0x00000008	VRAM4	VRAM5	VRAM6	VRAM7	B, H, W
0x0000000C	VRAM8	VRAM9	VRAM10	VRAM11	B, H, W
0x00000010	VRAM12	VRAM13	VRAM14	VRAM15	B, H, W
0x00000014	LDR		Reserved		B, H, W

■ **Register Offset Address Map (LCDE)**

OFFSET_ADDRESS	REGISTER_NAME				ACCESS_SIZE
	+3	+2	+1	+0	
0x00000000	LCD_KEYCDR				W
0x00000004	SEGER				B, H, W
0x00000008	COMVER				B, H, W

6.1. LCD Control Register 0: LCR0

The bit configuration of the LCD control register 0 is shown below.

This register selects a frame cycle with its clock to be used, selects display mode, enables the LCD controller to operate in the condition for selecting display/non-display mode, and controls LCD drive power.

■ LCR0

bit	7	6	5	4	3	2	1	0
Field	CSS	LCEN	VSEL	BK	MS1	MS0	FP1	FP0
Attribute	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Protection attribute	-	-	-	-	-	-	-	-
Initial value	0	0	0	1	0	0	0	0

[bit7] CSS: Clock selection bit

Select a clock to be used for this module.

CSS	Operation
0	Main clock
1	Sub clock

For the single clock products, set "0".

[bit6] LCEN: PSS Timer mode (main oscillation operation / sub oscillation operation) operation enable

LCEN	Operation
0	LCD display stop in the PSS timer mode(main oscillation operation/ sub oscillation operation)
1	LCD display in the PSS timer mode(main oscillation operation/ sub oscillation operation)

When LCEN="1", do not stop the clock oscillation which selected by the CSS bit during the transition to PSS mode.

LCD display stop when power shutdown.

[bit5] VSEL: LCD drive power control

VSEL	Operation
0	Internal division resistor disconnection
1	Internal division resistor connection

When an external division resistor is to be connected, the LCD drive power control bit (VSEL) must be set to "0".

[bit4] BK: Blanking selection

BK	Operation
0	LCD display
1	LCD no display

[bit3, bit2] MS1, MS0: Display mode selection

MS1	MS0	Display mode
0	0	LCD operation stop
0	1	1/2 duty output mode (Time division number: N=2, COM0, COM1)
1	0	1/3 duty output mode (Time division number: N=3, COM0 to COM2)
1	1	1/4 duty output mode (Time division number: N=4, COM0 to COM3)

When the display mode selection bits (MS [1:0]) are set to "00", the LCD controller's operation is stopped, and the common pin/segment pin output "L" level.

[bit1, bit0] FP1, FP0: Frame cycle

FP1	FP0	Frame cycle
0	0	For CSS=0 : (211 × N)/FCL For CSS=1 : (23 × N)/FCL
0	1	For CSS=0 : (212 × N)/FCL For CSS=1 : (24 × N)/FCL
1	0	For CSS=0 : (213 × N)/FCL For CSS=1 : (25 × N)/FCL
1	1	For CSS=0 : (214 × N)/FCL For CSS=1 : (26 × N)/FCL

FCL : Main clock (LCR0:CSS=0) or sub clock (LCR0:CSS=1)

N : Time division number (depends on the setting of the display mode selection bits MS1, MS0)

Select the setting for FP1 and FP0 that is the optimum condition for the frame frequency of the LCD panel to be used.

6.2. Data Memory for Display: VRAM

The data memory for display is shown below.

Memory (VRAM) area for setting of data for display

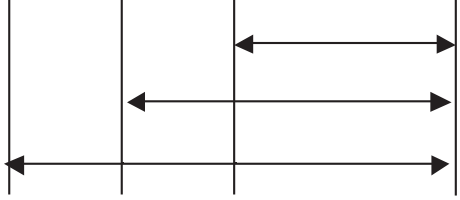
- VRAM0 (SEG0, SEG1)
- VRAM1 (SEG2, SEG3)
- VRAM2 (SEG4, SEG5)
- VRAM3 (SEG6, SEG7)
- VRAM4 (SEG8, SEG9)
- VRAM5 (SEG10, SEG11)
- VRAM6 (SEG12, SEG13)
- VRAM7 (SEG14, SEG15)
- VRAM8 (SEG16, SEG17)
- VRAM9 (SEG18, SEG19)
- VRAM10 (SEG20, SEG21)
- VRAM11 (SEG22, SEG23)
- VRAM12 (SEG24, SEG25)
- VRAM13 (SEG26, SEG27)
- VRAM14 (SEG28, SEG29)
- VRAM15 (SEG30, SEG31)

bit	7	6	5	4	3	2	1	0
Field	D07	D06	D05	D04	D03	D02	D01	D00
Attribute	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Protection attribute	-	-	-	-	-	-	-	-
Initial value	0	0	0	0	0	0	0	0

The RAM for display can be read /written with any timing regardless of the LCD controller/driver operation.

Correspondence between VRAM and Common Pins/Segment Pins

VRAM0	bit3	bit2	bit1	bit0	SEG0
	bit7	bit6	bit5	bit4	SEG1
VRAM1	bit3	bit2	bit1	bit0	SEG2
	bit7	bit6	bit5	bit4	SEG3
VRAM2	bit3	bit2	bit1	bit0	SEG4
	bit7	bit6	bit5	bit4	SEG5
VRAM3	bit3	bit2	bit1	bit0	SEG6
	bit7	bit6	bit5	bit4	SEG7
VRAM4	bit3	bit2	bit1	bit0	SEG8
	bit7	bit6	bit5	bit4	SEG9
VRAM5	bit3	bit2	bit1	bit0	SEG10
	bit7	bit6	bit5	bit4	SEG11
VRAM6	bit3	bit2	bit1	bit0	SEG12
	bit7	bit6	bit5	bit4	SEG13
VRAM7	bit3	bit2	bit1	bit0	SEG14
	bit7	bit6	bit5	bit4	SEG15
VRAM8	bit3	bit2	bit1	bit0	SEG16
	bit7	bit6	bit5	bit4	SEG17
VRAM9	bit3	bit2	bit1	bit0	SEG18
	bit7	bit6	bit5	bit4	SEG19
VRAM10	bit3	bit2	bit1	bit0	SEG20
	bit7	bit6	bit5	bit4	SEG21
VRAM11	bit3	bit2	bit1	bit0	SEG22
	bit7	bit6	bit5	bit4	SEG23
VRAM12	bit3	bit2	bit1	bit0	SEG24
	bit7	bit6	bit5	bit4	SEG25
VRAM13	bit3	bit2	bit1	bit0	SEG26
	bit7	bit6	bit5	bit4	SEG27
VRAM14	bit3	bit2	bit1	bit0	SEG28
	bit7	bit6	bit5	bit4	SEG29
VRAM15	bit3	bit2	bit1	bit0	SEG30
	bit7	bit6	bit5	bit4	SEG31
	COM3	COM2	COM1	COM0	



RAM area and a common pin used for 1/2 duty output mode

RAM area and a common pin used for 1/3 duty output mode

RAM area and a common pin used for 1/4 duty output mode

6.3. LCDC Control Register 1: LCR1

The bit configuration of the LCDC control register 1 is shown below.

■ LCR1

bit	7	6	5	4	3	2	1	0
Field	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
Attribute	R/W1	R/W1	R/W1	R/W1	R/W1	R/W1	R/W1	R/W1
Protection attribute	-	-	-	-	-	-	-	-
Initial value	0	0	0	0	0	0	0	0

[bit7 to bit0] Reserved

When the LCD is used, set "11111111" always. When it is not used, set "00000000".

6.4. Common Pin Switching Register: LCDCMR

The bit configuration of the common pin switching register is shown below.

■ LCDCMR

bit	7	6	5	4	3	2	1	0
Field	DTCH	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
Attribute	R/W	R/W0	R/W0	R/W0	R/W1	R/W1	R/W1	R/W1
Protection attribute	-	-	-	-	-	-	-	-
Initial value	0	0	0	0	0	0	0	0

[bit7] DTCH: Bias selection

DTCH	Operation
0	1/3 bias
1	1/2 bias

[bit6 to bit4] Reserved

When the LCD is used, set "000" always.

[bit3 to bit0] Reserved

When the LCD is used, set "1111" always. When it is not used, set "0000".

6.5. LCDC Static Control Register: LCRS

The bit configuration of the LCDC static control register is shown below.

■ LCRS

bit	7	6	5	4	3	2	1	0
Field	SCSS	LCSEN	LCS3	LCS2	LCS1	LCS0	FPS1	FPS0
Attribute	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Protection attribute	-	-	-	-	-	-	-	-
Initial value	0	0	0	0	0	0	0	0

[bit7] SCSS: Frame cycle generation clock for static drive selection bit

SCSS	Operation
0	Main clock
1	Sub clock

For the single clock products, set "0".

[bit6] LCSEN: PSS Timer mode (main oscillation operation / sub oscillation operation) operation enable

LCSEN	Operation
0	LCD display stop in the PSS timer mode(main oscillation operation/ sub oscillation operation)
1	LCD display in the PSS timer mode(main oscillation operation/ sub oscillation operation)

When LCSEN="1", please do not stop the clock oscillation which selected by the SCSS bit during the transition to PSS mode.

LCD display stop when power shutdown.

[bit5 to bit2] LCS3 to LCS0: Static drive selection

LCS3	LCS2	LCS1	LCS0	Static drive selection port
				Common / Segment output
0	0	0	0	OFF
0	0	0	1	ST0, ST1
0	0	1	0	ST0 to ST2
0	0	1	1	ST0 to ST3
0	1	0	0	ST0 to ST4
0	1	0	1	ST0 to ST5

LCS3	LCS2	LCS1	LCS0	Static drive selection port
				Common / Segment output
0	1	1	0	ST0 to ST6
0	1	1	1	ST0 to ST7
1	X	X	X	ST0 to ST8

Note:

- *For the use method for the static drive selection, refer to "4.5 Static Drive Output Waveform.*

[bit1, bit0] FPS1, FPS0: Frame cycle

FPS1	FPS0	Frame cycle
0	0	For SCSS=0 : (211 × 4)/FCL For SCSS=1 : (23 × 4)/FCL
0	1	For SCSS=0 : (212 × 4)/FCL For SCSS=1 : (24 × 4)/FCL
1	0	For SCSS=0 : (213 × 4)/FCL For SCSS=1 : (25 × 4)/FCL
1	1	For SCSS=0 : (214 × 4)/FCL For SCSS=1 : (26 × 4)/FCL

FCL : Main clock (LCRS:SCSS=0) or sub clock (LCRS:SCSS=1)

Select the setting for FPS1 and FPS0 that is the optimum condition for the frame frequency of the LCD panel to be used.

The frame cycle during static drive is equal to the cycle of the 1/4 duty output mode.

The static drive is enabled when SEGER setting and any setting other than LCS [3:0] = "0000" are executed.

The duty drive is enabled when SEGER and LCS [3:0] = "0000" are set.

6.6. Static LCD Display Data Register: LDR

The bit configuration of the static LCD display data register is shown below.

■ LDR0

bit	15	14	13	12	11	10	9	8
Field	-		-	-	-	-	-	ST8
Attribute	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R/W
Protection attribute	-	-	-	-	-	-	-	-
Initial value	0	0	0	0	0	0	0	0

[bit7 to bit1] Undefined bits

The read value is always "0". Writing has no effect on operation.

[bit0] ST8: Static output data

This bit is static output data for ST8.

■ LDR1 : Address $05BD_H$ (Access : Byte, Half-word, Word)

bit	7	6	5	4	3	2	1	0
Field	ST7	ST6	ST5	ST4	ST3	ST2	ST1	ST0
Attribute	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Protection attribute	-	-	-	-	-	-	-	-
Initial value	0	0	0	0	0	0	0	0

[bit7 to bit0] ST7 to ST0: Static output data

These bits are static output data for ST0 to ST7.

Static Output Data Correspondence Table

LDR	Operation
ST8	ST8 static output data
ST7	ST7 static output data
ST6	ST6 static output data
ST5	ST5 static output data
ST4	ST4 static output data
ST3	ST3 static output data
ST2	ST2 static output data
ST1	ST1 static output data
ST0	ST0 static output data

This register is for setting of data output to the LCD static drive port (set by the LCRS register). The LCD static drive is executed by inverting output of the specified data periodically.

Common output pins are not specified especially. Assign common output to one of segments. For example, if ST0 to ST8 are assigned static drive pins, ST8 is a common pin and set to LDR0:ST8 = "0". Furthermore, by setting LDR1: ST [7:0] = "11111111", the LCD selected with ST0 to ST7 is displayed.

6.7. Key Code Register: LCD_KEYCDR

This section shows the bit configuration of the key code register.

This register makes the write setting of a register that has the function for protection against erroneous writing (SEGER, COMVER). 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.

■ KEYCDR

bit	31	30	29	28	27	26	25	24
Field	KEY1	KEY0	SIZE1	SIZE0	Reserved			
Attribute	R0,W	R0,W	R0,W	R0,W	R0,WX	R0,WX	R0,WX	R0,WX
Protection attribute	-	-	-	-	-	-	-	-
Initial value	0	0	0	0	0	0	0	0
bit	23	22	21	20	19	18	17	16
Field	Reserved							
Attribute	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX
Protection attribute	-	-	-	-	-	-	-	-
Initial value	0	0	0	0	0	0	0	0
bit	15	14	13	12	11	10	9	8
Field	Reserved				ADR11	ADR10	ADR09	ADR08
Attribute	R0,WX	R0,WX	R0,WX	R0,WX	R0,W	R0,W	R0,W	R0,W
Protection attribute	-	-	-	-	-	-	-	-
Initial value	0	0	0	0	0	0	0	0
bit	7	6	5	4	3	2	1	0
Field	ADR07	ADR06	ADR05	ADR04	ADR03	ADR02	ADR01	ADR00
Attribute	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W	R0,W
Protection attribute	-	-	-	-	-	-	-	-
Initial value	0	0	0	0	0	0	0	0

[bit31 to bit30] KEY1 to KEY0: Key code setting bits

You need to 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, you need to set the key codes again from the beginning.

bit31:30		Description
0	0	1st key code
0	1	2nd key code
1	0	3rd key code
1	1	4th key code

[bit29 to bit28] SIZE [1:0]: Access size setting bits

These bits set the access size for writing to a key code target register.

Write the same data to these bits when writing the key codes of "00", "01", "10", and "11" to KEY [1:0] in this order.

bit29:28		Description
0	0	Set byte access.
0	1	Set half-word access.
1	0	Set word access.
1	1	Reserved

[bit27 to bit12] Reserved

Always write "0" to these bits.

[bit11 to bit0] ADR11 to ADR0: Address specification bits

These bits set the lower 12 bits of the address of a key code target register. Write the same data to these bits when writing the key codes of "00", "01", "10", and "11" to KEY [1:0] in this order.

bit11:0	Description
Set the lower 12 bits of the address of a key code target register.	

Note:

The following are the notes on key code setting.

- *You need to write "00", "01", "10", and "11" continuously to KEY [1:0] in this order in 4 write operations.*
  The key code setting becomes invalid immediately upon any writing to these bits in a different order. In such cases, you need to set the key codes again from the beginning.
- *You need to write the same pair of values to SIZE [1:0] and ADR [11:0] 4 times when writing data to KEY [1:0] 4 times.*
  The key code setting becomes invalid upon any writing of a different pair of values. In such cases, you need to set the key codes again from the beginning.
- *When the KEYCDR register is read during writing to KEY [1:0], the key code setting becomes invalid. In such cases, you need to set the key codes again from the beginning.*
- *When access to the address set in ADR [11:0] occurs during writing to KEY [1:0], the key code setting becomes invalid. In such cases, you need to set the key codes again from the beginning.*
  When access to any register other than the above occurs, the key code setting does not become invalid.
  Writing the remaining data to KEY [1:0] subsequently makes the key code setting valid.
- *Once the address set in KEYCDR: ADR [11:0] is accessed after KEY [1:0] is set normally, it is required to set the key codes again to access the address again.*

6.8. Segment Output Register: SEGER

This section shows the bit configuration of the Segment output Register.

Segment output register (SEGER) is a register which control the segment output enable of LCD controller duty driving driver.

■ SEGER

bit	31	30	29	28	27	26	25	24
Field	SEGE7	SEGE6	SEGE5	SEGE4	SEGE3	SEGE2	SEGE1	SEGE0
Attribute	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Protection attribute	-	-	-	-	-	-	-	-
Initial value	0	0	0	0	0	0	0	0

bit	23	22	21	20	19	18	17	16
Field	SEGE15	SEGE14	SEGE13	SEGE12	SEGE11	SEGE10	SEGE9	SEGE8
Attribute	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Protection attribute	-	-	-	-	-	-	-	-
Initial value	0	0	0	0	0	0	0	0

bit	15	14	13	12	11	10	9	8
Field	SEGE23	SEGE22	SEGE21	SEGE20	SEGE19	SEGE18	SEGE17	SEGE16
Attribute	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Protection attribute	-	-	-	-	-	-	-	-
Initial value	0	0	0	0	0	0	0	0

bit	7	6	5	4	3	2	1	0
Field	SEGE31	SEGE30	SEGE29	SEGE28	SEGE27	SEGE26	SEGE25	SEGE24
Attribute	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Protection attribute	-	-	-	-	-	-	-	-
Initial value	0	0	0	0	0	0	0	0

[bit31 to bit0] SEGEn (n=0 to 31)

SEGEn(n=0 to 31)	Operation
0	Disable LCDC segment output
1	Enable LCDC segment output

6.9. Common Output V Pin Control Register: COMVER

This section shows the bit configuration of the common output V pin control register.

Common output V pin control register (COMVER) is a register which control the common output enable of LCD controller static driving driver, input enable and selection of LCD controller reference voltage.

■ COMVER

bit	31	30	29	28	27	26	25	24
Field	VE3	VE2	VE1	VE0	COME3	COME2	COME1	COME0
Attribute	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Protection attribute	-	-	-	-	-	-	-	-
Initial value	0	0	0	0	0	0	0	0

bit	23	22	21	20	19	18	17	16
Field	Reserved							
Attribute	R1,WX	R1,WX	R1,WX	R1,WX	R1,WX	R1,WX	R1,WX	R1,WX
Protection attribute	-	-	-	-	-	-	-	-
Initial value	1	1	1	1	1	1	1	1

bit	15	14	13	12	11	10	9	8
Field	Reserved							
Attribute	R1,WX	R1,WX	R1,WX	R1,WX	R1,WX	R1,WX	R1,WX	R1,WX
Protection attribute	-	-	-	-	-	-	-	-
Initial value	1	1	1	1	1	1	1	1

bit	7	6	5	4	3	2	1	0
Field	Reserved							
Attribute	R1,WX	R1,WX	R1,WX	R1,WX	R1,WX	R1,WX	R1,WX	R1,WX
Protection attribute	-	-	-	-	-	-	-	-
Initial value	1	1	1	1	1	1	1	1

[bit31] VE3

VE3	Operation
1	Use V3 pin as the LCDC V3 reference voltage input
0	VCC is used for the LCDC V3 reference voltage

Please to switch on the main oscillation state when switching external V3 and VCC as LCDC reference supply V3 input.

[bit30 to bit28] VEn (n=0 to 2)

VEn(n=0 to 2)	Operation
0	Disable LCDC reference voltage input
1	Enable LCDC reference voltage input

[bit27 to bit24] COMEn (n=0 to 3)

COMEn(n=0 to 3)	Operation
0	Disable LCDC common output
1	Enable LCDC common output

[bit23 to bit0] Reserved

The read value is always "1". Writing has no effect on operation.

7. Q&A

This section shows Q&A of the LCD controller.

7.1 How can I Set Pins to COM Output Pins or SEG Output Pins?
7.2 How to Set VRAM?
7.3 How can I Setting the Frame Cycle?
7.4 How can I Set the Bias?
7.5 How can I Set the Duty?
7.6 How can I Control the LCD Operation Start/Stop?
7.7 How can I Execute/Cancel the Display?
7.8 How can I Display during the PSS Timer Mode (Main Oscillation Operation/ Sub Oscillation Operation)?
7.9 How can I Select Either Internal or External for the Division Resistor?
7.10 How can I Select Pin of V3 Voltage?
7.11 How can I Select Either Internal or External for the Division Resistor?
7.12 How can I Adjust the Brightness when the Internal Division Resistor is Used?
7.13 How can I Block the Current with the External Division Resistor when the LCD Stops?
7.14 How can I Display/Non-Display the LCD with Static Drive (ST0 to ST8)?

7.1. How can I Set Pins to COM Output Pins or SEG Output Pins?

This section shows how to set pins to COM output pins or SEG output pins.

COM output and SEG output setting.

With software setting, ports can be switched to COM, SEG output.

Set pins for COM output and SEG output to peripheral output. See chapter of "I/O PORTS" for pin setting.

<Duty drive>

Pin	Pin Setting Method	PFR Register / Bit Number
V0	Set pins to LCDC V0/V1/V2/V3 (LCDC reference power input). See 6.9	See "Segment output register" and "Common output V pin control register ".
V1		
V2		
V3		
COM0		
COM1		
COM2		
COM3		
SEG0	See 6.8	
SEG1		
SEG2		
SEG3		
SEG4		
SEG5		
SEG6		
SEG7		
SEG8		
SEG9		
SEG10		
SEG11		
SEG12		
SEG13		
SEG14		
SEG15		
SEG16		
SEG17		
SEG18		

Pin	Pin Setting Method	PFR Register / Bit Number
SEG19	See 6.8	See "Segment output register"
SEG20		
SEG21		
SEG22		
SEG23/ST0		
SEG24/ST1		
SEG25/ST2		
SEG26/ST3		
SEG27/ST4		
SEG28/ST5		
SEG29/ST6		
SEG30/ST7		
SEG31/ST8		

< Static drive >

COM output and SEG output setting.

With software setting, ports can be switched to COM, SEG output.

Set pins for static drive to peripheral output. See 6.8. for pin setting.

Pin	Pin Setting Method	PFR Register / Bit Number
V0	Set pins to LCDC V0/V3 (LCDC reference power input). See 6.9	See "Segment output register" and "Common output V pin control register ".
V3		
SEG23/ST0	See 6.8	
SEG24/ST1		
SEG25/ST2		
SEG26/ST3		
SEG27/ST4		
SEG28/ST5		
SEG29/ST6		
SEG30/ST7		
SEG31/ST8		

7.2. How to Set VRAM?

This section shows how to set VRAM.

<Duty drive>

The matrix of pins and bit locations of VRAM (n) is shown below. (n = 0 to 15)

1/2 Duty

Pin	COM1	COM0
SEG 2n	bit1	bit0
SEG 2n+1	bit5	bit4

1/3 Duty

Pin	COM2	COM1	COM0
SEG 2n	bit2	bit1	bit0
SEG 2n+1	bit6	bit5	bit4

1/4 Duty

Pin	COM3	COM2	COM1	COM0
SEG 2n	bit3	bit2	bit1	bit0
SEG 2n+1	bit7	bit6	bit5	bit4

(Non-selection waveforms are output from irrelevant pins.)

Example: 1/4 duty

When the bit6 of VRAMn is set to "1", the selection waveform is output from the SEGn+1 of COM2.

For bits with "0"setting, non-selection waveforms are output to pins.

7.3. How can I Setting the Frame Cycle?

This section shows how to set pins to COM output pins or SEG output pins.

<Duty drive>

The frame cycle can be set with the frame cycle bits (LCR0: FP [1:0]).The following settings are available.

Frame Cycle (When a Main Clock is Selected)	Selection Value
	Frame Cycle Bits (FP[1:0])
(2^{11} × N)/ Main clock frequency	Set "00".
(2^{12} × N)/ Main clock frequency	Set "01".
(2^{13} × N)/ Main clock frequency	Set "10".
(2^{14} × N)/ Main clock frequency	Set "11".

N (Time division number) = Value of MS [1:0] + "1"

Frame Cycle (When a Sub Clock is Selected)	Selection Value
	Frame Cycle Bits (FP[1:0])
(2^{3} × N)/ Sub clock frequency	Set "00".
(2^{4} × N)/ Sub clock frequency	Set "01".
(2^{5} × N)/ Sub clock frequency	Set "10".
(2^{6} × N)/ Sub clock frequency	Set "11".

N (Time division number) = Value of MS [1:0] + "1"

< Static drive >

The frame cycle can be set with the frame cycle bits (LCRS: FPS [1:0]).The following settings are available.

Frame Cycle	Selection Value
	Frame Cycle Bits (FPS[1:0])
(2^{11} × 4)/Main clock (FCL)	Set "00".
(2^{12} × 4)/ Main clock (FCL)	Set "01".
(2^{13} × 4)/ Main clock (FCL)	Set "10".
(2^{14} × 4)/ Main clock (FCL)	Set "11".

7.4. How can I Set the Bias?

This section shows how to set the bias.

<Duty drive>

Set the bias selection bit (LCDCMR: DTCH).

Bias	Bias Selection Bit (DTCH)
To set 1/3 bias	Set "0".
To set 1/2 bias	Set "1".

7.5. How can I Set the Duty?

This section shows how to set the duty.

<Duty drive>

Set the display mode selection bits (LCR0: MS [1:0]).

Control Detail	Display Mode Selection Bits (MS[1:0])	N (Time Division Number)
LCD operation stop (Pin output "L")	Set "00".	-
To set 1/2 duty output mode	Set "01".	2
To set 1/3 duty output mode	Set "10".	3
To set 1/4 duty output mode	Set "11".	4

The display mode selection bits are also the control bit for operation start/stop.

7.6. How can I Control the LCD Operation Start/Stop?

This section shows how to control the LCD operation start/stop.

<Duty drive>

Operation start/stop can be controlled with the display mode selection bits (LCR0: MS [1:0]).

See "7.5 How can I set the duty?".

<Static drive>

Operation start/stop can be controlled with the display mode selection bits (LCRS: LCS [3:0]).

7.7. How can I Execute/Cancel the Display?

This section shows how to execute/cancel the LCD display.

<Duty drive>

The two methods below are available.

■ Setting of the blanking selection bit (LCR0:BK)

Control Detail	Blanking Selection Bit (BK)
To execute LCD display	Set "0".
To cancel LCD display (A non-selection waveform is output to a segment pin.)	Set "1".

■ Cancellation of display with operation stop by the display mode selection bits (LCR0: MS [1:0]).

Control Detail	Display Mode Selection Bits (MS[1:0])
LCD operation stop ("L" output from the common pin and segment pin.)	Set "00".

7.8. How can I Display during the PSS Timer Mode (Main Oscillation Operation/ Sub Oscillation Operation)?

This section shows how to display during the PSS timer mode (main oscillation operation/ sub oscillation operation).

<Duty drive>

Set the PSS timer mode (main oscillation operation/ sub oscillation operation) operation enable bit (LCR0: LCEN).

Control Detail	PSS Timer Mode(Main Oscillation Operation/ Sub Oscillation Operation) Operation Enable Bit (LCEN)
For no LCD display in the PSS timer mode(main oscillation operation/ sub oscillation operation)	Set "0".
For LCD display in the PSS timer mode(main oscillation operation/ sub oscillation operation)	Set "1".

When LCEN="1", please do not stop the clock oscillation which selected by the CSS bit during the transition to PSS mode. LCD display stop when power shutdown.

<Static drive>

Set the PSS timer mode (main oscillation operation/ sub oscillation operation) operation enable bit (LCRS: LCSEN).

Control Detail	PSS Timer Mode(Main Oscillation Operation/ Sub Oscillation Operation) Operation Enable Bit (LCSEN)
For no LCD display in the PSS timer mode(main oscillation operation/ sub oscillation operation)	Set "0".
For LCD display in the PSS timer mode(main oscillation operation/ sub oscillation operation)	Set "1".

When LCSEN="1", please do not stop the clock oscillation which selected by the SCSS bit during the transition to PSS mode. LCD display stop when power shutdown.

7.9. How can I Select Either Internal or External for the Division Resistor?

This section shows how to select either internal or external for the division resistor.

<Duty drive>

Set the LCD drive power control bit (LCR0: VSEL).

Control Detail	LCD Drive Power Control Bit (VSEL)
For use of the external division resistor (The internal division resistor is disconnected.)	Set "0".
For use of the internal division resistor (The internal division resistor is connected.)	Set "1".

7.10. How can I Select Pin of V3 Voltage?

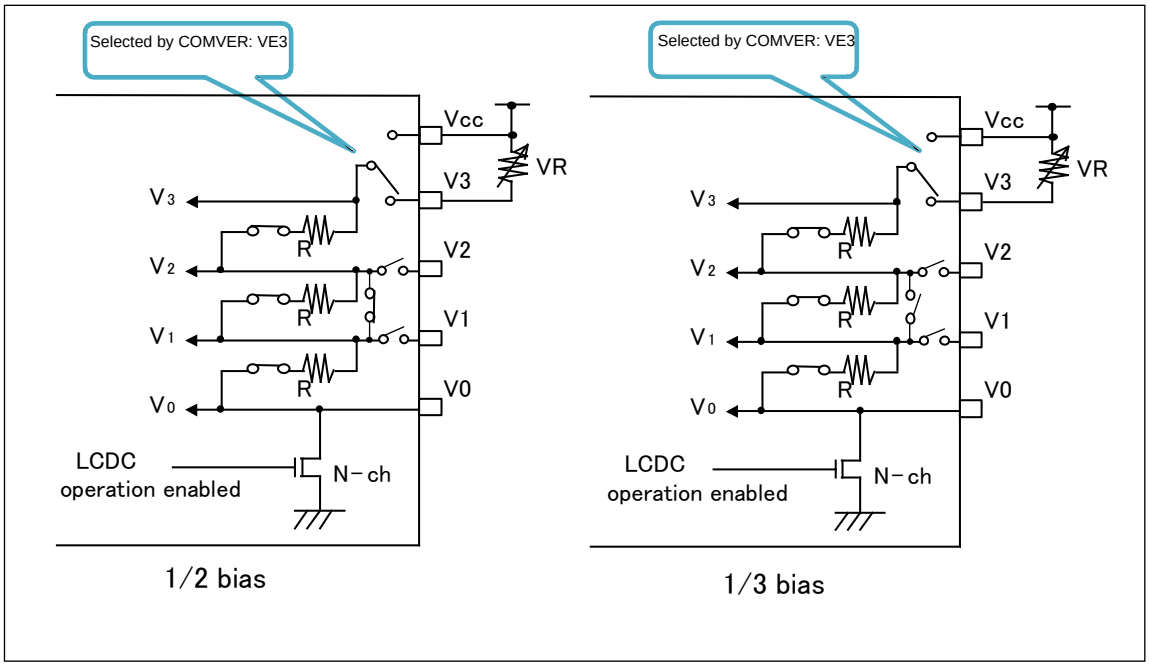
This section shows how to select the pin of V3 voltage.

Set the COMVER: VE3 bit. See chapter of 6.9 for details.

7.11. How can I Select Either Internal or External for the Division Resistor?

This section shows how to select either internal or external for the division resistor.

- When the internal division resistor is selected
- Even when the internal division resistor is used if the V3 pin is used as V3 voltage, the external resistor must be connected between Vcc and V3.



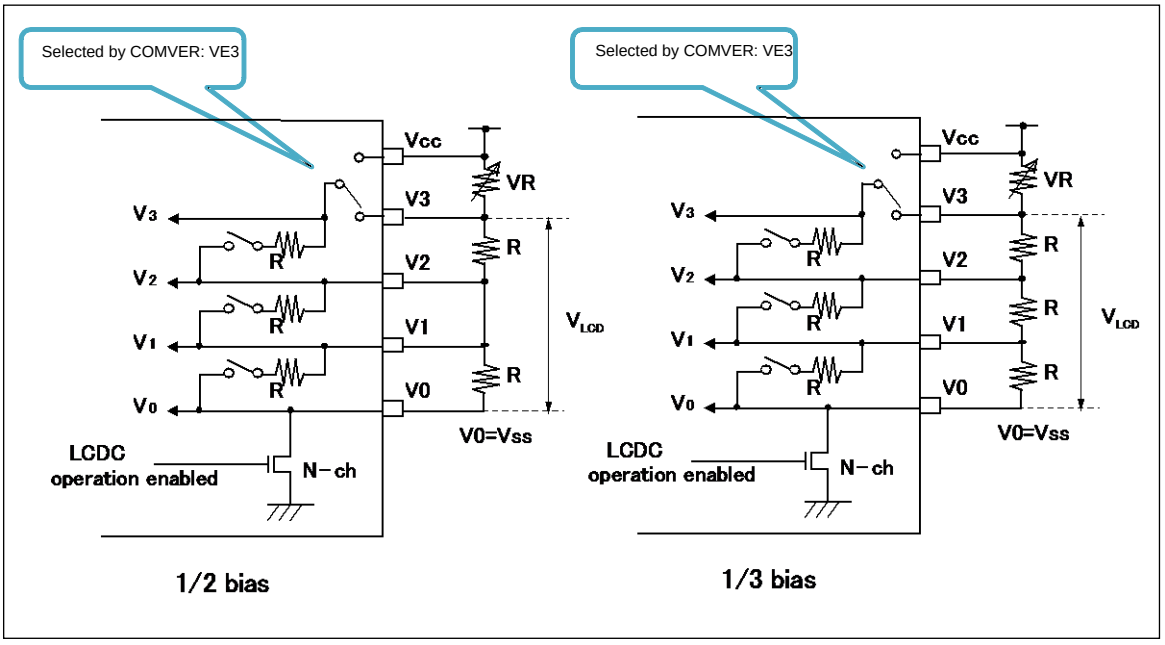
- When the external division resistor is selected
- Voltage for the LCD drive is set with the external division resistor connected to the power pins for LCD drive (V0 to V3).

Setting of LCD drive voltage

	V3	V2	V1	V0
1/2 bias	VLCD	1/2 VLCD	1/2 VLCD	VSS
1/3 bias	VLCD	2/3 VLCD	1/3 VLCD	VSS

V0 to V3 : Voltage of V0 to V3 pins

VLCD : Operation voltage of the LCD



- To avoid the effect of the internal division resistor, the LCD drive power control bit (LCR0: VSEL) must be set to "0" to disconnect the internal division resistor.

7.12. How can I Adjust the Brightness when the Internal Division Resistor is Used?

This section shows how to adjust the brightness when the internal division resistor is used.

If desired brightness cannot be obtained with the use of the internal division resistor, adjust the voltage V3 by putting a variable resister (VR) between the outer terminals Vcc and V3.

7.13. How can I Block the Current with the External Division Resistor when the LCD Stops?

This section shows how to block the current with the external division resistor when the LCD stops.

The V0 pin is connected to the Vss (GND) via a transistor internally. Therefore, when the external division resistor is used, by connecting the Vss side of the external division resistor to the V0 pin, the current which flows on the LCD controller which is stopped can be blocked. Block the current with the display mode selection bits (MS [1:0] = "00").

7.14. How can I Display/Non-Display the LCD with Static Drive (ST0 to ST8)?

This section shows how to display/non-display the LCD with static drive (ST0 to ST8).

When the LCD with static drive (ST0 to ST8) is switched to non-display after reset, set LDR0:ST8 = "0" and LDR1: ST [7:0] = "00000000", and set the static drive selection port (LCS [3:0]).

The same potential pulses are output from the static drive pins (ST0 to ST8).

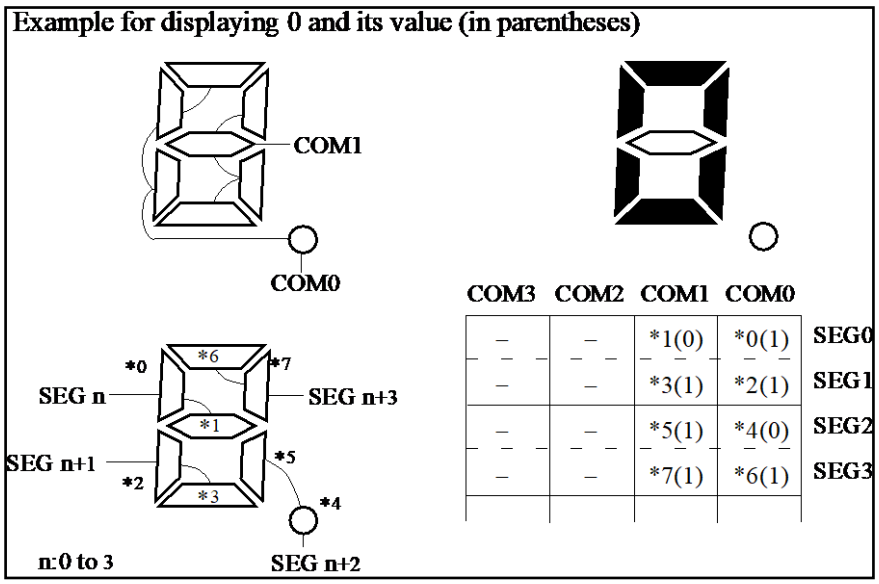
When the LCD with static drive (ST0 to ST8) is switched from non-display to display, set LDR0:ST8 and LDR1: ST [7:0] without change of (LCS [3:0]).

When the LCD with static drive (ST0 to ST8) is switched from display to non-display, set LDR0:ST8 = "0" and LDR1: ST [7:0] = "00000000" without change of the static drive selection port (LCS [3:0]).

The same potential pulses are output from the static drive pins (ST0 to ST8).

8. Sample Program

This section shows sample program.



Setting procedure 2

With the 1/3 duty drive method, make the LCD display a four-digit number , "0 1 2 3 ".

Initial setting (LCDC)

<Initial setting>

1. - Port

COM, SEG output setting for the port	Register name See 6.8. and 6.9.

2. - Setting of VRAM

Setting of VRAM	Register name VRAM00-VRAM05

3. - Setting of control register

Fixed value	Register name LCR1
Bias setting	LCDCMR
Setting of control register	LCR0
	. LCEN
	. VSEL
	. BK
	. MS[1:0]
	. FP[1:0]

<Others>

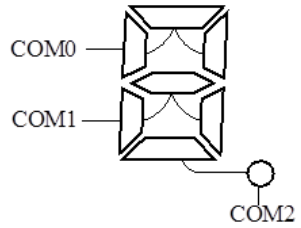
(Note)

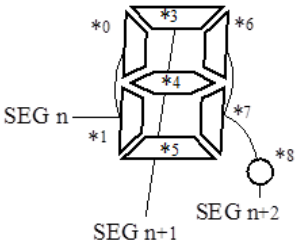
Clock-related setting and setting of _set_il (numerical value) in advance are required. See chapter of "CLOCK SYSTEM" and chapter of "INTERRUPT CONTROLLER" for details.

Program 2

```
void LCD_sample_2(void)
{
        LCDC_initial();
}
void LCDC_initial(void)
{
        PORT_SETTING_LCDC_OUT();        /* Set the LCD controller pin to */
                                        /* peripheral output.   */

        IO_VRAM00 = 0x53;
        IO_VRAM01 = 0x03;
        IO_VRAM02 = 0x30;
        IO_VRAM03 = 0x72;
        IO_VRAM04 = 0x01;
        IO_VRAM05 = 0x37;

        IO_LCR1.byte = 0xFF;        /* Set to FF. */
        IO_LCDCMR. byte = 0x00;     /* 1/3 bias */
        IO_LCR0.byte = 0x08;        /* Setting value =0000_1000 */
                                    /* bit7 = 0 Clock selection Main clock */
                                    /* bit6 = 0 LCEN Display stopped with PSS mode */
                                    /* bit5 = 0 VSEL Internal division resistor disconnected */
                                    /* bit4 = 0 BK Blanking selection bit */
                                    /* bit3-2 = 10 MS[1:0]  1/3 duty   mode */
                                    /* bit1-0 = 00 FP[1:0] */
}
```

Example for displaying 2 and its value (in parentheses)







COM3	COM2	COM1	COM0		
–	(-)	*1(1)	*0(0)	SEG0	First digit
–	*5(1)	*4(1)	*3(1)	SEG1	
–	*8(0)	*7(0)	*6(1)	SEG2	
–	–	1	0	SEG3	Second digit
–	1	1	1	SEG4	
–	0	0	1	SEG5	

Setting procedure 3

With the 1/4 duty drive method, make the LCD display a four-digit number, "0 1 2 3 ".

Initial setting (LCD)

<Initial setting>

1. - PORT — Register name

COM, SEG output setting for the port	See 6.8. and 6.9.

2. - Setting of VRAM — Register name

Setting of VRAM	VRAM00-VRAM03

3. - Setting of control register — registername . bit name

Fixed value	LCR1
Bias setting	LCDCMR
Setting of control register	LCR0
	. LCEN
	. VSEL
	. BK
	. MS[1:0]
	. FP[1:0]

<Others>

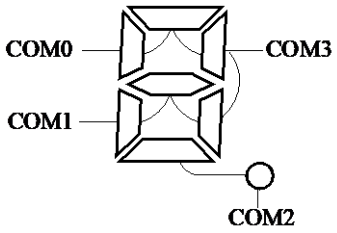
(Note)

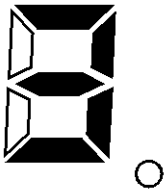
Clock-related setting and setting of _set_il (numerical value) in advance are required. See chapter of "CLOCK SYSTEM" and chapter of "INTERRUPT CONTROLLER" for details.

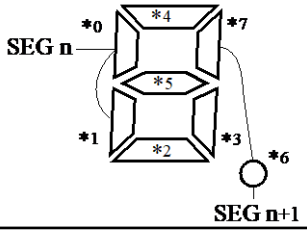
Program 3

```
void LCD_sample_3(void)
{
        LCD_initial();
}

void lcdc_initial(void)
{
        PORT_SETTING_LCDC_OUT();        /* Set the LCD controller pin to */
                                        /* peripheral output. */

        IO_VRAM00 = 0x9F;
        IO_VRAM01 = 0x88;
        IO_VRAM02 = 0xB6;
        IO_VRAM03 = 0xBC;

        IO_LCR1.byte = 0xFF;            /* Set to FF. */
        IO_LCDCMR. byte = 0x00;         /* 1/3 bias */
        IO_LCR0.byte = 0x0C;            /* Setting value =0000_1100 */
                                        /* bit7 = 0 Clock selection Main clock */
                                        /* bit6 = 0 LCEN   Display stopped with PSS mode */
                                        /* bit5 = 0 VSEL Internal division resistor disconnected */
                                        /* bit4 = 0 BK Blanking selection bit */
                                        /* bit3-2 = 11 MS[1:0] 1/4 duty mode */
                                        /* bit1-0 = 00 FP[1:0] */
}
```

Example for displaying 3 and its value (in parentheses)







COM3	COM2	COM1	COM0	
*3(1)	*2(1)	*1(0)	*0(1)	SEG0
*7(0)	*6(0)	*5(1)	*4(1)	SEG1

Setting procedure 4

With the static drive method, make the LCD display an one-digit number.

Initial setting (LCD)

<Initial setting>

1.	- PORT	Register name
	COM, SEG output setting for the port	See 6.8. and 6.9.

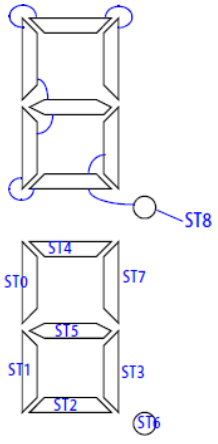
2.	- Data setting	Register name
	Data setting	LDR0 LDR1

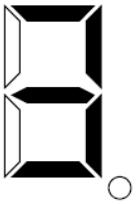
3.	- Setting of control register	Register name . bit name
	Fixed value	LCR1
	Bias setting	LCDCMR
	Setting of control register	LCRS
		. LCSEN
		. LCS[3:0]
		. FPS[1:0]

Program 4

```
void LCD_sample_4(void)
{
        LCD_initial();
}

void lcdc_initial(void)
{
        PORT_SETTING_LCDC_OUT();        /* Set the LCD controller pin to */
                                        /* peripheral output.   */

        IO_LDR0.byte = 0x00;
        IO_LDR1.byte = 0xBC;

        IO_LCR1.byte = 0xFF;            /* Set to FF. */
        IO_LCDCMR. byte = 0x00;         /* 1/3 bias */
        IO_LCRS.byte = 0x20;            /* Setting value =0001_0000 */
                                        /* bit7 = 0 Clock selection Main clock */
                                        /* bit6 = 0 LCSEN   Non-display with PSS mode   */
                                        /* bit5-2 = 1000   LCS[3:0]   ST0 to ST8 */
                                        /* bit1-0 = 00 FPS[1:0]   Frame cycle setting */
}
```

Example) Values to display 3





LDR0/ LDR1 register

ST 8	ST 7	ST 6	ST 5	
0	1	0	1	
ST 4	**ST 3**	**ST 2**	**ST 1**	**ST 0**
1	1	1	0	0

9. Notes

This section shows notes.

- Since resistance value of the external division resisters depends on the LCD to be used, connect the resistors with a suitable value.
- Non-selection level waveforms are output from the COM2 and COM3 pins in 1/2 duty display mode and from the COM3 pin in 1/3 duty display mode.
- If the settings of the LCD drive power control (VSEL), duty selection (MS [1:0]), frame cycle selection (FP [1:0]), etc. are not proper, the LCD does not display correctly.
- When neither the LCD nor general-purpose ports are used, connect a pull-up or pull-down resistor to V3 to V0 pins.
- The static drive is enabled when any setting other than LCS [3:0] = "0000" is set. The duty drive is enabled when LCS [3:0] = "0000" is set.
- Setting of display/non-display of the LCD with static drive (ST0 to ST8)

  When the LCD with static drive (ST0 to ST8) is switched to non-display after reset, set LDR0:ST8 = "0" and LDR1:ST [7:0] = "00000000", and set the static drive selection port (LCS [3:0]). The same potential pulses are output from the static drive pins (ST0 to ST8).

When the LCD with static drive (ST0 to ST8) is switched from non-display to display, set LDR0:ST8 and LDR1:ST [7:0] without change of the static drive selection port (LCS [3:0]).
When the LCD with static drive (ST0 to ST8) is switched from display to non-display, set LDR0:ST8 = "0" and LDR1:ST [7:0] = "00000000" without change of the static drive selection port (LCS [3:0]).

- The same potential pulses are output from the static drive pins (ST0 to ST8).
- When the LCD is used (static drive, duty drive), set the following.
- LCR1:ST [7:0] = "11111111"
  LCDCMR [3:0] = "1111"

- In case of power shutdown, it works as following.
- When port setting is LCD output, output the "Low".
- LCD control is power shutdown, registers (LCR0, VRAMn (n=0 to 15), LCR1, LCDCMR, LCRS, LDR) of LCD control is initialized when returning from power shutdown.
- Registers (LCD_KEYCDR, SEGER, COMVER) of LCD port-related is not initialized.

CHAPTER 29: Indicator PWM

This chapter explains the indicator PWM.

1. Overview
2. Configuration and Block Diagram
3. Operation of Indicator PWM
4. Registers
5. Precautions for Using This Device

CODE: INDICATOR_PWM-S6J3300-E1

1. Overview

The indicator PWM (Pulse Width Modulation) is an output timer that consists of a 16-bit down counter, a 16-bit data register for cycle setting, a 16-bit compare register for duty setting, and a pin controller.

Compared to other output timers, it is capable of output of waveforms with relatively long cycles. Output operation is also supported while in standby mode.

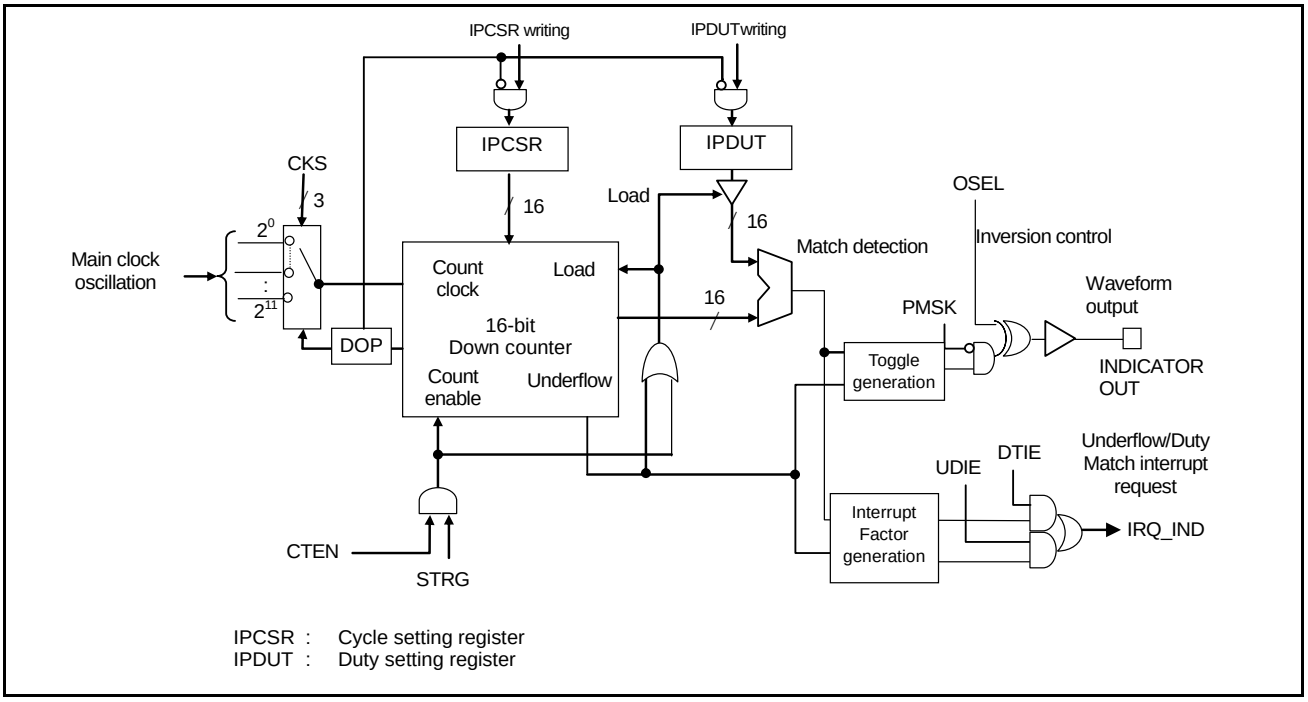
The 16-bit down counter clock can be selected from among six 6 types of clocks (divided by 1, 16, 64, 256, 1024, 2048).

For the count operation, an underload reloads from the PWM cycle register and the count is repeated.

Indicator PWM timer activation is by software trigger only.

2. Configuration and Block Diagram

Figure 29-1 Block Diagram of the Indicator PWM Timer



3. Operation of Indicator PWM

This section explains the operations of the indicator PWM timer.

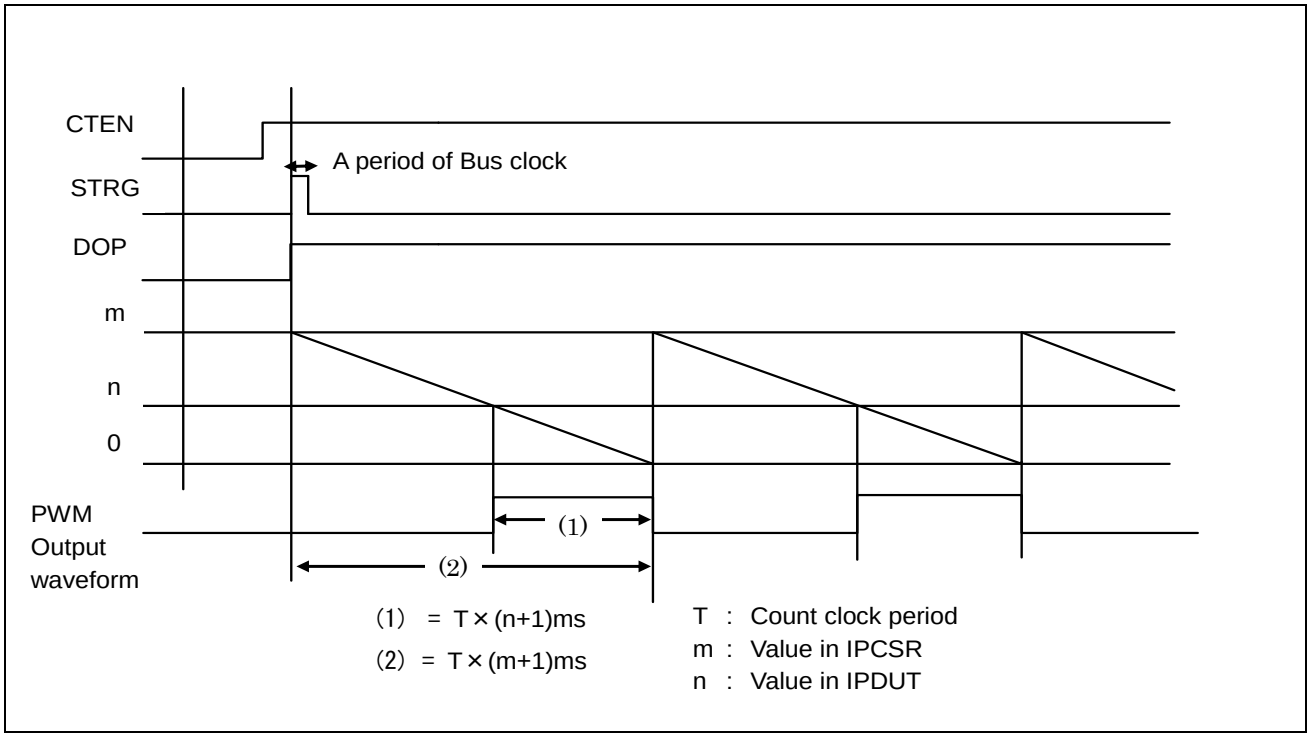
Note:

- *The explanations in this section assume that PWM output is normal polarity. When inverse polarity is set, invert the PWM output polarity shown in this explanation.*
- *For details on how to set inverse polarity, see the OSEL bit in 4.1 Timer Control Register (ITMCR).*

3.1. Indicator PWM Timer

The indicator PWM timer starts counting down from the set cycle value upon software 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. The cycle of the output pulse can be controlled by configuring IPCSR register settings. The duty ratio can be controlled by configuring IPDUT register settings.

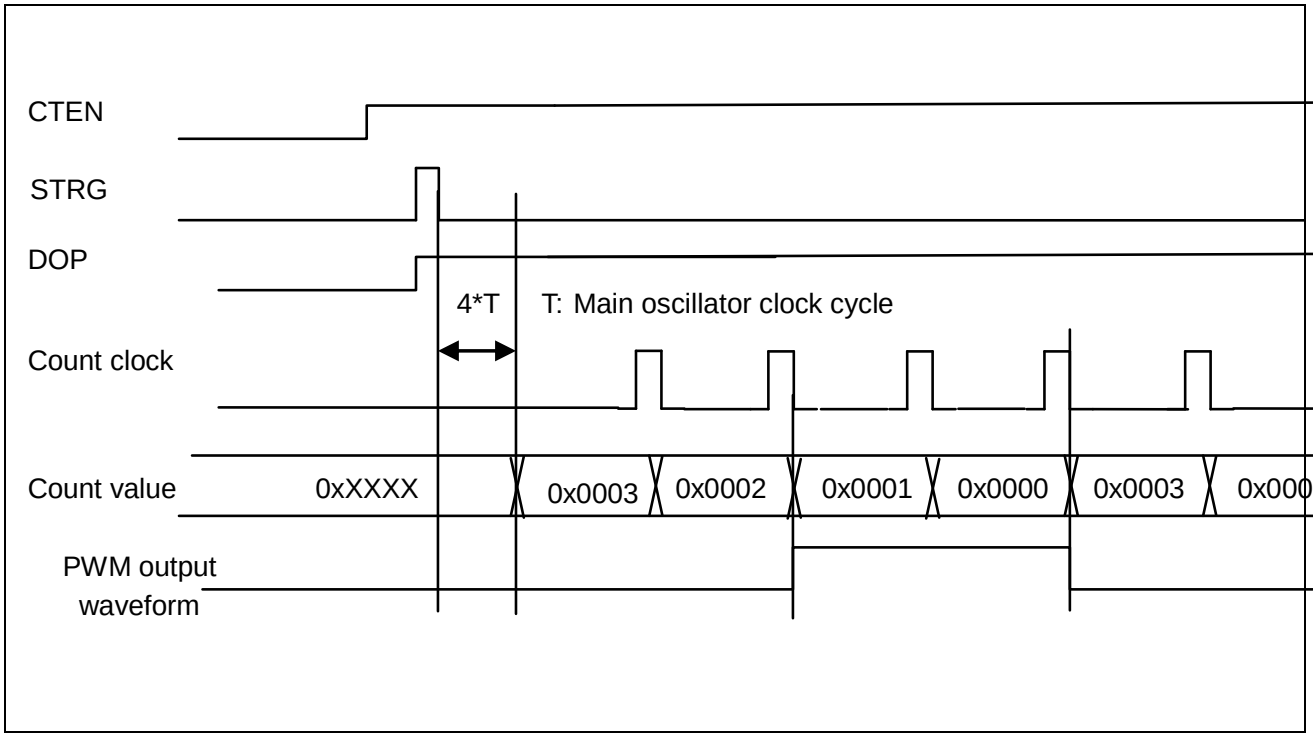
Figure 29-2: Indicator PWM Operation Timing Chart



3.2. PWM Timer Count Operation Start

Configure the PWM timer count lock select setting (ITMCR:CKS2-0), PWM cycle setting (IPCSR), and PWM duty setting (IPDUT) while the timer countdown is stopped (ICNTCR:DOP=0). After configuring settings, enable the count (ICNTCR:CTEN=1) and restart the counter operation by setting the software trigger (ICNTCR:STRG=1). As soon as the software setting is configured, the timer status becomes timer count operation in progress (ICNTCR:DOP=1).

Figure 29-3: Timing Chart when Indicator PWM Timer Count Operation Starts

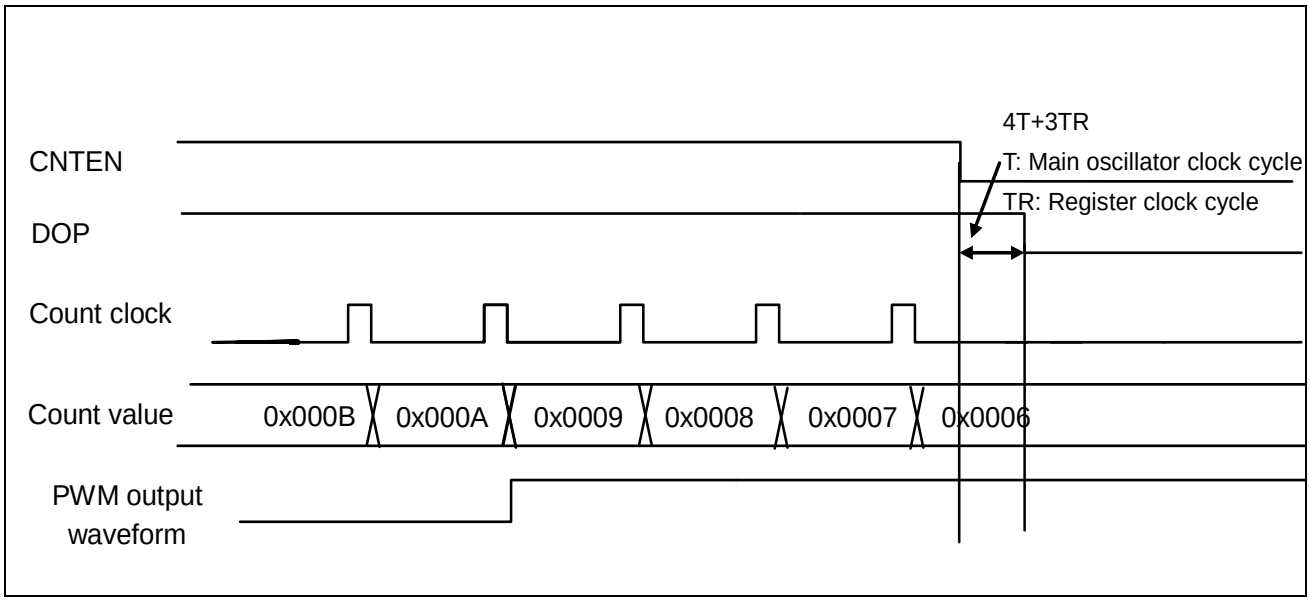


A time of 4T (T: main oscillator clock cycle) is required until the counter value is loaded after the STRG bit becomes 1. Figure 29-3 shows the timing chart for when the cycle setting value is 0x3 and the duty value is 0x1.

3.3. Timer Count Operation Stop

A timer count stop is requested by setting the timer count enable bit (ICNTCR:CTEN) to 0 while a timer count operation is in progress (ICNTCR:DOP=1). When the PWM timer stops the count, timer count operation stop (ICNTCR:DOP=0) is indicated as the timer status. PWM output maintains the output status as during timer count stop.

Figure 29-4: Timing Chart while Timer Count Operation is Stopped



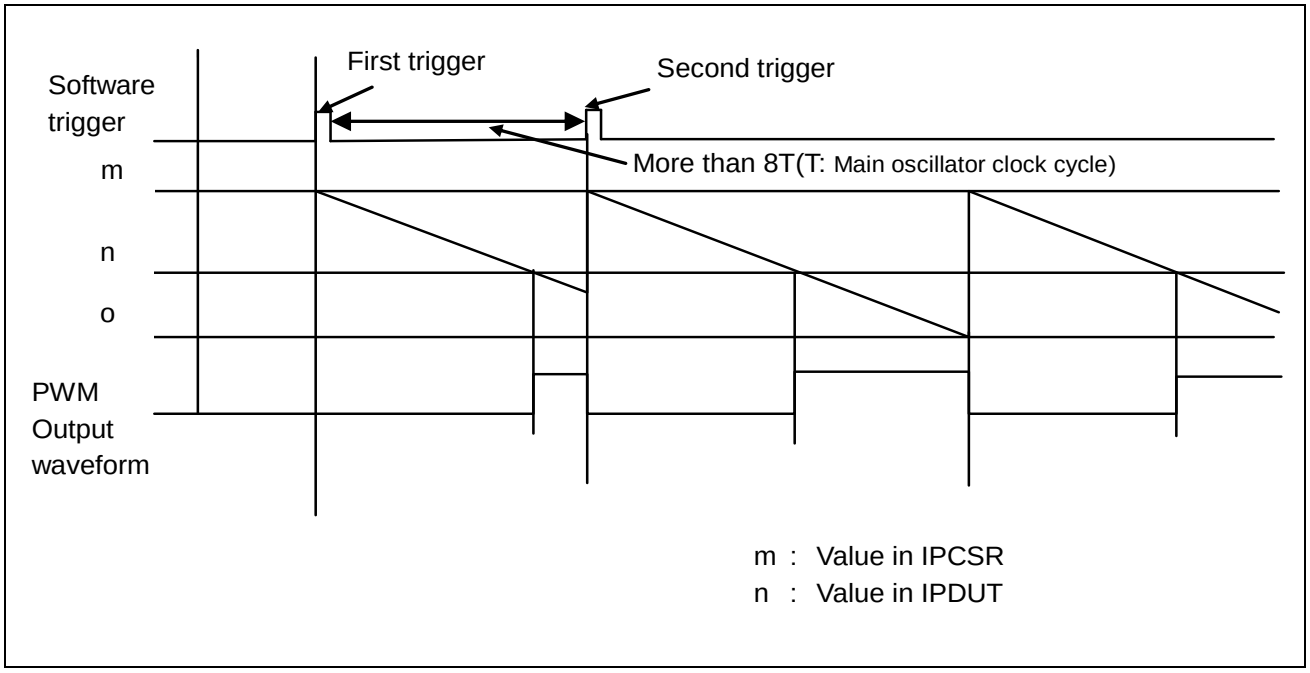
A time of 4T+3TR (T: main oscillator clock cycle, TR: register clock) is required until the counter is stopped (ICNTCR:DOP=0) after the CTEN bit becomes 0.

Figure 29-4 shows the interrupt factors and a timing chart for when the cycle setting value is 0xB and the duty value is 0x9.

3.4. Counter Value Updating while PWM Counter Operation is in Progress

Setting a software trigger (ICNTCR:STRG=1) while a PWM counter operation is in progress loads the values of the PWM cycle setting register (IPCSR) and PWM duty setting register (IPDUT) and updates the count value and duty setting. However, allow a time of at least 8T (T: main oscillator clock cycle) before setting a second software trigger. Failure to do so may result in the software trigger not being reflected.

Figure 29-5: Timing Chart for Counter Update while a Timer Count Operation is in Progress



3.5. Indicator PWM Interrupts

This section provides information about indicator PWM interrupt request flags, interrupt enable bits, and interrupt factors.

Note:

- *The explanations in this section assume that PWM output is normal polarity. When inverse polarity is set, invert the PWM output polarity shown in this explanation.*
- *For details on how to set inverse polarity, see the OSEL bit in 4.1 Timer Control Register (ITMCR).*

3.5.1. Interrupt Control Bits and Interrupt Factors

Table 29-1 lists interrupt control bits and interrupt factors. Underflow detection and detection of duty match share the same interrupt factory output signals (IRQ_IND).

Table 29-1: Interrupt Control Bits and Interrupt Factors

	Status Control Register (ISTC)			
	Interrupt Request Flag Bit	**Interrupt Request Enable Bit**	**Interrupt Factor**	**Interrupt Factor Output Signal**
Indicator PWM	UDIR : bit0	UDIE : bit4	Detection of underflow	IRQ_IND
	DTIR : bit1	DTIE : bit5	Detection of duty match	

An underflow in the PWM counter value causes the underflow interrupt request bit (ISTC:UDIR) of the status control register (ISTC) to be set to "1". Enabling underflow interrupt request enable (ISTC:UDIE=1) while this status exists will cause interrupt request output signal IRQ_IND to become "H". In the same way, "1" is set to the duty match interrupt request bit (ISTC:DTIR) of the status control register (ISTC) when the PWM counter count value matches the duty setting register (IPDUT) value. Enabling duty interrupt request enable (ISTC:DTIE=1) while this status exists will cause interrupt request output signal IRQ_IND to become "H".

3.5.2. Interrupt Factors and Timing Chart (PWM Output: Normal Polarity)

Figure 29-6: PWM Timer Interrupt Factors and Timing Chart

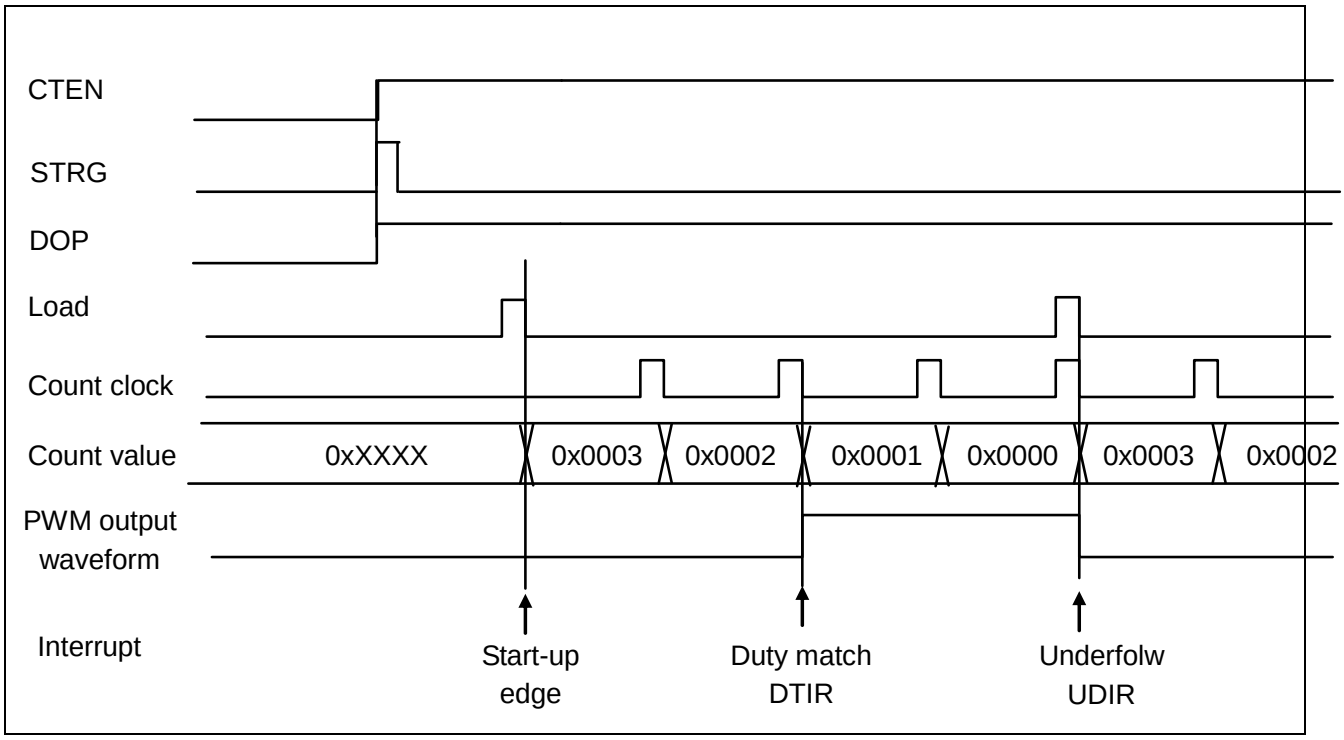


Figure 29-6 shows the interrupt factors and a timing chart for when the setting value is 0x3 and the duty value is 0x1.

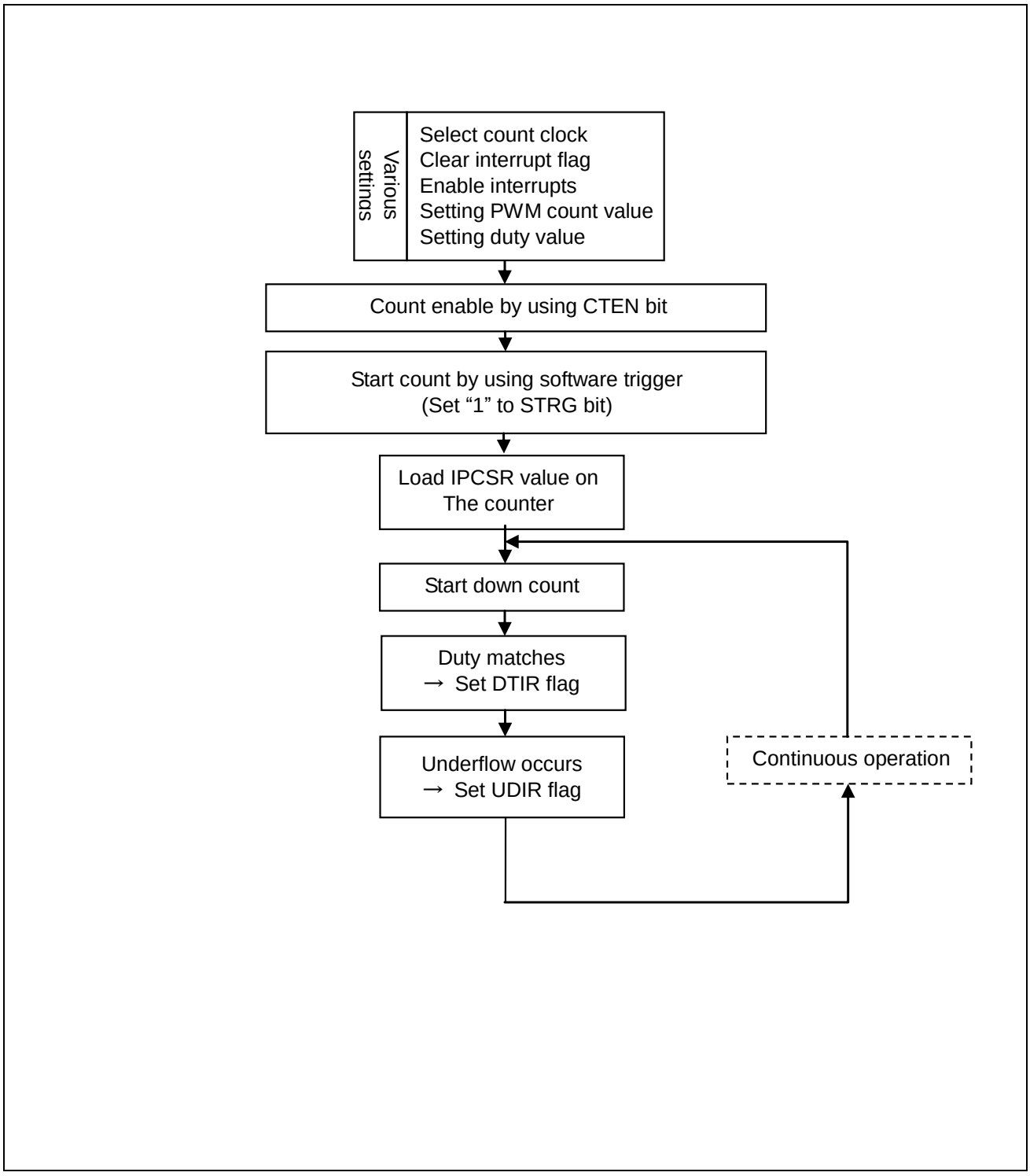
3.5.3. Return from PSS Mode

An interrupt of indicator PWM can be applied to a trigger of returning from PSS mode. Then the bus clock CLK_SYSC0P should be supplied during the PSS mode.

3.6. Operation Flowchart

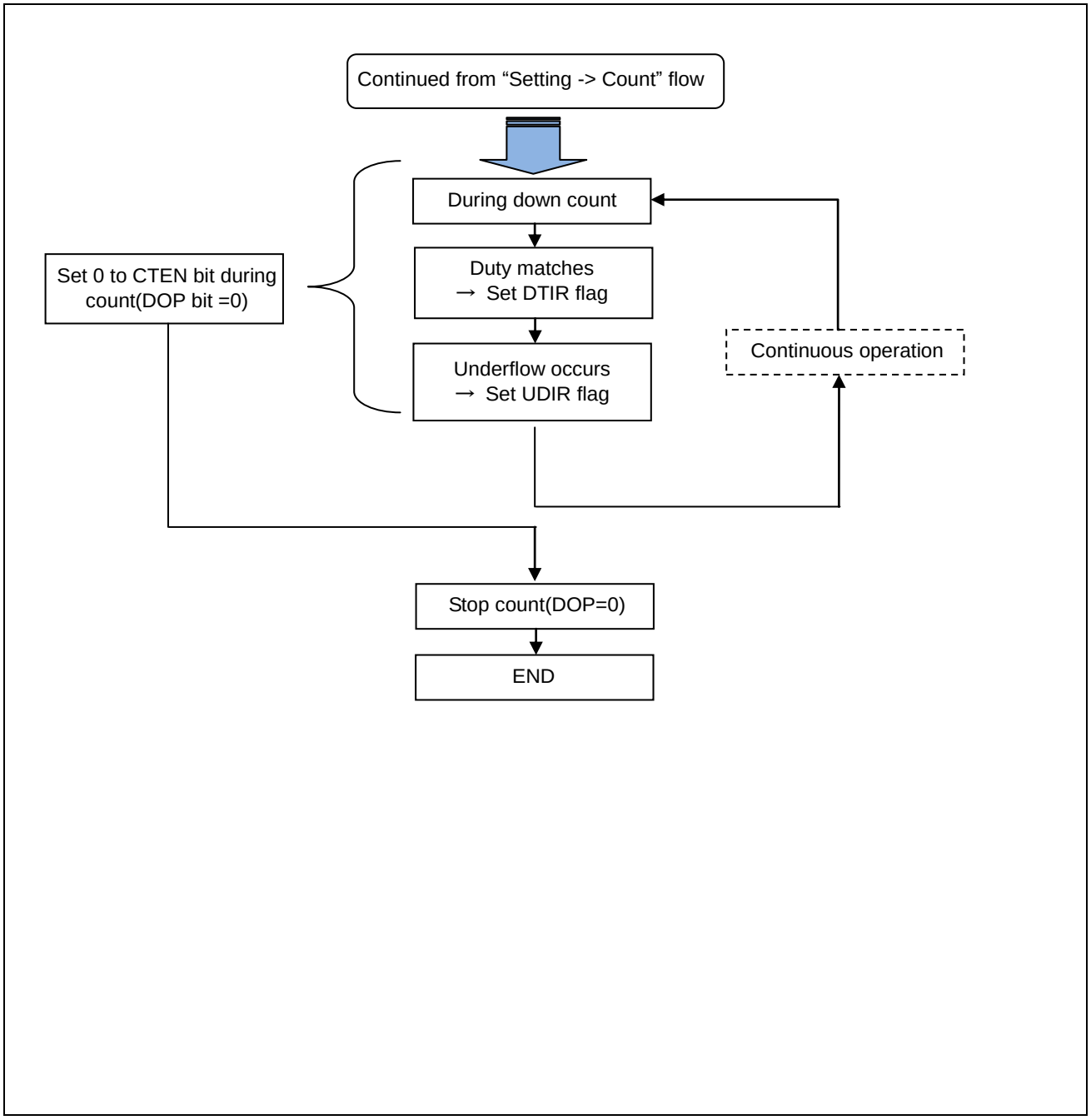
3.6.1. Operation Flowchart (Setting -> Count)

Figure 29-7



3.6.2. Operation Flowchart (Count -> Stop)

Figure 29-8



4. Registers

Table 29-2: Register List

Abbreviated Register Name	Register Name	Reference
ITMCR	Timer control register	4.1
ICNTCR	Count control register	4.2
ISTC	Status control register	4.3
ISTCC	Status control clear register	4.4
ISTCS	Status control set register	4.5
IPCSR	PWM cycle setting register	4.6
IPDUT	PWM duty setting register	4.7

4.1. Timer Control Register (ITMCR)

The timer control register (ITMCR) controls the indicator PWM timer. Note that there are bits that cannot be rewritten during timer operation (ICNTCR:DOP[bit]=1).

REGISTER_NAME	ITMCR
OFFSET	0x0C
ACCESS_SIZE	B H W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

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

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

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved	CKS2	CKS1	CKS0	Reserved	PMSK	Reserved	Reserved
ACCESS_TYPE	R0,WX	R,W	R,W	R,W	R0,WX	R/W	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	Reserved	Reserved	Reserved	Reserved	OSEL	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,WX	R0,WX	R0,WX	R0,WX	R/W	R0,WX	R0,WX	R0,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:15] Reserved

This is a reserved bit.

Writing data to these bits has no effect on operation.

[bit14:12] CKS2 to CKS0: Count clock selection bits

These bits select a count clock for the 16-bit down counter.

Stop the timer (ICNTCR:DOP[bit]=0) before changing the count clock.

CKS2	CKS1	CKS0	Description
0	0	0	Main oscillator clock
0	0	1	Main oscillator clock divided by 16
0	1	0	Main oscillator clock divided by 64
0	1	1	Main oscillator clock divided by 256
1	0	0	Main oscillator clock divided by 1024
1	0	1	Main oscillator clock divided by 2048
1	1	0	Main oscillator clock
1	1	1	Main oscillator clock

[bit11] Reserved

This is a reserved bit.

Writing data to these bits has no effect on operation.

[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.

Note:

- *If the output polarity specification bit (OSEL) in the timer control register (lower byte of ITMCR) is set for inverted output, setting the PMSK bit to "1" results in masking to H output.*

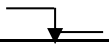
bit	Description
0	Normal output
1	Fixed at L output

[bit9:4] Reserved

These are reserved bits. Writing data to these bits has no effect on operation.

[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		

bit	Description
0	Normal polarity
1	Inverse polarity

[bit2:0] Reserved

These are reserved bits. Writing data to these bits has no effect on operation.

4.2. Count Control Register (ICNTCR)

The count control register(ICNTCR) is PWM timer count control register.

REGISTER_NAME	ICNTCR
OFFSET	0x10
ACCESS_SIZE	B H W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

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

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

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

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	DOP	Reserved	Reserved	Reserved	Reserved	Reserved	CTEN	STRG
ACCESS_TYPE	R,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R/W	R0,W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:8] Reserved

These are reserved bits. Writing data to these bits has no effect on operation.

[bit7] DOP: Count status bit

Setting the STRG bit to "1" while the CTEN bit is "1" will immediately make this bit "1".

The read value for this bit is "1" while a PWM timer count is in progress.

The read value for this bit is "0" while the PWM timer is stopped.

The read value for this bit is "0" while the PWM timer count is stopped after the CTEN bit is set to "0".

Set this bit to "0" before selecting a timer clock (ITMCR:CKS2-0), or configuring PWM cycle (IPCSR) or PWM duty (IPDUT) settings. These settings are ignored if they are configured while this bit is "1".

bit	Description
0	Timer count stopped
1	Timer count operation in progress

[bit6:2] Reserved

These are reserved bits. Writing data to these bits has no effect on operation.

[bit1] CTEN: Count operation enable bit

This bit enables operation of the down counter.

If "0" is written to this bit while a counter operation is in progress (DOP bit is "1"), a stop request will be output for the counter.

Counter stop can be checked by confirming that the counter status bit (DOP bit) is "0".

After CTEN bit is set to 0, if you re-set CTEN bit to "1", you must check DOP bit is "0".

bit	Description
0	Request stop
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".

Note:

- *Even if "1" is written to the CTEN and STRG bits at the same time, a software trigger is applied.*
- *When consecutively setting STRG bits, allow a time of at least 8T (T: main oscillator clock cycle) between settings. Failure to do so may result in a setting not being reflected.*

bit	Description
0	Disabled
1	Startup by software

4.3. Status Control Register (ISTC)

For details on writing to the status control register (ISTC), see "5.1 Notes to observe when accessing a register".

REGISTER_NAME	ISTC
OFFSET	0x14
ACCESS_SIZE	B H W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

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

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

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

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved	Reserved	DTIE	UDIE	Reserved	Reserved	DTIR	UDIR
ACCESS_TYPE	R0,WX	R0,WX	R,W	R,W	R0,WX	R0,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:6] Reserved

These are reserved bits. Writing data to these bits has no effect on operation.

[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 ISTCC:DTIEC bit clears this bit.

Writing "1" to the ISTCS:DTIES bit sets this bit.

bit	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 ISTCC:UDIEC bit clears this bit.

Writing "1" to the ISTCS:UDIES bit sets this bit.

bit	Description
0	Disable interrupt requests.
1	Enable interrupt requests.

[bit3:2] Reserved

These are reserved bits. Writing data to these bits has no effect on operation.

[bit1] DTIR: Duty match interrupt request bit

The DTIR bit is set to "1" when the count value matches the duty setting value.

Writing "1" to the ISTCC:DTIRC bit clears this bit.

This bit is read-only. Writing data to this bit has no effect on operation.

bit	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.

Writing "1" to the ISTCC:UDIRC bit clears this bit.

This bit is read-only. Writing data to this bit has no effect on operation.

bit	Description
0	The interrupt factor is cleared.
1	Detect the interrupt factor.

4.4. Status Control Clear Register (ISTCC)

The status control clear register (ISTCC) is a register that is used to clear bits of the status control register (ISTCC).

REGISTER_NAME	ISTCC
OFFSET	0x18
ACCESS_SIZE	B H W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

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

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

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

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved	Reserved	DTIEC	UDIEC	Reserved	Reserved	DTIRC	UDIRC
ACCESS_TYPE	R0,WX	R0,WX	R0,W	R0,W	R0,WX	R0,WX	R0,W	R0,W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:6] Reserved

These are reserved bits. Writing data to these bits has no effect on operation.

[bit5] DTIEC: Duty match interrupt request enable clear bit

If "1" is written to this bit, the ISTC:DTIE bit is cleared to "0".

"0" is always read from this bit.

bit	Description
0	Disabled
1	Clear the DTIE bit.

[bit4] UDIEC: Underflow interrupt request enable clear bit

If "1" is written to this bit, the ISTC:UDIE bit is cleared to "0".

"0" is always read from this bit.

bit	Description
0	Disabled
1	Clear the UDIE bit.

[bit3:2] Reserved

These are reserved bits. Writing data to these bits has no effect on operation.

[bit1] DTIRC: Duty match interrupt request clear bit

If "1" is written to this bit, the ISTC:DTIR bit is cleared to "0".

"0" is always read from this bit.

bit	Description
0	Disabled
1	Clear the DTIR bit.

[bit0] UDIRC: Underflow interrupt request clear bit

If "1" is written to this bit, the ISTC:UDIR bit is cleared to "0".

"0" is always read from this bit.

bit	Description
0	Disabled
1	Clear the UDIR bit.

4.5. Status Control Set Register (ISTCS)

The status control set register (ISTCS) is a register that is used to set bits of the status control register (ISTC).

