

# AURIX™ TC33x/TC32x errata sheet

Marking/Step: (E)ES-AA, AA

10619AERRA

## About this document

### Scope and purpose

This document describes the deviations of the device from the current user documentation, to support the assessment of the effects of these deviations on your custom hardware and software implementations.

Please take note of the following information:

- This errata sheet applies to all temperature and frequency versions and to all memory size variants, unless explicitly noted otherwise. For a derivative synopsis, see the latest datasheet or user manual
- Multiple device variants are covered in this one document. If an issue is related to a particular module, and this module is not specified for a specific device variant, then the issue does not apply to that device variant
  - For example, issues with the identifier "EMEM" (extension memory) do not apply to devices for which no extension memory is specified ("EMEM" is used only as a generic example and may not be a feature of the device that this document covers)
- Devices marked with EES or ES are engineering samples which may not be completely tested in all functional and electrical characteristics and are therefore only suitable for evaluation
  - The specific test conditions for EES and ES are documented in a separate status sheet
- Some of the errata have workarounds which may be supported by the tool vendors. Some corresponding compiler switches may need to be set. Please refer to the respective documentation of your compiler
- To understand the effect of issues relating to the on-chip debug system, please refer to the respective debug tool vendor documentation

**Table 1** Current documentation

AURIX™ TC3xx User's Manual	V2.0.0	2021-02
AURIX™ TC33x/TC32x Appendix to User's Manual	V2.0.0	2021-02
TC33x/TC32x AA-Step Data Sheet	V1.1	2021-03
TriCore™ TC1.6.2 Core Architecture Manual:		
• Core Architecture (Vol. 1)	V1.2.2	2020-01-15
• Instruction Set (Vol. 2)	V1.2.2	2020-01-15
AURIX™ TC3xx Safety Manual	V2.0	2021-05-03

**Note:** Please contact your nearest Infineon sales office for additional information.

### Conventions used in this document

Each erratum identifier follows the pattern [Module]\_[Arch].[Type][Number]:

- [Module] = subsystem, peripheral, or function affected by the erratum
- [Arch] = microcontroller architecture where the erratum was initially detected
  - AI = Architecture Independent
  - TC = TriCore™
- [Type] = category of deviation
  - [none] = Functional deviation

### About this document

- P = Parametric deviation
- H = Application hint
- [Number] = ascending sequential number within the three previous fields

**Note:** *[Number] As this sequence is used over several derivatives, including already solved deviations, gaps can occur inside this numbering sequence*

## Table of contents

	<b>About this document</b> .....	1
	<b>Table of contents</b> .....	3
<b>1</b>	<b>Errata overview</b> .....	4
<b>2</b>	<b>Functional deviations</b> .....	16
<b>3</b>	<b>Parametric deviations</b> .....	115
<b>4</b>	<b>Application hints</b> .....	118
	<b>Revision history</b> .....	202
	<b>Disclaimer</b> .....	211

**1 Errata overview**

**1 Errata overview**

List of errata referenced in this document.

**Table 2 Functional deviations**

<b>Issue title</b>	<b>Change</b>	<b>Page</b>
[BROM_TC.013] CAN BSL does not send error message if no valid baudrate is detected		16
[BROM_TC.014] Lockstep comparator alarm for CPU0 after warm PORST, system or application reset if lockstep is disabled		16
[BROM_TC.016] Uncorrectable ECC error in Boot Mode Headers		16
[CCU_TC.005] ASC and CAN bootstrap loaders may not work if external clock is missing		17
[CPU_TC.130] Data Corruption when ST.B to local DSPR coincides with external access to same address		17
[CPU_TC.131] Performance issue when MADD or MSUB instructions use E0 or D0 register as accumulator		18
[CPU_TC.132] Unexpected PSW values used upon Fast Interrupt entry		18
[CPU_TC.133] Test sequence for DTAG single or double bit errors		19
[DAP_TC.005] DAP client_read: dirty bit feature of Cerberus' Triggered Transfer Mode		20
[DAP_TC.007] Incomplete client_blockread telegram in DXCM mode when using the "read CRCup" option		20
[DMA_TC.066] DMA double buffering operations - Update address pointer		20
[DMA_TC.067] DMA Double Buffering Software Switch buffer overflow		21
[DMA_TC.068] DMA Double Buffering lost DMA request		21
[DMA_TC.071] Daisy Chain request is lost when repeat triggers too soon		22
[FLASH_TC.053] Erase size limit for PFLASH		23
[FLASH_TC.055] Multi-bit errors detected by PFlash are not communicated to SPB masters		23
[FLASH_TC.056] Reset value for register HF_ECCC is 0x0000 0000 - Documentation correction		24
[FlexRay_AI.087] After reception of a valid sync frame followed by a valid non-sync frame in the same static slot the received sync frame may be ignored		24
[FlexRay_AI.088] A sequence of received WUS may generate redundant SIR.WUPA/B events		25
[FlexRay_AI.089] Rate correction set to zero in case of SyncCalcResult=MISSING_TERM		25
[FlexRay_AI.090] Flag SFS.MRCS is set erroneously although at least one valid sync frame pair is received		25
[FlexRay_AI.091] Incorrect rate and / or offset correction value if second Secondary Time Reference Point (STRP) coincides with the action point after detection of a valid frame		26