REGISTER_NAME	ISTCS
OFFSET	0x1C
ACCESS_SIZE	B H W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

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

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

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

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved	Reserved	DTIES	UDIES	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,WX	R0,WX	R0,W	R0,W	R0,WX	R0,WX	R0,WX	R0,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:6] Reserved

This is a reserved bit.

Writing data to these bits has no effect on operation.

[bit5] DTIES: Duty match interrupt request enable set bit

If "1" is written to this bit, the ISTC:DTIE bit is set to "1".

"0" is always read from this bit.

bit	Description
0	Disabled
1	Set the DTIE bit.

[bit4] UDIES: Underflow interrupt request enable set bit

If "1" is written to this bit, the ISTC:UDIE bit is set to "1".

"0" is always read from this bit.

bit	Description
0	Disabled
1	Set the UDIE bit.

[bit3:0] Reserved

These are reserved bits. Writing data to these bits has no effect on operation.

4.6. PWM Cycle Setting Register (IPCSR)

The PWM cycle setting register (IPCSR) is a register with a buffer for setting a cycle. Transfer to the timer is performed upon software trigger activation. Note that rewriting is not supported while a timer count operation is in progress (ICNTCR:DOP[bit]=1).

REGISTER_NAME	IPCSR
OFFSET	0x00
ACCESS_SIZE	B H W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

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

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

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	IPCSR[15]	IPCSR[14]	IPCSR[13]	IPCSR[12]	IPCSR[11]	IPCSR[10]	IPCSR[9]	IPCSR[8]
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	IPCSR[7]	IPCSR[6]	IPCSR[5]	IPCSR[4]	IPCSR[3]	IPCSR[2]	IPCSR[1]	IPCSR[0]
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] Reserved

These are reserved bits. Writing data to these bits has no effect on operation.

[bit15:0] IPCSR: PWM cycle setting register

This register is for setting the PWM timer cycle. Transfer to the timer is performed upon software trigger activation.

Stop the timer (ICNTCR:DOP[bit]=0) before configuring the PWM cycle setting register (IPCSR) setting. Settings are ignored if they are configured while timer operation is in progress (ICNTCR:DOP[bit]=1).

4.7. PWM Duty Setting Register (IPDUT)

The PWM duty setting register (IPDUT) is a register for setting a duty. Transfer is performed upon software trigger activation. Note that rewriting is not supported while a timer count operation is in progress (ICNTCR:DOP[bit]=1).

REGISTER_NAME	IPDUT
OFFSET	0x04
ACCESS_SIZE	B H W
MULTIPLE	
NUMERIC_TYPE	
OTHER	

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

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

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	IPDUT[15]	IPDUT[14]	IPDUT[13]	IPDUT[12]	IPDUT[11]	IPDUT[10]	IPDUT[9]	IPDUT[8]
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	IPDUT[7]	IPDUT[6]	IPDUT[5]	IPDUT[4]	IPDUT[3]	IPDUT[2]	IPDUT[1]	IPDUT[0]
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] Reserved

This is a reserved bit.

Writing data to these bits has no effect on operation.

[bit15:0] IPDUT: PWM Duty Setting Register

This is the register for setting the duty. Transfer is performed upon software trigger activation.

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 IPCSR < IPDUT in the set values, the output for normal polarity is all "L", and the output for inverse polarity is all "H".

Stop the timer (ICNTCR:DOP[bit]=0) before configuring PWM duty setting register (IPDUT) settings. Settings are ignored if they are configured while timer operation is in progress (ICNTCR:DOP[bit]=1).

5. Precautions for Using This Device

This section explains precautions during use.

5.1. Notes to Observe when Accessing a Register

■ Status control register (ISTC) access

To clear a specific bit in this register, write "1" to the corresponding bit in the status control clear register (ISTCC).

To set a specific bit in this register, write "1" to the corresponding bit in the status control set register (ISTCS).

Data can be written directly to this register only when writing to all bits.

5.2. Indicator PWM Operation Precautions

■ Precautions when configuring settings using a program

Timer control register (ITMCR) CKS2-0 bits, PWM cycle setting register (IPCSR), and PWM duty setting register (IPDUT) settings are not reflected if they are configured while a count operation is in progress (ICNTCR:DOP=1). Be sure to stop the count (ICNTCR:DOP=0) before configuring these settings.

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.

When consecutively setting STRG bits, allow a time of at least 8T (T: main clock cycle) between settings. Failure to do so may result in a setting not being reflected.

■ Note that re-setting STRG bit during timer operation.

During PWM timer operation, the re-setting of the software trigger (ICNTCR:STRG="1") allows to update the counter value. However, as a result of different between register clock speed and counter clock(Main clock) speed, the count value of the counter at last time would reach the register value of IPDUT or IPCSR before the re-setting of the software trigger is reflected in the counter. In this case, Note that there is a possibility that the Duty match interrupt and Underflow interrupt is generated.

CHAPTER 30: Memory Protection Unit for AXI

This chapter explains the functions and operations of the Memory Protection Unit for AMBA AXI protocol bus (MPU AXI).

1. Overview
2. Configuration and Block Diagram
3. Operation of the MPU AXI
4. Registers
5. Notes on Using MPU AXI

CODE: MPUAXI-S6J3300-E1

1. Overview

This section describes the features of the MPU AXI.

■ Features of the MPU AXI

The MPU AXI module monitors the accesses from AXI masters and checks each access against an authorized set of access permissions. Access permissions ('permission attributes' here onwards) are defined by 'access permissions' bits. These bits are explained in Section 'MPU access permissions'. MPU AXI provides eight regions and one background region. Each region has corresponding access permission bits that defines the permission attributes for that particular region. Any unauthorized access to memory space is flagged using Non-Maskable Interrupt. MPU AXI also collects information about the unauthorized bus access and stores it in its internal registers.

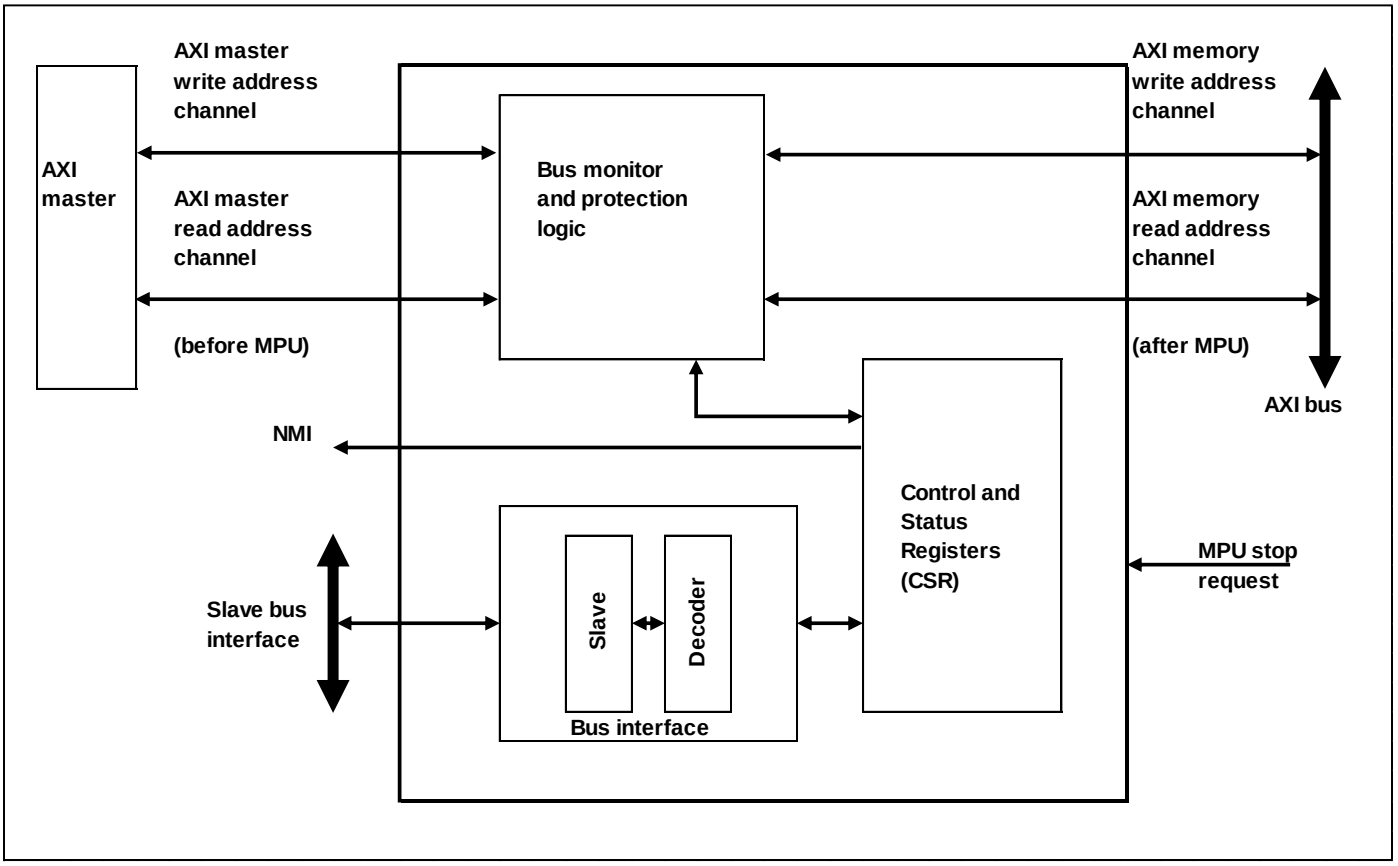
- Each of the eight regions in MPU AXI is specified using corresponding start address and end address
- Background region covers entire 4 GB address space
- On unauthorized access MPU AXI generates an NMI to the CPU
- MPU AXI collects information about the AXI master bus access that caused unauthorized access
- Supports 8-bit, 16-bit, and 32-bit bus accesses for configuration of registers in MPU AXI
- Supports lock, unlock feature for protection of registers from illegal write accesses
- Modification of registers in MPU AXI is only allowed in privileged mode
- Supports MPU stop feature that allows blocking of all the accesses to memory space
- Optionally supports privileged mode overwrite feature that allows overwrite of privilege attribute of memory side AXI interface

2. Configuration and Block Diagram

This section shows a block diagram of MPU AXI

■ Block Diagram of MPU AXI

Figure 30-1 Block Diagram of MPU AXI



■ Bus Interface

The bus masters can access the MPU AXI module through its slave bus interface.

■ Bus Monitor and Protection Logic

This logic monitors the transaction on AXI master interfaces. 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 to CSR logic that in turn generates NMI interrupt. All transactions on AXI master interfaces (including the transfer that caused permission violation) are blocked until NMI is cleared.

■ Control and Status Registers

The operation of MPU AXI can be controlled and monitored through its Control and Status Registers (CSR)

3. Operation of the MPU AXI

This section describes the operation of MPU AXI

MPU AXI provides start address and end address for each of the eight regions. MPU AXI regions are defined with granularity of 128 bytes.

Start address specifies the first address of the region and is specified by registers MPUXn_SADDR1 to MPUXn_SADDR8 for region 1 to region 8 respectively. Since the region granularity is 128 bytes least significant 7 bits of start addresses will read 0.

End addresses are specified by registers MPUXn_EADDR1 to MPUXn_EADDR8 for region 1 to region 8 respectively. Least significant 7 bits of End Address Registers are read-only bits and will always read '1'. This ensures the granularity of 128 bytes.

■ AXI Burst Monitoring

Bus monitor and protection logic monitors AXI master 'write address channel' signals and 'read address channel' signals ('AXI master interfaces' here onwards). The transactions on AXI master interfaces are manipulated (if required) before they are passed on to AXI memory 'write address channel' signals and 'read address channel' signals ('AXI memory interfaces' here onwards).

AXI protocol supports following features

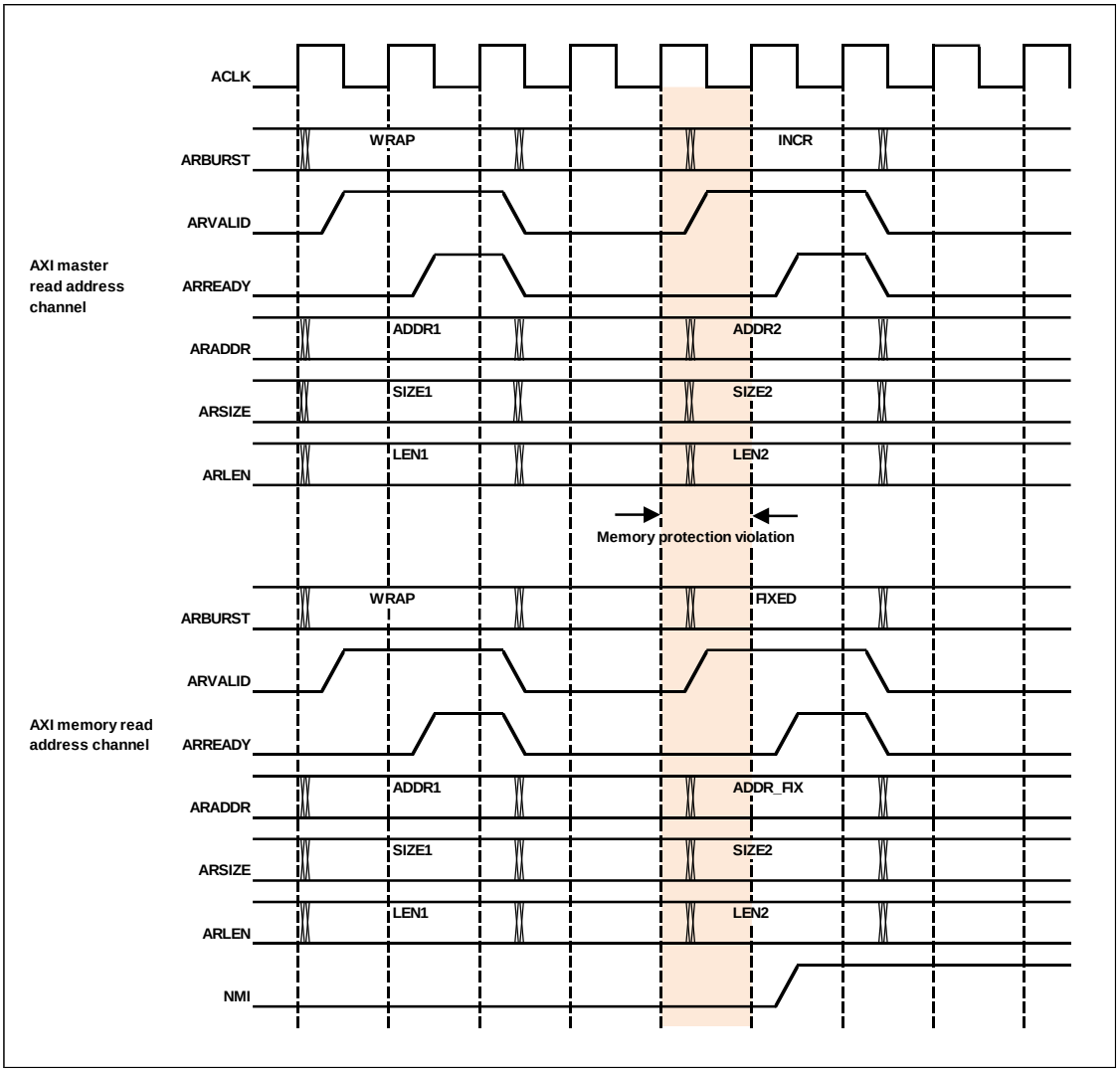
- Separate address or control and data phases
- Burst based transaction where only start address is issued
- Separate write and read address/control channels
- Separate write and read data channels

AXI master begins each burst by driving transaction control information and address of the first byte in the transaction. As the burst transaction progresses, AXI slave calculates the addresses of subsequent transfers in the burst. The AWLEN (write address channel signal), ARLEN (read address channel signal) specifies the number of data transfers for a burst transaction. Each burst can be of 1 to 16 transfers long. The AWSIZE, ARSIZE signals specifies the maximum number of data transfers in terms of bytes in each data transfer within a burst. The AXI protocol defines FIXED, incrementing and wrapping burst types.

As the addresses are defined in separate channels than data channels MPU AXI monitors and controls only address channels of AXI. If memory protection violation is detected entire burst transaction is manipulated to FIXED address burst with address of predefined value.

Figure 30-2 shows timings for read address channel signals. First burst is of WRAP type for which no memory protection violation was detected. Second burst is of INCR type for which memory protection violation was detected and hence MPU AXI manipulates the ARBURST to FIXED type and ARADDR to predefined value (shown as ADDR_FIX).

Figure 30-2 MPU AXI Example Timings



MPU AXI uses burst start address (AWADDR, ARADDR signals), burst type (AWBURST, ARBURST signals), burst length (AWLEN, ARLEN signals), and burst size (AWSIZE, ARSIZE signals) from AXI burst transaction to calculate the lowest address and the highest address of the burst and then both the addresses are used to find the region match.

For FIXED type of burst:

Lowest address = Highest address = Burst start address

For INCR type of burst:

Lowest address = Burst start address

Highest address = Burst start address + (num_bytes x num_transfers) – 1

where, num_bytes is number of bytes in each transfer of burst and

num_transfers is number of transfers in the burst

For WRAP type of burst:

Lowest address = Wrap boundary

Highest address = Wrap boundary + (num_bytes x num_transfers) – 1

MPU Access Permissions

Region control registers in MPU AXI, MPUXn_CTRL1 to MPUXn_CTRL8 are used to control the access permission for region 1 to region 8 respectively. Also MPUXn_CTRL0 is used to control the access permission for background region.

Table 30-1 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 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 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 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 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 memory protection violation.
'110'	read only	read only	Reads are permitted in privileged as well as non-privileged mode. Any write access would generate memory protection violation.
'111'	read, write	read, write	All bus transfers are permitted. No memory protection violation is generated in this mode.

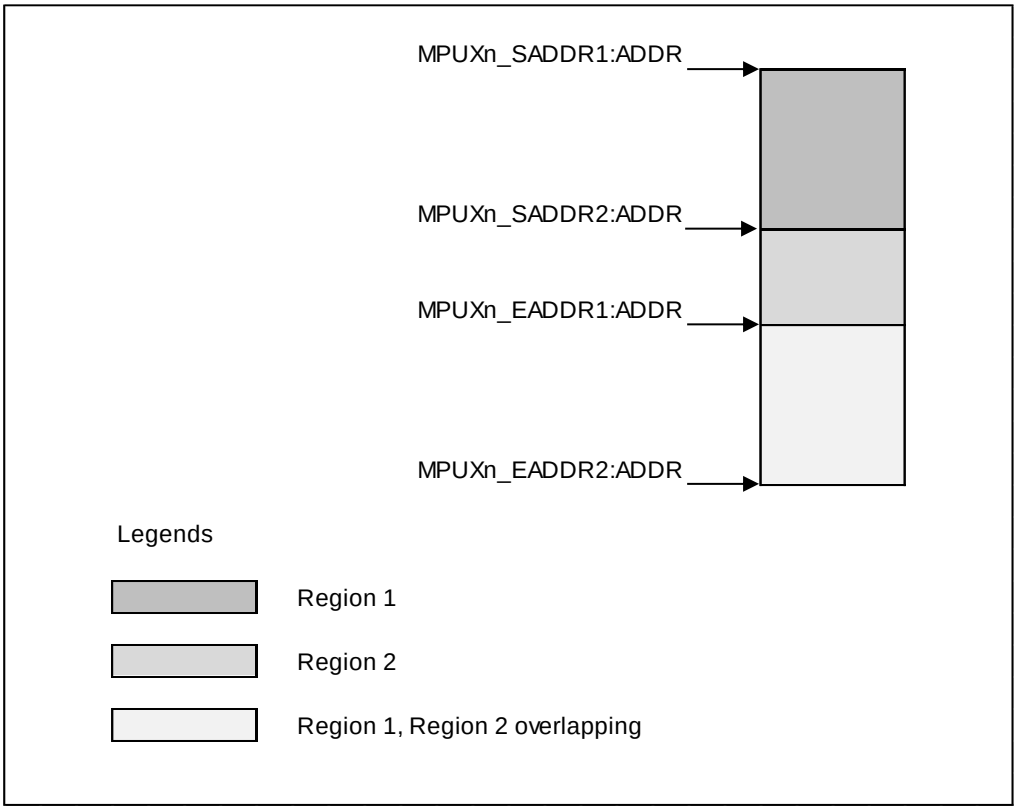
MPU Priority Decision

MPU AXI provides start address and end address for each region. Start address specifies the first address of the region and end address specifies the last address of the region. It is allowed in MPU AXI that the region may overlap each other. Hence it is also possible that an AXI transaction address may match with multiple regions.

For AXI protocol transaction following scenarios are possible.

2. Lowest address and highest address might fall in the same region.
3. Lowest address might match in one region, but highest address might match in some other region.
4. Scenario mentioned in point 1 here, happens for multiple regions due to region overlapping.
5. Scenario mentioned in point 2 here, happens for multiple regions due to region overlapping.

Figure 30-3 Example Region Overlapping



MPU AXI finds region match signals based on lowest address of AXI burst transaction and also separate region match signals based on highest address. Due to overlapping of regions it is possible lowest address and/or highest address may match with multiple regions.

MPU AXI supports upto eight regions. Each region is identified as region 1, region 2, region 3, and so on upto region 8. Region 8 is given highest priority, region 7 has second highest priority, and so on with region 1 as the second lowest priority region. Background region is the lowest priority region.

Whenever the AXI transaction lowest address matches with multiple regions the permissions corresponding to the highest priority (among all matching region) region is applied. Similarly, if AXI transaction highest address matches with multiple regions the permissions corresponding to highest priority (among all matching region) region is applied.

It is possible that lowest address and highest address may match with different highest priority regions. In this case the more restrictive permission is applied. Table 30-2 shows access permissions with corresponding restriction index for privileged mode access. Restriction index 1 has the most restrictive

access and access permissions with restriction index 3 has the least restrictive access. This table is applied when privileged mode attribute of current transaction is of type privileged mode.

Similarly, Table 30-3 shows access permissions with corresponding restriction index for non- privileged mode access. This table is applied when privileged mode attribute of current transaction is of type non-privileged mode.

Table 30-2 Restrictive Access Permission Matrix for Privileged Mode Access

AP Bits	Access in Privileged Mode	Restriction Index
'000', '100'	No access	1
'101', '110'	read only	2
'001', '010', '011', '111'	read, write	3

Table 30-3 Restrictive Access Permission Matrix for Non-Privileged Mode Access

AP Bits	Access in Non-Privileged Mode	Restriction Index
'000', '001', '100', '101'	No access	1
'010', '110'	read only	2
'011', '111'	read, write	3

■ Bus Monitor and Protection Logic

All transactions on the AXI master interfaces are monitored and checked for permitted access.

- Bus monitor and protection logic within MPU AXI compares the lowest address and highest address of current transaction with start addresses and end addresses of each region to find region match where the current transaction matches to among the eight defined regions
- As explained in Section MPU priority decision the AXI transaction address may match with multiple regions. The permission attributes of highest priority region are checked against the attributes of currently applied transaction from AXI master
- If the attributes of currently applied transaction are within permitted attributes, current transaction is passed on to the AXI memory interfaces
- If the attributes are not within permitted attributes current transaction is blocked. The Non Maskable Interrupt (MPUXn_CTRL0:NMI) flag is set. If the memory protection violation is detected on write address channel, the address and control information is stored in MPUXn_WERRA and MPUXn_WERRC registers respectively. If the memory protection violation is detected on read address channel, the address and control information is stored in MPUXn_RERRA and MPUXn_RERRC registers respectively. It is possible that memory protection violation is detected simultaneously on both the channels
- All further transaction are blocked until the MPUXn_CTRL0:NMI flag is cleared by software. Further monitoring of AXI master interfaces is also stalled until the MPUXn_CTRL0:NMI flag is cleared

When a transfer is blocked, MPU AXI does the following actions:

- Current transaction on AXI master is manipulated to FIXED address burst
- The transaction address is changed to predefined address. Check device specific datasheet for details of predefined address

MPU Stop Feature

Optionally MPU AXI supports MPU stop feature.

When this mode is enabled all accesses to memory space are blocked and MPU AXI does the following actions:

- Burst type signal is driven to FIXED type burst
- Burst address is driven to predefined FIXED address

Privileged Mode Overwrite Feature

Optionally MPU AXI supports privileged mode overwrite feature.

When this mode is enabled privileged mode attribute on the AXI memory interfaces are set by register bit setting MPUXn_CTRL0:PROT.

Note:

- *Bus monitor and protection logic for detecting memory protection violation uses privilege mode attribute on AXI master interfaces and not MPUXn_CTRL0:PROT bit setting even when privileged mode overwrite feature is enabled.*

4. Registers

This section describes the registers of MPU AXI

The MPU AXI module contains various registers to configure its operation, to monitor its status and to read the information it has collected from the AXI master interfaces at the time of the memory protection violation.

The MPU AXI module is allocated 1 KB of MCU address space for mapping the Configuration and Status Registers (i.e. CSRs). The address area allocated to MPU AXI and the Control and Status Registers in MPU AXI are explained in this section.

The suffix 'n' in the register name indicates that the register is an instance 'n' of the module.

■ Registers of MPU AXI

The following registers are available for MPU AXI:

- MPU AXI Control Register (MPUXn_CTRL0)
- MPU AXI NMI Enable Register (MPUXn_NMIEN)
- MPU AXI Write Error Control Register (MPUXn_WERRC)
- MPU AXI Write Error Address Register (MPUXn_WERRA)
- MPU AXI Read Error Control Register (MPUXn_RERRC)
- MPU AXI Read Error Address Register (MPUXn_RERRA)
- MPU AXI Region Control Registers (MPUXn_CTRL1 to 8)
- MPU AXI Start Address Registers (MPUXn_SADDR1 to 8)
- MPU AXI End Address Registers (MPUXn_EADDR1 to 8)
- MPU AXI Unlock Register (MPUXn_UNLOCK)
- MPU AXI Module ID Register (MPUXn_MID)

■ Memory Layout of MPU AXI Registers

Offset	+3	+2	+1	+0
0x00000000	MPUXn_CTRL0 00000000 00000000 00000001 00000000			
0x00000004	MPUXn_NMIEN 00000000 00000000 00000000 00000001			
0x00000008	MPUXn_WERRC 00000000 00000000 00000XXX XXXXXXX0			
0x0000000C	MPUXn_WERRA XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX			
0x00000010	MPUXn_RERRC 00000000 00000000 00000XXX XXXXXXX0			
0x00000014	MPUXn_RERRA XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX			
0x00000018	MPUXn_CTRL1 00000000 00000000 00000000 00000000			
0x0000001C	MPUXn_SADDR1 00000000 00000000 00000000 00000000			
0x00000020	MPUXn_EADDR1 00000000 00000000 00000000 01111111			

Offset	+3	+2	+1	+0
0x00000024	MPUXn_CTRL2 00000000 00000000 00000000 00000000			
0x00000028	MPUXn_SADDR2 00000000 00000000 00000000 00000000			
0x0000002C	MPUXn_EADDR2 00000000 00000000 00000000 01111111			
0x00000030	MPUXn_CTRL3 00000000 00000000 00000000 00000000			
0x00000034	MPUXn_SADDR3 00000000 00000000 00000000 00000000			
0x00000038	MPUXn_EADDR3 00000000 00000000 00000000 01111111			
0x0000003C	MPUXn_CTRL4 00000000 00000000 00000000 00000000			
0x00000040	MPUXn_SADDR4 00000000 00000000 00000000 00000000			
0x00000044	MPUXn_EADDR4 00000000 00000000 00000000 01111111			
0x00000048	MPUXn_CTRL5 00000000 00000000 00000000 00000000			
0x0000004C	MPUXn_SADDR5 00000000 00000000 00000000 00000000			
0x00000050	MPUXn_EADDR5 00000000 00000000 00000000 01111111			
0x00000054	MPUXn_CTRL6 00000000 00000000 00000000 00000000			
0x00000058	MPUXn_SADDR6 00000000 00000000 00000000 00000000			
0x0000005C	MPUXn_EADDR6 00000000 00000000 00000000 01111111			
0x00000060	MPUXn_CTRL7 00000000 00000000 00000000 00000000			
0x00000064	MPUXn_SADDR7 00000000 00000000 00000000 00000000			
0x00000068	MPUXn_EADDR7 00000000 00000000 00000000 01111111			
0x0000006C	MPUXn_CTRL8 00000000 00000000 00000000 00000000			
0x00000070	MPUXn_SADDR8 00000000 00000000 00000000 00000000			
0x00000074	MPUXn_EADDR8 00000000 00000000 00000000 01111111			

Offset	+3	+2	+1	+0
0x00000078	MPUXn_UNLOCK 00000000 00000000 00000000 00000000			
0x0000007C	MPUXn_MID 00000000 00000010 00001101 00000000			

4.1. MPU AXI Control Register (MPUXn_CTRL0)

MPU AXI Control Register can be used by the software to configure the MPU AXI. This register provides a bit to enable the MPU AXI 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 and MPU stop feature. Lastly it provides MPU AXI lock status and controls the status for Non-Maskable Interrupt.

■ **MPUXn_CTRL0**

MPUXn_CTRL0																															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
read0	read0	read0	read0	read0	AP[2]	AP[1]	AP[0]	read0	read0	read0	read0	read0	read0	MPUENC	MPUEN	read0	read0	read0	PROT	POEN	MPUSTOPEN	MPUSTOP	LST	read0	read0	read0	read0	read0	read0	NMICL	NMI
R0	R0	R0	R0	R0	RWP	RWP	RWP	R0	R0	R0	R0	R0	R0	RWP	R	R0	R0	R0	RWP	RWP	RWP	R	R	R0	R0	R0	R0	R0	R0	R0WP1	R
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0

Bit Position	Bit Field Name	Bit Description
[31:27]	read0	-
[26:24]	AP	Access Permissions for Background Region These bits are used to control access permissions for background region. For detailed description of these bits refer to Table 'Access Permissions'.
[23:18]	read0	-
[17]	MPUENC	MPU AXI Enable Control '0': MPU AXI monitoring and protection function is disabled '1': MPU AXI monitoring and protection function is enabled Read returns status of MPUENC bit. The region enable status to be effective takes some delay with reference to write on MPUENC bit. The actual enable status can be read through MPUXn_CTRL0:MPUEN bit.
[16]	MPUEN	MPU AXI Enable Status '0': MPU AXI monitoring and protection function is disabled. All accesses on AXI master write or read address channel inter- faces are passed on to AXI memory write or read address chan- nel interfaces without any protection '1': MPU AXI monitoring and protection function is enabled
[15:13]	read0	-
[12]	PROT	Privilege Attribute When privileged mode overwrite feature is available and also MPUXn_CTRL0:POEN = '1', privilege mode attribute on AXI memory interfaces are controlled by this bit. '0': Non-privilege mode '1': Privilege mode

Bit Position	Bit Field Name	Bit Description
[11]	POEN	Privileged Mode Overwrite Feature Enable '0': Privileged mode overwrite feature is disabled '1': Privileged mode overwrite feature is enabled Note: For availability of privileged mode overwrite feature, refer to device specific datasheet.
[10]	MPUSTOPEN	Enable for MPU STOP Feature '0': MPU stop feature is disabled '1': MPU stop feature is enabled This bit along with MPU stop input controls the STOP status of MPU AXI.
[9]	MPUSTOP	MPU Stop Status '0': MPU AXI is not stopped '1': MPU AXI is stopped (i.e.MPUXn_CTRL0:MPUSTOPEN = '1' and MPU stop input is asserted). All accesses on AXI master memory write or read address channel are converted to FIXED type burst with predefined address Note: For availability of MPU stop feature, refer to device specific datasheet.
[8]	LST	MPU Lock Status '0': MPU AXI is unlocked, registers in MPU AXI can be written '1': MPU AXI is locked, no registers (other than MPUXn_UNLOCK register) in MPU AXI can be written
[7:2]	read0	-
[1]	NMICL	NMI Interrupt Clear '0': No effect '1': Clears the NMI interrupt flag Read returns '0'.
[0]	NMI	NMI Interrupt Flag This bit indicates that the memory protection violation is detected for an AXI transaction.

Note:

- *The register bits MPUXn_CTRL0:AP, MPUXn_CTRL0:MPUSTOPEN, MPUXn_CTRL0:POEN and MPUXn_CTRL0:PROT in MPUXn_CTRL0 register can only be written when MPU is disabled (MPUXn_CTRL0:MPUEN = '0'). This also implies that when MPUXn_CTRL0:MPUEN = '1':*
  - *32-bit or 16-bit access to this register is not allowed*
  - *8-bit access is required to disable the MPU*
  - *8-bit access is required to clear the NMI interrupt flag*

4.2. MPU AXI NMI Enable Register (MPUXn_NMIEN)

MPU AXI NMI Enable Register can be used by software to reset NMI enable bit. Default value of NMI enable bit is '1'. This bit can be reset by software only once after reset operation.

■ MPUXn_NMIEN

MPUXn_NMIEN																															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	NMIEN
R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	RWP
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit Position	Bit Field Name	Bit Description
[31:8]	read0	-
[7:1]	read0	-
[0]	NMIEN	NMI Interrupt Enable This bit decides whether the NMI interrupt flag is routed to NMI interrupt signal or not. '0': NMI interrupt flag does not trigger NMI interrupt signal '1': NMI interrupt flag triggers NMI interrupt signal The value of this bit can be changed only once after reset.

4.3. MPU AXI Write Error Control Register (MPUXn_WERRC)

This is a read-only register that provides the software the control information of AXI transaction on AXI master write address channel interface for which memory protection violation was detected. Software can read this register to get privileged mode, burst length, burst size, and burst type information of AXI transaction.

■ MPUXn_WERRC

MPUXn_WERRC																															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	AWSIZE[2]	AWSIZE[1]	AWSIZE[0]	AWBURST[1]	AWBURST[0]	AWLEN[3]	AWLEN[2]	AWLEN[1]	AWLEN[0]	AWPROTPRIV	AWMPV
R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R	R	R	R	R	R	R	R	R	R	R
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	X	X	X	X	X	X	X	X	X	X	0

Bit Position	Bit Field Name	Bit Description
[31:16]	read0	-
[15:11]	read0	-
[10:8]	AWSIZE	AXI Transaction Burst Size This bit provides the status of AWSIZE[2:0] signals of AXI transaction for which memory protection violation is detected.
[7:6]	AWBURST	AXI Transaction Burst Type This bit provides the status of AWBURST[1:0] signals of AXI transaction for which memory protection violation is detected.
[5:2]	AWLEN	AXI Transaction Burst Length This bit provides the status of AWLEN[3:0] signals of AXI transaction for which memory protection violation is detected.
[1]	AWPROTPRIV	AXI Transaction Privileged Mode This bit provides the status of AWPROT[0] signal of AXI transaction for which memory protection violation is detected.
[0]	AWMPV	AXI Write Memory Protection Violation This bit indicates that memory protection violation is detected on AXI write address channel. Writing '1' to MPUXn_CTRL0:NMICL bit clears AWMPV bit.

4.4. MPU AXI Write Error Address Register (MPUXn_WERRA)

This is a read-only register that provides the software the write address of AXI transaction on AXI master write address channel interface for which memory protection violation was detected.

■ MPUXn_WERRA

MPUXn_WERRA																															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
AWADDR[31]	AWADDR[30]	AWADDR[29]	AWADDR[28]	AWADDR[27]	AWADDR[26]	AWADDR[25]	AWADDR[24]	AWADDR[23]	AWADDR[22]	AWADDR[21]	AWADDR[20]	AWADDR[19]	AWADDR[18]	AWADDR[17]	AWADDR[16]	AWADDR[15]	AWADDR[14]	AWADDR[13]	AWADDR[12]	AWADDR[11]	AWADDR[10]	AWADDR[9]	AWADDR[8]	AWADDR[7]	AWADDR[6]	AWADDR[5]	AWADDR[4]	AWADDR[3]	AWADDR[2]	AWADDR[1]	AWADDR[0]
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit Position	Bit Field Name	Bit Description
[31:0]	AWADDR	AXI Write Address Write address of AXI transaction for which memory protection violation is detected.

4.5. MPU AXI Read Error Control Register (MPUXn_RERRC)

This is a read-only register that provides the software the control information of AXI transaction on AXI master read address channel interface for which memory protection violation was detected. Software can read this register to get privileged mode, burst length, burst size, and burst type information of AXI transaction.

■ MPUXn_RERRC

MPUXn_RERRC																															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	ARSIZE[2]	ARSIZE[1]	ARSIZE[0]	ARBURST[1]	ARBURST[0]	ARLEN[3]	ARLEN[2]	ARLEN[1]	ARLEN[0]	ARPROTPRIV	ARMPV
R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R	R	R	R	R	R	R	R	R	R	R
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	X	X	X	X	X	X	X	X	X	X	0

Bit Position	Bit Field Name	Bit Description
[31:16]	read0	-
[15:11]	read0	-
[10:8]	ARSIZE	AXI Transaction Burst Size This bit provides the status of ARSIZE[2:0] signals of AXI transaction for which memory protection violation is detected.
[7:6]	ARBURST	AXI Transaction Burst Type This bit provides the status of ARBURST[1:0] signals of AXI transaction for which memory protection violation is detected.
[5:2]	ARLEN	AXI Transaction Burst Length This bit provides the status of ARLEN[3:0] signals of AXI transaction for which memory protection violation is detected.
[1]	ARPROTPRIV	AXI Transaction Privileged Mode This bit provides the status of ARPROT[0] signal of AXI transaction for which memory protection violation is detected.
[0]	ARMPV	AXI Read Memory Protection Violation This bit indicates that memory protection violation is detected on AXI read address channel. Writing '1' to MPUXn_CTRL0:NMICL bit clears ARMPV bit.

4.6. MPU AXI Read Error Address Register (MPUXn_RERRA)

This is a read-only register that provides the software the read address of AXI transaction on AXI master read address channel interface for which memory protection violation was detected.

■ MPUXn_RERRA

MPUXn_RERRA																															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
ARADDR[31]	ARADDR[30]	ARADDR[29]	ARADDR[28]	ARADDR[27]	ARADDR[26]	ARADDR[25]	ARADDR[24]	ARADDR[23]	ARADDR[22]	ARADDR[21]	ARADDR[20]	ARADDR[19]	ARADDR[18]	ARADDR[17]	ARADDR[16]	ARADDR[15]	ARADDR[14]	ARADDR[13]	ARADDR[12]	ARADDR[11]	ARADDR[10]	ARADDR[9]	ARADDR[8]	ARADDR[7]	ARADDR[6]	ARADDR[5]	ARADDR[4]	ARADDR[3]	ARADDR[2]	ARADDR[1]	ARADDR[0]
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit Position	Bit Field Name	Bit Description
[31:0]	ARADDR	AXI Read Address Read address of AXI transaction for which memory protection violation is detected.

4.7. MPU AXI Region Control Registers (MPUXn_CTRL1 to 8)

MPU AXI 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. MPUXn_CTRL1 Control Register for Region 1 is explained below.

■ MPUXn_CTRL1

MPUXn_CTRL1																															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	AP[2]	AP[1]	AP[0]	read0	read0	read0	read0	read0	read0	MPUENC	MPUEN
R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	RWP	RWP	RWP	R0	R0	R0	R0	R0	R0	RWP	R
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit Position	Bit Field Name	Bit Description
[31:16]	read0	-
[15:11]	read0	-
[10:8]	AP	Access Permissions These bits are used to control access permissions for region 1.
[7:2]	read0	-
[1]	MPUENC	Enable Control bit '0': Memory protection for region 1 is disabled '1': Memory protection for region 1 is enabled Read returns status of MPUENC bit. The region enable status to be effective takes some delay with reference to write on MPUENC bit. The actual enable status can be read through MPUXn_CTRL1:MPUEN bit.
[0]	MPUEN	Enable Status '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 does not have any effect.

Note:

- *Access permission bits (i.e. MPUXn_CTRL1 to 8:AP) can only be written when corresponding region is disabled (MPUXn_CTRL1 to 8:MPUEN = '0') or MPU is disabled (MPUXn_CTRL0:MPUEN = '0').*
- *This also implies that when MPUXn_CTRL0:MPUEN = '1' and MPUXn_CTRL1 to 8:MPUEN = '1':*
  - *32-bit or 16-bit access to this register is not allowed*
  - *8-bit access is required to disable the MPU region*

4.8. MPU AXI Start Address Registers (MPUXn_SADDR1 to 8)

Each region in MPU AXI can be set by specifying start address and end address for that particular region. MPUXn_SADDR1 to 8 registers are used to specify start address for eight regions. Start address indicates the first address for that region. MPUXn_SADDR1 Start Address Register for Region 1 is explained below.

■ MPUXn_SADDR1

MPUXn_SADDR1																															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
SADDR[31]	SADDR[30]	SADDR[29]	SADDR[28]	SADDR[27]	SADDR[26]	SADDR[25]	SADDR[24]	SADDR[23]	SADDR[22]	SADDR[21]	SADDR[20]	SADDR[19]	SADDR[18]	SADDR[17]	SADDR[16]	SADDR[15]	SADDR[14]	SADDR[13]	SADDR[12]	SADDR[11]	SADDR[10]	SADDR[9]	SADDR[8]	SADDR[7]	SADDR[6]	SADDR[5]	SADDR[4]	SADDR[3]	SADDR[2]	SADDR[1]	SADDR[0]
RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit Position	Bit Field Name	Bit Description
[31:0]	SADDR	Start Address Start address for region 1

Note:

- *MPUXn_SADDR1 to 8 registers can only be written when corresponding region is disabled (MPUXn_CTRL1 to 8:MPUEN = '0') or MPU is disabled (MPUXn_CTRL0:MPUEN = '0').*

4.9. MPU AXI End Address Registers (MPUXn_EADDR1 to 8)

Each region in MPU AXI can be set by specifying start address and end address for that particular region. MPUXn_EADDR1 to 8 registers are used to specify end addresses for eight regions. End Address indicates the last address for that region. MPUXn_EADDR1 End Address Register for Region 1 is explained below.

■ MPUXn_EADDR1

MPUXn_EADDR1																															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
EADDR[31]	EADDR[30]	EADDR[29]	EADDR[28]	EADDR[27]	EADDR[26]	EADDR[25]	EADDR[24]	EADDR[23]	EADDR[22]	EADDR[21]	EADDR[20]	EADDR[19]	EADDR[18]	EADDR[17]	EADDR[16]	EADDR[15]	EADDR[14]	EADDR[13]	EADDR[12]	EADDR[11]	EADDR[10]	EADDR[9]	EADDR[8]	EADDR[7]	EADDR[6]	EADDR[5]	EADDR[4]	EADDR[3]	EADDR[2]	EADDR[1]	EADDR[0]
RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1

Bit Position	Bit Field Name	Bit Description
[31:0]	EADDR	End Address End address for region 1

Note:

- *MPUXn_EADDR1 to 8 registers can only be written when corresponding region is disabled (MPUXn_CTRL1 to 8:MPUEN = '0') or MPU is disabled (MPUXn_CTRL0:MPUEN = '0').*

4.10. MPU AXI Unlock Register (MPUXn_UNLOCK)

The software can use this register to lock or unlock the MPU AXI registers for write access.

■ MPUXn_UNLOCK

MPUXn_UNLOCK																															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
UNLOCK[31]	UNLOCK[30]	UNLOCK[29]	UNLOCK[28]	UNLOCK[27]	UNLOCK[26]	UNLOCK[25]	UNLOCK[24]	UNLOCK[23]	UNLOCK[22]	UNLOCK[21]	UNLOCK[20]	UNLOCK[19]	UNLOCK[18]	UNLOCK[17]	UNLOCK[16]	UNLOCK[15]	UNLOCK[14]	UNLOCK[13]	UNLOCK[12]	UNLOCK[11]	UNLOCK[10]	UNLOCK[9]	UNLOCK[8]	UNLOCK[7]	UNLOCK[6]	UNLOCK[5]	UNLOCK[4]	UNLOCK[3]	UNLOCK[2]	UNLOCK[1]	UNLOCK[0]
R0WP	R0WP	R0WP	R0WP	R0WP	R0WP	R0WP	R0WP	R0WP	R0WP	R0WP	R0WP	R0WP	R0WP	R0WP	R0WP	R0WP	R0WP	R0WP	R0WP	R0WP	R0WP	R0WP	R0WP	R0WP	R0WP	R0WP	R0WP	R0WP	R0WP	R0WP	R0WP
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit Position	Bit Field Name	Bit Description
[31:0]	UNLOCK	MPU AXI Unlock The MPU AXI Unlock Register protects the MPU AXI module from being modified accidently by software. The MPU AXI 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 a zero. To lock the MPU AXI again software must write another value specific to lock. Write access to MPU AXI registers without unlocking or writing a value other than the lock or unlock values to this register causes a protection error. Notes: This register can not be written by 8-bit or 16-bit write access; any such access causes protection error. For more details on lock and unlock values, refer to the device specific datasheet.

4.11. MPU AXI Module ID Register (MPUXn_MID)

This is a read-only register with a unique module identification number which identifies the version of the MPU AXI module used in the MCU.

■ MPUXn_MID

MPUXn_MID																															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
MID[31]	MID[30]	MID[29]	MID[28]	MID[27]	MID[26]	MID[25]	MID[24]	MID[23]	MID[22]	MID[21]	MID[20]	MID[19]	MID[18]	MID[17]	MID[16]	MID[15]	MID[14]	MID[13]	MID[12]	MID[11]	MID[10]	MID[9]	MID[8]	MID[7]	MID[6]	MID[5]	MID[4]	MID[3]	MID[2]	MID[1]	MID[0]
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	1	0	1	0	0	0	0	0	0	0	0

Bit Position	Bit Field Name	Bit Description
[31:0]	MID	Module ID The MPU AXI module implemented in the device may vary from device to device. This register identifies the particular version of the hardware used in the device. This register helps in developing software to hardware version implemented in the device

5. Notes on Using MPU AXI

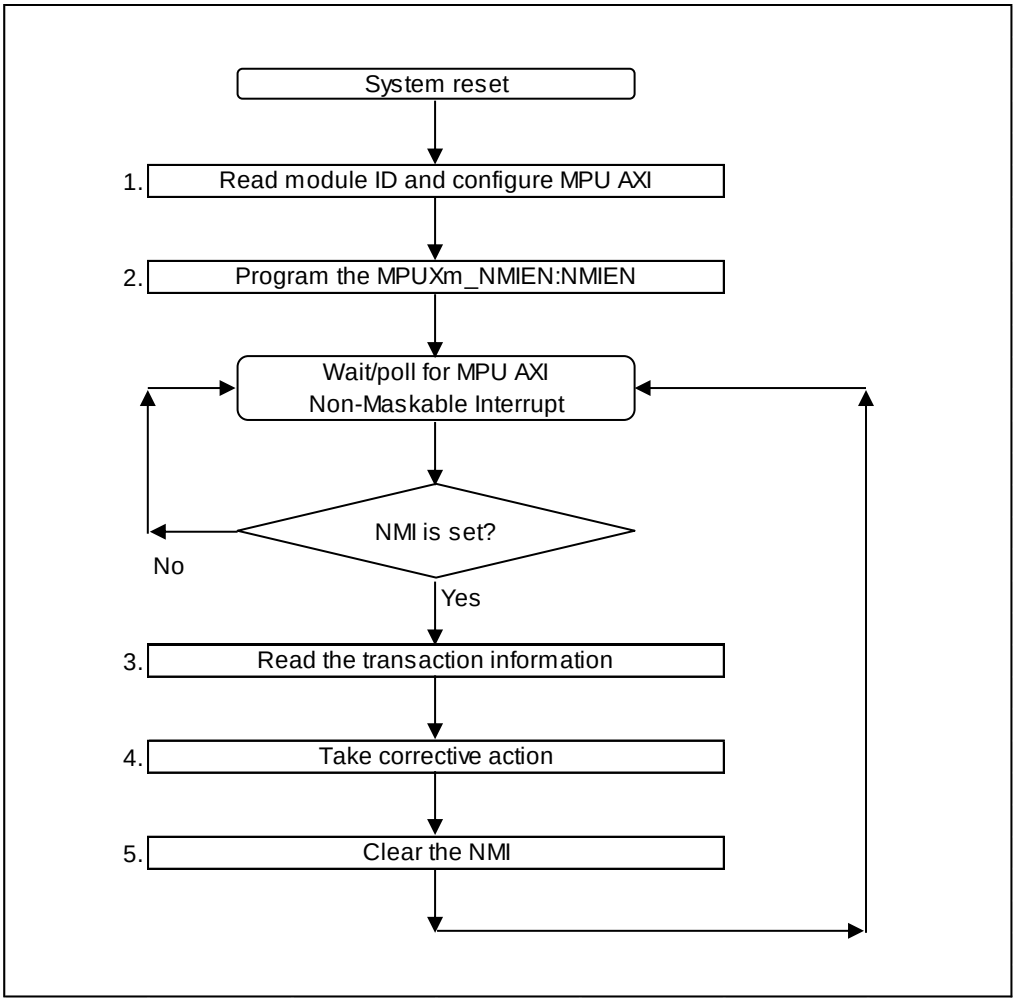
This section is the 'programmer's guide', which lists the usage notes for programming the MPU AXI module. It is recommended to read these guidelines before programming the MPU AXI module.

■ General Usage Notes

- Reserved bits return undefined values. The software programs shall be independent of the values read from the reserved register bits
- The MPU AXI supports storage of information of first memory protection violation on the bus. Therefore, once an NMI interrupt flag is set, the further monitoring of AXI master interface is stalled until MPUXn_CTRL0:NMI bit is cleared. This implies that any protection violation occurring on the bus while the MPUXn_CTRL0:NMI is set are simply ignored by MPU AXI. Software developers must therefore try to keep the ISR size small, so that the NMI interrupt of MPU AXI does not remain unattended for long.

■ Steps in Programming the MPU AXI Module

Figure 30-4 Programmer's Flowchart



1. After the system reset, the software detects the module ID number of MPU AXI, by reading the MPUXn_MID register. This helps it in identifying the attributes and capabilities supported by the MPU AXI module. Then the software configures MPU AXI by setting appropriate registers.

2. By default, MPU AXI propagates the MPUXn_CTRL0:NMI flag to the CPU through the Interrupt Controller. If polling mode is desired, the software can reset the MPUXn_NMIEN:NMIEN bit to '0'.

Note:

- *The MPUXn_NMIEN:NMIEN can be written only once after reset. Subsequent write accesses to this bit have no visible impact on the state of this bit.*

3. When the NMI is triggered or is in polling mode, if the software detects during its polling cycle that the MPUXn_CTRL0:NMI status flag is set, the CPU is invoked. This would read the status information collected and stored by MPU AXI in its CSR.

4. The software diagnoses the information about the transaction for which memory protection violation was detected and initiates a corrective action (if any).

5. Once the software has processed the information from the status registers, it shall clear the MPUXn_CTRL0:NMI flag by writing a '1' to the MPUXn_CTRL0:NMICL bit. Clearing MPUXn_CTRL0:NMI flag ensures that the MPU AXI starts monitoring the AXI master interfaces again for checking memory protection violation.

Note:

- *Software may clear the NMI flag before taking corrective action, hence steps 4 and 5 can be interchanged.*

CHAPTER 31: Memory Protection Unit for AHB

This chapter explains the function and operation of the Memory Protection Unit for the AMBA Advanced High Speed Bus (MPU AHB) in the Next Generation Microcontrollers.

1. Overview
2. Configuration and Block Diagram
3. Operation of the MPU AHB
4. Registers
5. Notes on Using the MPU AHB

CODE: MPUAHB-S6J3300-E1

1. Overview

This section describes the features 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 (AP) bits. These AP bits are explained in Section MPU access permissions.

The MPU AHB provides eight 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:

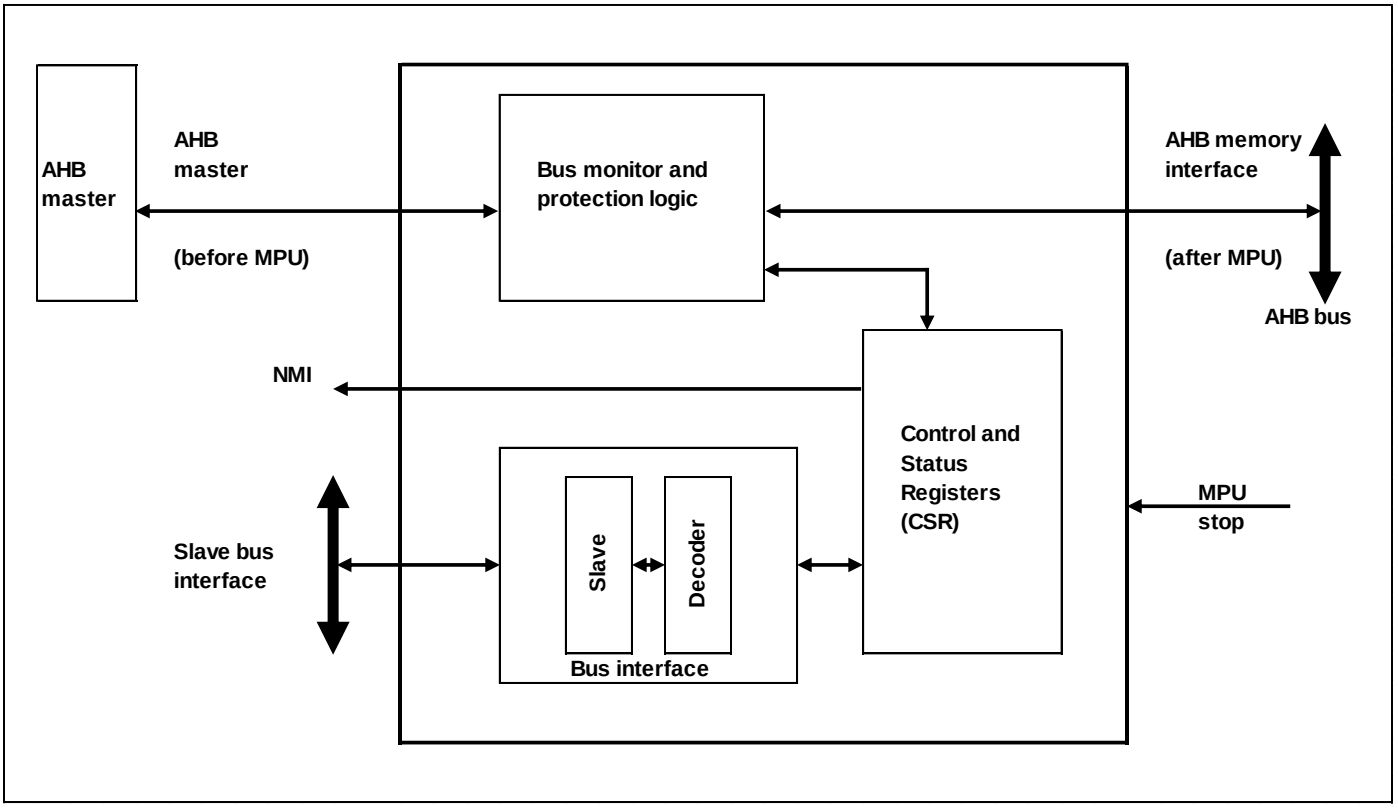
- Each of the eight regions in MPU AHB is specified using corresponding start address and end address
- The background region covers the entire 4 GB address space
- With an unauthorized access, the MPU AHB generates an NMI to the CPU
- The MPU AHB collects information about the AHB Master bus access 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
- It supports an MPU STOP feature that allows blocking of all the accesses to memory space
- It optionally supports a privileged mode overwrite feature that allows the privilege attribute of the memory side AHB interface to be overwritten

2. Configuration and Block Diagram

This section shows a block diagram of the MPU AHB.

■ Block Diagram of the MPU AHB

Figure 31-1 Block Diagram of MPU AHB



■ Bus Interface

The bus masters can access the MPU AHB module through its slave bus interface.

■ 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.

■ Control and Status Registers

The operation of the MPU AHB can be controlled and monitored through its Control and Status Registers (CSR).

3. Operation of the MPU AHB

This section describes the operation of the MPU AHB.

■ MPU AHB Region Granularity and Priority Decision

The MPU AHB supports up to 8 regions and provides the start and end addresses for each of these eight regions. MPU AHB regions are defined with a granularity of 32 bytes.

Each region is identified as Region 1, Region 2, Region 3 and so on up to Region 8, which is given highest priority. Region 7 has second highest priority and so on down to Region 1, which has the second lowest priority. The Background Region is designated the lowest priority region.

The Start Address specifies the first address of the region and is specified by registers MPUHn_SADDR1 to MPUHn_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.

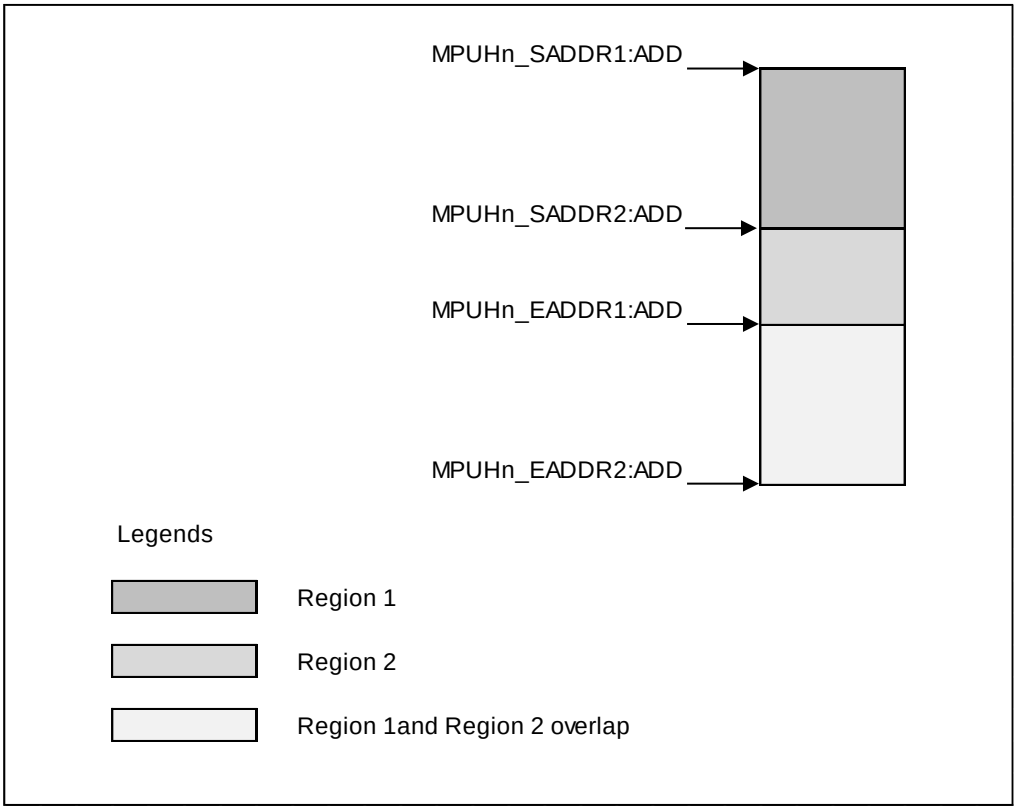
End Addresses are specified by registers MPUHn_EADDR1 to MPUHn_EADDR8n for region 1 to region 8 respectively. The least significant 5 bits of the End Address registers are read only bits and will always read 1. This ensures the granularity of 32 bytes.

In the MPU AHB, regions may overlap each other, as shown in Figure 31-2. Hence it is also possible that an AHB address for any transfer may match with multiple regions. Whenever the AHB Transfer Address matches with multiple regions, the permissions corresponding to the highest priority region (among all matching regions) is applied.

AHB Burst Monitoring

The MPU AHB only supports Single Transfer Monitoring and not AHB Burst Monitoring. Therefore, it can only be used together with AHB Masters that do not request Burst transfers.

Figure 31-2 Example Region Overlapping



MPU Access Permissions

Region Control registers in the MPU AHB (i.e. MPUHn_CTRL1 to MPUHn_CTRL8) are used to control the access permission to Regions 1 to 8. Also MPUHn_CTRL0 is used to control the access permission for the Background Region.

Table 31-1 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.

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 eight defined regions
- As explained in Section “MPU AHB region granularity and priority decision”, 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 (MPUHn_CTRL0:NMI) is set. The address and control information of the current transfer is stored in MPUHn_MERRA and MPUHn_MERRC respectively
- All further transfers are blocked until the MPUHn_CTRL0:NMI flag is cleared by software. Further monitoring of the AHB transfer addresses is also stalled until the MPUHn_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

MPU STOP Feature

Optionally the MPU AHB supports an MPU STOP feature.

When this mode is enabled, all accesses to memory space are blocked and 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

Privileged Mode Overwrite Feature

Optionally 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 MPUHn_CTRL0:PROT bit

Note:
Bus monitor and protection logic for detecting memory protection violation uses the privileged mode attribute on the AHB master interface and not the MPUHn_CTRL0:PROT bit, even when the privileged mode overwrite feature is enabled.

4. Registers

This section describes the registers of the MPU AHB

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 (CSRs). The address area allocated to the MPU AHB and the CSRs in the MPU AHB are explained in this section.

The suffix ‘n’ in the register name indicates that the register is an instance ‘n’ of the module.

■ Registers of MPU AHB

The following registers are available for the MPU AHB:

- MPU AHB Control Register (MPUHn_CTRL0)
- MPU AHB NMI Enable Register (MPUHn_NMIEN)
- MPU AHB Memory Error Control Register (MPUHn_MERRC)
- MPU AHB Memory Error Address Register (MPUHn_MERRA)
- MPU AHB Region Control Registers (MPUHn_CTRL1 to 8)
- MPU AHB Start Address Registers (MPUHn_SADDR1 to 8)
- MPU AHB End Address Registers (MPUHn_EADDR1 to 8)
- MPU AHB Unlock Register (MPUHn_UNLOCK)
- MPU AHB Module ID Register (MPUHn_MID)

■ Memory Layout of MPU AHB Registers

Offset	+3	+2	+1	+0
0x00000000	MPUHn_CTRL0 00000000 00000000 00000001 00000000			
0x00000004	MPUHn_NMIEN 00000000 00000000 00000000 00000001			
0x00000008	MPUHn_MERRC 00000000 00000000 00000000 00000000			
0x0000000C	MPUHn_MERRA 00000000 00000000 00000000 00000000			
0x00000010	MPUHn_CTRL1 00000000 00000000 00000000 00000000			
0x00000014	MPUHn_SADDR1 00000000 00000000 00000000 00000000			
0x00000018	MPUHn_EADDR1 00000000 00000000 00000000 00011111			
0x0000001C	MPUHn_CTRL2 00000000 00000000 00000000 00000000			
0x00000020	MPUHn_SADDR2 00000000 00000000 00000000 00000000			

Offset	+3	+2	+1	+0
0x00000024	MPUHn_EADDR2 00000000 00000000 00000000 00011111			
0x00000028	MPUHn_CTRL3 00000000 00000000 00000000 00000000			
0x0000002C	MPUHn_SADDR3 00000000 00000000 00000000 00000000			
0x00000030	MPUHn_EADDR3 00000000 00000000 00000000 00011111			
0x00000034	MPUHn_CTRL4 00000000 00000000 00000000 00000000			
0x00000038	MPUHn_SADDR4 00000000 00000000 00000000 00000000			
0x0000003C	MPUHn_EADDR4 00000000 00000000 00000000 00011111			
0x00000040	MPUHn_CTRL5 00000000 00000000 00000000 00000000			
0x00000044	MPUHn_SADDR5 00000000 00000000 00000000 00000000			
0x00000048	MPUHn_EADDR5 00000000 00000000 00000000 00011111			
0x0000004C	MPUHn_CTRL6 00000000 00000000 00000000 00000000			
0x00000050	MPUHn_SADDR6 00000000 00000000 00000000 00000000			
0x00000054	MPUHn_EADDR6 00000000 00000000 00000000 00011111			
0x00000058	MPUHn_CTRL7 00000000 00000000 00000000 00000000			
0x0000005C	MPUHn_SADDR7 00000000 00000000 00000000 00000000			
0x00000060	MPUHn_EADDR7 00000000 00000000 00000000 00011111			
0x00000064	MPUHn_CTRL8 00000000 00000000 00000000 00000000			
0x00000068	MPUHn_SADDR8 00000000 00000000 00000000 00000000			
0x0000006C	MPUHn_EADDR8 00000000 00000000 00000000 00011111			
0x00000070	MPUHn_UNLOCK 00000000 00000000 00000000 00000000			
0x00000074	MPUHn_MID 00000000 00000000 00000000 00000000			

4.1. MPU AHB Control Register (MPUHn_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 and MPU STOP feature. Lastly it provides status of MPU AHB lock bit and Non-Maskable Interrupt flag.

MPUHn_CTRL0																															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
read0	read0	read0	read0	read0	AP[2]	AP[1]	AP[0]	read0	read0	read0	read0	read0	read0	MPUENC	MPUEN	read0	read0	read0	PROT	POEN	MPUSTOPEN	MPUSTOP	LST	read0	read0	read0	read0	read0	read0	NMICL	NMI
R0	R0	R0	R0	R0	RWP	RWP	RWP	R0	R0	R0	R0	R0	R0	R0WP	R	R0	R0	R0	RWP	RWP	RWP	R	R	R0	R0	R0	R0	R0	R0	R0WP1	R
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0

Bit Position	Bit Field Name	Bit Description
[31:27]	read0	-
[26:24]	AP	Access Permissions for Background Region These bits are used to control access permissions for a background region. For more details about these bits, refer to Table 7.3-1. Note: AP[2:0] bits for background region can only be written when the MPU is disabled (MPUHn_CTRL0:MPUEN = '0').
[23:18]	read0	-
[17]	MPUENC	MPU AHB Enable Control '0': MPU AHB Monitoring and Protection Function is disabled '1': MPU AHB Monitoring and Protection Function is enabled Read returns '0'.
[16]	MPUEN	MPU AHB Enable Status '0': MPU AHB Monitoring and Protection Function is disabled. All accesses from the AHB Master Interface are passed on to the AHB memory interface without any protection '1': MPU AHB Monitoring and Protection Function is enabled
[15:13]	read0	-
[12]	PROT	Privileged Mode Attribute When the privileged mode overwrite feature is available and POEN = '1', the privilege attribute on AHB memory interface is controlled by this bit. '0': Non-privileged mode '1': Privileged mode

Bit Position	Bit Field Name	Bit Description
[11]	POEN	Privileged Mode Overwrite Feature Enable '0': Privileged mode overwrite feature is disabled '1': Privileged mode overwrite feature is enabled Note: For availability of the privileged mode overwrite feature, refer to the device-specific datasheet.
[10]	MPUSTOPEN	Enable for MPU STOP Feature
[9]	MPUSTOP	MPU STOP Status '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 Note: For availability of the MPU STOP feature, refer to the devicespecific datasheet.
[8]	LST	MPU Lock Status '0': MPU AHB is unlocked; registers in MPU AHB can be written '1': MPU AHB is locked; no registers (other than MPUHn_UNLOCK register) in MPU AHB can be written
[7:2]	read0	-
[1]	NMICL	NMI Interrupt Clear '0': No effect '1': Clears the NMI interrupt flag Read returns '0'.
[0]	NMI	NMI Interrupt Flag This interrupt flag indicates that the memory protection violation was detected for an AHB transfer.

Note:

- *The register bits MPUHn_CTRL0:AP[2:0], MPUHn_CTRL0:MPUSTOPEN, MPUHm_CTRL0:POEN and MPUHn_CTRL0:PROT can only be written when the MPU is disabled (MPUHn_CTRL0:MPUEN = 0).*

4.2. MPU AHB NMI Enable Register (MPUHn_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.

MPUHn_NMIEN																															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	NMIEN
R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	RWP
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit Position	Bit Field Name	Bit Description
[31:8]	read0	-
[7:1]	read0	-
[0]	NMIEN	NMI Interrupt Enable This bit decides whether the NMI interrupt flag is routed to an NMI interrupt signal or not. '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 (MPUHn_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.

MPUHn_MERRC																															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	HPROT	HWRITE
R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R	R
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit Position	Bit Field Name	Bit Description
[31:8]	read0	-
[7:2]	read0	-
[1]	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.
[0]	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 (MPUHn_MERRA)

This is a read-only register that provides the address of AHB transfer for which memory protection violation was detected.

MPUHn_MERRA																															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
HADDR[31]	HADDR[30]	HADDR[29]	HADDR[28]	HADDR[27]	HADDR[26]	HADDR[25]	HADDR[24]	HADDR[23]	HADDR[22]	HADDR[21]	HADDR[20]	HADDR[19]	HADDR[18]	HADDR[17]	HADDR[16]	HADDR[15]	HADDR[14]	HADDR[13]	HADDR[12]	HADDR[11]	HADDR[10]	HADDR[9]	HADDR[8]	HADDR[7]	HADDR[6]	HADDR[5]	HADDR[4]	HADDR[3]	HADDR[2]	HADDR[1]	HADDR[0]
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit Position	Bit Field Name	Bit Description
[31:0]	HADDR	AHB Address. The address of an AHB transfer for which memory protection violation was detected.

4.5. MPU AHB Region Control Registers (MPUHn_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. MPUHn_CTRL1 Control Register for Region 1 is explained below.

Refer to the device-specific datasheet for the number of regions available.

MPUHn_CTRL1																															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	read0	AP[2]	AP[1]	AP[0]	read0	read0	read0	read0	read0	read0	MPUENC	MPUEN
R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	R0	RWP	RWP	RWP	R0	R0	R0	R0	R0	R0	RWP	R
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit Position	Bit Field Name	Bit Description
[31:16]	read0	-
[15:11]	read0	-
[10:8]	AP	Access Permissions These bits are used to control access permissions for region 1. For a detailed description of these bits refer to Table 7.3-1.
[7:2]	read0	-
[1]	MPUENC	Enable Control '0': Memory Protection for region 1 is disabled '1': Memory Protection for region 1 is enabled Read returns '0'. The enable status of the region can be read through the MPUHn_CTRL1:MPUEN bit.
[0]	MPUEN	Enable Status '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:
For all eight MPU AHB Region Control registers, the Access Permission bits (i.e. MPUHn_CTRL1 to 8:AP) can only be written when the corresponding region is disabled (i.e. MPUHn_CTRL1 to 8:MPUEN = 0) or the MPU is disabled (MPUHn_CTRL0:MPUEN = 0). This also implies that when MPUHn_CTRL0:MPUEN = 1 and MPUHn_CTRL1 to 8:MPUEN = 1:

- 32- or 16-bit access to this register is not allowed
- 8-bit access is required to disable the MPU

4.6. MPU AHB Start Address Registers (MPUHn_SADDR1 to 8)

Each region in MPU AHB can be defined by specifying the start address and end address for that particular region. The MPUHn_SADDR1 to 8 registers are used to specify the start address for the eight regions. The start address indicates the first address for that region. MPUHn_SADDR1 is the Start Address Register for Region 1 and is explained below. Refer to the device-specific datasheet for the number of regions available .

MPUHn_SADDR1

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
SADDR[31]	SADDR[30]	SADDR[29]	SADDR[28]	SADDR[27]	SADDR[26]	SADDR[25]	SADDR[24]	SADDR[23]	SADDR[22]	SADDR[21]	SADDR[20]	SADDR[19]	SADDR[18]	SADDR[17]	SADDR[16]	SADDR[15]	SADDR[14]	SADDR[13]	SADDR[12]	SADDR[11]	SADDR[10]	SADDR[9]	SADDR[8]	SADDR[7]	SADDR[6]	SADDR[5]	SADDR[4]	SADDR[3]	SADDR[2]	SADDR[1]	SADDR[0]
RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	R0	R0	R0	R0	R0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit Position	Bit Field Name	Bit Description
[31:0]	SADDR	Start Address Start address for region 1

Note:
the MPUHn_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 (MPUHn_CTRL0:MPUEN = 0).

4.7. MPU AHB End Address Registers (MPUHn_EADDR1 to 8)

Each region in MPU AHB can be defined by specifying a start address and end address for that particular region. MPUHn_EADDR1 to 8 registers are used to specify the end addresses for the eight regions. The end address indicates the last address for that region. The MPUHn_EADDR1 End Address Register for Region 1 is explained below. Refer to the device-specific datasheet for the number of regions available.

MPUHn_EADDR1

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
EADDR[31]	EADDR[30]	EADDR[29]	EADDR[28]	EADDR[27]	EADDR[26]	EADDR[25]	EADDR[24]	EADDR[23]	EADDR[22]	EADDR[21]	EADDR[20]	EADDR[19]	EADDR[18]	EADDR[17]	EADDR[16]	EADDR[15]	EADDR[14]	EADDR[13]	EADDR[12]	EADDR[11]	EADDR[10]	EADDR[9]	EADDR[8]	EADDR[7]	EADDR[6]	EADDR[5]	EADDR[4]	EADDR[3]	EADDR[2]	EADDR[1]	EADDR[0]
RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	RWP	R1	R1	R1	R1	R1
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1

Bit Position	Bit Field Name	Bit Description
[31:0]	EADDR	End Address End address for region 1

Note:
MPUHn_EADDR1 to 8 registers can only be written when the corresponding region is disabled (MPUHn_CTRL1 to 8:MPUEN = 0) or the MPU is disabled (MPUHn_CTRL0:MPUEN = 0).

4.8. MPU AHB Unlock Register (MPUHn_UNLOCK)

The software can use this register to lock or unlock the MPU AHB registers for write access.

MPUHn_UNLOCK																															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
UNLOCK[31]	UNLOCK[30]	UNLOCK[29]	UNLOCK[28]	UNLOCK[27]	UNLOCK[26]	UNLOCK[25]	UNLOCK[24]	UNLOCK[23]	UNLOCK[22]	UNLOCK[21]	UNLOCK[20]	UNLOCK[19]	UNLOCK[18]	UNLOCK[17]	UNLOCK[16]	UNLOCK[15]	UNLOCK[14]	UNLOCK[13]	UNLOCK[12]	UNLOCK[11]	UNLOCK[10]	UNLOCK[9]	UNLOCK[8]	UNLOCK[7]	UNLOCK[6]	UNLOCK[5]	UNLOCK[4]	UNLOCK[3]	UNLOCK[2]	UNLOCK[1]	UNLOCK[0]
R0WP	R0WP	R0WP	R0WP	R0WP	R0WP	R0WP	R0WP	R0WP	R0WP	R0WP	R0WP	R0WP	R0WP	R0WP	R0WP	R0WP	R0WP	R0WP	R0WP	R0WP	R0WP	R0WP	R0WP	R0WP	R0WP	R0WP	R0WP	R0WP	R0WP	R0WP	R0WP
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit Position	Bit Field Name	Bit Description
[31:0]	UNLOCK	MPU AHB Unlock The MPU AHB Unlock Register protects the MPU AHB module from being modified accidently 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. **Notes:** This register cannot be written by an 8- or 16-bit write access; any such access causes a protection error. For more details on the lock and unlock values, refer to the device-specific datasheet.

4.9. MPU AHB Module ID Register (MPUHn_MID)

This is a read-only register with a unique module identification number which identifies the version of the MPU AHB module used in the MCU. Refer to the device-specific datasheet for the module identification number of the MPU AHB module.

■ MPU AHB Module ID Register (MPUHn_MID)

MPUHn_MID																															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
MID[31]	MID[30]	MID[29]	MID[28]	MID[27]	MID[26]	MID[25]	MID[24]	MID[23]	MID[22]	MID[21]	MID[20]	MID[19]	MID[18]	MID[17]	MID[16]	MID[15]	MID[14]	MID[13]	MID[12]	MID[11]	MID[10]	MID[9]	MID[8]	MID[7]	MID[6]	MID[5]	MID[4]	MID[3]	MID[2]	MID[1]	MID[0]
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit Position	Bit Field Name	Bit Description
[31:0]	MID	Module ID The MPU AHB module implemented in the device may vary from device to device. This register identifies the particular version of the hardware used in the device. This register helps in developing the software to hardware version implemented in the device Note: For more details refer to the device-specific datasheet for the Module ID number.

5. Notes on Using the MPU AHB

This section is the 'Programmer's Guide', which lists the usage notes for programming the MPU AHB module. Please read these guidelines before programming the module.

■ General Usage Notes

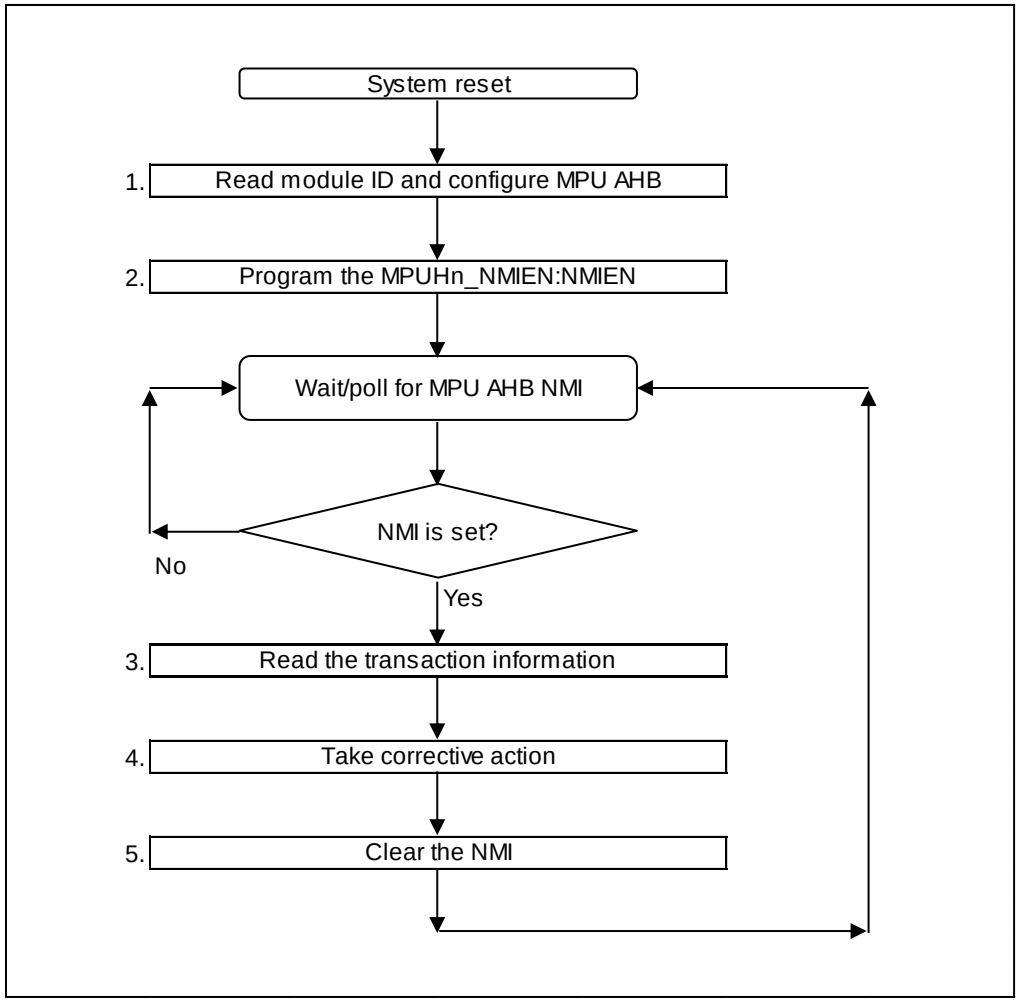
- Reserved bits return undefined values. The software programs shall be independent of the values read from the reserved register bits
- The MPU AHB supports storage of information of the first memory protection violation only on the bus. Therefore, once an NMI interrupt flag is set, further monitoring of the AHB master interface is stalled until the MPUHn_CTRL0:NMI bit is cleared. This implies that any memory protection violation occurring on the bus while the MPUHn_CTRL0:NMI is set is simply ignored by the MPU AHB

Note:

- *Software developers must therefore try to keep the ISR size small so that the NMI interrupt of the MPU AHB does not remain unattended for long.*

■ Steps in Programming the MPU AHB Module

Figure 31-3 Programmer's Flowchart



1. After the system reset, the software detects the module ID number of the MPU AHB by reading the MPUHn_MID register. This helps it in identifying the attributes and capabilities supported by the MPU AHB module. The software then configures the MPU AHB by setting the appropriate registers.

2. By default, the MPU AHB propagates the MPUHn_CTRL0:NMI flag to the CPU through the Interrupt Controller. If polling mode is desired, the software can reset the MPUHn_NMIEN:NMIEN bit to '0'.

Note:
The MPUHm_NMIEN:NMIEN can be written only once after reset. Subsequent write accesses to this bit have 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 MPUHn_CTRL0:NMI status flag is set), the CPU is invoked and reads the status information collected and stored by the 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 shall clear the MPUHn_CTRL0:NMI flag by writing "1" to the MPUHn_CTRL0:NMICL bit. Clearing the MPUHn_CTRL0:NMI flag ensures that the MPU AHB starts monitoring the AHB Master Interface again to check for memory protection violation.

Note:
- *Software may clear the NMI flag before taking corrective action; therefore steps 4 and 5 could be interchanged.*

CHAPTER 32: Sound System Configuration

This chapter explains the definition of sound system and configuration.

1. Overview
2. Configuration and Block Diagram
3. Operation
4. Note

CODE: SOUNDSYSTEM_CONFIG-S6J3300-E1

1. Overview

The sound system is defined as the function group of following modules.

Function Group	Function Module
Input module	DMAC
	Sound waveform generator (SWFG)
Sound mixer	Sound mixer (SMIX)
Output module	Audio DAC (DAC)
	PCM-PWM
	I2S

Here the configuration procedure and notes are described in this chapter for each applied scope when the modules are used as the sound system.

See the dedicated chapter that describes in detailed each module composing the sound system before referring this chapter.

1.1. Input Module

There are 2 types of sound sources.

One is a sound source that is generated by the sound waveform generator (SWFG). SWFG with easy configurations by software can generate 16 bit sound sources of PCM format, and output it as 32 bit monaural source.

The other is a sound source that is in internal or external storage. DMAC or CPU can transfer sound source in the embedded Flash memory or data RAM, or comes from external memory via communication interface such as I2S.

1.2. Sound Mixer

The sound mixer (SMIX) synthesizes inputted sound sources by saturation calculation, and performs various effects of sound volume (configured volume, mute effect, fade-in, or fade out) automatically by hardware operation.

1.3. Output Module

The mixed sound source is outputted out of microcontroller.

Audio DAC converts PCM sound source to level of analog voltage. It enables to mount analog amplifier directly at analog output port of MCU.

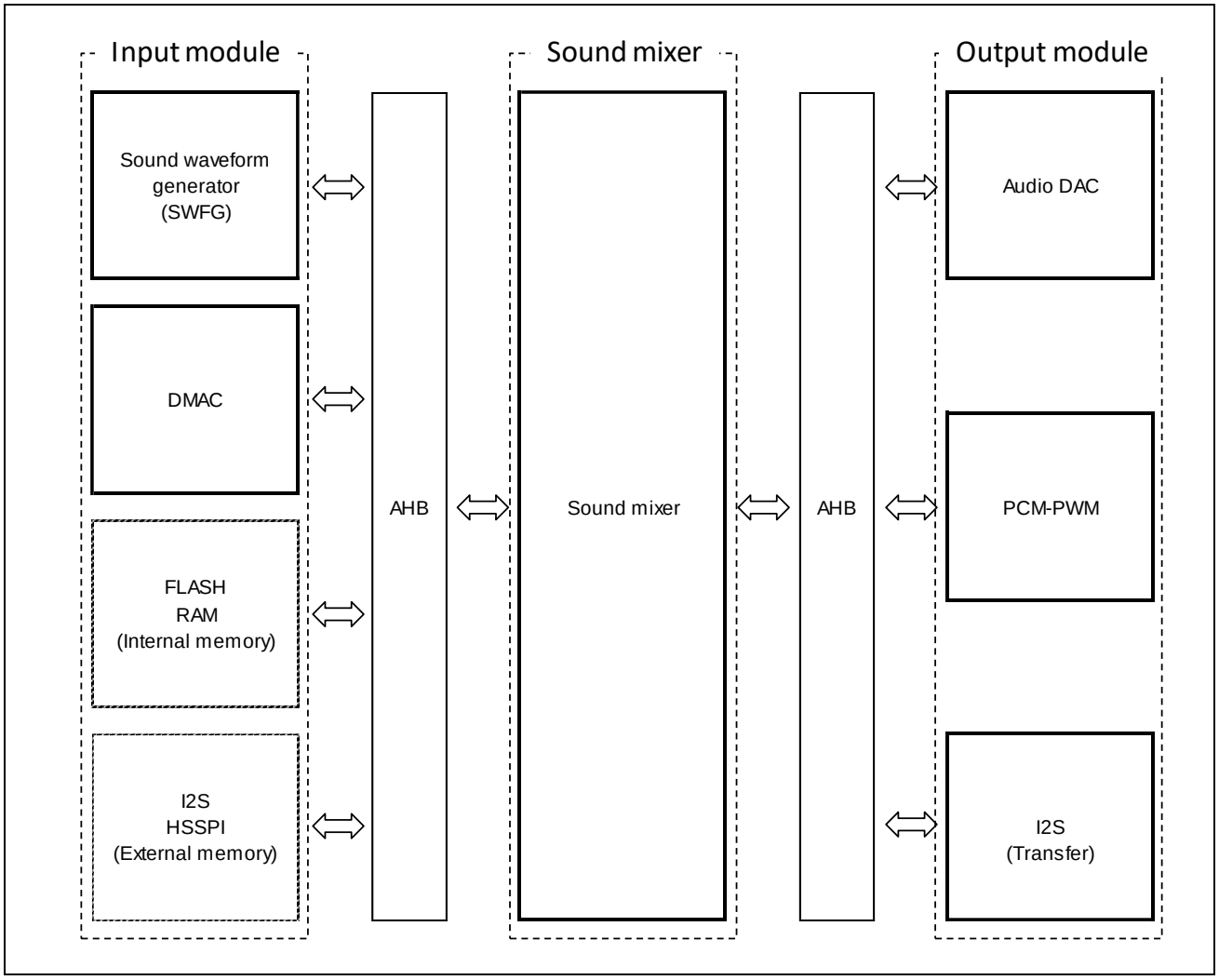
PCM-PWM converts PCM sound source to PWM (Pulse Width Modulation) waveform and outputs it. Through a kind of Integral circuit it can be converted to level of analog voltage.

I2S (Inter IC Sound) is a serial interface protocol. The sound system has its transfer facility as sound system function.

These 3 functions are exclusively used for the output module of the sound system.

2. Configuration and Block Diagram

Figure 32-1 Block Diagram of Sound System



Note:
I2S connected to the Sound Mixer supports only Transfer mode.

3. Operation

The operation of the sound system is a co-operation of the modules which are defined as in 1 Overview. The followings are descriptions of the procedures for each case.

The detailed explanation of the module operation and its note is described in each chapter. Before referring this chapter, see the chapter of each module in advance.

3.1. Start Operations

Take the following procedure of 3.1.1 to 3.1.3 to start up the operation of the sound system.

3.1.1. Configuration of Input Modules

SWFG Configuration
Start sound source generations configuring SWFG.

The generations will practically begin by means of each transfer request from the sound mixer.

DMAC Configuration
Start sound source transmissions configuring DMAC.

The transmissions will practically begin by means of each transfer request from SMIX.

3.1.2. Configuration of Sound Mixer Operations

Start a mixing operation configuring SMIX.

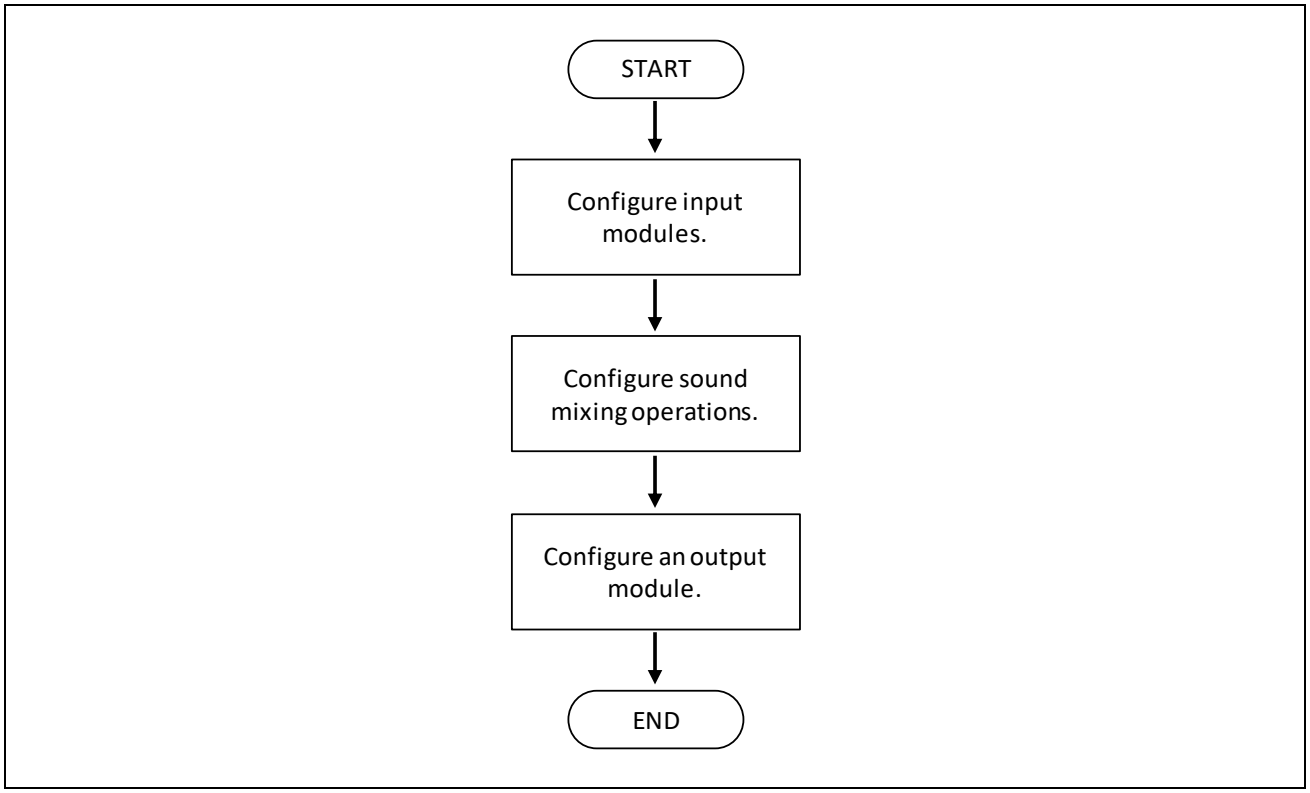
The mixing will practically begin by means of a transfer request from the output module.

3.1.3. Configuration of an Output Module

Start an output operation configuring one of the modules out of the audio DAC, PCM-PWM, or I2S.

The co-operation of the modules as a sound system will begins by means of a start operation of the output module.

Figure 32-2: Start Operations



Note:

- *CPU can transfer sound sources to the input channel PMIS0 to 4 of SMIX with data transfer requests of SMIX.*
- *Note some sound source which is transferred from SWFG is deleted by buffer initialization (clear control) during SMIX configuration. In order to avoid the phenomenon, WGSTART control to start sound source generation of SWFG can only be done after SMIX configuration.*

3.2. Addition of Sound Source Inputs

Take the following procedure to add additional sound sources to the operating sound system.

The existing mixed sound source can be continuously reproduced after the addition of new sound sources.

3.2.1. Configuration of Additional Input Sound Source

SWFG Configuration

Start additional sound source generations configuring channels of SWFG.

The generations will practically begin by means of each transfer request from SMIX.

DMA Configuration

Start additional sound source transmissions configuring a channel of DMA.

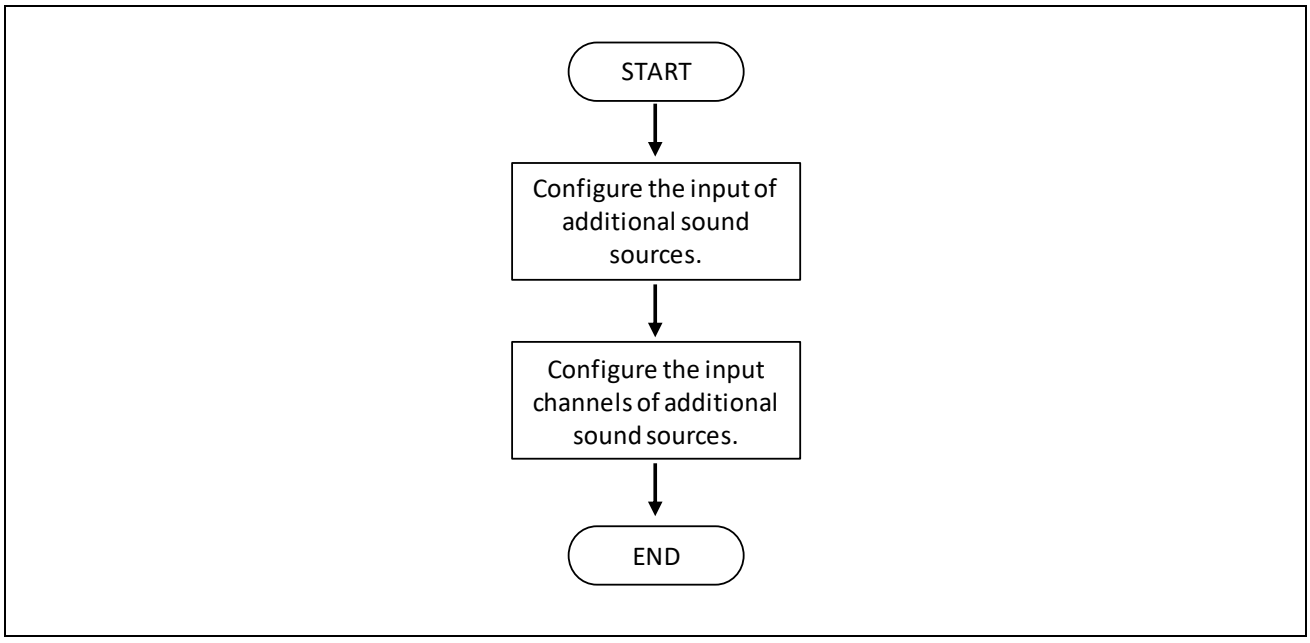
The transmissions will practically begin by means of each transfer request from SMIX.

3.2.2. Configuration of Additional Input Channel

Start an additional mixing operation configuring an input channel of SMIX.

The mixing will practically begin by means of a transfer request from the output module.

Figure 32-3: Addition of Sound Source Inputs



3.3. Stop Operations

Take the following procedure of 3.3.1 to 3.3.3 to stop the operation of the sound system.

3.3.1. Disabling the Output Module

Disable the output module. The procedure is described in the chapter of the audio-DAC, PCM-PWM and I2S in detailed.

Notes:

- *When the PCM-PWM or I2S is used as output module and the output module is changed from "enabled" to "disabled" (*1), the sound mixer should not be disabled and the output module should not be enabled during 16 sampling cycles. If this sequence is not followed, PCM data requested before disabling the output module may be transferred to the output module after enabling the output module. This may cause the DMA block error, Tx Block Size Error or FIFO overflow error.*
- *(*1) In PCM-PWM case, this means that the global enable bit (PCMPWMi_CONTROL,EN) is changed from 1 to 0.*
- *(*1) In I2S case, this means that the transmission enable bit (I2Sn_OPRREG:TXENB) or the I2S enable bit (I2Sn_OPRREG.START) is changed from 1 to 0.*

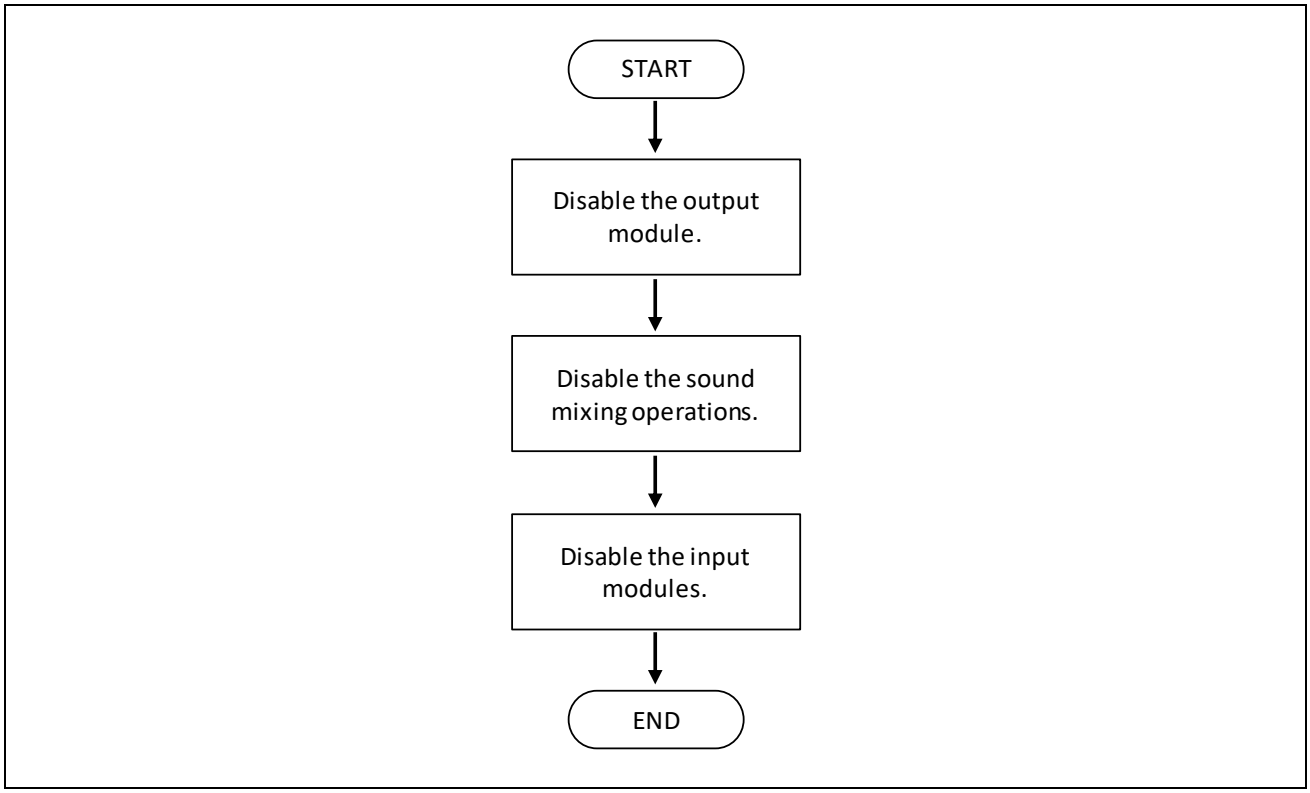
3.3.2. Disabling the Mixing

Disable the mixing operations of SMIX. The procedure is described in the chapter of SMIX in detailed.

3.3.3. Disabling the Input Modules

Disable the sound source generations and transmissions. The procedure is described in the chapter of DMA, interrupt and I2S in detailed.

Figure 32-4: Stop Operations



Note:

- *While the sound system operates continuously, a stop of a sound source generation of SWFG keeps all mixing operations of SMIX being hung up. Suspend a particular channel of SWFG only after disabling a corresponding input channel SMIX.*
- *DMA transfer needs a same operation as the case of SWFG. Before suspending DMA transfer, disable a corresponding input channel of SMIX.*

3.4. Operation of a Sound Source

An application doesn’t need the mixing operation but only needs an operation of an output module.

In the case the sound source should be transferred to SMIX as same as the case of a mixing of multi sound sources.

The sound system cannot directly transfer the sound source from an input module to an output module. Because every output module doesn’t support any transfer requests for CPU and DMA.

On the other hand, every output module supports a transfer request for SMIX, that is, it is only SMIX that can transfer the sound source to an output module.

Note:

- *SMIX only outputs the mixed sound source of 48 kHz sampling rate. Therefore the sound system cannot obtain the sound source with an original sampling rate even if the application doesn’t need mixing function.*
- *SMIX has the volume effect control of the mute, fade-in, and fade-out. They are quite effective for a sound source even if the mixing operation is not necessary.*

4. Configuration and limitation of each module

This section shows configurations and limitations about each module interface in the sound system.

4.1. Interface between SMIX and DAC

Output modules including DAC are connected to SMIX by a dedicated bus interface. On this interface, the sound source transfer between SMIX and DAC is triggered by a data request from DMA interface of DAC.

DAC has two initialization procedures (refer to the chapter of the DAC for the details). One is "DMA Interface Unused", and the other is "DMA Interface Used". DAC must be initialized by the procedure of "DMA Interface used", because the sound source is transferred from SMIX to DAC by using DMA interface.

Because it is impossible to transfer the sound source from DMAC to DAC directly, the sound source needs to be transferred to DAC via SMIX. Therefore, the initialization of the SMIX is necessary even if it is not necessary to do the mixing of the sound source. If the sound source is not mixing, only 1 PMIS input channel (any number of channels can be used) of SMIX should be enabled.

4.1.1. Configuration

The address of DAC must be set to Output destination address of SMIX (MACRO of MXOCTRL register). The actual setting value is the following.

- *MACRO of MXOCTRL register : 0x1*

Number of transfers to output destination of SMIX (DATAN of MXOCTRL register) must be same as FIFO empty threshold of DAC (FEST of DACTRL register).

4.1.2. Limitation

Because the sound sources are transferred surely from SMIX to DAC, the following registers must not be changed during the sound source translation.

Even if SMIX is disabled by MXCH register, the data transfer from SMIX to DAC is not stopped immediately. Therefore, the following register must not be changed after setting it at the initialization.

- *DACTRL*
- *MXOCTRL*

4.2. Interface between SMIX and PCM-PWM

Output modules including PCM-PWM are connected to SMIX by a dedicated bus interface. On this interface, the sound source transfer between SMIX and PCM-PWM is triggered by a data request from DMA interface of PCM-PWM.

PCM-PWM supports two type data transfers. One is "DMA mode", and the other is "CPU mode". PCM-PWM must be set to "DMA mode", because the sound source is transferred from SMIX to PCM-PWM by using DMA interface.

Because it is impossible to transfer the sound source from DMAC to PCM-PWM directly, the sound source needs to be transferred to PCM-PWM via SMIX. Therefore, the initialization of the SMIX is necessary even if it is not necessary to do the mixing of the sound source. If the sound source is not mixing, only 1 PMIS input channel (any number of channels can be used) of SMIX should be enabled.

4.2.1. Configuration

The address of PCM-PWM must be set to Output destination address of SMIX (MACRO of MXOCTRL register). The actual setting value is the following.

- *MACRO of MXOCTRL register : 0x2*

Number of transfers to output destination of SMIX (DATAN of MXOCTRL register) must be same as FIFO empty threshold of PCM-PWM (FEST of PCMPWMi_CONTROL register).

4.2.2. Limitation

Because the sound sources are transferred surely from SMIX to PCM-PWM, the following registers must not be changed during the sound source translation.

Even if SMIX is disabled by MXCH register, the data transfer from SMIX to PCM-PWM is not stopped immediately. Therefore, the following register must not be changed after setting it at the initialization.

- *MXOCTRL*

And, FEST of PCMPWMi_CONTROL must not be changed under the following condition.

- *EN of PCMPWMi_CONTROL is set to 1, and*
- *DMAEN of PCMPWMi_CONTROL is set to 1*

4.3. Interface between SMIX and I2S

Output modules including I2S are connected to SMIX by a dedicated bus interface. On this interface, the sound source transfer between SMIX and I2S is triggered by a data request from DMA interface of I2S.

I2S supports three type data transfers. These are "DMA", "Interrupt" and "Poling". I2S must be set to "DMA", because the sound source is transferred from SMIX to I2S by using DMA interface.

Because it is impossible to transfer the sound source from DMAC to I2S directly, the sound source needs to be transferred to I2S via SMIX. Therefore, the initialization of the SMIX is necessary even if it is not necessary to do the mixing of the sound source. If the sound source is not mixing, only 1 PMIS input channel (any number of channels can be used) of SMIX should be enabled.

4.3.1. Configuration

The address of I2S must be set to Output destination address of SMIX (MACRO of MXOCTRL register). The actual setting value is the following.

- *MACRO of MXOCTRL register : 0x3*

Number of transfers to output destination of SMIX (DATAN of MXOCTRL register) must be same as Tx FIFO Threshold of I2S (TFTH of I2Sn_INTCNT register).

4.3.2. Limitation

Because the sound sources are transferred surely from SMIX to I2S, the following registers must not be changed during the sound source translation.

Even if SMIX is disabled by MXCH register, the data transfer from SMIX to I2S is not stopped immediately. Therefore, the following register must not be changed after setting it at the initialization.

- *MXOCTRL*

And, TFTH of I2Sn_INTCNT must not be changed under the following condition.

- *START of I2Sn_OPRREG is set to 1, and*
- *TXENB of I2Sn_OPRREG is set to 1, and*
- *TXDMACT of I2Sn_DMAACT is set to 1*

4.4. Interface between SWFG and SMIX

SWFG is connected to SMIX by a dedicated bus interface. On this interface, the sound source transfer between SWFG and SMIX is triggered by a data request from SMIX.

4.4.1. Configuration

Sound sources from SWFG need to be input to MXWFGnDADR register of SMIX. Therefore, the address of MXWFGnDADR register should be set to WGCHADD1, WGCHADD2 and WGCHADD3. The actual setting values are the followings.

- *WGCHADD1 : 0x01040100*
- *WGCHADD2 : 0x010C0108*
- *WGCHADD3 : 0x00000110*

4.4.2. Limitation

Because the sound sources are transferred surely from SWFG to SMIX, the following registers must not be changed during the sound source translation.

Even if SWFG is disabled by WGCHEN register, the data transfer from SWFG to SMIX is not stopped immediately. Therefore, the following register must not be changed after setting it at the initialization.

- *WGCHADD1*
- *WGCHADD2*
- *WGCHADD3*

4.5. Interface between DMAC and SMIX

DMAC can be used to transfer the sound source to PMIS interface of SMIX.

4.5.1. Configuration

PMISn Request assert threshold setting of SMIX (FESTCHn of MXDRQCTRL register) must be same as the number of block count of DMAC.

4.5.2. Limitation

Because the sound sources are transferred surely from DMAC to SMIX, the following registers must not be changed during the sound source translation (from data request by SMIX to data transfer completion of DMAC).

- *PMISnMONO of MXCHMONO*

And, DMAENCHn of MXDREQCTRL must be 0 during the following procedures.

- *Clear PMIS Input Buffer of the channel (Write 1 to PMOSnBCLR of MXBUFFCLR)*
- *Enable the channel by MXCH register (change from 0 to 1)*
- *Change FESTCHn of MXDREQCTRL of the channel*

4.6. Interface between CPU and SMIX

CPU can be used to transfer the sound source to PMIS interface of SMIX. Data transfer request interrupt of SMIX can be used as trigger for the data transfer.

No special configuration or limitation to be considered.

5. Note

- *Every output module doesn't support any transfer requests for CPU and DMA. The transfer request only supports for the sound source transmission of SMIX. That is, CPU and DMA cannot directly transfer sound sources to the output modules without going through SMIX.*

CHAPTER 33: Graphic Subsystem

This chapter explains Graphics Subsystem.

1. Perface
2. Function
3. Application

1. Perface

1.1. About This Document

The 2D graphics sub-system is an IP developed by Cypress Semiconductor Corp.

This specification is an add-on document to

・HW Peripheral Manual for 2D Graphic Core v1.1 (S6J3300 Series)

・The chapter of LCD Bus Interface in this manual.

and describes the 2D graphics subsystem, which is built around the 2D and LCD Bus Interface modules and essentially adds a bus matrix to the core components for display control.

While the MCE and LCD Bus Interface cores in general are designed for re-use in other products, the SCE subsystem is a customized solution for integration into the Traveo MCU S6J3300 Series only.

For details on purpose of this document and its structure refer to the corresponding chapter in the MCE specification.

1.2. Reference Document

2D GRAPHICS CORE PROGRAMMERS REFERENCE MANUAL

Please contact Sales for further information.

2. Function

2.1. Feature Summary

・2D Graphics Core (for details see HW Peripheral Manual for 2D Graphic Core v1.1)

- 1 Display Controllers for 24-bit TTL panels
- Resolutions up to 40 MHz pixel clock (e.g. WVGA @ 60 Hz)

・LCD Bus Interface (for details see the chapter of LCD Bus Interface in this manual)

- Interfaces to LCD controller ICs with an Intel-8080 or Motorola-6800 compatible parallel interface
- Support for different display pixel formats (RGB888, RGB666, RGB565, RGB444, GRAY, B/W)

・AMBA Network Interconnect (Arm NIC301)

・Up to 80 MHz bus clock

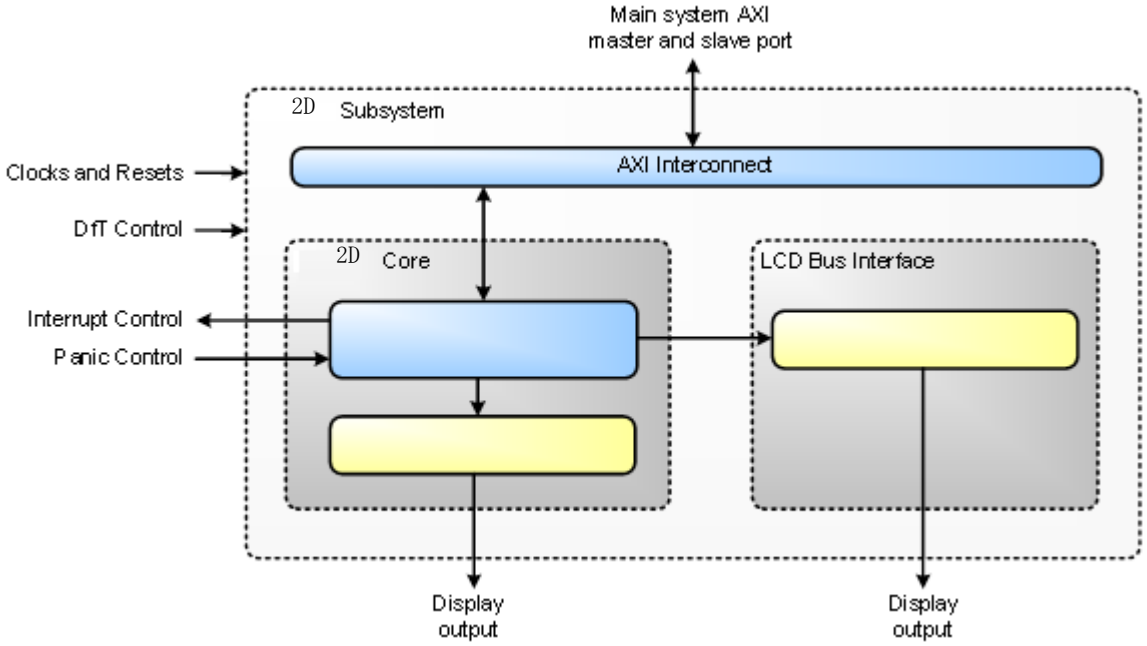
・Performance counters for AXI bus traffic

・AXI master and slave port for interconnection with main system bus

2.2. Block Diagrams

2.2.1. Top-Level

Figure 33-1: 2D Subsystem Block DiagramTop-Level



2.2.2. 2D Core

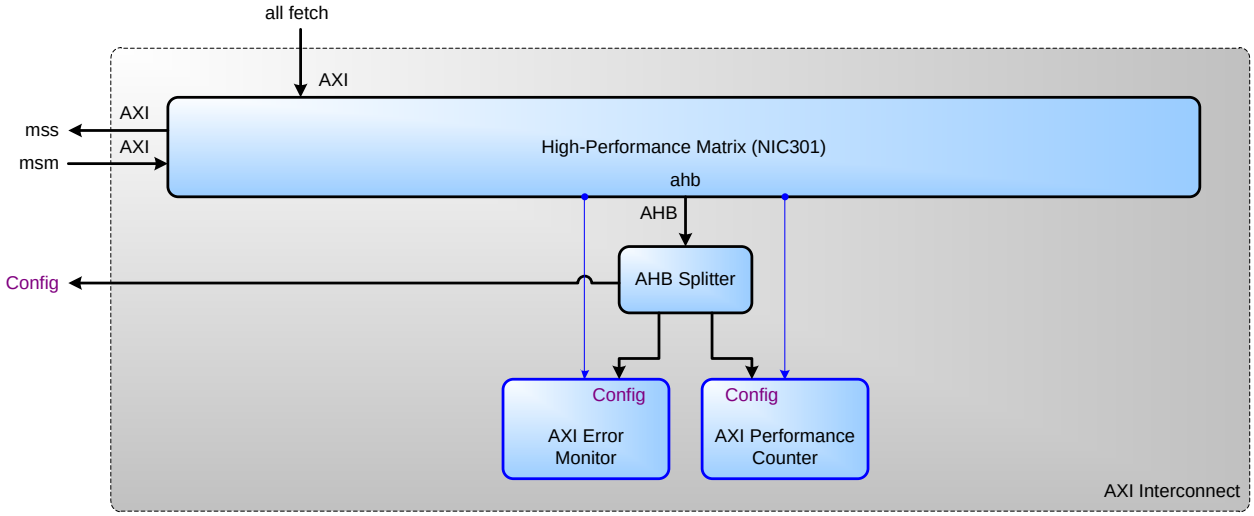
Refer to HW Peripheral Manual for 2D Graphic Core v1.1.

2.2.3. LCD Bus Interface

Refer to the chapter of LCD Bus Interface in this manual.

2.2.4. Interconnect

Figure 33-2: AXI Interconnect Block Diagram



Related topic: AXI Interconnect Function.

2.3. Functional Limitations

Table 33-1: Functional Limitations

Reference clock frequency: The peak reference clock must be lower or equal to 430 MHz to be able to generate small pixel clock frequencies for very small displays. The average reference clock must be at least 200 MHz to be able to generate pixel clock and shift display clock versus display data with sufficient accuracy.	max 430 MHz (peak) min 200 MHz (average)
Pixel clock frequency: The average frequency must be considered lower to cope with clock jitter and in case that spread spectrum modulation is used.	max 43 MHz (peak)
AXI bus clock frequency: The average frequency must be considered lower to cope with clock jitter and in case that spread spectrum modulation is used. The average frequency must be at least twice the used maximum average pixel clock frequency.	max 86 MHz (peak) min 2 * pixel clock (average)
For additional limitations, particularly on supported video modes, see HW Peripheral Manual for 2D Graphic Core v1.1.	

2.4. Clause Name

The AXI bus interconnect is conform to Arm AMBA® AXI and ACE Protocol Specification.It has 64 bits data and 32 bits address width.

Related topic: AXI Interconnect Block Diagram.

2.4.1. Network Interconnect (NIC)

The bus matrix is an Arm NIC301 (see Arm CoreLink® Network Interconnect NIC301 r2p3 – Technical Reference Manual).

It is configured for one AXI read layers (main system) plus read/write access to configuration registers via AHB.

2.4.2. Performance Analysis

Performance counters are provided to allow measurement of average bandwidth and latency during operation. The following features are supported:

- Manual and timer controlled measurement mode.
- Filter to measure bandwidth and utilization for each master by ID.
- Measurement counters:
  - GLOBAL_COUNTER for overall measurement time,
  - BUSY_COUNTER for number of data bus busy cycles,
  - DATA_COUNTER for number of data transfer cycles,
  - TRANSFER_COUNTER for number of transfers and
  - ADDRBUSY_COUNTER for number of address bus busy cycles,
  - LATENCY_COUNTER for average latency.
- Counter overflow detection.
- Outstanding Transfer Counters (OTC) which are used for latency measurement have to run immediately after reset, but can be disabled by software when there is no need for latency measurement.

2.4.3. Bus Monitor

A bus monitor observes all AXI transactions in order to detect error responses. It can store the ID of the first master that received an error response and can trigger an interrupt.

This feature is particularly useful in the context of this sub-system, because the master of the 2D graphics core do not implement a bus error detection, but simply ignore them.

3. Application

3.1. Map Tables

3.1.1. Interrupt Map

Related topic: Interrupt Setup.

The given '2D Core ID' is the bit number of the corresponding event as used in the Common Control configuration of the 2D Core module. The 'Interrupt ID' is the number by which the interrupt is integrated into the embodying main system.

Table 33-2: Interrupt Map

Interrupt Type	Interrupt ID	Interrupt Name	Interrupt Events	Core ID
IRQ	0	2D_ContentStream0		
			ExtDst0_ShdLoad	0
			ExtDst0_FrameComplete	1
			ExtDst0_SeqComplete	2
			FrameGen0_SecSync_On	22
			FrameGen0_SecSync_Off	23
			FrameGen0_Int1	10
IRQ	1	2D_SafetyStream0		
			ExtDst4_ShdLoad	3
			ExtDst4_FrameComplete	4
			ExtDst4_SeqComplete	5
			FrameGen0_PrimSync_On	20
			FrameGen0_PrimSync_Off	21
			FrameGen0_Int0	9
IRQ	2	2D_DisplayStream0		
			DisEngCfg_ShdLoad0	6
			DisEngCfg_FrameComplete0	7
			DisEngCfg_SeqComplete0	8
IRQ	3	2D_Signature0		
			Sig0_ShdLoad	13
			Sig0_Valid	14
			Sig0_Error	15
IRQ	4	2D_Display0_Sync0		
			FrameGen0_Int2	11
IRQ	5	2D_Display0_Sync1		
			FrameGen0_Int3	12
IRQ	6	2D_LCDBusIf_Control		
			SequencerSyncInterrupt	0
			SequencerErrorInterrupt	1
			TearingEffectInterrupt	4
IRQ	7	2D_LCDBusIf_InstrFifo		
			InstrFifoInterrupt	2
IRQ	8	2D_LCDBusIf_RxFifo		
			RxFifoInterrupt	3
NMI	0	2D_AICMonitor		
			AIC_ErrorResponse	

3.1.2. Address Map

Offset values in the following table are relative to address space of the embodying main system.

Table 33-3: Address Map

Offset [Hex in Bytes]	Size [Hex in Bytes]	Description	Register Map
0000_0000	4000_0000	Main system.	
4000_0000	1020_0000	Reserved; do not use.	
5020_0000	0000_0400	2D subsystem control.	SubSysCtrl
5020_0400	0000_4400	2D core (Display)	See HW Peripheral Manual for 2D Graphic Core v1.1
5020_4800	0000_0400	LCD Bus interface	See the chapter of LCD Bus Interface in this manual
5020_4C00	0000_F400	Reserved; do not use.	
5021_4000	0000_0400	AXI Interconnect – Performance Measurement Unit.	aic_performance
5021_4400	0000_0400	AXI Interconnect – Error Monitor.	aic_monitor
5021_4800	001E_B800	Reserved; do not use.	
5040_0000	0010_0000	AXI Interconnect – Network Interconnect (NIC301).	NIC
5050_0000	0FB0_0000	Reserved; do not use.	
6000_0000	A000_0000	Main system.	

3.1.3. Key Map

Inside this IP all address blocks, that can be protected, use one of the following key values:

Table 33-4: Key Map

Name	Value	Function
lock key	0x5651F763	Enable all access protection.
unlock key	0x691DB936	Disable all access protection.
privilege key	0xAEE95CDC	Enable non-privileged access protection.
unprivilege key	0xB5E2466E	Disable non-privileged access protection.
freeze key	0xFBE8B1E6	Freeze current protection status (cannot be changed any longer).

3.2. Common Setup

3.2.1. AXI/AHB Clock Setup

The configuration clock frequency used in 2D can be setup using register field ConfigClockSelect in the sub system control module. It specifies the divider used to generate the configuration clock from the AXI clock. All possible values of the divider are allowed.

The configuration clock can be changed at any time (e.g. also when transfers are in progress), but note that configuration access bandwidth and latency are greatly influenced by this setting.

3.2.2. Display Clock Setup

There is one display clock management unit available in the Subsystem Control module of 2D.It contains clock generation utility to generate the display clock for the 2D core from the input reference clock by using a fixed point divider. In addition the generated display clock can be shifted against the output display data to allow creation of sufficient margin for setup and hold times at the connected external device.

3.2.2.1. Display Clock Generation and Reset Control

The display clock generation can create the display clock from the input reference clock by application of a fixed point divider. To determine the best possible display clock frequency divide the reference clock frequency by a multiple of 2 of the desired display clock frequency (if 2D core DisEngCfg ClockCtrl setting is set to DIV2) or directly by the desired display clock frequency (in case the 2D core DisEngCfg ClockCtrl setting is set to DIV1), round this value to a fixed point with 8 decimal places and program the result to register field dsp0_ClockDivider. Then set register field dsp0_ClockEnable to 1 to start generation of the display clock.

The field dsp0_ClockDivider can be updated at any point in time (eg also when dsp0_ClockEnable is set to 1 and display clock is output). The display clock generation will then update the generated clocks frequency in such a manner that continuity of the generated display clock is guaranteed (eg no glitches are produced).

In addition to enabling the dsp0_ClockEnable setting the dsp0_SoftwareReset should be set to 0 to release the reset of the display logic and make it functional. Write dsp0_SoftwareReset to 1 only when the display engine has been completely shut down. Other use of the dsp0_SoftwareReset is applicable to debug purposes only.

Note that use of the generated display clock for display clock output is only allowed if 2D core DisEngCfg ClockCtrl setting is set to DIV2.

3.2.2.2. Display Clock Shift

To allow to create sufficient margin for setup and hold times at the external device the output display clock can be shifted. The dsp0_ClockOffset setting can shift the display clock up to (not including) 180 degrees in reference clock period steps. To achieve shifts larger or equal to 180 degrees the dsp0_ClockInvert setting can be used in addition to add 180 degrees to the shift achieved with dsp0_ClockOffset.

It is recommended to verify the achieved margins in setup and hold times by using eg an oscilloscope. Note that the generated display clock periods can jitter from cycle to cycle by the reference clock period due to the way the fixed point clock division works.

The dsp0_ClockOffset and dsp0_ClockInvert settings must not be changed when dsp0_ClockEnable is set to 1, otherwise corruption of displayed content is possible.

3.2.3. Reset Setup

All reset signals must be controlled by the embodying system. Each clock domain has a correlated HW reset. Resets must not be released before the corresponding clock is operating stable.

3.2.4. Configuration

Related topic: SW Interface.

3.2.4.1. IP Identifier

To get information about the 2D IP derivative and design revision:

Read out IPIdentifier register from Sub System Control unit.

Note, that the content of this register can be changed by SW.

3.2.4.2. Register Protection

Refer toHW Peripheral Manual for 2D Graphic Core v1.1.

The SubSysCtrl unit implements the same protection features as the graphics core. In addition, it contains the same protection feature for the nic configuration space.

See also the chapter of LCD Bus Interface in this manual.

3.2.5. Interrupts

Related topic: Interrupt Map.

Interrupt control functionality (enable, status, clear) is implemented at the following locations:

Interrupts from 2D Graphics Core:
Common Control Unit of 2D Core (refer to HW Peripheral Manual for 2D Graphic Core v1.1.
Implements Enable, Preset, Clear and Status.

Interrupts from LCD Bus Interface:
LCDBusIfInterruptEnable/Preset/Clear/Status

Interrupts from AIC Monitor:
MonitorInterruptEnable/Clear and ErrorType (= status).

3.2.6. Interconnect

3.2.6.1. Network Interconnect

In general refer to Arm CoreLink Network Interconnect NIC301 r2p3 – Technical Reference Manual and NIC registers.

The individual masters can be assigned different priorities that are used for arbitration. Values in the following table correspond to the reset priorities and can be changed. Connections marked with ‚X‘ do not exist.

Table 33-5: NIC301 Routing Priorities

Slave Interface	Master mss (M0)	Master ahb_ctrl (M1)	Master gpv (M2)
msm (S0)	X	1	1
fetchdecode0 (S1)	1	X	X
fetchlayer0 (S2)	1	X	X

Note, that this table is created from the perspective of the Network Interconnect and so a master is connected to a slave interface and slaves like the main system slave are connected to masters of the NIC.

A priority can be in the range from 0 to 15 with 0 being the lowest possible priority and 15 being the highest.

To change the priority of a master write the desired priority to the register field <slaveinterface>.ar_qos or <slaveinterface>.aw_qos in the NIC configuration register interface.

3.2.6.2. Performance Analysis

To use the performance analysis options of the interconnect do the following:

Set Mode to TIMER and OTCDisable to 0. Determine a period during which the measurement will be executed. Calculate the number of AXI clock cycles in this period. Choose

abs(max(0, ld(period_in_cycles) - 28))

as the Divider. Program Load to

period_in_cycles / (2 ^ Divider).

Set the MU<MasterID>_Switch field for all masters that shall be observed. The following is exemplary for master 0 (MU0_Switch).

Activate measurement by setting Enable to ‘1’ and wait until Enable is ‘0’ again.

The consumed bandwidth can be calculated by multiplying the MU0_Data with the AXI frequency and the width of the bus in bytes (8 for all masters except CmdseqRead, CmdseqWrite and DrawengRead for which it is 4).

Data bus utilization can be calculated by dividing MU0_Data by MU0_Busy.

Address bus utilization can be calculated by dividing MU0_Transfer by MU0_Addrbusy.

Average latency can be calculated by dividing MU0_Latency by MU0_Transfer. Note that this latency is measured from request to the last beat of the burst. Also note that for very short measurement periods in the order of magnitude of the burst lengths used this value will become unreliable.

3.2.6.3. Bus Monitor

The bus monitor can observe the transfers on the network interconnect and remember the first error response it detects.

To deactivate monitoring for certain masters set their corresponding bit in register MonitorDisable to ‘1’.

To check whether an error occurred read ErrorType and check whether it is set to NONE (no error occurred) or SLVERR or DECERR (error occurred). Alternatively the interrupt event AIC_ErrorResponse can be used.

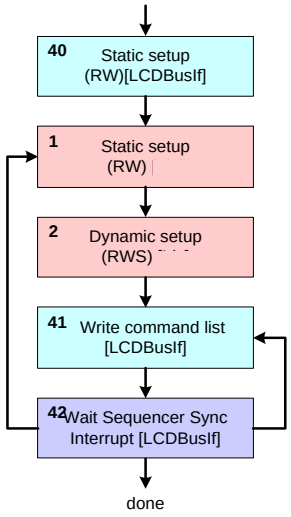
To get the master which received the error read ErrorID. For masters that use an ID you can read the ID used from ErrorSubID.

To clear the remembered error, so that the monitor may detect another error, write ‘1’ to MonitorClear.

3.3. 2D with LCD Bus Interface

3.3.1. Display Static Single-Thread

For typical LCD applications where the content is not changed with every frame, a simple control flow with disabled shadow registers can be used. In that 2D is programmed to compose the scene to be displayed and the LCDBusIF is fed with a command list to control the display content update operation:



[40] See chapter “Display Setup” in the chapter of LCD Bus Interface in this manual.

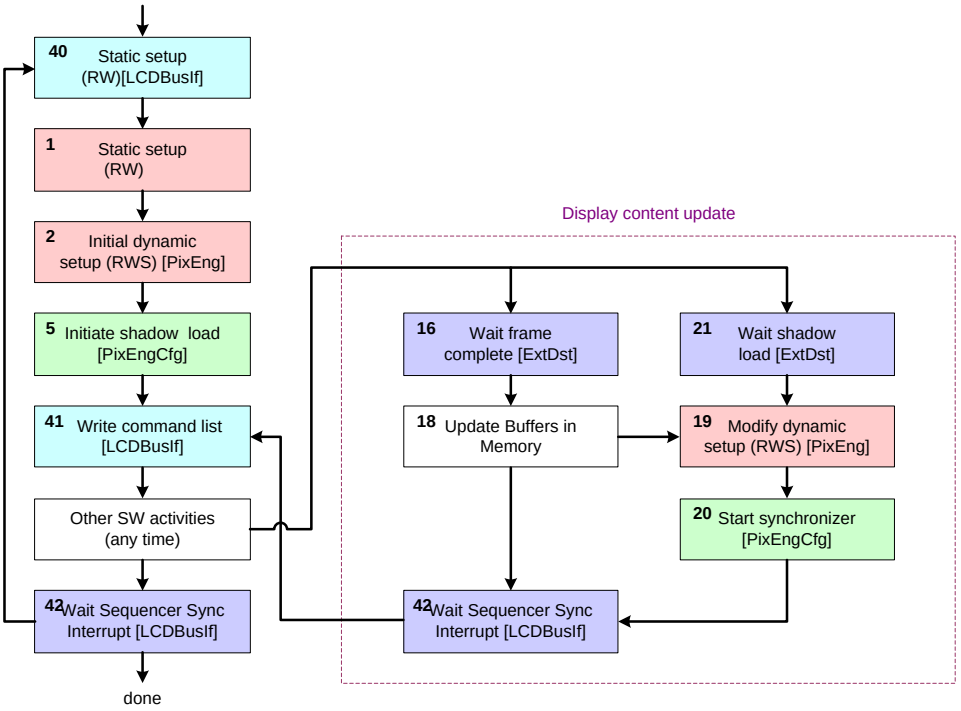
[1,2] See HW Peripheral Manual for 2D Graphic Core v1.1 chapter “Display Static Single-Thread”.

[41] See chapter “Control Flow” in the chapter of LCD Bus Interface in this manual.

[42] In order to detect that display content has been transferred, Software can either poll LCDBusIf register field OperationState to become IDLE or wait for interrupt SequencerSyncInterrupt.

3.3.2. Display Dynamic Single-Thread

When there is a need to modify parts of the configuration or content of display buffers in memory during display operation without tearing artefacts, the following control flow using shadow registers must be used:



For details regarding the definitions of static and dynamic setup, refert to HW Peripheral Manual for 2D Graphic Core v1.1 chapter “Display Dynamic Single-Thread”.

[40] See chapter “Display Setup” in the chapter of LCD Bus Interface in this manual.

[1,2] See HW Peripheral Manual for 2D Graphic Core v1.1 chapter “Display Dynamic Single-Thread”.

[5] See HW Peripheral Manual for 2D Graphic Core v1.1 chapter “Display Dynamic Single-Thread” but do not initiate shadow load for display stream

[41] See chapter “Control Flow” in the chapter of LCD Bus Interface in this manual.
This step should also clear the SequencerSyncInterrupt, ExtDst4_FrameComplete and ExtDst4_ShdLoad interrupts before starting.

[16] Wait for ExtDst4_FrameComplete interrupt.

[18] Update buffer content in memory.
If updating the buffer takes too long, then frame update rate at display will drop.
If frame update without reduction of rate is successful, interrupt TearingEffectInterrupt will not have happened before step [41] is complete.

[21] Wait for ExtDst4_ShdLoad interrupt.

[19,20] See HW Peripheral Manual for 2D Graphic Core v1.1 chapter “Display Dynamic Single-Thread”, but do not wait for the shadow load interrupt here.

[42] In order to detect that display content has been transferred and is complete, Software can either poll LCDBusIf register field OperationState to become IDLE or wait for interrupt SequencerSyncInterrupt.

3.3.3. Display Dynamic Multi-Thread

The Display Dynamic Multi-Thread control flow is not supported for operation of 2D with LCD Bus Interface.

3.4. 2D Core

Refer to HW Peripheral Manual for 2D Graphic Core v1.1.

3.5. LCD Bus Interface

Refer to the chapter of LCD Bus Interface in this manual.

3.6. SW Interface

3.6.1. General

Refer to corresponding section in HW Peripheral Manual for 2D Graphic Core v1.1.

3.6.2. SubSysCtrl

Addr. Offset	Register Name	Prot.	Register Description (Address Block 0)
0x0000	LockUnlock	key	Register to change the protection status of this address block.
0x0004	LockStatus	status	Protection status of this address block.
0x0008	IPIdentifier	yes	IP Identifier for this 2D derivate, needs to be unlocked.
0x000C	ConfigClockControl	yes	Controls config clock generation
0x0010	LCDBusIfInterruptEnable	yes	LCD Bus Interface interrupt enable register
0x0014	LCDBusIfInterruptPreset	yes	LCD Bus Interface interrupt preset register
0x0018	LCDBusIfInterruptClear	yes	LCD Bus Interface interrupt clear register
0x001C	LCDBusIfInterruptStatus	yes	LCD Bus Interface interrupt status register
Addr. Offset	Register Name	Prot.	Register Description (Address Block 1)
0x0040	dsp_LockUnlock	key	Register to change the protection status of this address block.
0x0044	dsp_LockStatus	status	Protection status of this address block.
0x0048	dsp0_ClockDivider	yes	Clock divider register
0x004C	dsp0_DomainControl	yes	Domain control register
0x0050	dsp0_ClockShift	yes	Clock shift register
Addr. Offset	Register Name	Prot.	Register Description (Address Block 2)
0x0300	hpm_LockUnlock	key	Register to change the protection status of the address block 3 (Offset is in the range of 0x0300 to 0x03FF) and HPM (Address is in the range of 0x50400000 to 0x504FFFFF).
0x0304	hpm_LockStatus	status	Protection status of the address block 3 and HPM.

3.6.2.1. LockUnlock

Register to change the protection status of this address block.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	LockUnlock[31:24]							
ACCESS_TYPE	W	W	W	W	W	W	W	W
PROT_TYPE	-							
INITIAL_VALUE	-	-	-	-	-	-	-	-

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	LockUnlock[23:16]							
ACCESS_TYPE	W	W	W	W	W	W	W	W
PROT_TYPE	-							
INITIAL_VALUE	-	-	-	-	-	-	-	-

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	LockUnlock[15:8]							
ACCESS_TYPE	W	W	W	W	W	W	W	W
PROT_TYPE	-							
INITIAL_VALUE	-	-	-	-	-	-	-	-

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	LockUnlock[7:0]							
ACCESS_TYPE	W	W	W	W	W	W	W	W
PROT_TYPE	-							
INITIAL_VALUE	-	-	-	-	-	-	-	-

[bit31:0] LockUnlock: The protection status is changed by writing one of the following key values to this field

bit	Description
0x5651F763	Decrements the unlock counter. When the counter value is null, lock protection is active. Reset counter value is 1.
0x691DB936	Increments the unlock counter. Max allowed value is 15.
0xAEE95CDC	Enables privilege protection. Disabled after reset.
0xB5E2466E	Disables privilege protection.
0xFBE8B1E6	Freezes current protection status. Writing keys to this register has no more effect until reset.
Other than above	Setting prohibited

When lock protection is active, no write but only read access is possible to all registers of this address block. When privilege protection is active, only privileged read and write access is possible. Both protections can be active at the same time. Reading this register returns the current unlock counter value, but results in an error response (for HW test purposes only).

3.6.2.2. LockStatus

Protection status of this address block.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	-	-	-	-	-	-	-	-
PROT_TYPE	-							
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	-	-	-	-	-	-	-	-
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	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	FreezeStatus
ACCESS_TYPE	-	-	-	-	-	-	-	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	Reserved	Reserved	Reserved	PrivilegeStatus	Reserved	Reserved	Reserved	LockStatus
ACCESS_TYPE	-	-	-	R	-	-	-	R
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:9] Reserved: Reserved bits

[bit8] FreezeStatus: Current freeze status

bit	Description
0	protection status can be changed
1	cannot be changed.

[bit7:5] Reserved: Reserved bits

[bit4] PrivilegeStatus: Current status of privilege protection

bit	Description
0	inactive
1	active.

[bit3:1] Reserved: Reserved bits

[bit0] LockStatus: Current status of lock protection

bit	Description
0	inactive (unlock counter > 0)
1	active (unlock counter == 0)

3.6.2.3. IPIdentifier

IP Identifier for this 2D derivate, needs to be unlocked.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	IPFamily[3:0]				IPConfiguration[3:0]			
ACCESS_TYPE	RW	RW	RW	RW	RW	RW	RW	RW
PROT_TYPE	-							
INITIAL_VALUE	0	0	1	0	0	0	1	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	IPApplication[3:0]				IPFeatureSet[3:0]			
ACCESS_TYPE	RW	RW	RW	RW	RW	RW	RW	RW
PROT_TYPE	-							
INITIAL_VALUE	0	1	0	1	0	0	0	1

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	IPEvolution[3:0]				DesignMaturityLevel[3:0]			
ACCESS_TYPE	RW	RW	RW	RW	RW	RW	RW	RW
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	1	0	1	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	DesignDeliveryID[3:0]				Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	RW	RW	RW	RW	RW	RW	RW	RW
PROT_TYPE	-							
INITIAL_VALUE	0	0	1	1	0	0	0	0

[bit31:28] IPFamily: IP family

bit	Description
0000	Generation 2010: 2D building block generation 2010
0001	Generation 2012: 2D building block generation 2012
0010	Generation 2013: 2D building block generation 2013
Other than above	Setting prohibited

[bit27:24] IPConfiguration: Ip configuration

bit	Description
0001	Module: 2D Graphics Core
0010	System: Graphics Subsystem with 2D
0011	System3D: Graphics Subsystem with 2D Graphics Core and Brontes 3D Engine
Other than above	Setting prohibited

[bit23:20] IPApplication: IP application of 2D Graphics Core

bit	Description
0001	B: Blit Engine only.
0010	D: Blit Engine and Display Controller.
0011	V: Display Controller only (with direct capture).
0100	G: Blit Engine, Display Controller (with direct capture), Capture Controller (buffered capture) and Drawing Engine.
0101	C: Display Controller only.
0110	H: Blit Engine, Display Controller and Drawing Engine.
Other than above	Setting prohibited

[bit19:16] IPFeatureSet: IP feature set (complexity of implemented features, e.g. availability of re-sampling filter etc)

bit	Description
0001	ECO
0010	LIGHT
0011	STANDARD
0100	PLUS
0101	EXTENSIVE
Other than above	Setting prohibited

[bit15:12] IPEvolution: IP evolution (increased for functional spec changes only when feature set keeps the same)

[bit11:8] DesignMaturityLevel: Design maturity level (corresponds to status at time of IP delivery, Fujitsu internal development stages)

bit	Description
0001	PreFS: Pre feasibility study
0010	FS: Feasibility study
0011	R0: Functionality complete
0100	R1: Verification complete
Other than above	Setting prohibited

[bit7:4] DesignDeliveryID: Design delivery ID (increased with each official delivery when maturity keeps the same)

[bit3:0] Reserved: Reserved bits

3.6.2.4. ConfigClockControl

Controls config clock generation

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	reserved	reserved	reserved	reserved	reserved	reserved	reserved	reserved
ACCESS_TYPE	-	-	-	-	-	-	-	-
PROT_TYPE	-							
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	-	-	-	-	-	-	-	-
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	reserved	reserved	reserved	reserved	reserved	reserved	reserved
ACCESS_TYPE	-	-	-	-	-	-	-	-
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	reserved	reserved	reserved	reserved	ConfigClockSelect[2:0]		
ACCESS_TYPE	-	-	-	-	-	RW	RW	RW
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	1

[bit31:3] Reserved: Reserved bits

[bit2:0] ConfigClockSelect: Divider of AXI clock to generate the config clock, given minus one. Please check the specification for the set of allowed values.

3.6.2.5. LCDBusIfInterruptEnable

LCD Bus Interface interrupt enable register

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	-	-	-	-	-	-	-	-
PROT_TYPE	-							
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	-	-	-	-	-	-	-	-
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	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	-	-	-	-	-	-	-	-
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	Reserved	Reserved	TearingEffectInterruptEnable	RxFifoInterruptEnable	InstrFifoInterruptEnable	SequencerSyncInterruptEnable	SequencerErrorInterruptEnable
ACCESS_TYPE	-	-	-	RW	RW	RW	RW	RW
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:5] Reserved: Reserved bits

[bit4] TearingEffectInterruptEnable: If set to '1' the TearingEffect interupt is enabled, if set to '0' the interrupt is disabled and only the InterruptStatus register can be used to see the status.

bits	Description
0	enabled
1	disabled

[bit3] RxFifoInterruptEnable: If set to '1' the RxFifo interupt is enabled, if set to '0' the interrupt is disabled and only the InterruptStatus register can be used to see the status.

bits	Description
0	enabled
1	disabled

[bit2] InstrFifoInterruptEnable: If set to '1' the TxFifo interupt is enabled, if set to '0' the interrupt is disabled and only the InterruptStatus register can be used to see the status.

bits	Description
0	enabled
1	disabled

[bit1] SequencerSyncInterruptEnable: If set to '1' the SequencerSync interupt is enabled, if set to '0' the interrupt is disabled and only the InterruptStatus register can be used to see the status.

bits	Description
0	enabled
1	disabled

[bit0] SequencerErrorInterruptEnable: If set to '1' the SequencerError interupt is enabled, if set to '0' the interrupt is disabled and only the InterruptStatus register can be used to see the status.

bits	Description
0	enabled
1	disabled

3.6.2.6. LCDBusIfInterruptPreset

LCD Bus Interface interrupt preset register

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	reserved	reserved	reserved	reserved	reserved	reserved	reserved	reserved
ACCESS_TYPE								
PROT_TYPE								
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								
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	reserved	reserved	reserved	reserved	reserved	reserved	reserved
ACCESS_TYPE								
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	reserved	reserved	TearingEffectInterruptPreset	RxFifoInterruptPreset	InstrFifoInterruptPreset	SequencerSyncInterruptPreset	SequencerErrorInterruptPreset
ACCESS_TYPE				W1P	W1P	W1P	W1P	W1P
PROT_TYPE								
INITIAL_VALUE	0	0	0	-	-	-	-	-

[bit31:5] Reserved: Reserved bits

[bit4] TearingEffectInterruptPreset: Write '1' to this field to set the TearingEffect interrupt.

[bit3] RxFifoInterruptPreset: Write '1' to this field to set the RxFifo interrupt.

[bit2] InstrFifoInterruptPreset: Write '1' to this field to set the TxFifo interrupt.

[bit1] SequencerSyncInterruptPreset: Write '1' to this field to set the SequencerSync interrupt.

[bit0] SequencerErrorInterruptPreset: Write '1' to this field to set the SequencerError interrupt.

3.6.2.7. LCDBusIfInterruptClear

LCD Bus Interface interrupt clear register

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	reserved	reserved	reserved	reserved	reserved	reserved	reserved	reserved
ACCESS_TYPE	-	-	-	-	-	-	-	-
PROT_TYPE	-							
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	-	-	-	-	-	-	-	-
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	reserved	reserved	reserved	reserved	reserved	reserved	reserved
ACCESS_TYPE	-	-	-	-	-	-	-	-
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	reserved	reserved	TearingEffectInterruptClear	RxFifoInterruptClear	InstrFifoInterruptClear	SequencerSyncInterruptClear	SequencerErrorInterruptClear
ACCESS_TYPE	-	-	-	W1P	W1P	W1P	W1P	W1P
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	-	-	-	-	-

[bit31:5] Reserved: Reserved bits

[bit4] TearingEffectInterruptClear: Write '1' to this field to clear the TearingEffect interrupt.

[bit3] RxFifoInterruptClear: Write '1' to this field to clear the RxFifo interrupt.

[bit2] InstrFifoInterruptClear: Write '1' to this field to clear the TxFifo interrupt.

[bit1] SequencerSyncInterruptClear: Write '1' to this field to clear the SequencerSync interrupt.

[bit0] SequencerErrorInterruptClear: Write '1' to this field to clear the SequencerError interrupt.

3.6.2.8. LCDBusIfInterruptStatus

LCD Bus Interface interrupt status register

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	reserved	reserved	reserved	reserved	reserved	reserved	reserved	reserved
ACCESS_TYPE	-	-	-	-	-	-	-	-
PROT_TYPE	-							
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	-	-	-	-	-	-	-	-
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	reserved	reserved	reserved	reserved	reserved	reserved	reserved
ACCESS_TYPE	-	-	-	-	-	-	-	-
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	reserved	reserved	TearingEffectInterruptStatus	RxFifoInterruptStatus	InstrFifoInterruptStatus	SequencerSyncInterruptStatus	SequencerErrorInterruptStatus
ACCESS_TYPE	-	-	-	R	R	R	R	R
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:5] Reserved: Reserved bits

[bit4] TearingEffectInterruptStatus: Status of the TearingEffect interrupt.

[bit3] RxFifoInterruptStatus: Status of the RxFifo interrupt.

[bit2] InstrFifoInterruptStatus: Status of the TxFifo interrupt.

[bit1] SequencerSyncInterruptStatus: Status of the SequencerSync interrupt.

[bit0] SequencerErrorInterruptStatus: Status of the SequencerError interrupt.

3.6.2.9. dsp_LockUnlock

Register to change the protection status of this address block.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	dsp_LockUnlock [31:24]							
ACCESS_TYPE	W	W	W	W	W	W	W	W
PROT_TYPE	-							
INITIAL_VALUE	-	-	-	-	-	-	-	-

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	dsp_LockUnlock [23:16]							
ACCESS_TYPE	W	W	W	W	W	W	W	W
PROT_TYPE	-							
INITIAL_VALUE	-	-	-	-	-	-	-	-

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	dsp_LockUnlock [15:8]							
ACCESS_TYPE	W	W	W	W	W	W	W	W
PROT_TYPE	-							
INITIAL_VALUE	-	-	-	-	-	-	-	-

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	dsp_LockUnlock [7:0]							
ACCESS_TYPE	W	W	W	W	W	W	W	W
PROT_TYPE	-							
INITIAL_VALUE	-	-	-	-	-	-	-	-

[bit31:0] dsp_LockUnlock: The protection status is changed by writing one of the following key values to this field

bit	Description
0x5651F763	Decrements the unlock counter. When the counter value is null, lock protection is active. Reset counter value is 1.
0x691DB936	Increments the unlock counter. Max allowed value is 15.
0xAEE95CDC	Enables privilege protection. Disabled after reset.
0xB5E2466E	Disables privilege protection.
0xFBE8B1E6	Freezes current protection status. Writing keys to this register has no more effect until reset.
Other than above	Setting prohibited

When lock protection is active, no write but only read access is possible to all registers of this address block. When privilege protection is active, only privileged read and write access is possible. Both protections can be active at the same time. Reading this register returns the current unlock counter value, but results in an error response (for HW test purposes only).

3.6.2.10. dsp_LockStatus

Protection status of this address block.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	-	-	-	-	-	-	-	-
PROT_TYPE	-							
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	-	-	-	-	-	-	-	-
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	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	dsp_FreezeStatus
ACCESS_TYPE	-	-	-	-	-	-	-	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	Reserved	Reserved	Reserved	dsp_PrivilegeStatus	Reserved	Reserved	Reserved	dsp_LockStatus
ACCESS_TYPE	-	-	-	R	-	-	-	R
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:9] Reserved: Reserved bits

[bit8] dsp_FreezeStatus: Current freeze status

bit	Description
0	protection status can be changed
1	cannot be changed.

[bit7:5] Reserved: Reserved bits

[bit4] dsp_PrivilegeStatus: Current status of privilege protection

bit	Description
0	Inactive
1	active.

[bit3:1] Reserved: Reserved bits

[bit0] dsp_LockStatus: Current status of lock protection

bit	Description
0	inactive (unlock counter > 0)
1	active (unlock counter == 0)

3.6.2.11. dsp0_ClockDivider

Clock divider register

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	-	-	-	-	-	-	-	-
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	dsp0_ClockDivider [23:16]							
ACCESS_TYPE	RW	RW	RW	RW	RW	RW	RW	RW
PROT_TYPE	-							
INITIAL_VALUE	0	0	1	0	0	0	0	0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	dsp0_ClockDivider [15:8]							
ACCESS_TYPE	RW	RW	RW	RW	RW	RW	RW	RW
PROT_TYPE	-							
INITIAL_VALUE	1	1	1	1	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	-	-	-	-	-	-	-	-
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:24] Reserved: Reserved bits

[bit23:8] dsp0_ClockDivider: Fixed point clock divider setting with 8 decimal places. Must be at least 2.0. Do not program in a way that a display clock in excess of the maximum frequency that the design can handle is generated, otherwise behaviour of the silicon can be undefined. Can be written while clock is already enabled, generated clock will adapt to the new setting then.

In case of TTL:

The fix point clock divider means unless the clock divider is an integer, there will be a mix of lower and higher clock frequencies. Maximal peak frequency is calculated as follows.

If floating part of the DIVIDER is >= 0.5: fDSP_CLK_max = fREF_CLK / (2 x int(dsp0/1_ClockDivider) + 1)

If floating part of the DIVIDER is < 0.5: fDSP_CLK_max = fREF_CLK / (2 x int(dsp0/1_ClockDivider))

(With DIVIDER = fREF_CLK / fDSP_CLK)

In case of RSDS:

Also unless the clock divider implemented is an integer, the duty cycle of the display clock is not 50%. The display clock consists of higher frequencies of fDSP_CLK_max = fREF / int(DIVIDER) and lower frequencies of fDSP_CLK_max = fREF / int(DIVIDER+1)

This needs to be considered for the RSDS case where both edges of the display clock are used.

[bit7:0] Reserved: Reserved bits

3.6.2.12. dsp0_DomainControl

Domain control register

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	-	-	-	-	-	-	-	-
PROT_TYPE	-							
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	dsp0_SoftwareReset
ACCESS_TYPE	-	-	-	-	-	-	-	RW
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	1

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	-	-	-	-	-	-	-	-
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	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	dsp0_Clock Enable
ACCESS_TYPE	-	-	-	-	-	-	-	RW
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:17] Reserved: Reserved bits

[bit16] dsp0_SoftwareReset: This field will only have an effect when field ClockEnable is set to 1.

bit	Description
0	make the display clock domain operational
1	keep the display clock domain in reset state

[bit15:1] Reserved: Reserved bits

[bit0] dsp0_ClockEnable: Make sure that ClockDivider is set to a valid value before enabling this.

bit	Description
0	Disable
1	Enable

3.6.2.13. dsp0_ClockShift

Clock shift register

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	-	-	-	-	-	-	-	-
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	dsp0_ClockOffset[23:16]							
ACCESS_TYPE	RW	RW	RW	RW	RW	RW	RW	RW
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	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	-	-	-	-	-	-	-	-
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	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	dsp0_Clock Invert
ACCESS_TYPE	-	-	-	-	-	-	-	RW
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	1

[bit31:24] Reserved: Reserved bits

[bit23:16] dsp0_ClockOffset: Sets the offset in reference clock cycles for the display clock output with reference to the data output. This has to be smaller than the integer part of ClockDivider.

[bit15:1] Reserved: Reserved bits

[bit0] dsp0_ClockInvert: Delays the display clock output for one generated display clock cycle.

With the settings in this register the display clock can be shifted relative to the display data. With the ClockOffset and an 2D core display clock setting of DIV2 a maximum shift of only 180 degrees is possible, so to achieve shifts larger than 180 degrees the ClockInvert bit can be set. This basically adds 180 degrees to the shift set by ClockOffset. If the 2D core display clock settings is set to DIV1 the display output must not be used (only the display bypass output can be used then). These settings will not affect the display bypass output of the 2D core. These settings must not be changed when ClockEnable is set to 1.

3.6.2.14. hpm_LockUnlock

Register to change the protection status of the address block 3 (Offset is in the range of 0x0300 to 0x03FF) and HPM (Address is in the range of 0x50400000 to 0x504FFFFF).

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	hpm_LockUnlock [31:24]							
ACCESS_TYPE	W	W	W	W	W	W	W	W
PROT_TYPE	-							
INITIAL_VALUE	-	-	-	-	-	-	-	-

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	hpm_LockUnlock [23:16]							
ACCESS_TYPE	W	W	W	W	W	W	W	W
PROT_TYPE	-							
INITIAL_VALUE	-	-	-	-	-	-	-	-

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	hpm_LockUnlock [15:8]							
ACCESS_TYPE	W	W	W	W	W	W	W	W
PROT_TYPE	-							
INITIAL_VALUE	-	-	-	-	-	-	-	-

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	hpm_LockUnlock [7:0]							
ACCESS_TYPE	W	W	W	W	W	W	W	W
PROT_TYPE	-							
INITIAL_VALUE	-	-	-	-	-	-	-	-

[bit31:0] hpm_LockUnlock: The protection status is changed by writing one of the following key values to this field

bit	Description
0x5651F763	Decrements the unlock counter. When the counter value is null, lock protection is active. Reset counter value is 1.
0x691DB936	Increments the unlock counter. Max allowed value is 15.
0xAEE95CDC	Enables privilege protection. Disabled after reset.
0xB5E2466E	Disables privilege protection.
0xFBE8B1E6	Freezes current protection status. Writing keys to this register has no more effect until reset.

When lock protection is active, no write but only read access is possible to all registers of this address block. When privilege protection is active, only privileged read and write access is possible. Both protections can be active at the same time. Reading this register returns the current unlock counter value, but results in an error response (for HW test purposes only).

3.6.2.15. hpm_LockStatus

Protection status of the address block 3 and HPM.

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	-	-	-	-	-	-	-	-
PROT_TYPE	-							
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	-	-	-	-	-	-	-	-
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	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	hpm_FreezeStatus
ACCESS_TYPE	-	-	-	-	-	-	-	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	Reserved	Reserved	Reserved	hpm_PrivilegeStatus	Reserved	Reserved	Reserved	hpm_LockStatus
ACCESS_TYPE	-	-	-	R	-	-	-	R
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:9] Reserved: Reserved bits

[bit8] hpm_FreezeStatus: Current freeze status

bit	Description
0	protection status can be changed
1	cannot be changed.

[bit7:5] Reserved: Reserved bits

[bit4] hpm_PrivilegeStatus: Current status of privilege protection

bit	Description
0	Inactive
1	active.

[bit3:1] Reserved: Reserved bits

[bit0] hpm_LockStatus: Current status of lock protection

bit	Description
0	inactive (unlock counter > 0)
1	active (unlock counter == 0)

3.6.3. 2D Core

See HW Peripheral Manual for 2D Graphic Core v1.1.

3.6.4. LCD Bus Interface

See the chapter of LCD BUS Interface in this manual.

3.6.5. NIC

The lock feature of this configuration interface is implemented in module SubSysCtrl in registers hpm_LockUnlock and hpm_LockStatus.

See also Arm CoreLink Network Interconnect NIC301 r2p3 – Technical Reference Manual.

Addr. Offset	Register Name	Prot.	Register Description (Address Block 1)
Addr. Offset	Register Name	Prot.	Register Description (Address Block 2)
Addr. Offset	Register Name	Prot.	Register Description (Address Block 3)
0x5044	ahb_ctrl_ahb_cntl	no	ahb_ctrl_ahb_cntl register
Addr. Offset	Register Name	Prot.	Register Description (Address Block 4)
0x42100	msm_read_qos	no	msm_read_qos register
0x42104	msm_write_qos	no	msm_write_qos register
Addr. Offset	Register Name	Prot.	Register Description (Address Block 5)
0x49100	fetchdecode0_read_qos	no	fetchdecode0_read_qos register
Addr. Offset	Register Name	Prot.	Register Description (Address Block 6)
0x4A100	fetchlayer0_read_qos	no	fetchlayer0_read_qos register

3.6.5.1. ahb_ctrl_ahb_cntl

ahb_ctrl_ahb_cntl register

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	reserved	reserved	reserved	reserved	reserved	reserved	reserved	reserved
ACCESS_TYPE	-	-	-	-	-	-	-	-
PROT_TYPE	-							
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	-	-	-	-	-	-	-	-
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	reserved	reserved	reserved	reserved	reserved	reserved	reserved
ACCESS_TYPE	-	-	-	-	-	-	-	-
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	reserved	reserved	reserved	reserved	reserved	ahb_ctrl_force_incr	ahb_ctrl_decerr_en
ACCESS_TYPE	-	-	-	-	-	-	RW	RW
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:2] Reserved: Reserved bits

[bit1] ahb_ctrl_force_incr: Reserved field. Do not change from reset value!

[bit0] ahb_ctrl_decerr_en:Unaligned transfers can not be correctly forwarded by this AXI to AHB Bridge and thus data corruption can occur. Setting this to 1 will not avoid the data corruption, it will only cause a decode error to be generated upon the unaligned transfer occuring.

bit	Description
0	Disable
1	Enable

3.6.5.2. msm_read_qos

msm_read_qos register

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	reserved	reserved	reserved	reserved	reserved	reserved	reserved	reserved
ACCESS_TYPE	-	-	-	-	-	-	-	-
PROT_TYPE	-							
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	-	-	-	-	-	-	-	-
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	reserved	reserved	reserved	reserved	reserved	reserved	reserved
ACCESS_TYPE	-	-	-	-	-	-	-	-
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	reserved	reserved	reserved	msm_ar_qos[3:0]			
ACCESS_TYPE	-	-	-	-	RW	RW	RW	RW
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	1

[bit31:4] Reserved: Reserved bits

[bit3:0] msm_ar_qos: Read channel QoS value. Higher value means higher priority in arbitration in interconnect.

3.6.5.3. msm_write_qos

msm_write_qos register

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	reserved	reserved	reserved	reserved	reserved	reserved	reserved	reserved
ACCESS_TYPE	-	-	-	-	-	-	-	-
PROT_TYPE	-							
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	-	-	-	-	-	-	-	-
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	reserved	reserved	reserved	reserved	reserved	reserved	reserved
ACCESS_TYPE	-	-	-	-	-	-	-	-
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	reserved	reserved	reserved	msm_aw_qos[3:0]			
ACCESS_TYPE	-	-	-	-	RW	RW	RW	RW
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	1

[bit31:4] Reserved: Reserved bits

[bit3:0] msm_aw_qos: Write channel QoS value. Higher value means higher priority in arbitration in interconnect.

3.6.5.4. fetchdecode0_read_qos

fetchdecode0_read_qos register

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	reserved	reserved	reserved	reserved	reserved	reserved	reserved	reserved
ACCESS_TYPE	-	-	-	-	-	-	-	-
PROT_TYPE	-							
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	-	-	-	-	-	-	-	-
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	reserved	reserved	reserved	reserved	reserved	reserved	reserved
ACCESS_TYPE	-	-	-	-	-	-	-	-
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	reserved	reserved	reserved	fetchdecode0_ar_qos			
ACCESS_TYPE	-	-	-	-	RW	RW	RW	RW
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	1

[bit31:4] Reserved: Reserved bits

[bit3:0] fetchdecode0_ar_qos: Read channel QoS value. Higher value means higher priority in arbitration in interconnect.

3.6.5.5. fetchlayer0_read_qos

fetchlayer0_read_qos register

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	reserved	reserved	reserved	reserved	reserved	reserved	reserved	reserved
ACCESS_TYPE	-	-	-	-	-	-	-	-
PROT_TYPE	-							
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	-	-	-	-	-	-	-	-
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	reserved	reserved	reserved	reserved	reserved	reserved	reserved
ACCESS_TYPE	-	-	-	-	-	-	-	-
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	reserved	reserved	reserved	fetchlayer0_ar_qos			
ACCESS_TYPE	-	-	-	-	RW	RW	RW	RW
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	1

[bit31:4] Reserved: Reserved bits

[bit3:0] fetchlayer0_ar_qos: Read channel QoS value. Higher value means higher priority in arbitration in interconnect.

3.6.6. aic_performance

Addr. Offset	Register Name	Prot.	Register Description (Address Block 0)
0x0000	Control	no	Control register
0x0004	Timer	no	Timer control register
0x0008	ID_Mask	no	ID mask register
0x000C	ID_Value	no	ID value register
0x0010	SW_Tag	no	Software tag register
0x0014	MeasurementTimeControl	no	Measurement time control
0x0018	MeasurementTime	no	Measurement time
0x001C	GlobalCounter	no	Global counter register
0x0020	MU0_Switch	no	MU0 source select register
0x0024	MU0_DataCounter	no	MU0 data counter register
0x0028	MU0_BusyCounter	no	MU0 busy counter register
0x002C	MU0_TransferCounter	no	MU0 transfer counter register
0x0030	MU0_AddrbusyCounter	no	MU0 address busy counter register
0x0034	MU0_LatencyCounter	no	MU0 latency counter register
0x0038	MU1_Switch	no	MU1 source select register
0x003C	MU1_DataCounter	no	MU1 data counter register
0x0040	MU1_BusyCounter	no	MU1 busy counter register
0x0044	MU1_TransferCounter	no	MU1 transfer counter register
0x0048	MU1_AddrbusyCounter	no	MU1 address busy counter register
0x004C	MU1_LatencyCounter	no	MU1 latency counter register

3.6.6.1. Control

Control register

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	OTCDisable	reserved	reserved	reserved	reserved	reserved	reserved	reserved
ACCESS_TYPE	RW	-	-	-	-	-	-	-
PROT_TYPE	-							
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	-	-	-	-	-	-	-	-
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	reserved	reserved	reserved	reserved	reserved	reserved	reserved
ACCESS_TYPE	-	-	-	-	-	-	-	-
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	reserved	reserved	reserved	reserved	reserved	Mode	Enable
ACCESS_TYPE	-	-	-	-	-	-	RW	RW
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31] OTCDisable: Disables OTC counters

bit	Description
0	Enable
1	Disable

[bit30:2] Reserved: Reserved bits

[bit1] Mode: Measurement Mode.

bit	Description
0	MANUAL: Manual measurement end
1	TIMER: Timer controlled measurement end

[bit0] Enable: Measurement Enable, write 1 to enable, read returns current status of measurement.

3.6.6.2. Timer

Timer control register

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Divider [31:28]				Load[27:24]			
ACCESS_TYPE	RW	RW	RW	RW	RW	RW	RW	RW
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	1	1	1	1

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Load[23:16]							
ACCESS_TYPE	RW	RW	RW	RW	RW	RW	RW	RW
PROT_TYPE	-							
INITIAL_VALUE	1	1	1	1	1	1	1	1

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Load[15:8]							
ACCESS_TYPE	RW	RW	RW	RW	RW	RW	RW	RW
PROT_TYPE	-							
INITIAL_VALUE	1	1	1	1	1	1	1	1

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Load[7:0]							
ACCESS_TYPE	RW	RW	RW	RW	RW	RW	RW	RW
PROT_TYPE	-							
INITIAL_VALUE	1	1	1	1	1	1	1	1

[bit31:28] Divider: Predivider value of timer; preset value is 2^Divider-1

[bit27:0] Load: Timer preset value

3.6.6.3. ID_Mask

ID mask register

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	reserved	reserved	reserved	reserved	reserved	reserved	reserved	reserved
ACCESS_TYPE	-	-	-	-	-	-	-	-
PROT_TYPE	-							
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	-	-	-	-	-	-	-	-
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	reserved	reserved	reserved	reserved	reserved	reserved	reserved
ACCESS_TYPE	-	-	-	-	-	-	-	-
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Mask[7:0]							
ACCESS_TYPE	RW	RW	RW	RW	RW	RW	RW	RW
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:8] reserved: Reserved bits

[bit7:0] Mask: ID mask register; 0 means ID check disabled.

3.6.6.4. ID_Value

ID value register

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	reserved	reserved	reserved	reserved	reserved	reserved	reserved	reserved
ACCESS_TYPE	-	-	-	-	-	-	-	-
PROT_TYPE	-							
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	-	-	-	-	-	-	-	-
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	reserved	reserved	reserved	reserved	reserved	reserved	reserved
ACCESS_TYPE	-	-	-	-	-	-	-	-
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Value[7:0]							
ACCESS_TYPE	RW	RW	RW	RW	RW	RW	RW	RW
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:8] reserved: Reserved bits

[bit7:0] Value: ID value that should match the ID of the transfer after Mask has been applied. Only if the ID matches will the transfer be considered for measurement.

3.6.6.5. SW_Tag

Software tag register

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Tag[31:24]							
ACCESS_TYPE	RW	RW	RW	RW	RW	RW	RW	RW
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Tag[23:16]							
ACCESS_TYPE	RW	RW	RW	RW	RW	RW	RW	RW
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Tag[15:8]							
ACCESS_TYPE	RW	RW	RW	RW	RW	RW	RW	RW
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Tag[7:0]							
ACCESS_TYPE	RW	RW	RW	RW	RW	RW	RW	RW
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:0] Tag: Software tag for correlation of measurement results and application point of execution.

3.6.6.6. MeasurementTimeControl

Measurement time control

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	MTEnable	reserved	reserved	reserved	reserved	reserved	reserved	reserved
ACCESS_TYPE	RW	-	-	-	-	-	-	-
PROT_TYPE	-							
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	MTDivider [19:16]			
ACCESS_TYPE	-	-	-	-	RW	RW	RW	RW
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	MTDivider [15:8]							
ACCESS_TYPE	RW	RW	RW	RW	RW	RW	RW	RW
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	MTDivider [7:0]							
ACCESS_TYPE	RW	RW	RW	RW	RW	RW	RW	RW
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31] MTEnable: Enable measurement time

[bit19:0] MTDivider: Predivider for measurement time

3.6.6.7. MeasurementTime

Measurement time

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Time [31:24]							
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	Time [23:16]							
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	Time [15:8]							
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	Time [7:0]							
ACCESS_TYPE	R	R	R	R	R	R	R	R
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:0] Time: Current timestamp of measurement.

3.6.6.8. GlobalCounter

Global counter register

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Global [31:24]							
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	Global [23:16]							
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	Global [15:8]							
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	Global [7:0]							
ACCESS_TYPE	R	R	R	R	R	R	R	R
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:0] Global: Counts total number of clock cycles of measurement period.

3.6.6.9. MU0_Switch

MU0 source select register

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	reserved	reserved	reserved	reserved	reserved	reserved	reserved	reserved
ACCESS_TYPE	-	-	-	-	-	-	-	-
PROT_TYPE	-							
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	-	-	-	-	-	-	-	-
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	reserved	reserved	reserved	reserved	reserved	reserved	reserved
ACCESS_TYPE	-	-	-	-	-	-	-	-
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	reserved	reserved	reserved	reserved	MU0_Switch[2:0]		
ACCESS_TYPE	-	-	-	-	-	RW	RW	RW
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:3] Reserved: Reserved bits

[bit2:0] MU0_Switch: Source select for measurement unit 0

bit	Description
0000	MSM_READ: msm read direction
0001	MSM_WRITE: msm write direction
0010	FETCHDECODE0_READ: fetchdecode0 read direction
0011	FETCHLAYER0_READ: fetchlayer0 read direction
Other than above	Setting prohibited

3.6.6.10. MU0_DataCounter

MU0 data counter register

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	MU0_Data [31:24]							
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	MU0_Data [23:16]							
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	MU0_Data [15:8]							
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	MU0_Data [7:0]							
ACCESS_TYPE	R	R	R	R	R	R	R	R
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:0] MU0_Data: Counts valid data cycles within measurement period.

3.6.6.11. MU0_BusyCounter

MU0 busy counter register

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	MU0_Busy [31:24]							
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	MU0_Busy [23:16]							
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	MU0_Busy [15:8]							
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	MU0_Busy [7:0]							
ACCESS_TYPE	R	R	R	R	R	R	R	R
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:0] MU0_Busy: Counts busy cycles within measurement period.

3.6.6.12. MU0_TransferCounter

MU0 transfer counter register

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	MU0_Transfer [31:24]							
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	MU0_Transfer [23:16]							
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	MU0_Transfer [15:8]							
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	MU0_Transfer [7:0]							
ACCESS_TYPE	R	R	R	R	R	R	R	R
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:0] MU0_Transfer: Counts number of transfers withing measurement period.

3.6.6.13. MU0_AddrbusyCounter

MU0 address busy counter register

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	MU0_Addrbusy [31:24]							
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	MU0_Addrbusy [23:16]							
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	MU0_Addrbusy [15:8]							
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	MU0_Addrbusy [7:0]							
ACCESS_TYPE	R	R	R	R	R	R	R	R
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:0] MU0_Addrbusy: Counts address bus busy cycles within measurement period.

3.6.6.14. MU0_LatencyCounter

MU0 latency counter register

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	MU0_Latency [31:24]							
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	MU0_Latency [23:16]							
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	MU0_Latency [15:8]							
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	MU0_Latency [7:0]							
ACCESS_TYPE	R	R	R	R	R	R	R	R
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:0] MU0_Latency: Accumulates outstanding transfers within measurement period.

3.6.6.15. MU1_Switch

MU1 source select register

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	reserved	reserved	reserved	reserved	reserved	reserved	reserved	reserved
ACCESS_TYPE	-	-	-	-	-	-	-	-
PROT_TYPE	-							
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	-	-	-	-	-	-	-	-
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	reserved	reserved	reserved	reserved	reserved	reserved	reserved
ACCESS_TYPE	-	-	-	-	-	-	-	-
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	reserved	reserved	reserved	reserved	MU1_Switch[2:0]		
ACCESS_TYPE	-	-	-	-	-	RW	RW	RW
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:3] Reserved: Reserved bits

[bit2:0] MU1_Switch: Source select for measurement unit 1

bit	Description
0000	MSM_READ: msm read direction
0001	MSM_WRITE: msm write direction
0010	FETCHDECODE0_READ: fetchdecode0 read direction
0011	FETCHLAYER0_READ: fetchlayer0 read direction
Other than above	Setting prohibited

3.6.6.16. MU1_DataCounter

MU1 data counter register

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	MU1_Data [31:24]							
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	MU1_Data [23:16]							
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	MU1_Data [15:8]							
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	MU1_Data [7:0]							
ACCESS_TYPE	R	R	R	R	R	R	R	R
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:0] MU1_Data: Counts valid data cycles within measurement period.