**(table continues...)**

**1 Errata overview**

**Table 2 (continued) Functional deviations**

Issue title	Change	Page
[FlexRay_AI.092] Initial rate correction value of an integrating node is zero if pMicroInitialOffsetA,B = 0x00		26
[FlexRay_AI.093] Acceptance of start-up frames received after reception of more than gSyncNodeMax sync frames		27
[FlexRay_AI.094] Sync frame overflow flag EIR.SFO may be set if slot counter is greater than 1024		27
[FlexRay_AI.095] Register RCV displays wrong value		28
[FlexRay_AI.096] Noise following a dynamic frame that delays idle detection may fail to stop slot		28
[FlexRay_AI.097] Loop back mode operates only at 10 MBit/s		28
[FlexRay_AI.099] Erroneous cycle offset during start-up after abort of start-up or normal operation		29
[FlexRay_AI.100] First WUS following received valid WUP may be ignored		29
[FlexRay_AI.101] READY command accepted in READY state		30
[FlexRay_AI.102] Slot status vPOC!SlotMode is reset immediately when entering HALT state		30
[FlexRay_AI.103] Received messages not stored in Message RAM when in Loop Back Mode		31
[FlexRay_AI.104] Missing start-up frame in cycle 0 at coldstart after FREEZE or READY command		31
[FlexRay_AI.105] RAM select signals of IBF1/IBF2 and OBF1/OBF2 in RAM test mode		32
[FlexRay_AI.106] Data transfer overrun for message transfers Message RAM to Output Buffer (OBF) or from Input Buffer (IBF) to Message RAM		32
[GETH_AI.043] Access type of register bit-fields not correctly marked in UM	new	34
[GTM_AI.254] TIM TDU: TDU_STOP=b101 not functional		44
[GTM_AI.308] TIM, ARU: Limitation that back-to-back TIM data transfers at full ARU clock rate cannot be transferred correctly with ARU dynamic routing feature		44
[GTM_AI.335] TOM output signal to SPE not functional if up/down counter mode is configured		45
[GTM_AI.340] TOM/ATOM: Generation of TRIG_CCU0/TRIG_CCU1 trigger signals skipped in initial phase of A/TOM SOMP one-shot mode		45
[GTM_AI.341] TOM/ATOM: False generation of TRIG_CCU1 trigger signal in SOMP one-shot mode with OSM_TRIG=1 when CM1 is set to value 1		46
[GTM_AI.345] SPE: Incorrect behaviour of direction change control via SPE_CMD.SPE_CTRL_CMD bits		47
[GTM_AI.346] ATOM SOMS mode: Shift cycle is not executed correctly in case the reload condition is deactivated with ATOM[i]_AGC_GLB_CTRL.UPEN = 0		48