3.6.6.17. MU1_BusyCounter

MU1 busy counter register

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	MU1_Busy [31:24]							
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	MU1_Busy [23:16]							
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	MU1_Busy [15:8]							
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	MU1_Busy [7:0]							
ACCESS_TYPE	R	R	R	R	R	R	R	R
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:0] MU1_Busy: Counts busy cycles within measurement period.

3.6.6.18. MU1_TransferCounter

MU1 transfer counter register

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	MU1_Transfer [31:24]							
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	MU1_Transfer [23:16]							
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	MU1_Transfer [15:8]							
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	MU1_Transfer [7:0]							
ACCESS_TYPE	R	R	R	R	R	R	R	R
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:0] MU1_Transfer: Counts number of transfers withing measurement period.

3.6.6.19. MU1_AddrbusyCounter

MU1 address busy counter register

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	MU1_Addrbusy [31:24]							
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	MU1_Addrbusy [23:16]							
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	MU1_Addrbusy [15:8]							
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	MU1_Addrbusy [7:0]							
ACCESS_TYPE	R	R	R	R	R	R	R	R
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:0] MU1_Addrbusy: Counts address bus busy cycles within measurement period.

3.6.6.20. MU1_LatencyCounter

MU1 latency counter register

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	MU1_Latency [31:24]							
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	MU1_Latency [23:16]							
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	MU1_Latency [15:8]							
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	MU1_Latency [7:0]							
ACCESS_TYPE	R	R	R	R	R	R	R	R
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:0] MU1_ Latency: Accumulates outstanding transfers within measurement period.

3.6.7. aic_monitor

Addr. Offset	Register Name	Prot.	Register Description (Address Block 0)
0x0000	MonitorDisable	no	Interconnect error monitor disable register
0x0004	MonitorInterruptEnable	no	Interconnect error monitor non maskable interrupt enable register
0x0008	MonitorInterruptClear	no	Interconnect error monitor non maskable interrupt clear register
0x000C	MonitorStatus	no	Interconnect error monitor status register

3.6.7.1. MonitorDisable

Interconnect error monitor disable register

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	reserved	reserved	reserved	reserved	reserved	reserved	reserved	reserved
ACCESS_TYPE	-	-	-	-	-	-	-	-
PROT_TYPE	-							
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	-	-	-	-	-	-	-	-
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	reserved	reserved	reserved	reserved	reserved	reserved	reserved
ACCESS_TYPE	-	-	-	-	-	-	-	-
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	reserved	reserved	reserved	reserved	fetchlayer0	fetchdecode0	msm
ACCESS_TYPE	-	-	-	-	-	RW	RW	RW
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:3] Reserved: Reserved bits

[bit2] fetchlayer0: Master fetchlayer0

[bit1] fetchdecode0: Master fetchdecode0

[bit0] msm: Master msm

The interconnect error monitor is disabled for every master for which the correspondig field is set to 1.

3.6.7.2. MonitorInterruptEnable

Interconnect error monitor non maskable interrupt enable register

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	reserved	reserved	reserved	reserved	reserved	reserved	reserved	reserved
ACCESS_TYPE	-	-	-	-	-	-	-	-
PROT_TYPE	-							
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	-	-	-	-	-	-	-	-
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	reserved	reserved	reserved	reserved	reserved	reserved	reserved
ACCESS_TYPE	-	-	-	-	-	-	-	-
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	reserved	reserved	reserved	reserved	reserved	reserved	MonitorInterruptEnable
ACCESS_TYPE	-	-	-	-	-	-	-	RW
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	1

[bit31:1] Reserved: Reserved bits

[bit0] MonitorInterruptEnable: This field can only be programmed once after reset. A second attempt at programming this will cause an error response and no change to the enable. If set to '1' the non maskable interupt is enabled, if set to '0' the non maskable interrupt is disabled and only the MonitorStatus field can be used to detect errors.

3.6.7.3. MonitorInterruptClear

Interconnect error monitor non maskable interrupt enable register

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	reserved	reserved	reserved	reserved	reserved	reserved	reserved	reserved
ACCESS_TYPE	-	-	-	-	-	-	-	-
PROT_TYPE	-							
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	-	-	-	-	-	-	-	-
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	reserved	reserved	reserved	reserved	reserved	reserved	reserved
ACCESS_TYPE	-	-	-	-	-	-	-	-
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	reserved	reserved	reserved	reserved	reserved	reserved	MonitorInterruptClear
ACCESS_TYPE	-	-	-	-	-	-	-	W1P
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	-

[bit31:1] Reserved: Reserved bits

[bit0] MonitorInterruptClear: Writing a 1 clears the non maskable interrupt and status and starts checking for next error.

3.6.7.4. MonitorStatus

Interconnect error monitor status register

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	reserved	reserved	reserved	reserved	reserved	reserved	reserved	reserved
ACCESS_TYPE	-	-	-	-	-	-	-	-
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	ErrorSubID[23:16]							
ACCESS_TYPE	R	R	R	R	R	R	R	R
PROT_TYPE	-							
INITIAL_VALUE	1	1	1	1	1	1	1	1

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	reserved	reserved	reserved	reserved	reserved	ErrorID [10:8]		
ACCESS_TYPE	-	-	-	-	-	R	R	R
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	1	1	1

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	reserved	reserved	reserved	reserved	reserved	reserved	ErrorType[1:0]	
ACCESS_TYPE	-	-	-	-	-	-	R	R
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:24] Reserved: Reserved bits

[bit23:16] ErrorSubID: ID provided by the master itself that caused the erronous request.

[bit15:11] Reserved: Reserved bits

[bit10:8] ErrorID: ID of master that did erronous request

bit	Description
0000	MSM: Master msm
0001	FETCHDECODE0: Master fetchdecode0
0010	FETCHLAYER0: Master fetchlayer0
0111	NONE: No error detected

[bit1:0] ErrorType: Type of detected error

bit	Description
0000	NONE: No error detected
0010	SLVERR: Slave signaled error
0011	DECERR: Decoder signaled error

CHAPTER 34: Automotive Remote Handler

This chapter explains the functions and operations of the Automotive Remote Handler (ARH).

1. Outline of ARH
2. ARH registers
3. Operation of ARH
4. Notes on using ARH
5. Use cases

CODE: APIXE-S6J3300-E1

1. Outline of ARH

This section describes the features and the block diagram of the ARH module.

■ Features of ARH

The ARH module supports two channels. The ARH module provides secure handling and control of remote peripheral hardware. It supports remote event handling to react to interrupts from remote peripherals. The features of the ARH module are listed as follows:

- Supports two Automotive Interconnect links
- Communicates between APIX® PHY and core
- Capable of transmission of write and read messages and reception of read response message
- Supports up to 16 Transaction Buffers with fixed propriety arbitration
- Contains Event Buffer for reception of events from remote device
- Can handle upto 128 events from remote device
- Supports bypass mode through use of Transaction Buffer 0, 1, 2, and 3
- Can handle up to 256 Transaction Frame index
- Supports bulk read and write access (DMA include)
- Support for error handling

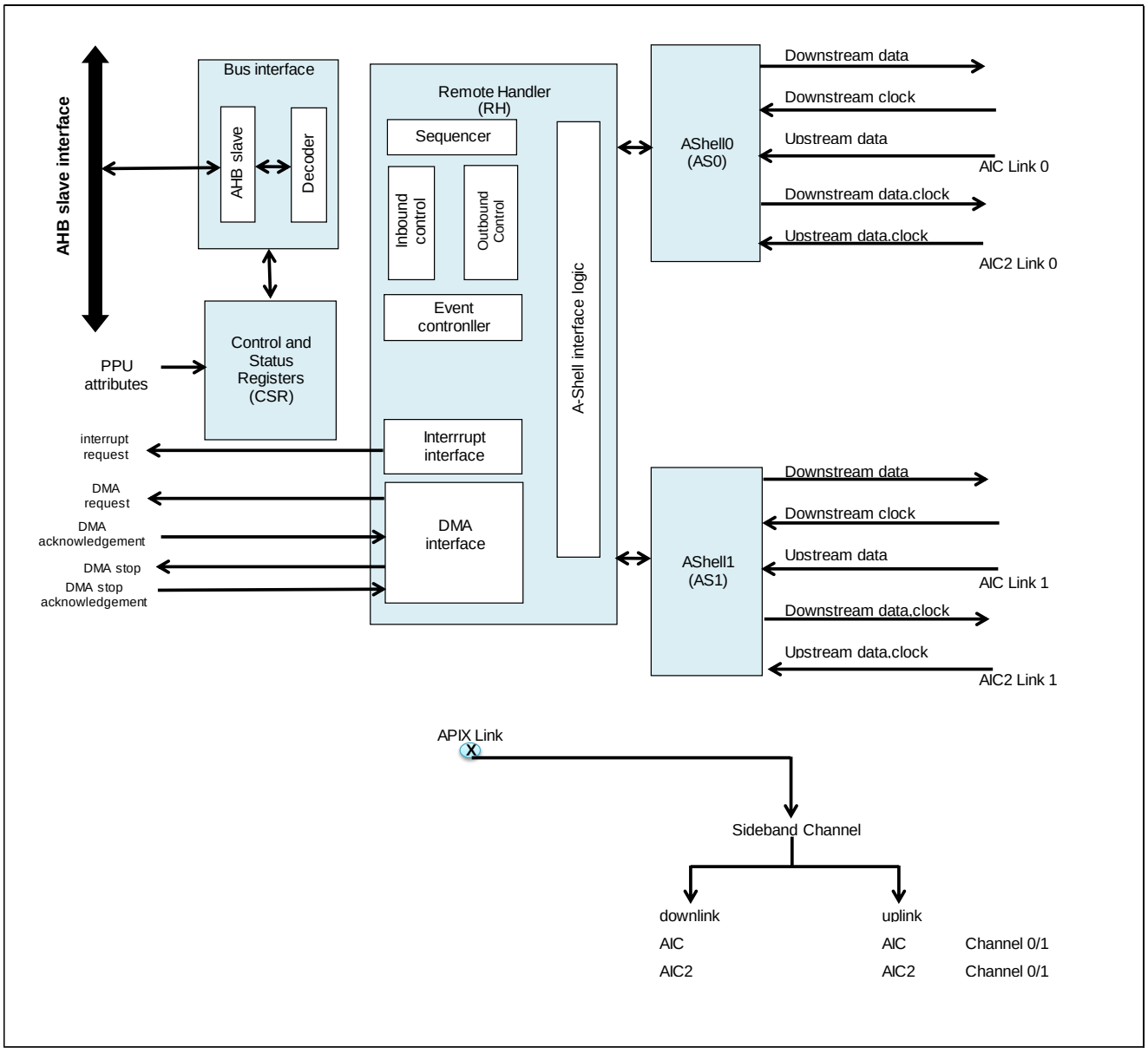
■ Abbreviations

Table 34-1: Abbrevations

Term	Meaning
AHB	AMBA High Speed Bus
AIC	Automotive interconnect
APIX®	Automotive Pixel Link
A-Shell	Automotive Shell
ARH	Automotive Remote Handler
BSU	Bus Support Unit
CPU	Central Processing Unit
CRC	Cyclic Redundancy Check
CSR	Control and Status Registers
EVBUF	Event Buffer
RH	Remote Handler
TB	Transaction Buffer
TF	Transaction Frame
AAC	A-Shell-to-A-Shell Connector
LPF	Low Pass Filter
RX	Receiver
TX	Transmitter
PRBS	Pseudo Random Binary Sequence
BIST	Built in Self Test
Misc	Miscellaneous

■ Block Diagram of ARH

Figure 34-1:



Control and Status Register

The operation of ARH can be controlled and monitored through its Control and Status Registers (CSR). For details of the CSR refer to Section 1-2.

AHB Interface and Address Decoding

The AHB master in the MCU can access the ARH module through its AHB slave interface. The address decoder decodes the AHB address bus.

DMA Interface

The ARH supports 16 DMA channels, each per Transaction Buffer. In case of a severe error (fatal) from A-Shell detected, the DMA transfers can be stopped by issuing a DMA stop request.

Remote Handler

The ARH Remote Handler block generates and handles various message types. It generates write messages, read request messages, and sends them to A-Shell. This then performs low level services on them and transmits them to the remote peripheral. The A-Shell receives read response messages and event messages from remote peripheral, performs low level services, and forwards the received frame to the Remote Handler for further processing. The Remote Handler on reception of frame from A-Shell, decodes the frame and stores the information for further processing by software.

A-Shell

The A-Shell interfaces between the Remote Handler and the remote peripheral. It provides various low level services which include transaction framing and de-framing, establishing and maintaining transaction
alignment, exchange of transaction utilization services, bit error detection, bit error management, and report of status information.
Connection over APIX® PHY, AIC or AIC2 link is possible.

Note: Please refer to the device datasheet for the availability of APIX® PHY, AIC or AIC2 link.

2. ARH Registers

The ARH 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 the memory protection violation. The ARH module is allocated 1 KB of MCU address space for mapping the Configuration and Status Registers (i.e. CSRs). This section describes the registers of the ARH in details.

The suffix ‘n’ in the register name indicates that the register is an instance ‘n’ of the module.

- Registers of ARH

The following registers are available for ARH:

- ARH Remote Handler Control Register (ARHn_RHCTRL)
- ARH Channel Control Register (ARHn_CHCTRL0 - Channel 0)
- ARH Channel Control Register (ARHn_CHCTRL1 - Channel 1)
- ARH Channel Status Register (ARHn_CHSTAT0 - Channel 0, ARHn_CHSTAT1 - Channel 1)
- ARH Channel Watchdog Control Register (ARHn_CHWDGCTL0 - Channel 0, ARHn_CHWDGCTL1 - Channel 1)
- ARH Watchdog Counter Register (ARHn_CHWDGCNT0 - Channel 0, ARHn_CHWDGCNT1 - Channel 1)
- ARH Transaction Buffer Control Register (ARHn_TBCTRL0 to 15)
- ARH Transaction Buffer Interrupt Register (ARHn_TBIRQ)
- ARH Transaction Buffer Index Register (ARHn_TBIDX0 - Channel 0, ARHn_TBIDX1 - Channel 1)
- ARH Transaction Frame Control Register (ARHn_TFCTRL00 to 15)
- ARH Transaction Frame Index Register (ARHn_TFIDX0 to 15)
- ARH Transaction Frame Address Register (ARHn_TFADDR0 to 15)
- ARH Transaction Frame Data Register (ARHn_TFDATA0 to 15)
- ARH Transaction Frame Data Register (ARHn_TFDATA0 to 15)
- ARH Event Control Register (ARHn_EVCTRL)
- ARH Event Interrupt Control Register (ARHn_EVIRQC)
- ARH Event Buffer Register (ARHn_EVBUF0)
- ARH Event Buffer Register (ARHn_EVBUF1)
- ARH APIX® Configuration Register (ARHn_APCFG00 - Channel 0)
- ARH APIX® Configuration Register (ARHn_APCFG10 - Channel 1)
- ARH APIX® Configuration Register (ARHn_APCFG01 - Channel 0)
- ARH APIX® Configuration Register (ARHn_APCFG11 - Channel 1)
- ARH APIX® Configuration Register (ARHn_APCFG02 - Channel 0)
- ARH APIX® Configuration Register (ARHn_APCFG12 - Channel 1)
- ARH APIX® Configuration Register (ARHn_APCFG03 - Channel 0, ARHn_APCFG13 - Channel 1)
- ARH APIX® Configuration Register (ARHn_APCFG04 - Channel 0, ARHn_APCFG14 - Channel 1)
- ARH Test Register (ARHn_TST)
- ARH Unlock Register (ARHn_UNLOCK)
- ARH Module ID Register (ARHn_MID)
- APIX® PHY Evaluation Transmitter Pattern Register (ARHn_EVAL0)
- APIX® PHY Evaluation Receiver Pattern Register (ARHn_EVAL1)
- APIX® PHY Evaluation Configuration Register (ARHn_EVAL2)
- APIX® PHY Evaluation Receiver Status Register (ARHn_EVAL3)
- APIX® PHY Evaluation Misc Test Enable Register (ARHn_EVAL4)
- APIX® PHY Evaluation Status Register (ARHn_EVAL5)
- APIX® PHY Evaluation Misc Test Configuration Register (ARHn_EVAL6)
- APIX® PHY Evaluation Misc Test Status Register (ARHn_EVAL7)

- Memory layout of ARH registers

Table 34-2: Memory Map of ARH Registers

Offset	+3	+2	+1	+0
0x00000000	ARHn_RHCTRL 00000000 00000000 00000000 00000001			
0x00000004	ARHn_CHCTRL0 00001111 00000000 00000000 00000000			
0x00000008	ARHn_CHSTAT0 00000000 00000000 00000000 00000000			
0x0000000C	ARHn_CHWDGCTL0 00000000 00000000 00000000 00000000			
0x00000010	ARHn_CHWDGCNT0 00000000 00000000 XXXXXXXX XXXXXXXX			
0x00000014	ARHn_CHCTRL1 00000111 00000000 00000000 00000000			
0x00000018	ARHn_CHSTAT1 00000000 00000000 00000000 00000000			
0x0000001C	ARHn_CHWDGCTL1 00000000 00000000 00000000 00000000			
0x00000020	ARHn_CHWDGCNT1 00000000 00000000 XXXXXXXX XXXXXXXX			
0x00000024	ARHn_TBCTRL0 00000000 00000000 00000000 00000000			
0x00000028	ARHn_TBCTRL1 00000000 00000000 00000000 00000000			
0x0000002C	ARHn_TBCTRL2 00000000 00000000 00000000 00000000			
0x00000030	ARHn_TBCTRL3 00000000 00000000 00000000 00000000			
0x00000034	ARHn_TBCTRL4 00000000 00000000 00000000 00000000			

Offset	+3	+2	+1	+0
0x00000038	ARHn_TBCTRL5 00000000 00000000 00000000 00000000			
0x0000003C	ARHn_TBCTRL6 00000000 00000000 00000000 00000000			
0x00000040	ARHn_TBCTRL7 00000000 00000000 00000000 00000000			
0x00000044	ARHn_TBCTRL8 00000000 00000000 00000000 00000000			
0x00000048	ARHn_TBCTRL9 00000000 00000000 00000000 00000000			
0x0000004C	ARHn_TBCTRL10 00000000 00000000 00000000 00000000			
0x00000050	ARHn_TBCTRL11 00000000 00000000 00000000 00000000			
0x00000054	ARHn_TBCTRL12 00000000 00000000 00000000 00000000			
0x00000058	ARHn_TBCTRL13 00000000 00000000 00000000 00000000			
0x0000005C	ARHn_TBCTRL14 00000000 00000000 00000000 00000000			
0x00000060	ARHn_TBCTRL15 00000000 00000000 00000000 00000000			
0x00000064	ARHn_TBIRQ 00000000 00000000		ARHn_TBIDX0 00000000	ARHn_TBIDX1 00000000
0x00000068	ARHn_TFCTRL0 00000000	ARHn_TFIDX0 00000000	ARHn_TFCTRL1 00000000	ARHn_TFIDX1 00000000
0x0000006C	ARHn_TFCTRL2 00000000	ARHn_TFIDX2 00000000	ARHn_TFCTRL3 00000000	ARHn_TFIDX3 00000000

Offset	+3	+2	+1	+0
0x00000070	ARHn_TFCTRL4 00000000	ARHn_TFIDX4 00000000	ARHn_TFCTRL5 00000000	ARHn_TFIDX5 00000000
0x00000074	ARHn_TFCTRL6 00000000	ARHn_TFIDX6 00000000	ARHn_TFCTRL7 00000000	ARHn_TFIDX7 00000000
0x00000078	ARHn_TFCTRL8 00000000	ARHn_TFIDX8 00000000	ARHn_TFCTRL9 00000000	ARHn_TFIDX9 00000000
0x0000007C	ARHn_TFCTRL10 00000000	ARHn_TFIDX10 00000000	ARHn_TFCTRL11 00000000	ARHn_TFIDX11 00000000
0x00000080	ARHn_TFCTRL12 00000000	ARHn_TFIDX12 00000000	ARHn_TFCTRL13 00000000	ARHn_TFIDX13 00000000
0x00000084	ARHn_TFCTRL14 00000000	ARHn_TFIDX14 00000000	ARHn_TFCTRL15 00000000	ARHn_TFIDX15 00000000
0x00000088	ARHn_TFADDR0 00000000 00000000 00000000 00000000			
0x0000008C	ARHn_TFADDR1 00000000 00000000 00000000 00000000			
0x00000090	ARHn_TFADDR2 00000000 00000000 00000000 00000000			
0x00000094	ARHn_TFADDR3 00000000 00000000 00000000 00000000			
0x00000098	ARHn_TFADDR4 00000000 00000000 00000000 00000000			
0x0000009C	ARHn_TFADDR5 00000000 00000000 00000000 00000000			
0x000000A0	ARHn_TFADDR6 00000000 00000000 00000000 00000000			
0x000000A4	ARHn_TFADDR7 00000000 00000000 00000000 00000000			

Offset	+3	+2	+1	+0
0x000000A8	ARHn_TFADDR8 00000000 00000000 00000000 00000000			
0x000000AC	ARHn_TFADDR9 00000000 00000000 00000000 00000000			
0x000000B0	ARHn_TFADDR10 00000000 00000000 00000000 00000000			
0x000000B4	ARHn_TFADDR11 00000000 00000000 00000000 00000000			
0x000000B8	ARHn_TFADDR12 00000000 00000000 00000000 00000000			
0x000000BC	ARHn_TFADDR13 00000000 00000000 00000000 00000000			
0x000000C0	ARHn_TFADDR14 00000000 00000000 00000000 00000000			
0x000000C4	ARHn_TFADDR15 00000000 00000000 00000000 00000000			
0x000000C8	ARHn_TFDATA0 00000000 00000000 00000000 00000000			
0x000000CC	ARHn_TFDATA1 00000000 00000000 00000000 00000000			
0x000000D0	ARHn_TFDATA2 00000000 00000000 00000000 00000000			
0x000000D4	ARHn_TFDATA3 00000000 00000000 00000000 00000000			
0x000000D8	ARHn_TFDATA4 00000000 00000000 00000000 00000000			
0x000000DC	ARHn_TFDATA5 00000000 00000000 00000000 00000000			

Offset	+3	+2	+1	+0
0x000000E0	ARHn_TFDATA6 00000000 00000000 00000000 00000000			
0x000000E4	ARHn_TFDATA7 00000000 00000000 00000000 00000000			
0x000000E8	ARHn_TFDATA8 00000000 00000000 00000000 00000000			
0x000000EC	ARHn_TFDATA9 00000000 00000000 00000000 00000000			
0x000000F0	ARHn_TFDATA10 00000000 00000000 00000000 00000000			
0x000000F4	ARHn_TFDATA11 00000000 00000000 00000000 00000000			
0x000000F8	ARHn_TFDATA12 00000000 00000000 00000000 00000000			
0x000000FC	ARHn_TFDATA13 00000000 00000000 00000000 00000000			
0x00000100	ARHn_TFDATA14 00000000 00000000 00000000 00000000			
0x00000104	ARHn_TFDATA15 00000000 00000000 00000000 00000000			
0x00000108	ARHn_EVCTRL 00000000 00000000 00000000 10000000			
0x0000010C	ARHn_EVIRQC 00000000 00000000 00000000 00000000			
0x00000110	ARHn_EVBUF0 0000000X XXXXXXXX 00000000 00000000			
0x00000114	ARHn_EVBUF1 XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX			

Offset	+3	+2	+1	+0
0x00000118	ARHn_APCFG00 00000000 00110000 00000000 10010000			
0x0000011C	ARHn_APCFG01 11110000 00000000 00000000 01001000			
0x00000120	ARHn_APCFG02 00000010 00000010 01000000 00000000			
0x00000124	ARHn_APCFG03 00100110 10100100 10011010 00000000			
0x00000128	ARHn_APCFG10 00000000 00110000 00000000 10010000			
0x0000012C	ARHn_APCFG11 11110000 00000000 00000000 01001000			
0x00000130	ARHn_APCFG12 00000010 00000010 01000000 00000000			
0x00000134	ARHn_APCFG13 00100110 10100100 10011010 00000000			
0x00000138	ARHn_TST 00000000 00000000 00000000 00000000			
0x0000013C	ARHn_UNLOCK 00000000 00000000 00000000 00000000			
0x00000140	ARHn_MID 00000000 00000000 00000000 00000000			
0x00000144	ARHn_APCFG04 00000000 00000000 00000000 00000000			
0x00000148	ARHn_APCFG14 00000000 00000000 00000000 00000000			
0x0000014C	ARHn_EVAL0 00000000 00000000 00000000 00000000			

Offset	+3	+2	+1	+0
0x00000150	ARHn_EVAL1 00000000 00000000 00000000 00000000			
0x00000154	ARHn_EVAL2 00000000 00000000 00000000 00001010			
0x00000158	ARHn_EVAL3 00000000 00000000 00000000 00000000			
0x0000015C	ARHn_EVAL4 00000000 00000000 00000000 00001100			
0x00000160	ARHn_EVAL5 00000000 00000000 00000000 00000000			
0x00000164	ARHn_EVAL6 00000000 00000000 00000000 00000000			
0x00000168	ARHn_EVAL7 00000000 00000000 00000000 00000000			

2.1.1. ARH Remote Handler Control Register (ARHn_RHCTRL)

ARH Remote Handler Control Register must be used by the software to program the ARH.
This register contains information on ARH module lock/unlock status, it contains various interrupts status information.
Trigger to Transaction Buffer, request for cancelling on pending buffer, and request for unlocking on waiting buffer are provided in this register.

■ ARH Remote Handler Control Register (ARHn_RHCTRL)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	UNLOCK	CANCEL	Reserved		TBNO[3:0]			
ACCESS_TYPE	R/W	R/W	R0,WX		R/W			
PROT_TYPE	-							
INITIAL_VALUE	0	0	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	WDG1	WDG0	FAT1	FAT0	Reserved	LV	OFL	EV
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R0,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	Reserved							LST
ACCESS_TYPE	R0,WX							R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	1

[bit31] UNLOCK: Transaction Buffer Unlock Request

Transaction buffer unlock request bit. For more details refer to Figure 34-3.

bit	Description
0	No unlock request made on TB buffer (ARHn_RHCTRL:TBNO) which is waiting
1	Request for unlock TB buffer (ARHn_RHCTRL: TBNO) which is in waiting state

Caution:

- *Upon unlock request, requested data (response) gets lost after using unlocked TB buffer with the same IDX.*

[bit30] CANCEL: Transaction Buffer Cancel Request

Transaction buffer cancel request bit. For more details refer to Figure 34-2.

bit	Description
0	No cancel request made on TB buffer (ARHn_RHCTRL:TBNO) which is in pending or active state
1	Request for cancel TB buffer (ARHn_RHCTRL:TBNO) which is pending or active state

[bit29:28] Reserved: Reserved bits

[bit27:24] TBNO[3:0]: Transaction Buffer Number Trigger (TBNO[3:0])

Writing starts transaction on buffer number TBNO if ARHn_RHCTRL:UNLOCK and ARHn_RHCTRL:CANCEL bits are '0'.

bit	Description
0000	Select TBNO 0 for performing transaction
…	
1111	Select TBNO 15 for performing transaction

[bit23:16] Reserved: Reserved bits

[bit15] WDG1: Watch Dog Timer 1 Interrupt

Read-only interrupt flag indicating the status of channel 1 watchdog transmit or receive interrupts.

bit	Description
0	Interrupt does not occur.
1	Transmit or receive interrupt (ARHn_CHWDGCTL1:WDTXIRQx/ARHn_CHWDGCTL1:WDRXIRQx) occurs depending on interrupt selection (ARHn_CHWDGCTL1:WTTX/ARHn_CHWDGCTL1:WTRX) and when corresponding interrupt enable is high.

[bit14] WDG0: Watch Dog Timer 0 Interrupt

Read-only interrupt flag indicating the status of channel 0 watchdog transmit or receive interrupts.

bit	Description
0	Interrupt does not occur.
1	Transmit or receive interrupt (ARHn_CHWDGCTL0:WDTXIRQx/ARHn_CHWDGCTL0:WDRXIRQx) occurs depending on interrupt selection (ARHn_CHWDGCTL0:WTTX/ARHn_CHWDGCTL0:WTRX) and when corresponding interrupt enable is high.

[bit13] FAT1: Fatal Interrupt Status from Channel 1

Read-only flag indicating fatal interrupt status.

bit	Description
0	When ARHn_CHCTRL1:FATIRQ = '1' and ARHn_CHCTRL1:FATEIN = '0'
1	When ARHn_CHCTRL1:FATIRQ = '1' and ARHn_CHCTRL1:FATEIN = '1'

[bit12] FAT0: Fatal Interrupt Status from Channel 0

Read-only flag indicating fatal interrupt status.

bit	Description
0	When ARHn_CHCTRL0:FATIRQ = '1' and ARHn_CHCTRL0:FATEIN = '0'
1	When ARHn_CHCTRL0:FATIRQ = '1' and ARHn_CHCTRL0:FATEIN = '1'

[bit11] Reserved: Reserved bits

[bit10] LV: Level Interrupt Status

Read-only flag indicating level interrupt status.

bit	Description
0	When ARHn_EVIRQC:EVIRQ = '1' and ARHn_EVIRQC:EVIEN = '0'
1	When ARHn_EVIRQC:EVIRQ = '1' and ARHn_EVIRQC:EVIEN = '1'

[bit9] OFL: Event Overflow Interrupt Status

Read-only flag indicating Event Buffer overflow interrupt status.

bit	Description
0	When ARHn_EVIRQC:OFLIRQ = '1' and ARHn_EVIRQC:OFLIEN = '0'
1	When ARHn_EVIRQC:OFLIRQ = '1' and ARHn_EVIRQC:OFLIEN = '1'

[bit8] EV: Event Interrupt Status

Read-only flag indicating Event Buffer interrupt status.

bit	Description
0	When ARHn_EVIRQC:EVIRQ = '1' and ARHn_EVIRQC:EVIEN = '0'
1	When ARHn_EVIRQC:EVIRQ = '1' and ARHn_EVIRQC:EVIEN = '1'

[bit7:1] Reserved: Reserved bits

[bit0] LST: ARH Lock Status

Read-only flag indicating ARH module lock status.

bit	Description
0	ARH module is unlocked
1	ARH module is locked

Note:

- *Following registers of ARH module are affected by LST bit. Registers ARHn_CHCTRL0, ARHn_CHCTRL1, and ARHn_EVCTRL cannot be written when LST = '1'. These registers can be written only when LST = '0'.*

2.1.2. ARH Channel Control Register (ARHn_CHCTRL0 - Channel 0)

ARH Channel Control Register can be used by software to program the channel.
ARHn_CHCTRL0 register is used for channel 0. This register can be written when ARHn_RHCTRL:LST = '0'.

■ ARH Channel Control Register (ARHn_CHCTRL0)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	BYPASS	FATEIN	Reserved	Reserved	PHYPLLRST	TXCFG	RSTRTA	INITRH
ACCESS_TYPE	R/W	R/W	R0,WX	R0,WX	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	1	1	1	1

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved	PHYPLLGOOD	PLLGOOD	UPHSK	DNHSK	FATAL	UPRDY	CONNECTED
ACCESS_TYPE	R0,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	UPVALID	DNVALID	FATIRQ	CRCERR	CRCTOUT	PERROR	READY	REMOTERST
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	UPVALIDCL	DNVALIDST	FATIRQCL	CRCERRCL	CRCTOUTCL	PERRORCL	READYCL	REMOTERSTCL
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] BYPASS: BYPASS Mode

In bypass mode, Transaction Buffer 0 (for A-Shell 0) is used for downstream data (outbound) and Transaction Buffer 1 (for A-Shell 0) is used for upstream data (inbound).
Transaction Buffer 2 (for A-Shell 1) is used for downstream data (outbound) and Transaction Buffer 3 (for A-Shell 1) is used for upstream data (inbound).

bit	Description
0	Remote Handler active
1	Remote Handler inactive

Note:

- *Valid written data in Transaction Buffer 0 or 2 is delivered to A-Shell by setting DNVALID.*
- *Valid received data in Transaction Buffer 1 or 3 from A-Shell is marked by setting UPVALID.*

[bit30] FATEIN: FATAL Interrupt Enable

bit	Description
0	Disables FATAL interrupt
1	Enables FATAL interrupt

[bit29:28] Reserved: Reserved bits

[bit27] PHYPLLRST: Reset for PLL

bit	Description
0	Normal Operation
1	Reset Active

[bit26] TXCFG: A-Shell Configuration

bit	Description
0	A-Shell and PHY running (write on ARHn_APCFG registers is disabled)
1	Enables A-Shell and PHY configuration (write on ARHn_APCFG registers is enabled which in turn configures A-Shell or PHY)

[bit25] RSTRTA: A-Shell Restart

bit	Description
0	A-Shell running
1	A-Shell initialization A-Shell stops operation, clears all state and history information then restart only when RSTRTA = '0'.

[bit24] INITRH: Remote Handler Initialization

bit	Description
0	Remote Handler running
1	Remote Handler initialization (no change of TB* and TF* registers)

Note:

- *ARHn_TBCTRL00:PENDING requests (set while INITRH == 1) is started with INITRH == 0.*

[bit23] Reserved: Reserved bits

[bit22] PHYPLLGOOD: APIX® PHY PLL Lock Status

Lock signal is directly fed into this register bit without any synchronization, indicating the actual status.

bit	Description
0	APIX® PHY PLL is unlocked
1	APIX® PHY PLL is locked

Note:

- *For devices which do not contain internal APIX® PHY then this bit is always be read 1. For APIX® PHY availability refer to device specific datasheet.*

[bit21] PLLGOOD: APIX® PHY PLL Lock Status

Lock signal is synchronized to reference clock and then reflected in this register bit.
When synchronization clock is not available, this bit is not updated.

bit	Description
0	APIX® PHY PLL is unlocked
1	APIX® PHY PLL is locked

Note:

- *For devices which do not contain internal APIX® PHY then this bit is always be read 1. For APIX® PHY availability refer to device specific datasheet.*

[bit20] UPHSK: Upstream Handshake

bit	Description
0	Indicates inbound handshake is in progress, or could not be performed
1	Indicates inbound handshake is performed

[bit19] DNHSK: Downstream Handshake

bit	Description
0	Indicates outbound handshake is in progress, or could not be performed
1	Indicates outbound handshake is performed

[bit18] FATAL: Fatal from A-Shell

bit	Description
0	Indicates that A-Shell is running without any severe errors
1	Indicates that A-Shell has encountered conditions where A-Shell cannot continue to deliver and receive transactions. FATAL is only one clock cycle active.

[bit17] UPRDY: Upstream Serial Channel Operation

bit	Description
0	Indicates that upstream serial channel (APIX® PHY) is not operational
1	Indicates that upstream serial channel (APIX® PHY) is operational

[bit16] CONNECTED: Connection Established

bit	Description
0	Indicates connection to remote APIX® is yet to be established
1	Indicates connection to remote APIX® is established

[bit15] UPVALID: Upstream Data Valid Status

bit	Description
0	Cleared by SW by setting ARHn_CHCTRL0:UPVALIDCL
1	Set by HW to mark upstream data as valid(ap_data_out_valid)

Note:

- *The bypass mode allows direct access and actions on downstream and upstream data and enable*
  ports of the A-Shell (without having the Remote Handler sequencer involved).
  In bypass mode Transaction Buffer 0 (for A-Shell 0) is used for downstream data (outbound) and Transaction Buffer 1 (for A-Shell 0) is used for upstream data (inbound).
  In bypass mode Transaction Buffer 2 (for A-Shell 1) is used for downstream data (outbound) and Transaction Buffer 3 (for A-Shell 1) is used for upstream data (inbound).
  Valid received data in Transaction Buffer 1 or 3 from A-Shell is marked by setting UPVALID.

[bit14] DNVALID: Downstream Data Valid Status

bit	Description
0	Cleared by HW after successful transfer to A-Shell
1	Set by SW by writing '1' to ARHn_CHCTRL0:DNVALIDST to mark downstream data as valid (ap_data_in_valid)

Note:

- *The bypass mode allows direct access and actions on downstream and upstream data and enable ports of the A-Shell (without having the Remote Handler sequencer involved). In bypass mode Transaction Buffer 0 (for A-Shell 0) is used for downstream data (outbound) and Transaction Buffer 1 (for A-Shell 0) is used for upstream data (inbound).*
- *In bypass mode Transaction Buffer 2 (for A-Shell 1) is used for downstream data (outbound) and Transaction Buffer 3 (for A-Shell 1) is used for upstream data (inbound).*
- *Valid written data in Transaction Buffer 0 or 2 is delivered to A-Shell by setting DNVALID.*

[bit13] FATIRQ: Fatal Interrupt Status

bit	Description
0	Fatal interrupt not active
1	Fatal interrupt active, triggered by ARHn_CHCTRL:FATAL

[bit12] CRCERR: CRC Error Status in Upstream Data

bit	Description
0	No CRC error in upstream data (inbound)
1	Indicates occurrence of CRC error in upstream data (inbound)

[bit11] CRCTOUT: CRC Timeout Status in Upstream Data

bit	Description
0	No CRC error in upstream data (inbound)
1	Indicates occurrence of CRC timeout in upstream data (inbound)

[bit10] PERROR: Protocol Error Status

bit	Description
0	No occurrence of protocol error
1	Indicates occurrence of protocol error

[bit9] READY: A-Shell Ready Status

bit	Description
0	Indicates that A-Shell is not ready to accept outbound transactions
1	Indicates that A-Shell is ready to accept outbound transactions

[bit8] REMOTERST: Remote A-Shell Restart Status

bit	Description
0	Indicates no restart (transaction realignment procedure) of remote A-Shell is invoked
1	Indicates a restart of remote A-Shell was performed

[bit7] UPVALIDCL: Clear Upstream Data Valid Status

bit	Description
0	No effect. Read to this register always reads '0'
1	Clears the upstream data valid status

[bit6] DNVALIDST: Set Downstream Data Valid Status

bit	Description
0	No effect. Read to this register always reads '0'
1	Sets the downstream data valid status

[bit5] FATIRQCL: Clear Fatal IRQ Interrupt Status

bit	Description
0	No effect. Read to this register always reads '0'
1	Clears the fatal IRQ interrupt status bit ARHn_CHCTRL:FATIRQ

[bit4] CRCERRCL: Clear CRC Error in Upstream Data Status

bit	Description
0	No effect. Read to this register always reads '0'
1	Clears the CRC error in upstream data (inbound) status bit ARHn_CHCTRL:CRCERR

[bit3] CRCTOUTCL: Clear CRC Timeout in Upstream Data Status

bit	Description
0	No effect. Read to this register always reads '0'
1	Clears the CRC timeout in upstream data (inbound) status bit ARHn_CHCTRL:CRCTOUT

[bit2] PERRORCL: Clear Protocol Error Status

bit	Description
0	No effect. Read to this register always reads '0'
1	Clears the protocol error status bit ARHn_CHCTRL:PERROR

[bit1] READYCL: Clear A-Shell Ready Status

bit	Description
0	No effect. Read to this register always reads '0'
1	Clears the A-Shell ready status bit ARHn_CHCTRL:READY

[bit0] REMOTERSTCL: Clear Remote A-Shell Restart Status

bit	Description
0	No effect. Read to this register always reads '0'
1	Clears the remote A-Shell restart status bit ARHn_CHCTRL:REMOTERST

2.1.3. ARH Channel Control Register (ARHn_CHCTRL1 - Channel 1)

ARH Channel Control Register can be used by software to program the channel.
ARHn_CHCTRL1 register is used for channel 1. This register can be written when ARHn_RHCTRL:LST = '0'.

■ ARH Channel Control Register (ARHn_CHCTRL1)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	BYPASS	FATEIN	Reserved	Reserved	Reserved	TXCFG	RSTRTA	INITRH
ACCESS_TYPE	R/W	R/W	R0,WX	R0,WX	R0,WX	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	1	1	1

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved	Reserved	PLLGOOD	UPHSK	DNHSK	FATAL	UPRDY	CONNECTED
ACCESS_TYPE	R0,WX	R0,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	UPVALID	DNVALID	FATIRQ	CRCERR	CRCTOUT	PERROR	READY	REMOTERST
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	UPVALIDCL	DNVALIDST	FATIRQCL	CRCERRCL	CRCTOUTCL	PERRORCL	READYCL	REMOTERSTCL
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] BYPASS: BYPASS Mode

In bypass mode, Transaction Buffer 0 (for A-Shell 0) is used for downstream data (outbound) and Transaction Buffer 1 (for A-Shell 0) is used for upstream data (inbound).
Transaction Buffer 2 (for A-Shell 1) is used for downstream data (outbound) and Transaction Buffer 3 (for A-Shell 1) is used for upstream data (inbound).

bit	Description
0	Remote Handler active
1	Remote Handler inactive

Note:

- *Valid written data in Transaction Buffer 0 or 2 is delivered to A-Shell by setting DNVALID.*
- *Valid received data in Transaction Buffer 1 or 3 from A-Shell is marked by setting UPVALID.*

[bit30] FATEIN: FATAL Interrupt Enable

bit	Description
0	Disables FATAL interrupt
1	Enables FATAL interrupt

[bit29:27] Reserved: Reserved bits

[bit26] TXCFG: A-Shell Configuration

bit	Description
0	A-Shell and PHY running (write on ARHn_APCFG registers is disabled)
1	Enables A-Shell and PHY configuration (write on ARHn_APCFG registers is enabled which in turn configures A-Shell or PHY)

[bit25] RSTRTA: A-Shell Restart

bit	Description
0	A-Shell running
1	A-Shell initialization A-Shell stops operation, clears all state and history information then restart only when RSTRTA = '0'.

[bit24] INITRH: Remote Handler Initialization

bit	Description
0	Remote Handler running
1	Remote Handler initialization (no change of TB* and TF* registers)

Note:

- *ARHn_TBCTRL00:PENDING requests (set while INITRH == 1) is started with INITRH == 0.*

[bit23:22] Reserved: Reserved bits

[bit21] PLLGOOD: APIX® PHY PLL Lock Status

Lock signal is synchronized to reference clock and then reflected in this register bit.
When synchronization clock is not available, this bit is not updated.

bit	Description
0	APIX® PHY PLL is unlocked
1	APIX® PHY PLL is locked

Note:

- *For devices which do not contain internal APIX® PHY then this bit is always be read 1. For APIX® PHY availability refer to device specific datasheet.*

[bit20] UPHSK: Upstream Handshake

bit	Description
0	Indicates inbound handshake is in progress, or could not be performed
1	Indicates inbound handshake is performed

[bit19] DNHSK: Downstream Handshake

bit	Description
0	Indicates outbound handshake is in progress, or could not be performed
1	Indicates outbound handshake is performed

[bit18] FATAL: Fatal from A-Shell

bit	Description
0	Indicates that A-Shell is running without any severe errors
1	Indicates that A-Shell has encountered conditions where A-Shell cannot continue to deliver and receive transactions. FATAL is only one clock cycle active.

[bit17] UPRDY: Upstream Serial Channel Operation

bit	Description
0	Indicates that upstream serial channel (APIX® PHY) is not operational
1	Indicates that upstream serial channel (APIX® PHY) is operational

[bit16] CONNECTED: Connection Established

bit	Description
0	Indicates connection to remote APIX® is yet to be established
1	Indicates connection to remote APIX® is established

[bit15] UPVALID: Upstream Data Valid Status

bit	Description
0	Cleared by SW by setting ARHn_CHCTRL1:UPVALIDCL
1	Set by HW to mark upstream data as valid(ap_data_out_valid)

Note:

- *The bypass mode allows direct access and actions on downstream and upstream data and enable*
  ports of the A-Shell (without having the Remote Handler sequencer involved).
  In bypass mode Transaction Buffer 0 (for A-Shell 0) is used for downstream data (outbound) and Transaction Buffer 1 (for A-Shell 0) is used for upstream data (inbound).
  In bypass mode Transaction Buffer 2 (for A-Shell 1) is used for downstream data (outbound) and Transaction Buffer 3 (for A-Shell 1) is used for upstream data (inbound).
  Valid received data in Transaction Buffer 1 or 3 from A-Shell is marked by setting UPVALID.

[bit14] DNVALID: Downstream Data Valid Status

bit	Description
0	Cleared by HW after successful transfer to A-Shell
1	Set by SW by writing '1' to ARHn_CHCTRL1:DNVALIDST to mark downstream data as valid (ap_data_in_valid)

Note:

- *The bypass mode allows direct access and actions on downstream and upstream data and enable ports of the A-Shell (without having the Remote Handler sequencer involved).*
  In bypass mode Transaction Buffer 0 (for A-Shell 0) is used for downstream data (outbound) and Transaction Buffer 1 (for A-Shell 0) is used for upstream data (inbound).
- *In bypass mode Transaction Buffer 2 (for A-Shell 1) is used for downstream data (outbound) and Transaction Buffer 3 (for A-Shell 1) is used for upstream data (inbound).*
- *Valid written data in Transaction Buffer 0 or 2 is delivered to A-Shell by setting DNVALID.*

[bit13] FATIRQ: Fatal Interrupt Status

bit	Description
0	Fatal interrupt not active
1	Fatal interrupt active, triggered by ARHn_CHCTRL:FATAL

[bit12] CRCERR: CRC Error Status in Upstream Data

bit	Description
0	No CRC error in upstream data (inbound)
1	Indicates occurrence of CRC error in upstream data (inbound)

[bit11] CRCTOUT: CRC Timeout Status in Upstream Data

bit	Description
0	No CRC error in upstream data (inbound)
1	Indicates occurrence of CRC timeout in upstream data (inbound)

[bit10] PERROR: Protocol Error Status

bit	Description
0	No occurrence of protocol error
1	Indicates occurrence of protocol error

[bit9] READY: A-Shell Ready Status

bit	Description
0	Indicates that A-Shell is not ready to accept outbound transactions
1	Indicates that A-Shell is ready to accept outbound transactions

[bit8] REMOTERST: Remote A-Shell Restart Status

bit	Description
0	Indicates no restart (transaction realignment procedure) of remote A-Shell is invoked
1	Indicates a restart of remote A-Shell was performed

[bit7] UPVALIDCL: Clear Upstream Data Valid Status

bit	Description
0	No effect. Read to this register always reads '0'
1	Clears the upstream data valid status

[bit6] DNVALIDST: Set Downstream Data Valid Status

bit	Description
0	No effect. Read to this register always reads '0'
1	Sets the downstream data valid status

[bit5] FATIRQCL: Clear Fatal IRQ Interrupt Status

bit	Description
0	No effect. Read to this register always reads '0'
1	Clears the fatal IRQ interrupt status bit ARHn_CHCTRL:FATIRQ

[bit4] CRCERRCL: Clear CRC Error in Upstream Data Status

bit	Description
0	No effect. Read to this register always reads '0'
1	Clears the CRC error in upstream data (inbound) status bit ARHn_CHCTRL:CRCERR

[bit3] CRCTOUTCL: Clear CRC Timeout in Upstream Data Status

bit	Description
0	No effect. Read to this register always reads '0'
1	Clears the CRC timeout in upstream data (inbound) status bit ARHn_CHCTRL:CRCTOUT

[bit2] PERRORCL: Clear Protocol Error Status

bit	Description
0	No effect. Read to this register always reads '0'
1	Clears the protocol error status bit ARHn_CHCTRL:PERROR

[bit1] READYCL: Clear A-Shell Ready Status

bit	Description
0	No effect. Read to this register always reads '0'
1	Clears the A-Shell ready status bit ARHn_CHCTRL:READY

[bit0] REMOTERSTCL: Clear Remote A-Shell Restart Status

bit	Description
0	No effect. Read to this register always reads '0'
1	Clears the remote A-Shell restart status bit ARHn_CHCTRL:REMOTERST

2.1.4. ARH Channel Status Register (ARHn_CHSTAT0 - Channel 0, ARHn_CHSTAT1 - Channel 1)

The software can use this register to read the A-Shell channel status. It contains various status information like upstream CRC error occurrence, upstream synchronization lost, and PLL synchronization lost. Register ARHn_CHSTAT0 for channel 0 is explained here.

■ ARH Channel Status Register (ARHn_CHSTAT0) Channel 0

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	UPCRC[7]	UPCRC[6]	UPCRC[5]	UPCRC[4]	UPCRC[3]	UPCRC[2]	UPCRC[1]	UPCRC[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

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

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	UPSYNC[7]	UPSYNC[6]	UPSYNC[5]	UPSYNC[4]	UPSYNC[3]	UPSYNC[2]	UPSYNC[1]	UPSYNC[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

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	PLLBAD[7]	PLLBAD[6]	PLLBAD[5]	PLLBAD[4]	PLLBAD[3]	PLLBAD[2]	PLLBAD[1]	PLLBAD[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

[bit31:24] UPCRC[7:0]: CRC Errors
Inbound CRC errors. The CRC errors occurred on the data transmission from remote.

[bit23:16] Reserved: Reserved bits

[bit15:8] UPSYNC[7:0]: Synchronization Lost
Indicates number of detected synchronization losses of upstream serial channel (APIX® PHY).

[bit7:0] PLLBAD[7:0]: PLL Synchronization Lost
Indicates number of detected PLL synchronization losses.

2.1.5. ARH Channel Watchdog Control Register (ARHn_CHWDGCTL0 - Channel 0, ARHn_CHWDGCTL1 - Channel 1)

This register provides Watchdog Timer control. Using this register generation of Watchdog interrupt can be controlled. Register ARHn_CHWDGCTL0 for channel 0 is explained here.

■ ARH Channel Watchdog Control Register (ARHn_CHWDGCTL0) Channel 0

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	WDTXIEN	WDRXIEN	Reserved	Reserved	WTTX[1]	WTTX[0]	WTRX[1]	WTRX[0]
ACCESS_TYPE	R/W	R/W	R0,WX	R0,WX	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	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

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	WDTXIRQ3	WDTXIRQ2	WDTXIRQ1	WDTXIRQ0	WDRXIRQ3	WDRXIRQ2	WDRXIRQ1	WDRXIRQ0
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	WDTXIRQ3CL	WDTXIRQ2CL	WDTXIRQ1CL	WDTXIRQ0CL	WDRXIRQ3CL	WDRXIRQ2CL	WDRXIRQ1CL	WDRXIRQ0CL
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] WDTXIEN: TX Watchdog Timer Interrupt Enable

bit	Description
0	Watchdog interrupt for TX is disabled
1	Watchdog interrupt for TX is enabled

[bit30] WDRXIEN: RX Watchdog Timer Interrupt Enable

bit	Description
0	Watchdog interrupt for RX is disabled
1	Watchdog interrupt for RX is enabled

[bit29:28] Reserved: Reserved bits

[bit27:26] WTTX[1:0]: TX Watchdog Timer Select

bit	Description
00	Select ARHn_CHWDGCTL0:WDTXIRQ0
01	Select ARHn_CHWDGCTL0:WDTXIRQ1
10	Select ARHn_CHWDGCTL0:WDTXIRQ2
11	Select ARHn_CHWDGCTL0:WDTXIRQ3

[bit25:24] WTRX[1:0]: RX Watchdog Timer Select

bit	Description
00	Select ARHn_CHWDGCTL0:WDRXIRQ0
01	Select ARHn_CHWDGCTL0:WDRXIRQ1
10	Select ARHn_CHWDGCTL0:WDRXIRQ2
11	Select ARHn_CHWDGCTL0:WDRXIRQ3

[bit23:16] Reserved: Reserved bits

[bit15] WDTXIRQ3: TX Watchdog Interrupt 3

bit	Description
0	No watchdog interrupt for TX generated when watchdog counter value = 219
1	Watchdog interrupt for TX generated when watchdog counter value = 219

[bit14] WDTXIRQ2: TX Watchdog Interrupt 2

bit	Description
0	No watchdog interrupt for TX generated when watchdog counter value = 216
1	Watchdog interrupt for TX generated when watchdog counter value = 216

[bit13] WDTXIRQ1: TX Watchdog Interrupt 1

bit	Description
0	No watchdog interrupt for TX generated when watchdog counter value = 214
1	Watchdog interrupt for TX generated when watchdog counter value = 214

[bit12] WDTXIRQ0: TX Watchdog Interrupt 0

bit	Description
0	No watchdog interrupt for TX generated when watchdog counter value = 213
1	Watchdog interrupt for TX generated when watchdog counter value = 213

[bit11] WDRXIRQ3: RX Watchdog Interrupt 3

bit	Description
0	No watchdog interrupt for RX generated when watchdog counter value = 219
1	Watchdog interrupt for RX generated when watchdog counter value = 219

[bit10] WDRXIRQ2: RX Watchdog Interrupt 2

bit	Description
0	No watchdog interrupt for RX generated when watchdog counter value = 218
1	Watchdog interrupt for RX generated when watchdog counter value = 218

[bit9] WDRXIRQ1: RX Watchdog Interrupt 1

bit	Description
0	No watchdog interrupt for RX generated when watchdog counter value = 217
1	Watchdog interrupt for RX generated when watchdog counter value = 217

[bit8] WDRXIRQ0: RX Watchdog Interrupt 0

bit	Description
0	No watchdog interrupt for RX generated when watchdog counter value = 216
1	Watchdog interrupt for RX generated when watchdog counter value = 216

[bit7] WDTXIRQ3CL: Clear WDTXIRQ3

bit	Description
0	No effect. Read to this register always reads '0'
1	Clears ARHn_CHWDGCTL0:WDTXIRQ3

[bit6] WDTXIRQ2CL: Clear WDTXIRQ2

bit	Description
0	No effect. Read to this register always reads '0'
1	Clears ARHn_CHWDGCTL0:WDTXIRQ2

[bit5] WDTXIRQ1CL: Clear WDTXIRQ1

bit	Description
0	No effect. Read to this register always reads '0'
1	Clears ARHn_CHWDGCTL0:WDTXIRQ1

[bit4] WDTXIRQ0CL: Clear WDTXIRQ0

bit	Description
0	No effect. Read to this register always reads '0'
1	Clears ARHn_CHWDGCTL0:WDTXIRQ0

[bit3] WDRXIRQ3CL: Clear WDRXIRQ3

bit	Description
0	No effect. Read to this register always reads '0'
1	Clears ARHn_CHWDGCTL0:WDTXIRQ3

[bit2] WDRXIRQ2CL: Clear WDRXIRQ2

bit	Description
0	No effect. Read to this register always reads '0'
1	Clears ARHn_CHWDGCTL0:WDTXIRQ2

[bit1] WDRXIRQ1CL: Clear WDRXIRQ1

bit	Description
0	No effect. Read to this register always reads '0'
1	Clears ARHn_CHWDGCTL0:WDTXIRQ1

[bit0] WDRXIRQ0CL: Clear WDRXIRQ0

bit	Description
0	No effect. Read to this register always reads '0'
1	Clears ARHn_CHWDGCTL0:WDTXIRQ0

2.1.6. ARH Watchdog Counter Register (ARHn_CHWDGCNT0 - Channel 0, ARHn_CHWDGCNT1 - Channel 1)

The software can use this register to read the status of internal Free Running Timer.
Register ARHn_CHWDGCNT0 for channel 0 is explained here.

■ ARH Watchdog Counter Register (ARHn_CHWDGCNT0) Channel 0

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

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

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	CNT[15]	CNT[14]	CNT[13]	CNT[12]	CNT[11]	CNT[10]	CNT[9]	CNT[8]
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	x	x	x	x	x	x	x	x

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	CNT[7]	CNT[6]	CNT[5]	CNT[4]	CNT[3]	CNT[2]	CNT[1]	CNT[0]
ACCESS_TYPE	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	x	x	x	x	x	x	x	x

[bit31:16] Reserved: Reserved bits

[bit15:0] CNT[15:0]: Free Running Timer
Reading this register provides the upper 16 bits of the 20 bits Free Running Timer.

2.1.7. ARH Transaction Buffer Control Register (ARHn_TBCTRL0 to 15)

The software can use these registers (ARHn_TBCTRL0 to ARHn_TBCTRL15) to program the Transaction Buffer. This register can be written when the corresponding Transaction Buffer state is ARHn_TBCTRL0:ACTIVE = '0', ARHn_TBCTRL0:WAITING = '0', and ARHn_TBCTRL0:PENDING = '0'. Register for ARHn_TBCTRL0 is explained here.

■ ARH Transaction Buffer Control Register (ARHn_TBCTRL0)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	TBCH	TBAINC	TBACT	TBIMD	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R/W	R/W	R/W	R/W	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	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	TBDEN	TBIEN
ACCESS_TYPE	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	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	Reserved	ACTIVE	UNLOCKED	CANCELED	WAITING	PENDING	TBIRQ
ACCESS_TYPE	R0,WX	R0,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	Reserved	Reserved	Reserved	Reserved	Reserved	UNLOCKEDCL	CANCELEDCL	TBIRQCL
ACCESS_TYPE	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,W	R0,W	R0,W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31] TBCH: Transaction Buffer Channel

bit	Description
0	Transaction Buffer assigned to channel 0
1	Transaction Buffer assigned to channel 1

–

[bit30] TBAINC: Transaction Buffer Address Increment

bit	Description
0	Transaction Buffer Address Increment is disabled
1	Transaction Buffer Address Increment is enabled

Note:

- *Increments address after WR access of TFDATA and transmission of TF.*
- *Increments address after reception of requested TF and RD access to TFDATA.*

[bit29] TBACT: Transaction Buffer Activated

bit	Description
0	Transaction Buffer is activated by WR access to ARHn_RHCTRL:TBNO
1	Transaction Buffer is activated by WR access to ARHn_RHCTRL:TBNO or TFDATA (RD or WR)

[bit28] TBIMD: Transaction Buffer Interrupt Mode

bit	Description
0	Transaction Buffer interrupt on TB idle (after transaction send)
1	Transaction Buffer interrupt on TB valid (after read request data reception)

[bit27:18] Reserved: Reserved bits

[bit17] TBDEN: Transaction Buffer DMA Enable

bit	Description
0	Transaction Buffer DMA is disabled
1	Transaction Buffer DMA is enabled

[bit16] TBIEN: Transaction Buffer Interrupt Enable

bit	Description
0	Transaction Buffer interrupt is disabled
1	Transaction Buffer interrupt is enabled

[bit15:14] Reserved: Reserved bits

[bit13] ACTIVE: Transaction Buffer request Active

bit	Description
0	No active data in Transaction Buffer
1	Active data in Transaction Buffer (delivery to A-Shell requested)

[bit12] UNLOCKED: Transaction Buffer Unlocked

bit	Description
0	No last action on this buffer
1	Last action on this Transaction Buffer was unlock

[bit11] CANCELED: Transaction Buffer Request Cancelled

bit	Description
0	No last action on this buffer
1	Last action on this Transaction Buffer was unlock

[bit10] WAITING: Transaction Buffer request Waiting

bit	Description
0	Not waiting for requested data
1	Transaction Buffer waiting for requested data

Note:

- *Waiting is cleared at reception of requested data in buffer.*

[bit9] PENDING: Transaction Buffer request Pending

bit	Description
0	No pending data in Transaction Buffer
1	Pending data in Transaction Buffer (not yet requested delivery to A-Shell)

[bit8] TBIRQ: Transaction Buffer Interrupt

bit	Description
0	Transaction Buffer interrupt not active
1	Transaction Buffer interrupt active. TBIRQ interrupt generated due to sending WR msg or reception of RD msg.

[bit7:3] Reserved: Reserved bits

[bit2] UNLOCKEDCL: Clear Transaction Buffer Unlocked

bit	Description
0	No effect. Read to this register always reads '0'
1	Clears the Transaction Buffer unlocked status

[bit1] CANCELEDCL: Clear Transaction Buffer Cancelled

bit	Description
0	No effect. Read to this register always reads '0'
1	Clears the Transaction Buffer cancelled status

[bit0] TBIRQCL: Clear Transaction Buffer Interrupt

bit	Description
0	No effect. Read to this register always reads '0'
1	Clears Transaction Buffer interrupt ARHn_TBCTRL0:TBIRQ

Figure 34-2: Example Showing Transaction Buffer States for Frame Transmission

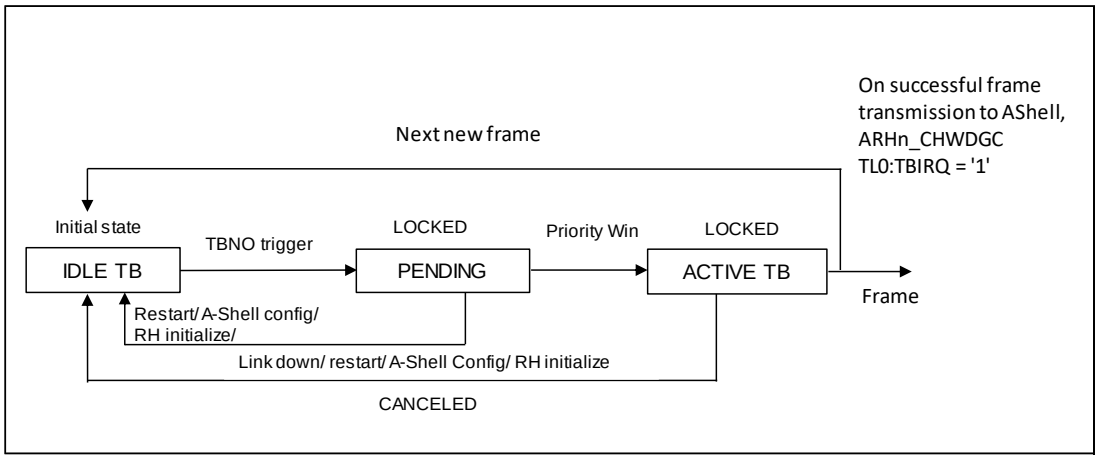


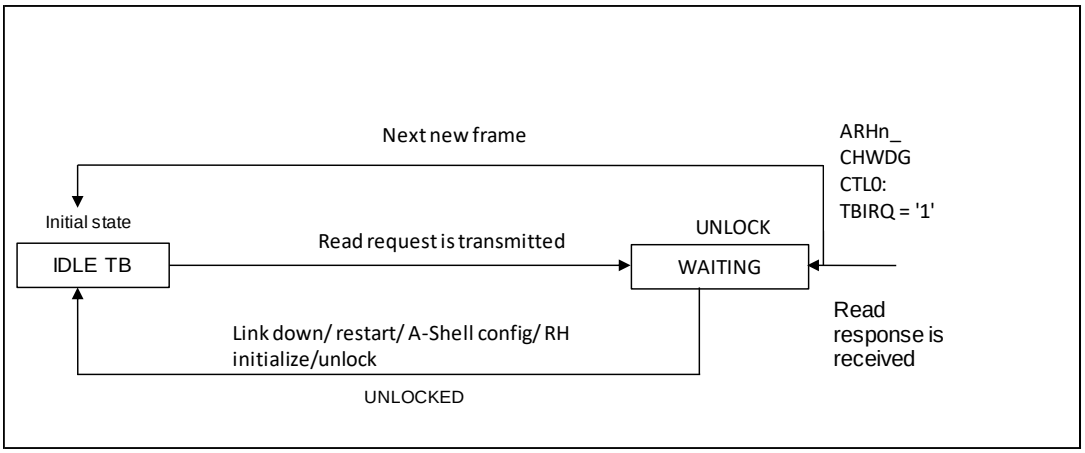
Figure 34-2 explains Transaction Buffer states during frame transmission.

Writing ARHn_RHCTRL:TBNO Transaction Buffer when ARHn_RHCTRL:UNLOCK = '0' and ARHn_RHCTRL:CANCEL = '0' moves the Transaction Buffer to pending/active state based on priority and if no transmission is pending.

A Transaction Buffer in pending/active state will revert back to idle state upon receipt of a ARHn_RHCTRL:CANCEL request or RESTART request.

A Transaction Buffer in active state will revert back to idle state after transmission of frame.

Figure 34-3: Example Showing Transaction Buffer States for Frame Reception



Note:

- *Only ARHn_TBCTRL0:WAITING Transaction Buffers can be UNLOCKED*
- *Only ARHn_TBCTRL0:PENDING or ARHn_TBCTRL0:ACTIVE Transaction Buffers can be CANCELLED.*
- *While in ARHn_TBCTRL0:PENDING or ARHn_TBCTRL0:ACTIVE states, the Transaction Buffer is LOCKED.*

Figure 34-3 explains Transaction Buffer states during frame reception.

A Transaction Buffer in idle state goes to waiting state on read request submission to remote device.

A Transaction Buffer in waiting state will revert back to idle upon ARHn_RHCTRL:UNLOCK, RESTART, or ARHn_CHCTRL0:INITRH request.

A Transaction Buffer in waiting state will revert back to idle state on reception of read response from remote device.

2.1.8. ARH Transaction Buffer Interrupt Register (ARHn_TBIRQ)

The software can read this register to get the status of Transaction Buffer interrupt of all the Transaction Buffers.

■ ARH Transaction Buffer Interrupt Register (ARHn_TBIRQ)

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	TBIRQ[15]	TBIRQ[14]	TBIRQ[13]	TBIRQ[12]	TBIRQ[11]	TBIRQ[10]	TBIRQ[9]	TBIRQ[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

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	TBIRQ[7]	TBIRQ[6]	TBIRQ[5]	TBIRQ[4]	TBIRQ[3]	TBIRQ[2]	TBIRQ[1]	TBIRQ[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

[bit15:0] TBIRQ: Transaction Buffer Interrupt

Read-only TB interrupt flag (ARHn_TBCTRL00 to 15:TBIRQ) generated when corresponding interrupt enable
is high (ARHn_TBCTRL00 to 15:TBIEN == '1') ARH.

bit	Description
0	Transaction Buffer interrupt is not active
1	Transaction Buffer interrupt is active.

Note:

- *TBIRQ can or is cleared by the following events:*
- *Cleared by SW on write access to ARHn_TBCTRL00 to 15:TBIRQCL flag with data '1'.*
- *Cleared by HW if ARHn_TBCTRL00 to 15:TBACT == '1' and read or write access to ARHn_TFDATA register (both CPU or DMA).*

2.1.9. ARH Transaction Buffer Index Register (ARHn_TBIDX0 - Channel 0, ARHn_TBIDX1 - Channel 1)

The software can use this register to find the index of currently active Transaction Buffer with active ARHn_TBCTRL0:TBIRQ and ARHn_TBCTRL0:TBIEN interrupt. The ARHn_TBIDX0 register is for channel 0, while ARHn_TBIDX1 register is for channel 1. Register ARHn_TBIDX0 for channel 0 is explained here.

■ ARH Transaction Buffer Index Register (ARHn_TBIDX0) Channel 0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	Reserved	Reserved	Reserved	TBIDX[4]	TBIDX[3]	TBIDX[2]	TBIDX[1]	TBIDX[0]
ACCESS_TYPE	R0,WX	R0,WX	R0,WX	R,WX	R,WX	R,WX	R,WX	R,WX
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit7:5] Reserved: Reserved bits

[bit4:0] TBIDX[4:0]: Transaction Buffer Index

These bits provide read-only index+1 of highest priority buffer with active (ARHn_TBCTRL00 to 15:TBIRQ == '1') and enabled (ARHn_TBCTRL00 to 15:TBIEN == '1') interrupt. If TBIDX == '0' indicates no interrupt for corresponding channel.

2.1.10. ARH Transaction Frame Control Register (ARHn_TFCTRL00 to 15)

The software can use these registers (ARHn_TFCTRL0 to ARHn_TFCTRL15) to program the Transaction Frame. These registers can be written when the corresponding Transaction Buffer state is ARHn_TBCTRL0:ACTIVE = '0', ARHn_TBCTRL0:WAITING = '0', and ARHn_TBCTRL0:PENDING = '0'. Register ARHn_TFCTRL0 is explained here.

■ ARH Transaction Frame Control Register (ARHn_TFCTRL0)

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	TFDASWP	TFAINV	Reserved	ERROR	SZ[1]	SZ[0]	OAEN	RW
ACCESS_TYPE	R/W	R/W	R0,WX	R,WX	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit7] TFDASWP: Transaction Frame Data Swap

Transaction frame data swap bit. For more details refer to Figure 34-5

bit	Description
0	Normal mode
1	Byte swapping

–

[bit6] TFAINV: Transaction Frame Address Inversion

Transaction frame address inversion bit. For more details refer to Figure 34-6

bit	Description
0	Normal mode
1	Address inversion

[bit5] Reserved: Reserved bits

[bit4] ERROR: Remote Handler RX Error

bit	Description
0	Normal operation
1	Remote Handler RX bus error occurred

[bit3:2] SZ[1:0]: Size

bit	Description
00	Byte
01	Half word
10	Word
11	Reserved

[bit1] OAEN: Offset Address Enable

bit	Description
0	Offset address is disabled
1	Offset address is enabled

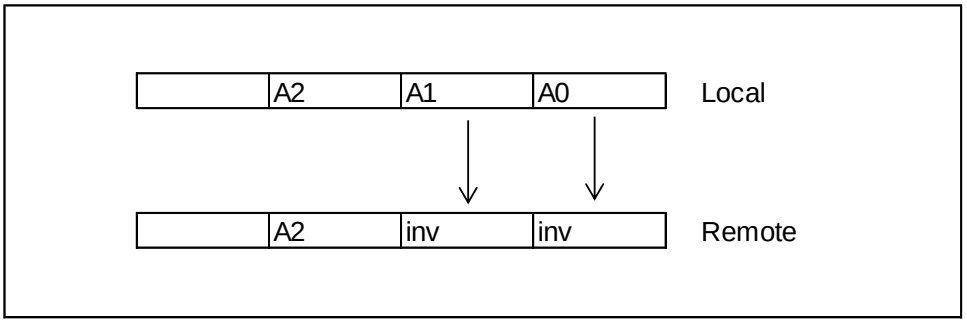
[bit0] RW: Read or Write

bit	Description
0	Read message to be sent
1	Write message to be sent

Note:

- *In address inversion mode the two least significant bits of the address are inverted.*

Figure 34-4: Example Showing Address Inversion (ARHn_TFCTRL00 to 15:TFAINV = '1')



– *In swapping mode depending on the configured size the following byte swapping of the data is as shown in Figure 34-5.*

Figure 34-5: ARH TFDATA Byte Swapping Depending on Swapping Mode and Configured Size

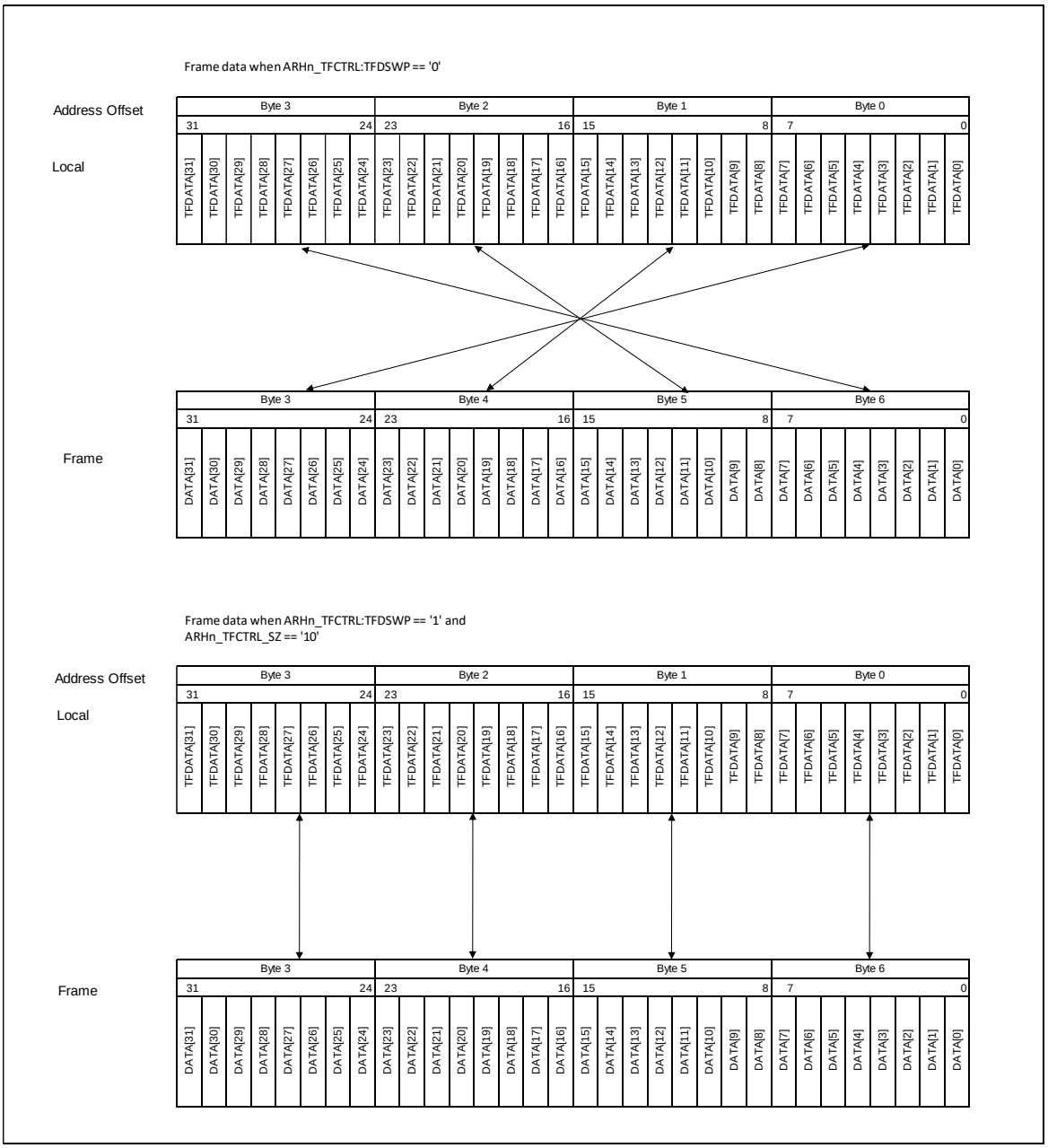
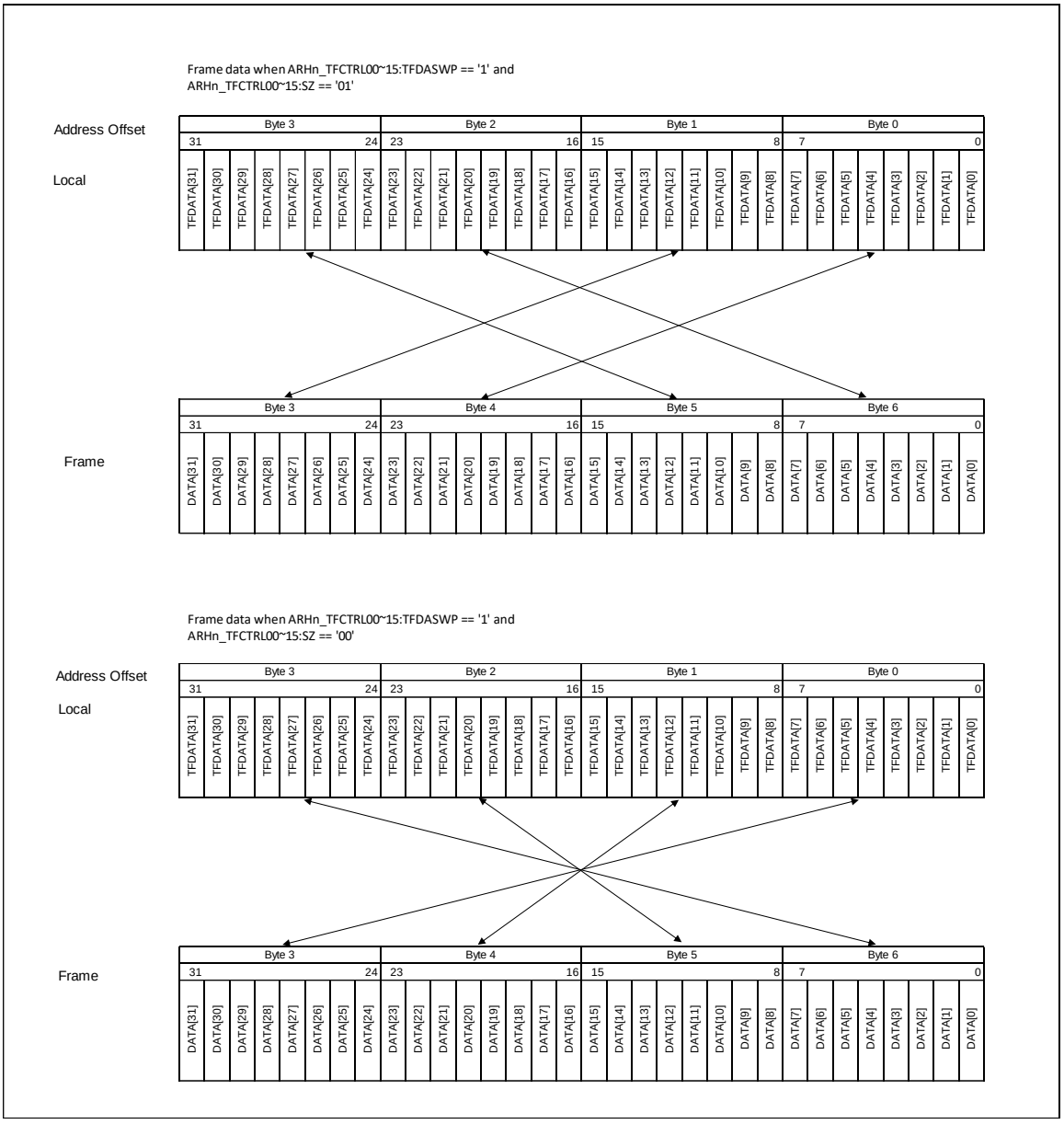


Figure 34-6: ARH TFDATA Byte Swapping Depending on Swapping Mode and Configured Size



2.1.11. ARH Transaction Frame Index Register (ARHn_TFIDX0 to 15)

The software can use these registers (ARHn_TFIDX0 to ARHn_TFIDX15) to program the index of the Transaction Frame.
These registers can be written when the corresponding Transaction Buffer state is ARHn_TBCTRL0:ACTIVE = '0', ARHn_TBCTRL0:WAITING = '0', and ARHn_TBCTRL0:PENDING = '0'.
Register ARHn_TFIDX0 is explained here.

■ ARH Transaction Frame Index Register (ARHn_TFIDX0)

IT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	IDX[7]	IDX[6]	IDX[5]	IDX[4]	IDX[3]	IDX[2]	IDX[1]	IDX[0]
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] IDX: Transaction Frame Index
Any number between 0 and 255. Index is used for read request. Received data from a read request is stored in an active Transaction Buffer with matching index. If there is no active Transaction Buffer with matching index (e.g. by using unlock), the received data is discarded.

2.1.12. ARH Transaction Frame Address Register (ARHn_TFADDR0 to 15)

The software can use these registers (ARHn_TFADDR0 to ARHn_TFADDR15) to program the address of Transaction Frame. This register can be written when the corresponding Transaction Buffer state is ARHn_TBCTRL0:ACTIVE = '0', ARHn_TBCTRL0:WAITING = '0', and ARHn_TBCTRL0:PENDING = '0'. Register ARHn_TFADDR0 is explained here.

- ARH Transaction Frame Index Register (ARHn_TFIDX0)

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

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	Reserved	Reserved	Reserved	Reserved	ADDR[19]	ADDR[18]	ADDR[17]	ADDR[16]
ACCESS_TYPE	R0,WX	R0,WX	R0,WX	R0,WX	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	ADDR[15]	ADDR[14]	ADDR[13]	ADDR[12]	ADDR[11]	ADDR[10]	ADDR[9]	ADDR[8]
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	ADDR[7]	ADDR[6]	ADDR[5]	ADDR[4]	ADDR[3]	ADDR[2]	ADDR[1]	ADDR[0]
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

[bit31:24] Reserved: Reserved bits

[bit23:20] Reserved: Reserved bits

[bit19:0] ADDR: Transaction Frame Address

2.1.13. ARH Transaction Frame Data Register (ARHn_TFDATA0 to 15)

The software can use these registers (ARHn_TFDATA0 to ARHn_TFDATA15) to program the Transaction Frame data of Transaction Frame. This register can be written when the corresponding Transaction Buffer state is ARHn_TBCTRL0:ACTIVE = '0', ARHn_TBCTRL0:WAITING = '0', and ARHn_TBCTRL0:PENDING = '0'.
Register AHn_TFDATA0 is explained here.

■ ARH Transaction Frame Data Register (ARHn_TFDATA0)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	TFDATA[31]	TFDATA[30]	TFDATA[29]	TFDATA[28]	TFDATA[27]	TFDATA[26]	TFDATA[25]	TFDATA[24]
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	TFDATA[23]	TFDATA[22]	TFDATA[21]	TFDATA[20]	TFDATA[19]	TFDATA[18]	TFDATA[17]	TFDATA[16]
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	TFDATA[15]	TFDATA[14]	TFDATA[13]	TFDATA[12]	TFDATA[11]	TFDATA[10]	TFDATA[9]	TFDATA[8]
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	TFDATA[7]	TFDATA[6]	TFDATA[5]	TFDATA[4]	TFDATA[3]	TFDATA[2]	TFDATA[1]	TFDATA[0]
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] TFDATA: Transaction Frame Data

Notes:

- ARHn_TFDATA0 to 15 Registers Read Conditions:

1. For a DAP Controller master, read is allowed if there is no PPU violation.
2. For a AHB master other than DAP Controller,
   A)If ARHn_TBCTRLn:TBACT = ‘1’ and if the transaction buffer state is ARHn_TBCTRLn:ACTIVE = ‘1’ or ARHn_TBCTRLn:WAITING = ‘1’ or ARHn_TBCTRLn:PENDING = ‘1’, error response is generated in both privileged as well as non-privileged mode.
   In non-privileged mode, if ARHn_TBCTRLn:TBACT = ‘1’ and if the transaction buffer state is ARHn_TBCTRLn:ACTIVE = ‘0’ and ARHn_TBCTRLn:WAITING = ‘0’ and ARHn_TBCTRLn:PENDING = ‘0’, read is allowed only if PPU access = ‘1’.
   For all other conditions, read is allowed if there is no PPU violation.

- ARHn_TFDATA0 to 15 Registers Write Conditions:

1. Write is not allowed if the transaction buffer state is ARHn_TBCTRLn:ACTIVE = ‘1’ or ARHn_TBCTRLn:WAITING = ‘1’ or ARHn_TBCTRLn:PENDING = ‘1’ in both privileged as well as non-privileged mode.
2. For all other conditions in non-privileged mode, write is allowed if there is no PPU violation.

- Transaction Buffer Trigger Conditions:

1. Read transaction from a DAP Controller master does not trigger transaction buffer.
2. For a read transaction from a AHB master other than DAP Controller or a write transaction from any AHB master,
   In privileged mode, transaction buffer is triggered only if ARHn_TBCTRLn:TBACT = ‘1’ and the transaction buffer state is ARHn_TBCTRLn:ACTIVE = ‘0’ and ARHn_TBCTRLn:WAITING = ‘0’ and ARHn_TBCTRLn:PENDING = ‘0’.
   In non-privileged mode, transaction buffer is triggered only if ARHn_TBCTRLn:TBACT = ‘1’ and the transaction buffer state is ARHn_TBCTRLn:ACTIVE = ‘0’ and ARHn_TBCTRLn:WAITING = ‘0’ and ARHn_TBCTRLn:PENDING = ‘0’ and PPU access = ‘1’.

2.1.14. ARH Event Control Register (ARHn_EVCTRL)

The software can use this register to program the Event Buffer. This register can be written when ARHn_RHCTRL:LST = '0'.

■ ARH Event Control Register (ARHn_EVCTRL)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	MODE	FRST	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R/W	R0,W	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	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

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	STATUS[7]	STATUS[6]	STATUS[5]	STATUS[4]	STATUS[3]	STATUS[2]	STATUS[1]	STATUS[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

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	LEVEL[7]	LEVEL[6]	LEVEL[5]	LEVEL[4]	LEVEL[3]	LEVEL[2]	LEVEL[1]	LEVEL[0]
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] MODE: FIFO Mode

bit	Description
0	Level mode, on FIFO full condition new events are discarded
1	Ring mode, oldest event is overwritten with latest event (circular buffering)

[bit30] FRST: FIFO Reset

bit	Description
0	FIFO is in normal operation
1	FIFO pointers are reset pulse (set to '0' after 1 cycle)

[bit29:16] Reserved: Reserved bits

[bit15:8] STATUS[7:0]: FIFO Fill Status

Current FIFO filling status read-only, the FIFO fill status ranges from 0 to 128.

[bit7:0] LEVEL[7:0]: FIFO Interrupt Level

FIFO Interrupt Level (128 default). The software can program any value in range of 0 to 128.

2.1.15. ARH Event Interrupt Control Register (ARHn_EVIRQC)

The software can use this register to program the event interrupts. This register provides enable and status information
for the level, overflow, and Event Buffer interrupts.

■ ARH Event Interrupt Control Register (ARHn_EVIRQC)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	LVIEN	OFLIEN	EVIEN	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R/W	R/W	R/W	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	LVIRQ	OFLIRQ	EVIRQ	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R,WX	R,WX	R,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	15	14	13	12	11	10	9	8
BIT_NAME	LVIRQCL	OFLIRQCL	EVIRQCL	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R0,W	R0,W	R0,W	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	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

[bit31] LVIEN: Level Interrupt Enable

bit	Description
0	Level interrupt is disabled
1	Level interrupt is enabled

[bit30] OFLIEN: Event Buffer Overflow Interrupt Enable

bit	Description
0	Event Buffer overflow interrupt not active
1	Event Buffer overflow interrupt active

[bit29] EVIEN: Event Buffer Interrupt Enable

bit	Description
0	Event Buffer interrupt is disabled
1	Event Buffer interrupt is enabled

[bit28:24] Reserved: Reserved bits

[bit23] LVIRQ: Level IRQ

bit	Description
0	Level interrupt not active
1	Level interrupt active (if STATUS >= LEVEL)

[bit22] OFLIRQ: Overflow IRQ

bit	Description
0	Event Buffer overflow interrupt not active
1	Event Buffer interrupt active

[bit21] EVIRQ: Event IRQ

bit	Description
0	Event Buffer interrupt is disabled
1	Event Buffer interrupt is enabled

[bit20:16] Reserved: Reserved bits

[bit15] LVIRQCL: Clear Level IRQ

bit	Description
0	No effect. Read to this register always reads '0'
1	Clears ARHn_EVIRQC:LVIRQ

[bit14] OFLIRQCL: Clear Event Buffer Overflow IRQ

bit	Description
0	No effect. Read to this register always reads '0'
1	Clears ARHn_EVIRQC:OFLIRQ

[bit13] EVIRQCL: Clear Event Buffer Event IRQ

bit	Description
0	No effect. Read to this register always reads '0'
1	Clears ARHn_EVIRQC:EVIRQ

[bit12:0] Reserved: Reserved bits

2.1.16. ARH Event Buffer Register (ARHn_EVBUF0)

The software can use this register to program the index number of event messages. It also provides status of channel number on which event was last received. This register can be written when ARHn_TST:TM = '1'.

■ ARH Event Buffer Register (ARHn_EVBUF0)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	EVCH
ACCESS_TYPE	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	x

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	EVIDX[7]	EVIDX[6]	EVIDX[5]	EVIDX[4]	EVIDX[3]	EVIDX[2]	EVIDX[1]	EVIDX[0]
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	x	x	x	x	x	x	x	x

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

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	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:25] Reserved: Reserved bits

[bit24] EVCH: Event Channel Number

Holds channel number from Remote Handler RX (remote device) event.

[bit23:16] EVIDX[7:0]: Event Index Number

0-255 holds index number from Remote Handler RX event.

[bit15:0] Reserved: Reserved bits

Notes:

- *A read access to ARHn_EVBUF0 triggers a retrieve of the current event message from the Event Buffer FIFO and returns the channel number and event index. Therefore, if ARHn_TST:TM == '0' non-DAP access in non-privileged mode can only be done with full access rights (i.e. when iPPU_ACCESS == '1'), else protection violation fault is triggered. However, if the DAP controller reads this register, the Event Buffer FIFO read pointer is not incremented.*
- *It is recommended to read first ARHn_EVBUF0 and after that ARHn_EVBUF1 registers.*

2.1.17. ARH Event Buffer Register (ARHn_EVBUF1)

The software can use this register to read the 4 byte payload data of event message. This register can be written when ARHn_TST:TM = '1'.

■ ARH Event Buffer Register (ARHn_EVBUF1)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	EVDATA[31]	EVDATA[30]	EVDATA[29]	EVDATA[28]	EVDATA[27]	EVDATA[26]	EVDATA[25]	EVDATA[24]
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	x	x	x	x	x	x	x	x

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	EVDATA[23]	EVDATA[22]	EVDATA[21]	EVDATA[20]	EVDATA[19]	EVDATA[18]	EVDATA[17]	EVDATA[16]
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	x	x	x	x	x	x	x	x

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	EVDATA[15]	EVDATA[14]	EVDATA[13]	EVDATA[12]	EVDATA[11]	EVDATA[10]	EVDATA[9]	EVDATA[8]
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	x	x	x	x	x	x	x	x

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	EVDATA[7]	EVDATA[6]	EVDATA[5]	EVDATA[4]	EVDATA[3]	EVDATA[2]	EVDATA[1]	EVDATA[0]
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	x	x	x	x	x	x	x	x

[bit31:0] EVDATA: Payload Data

The 4 byte payload data. A read access to ARHn_EVBUF1 returns the data part of the event message.

Notes:

- *It is recommended to read first ARHn_EVBUF0 and after that ARHn_EVBUF1 registers.*

2.1.18. ARH APIX® Configuration Register (ARHn_APCFG00 - Channel 0)

The software can use this register to program the APIX® PHY. This register ARHn_APCFG00 can be written when ARHn_CHCTRL0:TXCFG = '1'.

■ ARH APIX® Configuration Register (ARHn_APCFG00) Channel 0

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	ENPLL	OFFSETCOMP	ENDOWNSTREAM	ENUPSTREAM	Reserved	CPOCFILTER	DISABLECPOC	TXDEEMPH4NS
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	TXBWMODE[1]	TXBWMODE[0]	PXINCLKACTIVEEDGE	PXINJUMBLEEN	PXINCTRLPIGGYBACK[1]	PXINCTRLPIGGYBACK[0]	PXDATAWIDTH[1]	PXDATAWIDTH[0]
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	1	0	0	0	0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved	RXUPBWMODE[1]	RXUPBWMODE[0]	UPLENIENTWINDOW	RXUPSMPOFST[3]	RXUPSMPOFST[2]	RXUPSMPOFST[1]	RXUPSMPOFST[0]
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	JUMBLERMASKEN	SBUPSKIPSKIP	SBUPRESTARTONERROR	BOOTSUPPORTDISABLE	TRACEFEEDENABLE	CEB6DISABLE	CEB4DISABLE	IGNOREPLLGOOD
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	1	0	0	0	0

[bit31] ENPLL: APIX® PHY PLL Enable

bit	Description
0	Power Down
1	Power Up

[bit30] OFFSETCOMP: APIX® PHY Offset Compensation Enable

bit	Description
0	Offset Compensation Disabled
1	Offset Compensation Enabled

[bit29] ENDOWNSTREAM: APIX® PHY Downstream Enable

bit	Description
0	Downstream Disabled
1	Downstream Enabled

[bit28] ENUPSTREAM: APIX® PHY Upstream Enable

bit	Description
0	Upstream Disabled
1	Upstream Enabled

[bit27] Reserved: Reserved bits

[bit26] CPOCFILTER: APIX® PHY Charge Pump Offset Correction Filter Enable

bit	Description
0	Filter on offset correction amplifier output is disabled
1	Filter on offset correction amplifier output is enabled

[bit25] DISABLECPOC: APIX® PHY Charge Pump Offset Correction Amplifier Disable

bit	Description
0	Offset correction amplifier in charge pump is enabled/disabled by ARHn_APCFG00:ENPLL
1	Offset correction amplifier in charge pump is disabled

[bit24] TXDEEMPH4NS: APIX® PHY De-emphasis Signal Control

250 Mbps/125 Mbps modes:

bit	Description
0	De-emphasize after 2 ns
1	De-emphasize after 4 ns

500 Mbps mode:

bit	Description
0	De-emphasize after 2 ns
1	De-emphasize after 2 ns and suppress emphasis if previous run length = 1 UI

[bit23:22] TXBWMODE[1:0]: APIX® PHY TX Mode Select

bit	Description
00	125 Mbit/s (Low Bandwidth Mode 1)
01	250 Mbit/s (Low Bandwidth Mode 2)
10	500 Mbit/s (Half Bandwidth Mode)
11	1000 Mbit/s (Full Bandwidth Mode)

[bit21] PXINCLKACTIVEEDGE: A-Shell Active Clock Edge Configuration

Clock Edge at which pixel data and pixel controls are captured.

bit	Description
0	Falling edge
1	Rising edge

[bit20] PXINJUMBLEEN: A-Shell Cyclic Pixel Data Inversion Configuration

bit	Description
0	Disable
1	Enable

[bit19:18] PXINCTRLPIGGYBACK[1:0]: A-Shell Pixel Control Data Transmission Configuration

bit	Description
00	Never
01	Unused
10	With even pixels only
11	With every pixel

[bit17:16] PXDATAWIDTH[1:0]: A-Shell Pixel Data Width Configuration

bit	Description
00	10 bits
01	12 bits
10	18 bits
11	24 bits

[bit15] Reserved: Reserved bits

[bit14:13] RXUPBWMODE[1:0]: APIX® PHY RX Data Receive Rate

bit	Description
00	62.50 MBit/s (Full, Half, Low Bandwidth Mode)
01	41.67 MBit/s (Full Bandwidth Mode)
11	20.83 MBit/s (Half and Low Bandwidth Mode)
10	31.25 MBit/s (Full, Half, Low Bandwidth Mode)

[bit12] UPLENIENTWINDOW: A-Shell Synchronization Error Tolerance Configuration

Tolerates a single synchronization error within a defined timing window without automatic re-synchronization.

bit	Description
0	Disable
1	Enable

[bit11:8] RXUPSMPOFST[3:0]: APIX® PHY RX Bit Alignment Sampling Point

bit	Description
0000	Optimum sampling point when operating in 62.50 Mbit/s mode
0010	Optimum sampling point when operating in 41.67 MBit/s or 31.25 MBit/s mode
0100	Optimum sampling point when operating in 20.83 MBit/s mode

[bit7] JUMBLERMASKEN: A-Shell Pixel Data Mask Enable Configuration

Masks Pixel Data.

bit	Description
0	Disable
1	Enable

[bit6] SBUPSKIPSKIP: A-Shell Resynchronization in Upstream Channel Configuration

This bit accelerates resynchronization after loss of synchronization by assuming that the upstream receiver is still locked (but bits are corrupted).

bit	Description
0	Enable
1	Disable

[bit5] SBUPRESTARTONERROR: A-Shell Restart on Error in Upstream Channel Configuration

This bit forces realignment at bit Level in upstream channel whenever synchronization is lost.

bit	Description
0	Disable
1	Enable

[bit4] BOOTSUPPORTDISABLE: A-Shell Boot Support Disable Configuration

This bit specifies the condition when pixel data synchronization with the receiver is requested.

bit	Description
0	Wait for rising edge of pixel data enable
1	Wait for pixel data enable == '1'

[bit3] TRACEFEEDENABLE: A-Shell Trace Feed Enable Configuration

This bit enables the detection of discontinued pixel stream (pixel clock) or frequency below 4 MHz.

bit	Description
0	Enable
1	Disable

[bit2] CEB6DISABLE: A-Shell Consecutive 6 Bits Equivalence Check Configuration

Check for 6 consecutive bits with equal value.

bit	Description
0	Enable
1	Disable

[bit1] CEB4DISABLE: A-Shell Consecutive 4 Bits Equivalence Check Configuration

Check for 4 consecutive bits with equal value.

bit	Description
0	Enable
1	Disable

[bit0] IGNOREPLLGOOD: APIX® PHY PLLGOOD Output Ignore Configuration

Loss of PLL's synchronization to the reference results in a hardware reset.

bit	Description
0	Enable
1	Disable

2.1.19. ARH APIX® Configuration Register (ARHn_APCFG10 - Channel 1)

The software can use this register to program the APIX® PHY. This register ARHn_APCFG10 can be written when ARHn_CHCTRL1:TXCFG = '1'.

■ ARH APIX® Configuration Register (ARHn_APCFG10) Channel 1

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	reserved	reserved	reserved	reserved	reserved	reserved	reserved	reserved
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	reserved	reserved	PXINCLKACTIVEEDGE	PXINJUMBLEEN	PXINCTRLPIGGYBACK[1]	PXINCTRLPIGGYBACK[0]	PXDATAWIDTH[1]	PXDATAWIDTH[0]
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	1	0	0	0	0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	reserved	reserved	reserved	UPLENIENTWINDOW	reserved	reserved	reserved	reserved
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	JUMBLERMASKEN	SBUPSKIPSKIP	SBUPRESTARTONERROR	BOOTSUPPORTDISABLE	TRACEFEEDENABLE	CEB6DISABLE	CEB4DISABLE	IGNOREPLLGOOD
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	1	0	0	0	0

[bit31:22] Reserved: Reserved bits

[bit21] PXINCLKACTIVEEDGE: A-Shell Active Clock Edge Configuration

Clock Edge at which pixel data and pixel controls are captured.

bit	Description
0	Falling edge
1	Rising edge

[bit20] PXINJUMBLEEN: A-Shell Cyclic Pixel Data Inversion Configuration

bit	Description
0	Disable
1	Enable

[bit19:18] PXINCTRLPIGGYBACK[1:0]: A-Shell Pixel Control Data Transmission Configuration

bit	Description
00	Never
01	Unused
10	With even pixels only
11	With every pixel

[bit17:16] PXDATAWIDTH[1:0]: A-Shell Pixel Data Width Configuration

bit	Description
00	10 bits
01	12 bits
10	18 bits
11	24 bits

[bit15:13] Reserved: Reserved bits

[bit12] UPLENIENTWINDOW: A-Shell Synchronization Error Tolerance Configuration

Tolerates a single synchronization error within a defined timing window without automatic re-synchronization.

bit	Description
0	Disable
1	Enable

[bit11:8] Reserved: Reserved bits

[bit7] JUMBLERMASKEN: A-Shell Pixel Data Mask Enable Configuration

Masks Pixel Data.

bit	Description
0	Disable
1	Enable

[bit6] SBUPSKIPSKIP: A-Shell Resynchronization in Upstream Channel Configuration

This bit accelerates resynchronization after loss of synchronization by assuming that the upstream receiver is still locked (but bits are corrupted).

bit	Description
0	Enable
1	Disable

[bit5] SBUPRESTARTONERROR: A-Shell Restart on Error in Upstream Channel Configuration

This bit forces realignment at bit Level in upstream channel whenever synchronization is lost.

bit	Description
0	Disable
1	Enable

[bit4] BOOTSUPPORTDISABLE: A-Shell Boot Support Disable Configuration

This bit specifies the condition when pixel data synchronization with the receiver is requested.

bit	Description
0	Wait for rising edge of pixel data enable
1	Wait for pixel data enable == '1'

[bit3] TRACEFEEDENABLE: A-Shell Trace Feed Enable Configuration

This bit enables the detection of discontinued pixel stream (pixel clock) or frequency below 4 MHz.

bit	Description
0	Enable
1	Disable

[bit2] CEB6DISABLE: A-Shell Consecutive 6 Bits Equivalence Check Configuration

Check for 6 consecutive bits with equal value.

bit	Description
0	Enable
1	Disable

[bit1] CEB4DISABLE: A-Shell Consecutive 4 Bits Equivalence Check Configuration

Check for 4 consecutive bits with equal value.

bit	Description
0	Enable
1	Disable

[bit0] IGNOREPLLGOOD: APIX® PHY PLLGOOD Output Ignore Configuration

Loss of PLL's synchronization to the reference results in a hardware reset.

bit	Description
0	Enable
1	Disable

2.1.20. ARH APIX® Configuration Register (ARHn_APCFG01 - Channel 0)

The software can use this register to program the APIX® PHY.
This register can be written when ARHn_CHCTRL0:TXCFG = '1'.

■ ARH APIX® Configuration Register (ARHn_APCFG01) Channel 0

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	DDOWNENABLE	SBDOWNSMODE	CLKCORE1ENABLE	CLKCORE2ENABLE	CLKCORE3ENABLE	MASKPLLGOOD	reserved	reserved
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	0	0	0	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	CRGPMPCTRL[3]	CRGPMPCTRL[2]	CRGPMPCTRL[1]	CRGPMPCTRL[0]	PREDRVIINC	reserved	TXNOMSWING[3]	TXNOMSWING[2]
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	TXNOMSWING[1]	TXNOMSWING[0]	reserved	reserved	reserved	TXDEEMPH[2]	TXDEEMPH[1]	TXDEEMPH[0]
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	TRIGGERCYCLELENGTH[6]	TRIGGERCYCLELENGTH[5]	TRIGGERCYCLELENGTH[4]	TRIGGERCYCLELENGTH[3]	TRIGGERCYCLELENGTH[2]	TRIGGERCYCLELENGTH[1]	TRIGGERCYCLELENGTH[0]	reserved
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	1	0	0	1	0	0	0

[bit31] DDOWNENABLE: A-Shell Downstream Data Path Configuration

Clock Edge at which pixel data and pixel controls are captured.

bit	Description
0	Disable data mode / enable pixel stream mode
1	Enable data mode / disable pixel stream mode

[bit30] SBDOWNSMODE: APIX® PHY Downstream Sideband Data Relation to Core Clock

bit	Description
0	Asynchronous
1	Synchronous

[bit29] CLKCORE1ENABLE: APIX® PHY Core Clock Enable

bit	Description
0	Disable
1	Enable

[bit28] CLKCORE2ENABLE: A-Shell1 Core Clock Enable

bit	Description
0	Disable
1	Enable

[bit27] CLKCORE3ENABLE: A-Shell2 Core Clock Enable

bit	Description
0	Disable
1	Enable

[bit26] MASKPLLGOOD: PLLGOOD Mask bit for APIX® PHY

bit	Description
0	No effect
1	Masks the actual value of 'pll_good'

[bit25:24] Reserved: Reserved bits

[bit23:20] CRGPMPCTRL[3:0]: APIX® PHY Charge Pump Current Control

bit	Description
0000	5uA
0001	10uA
0010	15uA
…	…
1111	80uA

[bit19] PREDRVIINC: APIX® PHY Pre-Driver Output Swing Control

bit	Description
0	Normal operation
1	Higher swing provided on pre-driver output

[bit18] Reserved: Reserved bits

[bit17:14] TXNOMSWING[3:0]: APIX® PHY Transmit Swing

bit	Description
0000	Min (nominal 4 mA)
…	…
1111	Max (nominal 12 mA)

[bit13:11] Reserved: Reserved bits

[bit10:8] TXDEEMPH[2:0]: APIX® PHY Transmit De-emphasis Level

bit	Description
000	Min 0%
001	7%
…	…(1lsb = 7%)
111	Max 50%

[bit7:1] TRIGGERCYCLELENGTH[6:0]: A-Shell Sideband Downstream Trigger Output Pulse Pattern Cycle Length Configuration

bit	Description
0001001	Required when operating in full and half bandwidth mode
0010010	Required when operating in low bandwidth mode 2 (APIX® PHY core_clk == 62.5 MHz)
0100100	Required when operating in low bandwidth mode 1 (APIX® PHY core_clk == 62.5 MHz)
0100100	Required when operating in low bandwidth mode 2 (APIX® PHY core_clk == 125 MHz)
1001000	Required when operating in low bandwidth mode 1 (APIX® PHY core_clk == 125 MHz)

[bit0] Reserved: Reserved bits

2.1.21. ARH APIX® Configuration Register (ARHn_APCFG11 - Channel 1)

The software can use this register to program the APIX® PHY.
This register can be written when ARHn_CHCTRL0:TXCFG = '1'.

■ ARH APIX® Configuration Register (ARHn_APCFG11) Channel 1

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	DDOWNENABLE	SBDOWNSMODE	CLKCORE1ENABLE	CLKCORE2ENABLE	CLKCORE3ENABLE	MASKPLLGOOD	reserved	reserved
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	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	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	reserved	reserved	reserved	reserved	reserved	reserved	reserved
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	TRIGGERCYCLELENGTH[6]	TRIGGERCYCLELENGTH[5]	TRIGGERCYCLELENGTH[4]	TRIGGERCYCLELENGTH[3]	TRIGGERCYCLELENGTH[2]	TRIGGERCYCLELENGTH[1]	TRIGGERCYCLELENGTH[0]	reserved
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	1	0	0	1	0	0	0

[bit31] DDOWNENABLE: A-Shell Downstream Data Path Configuration

Clock Edge at which pixel data and pixel controls are captured.

bit	Description
0	Disable data mode / enable pixel stream mode
1	Enable data mode / disable pixel stream mode

Note:

– *For proper operation the following settings are mandatory:*

ARHn_APCFG10:PXDATAWIDTH[1:0] := '00'

ARHn_APCFG10:PXINCTRLPIGGYBACK[1:0] := '00'

[bit30] SBDOWNSMODE: APIX® PHY Downstream Sideband Data Relation to Core Clock

bit	Description
0	Asynchronous
1	Synchronous

[bit29] CLKCORE1ENABLE: APIX® PHY Core Clock Enable

bit	Description
0	Disable
1	Enable

[bit28] CLKCORE2ENABLE: A-Shell1 Core Clock Enable

bit	Description
0	Disable
1	Enable

[bit27] CLKCORE3ENABLE: A-Shell2 Core Clock Enable

bit	Description
0	Disable
1	Enable

[bit26] MASKPLLGOOD: PLLGOOD Mask bit for APIX® PHY

bit	Description
0	No effect
1	Masks the actual value of 'pll_good'

[bit25:8] Reserved: Reserved bits

[bit7:1] TRIGGERCYCLELENGTH[6:0]: A-Shell Sideband Downstream Trigger Output Pulse Pattern Cycle Length Configuration

bit	Description
0001001	Required when operating in full and half bandwidth mode
0010010	Required when operating in low bandwidth mode 2 (APIX® PHY core_clk == 62.5 MHz)
0100100	Required when operating in low bandwidth mode 1 (APIX® PHY core_clk == 62.5 MHz)
0100100	Required when operating in low bandwidth mode 2 (APIX® PHY core_clk == 125 MHz)
1001000	Required when operating in low bandwidth mode 1 (APIX® PHY core_clk == 125 MHz)

[bit0] Reserved: Reserved bits

2.1.22. ARH APIX® Configuration Register (ARHn_APCFG02 - Channel 0)

The software can use this register to program the APIX® PHY.
This register can be written when ARHn_CHCTRL0:TXCFG = '1'.

■ ARH APIX® Configuration Register (ARHn_APCFG02) Channel 0

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	TRIGGERACTIVELENGTH[6]	TRIGGERACTIVELENGTH[5]	TRIGGERACTIVELENGTH[4]	TRIGGERACTIVELENGTH[3]	TRIGGERACTIVELENGTH[2]	TRIGGERACTIVELENGTH[1]	TRIGGERACTIVELENGTH[0]	reserved
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	1	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	TRIGGEROFFSET[6]	TRIGGEROFFSET[5]	TRIGGEROFFSET[4]	TRIGGEROFFSET[3]	TRIGGEROFFSET[2]	TRIGGEROFFSET[1]	TRIGGEROFFSET[0]	reserved
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	1	0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	SBUPVALIDACTIVELENGTH[1]	SBUPVALIDACTIVELENGTH[0]	reserved	reserved	reserved	reserved	reserved	reserved
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	1	0	0	0	0	0	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	PLLMULT	OSCFILTER	TXINVERT	RXINVERT	PLLGOODSEL	reserved	reserved	SCPREEN
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R0,WX	R0,WX	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:25] TRIGGERACTIVELENGTH[6:0]:
A-Shell Sideband Downstream Trigger Output Active Length Configuration
This bit configures high pulse width of sideband downstream trigger output (in multiples of core clk cycle).

bits	Description
0000000	1 cycle
0000001	2 cycle
0000010	3 cycle
…	
11111111	72 cycle

[bit24] Reserved: Reserved bits

[bit23:17] TRIGGEROFFSET[6:0]:
A-Shell Sideband Downstream Trigger Output Offset Configuration
This bit configures the start position of sideband downstream trigger output (in multiples of core clk cycle relative to strobe position).

bit	Description
0000000	0 cycle
0000001	1 cycle
0000010	2 cycle
…	
11111111	71 cycle

[bit16] Reserved: Reserved bits

[bit15:14] SBUPVALIDACTIVELENGTH[1:0]:
A-Shell Sideband Upstream Valid Output Active Length Configuration
This bit configures high pulse width of sideband upstream valid output (in multiples of core clk cycle).

bit	Description
00	1 cycle
01	2 cycle
10	3 cycle
11	4 cycle

[bit13:8] Reserved: Reserved bits

[bit7] PLLMULT: APIX® PHY PLL Multiplier
Multiplier factor for PLL. PLL uses the following reference frequencies.

bit	Description
0	20 MHz
1	25 MHz

[bit6] OSCFILTER: APIX® PHY Oscillator Filter Enable

bit	Description
0	Disable
1	Enable 100 MHz LPF in oscillator path

[bit5] TXINVERT: APIX® PHY TX Data Invert Enable

bit	Description
0	Data not inverted
1	Invert data on TX

[bit4] RXINVERT: APIX® PHY RX Data Invert Enable

bit	Description
0	Data not inverted
1	Invert data on TX

[bit3] PLLGOODSEL: APIX® PLLGOOD Status Select

bit	Description
0	APIX® PLLGOOD is a reset criteria
1	APIX® PLLGOOD is not a reset criteria

[bit2:1] Reserved: Reserved bits

[bit0] SCPREEN: Sysclk Prescalar Enable

bit	Description
0	A-Shell clock is equal to bus clock
1	A-Shell clock is half of bus clock

Note:
Sysclk prescalar enable is common across Ashell 0 and 1.

2.1.23. ARH APIX® Configuration Register (ARHn_APCFG12 - Channel 1)

The software can use this register to program the APIX® PHY.
This register can be written when ARHn_CHCTRL1:TXCFG = '1'.

■ ARH APIX® Configuration Register (ARHn_APCFG12) Channel 1

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	TRIGGERACTIVELENGTH[6]	TRIGGERACTIVELENGTH[5]	TRIGGERACTIVELENGTH[4]	TRIGGERACTIVELENGTH[3]	TRIGGERACTIVELENGTH[2]	TRIGGERACTIVELENGTH[1]	TRIGGERACTIVELENGTH[0]	reserved
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	1	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	TRIGGEROFFSET[6]	TRIGGEROFFSET[5]	TRIGGEROFFSET[4]	TRIGGEROFFSET[3]	TRIGGEROFFSET[2]	TRIGGEROFFSET[1]	TRIGGEROFFSET[0]	reserved
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	1	0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	SBUPVALIDACTIVELENGTH[1]	SBUPVALIDACTIVELENGTH[0]	reserved	reserved	reserved	reserved	reserved	reserved
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	1	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	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

[bit31:25] TRIGGERACTIVELENGTH[6:0]:
A-Shell Sideband Downstream Trigger Output Active Length Configuration
This bit configures high pulse width of sideband downstream trigger output (in multiples of core clk cycle).

bits	Description
0000000	1 cycle
0000001	2 cycle
0000010	3 cycle
…	
11111111	72 cycle

[bit24] Reserved: Reserved bits

[bit23:17] TRIGGEROFFSET[6:0]:
A-Shell Sideband Downstream Trigger Output Offset Configuration
This bit configures the start position of sideband downstream trigger output (in multiples of core clk cycle relative to strobe position).

bit	Description
0000000	0 cycle
0000001	1 cycle
0000010	2 cycle
…	
11111111	71 cycle

[bit16] Reserved: Reserved bits

[bit15:14] SBUPVALIDACTIVELENGTH[1:0]:
A-Shell Sideband Upstream Valid Output Active Length Configuration
This bit configures high pulse width of sideband upstream valid output (in multiples of core clk cycle).

bit	Description
00	1 cycle
01	2 cycle
10	3 cycle
11	4 cycle

[bit13:0] Reserved: Reserved bits

2.1.24. ARH APIX® Configuration Register (ARHn_APCFG03 - Channel 0, ARHn_APCFG13 - Channel 1)

The software can use this register to program the A-Shell. This register can be written when ARHn_CHCTRL0:TXCFG = '1 '. Register ARHn_APCFG13 (used for channel 1) can be written when ARHn_CHCTRL1:TXCFG = '1 '.
Register ARHn_APCFG03 for channel 0 is explained here.

■ ARH APIX® Configuration Register (ARHn_APCFG03) Channel 0

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	SBDWNCLK	SBDWNDACLKCLENL[6]	SBDWNDACLKCLENL[5]	SBDWNDACLKCLENL[4]	SBDWNDACLKCLENL[3]	SBDWNDACLKCLENL[2]	SBDWNDACLKCLENL[1]	SBDWNDACLKCLENL[0]
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	1	1	0

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	SBUPDWIDTH	SBUPDACLK	SBDWNDWIDTH	SBDWNDACLK[1]	SBDWNDACLK[0]	EPHY	ESHELL	SPIOVERSB
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	1	0	1	0	0	1	0	0

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	CRCTIMEOUTVAL[3]	CRCTIMEOUTVAL[2]	CRCTIMEOUTVAL[1]	CRCTIMEOUTVAL[0]	WINDOWSIZE[3]	WINDOWSIZE[2]	WINDOWSIZE[1]	WINDOWSIZE[0]
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	1	1	0	1	0

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	ARQOFF	SUPPRESSITA	reserved	reserved	reserved	SBDWNDACLKCLENU[2]	SBDWNDACLKCLENU[1]	SBDWNDACLKCLENU[0]
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] SBDWNCLK: Functional Meaning of Sideband Downstream Trigger

This bit defines the functional meaning of sideband downstream trigger when its start position is set to '0'.

bits	Description
0	Request
1	Strobe

[bit30:24] SBDWNDACLKCLENL[6:0]: Sideband Downstream Clock Cycle Time

These bits [ARHn_APCFG03:SBDWNDACLKCLENU and SBDWNDACLKCLENL] configure cycle time of sideband downstream clock (in multiples of A-Shell core clock) when sideband downstream data are asynchronous of ARHn_APCFG03:SPIOVERSB is enabled.

These are lower bits of SBDWNDACLKCLEN[9:0].

[bit23] SBUPDWIDTH: Sideband Upstream Ports Enable

bits	Description
0	Enables Sideband upstream data[0]
1	Enables Sideband upstream data[1:0]

[bit22] SBUPDACLK: Sideband Upstream Data Validation

bits	Description
0	Sideband upstream data is validated with sideband upstream data valid
1	Sideband upstream data is validated with sideband upstream data[1]

[bit21] SBDWNDWIDTH: Sideband Downstream Ports Enable

bit	Description
0	Enables Sideband downstream data[0]
1	Enables Sideband downstream data[1:0]

[bit20:19] SBDWNDACLK[1:0]: Sideband Downstream Clock Generate Source

These bits decide the sideband downstream clock generation source.

bit	Description
00	Disable
01	With use of sideband downstream trigger (synchronous to core clk of APIX® PHY)
10	With use of internal counter (asynchronous to core clk of APIX® PHY)
11	Disable

[bit18] EPHY: Internal A-Shell to External APIX® PHY Connection Control

This bit controls the connection of internal A-Shell to external APIX® PHY through GPIO interface.

bit	Description
0	Disable
1	Enable

[bit17] ESHELL: Internal APIX® PHY to External A-Shell Connection Control

This bit controls the connection of internal APIX® PHY to external A-Shell through GPIO interface.

bit	Description
0	Disable
1	Enable

[bit16] SPIOVERSB: SPI Over Sideband Control

bit	Description
0	SPI transmission over sideband is disabled
1	SPI transmission over sideband is enabled

[bit15:12] CRCTIMEOUTVAL[3:0]: CRC Timeout Value

CRC timeout error is generated after N consecutive CRC errors(N = base x multiplier).

CRCTIMEOUTVAL[3:2]		CRCTIMEOUTVAL[1:0]	
bit	Multiplier	bit	Base
00	1	00	2
01	4	01	4
10	16	10	6
11	128	11	10

[bit11:8] WINDOWSIZE[3:0]: ACKNOWLEDGEMENT WINDOW SIZE

These bits defines the window size of the acknowledge protocol.
(supported sizes:1....12)

[bit7] ARQOFF: ARQ Control

bit	Description
0	ARQ enabled
1	ARQ disabled

[bit6] SUPPRESSITA: Outbound Idle Transaction Control

bit	Description
0	Outbound Idle transactions are sent
1	Outbound Idle transactions are not sent

[bit5:3] Reserved: Reserved bits

[bit2:0] SBDWNDACLKCLENU[2:0]: Sideband Downstream Clock Cycle Time

These bits[SBDWNDACLKCLENU and ARHn_APCFG03:SBDWNDACLKCLENL] configure cycle time of sideband downstream clock (in multiples of A-Shell core clock) when sideband downstream data are asynchronous of ARHn_APCFG03:SPIOVERSB is enabled.

These are upper bits of SBDWNDACLKCLEN[9:0].

2.1.25. ARH Test Register (ARHn_TST)

The software can use this register to test the Event Buffer memory.

■ ARH Test Register (ARHn_TST)

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

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

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	RW	TM
ACCESS_TYPE	R0,WX	R0,WX	R0,WX	R0,WX	R0,WX	R0,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	Reserved	ADDR[6]	ADDR[5]	ADDR[4]	ADDR[3]	ADDR[2]	ADDR[1]	ADDR[0]
ACCESS_TYPE	R0,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

[bit31:10] Reserved: Reserved bits

[bit9] RW: RW RAM mode

bits	Description
0	Read mode
1	Write mode

[bit8] TM: TM Test Mode

bits	Description
0	FIFO operation (Event Buffer mode)
1	RAM operation (test mode)

[bit7] Reserved: Reserved bits

[bit6:0] ADDR[6:0]: Address

Address of memory to be read or written. The address range is 0-127.

Note:

- *In RAM operation mode (ARHn_TST:TM == '1'), direct access to internal RAM is allowed for debug purpose. Read and write operation over the RAM is possible with RW bit. The address location for read or write is provided in ADDR[6:0] register bits. The read or write data is available in the registers ARHn_EVBUF0 and ARHn_EVBUF1.*

2.1.26. ARH Unlock Register (ARHn_UNLOCK)

The software should use this register to lock (0xA86AB10C) or unlock (0xA86ACCE5) ARHn_CHCTRL0, ARHn_CHCTRL1 and ARHn_EVCTRL registers for write access.

■ ARH Unlock Register (ARHn_UNLOCK)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	UNLOCK[31]	UNLOCK[30]	UNLOCK[29]	UNLOCK[28]	UNLOCK[27]	UNLOCK[26]	UNLOCK[25]	UNLOCK[24]
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	UNLOCK[23]	UNLOCK[22]	UNLOCK[21]	UNLOCK[20]	UNLOCK[19]	UNLOCK[18]	UNLOCK[17]	UNLOCK[16]
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	UNLOCK[15]	UNLOCK[14]	UNLOCK[13]	UNLOCK[12]	UNLOCK[11]	UNLOCK[10]	UNLOCK[9]	UNLOCK[8]
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	UNLOCK[7]	UNLOCK[6]	UNLOCK[5]	UNLOCK[4]	UNLOCK[3]	UNLOCK[2]	UNLOCK[1]	UNLOCK[0]
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] UNLOCK[31:0]: ARH Unlock

The ARH unlock register protects the ARH module from being modified accidentally by software.
The ARH registers ARHn_CHCTRL0, ARHn_CHCTRL1 and ARHn_EVCTRL cannot be written until this register has been written with a specific unlock value (0xA86ACCE5). The read to this register always returns a zero.
To lock the ARH again software must write another value specific to lock(0xA86AB10C). Write access without unlocking or writing values other than lock value or unlock value to the above mentioned registers cause protection error.

Note:

- *This register cannot be written by 8-bit or 16-bit write access, any such access causes protection error.*
- *For more details on LOCK and UNLOCK key refer to the device specific datasheet.*

2.1.27. ARH Module ID Register (ARHn_MID)

This is a read-only register with a unique Module Identification Number which identifies the version of the ARH module used in the MCU.

■ ARH Module ID Register (ARHn_MID)

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	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	MID[23]	MID[22]	MID[21]	MID[20]	MID[19]	MID[18]	MID[17]	MID[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

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	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	MID[7]	MID[6]	MID[5]	MID[4]	MID[3]	MID[2]	MID[1]	MID[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] MID[31:0]: Module ID

The ARH module implemented in the device may vary from device to device. This register identifies the particular version of the hardware used in the device. This register helps in developing software to the hardware version implemented in the device.

Note:

For more details on module ID number refer to device specific datasheet.

2.1.28. ARH APIX® Configuration Register (ARHn_APCFG04 - Channel 0, ARHn_APCFG14 - Channel 1)

These registers are used to configure the data and select the clock source for A-Shell. Register ARHn_APCFG04 can be written when ARHn_CHCTRL0:TXCFG = '1'. Register ARHn_APCFG14 (used for channel 1) can be written when ARHn_CHCTRL1:TXCFG = '1'. Register ARHn_APCFG04 for channel 0 is explained here.

■ ARH APIX® Configuration Register (ARHn_APCFG04) Channel 0

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	SYSCLKSEL	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R/W	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	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

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	AACCLKDATAOFST[1]	AACCLKDATAOFST[0]	AACCLKFREQ[11]	AACCLKFREQ[10]	AACCLKFREQ[9]	AACCLKFREQ[8]	AACCLKFREQ[7]	AACCLKFREQ[6]
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	AACCLKFREQ[5]	AACCLKFREQ[4]	AACCLKFREQ[3]	AACCLKFREQ[2]	AACCLKFREQ[1]	AACCLKFREQ[0]	DATAPATHSEL	AACCLKMODE
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] SYSCLKSEL: A-Shell0 System Clock Source Selector

This bit selects the clock source for APIX®.

bits	Description
0	Clock of 62.5 MHz from APIX® PHY is selected
1	Clock of either same or half of the bus clock frequency (depending on ARHn_APCFG02:SCPREEN) is selected

[bit30:16] Reserved: Reserved bits

[bit15:14] AACCLKDATAOFST[1:0]: AIC2 Clock Data Offset

These bits configures offset between the edges of AIC2 clock and AIC2 data in multiples of core clock.

bits	Description
00	1 core clock cycle
01	2 core clock cycle
10	3 core clock cycle
11	4 core clock cycles

[bit13:2] AACCLKFREQ[11:0]: AIC2 Clock Frequency

AIC2 Clock Frequency = core clock/(2*N).

bits	Description
000000000000	AIC2 Clock generator is off
000000000001	This configuration is not allowed
000000000010	N = 2
…	
1111111111111	N=4095

[bit1] DATAPATHSEL: Data Path Selection

bits	Description
0	A-Shell is connected to APIX® PHY
1	A-Shell is connected to AIC2

[bit0] AACCLKMODE: AIC2 Clock Mode

bits	Description
0	Data transfers at both edges of AIC2 clock
1	Data transfers at rising edges of AIC2 clock only

2.1.29. APIX® PHY Evaluation Transmitter Pattern Register (ARHn_EVAL0)

This register is used to configure the transmitter pattern for APIX® PHY evaluation.
Write access to this register is protected by PPU test attribute.

■ APIX® PHY Evaluation Transmitter Pattern Register (ARHn_EVAL0)

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

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

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	TXPATTERN[15]	TXPATTERN[14]	TXPATTERN[13]	TXPATTERN[12]	TXPATTERN[11]	TXPATTERN[10]	TXPATTERN[9]	TXPATTERN[8]
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	TXPATTERN[7]	TXPATTERN[6]	TXPATTERN[5]	TXPATTERN[4]	TXPATTERN[3]	TXPATTERN[2]	TXPATTERN[1]	TXPATTERN[0]
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

Note:

- *For devices which do not contain internal APIX® PHY then write to this register does not affect. For APIX® PHY availability refer to device specific datasheet.*

[bit31:16] Reserved: Reserved bits

[bit15:0] TXPATTERN[15:0]: TX Pattern Configuration

Configures the 16-bit user defined pattern to be used in the APIX® PHY pattern generator.

2.1.30. APIX® PHY Evaluation Receiver Pattern Register (ARHn_EVAL1)

This register shows the receiver pattern of APIX® PHY evaluation. Access to this register is protected by PPU test attribute.

■ APIX® PHY Evaluation Receiver Pattern Register (ARHn_EVAL1)

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

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

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	RXPATTERN[15]	RXPATTERN[14]	RXPATTERN[13]	RXPATTERN[12]	RXPATTERN[11]	RXPATTERN[10]	RXPATTERN[9]	RXPATTERN[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

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	RXPATTERN[7]	RXPATTERN[6]	RXPATTERN[5]	RXPATTERN[4]	RXPATTERN[3]	RXPATTERN[2]	RXPATTERN[1]	RXPATTERN[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

Note:

- *For devices which do not contain internal APIX® PHY then write to this register does not affect. For APIX® PHY availability refer to device specific datasheet.*

[bit31:16] Reserved: Reserved bits

[bit15:0] RXPATTERN[15:0]: RX Pattern Configuration

16-bit pattern received at RX (aligns when preamble is selected by ARHn_EVAL2:PATTERNGENEN[1:0]).

2.1.31. APIX® PHY Evaluation Receiver Pattern Register (ARHn_EVAL2)

This register is used to configure the APIX® PHY for evaluation. Access to this register is protected by PPU test attribute.

■ APIX® PHY Evaluation Configuration Register (ARHn_ EVAL2)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	PATTERNGENEN[1]	PATTERNGENEN[0]	RXEDGE	PATTERNRATE	PATTCHKSOURCE	ERRORHOLD[1]	ERRORHOLD[0]	Reserved
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	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	TXSOURCE[2]	TXSOURCE[1]	TXSOURCE[0]	ENLOOPBCK[1]	ENLOOPBCK[0]	RXEPTRIGGER	Reserved	Reserved
ACCESS_TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R0,WX	R0,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	LOOPBCKSWING[3]	LOOPBCKSWING[2]	LOOPBCKSWING[1]	LOOPBCKSWING[0]	Reserved	Reserved	Reserved	Reserved
ACCESS_TYPE	R/W	R/W	R/W	R/W	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	TESTPD	Reserved	Reserved	MASKPLLGOODSET	MASKPLLGOODCLR	RXREALIGNSET	RXREALIGNCLR	TESTENABLE
ACCESS_TYPE	R/W	R0,WX	R0,WX	R/W	R/W	R/W	R/W	R/W
PROT_TYPE	-							
INITIAL_VALUE	0	0	0	0	1	0	1	0

Note:

– *For devices which do not contain internal APIX® PHY then write to this register does not affect. For APIX® PHY availability refer to device specific datasheet.*

[bit31:30] PATTERNGENEN[1:0]: Pattern Generated and/or Checked

bit	Description
00	16 bit user defined pattern, inverting after 16 bits
01	16 bit user defined pattern
10	10-bit PRBS
11	Preamble 55AA (Pattern checker aligns to word. This is independent of Tx Source)

[bit29] RXEDGE: Edge Selector for Sampling

bit	Description
0	Sample on rising edge of 500 MHz
1	Sample on falling edge of 500 MHz

[bit28] PATTERNRATE: Pattern Rate Selector(for generator and checker)

bit	Description
0	According to ARHn_APCFG00:DWNBWMODE
1	62.5 Mb/s (for upstream test)

[bit27] PATTCHKSOURCE: Pattern Check Source

Select source of data for pattern checker.

bit	Description
0	From samplers
1	Rx Bit

[bit26:25] ERRORHOLD[1:0]: Hold Time for ARHn_EVAL2:RXEQTX on Error

bit	Description
00	16 ns (1 cycle period of 62.5 MHz clock)
01	256 ns (16 cycle periods of 62.5 MHz clock)
10	Until next ARHn_EVAL3:RXTXLOCKED
11	Until ARHn_EVAL2:PATTERNGENEN set to preamble

[bit24] Reserved: Reserved bits

[bit23:21] TXSOURCE[2:0]: Transmit Data Source

bits	Description
000	Tx Data from core (normal operation)
001	Pattern generator
010	Sampled data (500 Mbps)
011	Rx Bit
100	500 Mbps Toggle (01010101)
101	250 Mbps Toggle
110	125 Mbps Toggle
111	62.5 Mbps Toggle

Note:

- *The pattern generator is independent of TXSOURCE.*

[bit20:19] ENLOOPBCK[1:0]: Enable Loopback

Loopback from TX output to RX input

bit	Description
0x	Loopback disabled
10	Loopback from serializer output
11	Loopback from SDOUT

[bit18] RXEPTRIGGER: ARHn_EVAL3:RXEDGEPOS[3] Enable

0 ->1 Sets ARHn_EVAL3:RXEDGEPOS[3] so that ARHn_EVAL3:RXEDGEPOS is in range 4...11

After RXEPTRIGGER, ARHn_EVAL3:RXEDGEPOS indicates a shift of minimum 4500 MHz clock cycles (=0.5 UI @ 62.5 Mbps) before wrapping.

[bit17:16] Reserved: Reserved bits

[bit15:12] LOOPBCKSWING[3:0]: Loopback Swing Control

bits	Description
0000	48 mV (differential)
…	
1111	192 mV (differential)

[bit11:8] Reserved: Reserved bits

[bit7] TESTPD: Power down for Test

This bit powers down complete APIX® PHY macro. Used for production test only.

bits	Description
0	No effect
1	Power OFF. Overrides ARHn_APCFG00:PWRDOWN, ARHn_APCFG00:PWRDOWNPREEMP, ARHn_APCFG00:PWRDOWNNOM and ARHn_APCFG00:PWRDOWNUP power control signals.

[bit6:5] Reserved: Reserved bits

[bit4] MASKPLLGOODSET: A-Shell0 MASKPLLGOOD Set

This bit sets the MASKPLLGOOD output of A-Shell0.

bits	Description
0	No effect
1	Sets the MASKPLLGOOD output of A-Shell0

Note:

- *ARHn_EVAL2:MASKPLLGOODCLR has got higher priority than MASKPLLGOODSET.*

[bit3] MASKPLLGOODCLR: A-Shell0 MASKPLLGOOD Clear

This bit clears MASKPLLGOOD output of A-Shell0.

bits	Description
0	Clears the MASKPLLGOOD output of A-Shell0
1	No effect

[bit2] RXREALIGNSET: A-Shell0 RXREALIGN Set Bit

This bit clears MASKPLLGOOD output of A-Shell0.

bits	Description
0	No effect
1	Sets the RXREALIGN output from A-Shell0

Note:

- *ARHn_EVAL2:RXREALIGNCLR has got higher priority than RXREALIGNSET.*

[bit1] RXREALIGNCLR: A-Shell0 RXREALIGN Clear Bit

bits	Description
0	Clears the RXREALIGN output of A-Shell0
1	No change

[bit0] TESTENABLE: Enable BIST

bits	Description
0	Normal operation
1	Test using pattern generator and checker

Note:

- *BIST controls ARHn_EVAL2:RXEDGE and ARHn_EVAL2:PATTERNGENEN*
  0->1 Edge triggers start of test
  1->0 Edge interrupts any running test
-

2.1.32. APIX® PHY Evaluation Receiver Status Register (ARHn_EVAL3)

This register gives the status of various attributes of receiver. Access to this register is protected by PPU test attribute.

■ APIX® PHY Evaluation Receiver Status Register (ARHn_ EVAL3)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	RXTXLOCKED	RXEQTX	Reserved	Reserved	RXEDGEPOS[3]	RXEDGEPOS[2]	RXEDGEPOS[1]	RXEDGEPOS[0]
ACCESS_TYPE	R,WX	R,WX	R0,WX	R0,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	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

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

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	-							
INITIAL_VALUE	0	0	0	0	0	0	0	0

Note:

- *For devices which do not contain internal APIX® PHY then write to this register does not affect. For APIX® PHY availability refer to device specific datasheet.*

[bit31] RXTXLOCKED: Preamble Lock Status

bit	Description
0	Preamble expected, but not yet detected
1	Preamble pattern has been detected and pattern checker has aligned to preamble word. Checker expects to receive pattern (or PRBS) data (starting with first word that is not preamble)

[bit30] RXEQTX: RX Pattern status

bit	Description
0	RX pattern is not detected (because of Error/LoopBack disabled/Preamble not yet detected)
1	RX pattern is as expected from TX pattern generator (goes high when preamble is detected)

[bit29:28] Reserved: Reserved bits

[bit27:24] RXEDGEPOS[3:0]: Position of Last Edge in Upstream Data

RXEDGEPOS[2:0]

bit	Description
000	Data edge is aligned to 62.5 MHz clock rising edge
001	Number of 500 MHz clocks that data edge lags rising edge of 62.5 MHz clock
…	
111	

RXEDGEPOS [3]
Toggles state when RXEDGEPOS[2:0] crosses 000/111 boundary
Sets to !RXEDGEPOS[2] at rising edge of ARHn_EVAL2:RXEPTRIGGER

[bit23:0] Reserved: Reserved bits

2.1.33. APIX® PHY Evaluation Misc Test Enable Register (ARHn_EVAL4)

This register is used to enable various miscellaneous test functions and to program length of the BIST pattern for APIX® PHY. Access to this register is protected by PPU test attribute.

■ APIX® PHY Evaluation Misc Test Enable Register (ARHn_ EVAL4)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	TESTMISC [25]	TESTMISC [24]	TESTMISC [23]	TESTMISC [22]	TESTMISC [21]	TESTMISC [20]	TESTMISC [19]	TESTMISC [18]
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	TESTMISC [17]	TESTMISC [16]	TESTMISC [15]	TESTMISC [14]	TESTMISC [13]	TESTMISC [12]	TESTMISC [11]	TESTMISC [10]
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	TESTMISC [9]	TESTMISC [8]	TESTMISC [7]	TESTMISC [6]	TESTMISC [5]	TESTMISC [4]	TESTMISC [3]	TESTMISC [2]
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	TESTMISC [1]	TESTMISC [0]	LOCK2PATTERN	TESTLENGTH[4]	TESTLENGTH[3]	TESTLENGTH[2]	TESTLENGTH[1]	TESTLENGTH[0]
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

Note:

- *For devices which do not contain internal APIX® PHY then write to this register does not affect. For APIX® PHY availability refer to device specific datasheet.*

[bit31:6] TESTMISC[25:0]

[bit5] LOCK2PATTERN: Lock Pattern Selection

bit	Description
0	Lock to preamble {55AA55AA}
1	Skip preamble, lock to pattern

[bit4:0] TESTLENGTH[4:0]: Length of BIST Test Pattern (excluding preamble)

bit	Description
00000	(2^TestLength) clocks
…	
11101	
11110	No pattern test (preamble only)
11111	

2.1.34. APIX® PHY Evaluation Status Register (ARHn_EVAL5)

This register gives the status of the tests performed on APIX® PHY.
Access to this register is protected by PPU test attribute.

■ APIX® PHY Evaluation Status Register (ARHn_ EVAL5)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
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	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
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	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
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	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
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

Note:

– *For devices which do not contain internal APIX® PHY then write to this register does not affect. For APIX® PHY availability refer to device specific datasheet.*

Reserved [31:0] : Reserved

2.1.35. APIX® PHY Evaluation Misc Test Configuration Register (ARHn_EVAL6)

This register is used to configure for various miscellaneous test functions for APIX® PHY.
Access to this register is protected by PPU test attribute.

■ APIX® PHY Evaluation Misc Test Configuration Register (ARHn_ EVAL6)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	AXMCFG[31]	AXMCFG[30]	AXMCFG[29]	AXMCFG[28]	AXMCFG[27]	AXMCFG[26]	AXMCFG[25]	AXMCFG[24]
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	AXMCFG[23]	AXMCFG[22]	AXMCFG[21]	AXMCFG[20]	AXMCFG[19]	AXMCFG[18]	AXMCFG[17]	AXMCFG[16]
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	AXMCFG[15]	AXMCFG[14]	AXMCFG[13]	AXMCFG[12]	AXMCFG[11]	AXMCFG[10]	AXMCFG[9]	AXMCFG[8]
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	AXMCFG[7]	AXMCFG[6]	AXMCFG[5]	AXMCFG[4]	AXMCFG[3]	AXMCFG[2]	AXMCFG[1]	AXMCFG[0]
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

Note:

- *For devices which do not contain internal APIX® PHY then write to this register does not affect. For APIX® PHY availability refer to device specific datasheet.*

[bit31:0] AXMCFG[31:0]: Misc Test Configuration

Configuration for Misc Test (details unavailable).

2.1.36. APIX® PHY Evaluation Misc Test Status Register (ARHn_EVAL7)

This register gives the status of various miscellaneous tests for APIX® PHY.
Access to this register is protected by PPU test attribute.

■ APIX® PHY Evaluation Misc Test Status Register (ARHn_ EVAL7)

BIT_OFFSET	31	30	29	28	27	26	25	24
BIT_NAME	AXMST[31]	AXMST[30]	AXMST[29]	AXMST[28]	AXMST[27]	AXMST[26]	AXMST[25]	AXMST[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

BIT_OFFSET	23	22	21	20	19	18	17	16
BIT_NAME	AXMST[23]	AXMST[22]	AXMST[21]	AXMST[20]	AXMST[19]	AXMST[18]	AXMST[17]	AXMST[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

BIT_OFFSET	15	14	13	12	11	10	9	8
BIT_NAME	AXMST[15]	AXMST[14]	AXMST[13]	AXMST[12]	AXMST[11]	AXMST[10]	AXMST[9]	AXMST[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

BIT_OFFSET	7	6	5	4	3	2	1	0
BIT_NAME	AXMST[7]	AXMST[6]	AXMST[5]	AXMST[4]	AXMST[3]	AXMST[2]	AXMST[1]	AXMST[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

Note:

- *For devices which do not contain internal APIX® PHY then write to this register does not affect. For APIX® PHY availability refer to device specific datasheet.*

[bit31:0] AXMST[31:0]: Misc Test Status

Status of Misc Tests (details unavailable).

3. Operation of ARH

This chapter describes the operation of ARH.

■ Data transfers (frame types)

The various frame/message types generated and handled by ARH are shown in this section.

■ Downstream write transaction (Remote Handler TX -> Remote Handler RX)

Table 34-3 Write message transmitted by ARH to remote (Remote Handler RX)

Bit	Byte	Field	Bits		Value	Description
55	0	Control[3:0]	Tsize	1		Transaction size: '00' = byte, '01' = half word (2 bytes), '10' = word (4 bytes), '11 ' = Reserved
54			Tsize	0		
53			OAen			Offset address enable
52			WRn		1	Write transaction
51		Addr[19:0]	Addr	19		Byte address
50			Addr	18		
49			Addr	17		
48			Addr	16		
47	1		Addr	15		
46			Addr	14		
45			Addr	13		
44			Addr	12		
43			Addr	11		
42			Addr	10		
41			Addr	9		
40			Addr	8		
39	2		Addr	7		
38			Addr	6		
37			Addr	5		
36			Addr	4		
35			Addr	3		
34			Addr	2		
33			Addr	1		
32			Addr	0		

Table 34-4 Write message transmitted by ARH to remote (Remote Handler RX) (Continued)

Bit	Byte	Field	Bits		Value	Description
31	3	Data[31:0]	Data	31		Data, If Tsize = '00': then Data[31:24] is used, If Tsize = '01': then Data[31:16] is used If Tsize = '10': then Data[31:0] is used
30			Data	30		
29			Data	29		
28			Data	28		
27			Data	27		
26			Data	26		
25			Data	25		
24			Data	24		
23	4		Data	23		
22			Data	22		
21			Data	21		
20			Data	20		
19			Data	19		
18			Data	18		
17			Data	17		
16			Data	16		
15	5		Data	15		
14			Data	14		
13			Data	13		
12			Data	12		
11			Data	11		
10			Data	10		
9			Data	9		
8			Data	8		
7	6		Data	7		
6			Data	6		
5			Data	5		
4			Data	4		
3			Data	3		
2			Data	2		
1			Data	1		
0			Data	0		

■ Downstream read request transaction (From ARH TX -> Remote ARH RX)

Table 34-5 Read request message transmitted by ARH to remote (Remote Handler RX)

Bit	Byte	Field	Bits		Value	Description
55	0	Control[3:0]	Tsize	1		Transaction size: '00' = byte, '01' = half word (2 bytes), '10' = word (4 bytes), '11' = Reserved
54			Tsize	0		
53			OAen			Offset address enable
52			WRn		0	Read request transaction
51		Addr[19:0]	Addr	19		Byte address
50			Addr	18		
49			Addr	17		
48			Addr	16		
47	1		Addr	15		
46			Addr	14		
45			Addr	13		
44			Addr	12		
43			Addr	11		
42			Addr	10		
41			Addr	9		
40			Addr	8		
39	2		Addr	7		
38			Addr	6		
37			Addr	5		
36			Addr	4		
35			Addr	3		
34			Addr	2		
33			Addr	1		
32			Addr	0		

Table 34-6 Read request message transmitted by ARH to remote (Remote Handler RX) (Continued)

Bit	Byte	Field	Bits		Value	Description
31	3	Idx[7:0]	Idx	7		Read index
30			Idx	6		
29			Idx	5		
28			Idx	4		
27			Idx	3		
26			Idx	2		
25			Idx	1		
24			Idx	0		
23	4	Reserved	-			Unused
22			-			
21			-			
20			-			
19			-			
18			-			
17			-			
16			-			
15	5		-			
14			-			
13			-			
12			-			
11			-			
10			-			
9			-			
8			-			
7	6		-			
6			-			
5			-			
4			-			
3			-			
2			-			
1			-			
0			-			

■ Upstream read result (Remote Handler RX -> Remote Handler TX)

Table 34-7 Read response message received by ARH from remote (Remote Handler RX)

Bit	Byte	Field	Bits		Value	Description
55	0	Control[3:0]	Tsize	1		Transaction size: '00' = byte, '01' = halfword(2 bytes), '10' = word (4 bytes), '11' = Reserved
54			Tsize	0		
53			OAen		-	Offset address enable
52			WRn		0	Read result transaction
51		Reserved	Err			Chip internal bus error
50			-			Unused
49			-			
48			-			
47	1	Idx[7:0]	Idx	7		Read index
46			Idx	6		
45			Idx	5		
44			Idx	4		
43			Idx	3		
42			Idx	2		
41			Idx	1		
40			Idx	0		
39	2	Reserved	-			Unused
38			-			
37			-			
36			-			
35			-			
34			-			
33			-			
32			-			

Table 34-8 Read response message received by ARH from remote (Remote Handler RX) (Continued)

Bit	Byte	Field	Bits		Value	Description
31	3	Data[31:0]	Data	31		Read result data, If Tsize = '00': then Data[31:24] is used, If Tsize = '01': then Data[31:16] is used, If Tsize = '10': then Data[31:0] is used
30			Data	30		
29			Data	29		
28			Data	28		
27			Data	27		
26			Data	26		
25			Data	25		
24			Data	24		
23	4		Data	23		
22			Data	22		
21			Data	21		
20			Data	20		
19			Data	19		
18			Data	18		
17			Data	17		
16			Data	16		
15	5		Data	15		
14			Data	14		
13			Data	13		
12			Data	12		
11			Data	11		
10			Data	10		
9			Data	9		
8			Data	8		
7	6		Data	7		
6			Data	6		
5			Data	5		
4			Data	4		
3			Data	3		
2			Data	2		
1			Data	1		
0			Data	0		

■ Upstream event message (Remote Handler RX -> Remote Handler TX)

Table 34-9 Event message received by ARH from remote (Remote Handler RX)

Bit	Byte	Field	Bits		Value	Description
55	0	Control[3:0]	Tsize	1	11	Event message received
54			Tsize	0		
53			OAen		-	Offset address enable
52			WRn		-	Read result transaction
51		Reserved	Err			Chip internal bus error
50			-			
49			-			
48			-			
47	1	Idx[7:0]	Idx	7		Event index
46			Idx	6		
45			Idx	5		
44			Idx	4		
43			Idx	3		
42			Idx	2		
41			Idx	1		
40			Idx	0		
39	2	Reserved	-			Reserved
38			-			
37			-			
36			-			
35			-			
34			-			
33			-			
32			-			

Table 34-10 Event message received by ARH from remote (Remote Handler RX) (Continued)

Bit	Byte	Field	Bits		Value	Description
31	3	Data[31:0]	Data	31		Read result data
30			Data	30		
29			Data	29		
28			Data	28		
27			Data	27		
26			Data	26		
25			Data	25		
24			Data	24		
23	4		Data	23		
22			Data	22		
21			Data	21		
20			Data	20		
19			Data	19		
18			Data	18		
17			Data	17		
16			Data	16		
15	5		Data	15		
14			Data	14		
13			Data	13		
12			Data	12		
11			Data	11		
10			Data	10		
9			Data	9		
8			Data	8		
7	6		Data	7		
6			Data	6		
5			Data	5		
4			Data	4		
3			Data	3		
2			Data	2		
1			Data	1		
0			Data	0		

4. Notes on Using ARH

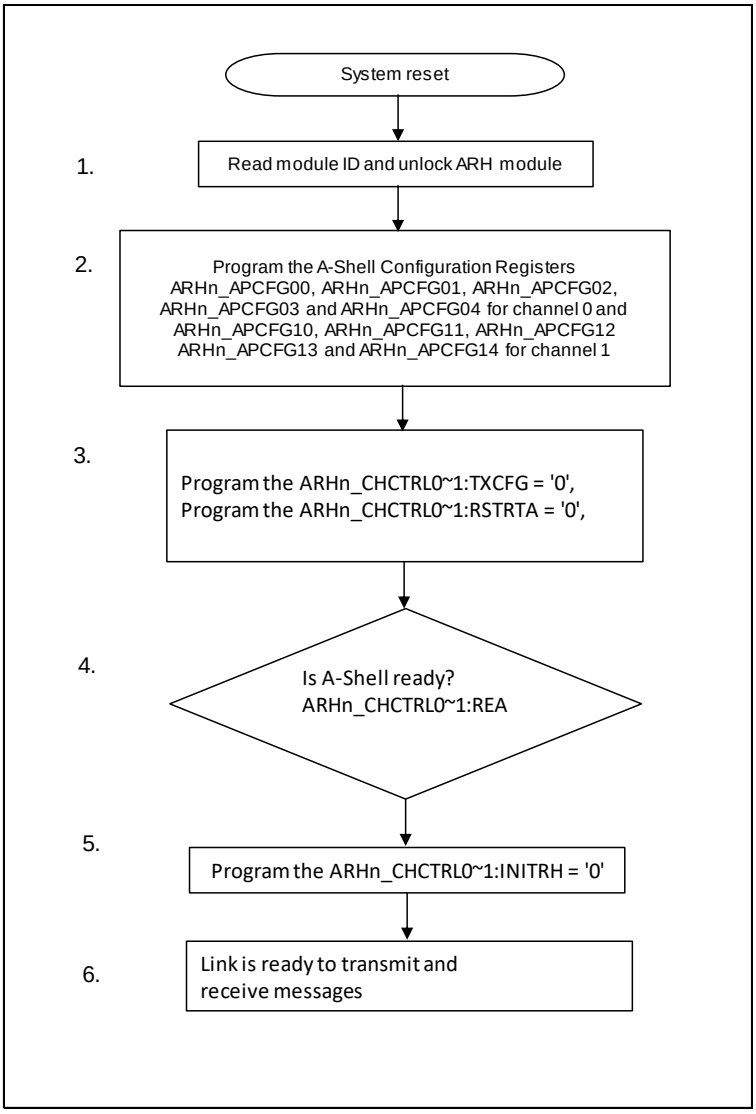
This section is the 'programmer's guide', which lists the usage notes for programming the ARH module. It is recommended to read these guidelines before programming the ARH module.

■ General usage notes

Reserved bits return undefined values. The software programs shall be independent of the values read from the reserved register bits.

■ Steps in programming the ARH module

Figure 34-7: Programmer's flowcharts



- After the system reset the steps in programming the ARH module are as follows:
- The software detects the module ID number of ARH by reading the ARHn_MID register. This helps it in identifying the attributes and capabilities supported by the ARH module. The software unlocks the ARH module by writing proper unlocking key to the ARHn_UNLOCK register. After unlocking the module the software configures ARH by setting appropriate registers.
- The software must program the ARH A-Shell Configuration Registers:
- ARHn_APCFG00, ARHn_APCFG01, ARHn_APCFG02, ARHn_APCFG03 and ARHn_APCFG04 for channel 0
- ARHn_APCFG10, ARHn_APCFG11, ARHn_APCFG12, ARHn_APCFG13 and ARHn_APCFG14 for channel 1
- Program the ARHn_CHCTRL0 to 1:TXCFG bit to '0', this makes the programming of ARH A-Shell Configuration Registers done in step 2 effective. The software should also program the ARHn_CHCTRL0 to 1:RSTRTA bit to '0', this restarts the A-Shell.
- The software can then monitor the A-Shell status bits ARHn_CHCTRL0 to 1:CONNECTED to see if the channel
is established.
- The software must program the ARHn_CHCTRL0 to 1:INITRH bit to '0', this will initialize the Remote Handler.
- The link is ready to transmit and receive messages. The software should now appropriately program the ARH Transaction Buffer Control Registers ARHn_TBCTRL00 to 15, ARH Transaction Frame Registers ARHn_TFCTRL00, ARHn_TFADDR00 to 15, ARHn_TFDATA00, and ARH Event Control Registers ARHn_EVCTRL.

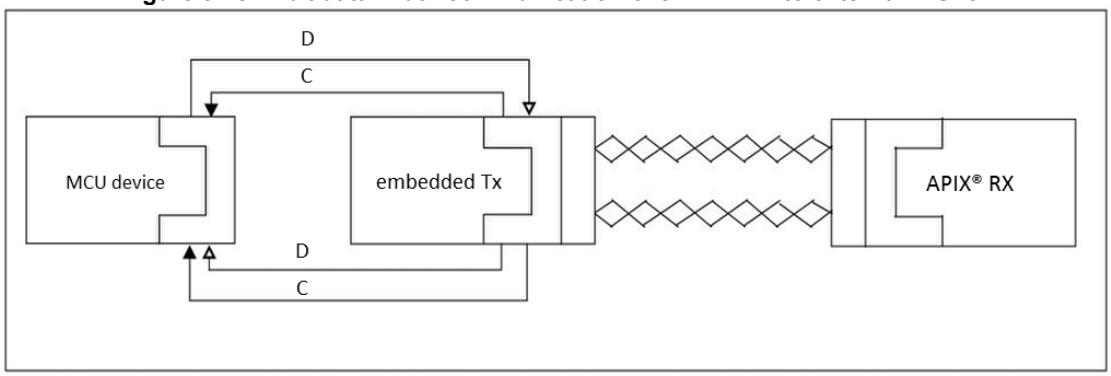
5. Use Cases

This section list various use cases for communication.

- Communication over Automotive Interconnect to external A-Shell

- 1-bit data width

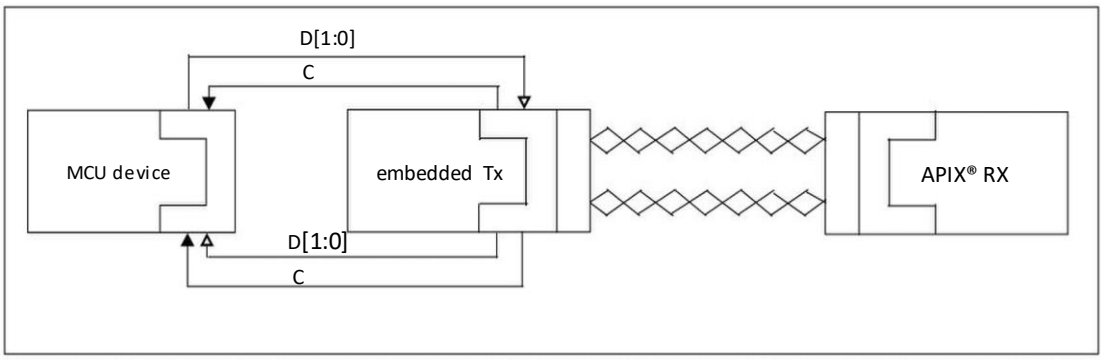
Figure 34-8: 1-bit data width communication over AIC link to external A-Shell



Register	Bit	Default	Value	Description
ARHn_APCFGn1	31	'1'	'0'	'0': Disable data mode/enable pixel stream mode '1': Enable data mode/disable pixel stream mode
ARHn_APCFGn1	29	'1'	'0'	'0': Disable data mode/enable pixel stream mode '1': Enable data mode/disable pixel stream mode
ARHn_APCFGn3	23	'1'	'0'	'0': sbup_data[0] '1': sbup_data[1:0]
ARHn_APCFGn3	21	'1'	'0'	'0': sbdown_data[0] '1': sbdown_data[1:0]
ARHn_APCFGn3	18	'0'	'1'	A-Shell: connect internal A-Shell to external APIX® PHY through GPIO interface '0': Disable '1': Enable

- 2-bit data width

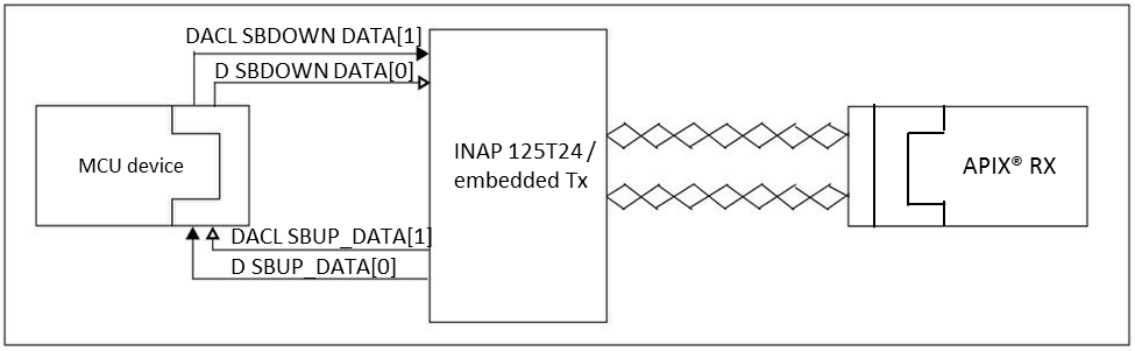
Figure 34-9: 2-bit data width communication over AIC link to external A-Shell



Register	Bit	Default	Value	Description
ARHn_APCFGn1	31	'1'	'0'	'0': Disable data mode/enable pixel stream mode '1': Enable data mode/disable pixel stream mode
ARHn_APCFGn1	29	'1'	'0'	'0': Disable data mode/enable pixel stream mode '1': Enable data mode/disable pixel stream mode
ARHn_APCFGn3	18	'0'	'1'	A-Shell: connect internal A-Shell to external APIX® PHY through GPIO interface '0': Disable '1': Enable

■ Communication over Automotive Interconnect to external PHY

Figure 34-10: Communication over AIC link to external PHY

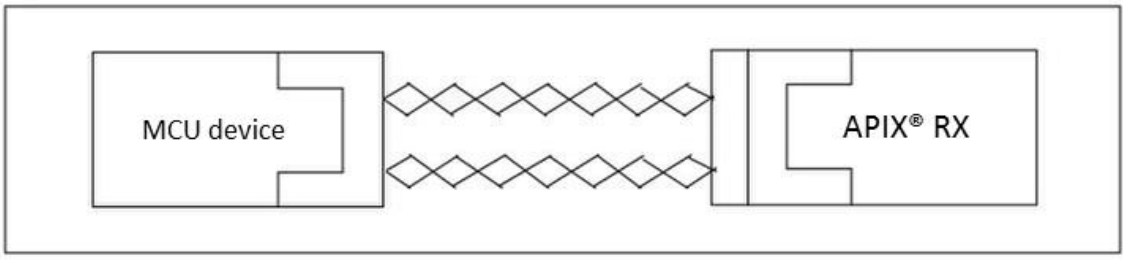


Register	Bit	Default	Value	Description
ARHn_APCFGn1	31	'1'	'0'	'0': Disable data mode/enable pixel stream mode '1': Enable data mode/disable pixel stream mode
ARHn_APCFGn1	29	'1'	'0'	'0': Disable data mode/enable pixel stream mode '1': Enable data mode/disable pixel stream mode
ARHn_APCFGn3	22	'0'	'1'	A-Shell: validate sbup_data with '0': sbup_valid '1': sbup_data[1]
ARHn_APCFGn3	20	'0'	'1'	A-Shell: generate sbdown clock and transmit as sbdown_data[1] '00': Disable '01': With use of sbdown_trigger (synchronous to core_clk of APIX® PHY) '10': With use of internal counter (asynchronous to core_clk of APIX PHY) '11': Disable
	19	'0'	'0'	
ARHn_APCFGn3	18	'0'	'1'	A-Shell: connect internal A-Shell to external APIX® PHY through GPIO interface '0': Disable '1': Enable

Register	Bit	Default	Value	Description
ARHn_APCFGn3	2	'0'	<configurable>	A-Shell: Configures cycle time of sbdown clock (multiples of A-Shell core clock) when sbdown_data are asynchronous (sbdown_data[1] is used as sbdown clock) or cfg_spi_over_sb is enabled. 0x0B: Recommended minimum (no low bandwidth mode, A-Shell and APIX® PHY operate at same core clock frequency) 0x14: Recommended minimum (low bandwidth mode 2, A-Shell, and APIX® PHY operate at 62.5 MHz) 0x26: Recommended minimum (low bandwidth mode 1, A-Shell, and APIX® PHY operate at 62.5 MHz).
	1	'0'		
	0	'0'		
	30	'0'		
	29	'1'		
	28	'0'		
	27	'0'		
	26	'1'		
	25	'1'		
	24	'0'		

- Communication over APIX® link to External PHY

Figure 34-11: Communication over APIX® link to external PHY



- Downlink over Pixelchannel

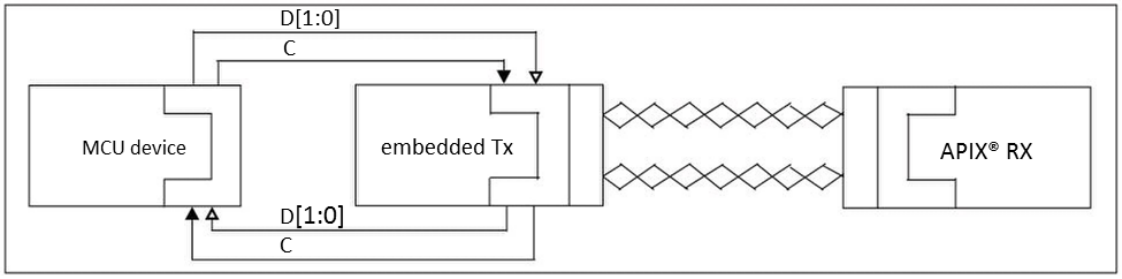
Downlink over Pixelchannel is provided by default configuration.

- Downlink over Sidebandchannel

Register	Bit	Default	Value	Description
APCFG01	31	1	0	0: Disable data mode/ Enable pixel stream mode 1: Enable data mode/ Disable pixel stream mode
APCFG03	18	1	0	0: Disable the connection of internal A-Shell to external APIX® PHY through GPIO interface 1: Enable the connection of internal A-Shell to external APIX® PHY through GPIO interface

■ Communication over Automotive Interconnect-2 to External Automotive Interconnect-2

Figure 34-12: Communication over Automotive Interconnect-2 to External Automotive Interconnect-2



Register	Bit	Default	Value	Description
APCFGn4	1	0	1	0:A-Shell is connected to APIX® PHY 1:A-Shell is connected to AIC2

Note:

– *If n = 0 indicates channel 0, n = 1 indicates channel 1.*

CHAPTER 35: Base Timer

This chapter explains the functions and operations of the base timer.

1. Overview
2. Registers

CODE: BT-S6J3300-E1

1. Overview

The base timer and its supervised target are described here.

See the standard information and specification of Traveo™ Platform Hardware Manual

2. Registers

See the chapter of BASE TIMER in Traveo™ Platform Hardware Manual.

Here, the difference in the offset addresses is indicated.

Table 35-1 Offset Addresses

CH No.	Offset_Address (Common PERI #0)	CH No.	Offset_Address (Common PERI #0)	CH No.	Offset_Address (Common PERI #0)	CH No.	Offset_Address (Common PERI #0)
24	0xB484_6000	36	0xB484_9000	48	0xB484_C000	60	0xB484_F000
25	0xB484_6400	37	0xB484_9400	49	0xB484_C400	61	0xB484_F400
26	0xB484_6800	38	0xB484_9800	50	0xB484_C800	62	0xB484_F800
27	0xB484_6C00	39	0xB484_9C00	51	0xB484_CC00	63	0xB484_FC00
28	0xB484_7000	40	0xB484_A000	52	0xB484_D000		
29	0xB484_7400	41	0xB484_A400	53	0xB484_D400		
30	0xB484_7800	42	0xB484_A800	54	0xB484_D800		
31	0xB484_7C00	43	0xB484_AC00	55	0xB484_DC00		
32	0xB484_8000	44	0xB484_B000	56	0xB484_E000		
33	0xB484_8400	45	0xB484_B400	57	0xB484_E400		
34	0xB484_8800	46	0xB484_B800	58	0xB484_E800		
35	0xB484_8C00	47	0xB484_BC00	59	0xB484_EC00		

d

Table 35-2 Register Map (Channel No.: 0) (12) (24) (36) (48) (60)

Offset	Register Name/Initial Value			
	+3	+2	+1	+0
0x0000_0000	Reserved 00000000_00000000		BTxx_PCSR/BTxx_PCSR/BTxx_PRLL /Reserved XXXXXXXX_XXXXXXXX	
0x0000_0004	Reserved 00000000_00000000		Reserved/BTxx_PDUT/ BTxx_PRLH/ BTxx_DTBF XXXXXXXX_XXXXXXXX /00000000_00000000 (BTxx_DTBF)	
0x0000_0008	Reserved 00000000_00000000		BTxx_TMR/ BTxx_TMR/ BTxx_TMR/Reserved 00000000_00000000 XXXXXXXX_XXXXXXXX*	
0x0000_000C	Reserved 00000000_00000000		BTxx_TMCR 00000000_00000000	
0x0000_0010	Reserved 00000000_00000000		BTxx_TMCR2 00000000	BTxx_STC 00000000
0x0000_0014	Reserved 00000000_00000000		Reserved 00000000	BTxx_STCC 00000000
0x0000_0018	Reserved 00000000_00000000		Reserved 00000000	BTxx_STCS 00000000
0x0000_0030	Reserved 11111111_11111111		Reserved 11111111	BT_BTSEL01 1111_0000
0x0000_0034	Reserved 11111111_11111111		BT_BTSSSR0/12/24/36/48/60 11111111_11111111	
0x0000_0038	Reserved 11111111_11111111		BT_BTTRR0/12/24/36/48/60 11110000_00000000	

* The initial value is XXXXXXXX_XXXXXXXX only during reload timer operation.

Table 35-3 Register Map (Channel No.: 1, 3, 5, 7, 9, and 11) (13, 15, 17, 19, 21, and 23) (25, 27, 29, 31, 33, and 35) (37, 39, 41, 43, 45 and 47) (49, 51, 53, 55, 57 and 59) (61, and 63)

Offset	Register Name/Initial Value			
	+3	+2	+1	+0
0x0000_0000	Reserved 00000000_00000000		BTxx_PCSR/ BTxx_PCSR/ BTxx_PRLL /Reserved XXXXXXXX_XXXXXXXX	
0x0000_0004	Reserved 00000000_00000000		Reserved/ BTxx_PDUT/ BTxx_PRLH/ BTxx_DTBF XXXXXXXX_XXXXXXXX /00000000_00000000(BTxx_DTBF)	
0x0000_0008	Reserved 00000000_00000000		BTxx_TMR/ BTxx_TMR/ BTxx_TMR/ Reserved 00000000_00000000 /XXXXXXXX_XXXXXXXX*	

Offset	Register Name/Initial Value			
	+3	+2	+1	+0
0x0000_000C	Reserved 00000000_00000000		BTxx_TMCR 00000000_00000000	
0x0000_0010	Reserved 00000000_00000000		BTxx_TMCR2 00000000	BTxx_STC 00000000
0x0000_0014	Reserved 00000000_00000000		Reserved 00000000	BTxx_STCC 00000000
0x0000_0018	Reserved 00000000_00000000		Reserved 00000000	BTxx_STCS 00000000
0x0000_001C	Reserved 00000000_00000000		Reserved/BTxx_PSDR/Reserved/Reserved 00000000_00000000	
0x0000_0020	Reserved 00000000_00000000		Reserved/BTxx_ADTR/Reserved/Reserved 00000000_00000000	

* The initial value is XXXXXXXX_XXXXXXXX only during reload timer operation.

Table 35-4 Register Map (Channel No.: 2, 4, 6, 8, and 10) (14, 16, 18, 20, and 22) (26, 28, 30, 32, and 34) (38, 40, 42, 44 and 46) (50, 52, 54, 56 and 58) (62)

Offset	Register Name/Initial Value			
	+3	+2	+1	+0
0x0000_0000	Reserved 00000000_00000000		BTxx_PCSR/ BTxx_PCSR/ BTxx_PRLL /Reserved XXXXXXXX_XXXXXXXX	
0x0000_0004	Reserved 00000000_00000000		Reserved/ BTxx_PDUT/ BTxx_PRLH / BTxx_DTBF XXXXXXXX_XXXXXXXX /00000000_00000000(BTxx_DTBF)	
0x0000_0008	Reserved 00000000_00000000		BTxx_TMR/ BTxx_TMR/ BTxx_TMR /Reserved 00000000_00000000 / XXXXXXXX_XXXXXXXX*	
0x0000_000C	Reserved 00000000_00000000		BTxx_TMCR 00000000_00000000	
0x0000_0010	Reserved 00000000_00000000		BTxx_TMCR2 00000000	BTxx_STC 00000000
0x0000_0014	Reserved 00000000_00000000		Reserved 00000000	BTxx_STCC 00000000
0x0000_0018	Reserved 00000000_00000000		Reserved 00000000	BTxx_STCS 00000000
0x0000_0030	Reserved 11111111_11111111		Reserved 11111111	BT_BTSEL23/45 /67/89/1011 1111_0000

* The initial value is XXXXXXXX_XXXXXXXX only during reload timer operation.

CHAPTER 36: Base Timer Port Definition

This chapter explains BASE TIMER port definition.

1. Overview
2. Definition
3. Supplementation
4. Operation and Registers

CODE: BTPORDEF-S6J3300-E1

1. Overview

This product series has maximum 32 units, and total 64 channels of base timer.

A unit of 2 channels of base timer can be controlled and applied as well as individual operation of each timer channel, and all the modes of base timer operation are described and defined in the chapter of BASE TIMER in Traveo™ Platform Hardware Manual.

This series particularly has package external port names and function port names of base timer function input or output instead of the function port names which are defined in Traveo™ Platform Hardware Manual, and it sometime brings about confusion for software design to understand the relations.

Here, this chapter aims to describe and clarify the followings.

- Unit and channel number definition
- Port name definition and supported operation mode

Again, the description of base timer function, block diagram, operation, and register should be referred in Traveo™ Platform Hardware Manual in detail.

2. Definition

2.1. Unit and Channel number definition

Algebra	Definition	Relation	Remark
u	Unit number	u = 0, 1,,,32	0 to 32
m	Channel number	m = 2u	0 to 63
n	Channel number	n = 2u+1=m+1	1 to 64

In this document Base timer is named as Base timer "u" with its unit number.

2.2. Port name definition and Supported operation mode

The function port name is described in Traveo™ Platform Hardware Manual.