**(table continues...)**

**1 Errata overview**

**Table 2 (continued) Functional deviations**

Issue title	Change	Page
[GTM_AI.347] TOM/ATOM: Reset of (A)TOM[i]_CH[x]_CN0 with TIM_EXT_CAPTURE are not correctly synchronized to selected CMU_CLK/CMU_FXCLK		48
[GTM_AI.349] TOM-SPE: OSM-Pulse width triggered by SPE_NIPD for selected CMU_FXCLK not correct		49
[GTM_AI.350] TOM-SPE: Update of SPE[i]_OUT_CTRL triggered by SPE_NIPD not working for a delay value 1 in TOM[i]_CH[x]_CM1		50
[GTM_AI.352] ATOM: Wrong reload of data from ARU in SOMS and SOMP mode if TIM_EXT_CAPTURE(x) or TRIGIN(x) is selected as clock source		50
[GTM_AI.353] SPEC-ATOM: Specification of the smallest possible PWM period in SOMP mode wrong, when ARU_EN=1		51
[GTM_AI.358] TOM/ATOM: Synchronous update of working register for RST_CCU0=1 and UDMODE=01 <sub>B</sub> not correct		52
[GTM_AI.359] TOM: Both edges on TOM_OUT_T at unexpected times for RST_CCU0=1 and UDMODE>0		53
[GTM_AI.360] SPEC-(A)TOM: PCM mode (BITREV=1) is only available for UDMODE=0		54
[GTM_AI.361] IRQ: Missing pulse in single-pulse interrupt mode on simultaneous interrupt and clear event		54
[GTM_AI.364] ATOM: ARU read request does not start at expected timepoint in UDMODE = 1 and UDMODE = 3		55
[GTM_AI.370] TOM/ATOM: Unexpected reset of CN0 in up-down counter mode and CM0 = 2		56
[GTM_AI.374] SPEC-ATOM: Statement on timing of duty cycle output level change not correct for SOMP up/down-counter mode	changed	56
[GTM_AI.375] ATOM: Data from ARU are read only once in SOMC mode even though ARU blocking mode is disabled while FREEZE = 1 and ENDIS = 0		57
[GTM_AI.376] TOM/ATOM: Interrupt trigger signals CCU0TC_IRQ and CCU1TC_IRQ are delayed by one CMU_CLK period related to the output signals		58
[GTM_AI.406] (A)TOM: FREEZE mode has no effect on (A)TOM_OUT_T in up-down counter mode with RST_CCU0 = 1		58
[GTM_AI.408] (A)TOM-RTL: Missing edge on output signal (A)TOM_OUT when CN0 is reset with force update event		59
[GTM_AI.410] GTM_AEI: The AEI bridge might not execute an accepted write transaction		60
[GTM_AI.411] A change of the BRIDGE_MODE register might be delayed indefinitely		60
[GTM_AI.419] TIM: Potentially wrong capture values		61
[GTM_AI.421] GTM_AEI: Changing BRIDGE_MODE.MSK_WR_RSP in pipeline mode can lead to violation of pipeline protocol		62
[GTM_AI.429] TIM: Missing glitch detection interrupt event		63
[GTM_AI.430] TIM: Unexpected increment of filter counter		63