■ I/O mode 0: 16-bit timer standard mode

Function port name in Traveo™ Platform Hardware Manual	Series particular port name	Remark
TIOBn	PPGu_TIN3	See RIC table.
TIOAn	PPGu_TOUT2	See port description or POF table
TIOBm	PPGu_TIN1	See RIC table.
TIOAm	PPGu_TOUT0	See port description or POF table

■ I/O mode 1: 32-bit timer full-function mode

Function port name in Traveo™ Platform Hardware Manual	Series particular port name	Remark
TIOBn	PPGu_TIN3	See RIC table.
TIOAn	PPGu_TIN2	See RIC table.
TIOBm	PPGu_TIN1	See RIC table.
TIOAm	PPGu_TOUT0	See port description or POF table

- I/O mode 2: PPG trigger 2-channel sharing mode

Function port name in Traveo™ Platform Hardware Manual	Series particular port name	Remark
TIOBn	Not used	-
TIOAn	PPGu_TOUT2	See port description or POF table
TIOBm	PPGu_TIN1	See RIC table.
TIOAm	PPGu_TOUT0	See port description list

- I/O mode 4: Timer start/stop mode

Function port name in Traveo™ Platform Hardware Manual	Series particular port name	Remark
TIOBn	Not used	-
TIOAn	PPGu_TOUT2	See port description or POF table
TIOBm	PPGu_TIN1	See RIC table.
TIOAm	PPGu_TOUT0	See port description or POF table

- I/O mode 5: Simultaneous soft start mode

Function port name in Traveo™ Platform Hardware Manual	Series particular port name	Remark
TIOBn	Not used	-
TIOAn	PPGu_TOUT2	See port description or POF table
TIOBm	Not used	-
TIOAm	PPGu_TOUT0	See port description or POF table

- I/O mode 6: Timer start/stop and simultaneous soft start mode

Function port name in Traveo™ Platform Hardware Manual	Series particular port name	Remark
TIOBn	Not used	-
TIOAn	PPGu_TOUT2	See port description or POF table
TIOBm	Not used	-
TIOAm	PPGu_TOUT0	See port description or POF table

- I/O mode 7: Timer start mode

Function port name in Traveo™ Platform Hardware Manual	Series particular port name	Remark
TIOBn	Not used	-
TIOAn	PPGu_TOUT2	See port description or POF table
TIOBm	PPGu_TIN1	See RIC table.
TIOAm	PPGu_TOUT0	See port description or POF table

Note:

– *See RIC (Resource Input Configuration), POF (port output function configuration), and Port description list in this Hardware Manual.*

3. Supplementation

3.1. Input port configuration

PPGu_TIN1, PPGu_TIN2, and PPGu_TIN3 are selected as input port of base timer by configuring RIC register. See the chapter of "PORT CONFIGURATION".

3.2. Output port configuration

PPGu_TOUT0 and PPGu_TOUT2 are selected as output port of base timer by configuring POF register. See the chapter of "PORT CONFIGURATION".

3.3. Channel Group configuration

The input terminals of Base Timer are classified as follows.

PPG0/1/2/3/4/5_TIN1 is the input terminal for CH0 to CH11.
PPG6/7/8/9/10/11_TIN1 is the input terminal for CH12 to CH23.
PPG12/13/14/15_TIN1 is the input terminal for CH24 to CH31.
PPG16/17/18/19/20/21_TIN1 is the input terminal for CH32 to CH43.
PPG22/23/24/25/26/27_TIN1 is the input terminal for CH44 to CH55.
PPG28/29/30/31_TIN1 is the is the input terminal for CH56 to CH63.

The BT_SSSR is the register which sets the base timer channel to be started simultaneously by “1” pulse input of the above terminal.
These channels are classified as following.

BT_SSSR0 register is for CH0 to CH11.
BT_SSSR12 register is for CH12 to CH23.
BT_SSSR24 register is for CH24 to CH35.
BT_SSSR36 register is for CH36 to CH47.
BT_SSSR48 register is for CH48 to CH59.
BT_SSSR60 register is for CH60 to CH63.

The channel group of Base Timer which can put the trigger by the external input terminal is not same as the channel
group which can be started simultaneously by the setting the BT_SSSR register.

If the channels from 24 to 35 are started simultaneously, the BT_SSSR24 register is set by “FFF” and both “PPG12/13/14/15_TIN1” and “PPG16/17/18/19/20/21_TIN1” should be asserted at same time.

4. Operation and Registers

See the chapter of BASE TIMER in Traveo™ Platform Hardware Manual.

CHAPTER 37: LCD Bus Interface

This chapter explains the LCD Bus Interface.

1. Preface
2. Function
3. Application

CODE: LCDBUS-S6J3300-E1

1. Preface

1.1. Purpose and Scope

This manual explains the functions and configurations of Cypress' LCDBusIF. It is intended for engineers engaged in the actual development of related products.

The LCDBusIF is an interface IP to communicate to external LCD controllers with an embedded graphic ram (GRAM). Its feature set is targeted to simplify the process of setting the LCD controllers configuration and updating the display content. It is applicable for Consumer, Industrial and Automotive market.

Note that there might be use cases documented in this manual that cannot be used in the context of a certain MCU, for example, because not all HW ports are connected to external pins or to the main system. So it is essential to check the related MCU manual accordingly.

1.2. Document Structure

The manual is divided into two main sections:

- Chapter **Function** contains pure functional descriptions only. It is a detailed documentation from feature point of view, without any details regarding setup and configuration.
- Chapter **Application** is a guideline for SW programmers. It describes which sub modules to setup and how to configure them for certain use cases. Also it contains a complete register list for all components (SW interface), interrupt descriptions and the address map.

Both chapters only cover aspects specific to this LCDBusIF IP. The related use cases, however, also require certain configurations in the embodying MCU system, for example, clock generation and pin multiplexing. For these refer to the relevant MCU manual.

2. Function

2.1. Feature Summary

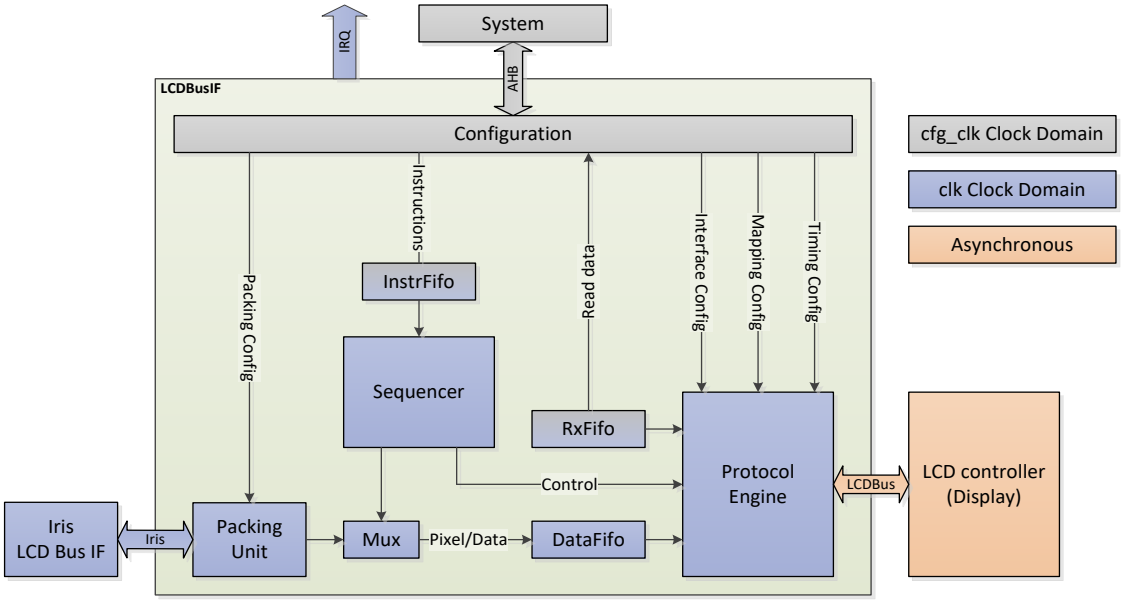
The LCDBusIF implements the following features:

- Interfaces to LCD controller ICs with an Intel-8080 or Motorola-6800 compatible LCDBus[1] with an 8-, 9-, 16- or 18-bit wide data bus
- Direct link from Iris graphics core
  - Wide range of input color formats (RGBA, Grayscale, Color Indexed or Compressed)
  - Memory efficient (sparse) scene composition
- LCD display and configuration access is controlled via a programmable command sequencer to load off the CPU
- Dedicated instruction to support content update from CPU or SW DMA
- AHB slave interface for configuration, command lists and pixel data
- Programmable interrupts (embedded in command lists) for synchronization with the CPU
- Tearing effect (VSYNC) interrupt (if supported by display)
- Support for different display pixel formats (like RGB666, RGB565, RGB444, GREY, B/W)

2.2. Block Diagram

In Figure 37-1 the LCDBusIF together with its external interfaces is depicted. Inside the conceptual structure and data flow is shown. The colors illustrate the different clock domains.

Figure 37-1: LCDBusIF Block Diagram



2.3. Functional Limitations

In Table 37-1 the Functional Limitations of the LCDBusIF are listed.

[1] In this document the parallel interface towards the LCD controller is refered to as LCDBus.

Table 37-1: Functional Limitations

Maximum resolution	WQVGA (480x272)
Maximum write cycle[2] frequency	clk/3 (clock frequency of the module divided by 3)

Additionally to the limitations from Table 37-1 certain display features can't be mapped with the current architecture:

- Any color format which requires color components from more than one pixel being transferred in one external interface word (*two pixel every three cycles mode*, used for example for RGB666 over 16-bit or RGB444 over 8-bit by some displays) won't be supported when pixel data is received from Iris graphics core.
- Some displays require manual increment of page address during content update (they only increment the column address). These displays won't be supported when pixel data is received from Iris graphics core.
- Additional address lines (to access GPIO or status information) offered by some displays are not supported.

2.4. Nomenclature

The market offers a wide variety of LCD controllers with an equally high variation of naming conventions. This section should define the nomenclature used throughout this document and maps it to names found on the market.

- **LCDBus:** describes the parallel data and control interface towards the LCD controller. The LCD controllers usually refers to it as the *Host Interface*. The LCDBus can adhere to different protocols. The LCDBusIF supports the Intel-8080 (I80) and Motorola-6800 (M68) protocols.
- **GRAM:** the Graphic RAM describes the memory in the LCD controller which stores the pixel data shown on the display.
- **Iris:** is the nickname of the 2-D graphics core integrated into many MCUs provided by Cypress.
- **Index Register:** the LCDBus protocols implement an indirect addressing where the address is first set with a command write access and the actual register contents are accessed with one or more data accesses subsequently. The Index Register is the name of the register storing the address.
- **Status Register:** some LCD controllers provide a status register which can be read with a command read access.
- **Data Register:** while the Index Register refers to the address of the current access, the Data Register refers to the contents referenced by the Index Register.
- **Tearing Effect:** synchronization signal some LCD controller provide to synchronize the GRAM update with the start of the vertical blanking period of their internal timing controller. This should allow to avoid tearing artefacts on the display due to changing GRAM contents.
- **Signal Names:** while the LCDBus protocols define clear roles for the signals involved, the naming of them is not standardized and variations exist. Especially the active low property is encoded by different means (SIG#, SIGx, xSIG, SIGn, nSIG, $\overline{\text{SIG}}$).In this specification the number sign (#) is used to encode an active-low property. Table 37-2 lists the signal names used in this document together with some aliases seen in LCD controller specifications.

[2] A write cycle refers to the minimum time between two write accesses on the LCD Bus and is usually defined in the AC characteristic of the display specification.

Table 37-2: Signal names and Typical Aliases Found in LCD Controller Specifications

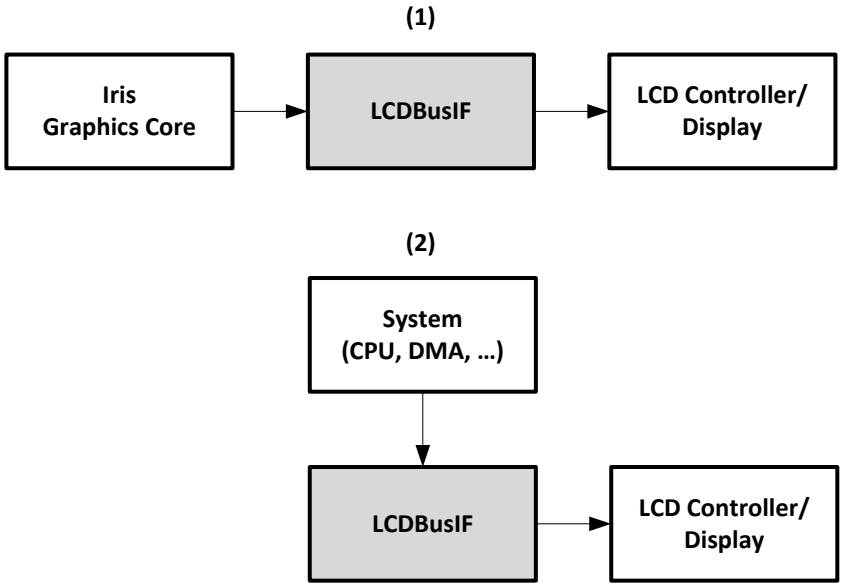
Signal	Description	LCD Controller Pin Names	Protocols
CS#	Chip select (active low)	nCS, CSn, CSx, XCS, CS#	I80, M68
WR#	Write select (active low)	nWR, WRn, WRx, XWR, WR#	I80
R/W#	Read/Write select (read = high, write = low)	R/Wn, R/Wx, R/W#	M68
RD#	Read select (active low)	nRD, RDn, RDx, XRD, RD#	I80
E	Enable	E	M68
D/C#	Data/Command select (data = high, command = low)	A0, RS, P/C#, D/C#	I80, M68
DB	Data Bus	DB, D, DAT	I80, M68
TE	Tearing Effect	TE, FMARK	I80, M68
RES#	Reset (active low)	RESET, RESX, XRES, RES#	I80, M68

2.5. Use Cases

The LCDBusIF acts as communication interface towards external LCD controllers. As such it allows updating the display content, but it is also required to set configuration register or read status registers of the LCD controller. The main use cases differentiate by the source of the display content, see Figure 37-2.

1. Content from Iris Graphics Core: the display scene is composed by the Iris graphics core, converted to the image format of the LCD controller and written to it. This is the preferred use case for systems having an Iris graphics core.
2. Content from system: the display scene is written to the LCDBusIF from any system master. This could for example be the CPU rendering content on the fly or a DMA copying an available display scene from memory.

Figure 37-2: Use Cases from a Data Flow Perspective



2.6. Protocol Engine

The protocol engine of the LCDBusIF is designed to support the different protocol flavors and characteristics as they are used by LCD controllers on the market. Their variance is normally determined by the following three factors:

1. Protocol Types: the LCDBusIF supports Intel-8080 (sometimes also called System-80) and Motorola-6800 kind of interfaces with different data widths and flavors regarding the read/write strobe signaling.
2. Access Timing: above mentioned protocols are all asynchronous, so to guarantee a stable communication timing characteristics need to be considered. The LCDBusIF implements these timing requirements by generating the output signals with a raster defined by module clock cycles.
3. Data Mapping: depending on the available LCDBus data width and the pixel format used internally the LCD controller might require several access cycles to read/write one pixel (e.g. an RGB565 pixel is transferred using 2 access cycles on an 8-bit interface)

The subsequent sections will detail how the LCDBusIF handles these variations.

2.6.1. Protocol Types

The LCDBusIF can support different LCDBus interface configurations. It supports Intel-8080 and Motorola-6800 type interfaces of different data bus widths. Additionally can either the RD#/WR#/E (Type 1) or the CS# (Type 2) signals be used as strobe signal while the other functions as select signal. Figure 37-3 and Figure 37-4 illustrate the different LCDBus protocol variants supported. See Table 37-3 for the different configuration options available. Note the different interpretation of the read/write interface signals depending on the interface dialect. Additionally the polarity of each interface signal can be configured to support LCD controllers having a special interpretation of the protocols.

Table 37-3: Interface Configuration Options

Interface	Variant	Strobe signal(s)	Select signal(s)
Intel-8080	Type 1	WR#, RD#	CS#
	Type 2	CS#	WR#, RD#
Motorola-6800	Type 1	E	CS#
	Type 2	CS#	E

Figure 37-3: Intel-8080 LCDBus Protocol Variants

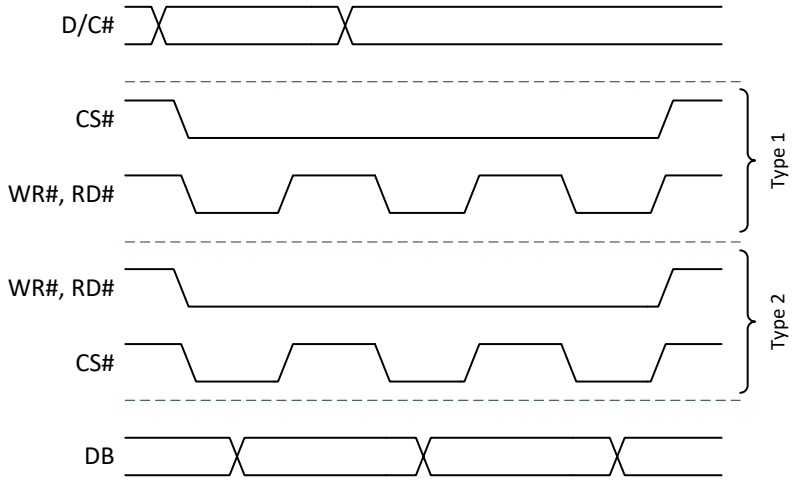
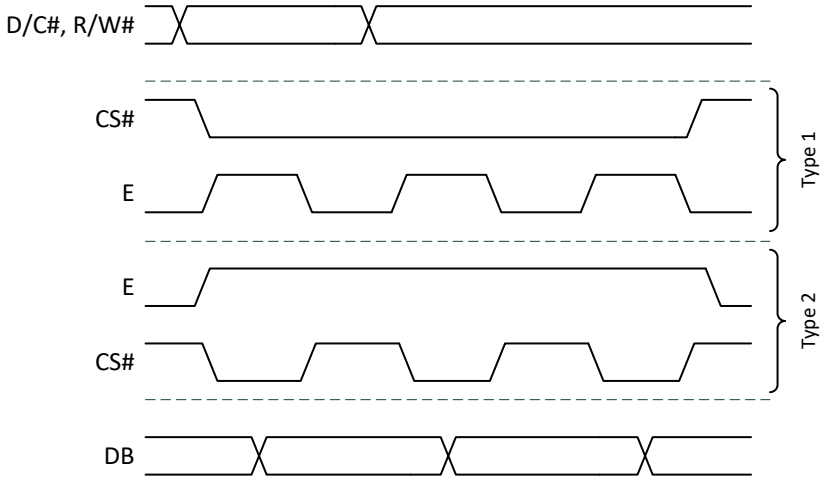


Figure 37-4: Motorola-6800 LCDBus Protocol Variants



2.6.2. Access Timing

The LCD controllers have different requirements for read and write timing usually described in the AC characteristics section of their specification. The LCDBusIF allows to adhere to these timing requirements by implementing a read and write cycle timing scheme. Within each cycle three windows are defined, the setup window, the active window and the hold window. The width of the window is defined in number of module clock cycles. The windows have the following functional meanings:

- Setup window: activate select, data/command and read/write select signals; control output enable according to the access direction
- Active window: update data bus for write access; active period for strobe signal

- Hold window: register data bus for read access; hold data bus and output enable for write access

Figure 37-5 below illustrates the timing for a command and data write sequence and Figure 37-6 the timing for a command write and data read sequence.

Note that the maximum pixel clock the LCDBusIF can support without causing tearing artefacts depend on the module clock and how well it can be matched to the required interface timing.

Figure 37-5: Illustration of Command/Register Write Sequence

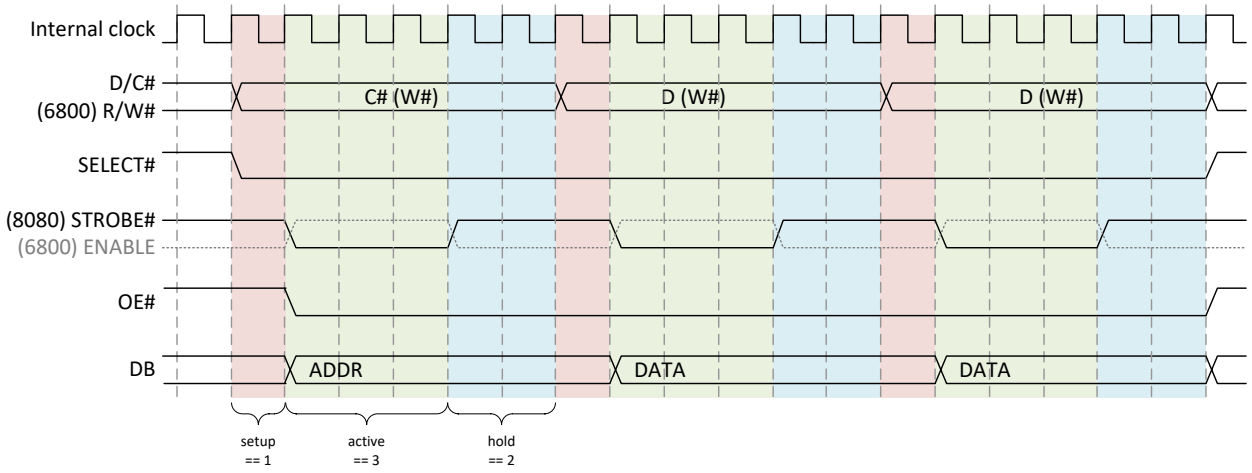
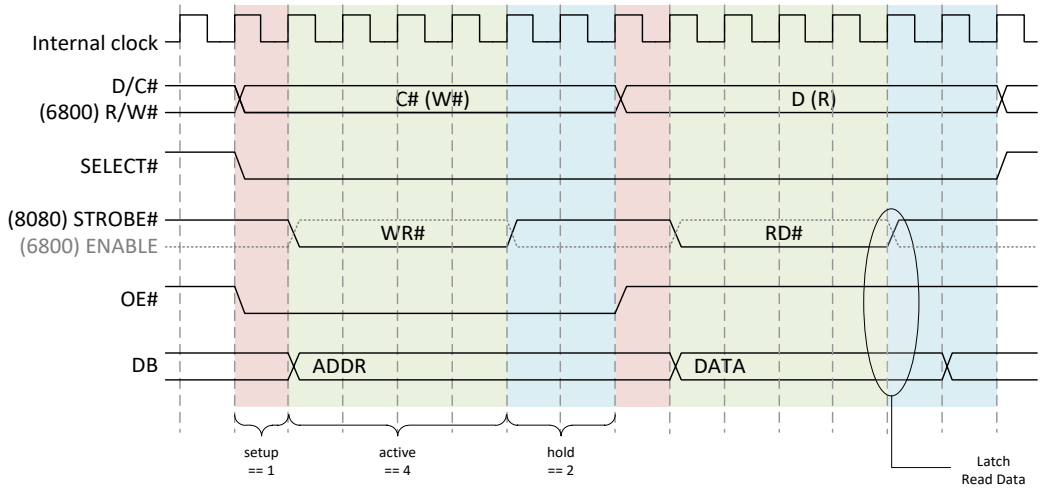


Figure 37-6: Illustration Command Write/Register Read Sequence



2.6.3. Data Mapping

The mapping configuration instructs the LCDBusIF how to transfer a given command or data word with the given interface width. Distinct transfer mappings can be programmed for a data or a command transfer. The LCD controller might have a native command/data width which is not equal to the interface width. Hence several transfers might be required (if the interface width is smaller than the command/data width) or the data might need to be properly aligned (if the interface width is bigger than the command/data width).

In Table 37-4 the mapping of different color formats depending on the interface width is illustrated. The number signifies the number of transfers required to transport one pixel via the corresponding interface width (the numbers in parenthesis should indicate that such a combination of interface width and color format is relatively unlikely).

Table 37-4: Correlation Between Interface Width and Color Format

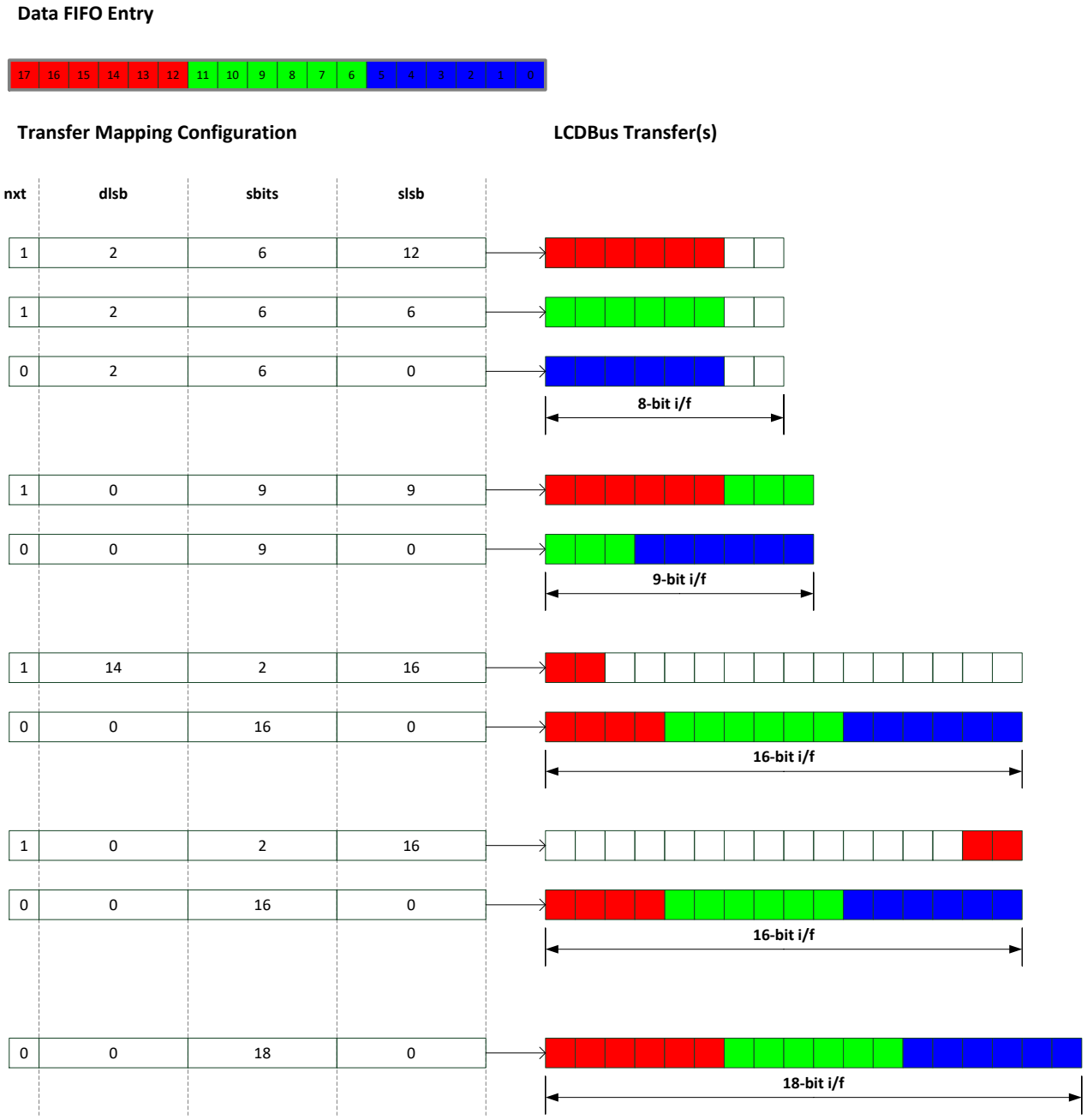
Interface Width / Color Format	18-bit	16-bit	9-bit	8-bit
Black/White	(1)	(1)	(1)	1
Gray 4-bit	(1)	(1)	(1)	1
Gray 8-bit	(1)	(1)	(1)	1
RGB565	1	1	2	2
RGB666	1	2	2	3
RGB888	2	2	3	3

Most LCD controllers have different requirements for access of their internal registers and access of the GRAM. Therefore the LCDBusIF provides one set of mapping registers for each access type (named command access and data access respectively).

The mapping is implemented as such that one command/data word (source) can be spread over up to three LCDBus (destination) transfers. For every transfer one register defines a vector to be extracted from the source and put into the destination transfer. The source vector is determined by a bit offset (slsb) and a number of bits (sbits). The position within the destination transfer is determined by another bit offset (dlsb). An additional bit (next) indicates if a subsequent destination transfer is required for this command/data word (next = 1) or if the protocol engine is done with it (next = 0). Figure 37-7 illustrates different mapping configurations to transport an RGB666 word via different interface widths.

For read transfer mapping the interpretation of source and destination needs to be swapped, i.e. dlsb will select the start bit of the read transfer received, dbits will define how many bits to extract and slsb will define where the data should be placed into the reception fifo word.

Figure 37-7: Different Mapping Scenarios for an RGB666 Color Format via Different Interface Widths



2.7. Iris Graphics Core Interface

The Iris graphics core, if available in the system, already provides a strong feature set to compose a display scene in a high quality and resource efficient way. In order to benefit from this the LCDBusIF offers an interface to the Iris graphics core that allows to request frames and convert the pixel information received into the color format required by the LCD controller.

2.7.1. Handshake

The image source from the Iris graphics core can be selected among the available content or safety streams in the system. The selected stream must be setup to compose the desired display scene with the correct frame dimension before the LCDBusIF should update the display content. The frame transaction is initiated from the LCDBusIF by sending a frame request (see IRIS_KICK instruction) to the Iris. Subsequently the LCDBusIF might stall the Iris until the display update is initiated (see IRIS_FRAME instruction). The Iris graphics core must not send any data without a frame request.

2.7.2. Data Packing

The packing configuration instructs the LCDBusIF how to transform the RGBA data received from the Iris interface to the color format required by the LCD controller. Input color is interpreted as 24-bit RGB. The configuration will define how many bits of each color component will be used (starting from most significant bit) and at which position it should be mapped into the transfer word.

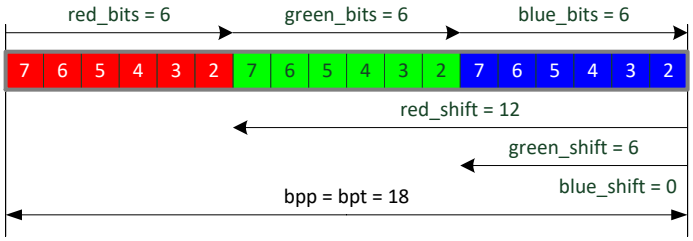
For color formats requiring less bits per pixel as available by the interface width multiple pixels can be packed into one transfer. This is often required for B/W or grayscale displays. A transfer in this context specifies an entry in the DataFifo. The LCDBusIF will pack as many complete pixel as possible into one transfer. Figure 37-8 illustrates some common packing configurations.

Figure 37-8: Packing Configurations for Different LCD Color Formats

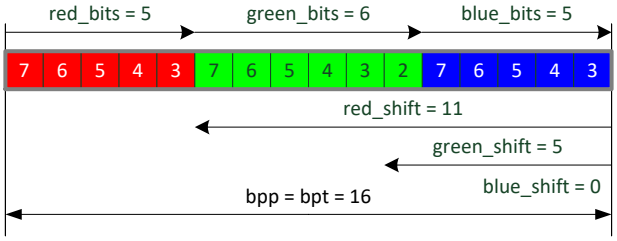
24-bit per pixel (Iris External Interface)



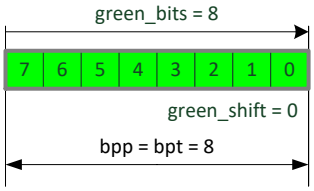
LCDBusIF Packing



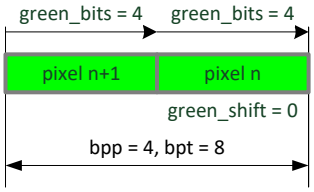
RGB666



RGB565



GREY8



GREY4

(bpp = Bits per pixel, bpt = Bits per transfer)

2.8. Command Sequencer

The operation of the LCDBusIF is controlled by a programmable command sequencer. A set of instructions are provided to allow the sequencer to synchronize and control the data flow between CPU, Iris graphics core and the LCD controller.

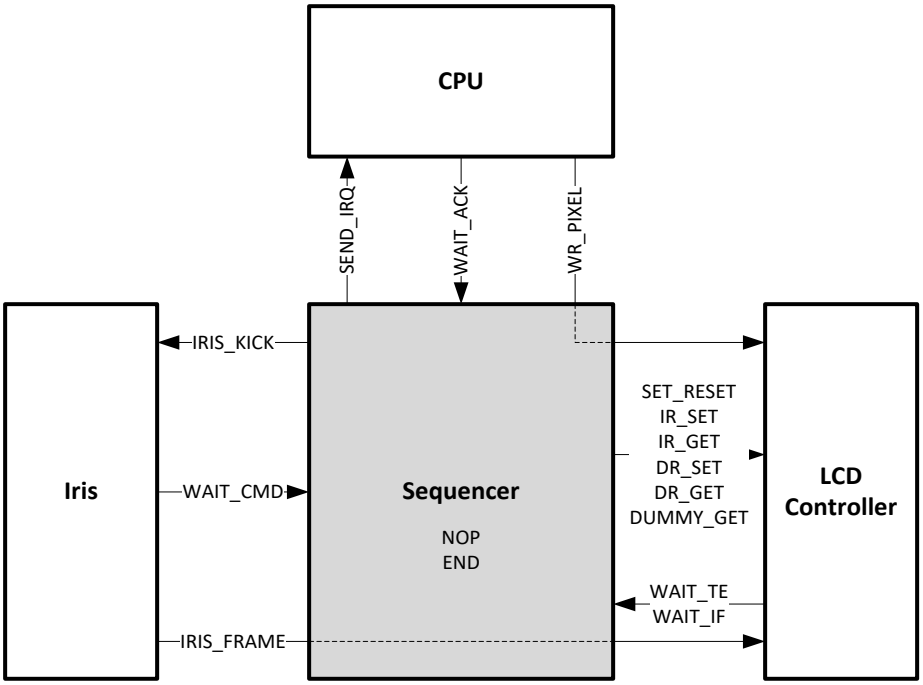
Table 37-5 illustrates the instruction format of the command sequencer. Each instruction is a 32-bit word whose most significant byte defines the type (opcode) of the instruction. Additional parameters can be encoded in the lower 3 bytes. Most instructions are self contained in a single 32-bit word, only the WR_PIXEL instruction is followed by a list of 32-bit words which contain the pixel data to be written to the LCD controller.

Table 37-5: Command Sequencer Instruction Format

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
Opcode								Parameters																							
Optional: pixel data																															

In Figure 37-9 the different instructions with their relation to the CPU, the Iris graphics core and the LCD controller are shown. Arrows to the sequencer indicate that the sequencer is synchronizing to an event from this source, arrows from the sequencer indicate that the sequencer is sending a request to this destination and arrows through the sequencer indicate the routing of the pixel stream (either from CPU or Iris graphics core) to the LCD controller.

Figure 37-9: Conceptual Overview of Sequencer Interaction with CPU, Iris and LCD Controller



In Table 37-6 the supported instructions together with their opcode, parameters and description are summarized. See Instruction Set for a more elaborate description.

Table 37-6: Summary of Instructions Supported by Sequencer

Instruction	Opcode	Description	Parameter(s)
NOP/WAIT	0x00	Wait cycle(s) (module clock)	count[23:0]
IR_SET IR_GET DR_SET DR_GET	0x10 0x11 0x12 0x13	Write index register Read status register Write data register Read data register	addr[17:0] count[17:0] data[17:0], type[param,pixel] count[17:0], type[param,pixel]
DUMMY_GET WAIT_TE WAIT_IF SET_RESET	0x14 0x15 0x16 0x1F	Dummy read Wait for sync input from display Wait until protocol engine is idle Set LCDBus RES# signal value	 level[low/high]
SEND_IRQ WAIT_ACK	0x20 0x21	Send synchronization interrupt to CPU Wait for acknowledge from CPU	
IRIS_KICK WAIT_CMD IRIS_FRAME	0x40 0x41 0x42	Send kick (frame request) to Iris Wait for command word at Iris interface Forward pixel data from Iris interface	
WR_PIXEL	0x80	Initiate writing of pixel data via instruction fifo	bpp[2:0], num_pixel[17:0], pixel[31:0]
END	0xFF	End of command list	

2.8.1. Instruction Set

The following sections will discuss each instruction individually.

2.8.1.1. NOP/WAIT

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
	0x00								Count																							

The NOP instruction doesn't have any side-effects but instructs the sequencer to wait a given number of module clock cycles before processing the next instruction. This allows implementing wait time requirements between the executions of other instructions.

Parameters:

- *Count*: (count + 1) number of module clock cycles to wait before loading next instruction.

The following formula can be used to calculate a required wait time:

Count = <WaitTime[s]> * <ModuleClockFreq[Hz]> – 1

For example with a given module clock frequency of 100 MHz and a required wait period of 50 ms the following NOP count should be programmed:

Count = $50\times10^{-3} * 100\times10^{6} – 1 = 5\times10^{6} – 1 = 0x4c4b3f$

2.8.1.2. IR_SET

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
0x10														Addr																	

The index register set instruction will generate an LCDBus write transfer with the D/C# signal set to command. The addr parameter will be driven on the data pins.

Note: the LCDBusIF will drive the Addr on the data pins as is, so if the LCD controller requires a special mapping for the address (e.g. {db[17:10],db[8:1]} gap assignment) the SW needs to properly convert the Addr. If the Addr to be transferred is wider than the interface width, two IR_SET commands need to be chained to transport the upper and lower part.

Parameters:

- *Addr:* address of the index register of the LCD controller. Will be mapped as command transfer.

2.8.1.3. IR_GET

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
0x11														Count																	

The status register get instruction will generate (count+1) number of LCDBus read transfers with the D/C# signal set to command.

Parameters:

- *Count*: set to number of read transfers to generate minus one (e.g. count == 0 will result in 1 read transfer, count == 3 will result in 4 read transfers).

Note: the read data will be mapped as command transfer.

2.8.1.4. DR_SET

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
0x12								Type						Data																	

The data register set instruction will generate an LCDBus write transfer with the D/C# signal set to data.

Parameters:

- *Data*: the data to be sent via the LCDBus.
- *Type*: select if data should be mapped as command (Type = 1) or data (Type = 0) transfer.

2.8.1.5. DR_GET

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
0x13								Type						Count																	

The data register get instruction will generate (count+1) number of LCDBus read transfers with the D/C# signal set to data.

Parameters:

- *Count*: set to number of read transfers to generate minus one (e.g. count == 0 will result in 1 read transfer, count == 3 will result in 4 read transfers).
- *Type*: select if read data should be mapped as command (Type = 1) or data (Type = 0) transfer.

2.8.1.6. DUMMY_GET

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
0x14																															

The dummy get instruction will generate a single read cycle on the LCDBus, but discard the read data. This allows discarding the data a lot of LCD controllers return upon the first read cycle of a read burst.

2.8.1.7. WAIT_TE

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
0x15																															

Wait for a rising edge of the LCD controllers tearing effect output.

2.8.1.8. WAIT_IF

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
0x16																															

Wait for protocol engine to finish all pending transactions.

2.8.1.9. SET_RESET

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
0x1F																															Level

Control the level of the external reset line. After reset the external reset line will have a low signal level (assuming most displays will have reset input that is active low).

Parameters:

- *Level*: 0 = drive logic 0; 1 = drive logic 1

2.8.1.10. SEND_IRQ

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
0x20																															

Send an interrupt pulse (SequencerSyncInterrupt, see Interrupt Map) to the CPU; can be used for synchronization with software.

2.8.1.11. WAIT_ACK

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
0x21																															

Wait for an acknowledge trigger from the CPU before continuing to the next instruction; can be used for synchronization with software in conjunction with the SEND_IRQ instruction.

2.8.1.12. IRIS_KICK

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
0x40																															

Send a kick pulse (frame request) on the Iris Interface.

Note: every kick sent must result in a frame accepted. So as a requirement, every IRIS_KICK must be followed by an IRIS_FRAME command eventually. Only one kick can be pending. If an IRIS_KICK opcode is read while a kick is already pending, the command will be ignored and the sequencer will go into an error state and send a SequencerErrorInterrupt (see also Error Detection).

2.8.1.13. WAIT_CMD

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
0x41																															

Wait until a command word has been received from the Iris Interface.

Note: (1) requires that an IRIS_KICK is pending otherwise the sequencer will run into timeout or wait forever. (2) the command received status will be true as soon as the command word has been received until either the WAIT_CMD or the IRIS_FRAME instruction is read. This way it is not required that this instruction is read before the Iris interface command word is actually received.

2.8.1.14. IRIS_FRAME

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
0x42																															

This command will enable the transfer of pixel information towards the protocol engine.

Note: requires that an IRIS_KICK is pending otherwise the sequencer will run into timeout or wait forever.

2.8.1.15. WR_PIXEL

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
0x80									Bpp					NumPixel																	
PixelData																															

Initiate update of display content directly via the configuration interface. After this instruction the sequencer will interpret all subsequent entries as pixel data until NumPixel have been written to the data fifo. Using the Bpp parameter each PixelData word can contain multiple pixel. Every output pixel will be written to the data fifo as if a DR_SET instruction with Type = 0 would have been initiated.

Parameters:

- *NumPixel*: number of output pixels to write to the data fifo minus one. SW is responsible to provide the required amount of input data depending of Bpp setting configured (see below).
- *Bpp*: data width of output pixels. The sequencer will extract 1..4 pixels from one PixelData word using a fixed unpacking scheme. Little endian organization is expected for pixel data. The following settings are supported:
  - 0: Extract 1 × 18-bit data from each instruction fifo entry ([17:0]). Number of required PixelData words is equal to NumPixel+1.
  - 1: Extract 2 × 16-bit data from each instruction fifo entry ([15:0], [31:16]). Number of required PixelData words is (NumPixel+1)/2.
  - 2: Extract 4 × 8-bit data from each instruction fifo entry ([7:0], [15:8], [23:16], [31:24]). Number of required PixelData words is (NumPixel+1)/4.
- *PixelData*: packed according to Bpp setting. Number of words depending on Bpp setting.

2.8.1.16. END

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
0xFF																															

End of current command list. If no more instructions are left in the instruction fifo the sequencer will go into idle state.

2.8.2. Error Detection

The command sequencer can detect three kinds of error conditions:

- Illegal opcode error: an instruction with an unsupported opcode is read
- Kick error: a frame request (IRIS_KICK) instruction is read although there is already a frame request pending
- Timeout error: a timeout mechanism can be enabled to notify the software about unexpected long execution times. With this mechanism enabled, a timeout counter will be preloaded with the configured timeout value every time a new instruction is read. With each cycle an instruction remains active in the command sequencer the timeout counter is decremented. The error materializes if the counter reaches 0 before the next instruction is read.

In case any of the above mentioned error conditions occurs the command sequencer will stop processing, go into an error state and trigger the SequencerErrorInterrupt. The error condition and the context upon the error detection can be read from the register interface for analysis.

To summarize a probably incomplete list of situations that will result in an error:

- Illegal opcode read by sequencer (will trigger an SequencerErrorInterrupt, ErrorCode = OPCODE)
- Sequencer timeout reached during execution (will trigger an SequencerErrorInterrupt, ErrorCode = TIMEOUT)
- An IRIS_KICK instruction read while a previous IRIS_KICK is still pending (will trigger an SequencerErrorInterrupt, ErrorCode = KICK)
- A WAIT_CMD or IRIS_FRAME instruction is pending without corresponding IRIS_KICK (will trigger an SequencerErrorInterrupt eventually if timeout mechanism is enabled, ErrorCode = TIMEOUT)
- A WAIT_CMD or IRIS_FRAME instruction is pending but no data is received due to misconfiguration of Iris (will trigger an SequencerErrorInterrupt eventually if timeout mechanism is enabled, ErrorCode = TIMEOUT)
- A WAIT_TE instruction is pending without an activated tearing effect signal (will trigger an SequencerErrorInterrupt eventually if timeout mechanism is enabled, ErrorCode = TIMEOUT)
- A WR_PIXEL instruction with insufficient pixel data (will trigger an SequencerErrorInterrupt eventually if timeout mechanism is enabled, ErrorCode = TIMEOUT)

3. Application

3.1. Map Tables

3.1.1. Interrupt Map

The LCDBusIF provides the interrupt signals as described in Table 37-7. Note that the LCDBusIF itself does not implement status, preset and clear registers for the interrupts, but relies on the integrating graphic subsystem for that.

Table 37-7: Interrupt Map

Name	Description
SequencerSyncInterrupt	Programmable sequencer interrupt for synchronization
SequencerErrorInterrupt	Sequencer Error Interrupt
InstrFifoInterrupt	Instruction Fifo Interrupt
RxFifoInterrupt	Reception Fifo Interrupt
TearingEffectInterrupt	Interrupt upon rising edge on tearing effect (TE) input

3.1.2. Key Map

Inside this IP all address blocks, that can be protected, use the same key values:

Table 37-8: Key Map

Name	Value	Function
Lock key	0x5651F763	Enable all access protection.
Unlock key	0x691DB936	Disable all access protection.
Privilege key	0xAEE95CDC	Enable non-privileged access protection.
Unprivileged key	0xB5E2466E	Disable non-privileged access protection.
Freeze key	0xFBE8B1E6	Freeze current protection status (cannot be changed any longer).

3.2. Getting Started

Assuming all configuration fields being in reset state, a minimal setup to initialize an exemplary LCD display and update its GRAM is:

- WriteTimingConfig:
  - WriteSetupCycles = 1
  - WriteActiveCycles = 2
  - WriteHoldCycles = 2
- ReadTimingConfig:
  - ReadSetupCycles = 1
  - ReadActiveCycles = 11 + 2
  - ReadHoldCycles = 6
- Command List (InstructionFifo), see Table 37-9
- Wait for SequencerSyncInterrupt or poll for OperationState to become IDLE

The example assumes a module clock period (tCLK) of 12.5 ns and an exemplary QVGA (320x240) LCD controller that has an 8-bit LCDBus implementing the Intel-8080 protocol (strobing via the read and write signals). Index register, configuration register and pixel width is 8-bit. The timing parameters should be assumed as in Table 37-10 and the command set as in Table 37-11.

Table 37-9: Example Command List

Instruction	Instruction (hex)	Description
SET_RESET(0) NOP(3999) SET_RESET(1) NOP(799999)	0x1f000000 0x00000f9f 0x1f000001 0x000c34ff	Toggle LCD controller reset according to reset requirements.
IR_SET(0x11)	0x10000011	Wakeup LCD controller.
IR_SET(0x2a) DR_SET(0x00) DR_SET(0x10)	0x1000002a 0x12000000 0x12000010	Set column address.
IR_SET(0x2b) DR_SET(0x00) DR_SET(0x20)	0x1000002b 0x12000000 0x12000020	Set page address.
IR_SET(0x3a) DR_SET(0x01)	0x1000003a 0x12000001	Set color mode.
IR_SET(0x2c)	0x1000002c	Initiate GRAM write procedure.
WAIT_TE() WR_PIXEL(2, 76799) *Pixel Data* ...	0x15000000 0x80212bff *0xff00ff00* ...	Synchronize to tearing effect line and write pixel data to GRAM.
IR_SET(0x29)	0x10000029	Switch display on.
SEND_IRQ()	0x21000000	Notify software about command list completion.
END()	0xff000000	End of command list.

Table 37-10: AC Characteristics of an Fictional LCD Controller

Symbol	Description	Min	Max	Unit
tCW	Chip select setup time	20	–	ns
tAH tAS	Address hold time Address setup time	20 20	–	ns
tCYCW tCCHW tCCLW	Write cycle time Write pulse H width Write pulse L width	100 35 35	–	ns
tCYCR tCCHR tCCLR	Read cycle time Read pulse H width Read pulse L width	255 90 150	–	ns
tDS tDH	Write data set time Write data hold time	20 20	–	ns
tACC tOH	Read data access time Read data disable time	– 15	145 80	ns
tRW tRT	Reset pulse width Reset clear time	50 –	– 10	us ms

Table 37-11: Register Set of a Fictional LCD Controller

Register Name	Index Address	Parameters	Description
SLPOUT	0x11	–	Wake up controller.
CASET	0x2a	CA[7:0], CA[15:8]	Column address set.
PASET	0x2b	PA[7:0], PA[15:8]	Page address set.
COLMOD	0x3a	Mode	Color Mode select.
RAMWR	0x2c	–	Initiate GRAM update.
DISPON	0x29	–	Switch display on.

3.3. Display Setup

The following sections should describe how the LCDBusIF can be configured to match the LCD controller it is attached to. A display setup involves the interface configuration and optionally the packing configuration (color format selection) in case the Iris graphics core interface should be used.

3.3.1. Interface Configuration

From the perspective of the LCDBusIF the LCDBus towards an LCD controller is defined by the interface type, the interface timing and interface mapping. Each of these properties should be discussed in the following sections.

3.3.1.1. Interface Type

To select the LCDBus protocol the LCDBusIF should use to communicate with the LCD controller, the following setup should be done:

1. Select LCDBus protocol using InterfaceType
2. Configure the LCDBus width using InterfaceWidth
3. Optional: adapt the interface signal polarities if they are in conflict with the default polarities InterfaceConfig.<signal>polarity

3.3.1.2. Interface Timing

Figure 37-11 shows typical AC characteristics diagrams for both Intel-8080 and Motorola-6800 type of LCDBus. Overlaid the Setup, Active and Hold scheme as used by the LCDBusIF is depicted. To calculate the required timing configuration, the following formulas must hold both for read and for write (time based results must be rounded up to the next integer value):

$$\text{SETUP} = \left(\frac{t_{AW} + t_{UNCERT}}{t_{CLK}}\right)$$

$$\text{(read) ACTIVE} = \max\left(\frac{t_{ACC} + t_{OUT2IN}}{t_{CLK}}, \frac{t_{CCLR}}{t_{CLK}}\right)$$

$$\text{(write) ACTIVE} = \max\left(\frac{t_{DS} + t_{UNCERT}}{t_{CLK}}, \frac{t_{CCLW}}{t_{CLK}}\right)$$

$$\text{(write) HOLD} = \max\left(\frac{t_{AH}}{t_{CLK}}, \frac{t_{CCHW}}{t_{CLK}} - \text{SETUP}, \frac{t_{DH}}{t_{CLK}} - \text{SETUP}\right)$$

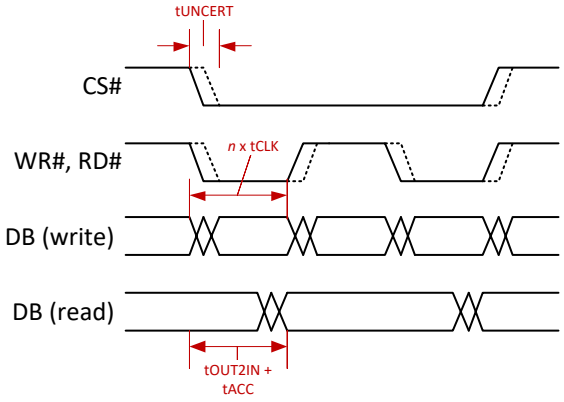
$$\text{(read) HOLD} = \max\left(\frac{t_{AH}}{t_{CLK}}, \frac{t_{CCHR}}{t_{CLK}} - \text{SETUP}\right)$$

$$\text{SETUP} + \text{ACTIVE (read)} + \text{HOLD (read)} \geq t_{CYCR}$$

$$\text{SETUP} + \text{ACTIVE (write)} + \text{HOLD (write)} \geq t_{CYCW}$$

where tCLK is the period of one module clock cycle, tUNCERT describes the error which results due to different routing delays of the depending signals and tOUT2IN describes the sum of maximum output and input delay. Both tUNCERT and tOUT2IN are properties of the device embedding the LCDBusIF and need to be looked up in the AC characteristics of the device specification (see Figure 37-10 for an illustration of these error terms).

Figure 37-10: AC Characteristics of the LCDBusIF from Device Point of View



Note how in Figure 37-11 the SETUP configuration contributes to the HOLD timing window for the data signals. Therefore the HOLD configuration can be reduced by the SETUP time (but must be at least 1). Table 37-12 summarizes the LCDBus timing symbols as used in Figure 37-11.

The calculated SETUP, ACTIVE and HOLD values must be programmed to the WriteTimingConfig and ReadTimingConfig registers. Note that the intended value must be programmed with minus one.

Figure 37-11: Timing specification for Intel-8080 and Motorola-6800 Interface protocols

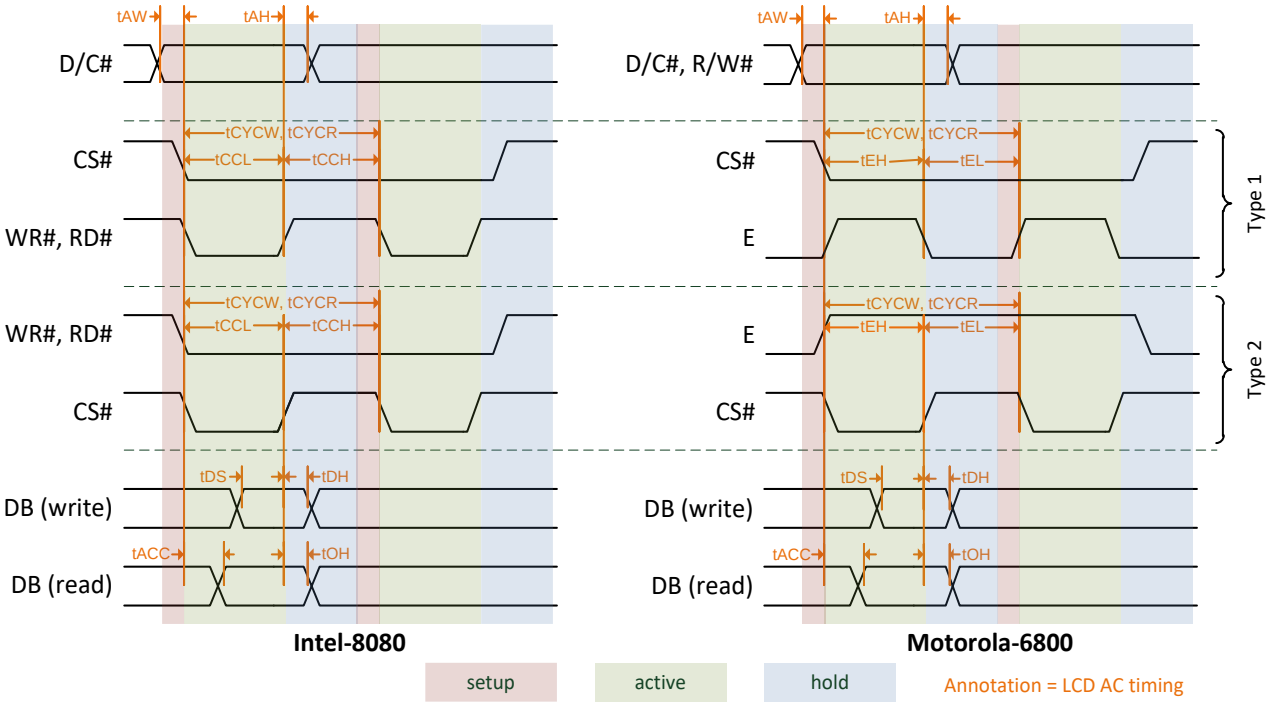


Table 37-12: Timing Symbol Descriptions

Symbol	Description
tAH tAW	Address hold time Address setup time
tCYCR/W tCCHR/W, tEH tCCLR/W, tEL	Read/Write cycle time Read/Write, Enable pulse H width Read/Write, Enable pulse L width
tDS tDH	Write data set time Write data hold time
tACC tOH	Read data access time Read data disable time
tRW tRT	Reset pulse width Reset clear time

3.3.1.3. Interface Mapping

Depending on the InterfaceWidth set above and the native LCDBus register and pixel data width a transfer mapping should be configured for both command and data accesses as described in Data Mapping using the DataTransferMapping0..2 and CommandTransferMapping0..2 register respectively.

3.3.2. Packing Configuration

To program the color format conversion from Iris 24-bit to the color format required by the LCD controller the following configuration should be done:

1. Program BitsPerPixel as defined by the color format.
2. In most cases BitsPerTransfer will be equal to BitsPerPixel. Only if the InterfaceWidth is greater than BitsPerPixel and the LCD controller can handle multiple pixel per LCDBus transfer, BitsPerTransfer should be set to the InterfaceWidth to enable the multiple pixel per transfer feature of the LCDBusIF.

3. Program the RGB color bits selection using the ColorComponentBits and ColorComponentShift registers.

Table 37-13 lists a selection of packing configurations for color formats often used by LCD controllers.

Table 37-13: Selection of standard LCD controller color formats

Color Format	ColorComponentBits			ColorComponentShift			DestinationAttributes	
	Red	Green	Blue	Red	Green	Blue	BPT	BPP
RGB666	6	6	6	12	6	0	18	18
RGB565	5	6	5	11	5	0	16	16
RGB444	4	4	4	8	4	0	12	12
GREY8	0	8	0	0	0	0	8	8
GREY4	0	4	0	0	0	0	8	4
B/W	0	1	0	0	0	0	8	1

3.4. Control Flow

In order to execute a command list the CPU writes instructions to the InstructionFifo. Using the InstrFifoStatus register the CPU should check how much space is left to avoid that the fifo runs full. The sequencer will start execution as soon as instructions are seen in the InstructionFifo. Each command list should be finished with an END instruction, when reaching it the sequencer will return into idle state. The instruction counter will reset with the next instruction read. If more instructions are remaining in the InstructionFifo the sequencer will restart execution.

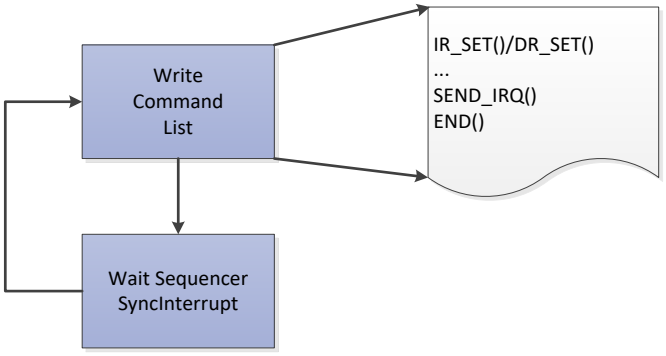
3.4.1. Generic Sequencer Control Flow

For command lists which don't depend on pixel content a simple control flow can be implemented.

1. The CPU writes the commands to be sent to the display to the InstructionFifo. In case it should be notified about the completion a SEND_IRQ command must be embedded in the command list.
2. In case an interrupt was setup, the CPU can wait for it to finish preparing the next command list. Alternatively the CPU can poll the OperationState field to become IDLE.

This flow is recommended for one shot operations like initialization, power-saving modes or shutdown.

Figure 37-12: Generic sequencer control flow



3.4.2. Iris Content Control Flow

In case the display content should be updated by the Iris a slightly modified control flow needs to be implemented.

1. The CPU configures the Iris graphics core to compose the desired scene. The LCD Bus Interface bypass needs to be active.
2. A command list is written which consists of the following sections:
   a. IRIS_KICK to trigger frame creation in the Iris
   b. WAIT_TE to synchronize to tearing effect signal of the timing controller inside the display

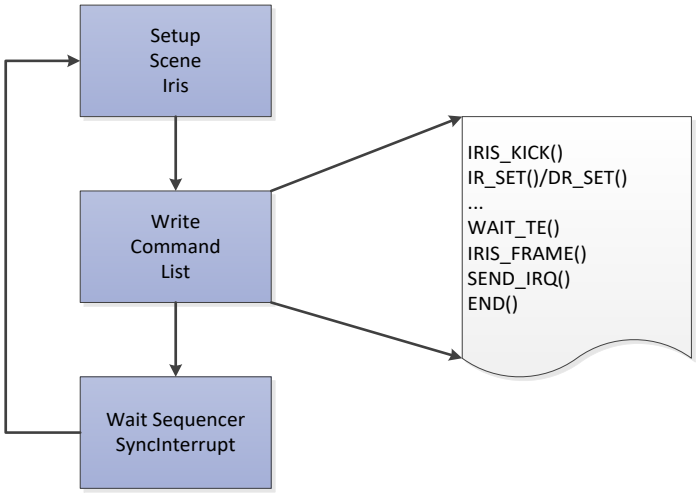
c. Optional: a sequence of IR_SET/DR_SET commands to prepare the display GRAM access

d. IRIS_FRAME to start sending pixels from the Iris to the display

e. SEND_IRQ to inform the CPU about command list completion (sequence complete interrupt)

3. Wait for SequencerSyncInterrupt

Figure 37-13: Control Flow for Updating Display Content via Iris Interface

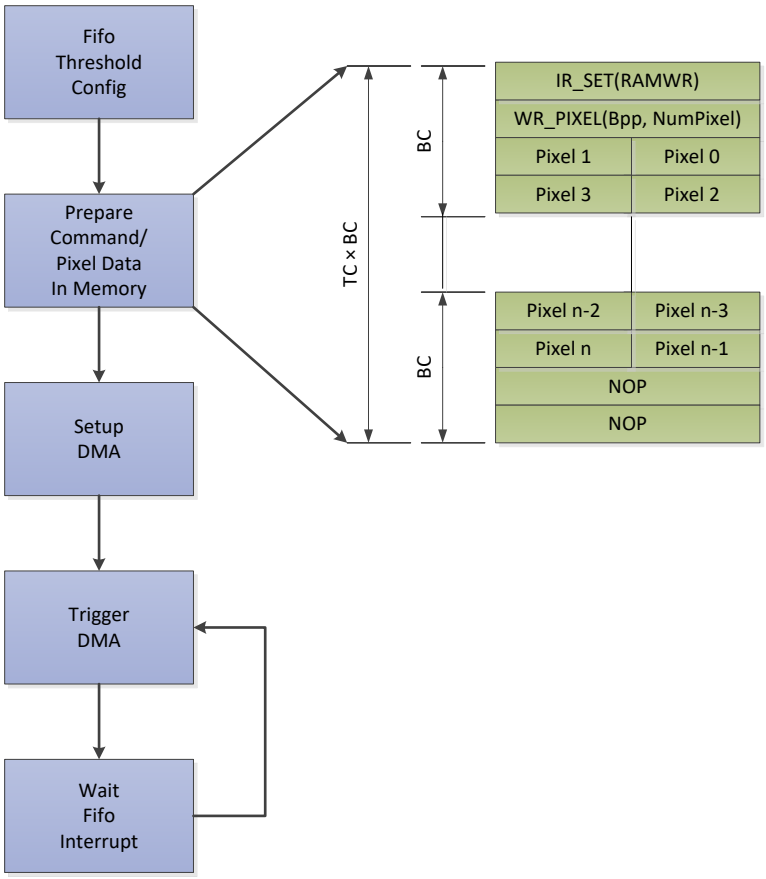


3.4.3. Software DMA Control Flow

For devices without an Iris graphic core a software controlled DMA flow can be implemented to efficiently update the display content without too much involvement from the CPU.

1. Setup InstructionFifo threshold configuration depending of desired burst length that should be configured in the DMA controller (see Fifo Operation).
2. Prepare pixel and command data in memory. Any prerequisite sequencer commands can either be embedded in the memory image or be written to the InstructionFifo beforehand. Either way the last instruction to the sequencer should be WR_PIXEL with the desired bits per pixel configuration (16-bit in this example) and the number of output pixel that should be written to the data fifo (n in this example). If the resulting number of transfers is not a multiple of the block count (BC) configured in the DMA controller the memory image should be padded with NOP instructions. The data must be formatted in the color format expected by the LCD controller. In case of an 8- or 16-bpp format multiple pixels can be packed into a 32-bit word.
3. Setup the DMA controller to transfer the memory image to the InstructionFifo. The number of transfers is usually configured by a transfer count (TC) and a block count (BC) parameter.
4. Trigger a DMA block transfer using the software trigger functionality of the DMA controller.
5. Wait for the threshold fifo interrupt and initiate another DMA block transfer until the DMA signals the end of transfer (Note: this scenario assumes that the DMA can fill the InstructionFifo faster than the LCDBusIF can read it).

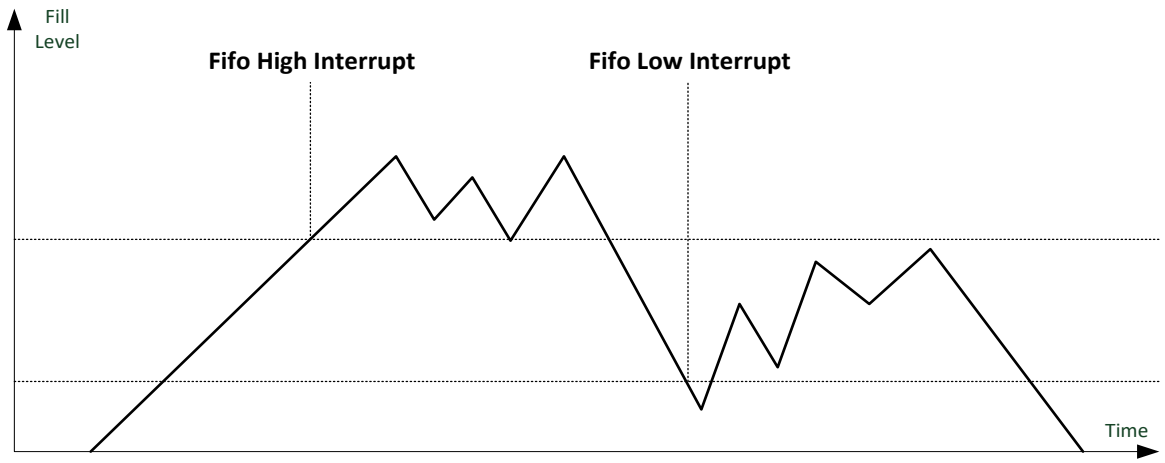
Figure 37-14: Control Flow for Content Update from Memory Using Software DMA



3.5. Fifo Operation

The LCDBusIF has two fifos which are accessible via the configuration interface. Both fifos implement a similar status and control interface for SW interaction. To trigger reception fifo read or instruction fifo write requests a programmable hysteresis on the fifo fill level is implemented. The software can configure to receive an interrupt either upon crossing a high or low level threshold. A high threshold interrupt will only be given if the low threshold boundary has been crossed before, similarly a low threshold interrupt will only be given if the high threshold boundary has been crossed before. This behavior is depicted in Figure 37-15. The following sections describe the behavior of the instruction and the reception fifo.

Figure 37-15: Fifo Hysteresis Implementation with Low and High Threshold Interrupt



3.5.1. Instruction Fifo

Data can be sent to instruction fifo using either a fixed or incremented address within its address space (InstructionFifo). The number of available entries left can be read from the fifo status register (FifoStatus). Writing to a full InstructionFifo is considered an error condition and should be avoided from SW. If the software does write to a full InstructionFifo the bus will be stalled for some time to give the InstructionFifo some time to empty. If the fifo remains full an error response will be given on the configuration bus interface and the data will be discarded.

To clear instruction fifo the error recovery procedure should be executed, see Error Recovery.

3.5.2. Reception Fifo

Data can be read from reception fifo using either a fixed or incremented addresses from its address space (RxFifo). The number of available entries can be read from the fifos status registers (FifoStatus). Reading from an empty RxFifo is considered an error condition and should be avoided from SW. If the software does read from an empty RxFifo the LCDBusIF will respond with unknown data.

The reception fifo can be cleared individually (RxClear) if data in the fifo is obsolete. Otherwise the reception fifo will also be cleared with the error recovery procedure, see Error Recovery.

3.6. Error Recovery

In case the LCDBusIF is entering an erroneous state the following procedure should be applied to recover to normal operation:

1. Optional: read the SequencerStatus register to analyze the error condition.
2. Optional: read the RxFifo if it has any contents of value.
3. Trigger a SoftwareReset. This will clear both the InstructionFifo and the RxFifo and the sequencer will return into IDLE state.
4. Optional: wait until the Iris is idle if a frame transmission is still pending.

Afterwards the sequencer is ready to process a new command list.

3.7. SW Interface

3.7.1. General

3.7.1.1. Register Addresses

The address of a register in a system environment is the sum of

- Register offset as specified in the following register tables.
- Address offset of the Iris core configuration in the address space of the embodying system.

3.7.1.2. Field Types

The field tables specify a *Type* for each field, which means the following:

Table 37-14: Field Types

R	Read only. Writing has no effect.
W	Write only. Reading always returns ‘0‘.
RW	Read and write to active configuration.
RWS	Read and write to shadowed configuration (must be activated by shadow load).
RW1C	Status bit. Set by HW. Writing ‘0‘ has no effect, writing ‘1‘ clears status. Reading returns current status.
W1P	Trigger bit. Writing ‘0‘ has no effect, writing ‘1‘ triggers some HW activity. Reading returns ‘0‘.

3.7.1.3. Field Formats

The field tables specify a *Format* for each field, which means the following:

Table 37-15: Field Formats

bit	Single bit.
Uint<*w*>	Unsigned integer with *w* bits.
Int<*w*>	Signed integer in two’s complement notation with *w* bits in total (including sign bit).
Ufix<*i*>.<*f*>	Unsigned fix-point with *I* integer and *f* fraction bits.
Fix<*i*>.<*f*>	Signed fix-point in two’s complement notation with *I* integer bits (including sign bit) and *f* fraction bits.
Float1e<*e*>m<*m*>	Floating point value with 1 sign bit, *e* exponent bits and *m* mantissa bits. Notation is according to IEEE 754.
Enum	Values have symbolic names (enumeration type).

3.7.1.4. Register Protection

If registers can be access protected can be read from *Prot* column in register tables:

Table 37-16: Register Lock Properties

key	Special register to change protection status of all registers in the same address block (= register table) by writing certain key values to it (see Key Map).
status	Special register to read current protection status.
no	Register that cannot be protected.
yes	Register that can be protected against non-privileged access (privilege key) and/or against any kind of write access (lock key).

3.7.2. Error Responses

The following conditions result in an error response on the AHB bus:

- Read or write access to an address with no register / no fields.
- Read or write access with transfer size other than 32-bit.
- Write access to a register that has read-only fields only.
- Read access to a register that has write-only fields only.

- Write access to a register with lock property ‘yes’ when lock status is active.
- Non-privileged read or write access to a register with lock property ‘yes’ or the lock/unlock register when privilege status is active (except read access to the lock status register, which is always allowed).
- Read access to lock/unlock register.
- Write access to the lock/unlock register with...
  - ...an invalid key value.
  - ...a valid key value when the freeze status is active.
  - ...the lock key value when internal unlock counter is 0.
  - ...the unlock key value when internal unlock counter is 15.
  - ...the privilege key value when the privilege status is active.
  - ...the un-privilege key value when the privilege status is not active.

3.7.3. Register Overview

Addr. Offset	Register Name	Prot.	Register Description (Address Block 0)
0x0000	LockUnlockLcd	key	Register to change the protection status of this address block.
0x0004	LockStatusLcd	status	Protection status of this address block.
0x0008	InterfaceConfig	yes	LCDBus interface configuration register.
0x000C	WriteTimingConfig	yes	Write timing configuration register. Sum of all fields is write cycle time in module clock cycles.
0x0010	ReadTimingConfig	yes	Read timing configuration register. Sum of all fields is read cycle time in module clock cycles.
0x0014	CommandTransferMapping0	yes	Transfer mapping for command data on LCDBus.
0x0018	CommandTransferMapping1	yes	Transfer mapping for command data on LCDBus.
0x001C	CommandTransferMapping2	yes	Transfer mapping for command data on LCDBus.
0x0020	DataTransferMapping0	yes	Transfer mapping for data/parameter data on LCDBus.
0x0024	DataTransferMapping1	yes	Transfer mapping for data/parameter data on LCDBus.
0x0028	DataTransferMapping2	yes	Transfer mapping for data/parameter data on LCDBus.
0x002C	ColorComponentBits	yes	Color component size for color format of LCD controller.
0x0030	ColorComponentShift	yes	Color component offset for color format of LCD controller.
0x0034	DestinationAttributes	yes	Configuration for packing unit.
Addr. Offset	**Register Name**	**Prot.**	**Register Description (Address Block 1)**
0x0040	LockUnlockControl	key	Register to change the protection status of this address block.
0x0044	LockStatusControl	status	Protection status of this address block.
0x0048	SequencerConfig	yes	Sequencer configuration register.
0x004C	InstructionFifoConfig	yes	Instruction fifo configuration register.
0x0050	ReceptionFifoConfig	yes	Reception fifo configuration register.
0x0054	SoftwareReset	yes	Software Reset Register.
0x0058	RxFifoControl	yes	Reception fifo control register.
0x005C	SequencerSync	yes	Sequencer synchronization register.
0x0060	SequencerStatus	yes	Status of the internal sequencer.
0x0064	FifoStatus	yes	Fifo status register.

Addr. Offset	Register Name	Prot.	Register Description (Address Block 2)
0x0080	LockUnlockFifo	key	Register to change the protection status of the fifo address blocks.
0x0084	LockStatusFifo	status	Protection status of this address block.
Addr. Offset	**Register Name**	**Prot.**	**Register Description (Address Block 3)**
		key	Same as for address block 2 of this unit.
		status	Same as for address block 2 of this unit.
0x0100	InstructionFifo[0..31]	yes	Instruction Fifo.
Addr. Offset	**Register Name**	**Prot.**	**Register Description (Address Block 4)**
		key	Same as for address block 2 of this unit.
		status	Same as for address block 2 of this unit.
0x0200	RxFifo[0..15]	yes	Reception Fifo.

3.7.4. LockUnlockLcd

Register to change the protection status of this address block.

Bits	Type	Reset	Format	Field Name	Description
[31,0]	W	-	enum	LockUnlockLcd	The protection status is changed by writing one of the following key values to this field: • 0x5651F763 = lock_key: Decrements the unlock counter. When the counter value is null, lock protection is active. Reset counter value is 1. • 0x691DB936 = unlock_key: Increments the unlock counter. Max allowed value is 15. • 0xAEE95CDC = privilege_key: Enables privilege protection. Disabled after reset. • 0xB5E2466E = unprivilege_key: Disables privilege protection. • 0xFBE8B1E6 = freeze_key: Freezes current protection status. Writing keys to this register has no more effect until reset.
When lock protection is active, no write but only read access is possible to all registers of this address block. When privilege protection is active, only privileged read and write access is possible. Both protections can be active at the same time. Reading this register returns the current unlock counter value, but results in an error response (for HW test purposes only).					

3.7.5. LockStatusLcd

Protection status of this address block.

Bits	Type	Reset	Format	Field Name	Description
[0]	R	0x0	bit	LockStatusLcd	Current status of lock protection: 0 = inactive (unlock counter > 0), 1 = active (unlock counter == 0).
[4]	R	0x0	bit	PrivilegeStatusLcd	Current status of privilege protection: 0 = inactive , 1 = active.
[8]	R	0x0	bit	FreezeStatusLcd	Current freeze status: 0 = protection status can be changed, 1 = cannot be changed.

3.7.6. InterfaceConfig

LCDBus interface configuration register.

Bits	Type	Reset	Format	Field Name	Description
[1,0]	RW	0x0	enum	InterfaceType	Describes the type of the external interface. • 0x0 = INTEL80_TYPE1: Intel 8080 compatible interface (strobe with RD#/WR#). • 0x1 = INTEL80_TYPE2: Intel 8080 compatible interface (strobe with CS#). • 0x2 = MOTOROLA68_TYPE1: Motorola 6800 compatible interface (strobe with E). • 0x3 = MOTOROLA68_TYPE2: Motorola 6800 compatible interface (strobe with CS#).
[9,8]	RW	0x0	enum	InterfaceWidth	Describes the data width of the external interface. • 0x0 = LCD8: 8-bit LCDBus • 0x1 = LCD9: 9-bit LCDBus • 0x2 = LCD16: 16-bit LCDBus • 0x3 = LCD18: 18-bit LCDBus
[16]	RW	0x0	enum	CsPolarity	Polarity for chip select output. • 0x0 = ACTIVE_LOW: CS is a low active signal. • 0x1 = ACTIVE_HIGH: CS is a high active signal.
[17]	RW	0x0	enum	WrPolarity	Polarity for WR# (8080) or R/W# (6800) otuput. • 0x0 = DEFAULT: WR# (R/W#) has default protocol polarity. • 0x1 = INVERTED: WR# (R/W#) has inverted protocol polarity.
[18]	RW	0x0	enum	RdPolarity	Polarity for RD# (8080) or E (6800) output. • 0x0 = DEFAULT: RD# (E) has default protocol polarity. • 0x1 = INVERTED: RD# (E) has inverted protocol polarity.
[19]	RW	0x0	enum	RsPolarity	Polarity of register select output. • 0x0 = COMMAND_LOW: RS is command with low signal level. • 0x1 = COMMAND_HIGH: RS is command with high signal level.
[20]	RW	0x0	enum	DataPolarity	Inversion for data signals upon write. • 0x0 = DEFAULT: Data bus is used as is. • 0x1 = INVERTED: Data bus is inverted.
[21]	RW	0x0	enum	TePolarity	Polarity for tearing effect input. • 0x0 = ACTIVE_HIGH: TE is a high active signal. • 0x1 = ACTIVE_LOW: TE is a low active signal.

3.7.7. WriteTimingConfig

Write timing configuration register. Sum of all fields is write cycle time in module clock cycles.

Bits	Type	Reset	Format	Field Name	Description
[7,0]	RW	0x4	uint8	WriteSetupCycles	Number of module clock cycles minus one to wait before wr#/cs#/en terminal can become active after address set (tAW).
[15,8]	RW	0x8	uint8	WriteActiveCycles	Number of module clock cycles minus one wr#/cs#/en terminal should remain active (tCCLW).
[23,16]	RW	0x4	uint8	WriteHoldCycles	Number of module clock cycles minus one to wait after wr#/cs#/en terminal became inactive before address can change (tAH).

3.7.8. ReadTimingConfig

Read timing configuration register. Sum of all fields is read cycle time in module clock cycles.

Bits	Type	Reset	Format	Field Name	Description
[7,0]	RW	0x4	uint8	ReadSetupCycles	Number of module clock cycles minus one to wait before rd#/cs#/en terminal can become active after address set (tAW).
[15,8]	RW	0x8	uint8	ReadActiveCycles	Number of module clock cycles minus one rd#/cs#/en terminal should remain active (tCCLR).
[23,16]	RW	0x4	uint8	ReadHoldCycles	Number of module clock cycles minus one to wait after rd#/cs#/en terminal became inactive before address can change (tAH).

3.7.9. CommandTransferMapping0

Transfer mapping for command data on LCDBus.

Bits	Type	Reset	Format	Field Name	Description
[4,0]	RW	0x8	uint5	CommandTransfer0Bits	Number of bits from InstructionFifo to be transfered with first transfer. Must be greater than 0.
[9,5]	RW	0x0	uint5	CommandTransfer0SrcLsb	Least significant bit of vector to extract from source (Wr=InstructionFifo, Rd=Interface data) (first transfer).
[14,10]	RW	0x0	uint5	CommandTransfer0DstLsb	Least significant bit in destination word (Wr=Interface data, Rd=RxFifo) to insert vector into (first transfer).
[15]	RW	0x0	bit	CommandTransfer0Next	If set, a second transfer will be required to transmit the command data, see CommandTransferMapping1.
Command mapping is applicable for IR_SET, IR_GET, DR_SET(type==1), DR_GET(type==1) instructions.					

3.7.10. CommandTransferMapping1

Transfer mapping for command data on LCDBus.

Bits	Type	Reset	Format	Field Name	Description
[4,0]	RW	0x0	uint5	CommandTransfer1Bits	Number of bits from InstructionFifo to be transfered with second transfer. Must be greater than 0 if CommandTransfer0Next is set.
[9,5]	RW	0x0	uint5	CommandTransfer1SrcLsb	Least significant bit of vector to extract from source (Wr=InstructionFifo, Rd=Interface data) (second transfer).
[14,10]	RW	0x0	uint5	CommandTransfer1DstLsb	Least significant bit in destination word (Wr=Interface data, Rd=RxFifo) to insert vector into (second transfer).
[15]	RW	0x0	bit	CommandTransfer1Next	If set, a third transfer will be required to transmit the command data, see CommandTransferMapping2.
Command mapping is applicable for IR_SET, IR_GET, DR_SET(type==1), DR_GET(type==1) instructions.					

3.7.11. CommandTransferMapping2

Transfer mapping for command data on LCDBus.

Bits	Type	Reset	Format	Field Name	Description
[4,0]	RW	0x0	uint5	CommandTransfer2Bits	Number of bits from InstructionFifo to be transfered with third transfer.
[9,5]	RW	0x0	uint5	CommandTransfer2SrcLsb	Least significant bit of vector to extract from source (Wr=InstructionFifo, Rd=Interface data) (third transfer).
[14,10]	RW	0x0	uint5	CommandTransfer2DstLsb	Least significant bit in destination word (Wr=Interface data, Rd=RxFifo) to insert vector into (third transfer).
Command mapping is applicable for IR_SET, IR_GET, DR_SET(type==1), DR_GET(type==1) instructions.					

3.7.12. DataTransferMapping0

Transfer mapping for data/parameter data on LCDBus.

Bits	Type	Reset	Format	Field Name	Description
[4,0]	RW	0x8	uint5	DataTransfer0Bits	Number of bits from DataFifo to be transfered with first transfer.
[9,5]	RW	0x0	uint5	DataTransfer0SrcLsb	Least significant bit of vector to extract from source (Wr=InstructionFifo, Rd=Interface data) (first transfer).
[14,10]	RW	0x0	uint5	DataTransfer0DstLsb	Least significant bit in destination word (Wr=Interface data, Rd=RxFifo) to insert vector into (first transfer).
[15]	RW	0x0	bit	DataTransfer0Next	If set, second transfer will be required to transmit a data word.
Data mapping is applicable for DR_SET(type==0), DR_GET(type==0), IRIS_FRAME and WR_PIXEL instructions.					

3.7.13. DataTransferMapping1

Transfer mapping for data/parameter data on LCDBus.

Bits	Type	Reset	Format	Field Name	Description
[4,0]	RW	0x0	uint5	DataTransfer1Bits	Number of bits from DataFifo to be transfered with second transfer.
[9,5]	RW	0x0	uint5	DataTransfer1SrcLsb	Least significant bit of vector to extract from source (Wr=InstructionFifo, Rd=Interface data) (second transfer).
[14,10]	RW	0x0	uint5	DataTransfer1DstLsb	Least significant bit in destination word (Wr=Interface data, Rd=RxFifo) to insert vector into (second transfer).
[15]	RW	0x0	bit	DataTransfer1Next	If set, third transfer will be required to transmit a data word.
Data mapping is applicable for DR_SET(type==0), DR_GET(type==0), IRIS_FRAME and WR_PIXEL instructions.					

3.7.14. DataTransferMapping2

Transfer mapping for data/parameter data on LCDBus.

Bits	Type	Reset	Format	Field Name	Description
[4,0]	RW	0x0	uint5	DataTransfer2Bits	Number of bits from DataFifo to be transfered with third transfer.
[9,5]	RW	0x0	uint5	DataTransfer2SrcLsb	Least significant bit of vector to extract from source (Wr=InstructionFifo, Rd=Interface data) (third transfer).
[14,10]	RW	0x0	uint5	DataTransfer2DstLsb	Least significant bit in destination word (Wr=Interface data, Rd=RxFifo) to insert vector into (third transfer).
Data mapping is applicable for DR_SET(type==0), DR_GET(type==0), IRIS_FRAME and WR_PIXEL instructions.					

3.7.15. ColorComponentBits

Color component size for color format of LCD controller.

Bits	Type	Reset	Format	Field Name	Description
[11,8]	RW	0x5	uint4	ComponentBitsBlue	Blue component bits.
[19,16]	RW	0x6	uint4	ComponentBitsGreen	Green component bits.
[27,24]	RW	0x5	uint4	ComponentBitsRed	Red component bits.
Size of RGB is from 0 to 8 bits. The sum of RGB bits must be greater than 0.					

3.7.16. ColorComponentShift

Color component offset for color format of LCD controller.

Bits	Type	Reset	Format	Field Name	Description
[12,8]	RW	0x0	uint5	ComponentShiftBlue	Blue component shift.
[20,16]	RW	0x5	uint5	ComponentShiftGreen	Green component shift.
[28,24]	RW	0xB	uint5	ComponentShiftRed	Red component shift.

3.7.17. DestinationAttributes

Configuration for packing unit.

Bits	Type	Reset	Format	Field Name	Description
[4,0]	RW	0x10	uint5	BitsPerTransfer	How many bits can be packed into one transfer (allows packing multiple bits into one LCDBus transfer).
[12,8]	RW	0x10	uint5	BitsPerPixel	How many bits are required for one pixel.
The packing factor per transfer is the integer division of BitsPerTransfer/BitsPerPixel. BitsPerTransfer must be greater than or equal to BitsPerPixel.					

3.7.18. LockUnlockControl

Register to change the protection status of this address block.

Bits	Type	Reset	Format	Field Name	Description
[31,0]	W	-	enum	LockUnlockControl	The protection status is changed by writing one of the following key values to this field: 0x5651F763 = lock_key: Decrements the unlock counter. When the counter value is null, lock protection is active. Reset counter value is 1. 0x691DB936 = unlock_key: Increments the unlock counter. Max allowed value is 15. 0xAEE95CDC = privilege_key: Enables privilege protection. Disabled after reset. 0xB5E2466E = unprivilege_key: Disables privilege protection. 0xFBE8B1E6 = freeze_key: Freezes current protection status. Writing keys to this register has no more effect until reset.
When lock protection is active, no write but only read access is possible to all registers of this address block. When privilege protection is active, only privileged read and write access is possible. Both protections can be active at the same time. Reading this register returns the current unlock counter value, but results in an error response (for HW test purposes only).					

3.7.19. LockStatusControl

Protection status of this address block.

Bits	Type	Reset	Format	Field Name	Description
[0]	R	0x0	bit	LockStatusControl	Current status of lock protection: 0 = inactive (unlock counter > 0), 1 = active (unlock counter == 0).
[4]	R	0x0	bit	PrivilegeStatusControl	Current status of privilege protection: 0 = inactive , 1 = active.
[8]	R	0x0	bit	FreezeStatusControl	Current freeze status: 0 = protection status can be changed, 1 = cannot be changed.

3.7.20. SequencerConfig

Sequencer configuration register.

Bits	Type	Reset	Format	Field Name	Description
[30,0]	RW	0x7FFFFFFF	uint31	InstructionTimeout	Value to preload the timeout counter with each instruction read.
[31]	RW	0x1	bit	InstructionTimeoutEnable	Enable or disable instruction timeout counter.
If the instruction timeout mechanism is enabled, a timeout counter will be preloaded with the InstructionTimeout value every time a new instruction is read. With each cycle an instruction remains active the timeout counter is decremented. In case the counter reaches 0 the sequencer will enter an error state and generate a SequencerErrorInterrupt.					

3.7.21. InstructionFifoConfig

Instruction fifo configuration register.

Bits	Type	Reset	Format	Field Name	Description
[5,0]	RW	0x0	uint6	InstrThresholdLow	The threshold below which the instruction fifo should generate a low interrupt.
[13,8]	RW	0x20	uint6	InstrThresholdHigh	The threshold above which the instruction fifo should generate a high interrupt.
[17,16]	RW	0x1	enum	InstrThresholdTrigger	Select trigger condition for InstrFifoInterrupt. • 0 = LOW: InstrFifoInterrupt triggers if level falls below low threshold after being above high level. • 1 = HIGH: InstrFifoInterrupt triggers if level raises above high threshold after being below low level. • 2 = BOTH: InstrFifoInterrupt triggers for both low and high threshold conditions.
Note: InstrThresholdHigh must be greater than InstrThresholdLow.					

3.7.22. ReceptionFifoConfig

Reception fifo configuration register.

Bits	Type	Reset	Format	Field Name	Description
[4,0]	RW	0x0	uint5	RxThresholdLow	The threshold below which the reception fifo should generate a low interrupt.
[12,8]	RW	0x10	uint5	RxThresholdHigh	The threshold above which the reception fifo should generate a high interrupt.
[17,16]	RW	0x1	enum	RxThresholdTrigger	Select trigger condition for RxFifoInterrupt. • 0 = LOW: RxFifoInterrupt triggers if level falls below low threshold after being above high level. • 1 = HIGH: RxFifoInterrupt triggers if level raises above high threshold after being below low level. • 2 = BOTH: RxFifoInterrupt triggers for both low and high threshold conditions.
Note: RxThresholdHigh must be greater than RxThresholdLow.					

3.7.23. SoftwareReset

Software Reset Register.

Bits	Type	Reset	Format	Field Name	Description
[0]	W1P	-	bit	SoftwareReset	Writting a 1 to this field will reset all fifos and control logic of LCDBusIF.

3.7.24. RxFifoControl

Reception fifo control register.

Bits	Type	Reset	Format	Field Name	Description
[0]	W1P	-	bit	RxClear	Writting a 1 to this field will clear the contents of the RxFifo only.

3.7.25. SequencerSync

Sequencer synchronization register.

Bits	Type	Reset	Format	Field Name	Description
[0]	W1P	-	bit	SequencerAck	A write to this bit will allow the sequencer to continue from the WAIT_ACK instruction.

3.7.26. SequencerStatus

Status of the internal sequencer.

Bits	Type	Reset	Format	Field Name	Description
[1,0]	R	0x0	enum	OperationState	Indicate operation status. • 0x0 = IDLE: Processing finished, nothing to do. • 0x1 = BUSY: Currently processing instruction list. • 0x3 = ERROR: An error has occured
[2]	R	0x0	enum	WaitAck	Indicate if a WAIT_ACK instruction is currently pending. • 0x0 = DONE: No waiting for acknowledge. • 0x1 = PENDING: Acknowledge from CPU is pending.
[7,4]	R	0x0	enum	ErrorCode	Type of error that has occured • 0x0 = NONE: No error. • 0x1 = OPCODE: Illegal opcode error. • 0x2 = KICK: Too many outstanding kicks error. • 0x3 = TIMEOUT: Sequencer timout reached.
[13,8]	R	-	uint6	InstructionCounter	Number of instructions processed in current command list.
[23,16]	R	0x0	enum	InstructionRegister	Current instruction executed by sequencer. • 0x00 = NOP: Currently executing NOP instruction. • 0x10 = IR_SET: Currently executing IR_SET instruction. • 0x11 = IR_GET: Currently executing IR_GET instruction. • 0x12 = DR_SET: Currently executing DR_SET instruction. • 0x13 = DR_GET: Currently executing DR_GET instruction. • 0x14 = DUMMY_GET: Currently executing DUMMY_GET instruction. • 0x15 = WAIT_TE: Currently executing WAIT_TE instruction. • 0x16 = WAIT_IF: Currently executing WAIT_IF instruction. • 0x1F = SET_RESET: Currently executing SET_RESET instruction. • 0x20 = SEND_IRQ: Currently executing SEND_IRQ instruction. • 0x21 = WAIT_ACK: Currently executing WAIT_ACK instruction. • 0x40 = IRIS_KICK: Currently executing IRIS_KICK instruction. • 0x41 = WAIT_CMD: Currently executing WAIT_CMD instruction. • 0x42 = IRIS_FRAME: Currently executing IRIS_FRAME instruction. • 0x80 = WR_PIXEL: Currently executing WR_PIXEL instruction. • 0xFF = END: Currently executing END instruction.
[25,24]	R	0x0	enum	IrisInterfaceStatus	Status of Iris interface. • 0x0 = IDLE: Nothing pending. • 0x1 = KICK: Kick sent, frame pending. • 0x2 = CMD: Command received, pixel transmission pending. • 0x3 = FRAME: Frame transmission ongoing.
[28]	R	0x0	enum	LcdInterfaceState	Indicate LCDBus status. 0x0 = IDLE: No transactions pending. 0x1 = BUSY: Currently sending transactions.

3.7.27. FifoStatus

Fifo status register.

Bits	Type	Reset	Format	Field Name	Description
[5,0]	R	0x20	uint6	InstrSpace	Available space in InstructionregistFifo in number of entries.
[8]	R	0x1	bit	InstrEmpty	Indicates if InstructionFifo is empty.
[12]	R	0x0	bit	InstrFull	Indicates if InstructionFifo is full.
[20,16]	R	0x0	uint5	RxLevel	Number of entries in RxFifo.
[24]	R	0x1	bit	RxEmpty	Indicates if RxFifo is empty.
[28]	R	0x0	bit	RxFull	Indicates if RxFifo is full.

3.7.28. LockUnlockFifo

Register to change the protection status of the fifo address blocks.

Bits	Type	Reset	Format	Field Name	Description
[31,0]	W	-	enum	LockUnlockFifo	The protection status is changed by writing one of the following key values to this field: • 0x5651F763 = lock_key: Decrements the unlock counter. When the counter value is null, lock protection is active. Reset counter value is 1. • 0x691DB936 = unlock_key: Increments the unlock counter. Max allowed value is 15. • 0xAEE95CDC = privilege_key: Enables privilege protection. Disabled after reset. • 0xB5E2466E = unprivilege_key: Disables privilege protection. • 0xFBE8B1E6 = freeze_key: Freezes current protection status. Writing keys to this register has no more effect until reset.
When lock protection is active, no write but only read access is possible to all registers of this address block. When privilege protection is active, only privileged read and write access is possible. Both protections can be active at the same time. Reading this register returns the current unlock counter value, but results in an error response (for HW test purposes only).					

3.7.29. LockStatusFifo

Protection status of this address block.

Bits	Type	Reset	Format	Field Name	Description
[0]	R	0x0	bit	LockStatusFifo	Current status of lock protection: 0 = inactive (unlock counter > 0), 1 = active (unlock counter == 0).
[4]	R	0x0	bit	PrivilegeStatusFifo	Current status of privilege protection: 0 = inactive , 1 = active.
[8]	R	0x0	bit	FreezeStatusFifo	Current freeze status: 0 = protection status can be changed, 1 = cannot be changed.

3.7.30. InstructionFifo[0..31]

Instruction Fifo.

Bits	Type	Reset	Format	Field Name	Description
[31,0]	W	-	uint32	Instruction	Command instructions to be executed by the sequencer.
The sequencer (if IDLE) will start operation as soon as instructions are added to the fifo. Instruction format is upper byte for opcode and lower three bytes for parameter(s) if any. If the InstructionFifo is written while being full the configuration interface will wait for defined period for a free entry before discarding the Instruction and giving an ERROR response.					

3.7.31. RxFifo[0..15]

Reception Fifo.

Bits	Type	Reset	Format	Field Name	Description
[17,0]	R	-	uint18	RxEntry	Data received from the LCDBus.
Read data from the LCDBus packed according CommandTransferMapping or DataTransferMapping depending on the triggering instruction. If the RxFifo is read while being empty, the data will be unknown. No ERROR response is generated.					

CHAPTER 38: Appendix

This chapter explains pin status in each cpu state.

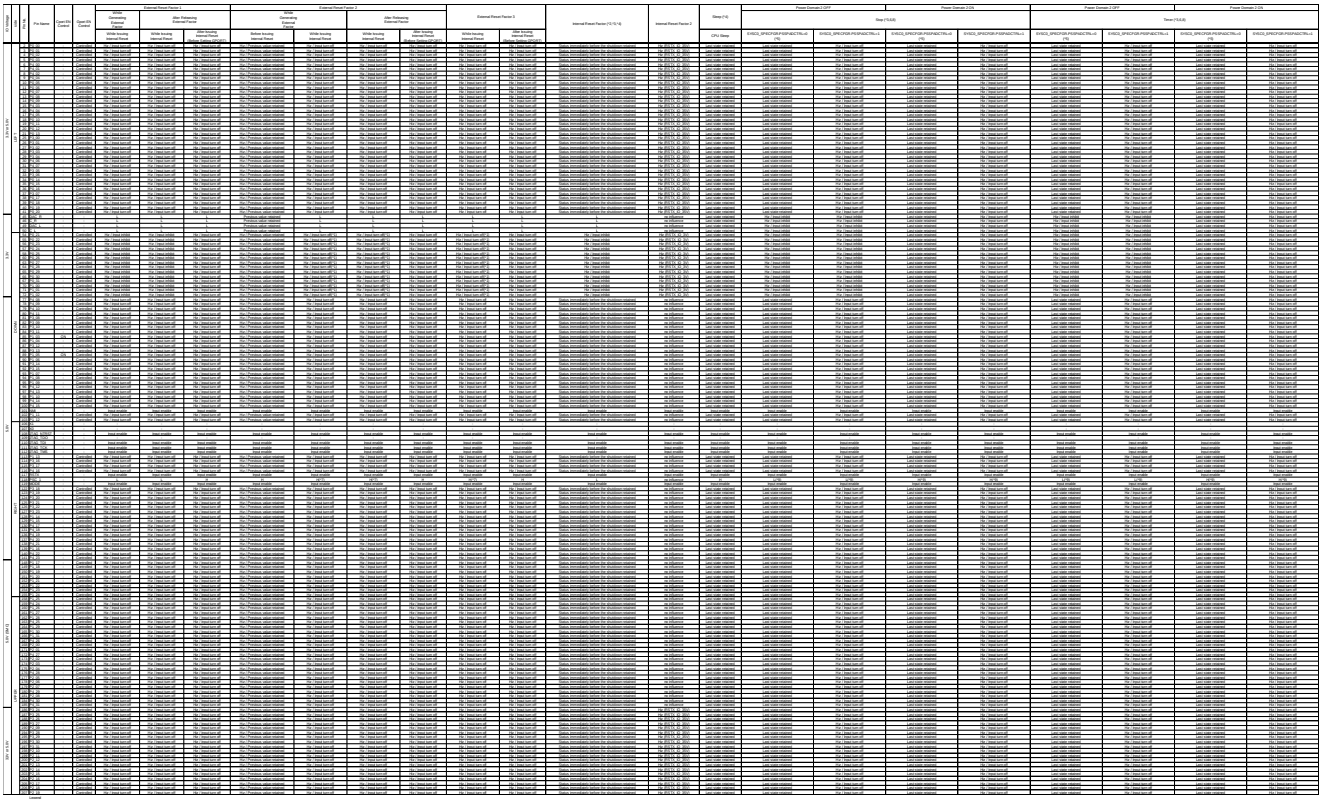
1. Pin Status in Each CPU State

CODE: APPENDIX-S6J3300-E1

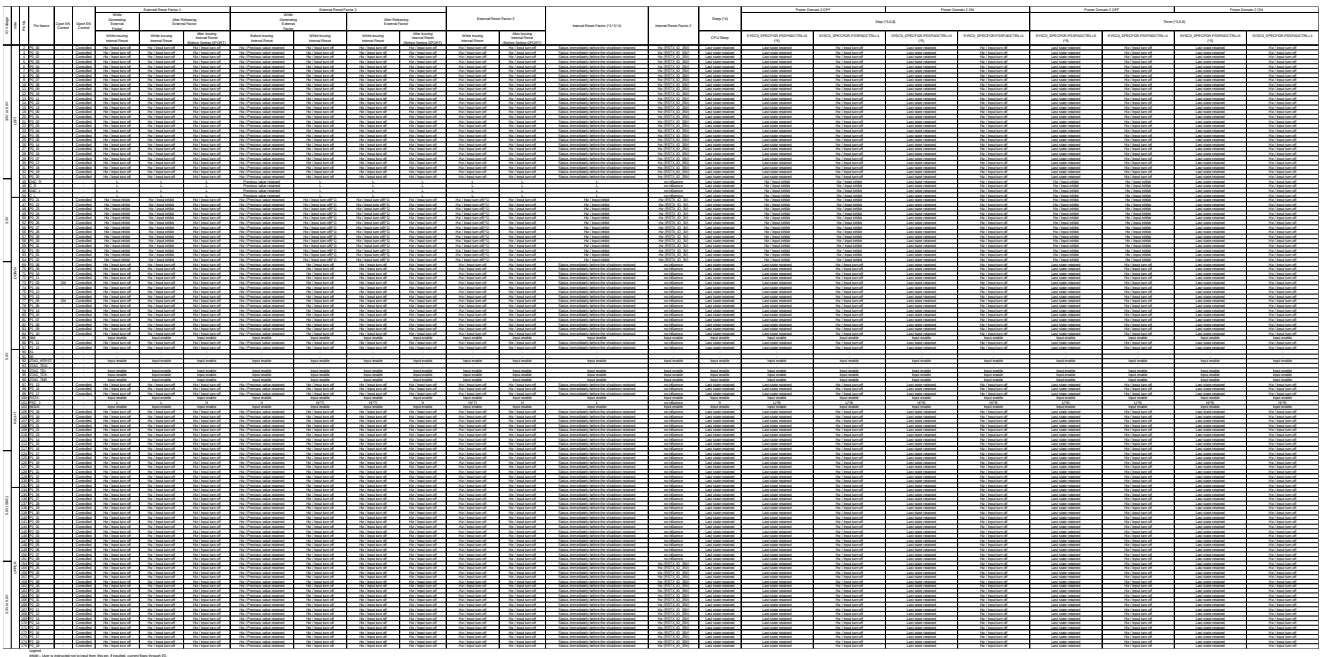
1. Pin Status in Each CPU State

This section shows the pin status in each cpu state.

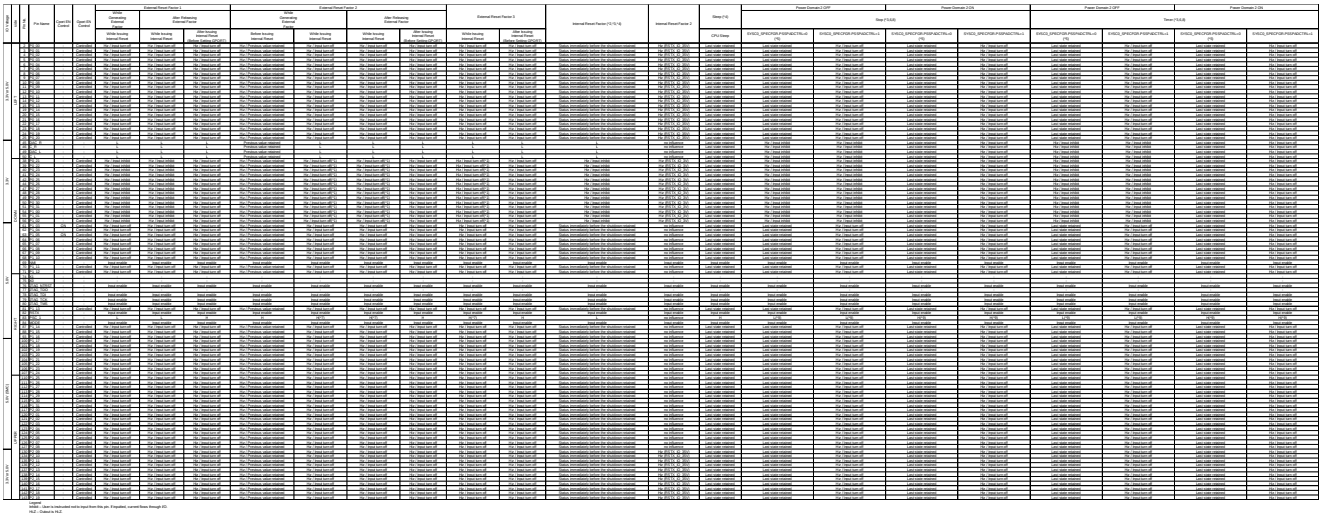
1.1. TEQFP-208 Pin



1.2. TEQFP-176 Pin



1.3. TEQFP-144 Pin



CHAPTER 39: Major Changes

<table>
<tr><th>Page</th><th>Section</th><th>Change Results</th></tr>
<tr><td colspan="3">Rev.*A</td></tr>
<tr><td>20</td><td>CHAPTER 1
Overview
1.Overview</td><td>Revised the below:
Error)
S6J3300 is a microcontroller series which is to be applied to automotive systems representative of a graphical cluster control unit on a dashboard.

Correct)
S6J3300 is a microcontroller series which is to be applied to automotive systems.
S6J3310/20/30/40 is a microcontroller series for instrument clusters with small thin-film transistor (TFT) displays.
S6J3350 is a microcontroller series for body and gateway.</td></tr>
<tr><td>21</td><td>CHAPTER 1
Overview
2.Document Definition</td><td>Revised the below:
Error)
Document Code
002-10186

Correct)
Document Code
002-10635(S6J3310/20/30/40)
002-10634(S6J3350)</td></tr>
<tr><td>28
38</td><td>CHAPTER 2
Function List
1.Function List

CHAPTER 3
Product Description
2.Product Description</td><td>Revised the below:
Error)
TC-RAM
System-RAM
Retention RAM

Correct)
TCRAM
System SRAM
Backup RAM</td></tr>
<tr><td>29</td><td>CHAPTER 2
Function List
1.Function List</td><td>Revised the below:
Error)
<table><tr><td>Graphic engine clock</td><td>200MHz (Max)</td></tr><tr><td>Graphic AXI clock</td><td>80MHz(Max)</td></tr><tr><td>Display clock</td><td>40MHz</td></tr></table>
Correct)
<table><tr><td>Graphic engine clock</td><td>80MHz (Max)</td></tr><tr><td>Graphic AXI clock</td><td>80MHz (Max)</td></tr><tr><td>Display clock</td><td>25MHz</td></tr></table></td></tr>
</table>

<table>
<tr><th>Page</th><th>Section</th><th>Change Results</th></tr>
<tr><td>29</td><td>CHAPTER 2
Function List
1.Function List</td><td>Deleted the below:
Error)
2.5D support Available
Warping Available</td></tr>
<tr><td>30</td><td>CHAPTER 2
Function List
2.1.Basic
Option</td><td>The MK_CEER option of T,E,F,V,G, and H part number added into the series.
Added the below:(2 places)
Error)
* Option :
<table>
<tr><th>Digit</th><th>SHE</th><th>VCC</th><th>DVCC</th></tr>
<tr><td>S</td><td rowspan="3">ON</td><td>5V</td><td>5V</td></tr>
<tr><td>A</td><td rowspan="2">3V</td><td>5V</td></tr>
<tr><td>B</td><td>3V</td></tr>
<tr><td>U</td><td rowspan="3">OFF</td><td>5V</td><td>5V</td></tr>
<tr><td>C</td><td rowspan="2">3V</td><td>5V</td></tr>
<tr><td>D</td><td>3V</td></tr>
</table>

Correct)
Option :
<table>
<tr><th>Digit</th><th>SHE</th><th>MK_CEER*</th><th>VCC</th><th>DVCC</th></tr>
<tr><td>S</td><td rowspan="3">ON</td><td rowspan="6">Fixed to Enable</td><td>5V</td><td rowspan="2">5V</td></tr>
<tr><td>A</td><td rowspan="2">3V</td></tr>
<tr><td>B</td><td>3V</td></tr>
<tr><td>U</td><td rowspan="3">OFF</td><td>5V</td><td rowspan="2">5V</td></tr>
<tr><td>C</td><td rowspan="2">3V</td></tr>
<tr><td>D</td><td>3V</td></tr>
<tr><td>T</td><td rowspan="3">ON</td><td rowspan="6">Selectable</td><td>5V</td><td rowspan="2">5V</td></tr>
<tr><td>E</td><td rowspan="2">3V</td></tr>
<tr><td>F</td><td>3V</td></tr>
<tr><td>V</td><td rowspan="3">OFF</td><td>5V</td><td rowspan="2">5V</td></tr>
<tr><td>G</td><td rowspan="2">3V</td></tr>
<tr><td>H</td><td>3V</td></tr>
</table>
* Chip Erase Enable Register</td></tr>
<tr><td>30</td><td>CHAPTER 2
Function List
2.1.Basic
Option</td><td>Revised the word “Main” in the “Flash column to “Progam” as below:(2 places)
Error)
* Memory size :
<table>
<tr><th rowspan="2">Digit</th><th colspan="2">Flash</th><th colspan="2">RAM</th></tr>
<tr><th>Main</th><th>Work</th><th>Main</th><th>Back up</th></tr>
<tr><td>E</td><td>4,160KB</td><td>112KB</td><td>512KB</td><td>16+16KB</td></tr>
<tr><td>D</td><td>3,136KB</td><td>112KB</td><td>512KB</td><td>16+16KB</td></tr>
<tr><td>C</td><td>2,112KB</td><td>112KB</td><td>512KB</td><td>16+16KB</td></tr>
<tr><td>B</td><td>1,600KB</td><td>112KB</td><td>512KB</td><td>16+16KB</td></tr>
</table>

Correct)
Memory size :
<table>
<tr><th rowspan="2">Digit</th><th colspan="2">Flash</th><th colspan="2">RAM</th></tr>
<tr><th>Program</th><th>Work</th><th>Main</th><th>Backup</th></tr>
<tr><td>E</td><td>4,160KB</td><td>112KB</td><td>512KB</td><td>16+16KB</td></tr>
<tr><td>D</td><td>3,136KB</td><td>112KB</td><td>384KB</td><td>16+16KB</td></tr>
<tr><td>C</td><td>2,112KB</td><td>112KB</td><td>256KB</td><td>16+16KB</td></tr>
<tr><td>B</td><td>1,600KB</td><td>112KB</td><td>192KB</td><td>16+16KB</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>Page</th><th>Section</th><th>Change Results</th></tr>
<tr><td>32</td><td>CHAPTER 2
Function List
2.3.Restriction</td><td>Revised AN33 to 38 as below:

Error)
<table><tr><td>Analog input port (12bit-ADC)</td><td>AN0 to 3, AN19 to 20,
AN22 to 23, AN27, AN33 to 34,
AN35 to 38, AN48</td><td>AN0 to 7, AN10 to 11,
AN14 to 15, AN19 to 20,
AN22 to 23, AN25 to AN27,
AN28 to 30, AN33 to 34,
AN35 to 38, AN48</td></tr></table>
Correct)
<table><tr><td>Analog input port (12bit-ADC)</td><td>AN0 to 3, AN19 to 20,
AN22 to 23, AN27, AN33 to 38,
AN48</td><td>AN0 to 7, AN10 to 11,
AN14 to 15, AN19 to 20,
AN22 to 23, AN25 to 30,
AN33 to 38, AN48</td></tr></table></td></tr>
<tr><td>32</td><td>CHAPTER 2
Function List
2.3.Restriction</td><td>Revised the below:
Error)
<table><tr><td>CAN</td><td>RX4_0, TX4_0
RX7_0, TX7_0</td><td>RX4_0, TX4_0
RX7_0, TX7_0
RX0_1, TX0_1
RX1_1, TX1_1
RX2_1, TX2_1
RX3_1, TX3_1
RX5_1, TX5_1
RX6_1, TX6_1</td></tr></table>
Correct)
<table><tr><td>CAN</td><td>RX0_2, TX0_2
RX3_2, TX3_2
RX4_0, TX4_0
RX7_0, TX7_0
RX7_1, TX7_1</td><td>RX0_1, TX0_1
RX0_2, TX0_2
RX1_1, TX1_1
RX2_1, TX2_1
RX3_1, TX3_1
RX3_2, TX3_2
RX4_0, TX4_0
RX5_1, TX5_1
RX6_1, TX6_1
RX7_0, TX7_0
RX7_1, TX7_1</td></tr></table></td></tr>
</table>

<table>
<tr><th>Page</th><th>Section</th><th>Change Results</th></tr>
<tr><td>32</td><td>CHAPTER 2
Function List
2.3.Restriction</td><td>Revised the below:
Error)
<table>
<tr><td>BaseTimer</td><td>PPG16/17/18/19/20_TOUT0_0
PPG16/17/18/19/20_TOUT2_0
PPG21/23/24/25/26_TOUT0_0
PPG21/23/24/25/26_TOUT2_0
PPG27/28/29/30/31_TOUT0_0
PPG27/28/29/30/31_TOUT2_0

PPG22/23/24/25/26/27_TIN1_0
PPG28/29/30/31_TIN1_0</td><td>PPG16/17/18/19/20_TOUT0_0
PPG16/17/18/19/20_TOUT2_0
PPG21/22/23/24/25/26_TOUT0_0
PPG21/22/23/24/25/26_TOUT2_0
PPG27/28/29/30/31_TOUT0_0
PPG27/28/29/30/31_TOUT2_0

PPG16/17/18/19/20/21_TIN1_0
PPG22/23/24/25/26/27_TIN1_0
PPG28/29/30/31_TIN1_0

PPG4/5/6/7/8/9_TOUT0_1
PPG4/5/6/7/8/9_TOUT2_1
PPG10/11/12/13/15_TOUT0_1
PPG10/11/12/13/14/15_TOUT2_1
PPG0/1/2/3/4/5_TIN1_1
PPG6/7/8/9/10/11_TIN1_1
PPG12/13/14/15_TIN1_1</td></tr>
</table>
Correct)
<table>
<tr><td>BaseTimer</td><td>PPG16/17/18/19_TOUT0_0
PPG16/17/18/19_TOUT2_0
PPG20/21/22/23_TOUT0_0
PPG20/21/22/23_TOUT2_0
PPG24/25/26/27_TOUT0_0
PPG24/25/26/27_TOUT2_0
PPG28/29/30/31_TOUT0_0
PPG28/29/30/31_TOUT2_0

PPG16/17/18/19/20/21_TIN1_0
PPG22/23/24/25/26/27_TIN1_0
PPG28/29/30/31_TIN1_0</td><td>PPG16/17/18/19_TOUT0_0
PPG16/17/18/19_TOUT2_0
PPG20/21/22/23_TOUT0_0
PPG20/21/22/23_TOUT2_0
PPG24/25/26/27_TOUT0_0
PPG24/25/26/27_TOUT2_0
PPG28/29/30/31_TOUT0_0
PPG28/29/30/31_TOUT2_0

PPG16/17/18/19/20/21_TIN1_0
PPG22/23/24/25/26/27_TIN1_0
PPG28/29/30/31_TIN1_0

PPG4/5/6/7/8/9_TOUT0_1
PPG4/5/6/7/8/9_TOUT2_1
PPG10/11/12/13/15_TOUT0_1
PPG10/11/12/13/14/15_TOUT2_1
PPG0/1/2/3/4/5_TIN1_1
PPG6/7/8/9/10/11_TIN1_1
PPG12/13/14/15_TIN1_1</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>Page</th><th>Section</th><th>Change Results</th></tr>
<tr><td>38</td><td>CHAPTER 3 Product Description 2.Product Description</td><td>Added notes as below:
Error)
<table><tr><td>Internal Memories System SRAM</td><td>384kByte</td></tr><tr><td>Embedded Program/Work Flash Memory</td><td>Embedded Program Flash can be accessed with 0-wait-cycle if CPU frequency is 80MHz or less.
0-wait-cycle: 80MHz or less.
1-wait-cycle: 160MHz or less.
2-wait-cycle: more than 160MHz.
The maximum frequency should be referred in datasheet.

Erase suspend is supported. Reading and writing to the other sector are possible when Flash Erase is suspended.
Serial Flash programing and Parallel Flash programing are supported.
Margin mode is not supported.</td></tr></table>
Correct)
<table><tr><td>Internal Memories System SRAM</td><td>384kByte
1 wait cycle is necessary for RAM read at over 120MHz.</td></tr><tr><td>Embedded Program/Work Flash Memory</td><td>Embedded Program Flash can be accessed with 0-wait-cycle if CPU frequency is 80MHz or less.
0-wait-cycle: 80MHz or less.
1-wait-cycle: 160MHz or less.
2-wait-cycle: more than 160MHz.

Work Flash can be accessed with 0-wait-cycle if CPU frequency is 12.5MHz or less.
7-wait-cycle: 80MHz or less.
13-wait-cycle: 160MHz or less.

The wait-cycle setting see the Traveo™ Platform Hardware Manual in details.
The maximum frequency should be referred in datasheet.

Erase suspend is supported. Reading and writing to the other sector are possible when Flash Erase is suspended.
Serial Flash programing and Parallel Flash programing are supported.
Margin mode is not supported.</td></tr></table></td></tr>
<tr><td>41</td><td>CHAPTER 3 Product Description 2.Product Description</td><td>Revised the below:
Error)
Note:
The description of the preliminary documentation will be changed without any notification.

Correct)
Note:
See the common information of the option features on the Traveo™ Platform Hardware Manual.
In the case of option of default and Other Type, S6J3300 select the default.</td></tr>
<tr><td colspan="3">Rev.*B</td></tr>
<tr><td>22</td><td>CHAPTER 1: Overview 3. Register Attribute</td><td>Revised the below:
Error)
3.1. Read and Write
RX
3.2. Protection
WP

Correct)
3.1. Read and Write
RX, RN
3.2. Protection
WP, RN</td></tr>
</table>

Page	Section	Change Results
37	CHAPTER 3 Product Description 2.Product Description	Revised the below: Error) System Control Main and sub oscillator is available. − A wide range of 3.6 - 4MHz is available for main oscillator Correct) System Control Main and sub oscillator is available. − A wide range of 3.6 - 16MHz is available for main oscillator
29 38 39 40	CHAPTER 2 Function List 1.Function List CHAPTER 3 Product Description 2.Product Description	Revised the below: Error) Detailed (TYPO) Correct) Detail
31	CHAPTER 2 Function List 2. Optional Function	Revised the below: Error) Chip ID | JTAG ID 0x10123000 | 0x1000B5CF Correct) See the Chip ID specification on the Datasheet in detail.
38	CHAPTER 3 Product Description 2.Product Description	Revised the below: Error) Retention RAM (TYPO) Correct) Backup RAM
40	CHAPTER 3 Product Description 2.Product Description	Revised Graphics Subsystem clock frequency as below: Error) 200 MHz maximum clock frequency Video modes up to 50 MHz pixel clock Correct) 80 MHz maximum clock frequency Video modes up to 25 MHz pixel clock
112	CHAPTER 10 Port Description 2.Port Description List	Revised P0_20 Description the below: Error) General-Purpose I/O port Correct) General-Purpose input port
231 232 233 234	CHAPTER 12: State Transition 3. Fetching the Operation Mode	Revised Operation Mode Fetch Timing Chart the below: Correct) Corrects of Operation Mode Fetch Timing Chart of Reset

<table>
<tr><th>Page</th><th>Section</th><th>Change Results</th></tr>
<tr><td>242</td><td>CHAPTER 13 Low Voltage Detection 3.Operation 3.1.LVD operation</td><td>Added notes as below:

Correct)
− The detection/release threshold values of following LVD channels are potentially below supply range defined in Recommended operating condition of Datasheet.
LVDL0
LVDL1
LVDL2
LVDH0
LVDH1
LVDH2

− Please use these LVD channels with your own risk
− Please monitor the external power supplies on the PCB if needed</td></tr>
<tr><td>243</td><td>CHAPTER 13:Low Voltage Detection 4.Registers</td><td>Revised the below:
Error)
[Bit26:25] LVDL1V Internal low-voltage detection voltage setting bits
<table>
<tr><th>Bit 26:25</th><th>Voltage [V]</th></tr>
<tr><td>00</td><td>0.875(Initial value)</td></tr>
<tr><td>01</td><td>0.95</td></tr>
<tr><td>10</td><td>1.00</td></tr>
<tr><td>11</td><td>1.025</td></tr>
</table>
Correct)
[Bit26:25] LVDL1V Internal low-voltage detection voltage setting bits
<table>
<tr><th>Bit 26:25</th><th>Voltage [V]</th></tr>
<tr><td>00</td><td>0.875(Initial value)</td></tr>
<tr><td>01</td><td>0.95</td></tr>
<tr><td>10</td><td>1.00</td></tr>
<tr><td>11</td><td>1.05</td></tr>
</table></td></tr>
<tr><td>244</td><td>CHAPTER 13:Low Voltage Detection 4.Registers</td><td>Revised the below:
Error)
[Bit18:17] LVDL2V Expanded internal low-voltage detection voltage setting bits
<table>
<tr><th>Bit 18:17</th><th>Voltage [V]</th></tr>
<tr><td>00</td><td>0.875</td></tr>
<tr><td>01</td><td>0.95</td></tr>
<tr><td>10</td><td>0.7875</td></tr>
<tr><td>11</td><td>0.8125(Initial value)</td></tr>
</table>
Correct)
[Bit18:17] LVDL2V Expanded internal low-voltage detection voltage setting bits
<table>
<tr><th>Bit 18:17</th><th>Voltage [V]</th></tr>
<tr><td>00</td><td>0.875</td></tr>
<tr><td>01</td><td>0.95</td></tr>
<tr><td>10</td><td>0.925</td></tr>
<tr><td>11</td><td>0.8125(Initial value)</td></tr>
</table></td></tr>
</table>

<table>
<tr><th>Page</th><th>Section</th><th>Change Results</th></tr>
<tr><td>244</td><td>CHAPTER 13:Low Voltage Detection 4.Registers</td><td>Revised the below:
Error)
[Bit11:9] LVDH1V External low-voltage detection voltage setting bits
<table>
<tr><th>Bit 11:9</th><th>Voltage [V]</th></tr>
<tr><td>000</td><td>2.7</td></tr>
<tr><td>001</td><td>2.8</td></tr>
<tr><td>010</td><td>2.9</td></tr>
<tr><td>011</td><td>3.8</td></tr>
<tr><td>100</td><td>4.0(initial value for Chip ID :0x1000B5CF product)</td></tr>
<tr><td>101</td><td>4.2</td></tr>
<tr><td>110</td><td>2.5</td></tr>
<tr><td>111</td><td>2.6(initial value for Chip ID: 0x0012A000 product and ID Code: 0x00122000 product)</td></tr>
</table>
Correct)
[Bit11:9] LVDH1V External low-voltage detection voltage setting bits
<table>
<tr><th>Bit 11:9</th><th>Voltage [V]</th></tr>
<tr><td>000</td><td>2.7</td></tr>
<tr><td>001</td><td>2.8</td></tr>
<tr><td>010</td><td>3.6</td></tr>
<tr><td>011</td><td>3.8</td></tr>
<tr><td>100</td><td>4.0(initial value for Chip ID : 0x10122xxx product)</td></tr>
<tr><td>101</td><td>4.2</td></tr>
<tr><td>110</td><td>2.5</td></tr>
<tr><td>111</td><td>2.6(initial value for Chip ID: 0x10128xxx product and 0x10120xxx product)</td></tr>
</table></td></tr>
<tr><td>245</td><td></td><td>Revised the below:
Error)
[Bit3:1] LVDH2V Expanded external low-voltage detection voltage setting bits
<table>
<tr><th>Bit 3:1</th><th>Voltage [V]</th></tr>
<tr><td>000</td><td>2.7</td></tr>
<tr><td>001</td><td>2.8</td></tr>
<tr><td>010</td><td>2.9</td></tr>
<tr><td>011</td><td>3.8</td></tr>
<tr><td>100</td><td>4.0</td></tr>
<tr><td>101</td><td>4.2</td></tr>
<tr><td>110</td><td>2.5</td></tr>
<tr><td>111</td><td>2.6(Initial value)</td></tr>
</table>
Correct)
[Bit3:1] LVDH2V Expanded external low-voltage detection voltage setting bits
<table>
<tr><th>Bit 3:1</th><th>Voltage [V]</th></tr>
<tr><td>000</td><td>2.7</td></tr>
<tr><td>001</td><td>2.8</td></tr>
<tr><td>010</td><td>3.6</td></tr>
<tr><td>011</td><td>3.8</td></tr>
<tr><td>100</td><td>4.0</td></tr>
<tr><td>101</td><td>4.2</td></tr>
<tr><td>110</td><td>2.5</td></tr>
<tr><td>111</td><td>2.6(Initial value)</td></tr>
</table></td></tr>
<tr><td>252</td><td>CHAPTER 14: Serial Programming 5. Operation 5.1. Timing Chart</td><td>Revised the below:
Error)
Operation state name of Oscillator Stabilization time
2.33ms(Max)

Correct)
Operation state name of Oscillator and flash Stabilization time
2.93ms(Max)</td></tr>
</table>

<table>
<tr><th>Page</th><th>Section</th><th>Change Results</th></tr>
<tr><td>484</td><td>CHAPTER 19: Sound Waveform Generator
3. Operation of the Sound Waveform Generator
3.2. Interrupt</td><td>Revised the below:
Error)
SWFG AHB Slave Interface access error

Correct)
SWFG AHB Master Interface access error</td></tr>
<tr><td>499</td><td>CHAPTER 19: Sound Waveform Generator
4. Registers</td><td>Revised the below:
4.7. Waveform Generator Channel n Control Register1 (WGCHnCTRL1, n=0 to 4)
[bit20:16] FSLEN

Error)
(Non)

Correct)
Note
When 00000 is written to these bits, the read value will be 00001 instead of 00000.</td></tr>
<tr><td>510
511
514</td><td>CHAPTER 19: Sound Waveform Generator
4. Registers</td><td>Revised the below:
4.12. Waveform Generator Interrupt State Register (WGINTRSTATE)
[bit31:13, 7:1] Reserved
4.14. Waveform Generator AHB Bus Error Register (WGAHBERR)
[bit31:2] Reserved

Error)
(Non)

Correct)
This bit is read-only, and writing is prohibited.
Writing to this bit generates an SWFG AHB Slave interface access error.</td></tr>
<tr><td>530</td><td>CHAPTER 20: Sound Mixer
4. Registers</td><td>Revised the below:
4.1. Mixer Channel Register (MXCH)
[bit9:5] PMIS4 to PMIS0

Error)
<table><tr><th>bit</th><th>Description</th></tr><tr><td>0</td><td>WFGn input invalid</td></tr><tr><td>1</td><td>WFGn input valid</td></tr></table>
Correct)
<table><tr><th>PMISn (n=0 to 4)</th><th>Description</th></tr><tr><td>0</td><td>PMISn input invalid</td></tr><tr><td>1</td><td>PMISn input valid</td></tr></table></td></tr>
<tr><td>530</td><td>CHAPTER 20: Sound Mixer
4. Registers</td><td>Revised the below:
4.1. Mixer Channel Register (MXCH)
[bit4:0] WFG4 to WFG0

Error)
<table><tr><th>bit</th><th>Description</th></tr><tr><td>0</td><td>WFGn input invalid</td></tr><tr><td>1</td><td>WFGn input valid</td></tr></table>
Correct)
<table><tr><th>WFGn (n=0 to 4)</th><th>Description</th></tr><tr><td>0</td><td>WFGn input invalid</td></tr><tr><td>1</td><td>WFGn input valid</td></tr></table></td></tr>
</table>

<table>
<tr><th>Page</th><th>Section</th><th>Change Results</th></tr>
<tr><td>542</td><td>CHAPTER 20: Sound Mixer
4. Registers</td><td>Revised the below:
4.9. Mixer Channel Mute Register (MXCHMUTE)
[bit9:5] PMIS4MUTE to PMIS0MUTE

Error)
<table><tr><th>bit</th><th>Description</th></tr><tr><td>0</td><td>WFGn input invalid</td></tr><tr><td>1</td><td>WFGn input valid</td></tr></table>
Correct)
<table><tr><th>PMISnMUTE (n=0 to 4)</th><th>Description</th></tr><tr><td>0</td><td>PMISn input invalid</td></tr><tr><td>1</td><td>PMISn input valid</td></tr></table></td></tr>
<tr><td>542</td><td>CHAPTER 20: Sound Mixer
4. Registers</td><td>Revised the below:
4.9. Mixer Channel Mute Register (MXCHMUTE)
[bit4:0] WFG4MUTE to WFG0MUTE

Error)
<table><tr><th>bit</th><th>Description</th></tr><tr><td>0</td><td>WFGn input invalid</td></tr><tr><td>1</td><td>WFGn input valid</td></tr></table>
Correct)
<table><tr><th>WFGnMUTE (n=0 to 4)</th><th>Description</th></tr><tr><td>0</td><td>WFGn input invalid</td></tr><tr><td>1</td><td>WFGn input valid</td></tr></table></td></tr>
<tr><td>550</td><td>CHAPTER 20: Sound Mixer
4. Registers</td><td>Revised the below:
4.16. Mixer Channel Fade_in/out Enable Register (MXCHFADEEN)
[bit21:20] MXDFADEEN

Error)
(Non)

Correct)
<table><tr><th>Bit[1:0]</th><th>Description</th></tr><tr><td>00</td><td>No fade in/fade out</td></tr><tr><td>01</td><td>Setting prohibited</td></tr><tr><td>10</td><td>Enable fade in operation start.</td></tr><tr><td>11</td><td>Enable fade out operation start.</td></tr></table></td></tr>
<tr><td>550
551</td><td>CHAPTER 20: Sound Mixer
4. Registers</td><td>Revised the below:
4.16. Mixer Channel Fade_in/out Enable Register (MXCHFADEEN)
[bit19:10] PMIS0FADEEN to PMIS0FADEEN

Error)
(Non)

Correct)
<table><tr><th>PMISnFADEEN[1:0] (n=0 to 4)</th><th>Description</th></tr><tr><td>00</td><td>No fade in/fade out</td></tr><tr><td>01</td><td>Setting prohibited</td></tr><tr><td>10</td><td>Enable fade in operation start.</td></tr><tr><td>11</td><td>Enable fade out operation start.</td></tr></table></td></tr>
</table>

<table>
<tr><th>Page</th><th>Section</th><th>Change Results</th></tr>
<tr><td>551</td><td>CHAPTER 20: Sound Mixer
4. Registers</td><td>Revised the below:
4.16. Mixer Channel Fade_in/out Enable Register (MXCHFADEEN)
[bit9:0] WFG4FADEEN to WFG0FADEEN

Error)
<table>
<tr><th>Bit[1:0]</th><th>Description</th></tr>
<tr><td>00</td><td>No fade in/fade out</td></tr>
<tr><td>01</td><td>Setting prohibited</td></tr>
<tr><td>10</td><td>Enable fade in operation start.</td></tr>
<tr><td>11</td><td>Enable fade out operation start.</td></tr>
</table>
Correct)
<table>
<tr><th>WFGnFADEEN[1:0]
(n=0 to 4)</th><th>Description</th></tr>
<tr><td>00</td><td>No fade in/fade out</td></tr>
<tr><td>01</td><td>Setting prohibited</td></tr>
<tr><td>10</td><td>Enable fade in operation start.</td></tr>
<tr><td>11</td><td>Enable fade out operation start.</td></tr>
</table></td></tr>
<tr><td>552</td><td>CHAPTER 20: Sound Mixer
4. Registers</td><td>Revised the below:
4.17. Mixer Buffer Clear Register (MXBUFFCLR)
[bit10] OUTBCLR

Error)
(Non)

Correct)
<table>
<tr><th rowspan="2">Bit</th><th colspan="2">Description</th></tr>
<tr><th>Read</th><th>Write</th></tr>
<tr><td>0</td><td>Not under initialization</td><td>Does not have any effect on operation.</td></tr>
<tr><td>1</td><td>Initialization wait</td><td>Initialize buffer (This bit automatically cleared to 0 following initialization.)</td></tr>
</table></td></tr>
<tr><td>553</td><td>CHAPTER 20: Sound Mixer
4. Registers</td><td>Revised the below:
4.17. Mixer Buffer Clear Register (MXBUFFCLR)
[bit9:5] PMIS4BCLR to PMIS0BCLR

Error)
(Non)

Correct)
<table>
<tr><th rowspan="2">PMISnBCLR
(n=0 to 4)</th><th colspan="2">Description</th></tr>
<tr><th>Read</th><th>Write</th></tr>
<tr><td>0</td><td>Not under initialization</td><td>Does not have any effect on operation.</td></tr>
<tr><td>1</td><td>Initialization wait</td><td>Initialize buffer (This bit automatically cleared to 0 following initialization.)</td></tr>
</table></td></tr>
<tr><td>553</td><td>CHAPTER 20: Sound Mixer
4. Registers</td><td>Revised the below:
4.17. Mixer Buffer Clear Register (MXBUFFCLR)
[bit4:0] WFG4BCLR to WFG0BCLR

Error)
<table>
<tr><th rowspan="2">Bit</th><th colspan="2">Description</th></tr>
<tr><th>Read</th><th>Write</th></tr>
<tr><td>0</td><td>Not under initialization</td><td>Does not have any effect on operation.</td></tr>
<tr><td>1</td><td>Initialization wait</td><td>Initialize buffer (This bit automatically cleared to 0 following initialization.)</td></tr>
</table>
Correct)
<table>
<tr><th rowspan="2">WFGnBCLR
(n=0 to 4)</th><th colspan="2">Description</th></tr>
<tr><th>Read</th><th>Write</th></tr>
<tr><td>0</td><td>Not under initialization</td><td>Does not have any effect on operation.</td></tr>
<tr><td>1</td><td>Initialization wait</td><td>Initialize buffer (This bit automatically cleared to 0 following initialization.)</td></tr>
</table></td></tr>
</table>

<table>
<tr><th>Page</th><th>Section</th><th>Change Results</th></tr>
<tr><td>554</td><td>CHAPTER 20: Sound Mixer
4. Registers</td><td>Revised the below:
4.18. Mixer FADE_in/out Clear Register (MXFADECLR)
[bit10] MIXCLR

Error)
(Non)

Correct)
<table>
<tr><th rowspan="2">Bit</th><th colspan="2">Description</th></tr>
<tr><th>Read</th><th>Write</th></tr>
<tr><td>0</td><td>Not under initialization</td><td>Does not have any effect on operation.</td></tr>
<tr><td>1</td><td>Initialization wait</td><td>Initialize buffer (This bit automatically cleared to 0 following initialization.)</td></tr>
</table></td></tr>
<tr><td>555</td><td>CHAPTER 20: Sound Mixer
4. Registers</td><td>Revised the below:
4.18. Mixer FADE_in/out Clear Register (MXFADECLR)
[bit9:5] PMIS4CLR to PMIS0CLR

Error)
(Non)

Correct)
<table>
<tr><th rowspan="2">PMISnCLR
(n=0 to 4))</th><th colspan="2">Description</th></tr>
<tr><th>Read</th><th>Write</th></tr>
<tr><td>0</td><td>Not under initialization</td><td>Does not have any effect on operation.</td></tr>
<tr><td>1</td><td>Initialization wait</td><td>Initialize buffer (This bit automatically cleared to 0 following initialization.)</td></tr>
</table></td></tr>
<tr><td>555</td><td>CHAPTER 20: Sound Mixer
4. Registers</td><td>Revised the below:
4.18. Mixer FADE_in/out Clear Register (MXFADECLR)
[bit4:0] WFG4CLR to WFG0CLR

Error)
<table>
<tr><th rowspan="2">Bit</th><th colspan="2">Description</th></tr>
<tr><th>Read</th><th>Write</th></tr>
<tr><td>0</td><td>Not under initialization</td><td>Does not have any effect on operation.</td></tr>
<tr><td>1</td><td>Initialization wait</td><td>Initialize buffer (This bit automatically cleared to 0 following initialization.)</td></tr>
</table>
Correct)
<table>
<tr><th rowspan="2">WFGnCLR
(n=0 to 4)</th><th colspan="2">Description</th></tr>
<tr><th>Read</th><th>Write</th></tr>
<tr><td>0</td><td>Not under initialization</td><td>Does not have any effect on operation.</td></tr>
<tr><td>1</td><td>Initialization wait</td><td>Initialize buffer (This bit automatically cleared to 0 following initialization.)</td></tr>
</table></td></tr>
<tr><td>556</td><td>CHAPTER 20: Sound Mixer
4. Registers</td><td>Revised the below:
4.19. Mixer Interrupt Enable Register (MXINTREN)
[bit31] AHBERR

Error)
(Non)

Correct)
<table>
<tr><th>Bit</th><th>Description</th></tr>
<tr><td>0</td><td>Disable interrupts.</td></tr>
<tr><td>1</td><td>Enable interrupts.</td></tr>
</table></td></tr>
<tr><td>557</td><td>CHAPTER 20: Sound Mixer
4. Registers</td><td>Revised the below:
4.19. Mixer Interrupt Enable Register (MXINTREN)
[bit28:24] PMIS4DMAERR to PMIS0DMAERR

Error)
(Non)

Correct)
<table>
<tr><th>PMISnDMAERR
(n=0 to 4)</th><th>Description</th></tr>
<tr><td>0</td><td>Disable interrupts.</td></tr>
<tr><td>1</td><td>Enable interrupts.</td></tr>
</table></td></tr>
</table>

<table>
<tr><th>Page</th><th>Section</th><th>Change Results</th></tr>
<tr><td>557</td><td>CHAPTER 20: Sound Mixer
4. Registers</td><td>Revised the below:
4.19. Mixer Interrupt Enable Register (MXINTREN)
[bit12:8] PMIS4BUFOVFL to PMIS0BUFOVFL

Error)
(Non)

Correct)
<table><tr><th>PMISnBUFOVFL (n=0 to 4)</th><th>Description</th></tr><tr><td>0</td><td>Disable interrupts.</td></tr><tr><td>1</td><td>Enable interrupts.</td></tr></table></td></tr>
<tr><td>557</td><td>CHAPTER 20: Sound Mixer
4. Registers</td><td>Revised the below:
4.19. Mixer Interrupt Enable Register (MXINTREN)
[bit4:0] PMIS4BUFDREQ to PMIS0BUFDREQ

Error)
<table><tr><th>Bit</th><th>Description</th></tr><tr><td>0</td><td>Disable interrupts.</td></tr><tr><td>1</td><td>Enable interrupts.</td></tr></table>
Correct)
<table><tr><th>PMISnBUFDREQ (n=0 to 4)</th><th>Description</th></tr><tr><td>0</td><td>Disable interrupts.</td></tr><tr><td>1</td><td>Enable interrupts.</td></tr></table></td></tr>
<tr><td>558</td><td>CHAPTER 20: Sound Mixer
4. Registers</td><td>Revised the below:
4.20. Mixer Interrupt Status Register (MXINTRSTATE)
[bit31] AHBERR

Error)
(Non)

Correct)
<table><tr><th>Bit</th><th>Description</th></tr><tr><td>0</td><td>No error/no transfer request</td></tr><tr><td>1</td><td>Error/transfer request present</td></tr></table>
Notes:
- This bit is read-only, and writing is prohibited.
- Writing to this bit generates a sound mixer AHB Slave Interface access error.</td></tr>
<tr><td>558
559
563
564
566
567
568
569
570
571</td><td>CHAPTER 20: Sound Mixer
4. Registers</td><td>Revised the below:
4.20. Mixer Interrupt Status Register (MXINTRSTATE)
[bit30:29, 23:13, 7:5] Reserved
4.22. Mixer Input Buffer Count1 Register (MXINBUFFCNT1)
[bit31:30, 23:22, 15:14, 7:5] Reserved
4.23. Mixer Input Buffer Count2 Register (MXINBUFFCNT2)
[bit31:14, 7:6] Reserved
4.24. Mixer Channel Buffer Count Register (MXCHBUFFCNT)
[bit31:28, 7:5] Reserved
4.25. Mixer Output Buffer Count Register (MXOUTBUFFCNT)
[bit31:6] Reserved
4.26. Mixer AHB Bus Error Register (MXAHBERR)
[bit31:22] Reserved

Error)
(Non)

Correct)
This bit is read-only, and writing is prohibited.
Writing to this bit generates a sound mixer AHB Slave interface access error.</td></tr>
</table>

<table>
<tr><th>Page</th><th>Section</th><th>Change Results</th></tr>
<tr><td>559</td><td>CHAPTER 20: Sound Mixer
4. Registers</td><td>Revised the below:
4.20. Mixer Interrupt Status Register (MXINTRSTATE)
[bit28:24] PMIS4DMAERR to PMIS0DMAERR

Error)
(Non)

Correct)
<table>
<tr><th>PMISnDMAERR (n=0 to 4)</th><th>Description</th></tr>
<tr><td>0</td><td>No error/no transfer request.</td></tr>
<tr><td>1</td><td>Error/transfer request present.</td></tr>
</table>
Notes:
- This bit is read-only, and writing is prohibited.
- Writing to this bit generates a sound mixer AHB Slave Interface access error.</td></tr>
<tr><td>559</td><td>CHAPTER 20: Sound Mixer
4. Registers</td><td>Revised the below:
4.20. Mixer Interrupt Status Register (MXINTRSTATE)
[bit12:8] PMIS4BUFOVFL to PMIS0BUFOVFL

Error)
(Non)

Correct)
<table>
<tr><th>PMISnBUFOVFL (n=0 to 4)</th><th>Description</th></tr>
<tr><td>0</td><td>No error/no transfer request.</td></tr>
<tr><td>1</td><td>Error/transfer request present.</td></tr>
</table>
Notes:
- This bit is read-only, and writing is prohibited.
- Writing to this bit generates a sound mixer AHB Slave Interface access error.</td></tr>
<tr><td>560</td><td>CHAPTER 20: Sound Mixer
4. Registers</td><td>Revised the below:
4.20. Mixer Interrupt Status Register (MXINTRSTATE)
[bit4:0] PMIS4BUFDREQ to PMIS0BUFDREQ

Error)
<table>
<tr><th>Bit</th><th>Description</th></tr>
<tr><td>0</td><td>Disable interrupts.</td></tr>
<tr><td>1</td><td>Enable interrupts.</td></tr>
</table>
Correct)
<table>
<tr><th>PMISnBUFDREQ (n=0 to 4)</th><th>Description</th></tr>
<tr><td>0</td><td>Disable interrupts.</td></tr>
<tr><td>1</td><td>Enable interrupts.</td></tr>
</table></td></tr>
<tr><td>561</td><td>CHAPTER 20: Sound Mixer
4. Registers</td><td>Revised the below:
4.21. Mixer Interrupt Clear Register (MXINTRCLR)
[bit31] AHBERR

Error)
(Non)
Correct)
<table>
<tr><th>Bit</th><th>Description</th></tr>
<tr><td>0</td><td>Does not have any effect on operation.</td></tr>
<tr><td>1</td><td>Clear interrupts.</td></tr>
</table></td></tr>
</table>

<table>
<tr><th>Page</th><th>Section</th><th>Change Results</th></tr>
<tr><td>562</td><td>CHAPTER 20: Sound Mixer
4. Registers</td><td>Revised the below:
4.21. Mixer Interrupt Clear Register (MXINTRCLR)
[bit28:24] PMIS4DMAERR to PMIS0DMAERR

Error)
(Non)

Correct)
<table>
<tr><th>PMISnDMAERR (n=0 to 4)</th><th>Description</th></tr>
<tr><td>0</td><td>Does not have any effect on operation.</td></tr>
<tr><td>1</td><td>Clear interrupts.</td></tr>
</table></td></tr>
<tr><td>562</td><td>CHAPTER 20: Sound Mixer
4. Registers</td><td>Revised the below:
4.21. Mixer Interrupt Clear Register (MXINTRCLR)
[bit12:8] PMIS4BUFOVFL to PMIS0BUFOVFL

Error)
(Non)

Correct)
<table>
<tr><th>PMISnBUFOVFL (n=0 to 4)</th><th>Description</th></tr>
<tr><td>0</td><td>Does not have any effect on operation.</td></tr>
<tr><td>1</td><td>Clear interrupts.</td></tr>
</table></td></tr>
<tr><td>562</td><td>CHAPTER 20: Sound Mixer
4. Registers</td><td>Revised the below:
4.21. Mixer Interrupt Clear Register (MXINTRCLR)
[bit4:0] PMIS4BUFDREQ to PMIS0BUFDREQ

Error)
<table>
<tr><th>CHnEND (n=0 to 4)</th><th>Description</th></tr>
<tr><td>0</td><td>Does not have any effect on operation.</td></tr>
<tr><td>1</td><td>Clear interrupts.</td></tr>
</table>
Correct)
<table>
<tr><th>PMISnBUFDREQ (n=0 to 4)</th><th>Description</th></tr>
<tr><td>0</td><td>Does not have any effect on operation.</td></tr>
<tr><td>1</td><td>Clear interrupts.</td></tr>
</table></td></tr>
<tr><td>563
564
566</td><td>CHAPTER 20: Sound Mixer
4. Registers</td><td>Revised the below:
4.22. Mixer Input Buffer Count1 Register (MXINBUFFCNT1)
[bit29:24] PMIS2CNT[5:0], [bit21:16] PMIS1CNT[5:0]
4.23. Mixer Input Buffer Count2 Register (MXINBUFFCNT2)
[bit13:8] PMIS4CNT[5:0]

Error)
(Non)

Correct)
<table>
<tr><th>Bit[5:0]</th><th>Description</th></tr>
<tr><td>000000</td><td>Buffer is empty.</td></tr>
<tr><td>000001
To
111111</td><td>Used input buffer volume</td></tr>
</table>
Notes:
- This bit is read-only, and writing is prohibited.
- Writing to this bit generates a sound mixer AHB Slave Interface access error.</td></tr>
</table>

<table>
<tr><th>Page</th><th>Section</th><th>Change Results</th></tr>
<tr><td>565</td><td>CHAPTER 20: Sound Mixer
4. Registers</td><td>Revised the below:
4.22. Mixer Input Buffer Count1 Register (MXINBUFFCNT1)
[bit4:0] WFG4C to WFG0C

Error)
<table><tr><th>Bit</th><th>Description</th></tr><tr><td>0</td><td>Without data.</td></tr><tr><td>1</td><td>With data</td></tr></table>
Correct)
<table><tr><th>WFGnC
(n=0 to 4)</th><th>Description</th></tr><tr><td>0</td><td>Without data.</td></tr><tr><td>1</td><td>With data</td></tr></table></td></tr>
<tr><td>568
569</td><td>CHAPTER 20: Sound Mixer
4. Registers</td><td>Revised the below:
4.24. Mixer Channel Buffer Count Register (MXCHBUFFCNT)
[bit27:8] PMIS4CNT[3:0] to PMIS0CNT[3:0]

Error)
(Non)

Correct)
<table><tr><th>PMISnCNT[3:0]
(n=0 to 4)</th><th>Description</th></tr><tr><td>000000</td><td>Buffer is empty.</td></tr><tr><td>000001
To
111111</td><td>Internal buffer used volume display</td></tr></table>
Notes:
- This bit is read-only, and writing is prohibited.
- Writing to this bit generates a sound mixer AHB Slave Interface access error.</td></tr>
<tr><td>569</td><td>CHAPTER 20: Sound Mixer
4. Registers</td><td>Revised the below:
4.24. Mixer Channel Buffer Count Register (MXCHBUFFCNT)
[bit4:0] WFG4C to WFG0C

Error)
<table><tr><th>Bit[5:0]</th><th>Description</th></tr><tr><td>000000</td><td>Buffer is empty.</td></tr><tr><td>000001
To
111111</td><td>Internal buffer used volume display</td></tr></table>
Correct)
<table><tr><th>WFGnC
(n=0 to 4)</th><th>Description</th></tr><tr><td>0</td><td>Buffer is empty</td></tr><tr><td>1</td><td>Internal buffer used</td></tr></table></td></tr>
<tr><td>571</td><td>CHAPTER 20: Sound Mixer
4. Registers</td><td>Revised the below:
4.26. Mixer AHB Bus Error Register (MXAHBERR)
[bit21:20] CNTREGERR[1:0]

Error)
(Non)

Correct)
<table><tr><th>Bit[1:0]</th><th>Description</th></tr><tr><td>00</td><td>There is no error.</td></tr><tr><td>01</td><td>Address error</td></tr><tr><td>10</td><td>Write access to Read Only register</td></tr><tr><td>11</td><td>Access size error</td></tr></table></td></tr>
</table>

<table>
<tr><th>Page</th><th>Section</th><th>Change Results</th></tr>
<tr><td>572</td><td>CHAPTER 20: Sound Mixer
4. Registers</td><td>Revised the below:
4.26. Mixer AHB Bus Error Register (MXAHBERR)
[bit19:10] PMIS4ERR[1:0] to PMIS0ERR[1:0]

Error)
(Non)

Correct)
<table>
<tr><th>PMISnERR[1:0] (n=0 to 4)</th><th>Description</th></tr>
<tr><td>00</td><td>There is no error.</td></tr>
<tr><td>01</td><td>Address error</td></tr>
<tr><td>10</td><td>Write access to Read Only register</td></tr>
<tr><td>11</td><td>Access size error</td></tr>
</table></td></tr>
<tr><td>572</td><td>CHAPTER 20: Sound Mixer
4. Registers</td><td>Revised the below:
4.26. Mixer AHB Bus Error Register (MXAHBERR)
[bit9:0] WFG4ERR[1:0] to WFG0ERR[1:0]

Error)
<table>
<tr><th>Bit[1:0]</th><th>Description</th></tr>
<tr><td>00</td><td>There is no error.</td></tr>
<tr><td>01</td><td>Address error</td></tr>
<tr><td>10</td><td>Write access to Read Only register</td></tr>
<tr><td>11</td><td>Access size error</td></tr>
</table>
Correct)
<table>
<tr><th>WFGnERR[1:0] (n=0 to 4)</th><th>Description</th></tr>
<tr><td>00</td><td>There is no error.</td></tr>
<tr><td>01</td><td>Address error</td></tr>
<tr><td>10</td><td>Write access to Read Only register</td></tr>
<tr><td>11</td><td>Access size error</td></tr>
</table></td></tr>
<tr><td>573</td><td>CHAPTER 20: Sound Mixer
4. Registers</td><td>Revised the below:
4.27. Mixer WFGn Data Address Register
[bit31:0] WFGnDADR [31:0]
Error)
Note:
Sound waveform generator output may have a fixed connection to WFG0 to 4. It should be noted that there may be specification limits on specific models.
If sound waveform generator output has a fixed connection, this register cannot be written to from the CPU. Writing generates an error.

Correct)
Note:
Because Sound waveform generator output has a fixed connection to this register, this register cannot be written to from the CPU. Writing generates an error.</td></tr>
</table>

<table>
<tr><th>Page</th><th>Section</th><th>Change Results</th></tr>
<tr><td>1152
1153
1155
1156
1167
1168
1172
1173
1175
1194</td><td>CHAPTER 33:Graphic Subsystem
1.1.About This Document
2.1.Feature Summary
2.2.2.2D Core
2.3.Functional Limitations
3.2.4.2Register Protection
3.2.5.Interrupts
3.3.1.Display Static Single-Thread
3.3.2.Display Dynamic Single-Thread
3.4.2D Core
3.6.SW Interface
3.6.3.2D Core</td><td>Revised revision 1.02 to 1.1 as below:
Error)
HW Peripheral Manual for 2D Graphic Core v1.02

Correct)
HW Peripheral Manual for 2D Graphic Core v1.1</td></tr>
<tr><td>1179</td><td>CHAPTER 33:Graphic Subsystem
3.6.2.3 IPIdentifier</td><td>Revised INITIAL_VALUE as below:
Error)
<table>
<tr><td>BIT_OFFSET</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td></tr>
<tr><td>BIT_NAME</td><td colspan="4">DesignDeliveryID[4:0]</td><td>Reserved</td><td>Reserved</td></tr>
<tr><td>ACCESS_TYPE</td><td>RW</td><td>RW</td><td>RW</td><td>RW</td><td>RW</td><td>RW</td></tr>
<tr><td>PROT_TYPE</td><td colspan="6">-</td></tr>
<tr><td>INITIAL_VALUE</td><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td><td>0</td></tr>
</table>
Correct)
<table>
<tr><td>BIT_OFFSET</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td></tr>
<tr><td>BIT_NAME</td><td colspan="4">DesignDeliveryID[3:0]</td><td>Reserved</td><td>Reserved</td></tr>
<tr><td>ACCESS_TYPE</td><td>RW</td><td>RW</td><td>RW</td><td>RW</td><td>RW</td><td>RW</td></tr>
<tr><td>PROT_TYPE</td><td colspan="6">-</td></tr>
<tr><td>INITIAL_VALUE</td><td>0</td><td>0</td><td>1</td><td>1</td><td>0</td><td>0</td></tr>
</table></td></tr>
<tr><td>1179</td><td>CHAPTER 33:Graphic Subsystem
3.6.3.2D Core</td><td>Revised the bit length as below:
Error)
IPFamily[4:0]
IPConfiguration[4:0]
IPApplication[4:0]
IPFeatureSet[4:0]
IPEvolution[4:0]
DesignMaturityLevel[4:0]
DesignDeliveryID[4:0]
(TYPO)

Correct)
IPFamily[3:0]
IPConfiguration[3:0]
IPApplication[3:0]
IPFeatureSet[3:0]
IPEvolution[3:0]
DesignMaturityLevel[3:0]
DesignDeliveryID[3:0]</td></tr>
</table>

<table>
<tr><th>Page</th><th>Section</th><th>Change Results</th></tr>
<tr><td colspan="3">Rev.*C</td></tr>
<tr><td>39</td><td>CHAPTER 3:Product Description Table 2 1 Features Description A/D Converter</td><td>Revised the below:
Error)
(None)
12bit resolution, 2 unit
64 annels of analog input for TEQFP208
48 annels of analog input for TEQFP176
35 annel of analog input for TEQFP144
24 annels of them are shared with the SMC for TEQFP208/176/144
External trigger and timer trigger are available.
The description of the A/D converter function should be referred in the S6J3300 Hardware Manual. Though the apter of I/O port in Traveo™ Platform Hardware Manual describes another A/D converter function, do not refer it.
Correct)
12bit resolution, 2 unit(Unit0 is possible to select annels 0-31. Unit1 is possible to select annels 32-63.)
64 annels of analog input for TEQFP208
48 annels of analog input for TEQFP176
35 annel of analog input for TEQFP144
24 annels of them are shared with the SMC for TEQFP208/176/144
External trigger and timer trigger are available.
The description of the A/D converter function should be referred in the S6J3300 Hardware Manual. Though the apter of I/O port in Traveo™ Platform Hardware Manual describes another A/D converter function, do not refer it.
A/D annel Control Register (ADC12Bn_CTRL0)[bit5:0] ANIN[5:0] : Analog Input Selection bits
This register setting is possible of annel 0-31 (the register value is 00_0000 to 01_1111).</td></tr>
<tr><td>93
94</td><td>CHAPTER 10:Port Description Table 2 1 Features Description A/D Converter</td><td>Revised the below:
Error)
ADC Analog [0 to 63] input pin

Correct)
ADC Unit0 [ch.0 to ch.31] inpit pin
ADC Unit1 [ch.32 to ch.63] input pin</td></tr>
<tr><td>231
232</td><td>CHAPTER 12:State Transition 3.Fetching the Operation Mode</td><td>Overall review.
Reviewing the consistency.</td></tr>
<tr><td>243</td><td>CHAPTER 13:Low Voltage Detection 4.Registers</td><td>Revised the below:
Error)
[Bit26:25] LVDL1V Internal low-voltage detection voltage setting bits
<table><tr><th>Bit 26:25</th><th>Voltage [V]</th></tr><tr><td>00</td><td>0.875(Initial value)</td></tr><tr><td>01</td><td>0.95</td></tr><tr><td>10</td><td>1.00</td></tr><tr><td>11</td><td>1.05</td></tr></table>
Correct)
[Bit26:25] LVDL1V Internal low-voltage detection voltage setting bits
<table><tr><th>Bit 26:25</th><th>Voltage [V]</th></tr><tr><td>00 *</td><td>0.875(Initial value)</td></tr><tr><td>01 *</td><td>0.95</td></tr><tr><td>10 *</td><td>1.00</td></tr><tr><td>11 *</td><td>1.05</td></tr></table>*: These detection voltage level settings are below the minimum operation voltage. Between these detection voltage and the minimum operation voltage, MCU functions are not guaranteed except for thelow voltage detector.</td></tr>
</table>

<table>
<tr><th>Page</th><th>Section</th><th>Change Results</th></tr>
<tr><td>244</td><td>CHAPTER 13:Low Voltage Detection 4.Registers</td><td>Revised the below:
Error)
[Bit18:17] LVDL2V Expanded internal low-voltage detection voltage setting bits
<table>
<tr><th>Bit 18:17</th><th>Voltage [V]</th></tr>
<tr><td>00</td><td>0.875</td></tr>
<tr><td>01</td><td>0.95</td></tr>
<tr><td>10</td><td>0.925</td></tr>
<tr><td>11</td><td>0.8125(Initial value)</td></tr>
</table>
Correct)
[Bit18:17] LVDL2V Expanded internal low-voltage detection voltage setting bits
<table>
<tr><th>Bit 18:17</th><th>Voltage [V]</th></tr>
<tr><td>00 *</td><td>0.875</td></tr>
<tr><td>01 *</td><td>0.95</td></tr>
<tr><td>10 *</td><td>0.925</td></tr>
<tr><td>11 *</td><td>0.8125(Initial value)</td></tr>
</table>
*: These detection voltage level settings are below the minimum operation voltage. Between these detection voltage and the minimum operation voltage, MCU functions are not guaranteed except for thelow voltage detector.</td></tr>
<tr><td>244</td><td>CHAPTER 13:Low Voltage Detection 4.Registers</td><td>Revised the below:
Error)
[Bit11:9] LVDH1V External low-voltage detection voltage setting bits
<table>
<tr><th>Bit 11:9</th><th>Voltage [V]</th></tr>
<tr><td>000</td><td>2.7</td></tr>
<tr><td>001</td><td>2.8</td></tr>
<tr><td>010</td><td>3.6</td></tr>
<tr><td>011</td><td>3.8</td></tr>
<tr><td>100</td><td>4.0(initial value for ip ID : 0x10122xxx product)</td></tr>
<tr><td>101</td><td>4.2</td></tr>
<tr><td>110</td><td>2.5</td></tr>
<tr><td>111</td><td>2.6(initial value for ip ID: 0x10128xxx product and 0x10120xxx product)</td></tr>
</table>
Correct)
[Bit11:9] LVDH1V External low-voltage detection voltage setting bits
<table>
<tr><th>Bit 11:9</th><th>Voltage [V]</th></tr>
<tr><td>000 *</td><td>2.7</td></tr>
<tr><td>001</td><td>2.8</td></tr>
<tr><td>010</td><td>3.6</td></tr>
<tr><td>011</td><td>3.8</td></tr>
<tr><td>100</td><td>4.0(initial value for ip ID : 0x10122xxx product)</td></tr>
<tr><td>101</td><td>4.2</td></tr>
<tr><td>110 *</td><td>2.5</td></tr>
<tr><td>111 *</td><td>2.6(initial value for ip ID: 0x10128xxx product and 0x10120xxx product)</td></tr>
</table>
*: These detection voltage level settings are below the minimum operation voltage. Between these detection voltage and the minimum operation voltage, MCU functions are not guaranteed except for thelow voltage detector.</td></tr>
</table>

<table>
<tr><th>Page</th><th>Section</th><th>Change Results</th></tr>
<tr><td>245</td><td>CHAPTER 13:Low Voltage Detection 4.Registers</td><td>Revised the below:
Error)
[Bit3:1] LVDH2V Expanded external low-voltage detection voltage setting bits
<table>
<tr><th>Bit 3:1</th><th>Voltage [V]</th></tr>
<tr><td>000</td><td>2.7</td></tr>
<tr><td>001</td><td>2.8</td></tr>
<tr><td>010</td><td>3.6</td></tr>
<tr><td>011</td><td>3.8</td></tr>
<tr><td>100</td><td>4.0</td></tr>
<tr><td>101</td><td>4.2</td></tr>
<tr><td>110</td><td>2.5</td></tr>
<tr><td>111</td><td>2.6(Initial value)</td></tr>
</table>
Correct)
[Bit3:1] LVDH2V Expanded external low-voltage detection voltage setting bits
<table>
<tr><th>Bit 3:1</th><th>Voltage [V]</th></tr>
<tr><td>000 *</td><td>2.7</td></tr>
<tr><td>001</td><td>2.8</td></tr>
<tr><td>010</td><td>3.6</td></tr>
<tr><td>011</td><td>3.8</td></tr>
<tr><td>100</td><td>4.0</td></tr>
<tr><td>101</td><td>4.2</td></tr>
<tr><td>110 *</td><td>2.5</td></tr>
<tr><td>111 *</td><td>2.6(Initial value)</td></tr>
</table>
*: These detection voltage level settings are below the minimum operation voltage. Between these detection voltage and the minimum operation voltage, MCU functions are not guaranteed except for thelow voltage detector.</td></tr>
<tr><td>518</td><td>CHAPTER 20:Sound Mixer 3.1.1. Mixing Start</td><td>Revised the below:
Error)
Note:
− The number of transfers of mixed sound sources from the sound mixer must match the number of output destination transfer requests. When the number of transfers on the sound mixer side is greater, the output destination will generate a DMAI error.