**(table continues...)**

**1 Errata overview**

**Table 2 (continued) Functional deviations**

<b>Issue title</b>	<b>Change</b>	<b>Page</b>
[GTM_AI.431] TIM: Glitch detection interrupt event of filter is not a single cycle pulse		64
[GTM_AI.442] GTM Top Level: GTM_HALT mode not functional when cluster 0 clock is disabled		65
[GTM_AI.454] (A)TOM: No output if trigger generation feature is used		65
[GTM_AI.462] (A)TOM: Missing CCU0TC_IRQ interrupt signal		65
[GTM_AI.465] (A)TOM: Missing CCU0TC_IRQ interrupt signal for UDMODE > 0		66
[GTM_AI.466] TOM: Unexpected behavior of TOM_OUT_T for UDMODE>0		67
[GTM_AI.487] GTM_AEI: Changing BRIDGE_MODE[2:0] in pipeline mode can lead to violation of pipeline protocol		67
[GTM_AI.488] GTM_AEI: Turning off BRIDGE_MODE.MSK_WR_RSP in asynchronous mode might lead to following transactions being corrupted		68
[GTM_AI.516] SPE-RTL: IRQ raised on disabled inputs		68
[GTM_AI.517] (A)TOM: Missing edge on output signal (A)TOM_OUT		69
[GTM_AI.522] (A)TOM: Edge at output signal (A)TOM_OUT does not occur	changed	70
[GTM_AI.527] GTM-ARCH: CPU bus access is not acknowledged		70
[GTM_AI.538] RTL-TIM: Wrong functionality of the overflow bit TIM[i]_CH[x]_IRQ_NOTIFY.CNTOFL	new	71
[GTM_AI.539] RTL-TIM: Unexpected overflow bit TIM[i]_CH[x]_IRQ_NOTIFY.CNTOFL	new	71
[GTM_AI.541] GTM-ARCH: Bit-field GTM_RST.BRIDGE_MODE_WRDIS not writable	new	72
[GTM_TC.020] Debug/Normal read access control via bit-field ODA.DRAC		72
[GTM_TC.026] Table “GTM IP Application Constraints” #1 (DPLL) - Documentation correction		73
[GTM_TC.031]Connections of ADC_TRIG4 signals - Correction in TC3xx appendix		73
[GTM_TC.033] Confirmation about 3rd party IPs RWH register type		74
[INT_TC.003] Wrong number of SRB groups specified		82
[MCMCAN_AI.015] Edge filtering causes mis-synchronization when falling edge at Rx input pin coincides with end of integration phase		83
[MCMCAN_AI.017] Retransmission in DAR mode due to lost arbitration at the first two identifier bits		84
[MCMCAN_AI.018] Tx FIFO message sequence inversion		85
[MCMCAN_AI.019] Unexpected High Priority Message (HPM) interrupt		86
[MCMCAN_AI.022] Message order inversion when transmitting from dedicated Tx Buffers configured with same Message ID		87
[MCMCAN_AI.023] Incomplete description in section “Dedicated Tx Buffers” and “Tx Queue” of the M_CAN documentation in the user manual related to transmission from multiple buffers configured with the same Message ID		88

**(table continues...)**

**1 Errata overview**

**Table 2 (continued) Functional deviations**

Issue title	Change	Page
[MCMCAN_AI.024] Frame transmitted despite confirmed transmit cancellation		89
[MCMCAN_AI.025] Sporadic data corruption (payload) in case acceptance filtering is not finished before reception of data R3 (DB7..DB4) is completed		90
[MCMCAN_TC.006] MCMCAN specific access protection mechanisms		92
[MCMCAN_TC.007] Incorrect access condition of bit-fields in the user manual		93
[MEMMAP_TC.001] Size of PFLASH and DFLASH - Correction to TC33xEXT and TC33x/TC32x Appendix		93
[MTU_TC.012] Security of CPU cache memories during runtime is limited		94
[MTU_TC.017] Unexpected alarms after application reset		94
[MTU_TC.018] Gated SRAM alarms		95
[PADS_TC.011] Pull-ups activate on specific analog inputs upon PORST		96
[PADS_TC.013] Buffer type definition for P21.2: no ES functionality - Data Sheet documentation correction		96
[PADS_TC.016] Pull-ups active on P33 and P34 pins in standby mode when SCR is disabled and VEXT not supplied		96
[PER_PLL_TC.002] Peripheral PLL K3 Divider Operation		97
[PMS_TC.005] Voltage rise at P33 and P34 up to $V_{EVRSB}$ during start-up and up to $V_{LVDRSTSB}$ during power-down		98
[PMS_TC.006] PORST not released during cold power-on reset until $V_{DDM}$ is available		98
[PMS_TC.007] VDDP3 or VDD Overvoltage during start-up may not be detected by PBIST		99
[PMS_TC.011] VEXT supplied PU2 and PD2 pads always in tristate after standby entry - Documentation correction		99
[PMS_TC.014] Parasitic coupling on shared ADC pins depending on supply voltages		100
[PMS_TC.015] EVRC synchronization – Documentation update for register EVRSDCTRL11 (PMS) and EVRSDCTRL2 (PMSLE)		101
[PORTS_TC.009] PCSR register incompletely documenting use for EVADC PDD and MD feature - Update to TC33x/TC32x appendix		101
[QSPI_TC.006] Baud rate error detection in slave mode (error indication in current frame)		102
[QSPI_TC.009] USR Events for PT1=2 (SOF: Start of Frame)		102
[QSPI_TC.010] Move Counter Mode - USR Events for PT1=4 (RBF: Receive Buffer Filled)		103
[QSPI_TC.013] Slave: No RxFIFO write after transmission upon change of BACON.MSB		103
[QSPI_TC.014] Slave: Incorrect parity bit upon TxFIFO underflow		103
[QSPI_TC.016] Master: Move Counter Mode - Counter underflows when data is present in the TxFIFO while in the last TRAIL state of the previous transaction		103
[QSPI_TC.017] Slave: Reset when receiving an unexpected number of bits		104