Correct)
Note:
− The number of transfers of mixed sound sources from the sound mixer must match the number of output destination transfer requests. When the number of transfers on the sound mixer side is greater, the output destination will generate a DMA transfer error.</td></tr>
<tr><td>523</td><td>CHAPTER 20:Sound Mixer 3.5. Data Request Control</td><td>Revised the below:
Error)
Mixer macros generate two types of data requests for WFT (WFGDATAREQ[4:0]) and for DMAC (MX_DMA_REQ[9:0]).
(TYPO)

Correct)
Mixer macros generate two types of data requests for WFT (WFGDATAREQ[4:0]) and for DMAC (MX_DMA_REQ[4:0]).</td></tr>
<tr><td>533</td><td>CHAPTER 20:Sound Mixer 4. Registers</td><td>Revised the below:
4.3. Mixer Date Request Control Register (MXDRQCTRL)
[bit3:0] FESTCH0: Data transfer to PMIS0 request assert threshold setting
Error)
Note:
(None)

Correct)
Note:
− FESTCHn bits must not be changed when DMAENCHn bit is 1.</td></tr>
</table>

<table>
<tr><th>Page</th><th>Section</th><th>Change Results</th></tr>
<tr><td>872</td><td>CHAPTER 24:Inter IC Sound (I2S) 1. Overview</td><td>Revised the below:
Error)
(None)

Correct)
Also refer to the chapter of "Sound System Configuration" on this manual how to configure the I2S which is connected to the Sound Mixer.</td></tr>
<tr><td>873</td><td>CHAPTER 24:Inter IC Sound (I2S) 2. Configuration and Block Diagram</td><td>Revised the below:

Error
This section describes a block diagram of I2S.
(TYPO)

Correct
This section describes the block diagram of I2S.</td></tr>
<tr><td>890</td><td>CHAPTER 24:Inter IC Sound (I2S) 4. Registers</td><td>Revised the below:
4.1. Reception FIFO Data Register (I2Sn_RXFDAT0 to 15)
Error)
<table><tr><td>REGISTER_NAME</td><td>YOURPERIn_CSRi</td></tr><tr><td>OFFSET</td><td>0 + i*4</td></tr><tr><td>ACCESS_SIZE</td><td>B H W</td></tr><tr><td>MULTIPLE</td><td>0:15</td></tr><tr><td>NUMERIC_TYPE</td><td></td></tr><tr><td>OTHER</td><td></td></tr></table>
Correct)
<table><tr><td>REGISTER_NAME</td><td>I2Sn_RXFDATi</td></tr><tr><td>OFFSET</td><td>0x0 + i*0x4</td></tr><tr><td>ACCESS_SIZE</td><td>B H W</td></tr><tr><td>MULTIPLE</td><td>0:15</td></tr><tr><td>NUMERIC_TYPE</td><td></td></tr><tr><td>OTHER</td><td></td></tr></table></td></tr>
<tr><td>892</td><td>CHAPTER 24:Inter IC Sound (I2S) 4. Registers</td><td>Revised the below:
4.2. Transmission FIFO Data Register (I2Sn_TXFDAT0 to 15)
Error)
<table><tr><td>REGISTER_NAME</td><td>YOURPERIn_CSRi</td></tr><tr><td>OFFSET</td><td>0 + i*4</td></tr><tr><td>ACCESS_SIZE</td><td>B H W</td></tr><tr><td>MULTIPLE</td><td>0:15</td></tr><tr><td>NUMERIC_TYPE</td><td></td></tr><tr><td>OTHER</td><td></td></tr></table>
Correct)
<table><tr><td>REGISTER_NAME</td><td>I2Sn_TXFDATi</td></tr><tr><td>OFFSET</td><td>0x0 + i*0x4</td></tr><tr><td>ACCESS_SIZE</td><td>B H W</td></tr><tr><td>MULTIPLE</td><td>0:15</td></tr><tr><td>NUMERIC_TYPE</td><td></td></tr><tr><td>OTHER</td><td></td></tr></table></td></tr>
</table>

<table>
<tr><th>Page</th><th>Section</th><th>Change Results</th></tr>
<tr><td>893</td><td>CHAPTER 24:Inter IC Sound (I2S) 4. Registers</td><td>Revised the below:
4.3. Control Register (I2Sn_CNTREG)
Error)
<table>
<tr><td>REGISTER_NAME</td><td>YOURPERIn_CSRi</td></tr>
<tr><td>OFFSET</td><td>0 + i*4</td></tr>
<tr><td>ACCESS_SIZE</td><td>B H W</td></tr>
<tr><td>MULTIPLE</td><td>0:15</td></tr>
<tr><td>NUMERIC_TYPE</td><td></td></tr>
<tr><td>OTHER</td><td></td></tr>
</table>
Correct)
<table>
<tr><td>REGISTER_NAME</td><td>I2Sn_CNTREG</td></tr>
<tr><td>OFFSET</td><td>0x80</td></tr>
<tr><td>ACCESS_SIZE</td><td>B H W</td></tr>
<tr><td>MULTIPLE</td><td></td></tr>
<tr><td>NUMERIC_TYPE</td><td></td></tr>
<tr><td>OTHER</td><td></td></tr>
</table></td></tr>
<tr><td>897</td><td>CHAPTER 24:Inter IC Sound (I2S) 4. Registers</td><td>Revised the below:
4.4. Channel Control Register 0 (I2Sn_MCR0REG)
Error)
<table>
<tr><td>REGISTER_NAME</td><td>YOURPERIn_CSRi</td></tr>
<tr><td>OFFSET</td><td>0 + i*4</td></tr>
<tr><td>ACCESS_SIZE</td><td>B H W</td></tr>
<tr><td>MULTIPLE</td><td>0:15</td></tr>
<tr><td>NUMERIC_TYPE</td><td></td></tr>
<tr><td>OTHER</td><td></td></tr>
</table>
Correct)
<table>
<tr><td>REGISTER_NAME</td><td>I2Sn_MCR0REG</td></tr>
<tr><td>OFFSET</td><td>0x84</td></tr>
<tr><td>ACCESS_SIZE</td><td>B H W</td></tr>
<tr><td>MULTIPLE</td><td></td></tr>
<tr><td>NUMERIC_TYPE</td><td></td></tr>
<tr><td>OTHER</td><td></td></tr>
</table></td></tr>
<tr><td>901</td><td>CHAPTER 24:Inter IC Sound (I2S) 4. Registers</td><td>Revised the below:
4.5. Channel Control Register 1 (I2Sn_MCR1REG)
Error)
<table>
<tr><td>REGISTER_NAME</td><td>YOURPERIn_CSRi</td></tr>
<tr><td>OFFSET</td><td>0 + i*4</td></tr>
<tr><td>ACCESS_SIZE</td><td>B H W</td></tr>
<tr><td>MULTIPLE</td><td>0:15</td></tr>
<tr><td>NUMERIC_TYPE</td><td></td></tr>
<tr><td>OTHER</td><td></td></tr>
</table>
Correct)
<table>
<tr><td>REGISTER_NAME</td><td>I2Sn_MCR1REG</td></tr>
<tr><td>OFFSET</td><td>0x88</td></tr>
<tr><td>ACCESS_SIZE</td><td>B H W</td></tr>
<tr><td>MULTIPLE</td><td></td></tr>
<tr><td>NUMERIC_TYPE</td><td></td></tr>
<tr><td>OTHER</td><td></td></tr>
</table></td></tr>
</table>

<table>
<tr><th>Page</th><th>Section</th><th>Change Results</th></tr>
<tr><td>902</td><td>CHAPTER 24:Inter IC Sound (I2S) 4. Registers</td><td>Revised the below:
4.6. Channel Control Register 2 (I2Sn_MCR2REG)
Error)
<table><tr><td>REGISTER_NAME</td><td>YOURPERIn_CSRi</td></tr><tr><td>OFFSET</td><td>0 + i*4</td></tr><tr><td>ACCESS_SIZE</td><td>B H W</td></tr><tr><td>MULTIPLE</td><td>0:15</td></tr><tr><td>NUMERIC_TYPE</td><td></td></tr><tr><td>OTHER</td><td></td></tr></table>
Correct)
<table><tr><td>REGISTER_NAME</td><td>I2Sn_MCR2REG</td></tr><tr><td>OFFSET</td><td>0x8C</td></tr><tr><td>ACCESS_SIZE</td><td>B H W</td></tr><tr><td>MULTIPLE</td><td></td></tr><tr><td>NUMERIC_TYPE</td><td></td></tr><tr><td>OTHER</td><td></td></tr></table></td></tr>
<tr><td>903</td><td>CHAPTER 24:Inter IC Sound (I2S) 4. Registers</td><td>Revised the below:
4.7. Operation Control Register (I2Sn_OPRREG)
Error)
<table><tr><td>REGISTER_NAME</td><td>YOURPERIn_CSRi</td></tr><tr><td>OFFSET</td><td>0 + i*4</td></tr><tr><td>ACCESS_SIZE</td><td>B H W</td></tr><tr><td>MULTIPLE</td><td>0:15</td></tr><tr><td>NUMERIC_TYPE</td><td></td></tr><tr><td>OTHER</td><td></td></tr></table>
Correct)
<table><tr><td>REGISTER_NAME</td><td>I2Sn_OPRREG</td></tr><tr><td>OFFSET</td><td>0x90</td></tr><tr><td>ACCESS_SIZE</td><td>B H W</td></tr><tr><td>MULTIPLE</td><td></td></tr><tr><td>NUMERIC_TYPE</td><td></td></tr><tr><td>OTHER</td><td></td></tr></table></td></tr>
<tr><td>905</td><td>CHAPTER 24:Inter IC Sound (I2S) 4. Registers</td><td>Revised the below:
4.8. Software Reset Register (I2Sn_SRST)
Error)
<table><tr><td>REGISTER_NAME</td><td>YOURPERIn_CSRi</td></tr><tr><td>OFFSET</td><td>0 + i*4</td></tr><tr><td>ACCESS_SIZE</td><td>B H W</td></tr><tr><td>MULTIPLE</td><td>0:15</td></tr><tr><td>NUMERIC_TYPE</td><td></td></tr><tr><td>OTHER</td><td></td></tr></table>
Correct)
<table><tr><td>REGISTER_NAME</td><td>I2Sn_SRST</td></tr><tr><td>OFFSET</td><td>0x94</td></tr><tr><td>ACCESS_SIZE</td><td>B H W</td></tr><tr><td>MULTIPLE</td><td></td></tr><tr><td>NUMERIC_TYPE</td><td></td></tr><tr><td>OTHER</td><td></td></tr></table></td></tr>
</table>

<table>
<tr><th>Page</th><th>Section</th><th>Change Results</th></tr>
<tr><td>906</td><td>CHAPTER 24:Inter IC Sound (I2S) 4. Registers</td><td>Revised the below:
4.9. Interrupt Control Register (I2Sn_INTCNT)
Error)
<table><tr><td>REGISTER_NAME</td><td>YOURPERIn_CSRi</td></tr><tr><td>OFFSET</td><td>0 + i*4</td></tr><tr><td>ACCESS_SIZE</td><td>B H W</td></tr><tr><td>MULTIPLE</td><td>0:15</td></tr><tr><td>NUMERIC_TYPE</td><td></td></tr><tr><td>OTHER</td><td></td></tr></table>
Correct)
<table><tr><td>REGISTER_NAME</td><td>I2Sn_INTCNT</td></tr><tr><td>OFFSET</td><td>0x98</td></tr><tr><td>ACCESS_SIZE</td><td>B H W</td></tr><tr><td>MULTIPLE</td><td></td></tr><tr><td>NUMERIC_TYPE</td><td></td></tr><tr><td>OTHER</td><td></td></tr></table></td></tr>
<tr><td>909</td><td>CHAPTER 24:Inter IC Sound (I2S) 4. Registers</td><td>Revised the below:
4.9. Interrupt Control Register (I2Sn_INTCNT)
[bit11:8] TFTH : Tx FIFO Threshold
Error)
Note:
(None)

Correct)
Note:
− These bits must not be changed during DMA transfers</td></tr>
<tr><td>909</td><td>CHAPTER 24:Inter IC Sound (I2S) 4. Registers</td><td>Revised the below:
4.9. Interrupt Control Register (I2Sn_INTCNT)
[bit3:0] RFTH : Rx FIFO Threshold
Error)
Note:
(None)

Correct)
Note:
− These bits must not be changed during DMA transfers.</td></tr>
</table>

<table>
<tr><th>Page</th><th>Section</th><th>Change Results</th></tr>
<tr><td>910</td><td>CHAPTER 24:Inter IC Sound (I2S) 4. Registers</td><td>Revised the below:
4.10. Status Register (I2Sn_STATUS)
Error)
<table><tr><td>REGISTER_NAME</td><td>YOURPERIn_CSRi</td></tr><tr><td>OFFSET</td><td>0 + i*4</td></tr><tr><td>ACCESS_SIZE</td><td>B H W</td></tr><tr><td>MULTIPLE</td><td>0:15</td></tr><tr><td>NUMERIC_TYPE</td><td></td></tr><tr><td>OTHER</td><td></td></tr></table>
Correct)
<table><tr><td>REGISTER_NAME</td><td>I2Sn_STATUS</td></tr><tr><td>OFFSET</td><td>0x9C</td></tr><tr><td>ACCESS_SIZE</td><td>B H W</td></tr><tr><td>MULTIPLE</td><td></td></tr><tr><td>NUMERIC_TYPE</td><td></td></tr><tr><td>OTHER</td><td></td></tr></table></td></tr>
<tr><td>911</td><td>CHAPTER 24:Inter IC Sound (I2S) 4. Registers</td><td>Revised the below:
4.10. Status Register (I2Sn_STATUS)
[bit29] FERR : Frame Error
Error)
− Frame synchronous signal can not be received with the set frame rate in the free-running mode (TYPO)

Correct)
− Frame synchronous signal cannot be received with the set frame rate in the free-running mode</td></tr>
<tr><td>914</td><td>CHAPTER 24:Inter IC Sound (I2S) 4. Registers</td><td>Revised the below:
4.11. DMA Activate Register (I2Sn_DMAACT)
Error)
<table><tr><td>REGISTER_NAME</td><td>YOURPERIn_CSRi</td></tr><tr><td>OFFSET</td><td>0 + i*4</td></tr><tr><td>ACCESS_SIZE</td><td>B H W</td></tr><tr><td>MULTIPLE</td><td>0:15</td></tr><tr><td>NUMERIC_TYPE</td><td></td></tr><tr><td>OTHER</td><td></td></tr></table>
Correct)
<table><tr><td>REGISTER_NAME</td><td>I2Sn_DMAACT</td></tr><tr><td>OFFSET</td><td>0xA0</td></tr><tr><td>ACCESS_SIZE</td><td>B H W</td></tr><tr><td>MULTIPLE</td><td></td></tr><tr><td>NUMERIC_TYPE</td><td></td></tr><tr><td>OTHER</td><td></td></tr></table></td></tr>
</table>

<table>
<tr><th>Page</th><th>Section</th><th>Change Results</th></tr>
<tr><td>916</td><td>CHAPTER 24:Inter IC Sound (I2S) 4. Registers</td><td>Revised the below:
4.12. Debug Register (I2Sn_DEBUG)
Error)
<table><tr><td>REGISTER_NAME</td><td>YOURPERIn_CSRi</td></tr><tr><td>OFFSET</td><td>0 + i*4</td></tr><tr><td>ACCESS_SIZE</td><td>B H W</td></tr><tr><td>MULTIPLE</td><td>0:15</td></tr><tr><td>NUMERIC_TYPE</td><td></td></tr><tr><td>OTHER</td><td></td></tr></table>
Correct)
<table><tr><td>REGISTER_NAME</td><td>I2Sn_DEBUG</td></tr><tr><td>OFFSET</td><td>0xA4</td></tr><tr><td>ACCESS_SIZE</td><td>B H W</td></tr><tr><td>MULTIPLE</td><td></td></tr><tr><td>NUMERIC_TYPE</td><td></td></tr><tr><td>OTHER</td><td></td></tr></table></td></tr>
<tr><td>917</td><td>CHAPTER 24:Inter IC Sound (I2S) 4. Registers</td><td>Revised the below:
4.13. Module ID Register (I2Sn_MIDREG)
Error)
<table><tr><td>REGISTER_NAME</td><td>YOURPERIn_CSRi</td></tr><tr><td>OFFSET</td><td>0 + i*4</td></tr><tr><td>ACCESS_SIZE</td><td>B H W</td></tr><tr><td>MULTIPLE</td><td>0:15</td></tr><tr><td>NUMERIC_TYPE</td><td></td></tr><tr><td>OTHER</td><td></td></tr></table>
Correct)
<table><tr><td>REGISTER_NAME</td><td>I2Sn_MIDREG</td></tr><tr><td>OFFSET</td><td>0xA8</td></tr><tr><td>ACCESS_SIZE</td><td>B H W</td></tr><tr><td>MULTIPLE</td><td></td></tr><tr><td>NUMERIC_TYPE</td><td></td></tr><tr><td>OTHER</td><td></td></tr></table></td></tr>
<tr><td>941</td><td>CHAPTER 26:PCMPWM</td><td>Revised the below:
Error)
This chapter explains the function and operation of the PCMPWM Module
(TYPO)

Correct
This chapter explains the function and operation of the PCMPWM module</td></tr>
<tr><td>942</td><td>CHAPTER 26:PCMPWM 1. Overview</td><td>Revised the below:
Error)
(None)

Correct)
Also refer to the chapter of "Sound System Configuration" on this manual how to configure the PCMPWM.</td></tr>
<tr><td>958</td><td>CHAPTER 26:PCMPWM 4. Registers</td><td>Revised the below:
Error)
Note:
− The suffix ‘i’ in the register name indicates that the register is an instance ‘i’ of the module..
(TYPO)

Correct)
Note:
− The suffix ‘i’ in the register name indicates that the register is in instance ‘i’ of the module..</td></tr>
</table>

Page	Section	Change Results
960	CHAPTER 26:PCMPWM 4. Registers	Revised the below: 4.1. PCMPWM Control Register (PCMPWMi_CONTROL) [bit20:16] FEST[4:0] : FIFO Empty Space Threshold Error) (None) Correct) Note: − These bits must not be changed during DMA transfers.
1152 1153 1155 1167 1172 1173 1175	CHAPTER 33: Graphics Subsystem	Revised the below: Error) HW Peripheral Manual for LCD Bus Interface (LCDBusIF) v1.0 Correct) HW Peripheral Manual for LCD Bus Interface (LCDBusIF) v1.11
Rev.*E		
3	Preface	Revised the below: Error) http://www.spansion.com/support/microcontrollers/ Correct) http://www.cypress.com/support/microcontrollers/
21	CHAPTER 1:Overview 2. Document Definition	Revised the below: Error) Under consideration Correct) 002-03898 002-04455 002-04446 002-09716 002-04452 002-04096 002-12061 002-02495
29	CHAPTER 2:Function List 1. Function List	Revised the below: CAN-FD RAM (ECC supported) Error) 16KB/ch It equivalents to 128 message buffer per channel of CCAN module Correct) 16KB/ch It equivalents to 128 message buffer per channel of MCAN module

<table>
<tr><th>Page</th><th>Section</th><th>Change Results</th></tr>
<tr><td>30</td><td>CHAPTER 2:Function List
2.1. Basic Option</td><td>Revise the below

Function : Digit = 5
Memory size :
Error)
<table>
<tr><th rowspan="2">Digit</th><th colspan="2">Flash</th><th colspan="2">RAM</th></tr>
<tr><th>Program</th><th>Work</th><th>Main</th><th>Backup</th></tr>
<tr><td>E</td><td>4,160KB</td><td>112KB</td><td>512KB</td><td>16+16KB</td></tr>
<tr><td>D</td><td>3,136KB</td><td>112KB</td><td>384KB</td><td>16+16KB</td></tr>
<tr><td>C</td><td>2,112KB</td><td>112KB</td><td>256KB</td><td>16+16KB</td></tr>
<tr><td>B</td><td>1,600KB</td><td>112KB</td><td>192KB</td><td>16+16KB</td></tr>
</table>
Correct)
<table>
<tr><th rowspan="2">Digit</th><th colspan="2">Flash</th><th colspan="2">RAM</th></tr>
<tr><th>Program</th><th>Work</th><th>Main</th><th>Backup</th></tr>
<tr><td>E</td><td>4,160KB</td><td>112KB</td><td>512KB</td><td>16+16KB</td></tr>
<tr><td>D</td><td>3,136KB</td><td>112KB</td><td>384KB</td><td>16+16KB</td></tr>
</table></td></tr>
<tr><td>31</td><td>CHAPTER 2:Function List
2.1. Basic Option</td><td>Revised the below:

Error)
(None)

Correct)
− In case of function digit 5, the main RAM size is configured as follows according to memory digit.
E : TCRAM 128KB + System-RAM 384KB
D : TCRAM 128KB + System-RAM 256KB</td></tr>
<tr><td>37</td><td>CHAPTER 3:Product Description
2. Product Description</td><td>Revised Table 2-1 Features Description

Power Domain (PD)
Error)
The product series supports the power off control of PD1, PD2 (including PD3 and 5), and PD6.

Correct)
The product series supports the power off control of PD1, PD2 (including PD3 and 5), PD4_0, PD4_1 and PD6.</td></tr>
<tr><td>38</td><td>CHAPTER 3:Product Description
2. Product Description</td><td>Revised Table 2-1 Features Description

Backup RAM
Error)
32kByte

Correct)
32kByte (Backup-RAM0 : 16KB, Backup-RAM1 : 16KB)</td></tr>
<tr><td>38</td><td>CHAPTER 3:Product Description
2. Product Description</td><td>Revised Table 2-1 Features Description

Embedded Program/Work Flash Memory
Error)
7-wait-cycle: 80MHz or less.
13-wait-cycle: 160MHz or less.

Correct)
6-wait-cycle: 80MHz or less.
12-wait-cycle: 160MHz or less.</td></tr>
</table>

<table>
<tr><th>Page</th><th>Section</th><th>Change Results</th></tr>
<tr><td>39</td><td>CHAPTER 3:Product Description
2. Product Description</td><td>Revised Table 3-1 Features Description

I2S
Error)
− I2S0 only supports the output of sound sources.
− I2S1 supports both the input and the output.
− I2S has its own PPU, but the function is fixed to disable.

Correct)
− I2S0 can output sound sources which are processed by Sound System.
− I2S1 can input sound sources which are processed by Sound System.
− I2S has its own PPU, but the function is fixed to disable.
See the "Sound System Configuration" of S6J3300 Hardware Manual in detail.</td></tr>
<tr><td>40</td><td>CHAPTER 3: Product Description
2. Product Description</td><td>Revised Table 3-1 Features Description:

Multi-Functional Serial (MFS)
Error)
Multi-Functional Serial (MFS)
Only 2 ports of MFS have the dedicated I/O for I2C.
See the datasheet in detail.

Correct)
Multi-functional Serial (MFS)
Only 2 ports of MFS have the dedicated I/O for I2C.
See the datasheet in detail.
The I2C is not designed to be hot swappable.</td></tr>
<tr><td>43</td><td>CHAPTER 3: Product Description
2.2. Reset Signal</td><td>Added the below:

Correct)
2.2. Reset Signal</td></tr>
<tr><td>44</td><td>CHAPTER 3: Product Description
3.2. Error Response</td><td>Added the below

Correct
3.2. Error Response</td></tr>
<tr><td>46</td><td>CHAPTER 3: Product Description
3.3. Restriction</td><td>Revised the below:

Error)
Power domain
SYSC0_PLLPDCFGR.PD5_xEN

Correct)
Power domain
SYSC0_PSSPDCFGR.PD5_xEN</td></tr>
<tr><td>46</td><td>CHAPTER 3: Product Description
3.3. Restriction</td><td>Added the below:

Correct)
<table>
<tr><th>Function</th><th>Related Register and Configuration</th><th>Restriction</th><th>Remark</th></tr>
<tr><td>Reset</td><td>-</td><td>All RAM data (include TCRAM, System RAM, Backup RAM) are not guaranteed after release of RSTX pin input reset.</td><td>This restriction is applied to the following product type.
- S6J33xx xxC</td></tr>
</table></td></tr>
</table>

<table>
<tr><th>Page</th><th>Section</th><th>Change Results</th></tr>
<tr><td>46</td><td>CHAPTER 3: Product Description
3.3. Restriction</td><td>Added the below:

Correct)
<table>
<tr><th>Function</th><th>Related Register and Configuration</th><th>Restriction</th><th>Remark</th></tr>
<tr><td>Security</td><td>TCFCFG0_CSWP0.SWP0 to SWP31 bit
TCFCFG0_CSWP1to8.SWP0 to SWP31 bit</td><td>In order to write successfully any sector of Code Flash, all Sectors must have their SWP (TCFCFG0_CSWP0.SWP0 to SWP31 , TCFCFG0_CSWP1.SWP0 to SWP31) set to 1</td><td>This restriction is applied to the following product type.
- S6J33xxxxC</td></tr>
</table></td></tr>
<tr><td>48</td><td>CHAPTER 4:Block Diagram
1. Block Diagram</td><td>Revised Figure 4-1 Block Diagram

Error)
SCU
Clock / Power /
Mode / State /
Timer /
LVDET / Wake-
Up / CSV / RTC /
HW-WDT / ext
IRQ

Correct)
SYSC0
Clock / Power /
Mode / State /
Timer /
LVDET / Wake-
Up / CSV / RTC /
HW-WDT / ext
IRQ</td></tr>
<tr><td>53</td><td>CHAPTER 5: Clock Configuration
2.Operation</td><td>Revised Table 2-1 Source Clock of Each Function

Error)
CLK_SWWDT
Internal CR oscillator
(High frequency)
Internal CR oscillator
(Low frequency)

Correct)
CLK_SWWDT
Internal CR oscillator
(High frequency) or
Internal CR oscillator
(Low frequency) or
Main clock or
Sub clock</td></tr>
</table>

Page	Section	Change Results
53	CHAPTER 5: Clock Configuration 2.Operation	Revised the below: Added) Table5-1 Function : I2S1 Clock Name : CLK_HAPP1B0 / CLK_CD4 Source Clock System : SSCG0 / PLL2 Configuration : See Traveo Platform Hardware Manual / See Traveo Platform Hardware Manual It is described as "ECLK" in the chapter of I2S. Error) Table2-1 Function : I2S0 Clock Name : CLK_CD4 Source Clock System : PLL2 Configuration : See Traveo Platform Hardware Manual Function : PCM-PWM Clock Name : CLK_CD4 Source Clock System : PLL2 Configuration : See Traveo Platform Hardware Manual Correct) Table5-1 Function : I2S0 Clock Name : CLK_CD4 / CLK_CD5B0 Source Clock System : PLL2 / PLL3 Configuration : See Traveo Platform Hardware Manual It is described as "ECLK" in the chapter of I2S. / See Traveo Platform Hardware Manual Function : PCM-PWM Clock Name : CLK_CD4 / CLK_CD5B0 Source Clock System : PLL2 / PLL3 Configuration : See Traveo Platform Hardware Manual It is described as "PWM_CLK" in the chapter of PCMPWM. / See Traveo Platform Hardware Manual
54	CHAPTER 5: Clock Configuration 2.Operation	Revised the below: Error) Table2-2 Source Clock System : PLL2 Clock Name : INDICATINDICAT Function : Stereo Audio DAC Local Clock Name : CLKDACI Correct) Table5-2 Source Clock System : PLL2 Clock Name : CLK_CD4 Function : Stereo Audio DAC Local Clock Name : CLKDACI

Page	Section	Change Results
54	CHAPTER 5: Clock Configuration 2.Operation	Revised the below: Added) Table5-2 Source Clock System : SSCG0 Clock Name : CLK_HAPP1B0 Function : Inter IC Sound(I2S1) Local Clock Name : internal clock Error) Table2-2 Source Clock System : External input clock Clock Name : External input clock Function : Inter IC Sound(I2S) Local Clock Name : SCK/ECLK Source Clock System : PLL2 Clock Name : CLK_CD4 Function : Inter IC Sound(I2S) Local Clock Name : ECLK Source Clock System : PLL3 Clock Name : CLK_CD5B0 Function : Inter IC Sound(I2S) Local Clock Name : No Define Correct) Table5-2 Source Clock System : External input clock Clock Name : External input clock Function : Inter IC Sound(I2S0/I2S1) Local Clock Name : SCK/ECLK Source Clock System : PLL2 Clock Name : CLK_CD4 Function : Inter IC Sound(I2S0/I2S1) Local Clock Name : ECLK Source Clock System : PLL3 Clock Name : CLK_CD5B0 Function : Inter IC Sound(I2S0) Local Clock Name : internal clock
54	CHAPTER 5: Clock Configuration 2.Operation	Revised the below: Error) The frequency of CLK_CPU should be 232MHz or less if its source clock is SSCG. Correct) The frequency of CLK_CPU should be 240MHz or less.
64	CHAPTER 7:Memory and Base Address Map 1. Overview	Revised the below: Deleted) − Backup RAM area (BACKUP_RAM) [0E80_0000 to 0E80_7FFF]
66	CHAPTER 7:Memory and Base Address Map 2. Memory Map	Added the below table: Added) Memory map of BACKUP_RAM START Address END Address Part 0E80_0000 0E80_3FFF BACKUP_RAM in PD4_0 0E80_4000 0E80_7FFF BACKUP_RAM in PD4_1

<table>
<tr><th>Page</th><th>Section</th><th>Change Results</th></tr>
<tr><td>69</td><td>CHAPTER 7: Memory and Base Address Map
2. Base Address Map</td><td>Revised Base Address Map

Error)
START Address : B475_0000
END Address : B475_7FFF
Group Function : Common PERI #2 PPU
PPU_No : 77

Correct)
START Address : B475_0000
END Address : B475_7FFF
Group Function : Common PERI #2 PPU
PPU_No : -</td></tr>
<tr><td>86
87
88
89
90
91</td><td>CHAPTER 9:DMA Channel Activation Factors
1. Factors List</td><td>The shading parts added as below.

Error)
<table>
<tr><th>Number of Channels</th><th>Peripheral Functions</th></tr>
<tr><td>0 to 8</td><td>Reserved</td></tr>
<tr><td>9</td><td>WORK FLASH</td></tr>
<tr><td>:</td><td>:</td></tr>
<tr><td>396</td><td>ARH0 Remote Handler DMA request 15</td></tr>
<tr><td>397 to 511</td><td>Reserved</td></tr>
</table>
Correct)
<table>
<tr><th>client number</th><th>Peripheral Functions</th><th>Enable Setting of IRQ request</th><th>Remarks</th></tr>
<tr><td>0 to 8</td><td>Reserved</td><td>-</td><td></td></tr>
<tr><td>9</td><td>WORK FLASH</td><td>Unnecessary</td><td></td></tr>
<tr><td>:</td><td>:</td><td>:</td><td></td></tr>
<tr><td>396</td><td>ARH0 Remote Handler DMA request 15</td><td>Unnecessary</td><td></td></tr>
<tr><td>397 to 511</td><td>Reserved</td><td>-</td><td></td></tr>
</table>
*1: The interrupt factor flags of peripheral function are cleared automatically by acceptance of DMA transfer request.</td></tr>
<tr><td>86
89</td><td>CHAPTER 9:DMA Channel Activation Factors
1. Factors List</td><td>Revised the below

Error)
Number of Channels
48 : CAN_FD ch.5
49 : CAN_FD ch.6
50 : CAN_FD ch.7
321 : CAN_FD ch.0
322 : CAN_FD ch.1
323 : CAN_FD ch.2
324 : CAN_FD ch.3
325 : CAN_FD ch.4

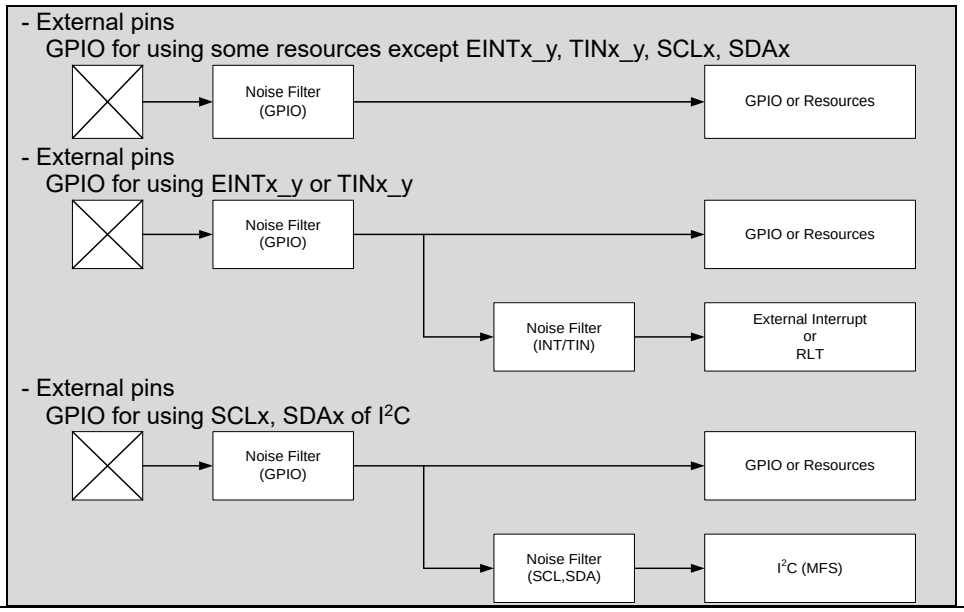
Correct)
Number of Channels
48 : Reserved
49 : Reserved
50 : Reserved
321 : Reserved
322 : Reserved
323 : Reserved
324 : Reserved
325 : Reserved</td></tr>
</table>

Page	Section	Change Results
92	CHAPTER 9:DMA Channel Activation Factors 2. Note	Added the 2. Note. The shading parts added as below. **2. Note** This product supports 397 DMA clients. − 'm' represents client number. m = 8, 9, …, 396 − 'M' represents number of client I/F. M = 397 This product does not support 'm = 397 to 511'. Accessing to those number of DMAi_CMICICm is responded with bus error. See the platform manual and chapter DMA controller in detail.
93	CHAPTER 10: Port Description 1. Port Description List 2. Remark	Revised the below: The information on this chapter moved to Datasheet. Error) 1. Port Description List 2. Remark Correct) *Delete of 1. Port Description List and 2. Remark Add comments: Please for details refer to the Device Datasheet. (The information on this chapter moved to Datasheet.)
97	CHAPTER 11: Port Configuration 2. Configuration and Block Diagram	Revised Figure 11-1 Block Diagram of Port Function Added) Added Pull-up/down control using POE
180	CHAPTER 11: Port Configuration 3.3. Analog I/O Setting	Revised the below: Error) − The PPC_PCFGR:PIE register of target port makes disable digital input to set to 0. − The corresponding POF should be set to 0. − The GPIO_DDR register of target port makes input control to set to 0. − The PPC_PCFGR:PDE/PUE register of target port makes disable Pull-Up/Pull-Down to set to 0. Correct) ADC − The PPC_PCFGR:PIE register of target port makes disable digital input to set to 0. − The corresponding POF. (typically set to 0) − The GPIO_DDR register of target port. (typically set to 0) − The PPC_PCFGR:PDE/PUE register of target port. (typically set 0) LCDC − The PPC_PCFGR:PIE register of target port makes disable digital input to set to 0. − The corresponding POF should be set to 0. − The GPIO_DDR register of target port makes input control to set to 0. − The PPC_PCFGR:PDE/PUE register of target port makes disable Pull-Up/Pull-Down to set to 0. Note: − Regarding the analog switch setting, see 'CHAPTER of 12/10/8-BIT Analog to Digital Converter' and 'CHAPTER of LCDC Controller'.
186	CHAPTER 11: Port Configuration 3.5. Output Drive Capacity Setting	Revised the below: Error) TEQFP-208 : Remaks 86 : blank 89 : blank Correct) TEQFP-208 : Remaks 86 : *1-1 89 : *1-1

<table>
<tr><th>Page</th><th>Section</th><th>Change Results</th></tr>
<tr><td>205</td><td>CHAPTER 11: Port Configuration
6.1. Noise Filter</td><td>
Added the 6. Precautions 6.1. Noise Filter.
The shading parts added as below.

6. Precautions
6.1. Noise Filter
Each GPIO has the noise filter for noise removal from external input. Also, some resources have additional dedicated noise filter for resource input. See the following Configuration and Diagram.

Table 11-3: Configuration for Noise Filter
<table>
<tr><th>Noise filter for resource</th><th>PPC_PCFGRiij: NFE</th><th>EICxx_NFER: NFEn</th><th>RLTn_TMCSR: NFE</th><th>RIC_RESINn: RESSEL</th></tr>
<tr><td>GPIO</td><td>Enable/Disable</td><td>- *1</td><td>- *1</td><td>- *1</td></tr>
<tr><td>External interrupt (EINT)</td><td>Disable *2</td><td>Enable/Disable</td><td>- *1</td><td>- *1</td></tr>
<tr><td>Reload Timer input (TIN)</td><td>Disable *2</td><td>- *1</td><td>Enable/Disable</td><td>- *1</td></tr>
<tr><td>I²C interface (SCL, SDA)</td><td>Disable *2</td><td>- *1</td><td>- *1</td><td>Enable/Disable</td></tr>
</table>
*1: It is unrelated to register's setting.
*2: When selecting a Noise Filter other than for GPIO, set the Port Setting Register (PPC_PCFGRiij: NFE) to Disable.

Figure 11-8: Diagram of Noise Filter


</td></tr>
<tr><td>210</td><td>CHAPTER 12:State Transition
2. Diagram of State Transition</td><td>
Revised Figure 12-2 RUN/PSS State definitions:

Error)
Fast-CR
PSS TIMER MODE : 4MHz / disable
PSS STOP MODE : 4MHz / disable
PSS TIMER MODE SHUTDOWN(PD6_ON) : Prohibition of an oscillation

Correct)
Fast-CR
PSS TIMER MODE : 4MHz
PSS STOP MODE : 4MHz
PSS TIMER MODE SHUTDOWN(PD6_ON) : 4MHz
</td></tr>
</table>

<table>
<tr><th>Page</th><th>Section</th><th>Change Results</th></tr>
<tr><td>210</td><td>CHAPTER 12:State Transition
2. Diagram of State Transition</td><td>Revised Figure 12-2 RUN/PSS State definitions:

Error)
PD4
PSS Timer mode shutdown (PD6_ON) : ON
PSS Timer mode shutdown (PD6_OFF) : ON
PSS STOP mode Shutdown : ON

Correct)
PD4_0/PD4_1
PSS Timer mode shutdown (PD6_ON) : ON or OFF
PSS Timer mode shutdown (PD6_OFF) : ON or OFF
PSS STOP mode Shutdown : ON or OFF</td></tr>
<tr><td>215</td><td>CHAPTER 12:State Transition
4. Changes to PSS and RUN</td><td>Revised Figure 12-5 Changes to PSS Timing Chart

Added)
PSC1 and VCC12</td></tr>
<tr><td>215</td><td>CHAPTER 12:State Transition
4. Changes to PSS and RUN</td><td>Revised Figure 12-5 Changes to PSS Timing Chart

Added)
*1. During power-off sequence, it is necessary to switch off VCC12 by driving PSC1 pin low by entering PSS mode (power domain 2 off). If VCC12 needs to be switched off by other means, RSTX needs to be asserted before switching off VCC12 to inactivate the operation of VCC12 supplied domain below the operation assurance range.</td></tr>
<tr><td>223</td><td>CHAPTER 13: Low Voltage Detection
4. Registers</td><td>Revised the below:
[Bit26:25] LVDL1V Internal low-voltage detection voltage setting bits

Error)
<table>
<tr><th>Bit 26:25</th><th>Voltage [V]</th></tr>
<tr><td>00 *</td><td>0.875(Initial value)</td></tr>
<tr><td>01 *</td><td>0.95</td></tr>
<tr><td>10 *</td><td>1.00</td></tr>
<tr><td>11 *</td><td>1.05</td></tr>
</table>
*: These detection voltage level settings are below the minimum operation voltage. Between these detection voltage and the minimum operation voltage, MCU functions are not guaranteed except for the low voltage detector.

Correct)
<table>
<tr><th>Bit 26:25</th><th>Voltage [V]</th><th>Guaranteed MCU operation voltage range</th></tr>
<tr><td>00 *</td><td>0.875(Initial value)</td><td rowspan="4">No</td></tr>
<tr><td>01 *</td><td>0.95</td></tr>
<tr><td>10 *</td><td>1.00</td></tr>
<tr><td>11 *</td><td>1.05</td></tr>
</table>
*: This LVD cannot be used to reliably generate a reset before voltage dips below minimum guaranteed MCU operation voltage, as these detection levels are below the minimum guaranteed MCU operation voltage.</td></tr>
</table>

<table>
<tr><th>Page</th><th>Section</th><th>Change Results</th></tr>
<tr><td>224</td><td>CHAPTER 13: Low Voltage Detection
4. Registers</td><td>Revised the below:
[Bit18:17] LVDL2V Expanded internal low-voltage detection voltage setting bits

Error)
<table>
<tr><th>Bit 18:17</th><th>Voltage [V]</th></tr>
<tr><td>00 *</td><td>0.875</td></tr>
<tr><td>01 *</td><td>0.95</td></tr>
<tr><td>10 *</td><td>0.925</td></tr>
<tr><td>11 *</td><td>0.8125(Initial value)</td></tr>
</table>
*: These detection voltage level settings are below the minimum operation voltage. Between these detection voltage and the minimum operation voltage, MCU functions are not guaranteed except for the low voltage detector.

Correct)
<table>
<tr><th>Bit 18:17</th><th>Voltage [V]</th><th>Guaranteed MCU operation voltage range</th></tr>
<tr><td>00 *</td><td>0.875</td><td rowspan="4">No</td></tr>
<tr><td>01 *</td><td>0.95</td></tr>
<tr><td>10 *</td><td>0.925</td></tr>
<tr><td>11 *</td><td>0.8125(Initial value)</td></tr>
</table>
*: This LVD cannot be used to reliably generate a reset before voltage dips below minimum guaranteed MCU operation voltage, as these detection levels are below the minimum guaranteed MCU operation voltage.</td></tr>
<tr><td>224</td><td>CHAPTER 13: Low Voltage Detection
4. Registers</td><td>Revised the below:
[Bit14] LVDH1S: LVDH1 operation selection bit

Error)
Note:
− LVDH1V will be initial value with reset. If LVDH1V is changed from initial vale, LVDH1S should be configured as interrupt.

Correct)
Note:
− LVDH1 Reset level can be changed by EXVRSTCNT bit in SYSC0_SPECFGR register.
If VCC12 needs to be switched off without entering PSS mode, please set LVDH1S bit to "0", and moreover set EXVRSTCNT bit to "0"</td></tr>
</table>

<table>
<tr><th>Page</th><th>Section</th><th>Change Results</th></tr>
<tr><td>224</td><td>CHAPTER 13: Low Voltage Detection
4. Registers</td><td>Revised the below:
[Bit11:9] LVDH1V External low-voltage detection voltage setting bits

Error)
<table>
<tr><th>Bit 11:9</th><th>Voltage [V]</th></tr>
<tr><td>000 *</td><td>2.7</td></tr>
<tr><td>001</td><td>2.8</td></tr>
<tr><td>010</td><td>3.6</td></tr>
<tr><td>011</td><td>3.8</td></tr>
<tr><td>100</td><td>4.0(initial value for Chip ID : 0x10122xxx product)</td></tr>
<tr><td>101</td><td>4.2</td></tr>
<tr><td>110 *</td><td>2.5</td></tr>
<tr><td>111 *</td><td>2.6(initial value for Chip ID: 0x10128xxx product and 0x10120xxx product)</td></tr>
</table>
*: These detection voltage level settings are below the minimum operation voltage. Between these detection voltage and the minimum operation voltage, MCU functions are not guaranteed except for the low voltage detector.

Correct)
<table>
<tr><th rowspan="2">Bit 11:9</th><th rowspan="2">Voltage [V]</th><th colspan="2">Guaranteed MCU operation voltage range</th></tr>
<tr><th>Chip ID 0x10122xxx product</th><th>Chip ID 0x10128xxx, 0x10120xxx product</th></tr>
<tr><td>000 *</td><td>2.7</td><td>No</td><td>No</td></tr>
<tr><td>001</td><td>2.8</td><td rowspan="5">Yes</td><td>Yes</td></tr>
<tr><td>010</td><td>3.6</td><td rowspan="4">Not supported</td></tr>
<tr><td>011</td><td>3.8</td></tr>
<tr><td>100</td><td>4.0(initial value for Chip ID : 0x10122xxx product)</td></tr>
<tr><td>101</td><td>4.2</td></tr>
<tr><td>110 *</td><td>2.5</td><td rowspan="2">No</td><td rowspan="2">No</td></tr>
<tr><td>111 *</td><td>2.6(initial value for Chip ID: 0x10128xxx product and 0x10120xxx product)</td></tr>
</table>
*: This LVD cannot be used to reliably generate a reset before voltage dips below minimum guaranteed MCU operation voltage, as these detection levels are below the minimum guaranteed MCU operation voltage.</td></tr>
</table>

<table>
<tr><th>Page</th><th>Section</th><th>Change Results</th></tr>
<tr><td>225</td><td>CHAPTER 13: Low Voltage Detection
4. Registers</td><td>Revised the below:
[Bit3:1] LVDH2V Expanded external low-voltage detection voltage setting bits

Error)
<table>
<tr><th>Bit 3:1</th><th>Voltage [V]</th></tr>
<tr><td>000 *</td><td>2.7</td></tr>
<tr><td>001</td><td>2.8</td></tr>
<tr><td>010</td><td>3.6</td></tr>
<tr><td>011</td><td>3.8</td></tr>
<tr><td>100</td><td>4.0</td></tr>
<tr><td>101</td><td>4.2</td></tr>
<tr><td>110 *</td><td>2.5</td></tr>
<tr><td>111 *</td><td>2.6(Initial value)</td></tr>
</table>
*: These detection voltage level settings are below the minimum operation voltage. Between these detection voltage and the minimum operation voltage, MCU functions are not guaranteed except for the low voltage detector.

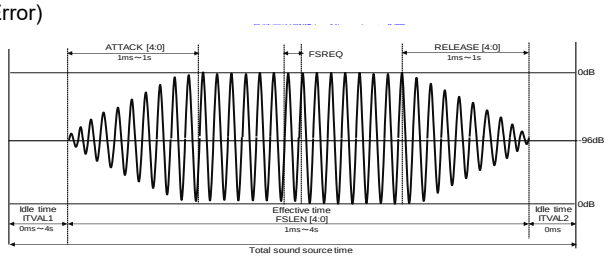
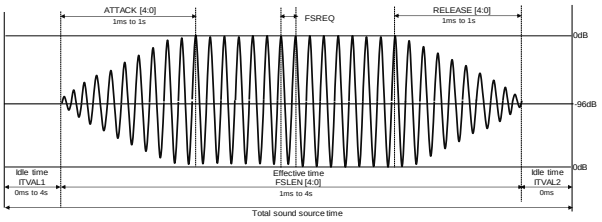
Correct)
<table>
<tr><th>Bit 3:1</th><th>Voltage [V]</th><th>Guaranteed MCU operation voltage range</th></tr>
<tr><td>000 *</td><td>2.7</td><td>No</td></tr>
<tr><td>001</td><td>2.8</td><td>Yes</td></tr>
<tr><td>010</td><td>3.6</td><td rowspan="5">Not support</td></tr>
<tr><td>011</td><td>3.8</td></tr>
<tr><td>100</td><td>4.0</td></tr>
<tr><td>101</td><td>4.2</td></tr>
<tr><td>110 *</td><td>2.5</td><td rowspan="2">No</td></tr>
<tr><td>111 *</td><td>2.6(Initial value)</td></tr>
</table>
*: This LVD cannot be used to reliably generate a reset before voltage dips below minimum guaranteed MCU operation voltage, as these detection levels are below the minimum guaranteed MCU operation voltage.</td></tr>
<tr><td>241</td><td>CHAPTER 15: 12-/10-/8-bit Analog to Digital Converter 3.1. A/D Conversion Flow</td><td>Revised the below:

Error)
Figure 23 A/D Conversion Flow
(4)Step
Conversion

Correct)
Figure 15-2 A/D Conversion Flow
(4)Step
Comparison</td></tr>
<tr><td>253</td><td>CHAPTER 15: 12-/10-/8-bit Analog to Digital Converter 3.5. Group Processing</td><td>Revised the below:

Error)
⋯ and trigger status of channel 0 is set.

Correct)
... and trigger status of channel 1 is set (channel 1 is the last converted channel set to resume).</td></tr>
</table>

Page	Section	Change Results
283	CHAPTER 15: 12-/10-/8-bit Analog to Digital Converter 5.4. Pulse Counter Control Registers (ADC12Bn_PCCTRL0 to 63)	Revised the below: Error) [bit20:16] PCTNRL[4:0] : Pulse Negative Counter Reload Register The negative counter is reloaded with the value from this register when "1" is written to the corresponding ADC12Bn_PCIRQC0.PCIRQC0 bit or on any positive event from the appropriate range comparator. [bit15:8] PCTPCT[7:0] : Pulse Positive Counter Register − Writing "1" to the corresponding ADC12Bn_PCIRQC0.PCIRQ bit. Therefore, in order to reload negative counter immediately after set to PCTPRL[7:0], the corresponding ADC12Bn_PCIRQC0 to 1.PCIRQC bit should be written "1". [bit7:0] PCTPRL[7:0] : Pulse Positive Counter Reload Register The positive counter is reloaded with the value from this register when "1" is written to the corresponding ADC12Bn_PCIRQC0.PCIRQC0 bit or on expiration of the corresponding negative counter (PCTNCT). Correct) [bit20:16] PCTNRL[4:0] : Pulse Negative Counter Reload Register The negative counter is reloaded with the value from this register when "1" is written to the corresponding ADC12Bn_PCIRQC0 to 1.PCIRQC bit or on any positive event from the appropriate range comparator. [bit15:8] PCTPCT[7:0] : Pulse Positive Counter Register − Writing "1" to the corresponding ADC12Bn_PCIRQC0 to 1 .PCIRQ bit. Therefore, in order to reload positive counter immediately after set to PCTPRL[7:0], the corresponding ADC12Bn_PCIRQC0 to 1.PCIRQC bit should be written "1". [bit7:0] PCTPRL[7:0] : Pulse Positive Counter Reload Register The positive counter is reloaded with the value from this register when "1" is written to the corresponding ADC12Bn_PCIRQC0 to 1.PCIRQC bit or on expiration of the corresponding negative counter (PCTNCT).
460	CHAPTER 19: Sound Waveform Generator 1. Overview	Added comments as below: Correct) Also refer to the chapter of "Sound System configuration" how to configure the SWFG.
462	P458 CHAPTER 19: Sound Waveform Generator 3. Operation of the Sound Waveform Generator	Revised the below: 3.1. Sound Source Specifications Error)  Correct) 

Page	Section	Change Results
463	CHAPTER 19: Sound Waveform Generator 3. Operation of the Sound Waveform Generator	Revised the below: 3.1. Sound Source Specifications 3.1.6. Fade Out Time (RELEASE) Error) Notes: Each configuration should be under condition below. Attack < FSLEN, Release < FSLEN, Attack + Release < FSELEN Correct) Notes: Each configuration should be under condition below. Attack ≦ FSLEN, Release ≦ FSLEN and Attack + Release ≦ FSLEN
464	CHAPTER 19: Sound Waveform Generator 3. Operation of the Sound Waveform Generator	Delete comments as below: 3.2. Interrupt Error) − SWFG AHB Master Interface access error This interrupt is generated when an error response is received for register access or another operation on the AHB Master Interface of SWFG. SWFG indicates the status of this error only. However, SWFG does not have functions enabling or clearing an interrupt generated by this error.
474 475 476	CHAPTER 19: Sound Waveform Generator 4. Registers	Added note as below: 4.3. Waveform Generator Channel Address1 Register (WGCHADD1) [bit28:16] CH1ADD: Channel 1 output destination address [bit12:0] CH0ADD: Channel 0 output destination address 4.4. Waveform Generator Channel Address2 Register (WGCHADD2) [bit28:16] CH3ADD: Channel 3 output destination address [bit12:0] CH2ADD: Channel 2 output destination address 4.5. Waveform Generator Channel Address3 Register (WGCHADD3) [bit12:0] CH4ADD: Channel 4 output destination address Correct) Notes: These bits can be set at the initialization. After that, it should not be changed.
478	CHAPTER 19: Sound Waveform Generator 4. Registers	Revised the below: 4.7. Waveform Generator Channel n Control Register1 (WGCHnCTRL1, n=0 to 4) Error) REGISTER_NAME Waveform Generator Channel1 Control Register (WGCH1CTRL1) Correct) REGISTER_NAME Waveform Generator Channel n Control Register (WGCHnCTRL1)
496	CHAPTER 20:Sound Mixer 1. Overview	Added comments as below: Correct) Also refer to the chapter of "Sound System configuration" how to configure the sound mixer.
498	CHAPTER 20:Sound Mixer 3. Operation of the Sound Mixer	Added notes as below: 3.1.1. Mixing Start Notes: Correct) − In the step 5, completion of the initialization is required before executing the next step. Completion can be confirmed by reading the applicable bit of the MXBUFFCLR register, because the bit is automatically cleared to 0 following initialization.

Page	Section	Change Results
498	CHAPTER 20:Sound Mixer 3. Operation of the Sound Mixer	Added notes as below: 3.1.2. Mixing Stop Correct) Notes: If the DMAC is used for the sound source transmission, DMAEN bit of MXDRQCTRL set to 0 after the transmission is completed.
503	CHAPTER 20:Sound Mixer 3. Operation of the Sound Mixer	Revised the below:: 3.5. Data Request Control Error) WFT Correct) WFG
503	CHAPTER 20:Sound Mixer 3. Operation of the Sound Mixer	Revised the below: 3.5. Data Request Control 2. Data request for DMAC Error) MX_DMA_REQ[0] is for PCM/I2S Channel 1, while MX_DMA_REQ[4] is for PCM/I2S Channel 5. Correct) MX_DMA_REQ[0] is for PMIS Channel 0, while MX_DMA_REQ[4] is for PMIS Channel 4.
503	CHAPTER 20:Sound Mixer 3. Operation of the Sound Mixer	Added note as below: 3.5. Data Request Control Correct) Notes: If the DMAC is used for the sound source transmission, the corresponding DMAEN bit of MXDRQCTRL is set to 0 after the DMA-transmission is completed.
508	CHAPTER 20:Sound Mixer 3. Operation of the Sound Mixer	Revised the below: 3.10. Sound Source Sampling Rate 3.10.2. Sampling Rate Conversion and FIR Filter Error) Note: − Filter characteristics have been selected based on Spansion rules. Correct) Note: − Filter characteristics have been selected based on Cypress rules.
509 516 518 519 520 523 524 525 527 528 530 834	CHAPTER 20:Sound Mixer 4. Registers CHAPTER 23: Stereo Audio DAC 3. Operation 3.12. Bridge-Tied Load (BTL)	Revised the below: Error) Chanel Correct) Channel

Page	Section	Change Results
510	CHAPTER 20:Sound Mixer 4. Registers	Added note as below: 4.1. Mixer Channel Register (MXCH) [bit9] PMIS4: PMIS4 input setting [bit8] PMIS3: PMIS3 input setting [bit7] PMIS2: PMIS2 input setting [bit6] PMIS1: PMIS1 input setting [bit5] PMIS0: PMIS0 input setting Correct) Note: PMISn bit must not be changed from 0 to 1 when DMAENCHn bit of MXDRQCTRL register is 1.
511	CHAPTER 20:Sound Mixer 4. Registers	Added note as below: 4.2. Mixer Output Control Register (MXOCTRL) [bit11:8] DATATN[3:0]: Number of transfers to output destination [bit2:0] MACRO[2:0]: Output destination address Correct) Note: These bits can be set at the initialization. After that, it should not be changed.
517	CHAPTER 20:Sound Mixer 4. Registers	Revised the below: 4.5. Mixer Channel Monaural Register (MXCHMONO) [bit1:0] PMIS0MONO: PMIS0 sound source processing mode setting Error) PMISnMONO[1:0] n=0 to 4 Correct) PMISnMONO[1:0] (n=0 to 4)
517	CHAPTER 20:Sound Mixer 4. Registers	Added note as below: 4.5. Mixer Channel Monaural Register (MXCHMONO) [bit9:8] PMIS4MONO: PMIS4 sound source processing mode setting [bit7:6] PMIS3MONO: PMIS3 sound source processing mode setting [bit5:4] PMIS2MONO: PMIS2 sound source processing mode setting [bit3:2] PMIS1MONO: PMIS1 sound source processing mode setting [bit1:0] PMIS0MONO: PMIS0 sound source processing mode setting Correct) Note: These bits must not be changed during sound source transmission of the channel.
519	CHAPTER 20:Sound Mixer 4. Registers	Revised the below: 4.7. Mixer Channel Volume2 Register (MXCHVOL2) [bit15:8] PMIS0VOL: Gain setting for PMIS0VOL volume control Error) [bit15:8] PMIS0VOL: Gain setting for PMIS2VOL volume control Correct) [bit15:8] PMIS0VOL: Gain setting for PMIS0VOL volume control
533	CHAPTER 20:Sound Mixer 4. Registers	Added note as below: 4.17. Mixer Buffer Clear Register (MXBUFFCLR) [bit9] PMIS4BCLR: PMIS4 input buffer initialization [bit8] PMIS3BCLR: PMIS3 input buffer initialization [bit7] PMIS2BCLR: PMIS2 input buffer initialization [bit6] PMIS1BCLR: PMIS1 input buffer initialization [bit5] PMIS0BCLR: PMIS0 input buffer initialization Correct) Note: PMISn input buffer initialization must not be executed when DMAENCHn bit of MXDRQCTRL register is 1.

Page	Section	Change Results
554	CHAPTER 20:Sound Mixer 4. Registers	Revised the below: 4.28. Mixer PCM/I2S n Data Address Register0-15 (MXPMISnDADR0-15, n=0 to 4) Delete of blank. Error) PMIS n DADR0 Correct) PMISnDADR0
820	CHAPTER 23: Stereo Audio DAC 1. Overview	Added comments as below: Correct) Also refer to the chapter of "Sound System configuration" how to configure the Stereo Audio DAC.
841	CHAPTER 23: Stereo Audio DAC 4. Registers	Added note as below: 4.4. DAC Control Register(DACTRL) [bit20:16] FEST[4:0]:FIFO Empty threshold [bit8] DMAEN: DMA enable Correct) Note: These bits must not be changed during the DMA transfers. / This bit must not be changed during the DMA transfers
852	CHAPTER 24: Inter IC Sound (I2S) 1. Overview	Revised the below: Error) I2S module is a full duplex, synchronous serial audio interface for multi channel specification. Correct) I2S module is a full duplex, synchronous serial audio interface for multichannel specification.
853	CHAPTER 24: Inter IC Sound (I2S) 2. Configuration and Block Diagram	Revised the below: Clocking of I2S Error) The supply clock of I2S can be internal (CLK_HAPPS1B0) or external (ECLK) source. For details refer to device specific datasheet. This clock is then prescaled to required frequency through I2S control register bits I2Sn_CNTREG:CKRT[5:0] Correct) The supply clock of I2S can be internal or external (ECLK) source. This clock is then pre-scaled to required frequency through I2S control register bits I2Sn_CNTREG:CKRT[5:0]
868 869	CHAPTER 24: Inter IC Sound (I2S) 4. Registers	Added comments as below: Memory Layout of I2S Registers Corrects) Access Size in Register table
872	CHAPTER 24: Inter IC Sound (I2S) 4. Registers	Revised the below: 4.2. Transmission FIFO Data Register (I2Sn_TXFDAT0 to 15) Error) OFFSET: 0x0 + i*0x4 Correct) OFFSET: 0x40 + i*0x4

Page	Section	Change Results
873 875 889 892	CHAPTER 24: Inter IC Sound (I2S) 4. Registers	Revised the below: 4.3. Control Register (I2Sn_CNTREG) [bit31:26] CKRT : Clock Divider [bit10] ECKM : Clock Selector 4.9. Interrupt Control Register (I2Sn_INTCNT) [bit5:4] RPTMR : Rx Completion Timer 4.10. Status Register (I2Sn_STATUS) [bit19] EOPI : Interrupt Flag for Rx Timer Error) CLK_HAPPS1B0 Correct) Internal clock
911	CHAPTER 25:Programmable CRC 4. Registers	Revised the below: Figure 25-6 Memory Layout of Programmable CRC Added) ”ACCESS_SIZE" B, H, W
1009	CHAPTER 28: LCD Controller 6.3. LCDC Control Register 1: LCR1	Revised the below: Error) [bit7 to bit0] Reserved When the LCD is used, set "11111111" always. Correct) [bit7 to bit0] Reserved When the LCD is used, set "11111111" always. When it is not used, set "00000000".
1010	CHAPTER 28: LCD Controller 6.4. Common Pin Switching Register: LCDCMR	Revised the below: Error) [bit3 to bit0] Reserved When the LCD is used, set "1111" always. Correct) [bit3 to bit0] Reserved When the LCD is used, set "1111" always. When it is not used, set "0000".
1122 1123 1124	CHAPTER 32: Sound System Configuration 1. Overview 2. Configuration and Block Diagram 3. Operation	Revised the below: Function Module 1.1. Input Module 3.1.1. Configuration of Input Modules Error) DMA Correct) DMAC
1123	CHAPTER 32: Sound System Configuration 2. Configuration and Block Diagram	Added note as below: Figure 32-1 Correct) Note: I2S connected to the Sound Mixer supports only Transfer mode.

<table>
<tr><th>Page</th><th>Section</th><th>Change Results</th></tr>
<tr><td>1127</td><td>CHAPTER 32:Sound System Configuration 3.3.1 Disabling the Output Module</td><td>Added note as below:

Correct)
Notes:
− When the PCM-PWM or I2S is used as output module and the output module is changed from "enabled" to "disabled" (*1), the sound mixer should not be disabled and the output module should not be enabled during 16 sampling cycles. If this sequence is not followed, PCM data requested before disabling the output module may be transferred to the output module after enabling the output module. This may cause the DMA block error, Tx Block Size Error or FIFO overflow error.
− (*1) In PCM-PWM case, this means that the global enable bit (PCMPWMi_CONTROL,EN) is changed from 1 to 0.
− (*1) In I2S case, this means that the transmission enable bit (I2Sn_OPRREG:TXENB) or the I2S enable bit (I2Sn_OPRREG.START) is changed from 1 to 0.</td></tr>
<tr><td>1130</td><td>CHAPTER 32: Sound System Configuration 4. Configuration and limitation of each module</td><td>Added chapter as below:

Corrects)
4. Configuration and limitation of each module
4.1. Interface between SMIX and DAC
4.2. Interface between SMIX and PCM-PWM
4.3. Interface between SMIX and I2S
4.4. Interface between SWFG and SMIX
4.5. Interface between DMAC and SMIX
4.6. Interface between CPU and SMIX</td></tr>
<tr><td>1136
1137
1139
1151
1156
1157
1159
1178</td><td>CHAPTER 33: Graphics Subsystem</td><td>Revised the below:
Add Chapter of LCD Bus Interface

Error)
HW Peripheral Manual for LCD Bus Interface (LCDBusIF) v1.11

Correct)
The chapter of LCD Bus Interface in this manual</td></tr>
<tr><td>1145</td><td>CHAPTER 33: Graphics Subsystem 3.1.2 Address map</td><td>Revised the below:

Error)
<table>
<tr><th>Offset [Hex in Bytes]</th><th>Size [Hex in Bytes]</th><th>Description</th><th>Register Map</th></tr>
<tr><td>:</td><td>:</td><td>:</td><td>:</td></tr>
<tr><td>5020_0400</td><td>0000_4400</td><td>2D core (Display)</td><td>See [1]</td></tr>
<tr><td>5020_4800</td><td>0000_0400</td><td>LCD Bus interface</td><td>See [2]</td></tr>
</table>
Correct)
<table>
<tr><th>Offset [Hex in Bytes]</th><th>Size [Hex in Bytes]</th><th>Description</th><th>Register Map</th></tr>
<tr><td>:</td><td>:</td><td>:</td><td>:</td></tr>
<tr><td>5020_0400</td><td>0000_4400</td><td>2D core (Display)</td><td>See HW Peripheral Manual for 2D Graphic Core v1.1</td></tr>
<tr><td>5020_4800</td><td>0000_0400</td><td>LCD Bus interface</td><td>See the chapter of LCD Bus Interface in this manual</td></tr>
</table></td></tr>
</table>

Page	Section	Change Results
1173	CHAPTER 33: Graphics Subsystem 3.6.2.11 dsp0_ClockDivider	Revised the below: [bit23:8] dsp0_ClockDivider Added) In case of TTL: The fix point clock divider means unless the clock divider is an integer, there will be a mix of lower and higher clock frequencies. Maximal peak frequency is calculated as follows. If floating part of the DIVIDER is >= 0.5: fDSP_CLK_max = fREF_CLK / (2 x int(dsp0/1_ClockDivider) + 1) If floating part of the DIVIDER is < 0.5: fDSP_CLK_max = fREF_CLK / (2 x int(dsp0/1_ClockDivider)) (With DIVIDER = fREF_CLK / fDSP_CLK) In case of RSDS: Also unless the clock divider implemented is an integer, the duty cycle of the display clock is not 50%. The display clock consists of higher frequencies of fDSP_CLK_max = fREF / int(DIVIDER) and lower frequencies of fDSP_CLK_max = fREF / int(DIVIDER+1) This needs to be considered for the RSDS case where both edges of the display clock are used.
1327 - 1366	CHAPTER 37: LCD Bus Interface	Add the this chapter.
1367	CHAPTER 38: Appendix	Revised the below: Error) CHAPTER 37: Pin Status in Each CPU State Correct) CHAPTER 38: Appendix
1368 1369 1370	CHAPTER 38: Appenidix 1.1. TEQFP-208 Pin 1.2. TEQFP-176 Pin 1.3. TEQFP-144 Pin	Revised the below: Error) *3 Before shutoff power domain2, always retain I/O control in power domain 2 (by setting the SYSC0_SPECFGR.HOLDIO_PD2 to 1). always set 3V I/O control (by setting the SYSC0_SPECFGR.IO3RSTC to 1). always set 35V I/O control (by setting the SYSC0_SPECFGR.IO35RSTC to 1). Correct) *3 I/O must be retained by setting the SYSC0_SPECFGR.HOLDIO_PD2 to 1, before power domain2 shutoff. IO reset must be enabled to avoid unstable I/O state, before relevant external IO power supply shutoff. - Reset for 3V-I/O must be enabled by setting the SYSC0_SPECFGR.IO3RSTC to 1, before VCC3 power supply shutoff. - Reset for 35V-I/O must be enabled by setting the SYSC0_SPECFGR.IO35RSTC to 1, before VCC35 power supply shutoff. IO reset is not required if power supply is retained.
1368 1369 1370	CHAPTER 38: Appenidix 1.1. TEQFP-208 Pin 1.2. TEQFP-176 Pin 1.3. TEQFP-144 Pin	Add the below Correct) Legend Inhibit – User is instructed not to input from this pin. If inputted, current flows through I/O. Input turned off – Input is actually disconnected.
1371	CHAPTER 39: Major Changes	Revised the below Error) CHAPTER 38: APPENDIX: Major Changes Correct) CHAPTER 39: Major Changes
Rev.*F		

Page	Section	Change Results
29	CHAPTER 2: Function List 1. Function List	Revised the below Function : CRC Error) 1 unit Correct) 4 unit
29	CHAPTER 2: Function List 1. Function List	Revised the below Function : DDR HSSPI Error) 2 ch Correct) 1 ch
37	CHAPTER 3: Product Description 2. Product Description	Add the below Feature: Clock Correct) Main Oscillation Stabilization Wait Time (at 4 MHz):8.19ms (Initial value)
37	CHAPTER 3: Product Description 2. Product Description	Add the below Feature: Embedded CR oscillation Correct) See the Traveo™ Platform Hardware Manual in detail. Stabilization time is as followings. − 0.35 ms to 0.8 ms for 4 MHz (Fast clock) − 0.43 ms to 1.28 ms for 100 kHz (Slow clock)
37	CHAPTER 3: Product Description 2. Product Description	Revised the below Feature:. Reset Error) Based on Cortex R5F platform Following resets are not mounted on this device. − INITX − SRSTX Correct) RSTX pin + MD pin simultaneous assert INITX (Same as INITX pin input) − Occurrence factor: Simultaneously inputting “L” level to RSTX pin and inputting “L” level to MD pin − Release factor: Inputting “H” level to RSTX pin See the Traveo™ Platform Hardware Manual in detail. Following resets are not mounted on this device. − SRSTX (and nSRST pin) The product series does not support EX5VRST and writing EX5VRSTCNT bits in SYSC0_SPECFGR has no effect.
38	CHAPTER 3: Product Description 2. Product Description	Revised the below Feature:. PLL / SSCG PLL Error) Down spread mode is only supported and available. Correct) Product supports down spread and center spread modes with the conditions defined in chapter "Internal Clock Timing" on the datasheet.
40	CHAPTER 3: Product Description 2. Product Description	Delete comments as below Feature:. I2S Error) − I2S has its own PPU, but the function is fixed to disable.

Page	Section	Change Results
40	CHAPTER 3: Product Description 2. Product Description	Revised the below Feature:. Hyper BUS I/F Error) The following register is not supported and cannot be used. − Controller Status Register (HYPERBUSIn_CSR) − Interrupt Status Register (HYPERBUSIn_ISR) − Write Protection Register (HYPERBUSIn_WPR) − Test Register (HYPERBUSIn_TEST) Correct) The following register is not supported and cannot be used. − Controller Status Register (HYPERBUSIn_CSR) − Interrupt Enable Register (HYPERBUSIn_IEN) − Interrupt Status Register (HYPERBUSIn_ISR) − Write Protection Register (HYPERBUSIn_WPR) − Test Register (HYPERBUSIn_TEST)
41	CHAPTER 3: Product Description 2. Product Description	Add the below Correct) Feature: Power Supply Control(PSC) PSC (PSC_1) output is used for external 1.2-V power supply module control and automatically switched with the following condition. "High": Request to supply VCC12 - "Power ON Reset" is released - CPU wakes up from PSS shutdown mode "Low": Request to stop supplying VCC12 - CPU transfers from RUN mode to PSS shutdown mode. For timing chart of output signals include PSC in detail, see the "S6J3300 Hardware Manual" and chapter "State Transition"
46	CHAPTER 3: Product Description 3.3. Restriction	Add the below Function: Power domain Correct) Related Register and Configuration : SYSC0_RUNPDCFGR.PD6_1EN SYSC0_PSSPDCFGR.PD6_1EN SYSC0_APPPDCFGR.PD6_1EN SYSC0_STSPDCFGR.PD6_1EN Restriction : PD6 is controlled by PD6_0EN only. PD6 is not effected by PD6_1EN setting.
68	CHAPTER 7: Memory and Base Address Map 3. Base Address Map	Revised the below Error) START Address / END Address / Group / Function / PPU_No B064_0000 / B064_0FFF / MCU_CONFIG_GROUP / PWM / 364 Correct) START Address / END Address / Group / Function / PPU_No B064_0000 / B064_0FFF / MCU_CONFIG_GROUP / Indicator PWM / 364

Page	Section	Change Results
95	CHAPTER 11: Port Configuration	Revised the below Error) 1. Overview 2. Configuration and Block Diagram 3. Operation 4. Registers 5. Configuration Procedure Correct) 1. Overview 2. Configuration and Block Diagram 3. Operation 4. Registers 5. Configuration Procedure 6. Precautions
185	CHAPTER 11: Port Configuration 3.5. Output Drive Capacity Setting	Add the below Correct) Note that the drive capacity depends on power supply voltage. The table only describes the representative value when 5V is used as standard value of power supply usage. Please see the actual characteristics in datasheet.
206	CHAPTER 11: Port Configuration 6. Precautions	Add the below Correct) 6.2 I^2C setting For I2C fast-mode output pins“P1_09 (SCL16), P1_10 (SDA16), P1_15 (SCL17), P1_16 (SDA17)”to output Hi-Z after reset, the registers must be set in the following order. (1) Set the MD (Operation Mode Setting Bits) of the SMR (Serial Mode Register) to I2C mode. (2) Set the POF (Port Output Function Selection Bit) of the PPC_PCFGRijj (Port Setting Register) to I2C. If not set in the above order, the MCU will output "H" level before outputting Hi-Z to each port.
223	CHAPTER 13: Low Voltage Detection 4. Registers	Delete comments as below [Bit30] LVDL1S Internal low-voltage detection voltage operation selection bit Error) Note: - LVDL1V will be initial value with reset. If LVDL1V is changed from initial value, LVDL1S should be configured as interrupt.
223	CHAPTER 13: Low Voltage Detection 4. Registers	Delete comments as below [Bit26:25] LVDL1V Internal low-voltage detection voltage setting bits Error) Note: - LVDL1V will be initial value with reset. If LVDL1V is changed from initial vale, LVDL1S should be configured as interrupt.
223	CHAPTER 13: Low Voltage Detection 4. Registers	Delete comments as below [Bit22] LVDL2S Internal low-voltage detection voltage operation selection bit Error) Note: - LVDL2V will be initial value with reset. If LVDL2V is changed from initial vale, LVDL2S should be configured as interrupt.

Page	Section	Change Results
223	CHAPTER 13: Low Voltage Detection 4. Registers	Delete comments as below [Bit18:17] LVDL2V Expanded internal low-voltage detection voltage setting bits Error) Notes: - LVDL2V will be initial value with reset. If LVDL2V is changed from initial vale, LVDL2S should be configured as interrupt.
224	CHAPTER 13: Low Voltage Detection 4. Registers	Delete comments as below [Bit11:9] LVDH1V External low-voltage detection voltage setting bits Error) Note: - LVDH1V will be initial value with reset. If LVDH1V is changed from initial vale, LVDH1S should be configured as interrupt.
225	CHAPTER 13: Low Voltage Detection 4. Registers	Delete comments as below [Bit6] LVDH2S Internal low-voltage detection voltage operation selection bit Error) Note: - LVDH2S should be used as Interrupt. If LVDH2S is configured as Reset, the reset itself will be released immediately because LVDH2 is initially disabled.
500	CHAPTER 20: Sound Mixer 3.2. Volume Effects	Add the below Correct) Notes: – Even if the minimum gain or mute is applied to the sound source, the mixer does not always output ALL 0 data. In that case, positive data become ALL 0, but negative data become ALL 1 (-1 in decimal). Due to this, slight sound may be heard in accordance with your output environment.
582 583	CHAPTER 21: Ethernet MAC 3.1.8 Priority Queuing in the Ethernet MAC DMA	Revised and add the below Error) In the transmit direction, higher priority queues are always serviced before lower priority queues. Correct) In the transmit direction, higher priority queues are serviced before lower priority queues. Error) - Correct) If a higher priority transmit queue contains relatively longer buffer, it can happen that a shorter frame from a lower priority queue is transferred before the higher prority queue. Error) Each screener register is programmable vie the APB Slave Interface. Correct) Each screener register is programmable vie the AHB Slave Interface.
852	CHAPTER 24: Inter IC Sound (I2S) 1. Overview	Revised the below Error) During the master mode, SCK clock can be output by dividing external clock or internal clock (it is selectable by register). Correct) During the master mode, SCK clock can be output by dividing external clock or internal bus clock (it is selectable by register).

Page	Section	Change Results
852	CHAPTER 24: Inter IC Sound (I2S) 1. Overview	Revised the below Error) ▪ Selecting clock frequency source of serial bit clock in the master mode (internal and external clock) ▪ Setting clock frequency dividing ratio in the master mode ▪ Frequency of SCK = (frequency of internal clock or external clock) / (2 x I2Sn_CNTREG:CKRT[5:0]) Correct) ▪ Selecting clock frequency source of serial bit clock in the master mode (internal bus clock and external clock) ▪ Setting clock frequency dividing ratio in the master mode ▪ Frequency of SCK = (frequency of internal bus clock or external clock) / (2 x I2Sn_CNTREG:CKRT[5:0])
853	CHAPTER 24: Inter IC Sound (I2S) 2. Configuration and Block Diagram	Revised the below Error) The supply clock of I2S can be internal or external (ECLK) source. Correct) The supply clock of I2S can be internal bus clock or external (ECLK) clock source.
873	CHAPTER 24: Inter IC Sound (I2S) 4.3. Control Register (I2Sn_CNTREG)	Revised the below [bit31:26] CKRT : Clock Divider Error) Internal clock is divided at I2Sn_CNTREG:ECKM = '0', and external clock (ECLK) is divided at I2Sn_CNTREG:ECKM = '1'. Correct) Internal bus clock is divided at I2Sn_CNTREG:ECKM = '0', and external clock (ECLK) is divided at I2Sn_CNTREG:ECKM = '1'.
875	CHAPTER 24: Inter IC Sound (I2S) 4.3. Control Register (I2Sn_CNTREG)	Revised the below [bit10] ECKM : Clock Selector Error) Bit 0:Internal clock is divided and output Correct) Bit 0:Internal bus clock is divided and output
889	CHAPTER 24: Inter IC Sound (I2S) 4.9. Interrupt Control Register (I2Sn_INTCNT)	Revised the below [bit5:4] RPTMR : Rx Completion Timer Error) Bit 01:54000 Internal clock cycles Bit 10:108000 Internal clock cycles Bit 11:216000 Internal clock cycles Correct) Bit 01:54000 Internal bus clock cycles Bit 10:108000 Internal bus clock cycles Bit 11:216000 Internal bus clock cycles
892	CHAPTER 24: Inter IC Sound (I2S) 4.10. Status Register (I2Sn_STATUS)	Revised the below [bit19] EOPI : Interrupt Flag for Rx Timer Error) When the reception FIFO is not empty and the number of data is less than or equal to the threshold, the reception timer is incremented with each internal clock cycle. Correct) When the reception FIFO is not empty and the number of data is less than or equal to the threshold, the reception timer is incremented with each internal bus clock cycle.

<table>
<tr><th>Page</th><th>Section</th><th>Change Results</th></tr>
<tr><td>1055</td><td>CHAPTER 29: Indicator PWM
3.5. Indicator PWM Interrupts</td><td>Add the below

Correct)
3.5.3. Return from PSS Mode

An interrupt of indicator PWM can be applied to a trigger of returning from PSS mode. Then the bus clock CLK_SYSC0P should be supplied during the PSS mode.</td></tr>
<tr><td>1368
1369
1370</td><td>CHAPTER 38: Appenidix
1.1. TEQFP-208 Pin
1.2. TEQFP-176 Pin
1.3. TEQFP-144 Pin</td><td>Add the below

Correct)
Hi-Z – Output is Hi-Z.</td></tr>
<tr><td colspan="3">Rev.*G</td></tr>
<tr><td>33</td><td>CHAPTER 3: Product Description
3.2. Product Description
Table 3-1: Product Features</td><td>Added the shading parts as below:

Correct)<table><tr><td>Feature</td><td>Description</td></tr><tr><td>A/D Converter</td><td>AN39 to AN63 are not support for S6J33xxxAx, S6J33xxxCx, S6J33xxxEx, and S6J33xxxGx option.</td></tr></table></td></tr>
</table>

<table>
<tr><th>Page</th><th>Section</th><th>Change Results</th></tr>
<tr><td>36</td><td>CHAPTER 3: Product Description
2.1. Ethernet</td><td>Deleted the shading parts as below:

Error)
<table>
<tr><th>Functions</th><th>Remark</th></tr>
<tr><td>Direct Memory Access Interface.
- partial store and forward
- force max amba burst tx/rc
- Priority Queueing (Screening)</td><td></td></tr>
<tr><td>External FIFO Interface</td><td></td></tr>
<tr><td>Additional Low Latency TX FIFO Interface for DMA configurations</td><td></td></tr>
<tr><td>MAC Transmit Block
- half-duplex
- collision
- back_pressure</td><td></td></tr>
<tr><td>MAC Filtering Block
- external address match
- VLAN tag
- Wakeup On Lan</td><td></td></tr>
<tr><td>IEEE 1588 and IEEE 802.1AS Support</td><td></td></tr>
<tr><td>MAC PFC Priority Based Pause Frame Support</td><td></td></tr>
<tr><td>Energy Efficient Ethernet support</td><td></td></tr>
<tr><td>LPI Operation in Cadence IP</td><td></td></tr>
<tr><td>802.1Qav Support – Credit Based Shaping</td><td></td></tr>
<tr><td>PHY Interface
- GMII
- SGMII
- TBI</td><td></td></tr>
<tr><td>10/100/1000 Operation
- 10 M
- 1000 M</td><td></td></tr>
<tr><td>SGMII Operation</td><td></td></tr>
<tr><td>Jumbo Frames</td><td></td></tr>
<tr><td>Physical Control Sub-Layer</td><td></td></tr>
</table>
Correct)
<table>
<tr><th>Functions</th><th>Remark</th></tr>
<tr><td>External FIFO Interface</td><td></td></tr>
<tr><td>Additional Low Latency TX FIFO Interface for DMA configurations</td><td></td></tr>
<tr><td>MAC Transmit Block
- half-duplex
- collision
- back_pressure</td><td></td></tr>
<tr><td>MAC Filtering Block
- external address match
- Wakeup On Lan</td><td></td></tr>
<tr><td>Energy Efficient Ethernet support</td><td></td></tr>
<tr><td>LPI Operation in Cadence IP</td><td></td></tr>
<tr><td>PHY Interface
- GMII
- SGMII
- TBI</td><td></td></tr>
<tr><td>10/100/1000 Operation
- 1000 M</td><td></td></tr>
<tr><td>SGMII Operation</td><td></td></tr>
<tr><td>Jumbo Frames</td><td></td></tr>
<tr><td>Physical Control Sub-Layer</td><td></td></tr>
</table></td></tr>
</table>

<table>
<tr><th>Page</th><th>Section</th><th>Change Results</th></tr>
<tr><td>70</td><td>CHAPTER 9: DMA Channel Activation Factors
1. Factors list</td><td>Added the shading parts as below:

Correct)
<table>
<tr><th>Client number</th><th>Peripheral Functions</th><th>Enable Setting of IRQ request</th><th>Remarks</th></tr>
<tr><td>0 to 8</td><td>Reserved</td><td>-</td><td></td></tr>
<tr><td>9</td><td>WORK FLASH</td><td>Unnecessary</td><td>*2</td></tr>
</table>
</td></tr>
<tr><td>75</td><td></td><td>Added the shading parts as below:

Correct)
*2: DSTP_ACK function is set by DMAi_CMCICm:BEHSTPACK.</td></tr>
<tr><td>491,</td><td>CHAPTER 21: Ethernet MAC</td><td>Added the shading parts as below:

Correct)
1. Overview
2. Configuration and Block Diagram
3. Operation of the Ethernet MAC
4. Registers
5. Functional Limitations</td></tr>
<tr><td>690, 691</td><td>CHAPTER 21: Ethernet MAC
5. Functional Limitations</td><td>Added the below Functional Limitations section:

Correct)
5. Functional Limitations</td></tr>
<tr><td>746</td><td>CHAPTER 23: Stereo Audio DAC
3.10. Sampling Frequency and Oversampling Setting of DA Converter
Table 23-5 Setting of Dividing Frequency</td><td>Added the shading parts as below:

Correct)
NOTE
- The exact value cannot be configured. But the deviation or mismatch can be minimized.</td></tr>
<tr><td colspan="3">Rev.*H</td></tr>
</table>

<table>
<tr><th>Page</th><th>Section</th><th>Change Results</th></tr>
<tr><td>44</td><td>CHAPTER 5: Clock Configuration
2. Operation</td><td>
Revised the shading parts as below:

Table 5-1 Source Clock of Each Function

Error)
<table>
<tr><th>Function</th><th>Clock Name</th><th>Source Clock System</th><th>Configuration</th></tr>
<tr><td>CPU</td><td>CLK_CPU</td><td>SSCG0</td><td>See Traveo™ Platform Hardware Manual</td></tr>
<tr><td>FLASH</td><td>CLK_FCLK</td><td>SSCG0</td><td>See Traveo™ Platform Hardware Manual</td></tr>
<tr><td>HPM</td><td>CLK_HPM</td><td>SSCG0</td><td>See Traveo™ Platform Hardware Manual</td></tr>
<tr><td>DMA</td><td>CLK_DMA</td><td>SSCG0</td><td>See Traveo™ Platform Hardware Manual</td></tr>
<tr><td>LLPBM</td><td>CLK_LLPBM2</td><td>SSCG0</td><td>See Traveo™ Platform Hardware Manual</td></tr>
<tr><td>MCU configuration</td><td>CLK_SYSC0H
CLK_COMH
CLK_RAM0H
CLK_RAM1H</td><td>SSCG0</td><td>See Traveo™ Platform Hardware Manual</td></tr>
<tr><td>Ethernet Media-LB</td><td>CLK_HAPP1B0</td><td>SSCG0</td><td>See Traveo™ Platform Hardware Manual</td></tr>
<tr><td>Registers of Indicator PWM and LCDE</td><td>CLK_SYSC0P</td><td>SSCG0</td><td>See Traveo™ Platform Hardware Manual</td></tr>
</table>
Correct)
<table>
<tr><th>Function</th><th>Clock Name</th><th>Source Clock System</th><th>Configuration</th></tr>
<tr><td>CPU</td><td>CLK_CPU</td><td>SSCG0/PLL0</td><td>See Traveo™ Platform Hardware Manual</td></tr>
<tr><td>FLASH</td><td>CLK_FCLK</td><td>SSCG0/PLL0</td><td>See Traveo™ Platform Hardware Manual</td></tr>
<tr><td>HPM</td><td>CLK_HPM</td><td>SSCG0/PLL0</td><td>See Traveo™ Platform Hardware Manual</td></tr>
<tr><td>DMA</td><td>CLK_DMA</td><td>SSCG0/PLL0</td><td>See Traveo™ Platform Hardware Manual</td></tr>
<tr><td>LLPBM</td><td>CLK_LLPBM2</td><td>SSCG0/PLL0</td><td>See Traveo™ Platform Hardware Manual</td></tr>
<tr><td>MCU configuration</td><td>CLK_SYSC0H
CLK_COMH
CLK_RAM0H
CLK_RAM1H</td><td>SSCG0/PLL0</td><td>See Traveo™ Platform Hardware Manual</td></tr>
<tr><td>Ethernet Media-LB</td><td>CLK_HAPP1B0</td><td>SSCG0/PLL0</td><td>See Traveo™ Platform Hardware Manual</td></tr>
<tr><td>Registers of Indicator PWM and LCDE</td><td>CLK_SYSC0P</td><td>SSCG0/PLL0</td><td>See Traveo™ Platform Hardware Manual</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>Page</th><th>Section</th><th>Change Results</th></tr>
<tr><td>45, 46</td><td></td><td>
Revised the shading parts as below:

Table 5-2 Source Clock - Local Clock List

Error)
<table>
<tr><th>Source Clock System</th><th>Clock Name</th><th>Function</th><th>Local Clock Name</th><th>Description</th></tr>
<tr><td>SSCG0</td><td>CLK_HAPP1B0</td><td>Ethernet MAC</td><td>No Define</td><td>-</td></tr>
<tr><td>SSCG0</td><td>CLK_HAPP1B0</td><td>Hyperbus Interface (MCU)</td><td>No Define</td><td>Operation Clock for Control register</td></tr>
<tr><td>SSCG0</td><td>CLK_HPM</td><td>Hyperbus Interface (MCU)</td><td>No Define</td><td>Operation Clock for Main controller</td></tr>
<tr><td>SSCG0</td><td>CLK_SYSC0P</td><td>CLK_LCP1A:or PWM</td><td>No Define</td><td>-</td></tr>
<tr><td>SSCG0</td><td>CLK_HAPP1B0</td><td>Media Local Bus Interface (MediaLB)</td><td>No Define</td><td>-</td></tr>
<tr><td>SSCG0</td><td>CLK_HAPP1B0</td><td>Memory Protection Unit for AHB</td><td>No Define</td><td>-</td></tr>
<tr><td>SSCG0</td><td>CLK_HAPP1B0</td><td>Memory Protection Unit for AHB</td><td>No Define</td><td>-</td></tr>
<tr><td>SSCG0</td><td>CLK_HAPP1B0</td><td>Memory Protection Unit for AXI</td><td>No Define</td><td>-</td></tr>
<tr><td>SSCG0</td><td>CLK_HAPP1B0</td><td>Memory Protection Unit for AXI</td><td>No Define</td><td>-</td></tr>
<tr><td>SSCG0</td><td>CLK_HAPP1B0</td><td>Programmable CRC</td><td>No Define</td><td>-</td></tr>
</table>
Correct)
<table>
<tr><th>Source Clock System</th><th>Clock Name</th><th>Function</th><th>Local Clock Name</th><th>Description</th></tr>
<tr><td>PLL0 or SSCG0</td><td>CLK_HAPP1B0</td><td>Ethernet MAC</td><td>No Define</td><td>-</td></tr>
<tr><td>PLL0 or SSCG0</td><td>CLK_HAPP1B0</td><td>Hyperbus Interface (MCU)</td><td>No Define</td><td>Operation Clock for Control register</td></tr>
<tr><td>PLL0 or SSCG0</td><td>CLK_HPM</td><td>Hyperbus Interface (MCU)</td><td>No Define</td><td>Operation Clock for Main controller</td></tr>
<tr><td>PLL0 or SSCG0</td><td>CLK_SYSC0P</td><td>CLK_LCP1A:or PWM</td><td>No Define</td><td>-</td></tr>
<tr><td>PLL0 or SSCG0</td><td>CLK_HAPP1B0</td><td>Media Local Bus Interface (MediaLB)</td><td>No Define</td><td>-</td></tr>
<tr><td>PLL0 or SSCG0</td><td>CLK_HAPP1B0</td><td>Memory Protection Unit for AHB</td><td>No Define</td><td>-</td></tr>
<tr><td>PLL0 or SSCG0</td><td>CLK_HAPP1B0</td><td>Memory Protection Unit for AHB</td><td>No Define</td><td>-</td></tr>
<tr><td>PLL0 or SSCG0</td><td>CLK_HAPP1B0</td><td>Memory Protection Unit for AXI</td><td>No Define</td><td>-</td></tr>
<tr><td>PLL0 or SSCG0</td><td>CLK_HAPP1B0</td><td>Memory Protection Unit for AXI</td><td>No Define</td><td>-</td></tr>
<tr><td>PLL0 or SSCG0</td><td>CLK_HAPP1B0</td><td>Programmable CRC</td><td>No Define</td><td>-</td></tr>
</table>
</td></tr>
<tr><td>51</td><td>CHAPTER 7: Memory and Base Address Map
1. Overview</td><td>
Added the shading parts as below:

Error)
- Peripheral area (Peri area) [B000_0000 to B7FF_FFFF]
- Error Config area (ERRCFG) [FFFE_E000 to FFFE_FFFF]
- AppS area [B800_0000 to BFFF_FFFF]
- Register area in Graphic subsystem [5020_0000 to 5FFF_FFFF]

Correct)
- Peripheral area (Peri area) [B000_0000 to B7FF_FFFF]
- Error Config area (ERRCFG) [FFFE_E000 to FFFE_FFFF]
- AppS area [B800_0000 to BFFF_FFFF]
- Register area in Graphic subsystem [5020_0000 to 5FFF_FFFF]
- Backup RAM area (BACKUP_RAM) [0E80_0000 to 0E87_FFFF]
</td></tr>
<tr><td>153</td><td>CHAPTER 11: Port Configuration
3. Operation
3.2. Port Output Function Configuration
3.2.1. Standard Configuration</td><td>
Revised the shading parts as below:

Error)
<table>
<tr><th rowspan="2">Register (Offset)</th><th rowspan="2">Port</th><th colspan="8">Resource Functional Outputs</th></tr>
<tr><th>POF = 0</th><th>POF = 1</th><th>POF = 2</th><th>POF = 3</th><th>POF = 4</th><th>POF = 5</th><th>POF = 6</th><th>POF = 7</th></tr>
<tr><td>PPC_PCFGR112 (0x0058)</td><td>P1_12</td><td>GPIO_PODR1:POD12</td><td>SCS161_0</td><td>-</td><td>-</td><td>OCU8_OTD1_0</td><td>TOT17_0</td><td>PPG4_TOUT2_0</td><td>TX2_0</td></tr>
</table>
Correct)
<table>
<tr><th rowspan="2">Register (Offset)</th><th rowspan="2">Port</th><th colspan="8">Resource Functional Outputs</th></tr>
<tr><th>POF = 0</th><th>POF = 1</th><th>POF = 2</th><th>POF = 3</th><th>POF = 4</th><th>POF = 5</th><th>POF = 6</th><th>POF = 7</th></tr>
<tr><td>PPC_PCFGR112 (0x0058)</td><td>P1_12</td><td>GPIO_PODR1:POD12</td><td>SCS161_0</td><td>-</td><td>-</td><td>OCU8_OTD1_0</td><td>TOT17_0</td><td>-</td><td>TX2_0</td></tr>
</table>
</td></tr>
</table>

Revision History

Document Revision History

Document Title: S6J3300 Series 32-bit Microcontroller Traveo™ Family Hardware Manual Document Number: 002-10185		
Revision	**ECN No.**	**Description of Change**
**	5063955	New Specification
*A	5109148	Added part number and modified errors. For details, please see “Major Changes”
*B	5208201	Correct LVD voltage, Sound System Register. Change Main OSC Frequency. Add Note for LVD. For details, please see “APPENDIX: Major Changes”
*C	5371648	Additional comments of A/D Converter Unit vs Channels(Product Description and Port Description), State Transition, and Low Voltage Detection Registers TYPO: Inter IC Sound (I2S) Overview/ Configuration and Block Diagram/Registers, PCMPWM Registers For detail, see “Major Changes”
*D	5746505	Updated logo and Copyright.
*E	5782338	Correct LVD voltage guaranteed comments, Correct Sound Waveform Generator comments Add Chapter LCD Bus Interface For details, please see “Major Changes”
*F	5978188	Change Function list, Product Description and so on. For details, please see “Major Changes”
*G	6221278	Add *2 comment for WORK FLASH of DMA Channel Activation Factors List. Add an option limitation for Description of A/D Converter of Product Description. Update Ethernet Support Functions. Add Functional Limitations section of Ethernet MAC. Add Note for Setting of Dividing Frequency of Stereo Audio DAC. For details, please see "Major Changes"
*H	6944062	Updated some sentences. For details, please see “Major Changes”