**(table continues...)**

**1 Errata overview**

**Table 2 (continued) Functional deviations**

Issue title	Change	Page
[SAFETY_TC.023] MCU infrastructure Safety Related Function - Documentation update		104
[SAFETY_TC.024] Clock alive monitor for $f_{SPB}$ - Documentation update		104
[SAFETY_TC.025] Wrong alarm listed in safety mechanism SM[HW]:SRI:SRI_TRANSACTION_INTEGRITY		105
[SAFETY_TC.026] Alarm for SM[HW]:IR:CFG_MONITOR - Documentation update		105
[SAFETY_TC.027] Single point fault detection for lockstep CPUs - Documentation update		105
[SAFETY_TC.029] Allow software writes to the OLDA address range in a safe system		106
[SCR_TC.015] Bit SCU_PMCON1.WCAN_DIS does not disable WCAN PCLK input		106
[SCR_TC.016] DUT response to first telegram has incorrect C_START value		106
[SCR_TC.018] SSC Receive FIFO not working		107
[SCR_TC.019] Accessing the XRAM while SCR is in reset state		107
[SCR_TC.020] Stored address in mon_RETH may be wrong after a break event		107
[SCR_TC.021] RTC not counting after reset if P33.10 is high		108
[SCR_TC.022] Effect of application or system reset and warm PORST on MC77_ECCD and MC78_ECCD for SCR RAMs		108
[SCR_TC.023] External interrupts EXINT0, EXINT1 may get locked		108
[SCR_TC.024] Field ADRES in register ADCOMP_RES - Documentation correction		109
[SCR_TC.033] [IR] External Interrupts 0 and 1 are not able to exit the Idle Mode of XC800 core		109
[SCU_TC.031] Bits SCU_STSTAT.HWCFGx (x=1-5) could have an unexpected value in application if pins HWCFGx are left unconnected		110
[SCU_TC.033] $\overline{\text{TESTMODE}}$ pin shall be held at static level during LBIST		110
[SCU_TC.036] Concurrent reset requests from CERBERUS do not result in all reset requests captured in reset status register		111
[SMU_TC.012] Unexpected alarms when registers FSP or RTC are written		112
[SMU_TC.013] Unexpected setting of Alarm Missed Event bit xAEM in Alarm Executed Status register SMU_AEX		113
[SMU_TC.015] SMU alarm emulation might trigger unwanted active alarm reaction		113
[SMU_TC.017] Alarm type indication (level or pulse) for SBCU SPB and EBCU BBB bus error event		114

**Table 3 Parametric deviations**

Issue title	Change	Page
[ADC_TC.P017] Increased RMS noise for TC33x/32x devices		115
[CCU_TC.P001] Back-up clock accuracy after trimming - Disregard datasheet footnote		115

**(table continues...)**

**1 Errata overview**

**Table 3 (continued) Parametric deviations**

Issue title	Change	Page
[FLASH_TC.P003] Program Flash Erase Time per Multi-Sector Command		115
[PADS_TC.P014] Electrical characteristics for P20.2/ $\overline{\text{TESTMODE}}$		116
[PORST_TC.P002] $V_{IH}$ and $V_{IL}$ definition for PORST pad - Additional Data Sheet footnote		116
[PWR_TC.P015] Power pattern definition - Documentation update to TC33x/TC32x Data Sheet V1.1		116
[VEVRSB_TC.P001] Bonding of VEVRSB pad on LQFP packages - Data Sheet documentation correction		117

**Table 4 Application hints**

Issue title	Change	Page
[ADC_TC.H026] Additional waiting phase in slow standby mode		118
[ADC_TC.H032] ADC accuracy parameters - Definition		118
[ADC_TC.H033] Basic initialization sequence for primary and secondary EVADC groups		118
[ADC_TC.H035] Effect of input leakage current on Broken Wire Detection		119
[ADC_TC.H040] Selection of masters for synchronization groups - Documentation update to TC33x/TC32x Appendix		120
[ADC_TC.H043] Information on supervision signal $V_{ANACOMM}$ not relevant - Documentation update		121
[ADC_TC.H044] Start-up calibration timing in synchronized mode - Documentation update		121
[ADC_TC.H045] Level selection for broken wire detection feature		121
[ADC_TC.H046] Incorrect number of EVADC kernels in TC33x/TC32x Datasheet		122
[ADC_TC.H048] EVADC sampling time setting below 300 ns lead to $V_{DDK}$ signal conversion inaccuracy		122
[ASCLIN_TC.H001] Bit field FRAMECON.IDLE in LIN slave tasks		123
[ASCLIN_TC.H006] Sample point position when using three samples per bit		123
[ASCLIN_TC.H007] Handling Tx FIFO and Rx FIFO interrupts in single move mode		123
[ASCLIN_TC.H008] SPI master timing – Additional information to Data Sheet characteristics		124
[ASCLIN_TC.H012] Unexpected collision detection flag raised when soft suspend request is raised and ACK has not arrived		124
[BROM_TC.H009] Re-enabling lockstep via BMHD		125
[BROM_TC.H014] SSW behavior in case of wrong state or uncorrectable error in UCBs - Documentation Update		125
[BROM_TC.H020] Processing in case no valid BMHD found		125
[CCU6_TC.H001] CCU6 module clock source information - Documentation Update		126

**(table continues...)**

**1 Errata overview**

**Table 4 (continued) Application hints**

Issue title	Change	Page
[CCU_TC.H012] Configuration of the Oscillator- Documentation Update		126
[CLC_TC.H001] Description alignment for bits DISR, DISS, EDIS in register CLC - Documentation Update		126
[CPU_TC.H022] Store buffering and the effect of bit SMACON.IODT		127
[CPU_TC.H023] CPU_SYSCON register safety protection description clarification		127
[CPU_TC.H024] Usage of atomic instructions SWAPMSK.W and LDMST to access registers with bit-fields that can also be updated by hardware (rwh)		128
[CPU_TC.H025] Avoiding unbounded delays in store buffer residency		129
[CPU_TC.H027] Performance impact depending on the enabling of the Core MPU protection	new	140
[DMA_TC.H018] Maximum size of circular buffers is 32 Kbytes		141
[DTS_TC.H002] Unexpected alarms after start-up/wake-up when temperature is close to lower/upper limit		142
[EVR_TC.H001] External input capacitor value - Additional Data Sheet footnote		142
[FLASH_TC.H021] Flash Wait State configuration		142
[FLASH_TC.H024] PFLASH erase and program time is affected by time slicing but not clearly documented		143
[FLASH_TC.H026] Additional information about Test Pass Marker		143
[FlexRay_AI.H004] Only the first message can be received in External Loop Back mode		143
[FlexRay_AI.H005] Initialization of internal RAMs requires one eray_bclk cycle more		144
[FlexRay_AI.H006] Transmission in ATM/Loopback mode		144
[FlexRay_AI.H007] Reporting of coding errors via TEST1.CERA/B		144
[FlexRay_AI.H009] Return from test mode operation		144
[FlexRay_AI.H010] Driver software must launch CLEAR_RAMs command before reading from E-Ray RAMs		145
[FlexRay_AI.H011] Behavior of interrupt flags in FlexRay™ Protocol Controller (E-Ray)		145
[FlexRay_TC.H003] Initialization of E-Ray RAMs - Documentation update		146
[FlexRay_TC.H004] Bit WRECC in register TEST2 has no function		146
[FlexRay_TC.H005] E-Ray OTGB2 trigger set active even if disabled		146
[FPI_TC.H003] Burst write access may lead to data corruption and to a stalling issue		147
[GPT12_TC.H002] Bits TxUD and TxUDE in incremental interface mode - Additional information		147
[GTM_AI.H480] SPEC-TIM: Wrong action description for TPIM mode		147
[GTM_AI.H481] SPEC-TIM: Wrong description for TBCM mode		148
[GTM_AI.H482] SPEC-TIM: Wrong description in TBCM mode regarding TIM[i]_CH[x]_CTRL.GPR1_SEL bit field		149

**(table continues...)**

**1 Errata overview**

**Table 4 (continued) Application hints**

Issue title	Change	Page
[GTM_AI.H497] SPEC-SPE wiring in figure is wrong		150
[GTM_AI.H512] SPEC-SPE: Wrong signal names		150
[GTM_AI.H519] SPEC-(A)TOM: Misleading description of Continuous Counting Up Mode		150
[GTM_AI.H520] SPEC-(A)TOM: Description for 100% duty cycle incorrect		151
[GTM_AI.H521] SPEC-ATOM: Missing information for SOMB mode		152
[GTM_AI.H525] SPEC-(A)TOM: Description for 0% duty cycle incorrect		152
[GTM_AI.H526] SPEC-(A)TOM: Missing information for SOMP mode	changed	153
[GTM_AI.H528] Spec-(A)TOM: Missing priority information		153
[GTM_AI.H540] SPEC-TIM: Wrong update condition in TSSM mode	new	154
[GTM_AI.H543] SPEC-TIM: Wrong clock name referenced	new	154
[GTM_AI.H803] SPEC-(A)TOM: Missing priority information for register update	changed	154
[GTM_TC.H010] Trigger Selection for EVADC and EDSADC		155
[GTM_TC.H019] Register GTM_RST - Documentation Update		155
[GTM_TC.H021] Interrupt strategy mode selection in IRQ_MODE		156
[GTM_TC.H027] Register ODA (OCDS Debug Access) - Documentation update		157
[GTM_TC.H035] Type of bit-fields for xxx_IRQ_NOTIFY registers should be marked as rwh		157
[INT_TC.H006] Number of SRNs supporting external interrupt/service requests – Documentation update		158
[INT_TC.H007] Interrupt router SRC_xxx register is not always read with correct value		158
[ISTANDBY_TC.H001] Characteristics of standby current $I_{STANDBY}$ on TC33x/TC32x in QFP-80 and QFP-100 packages		159
[LBIST_TC.H003] Update reset behavior of LBISTCTRL0 and LBISTCTRL3 register - Additional information		159
[LBIST_TC.H005] Effects of LBIST execution on P33.8		159
[MBIST_TC.H001] Destructive MBIST requires DSPR0 initialization		160
[MBIST_TC.H002] Time for 4N non-destructive test		160
[MCMCAN_AI.H001] Behavior of interrupt flags in CAN Interface (MCMCAN)		160
[MCMCAN_AI.H002] Bus off recovery		161
[MCMCAN_TC.H001] Behavior of undefined data bytes read from Receive Buffer		162
[MCMCAN_TC.H006] Unintended behavior of receive timeout interrupt		162
[MCMCAN_TC.H007] Delayed time triggered transmission of frames		162
[MCMCAN_TC.H008] Parameter “CAN Frequency” - Documentation update to symbol in Data Sheet		163

**(table continues...)**

**1 Errata overview**

**Table 4 (continued) Application hints**

<b>Issue title</b>	<b>Change</b>	<b>Page</b>
[MTU_TC.H015] ALM7[0] may be triggered after cold PORST		163
[MTU_TC.H016] MCI_FAULTSTS.OPERR[2] may be triggered at power-up in case LBIST is not run		163
[MTU_TC.H019] Application reset value of register SRC_MTUDONE different to documentation		164
[MTU_TC.H020] SIQ 53 - ALM7[1] unexpectedly raised after an application reset		164
[NVM_TC.H001] References to DMU_HP_PROCONTTP – Typo in TC3xx user manual		164
[OCDS_TC.H014] Avoiding failure of key exchange command due to overwrite of COMDATA by firmware		165
[OCDS_TC.H015] System or Application Reset while OCDS and lockstep monitoring are enabled		165
[OCDS_TC.H016] Release of application reset via OJCONF may fail		166
[OCDS_TC.H018] Unexpected stop of Startup Software after system or application reset		166
[OSC_TC.H002] Split the external crystal mode and the external input clock mode parameters of MHz oscillator in the TC3xx datasheet		166
[OSC_TC.H003] XTAL1 SR parameter change for compatibility with BGT60ATR24AIP	new	167
[PACKAGE_TC.H003] Exposed pad dimensions and package outlines for QFP packages - Updates to TC33x/TC32x Data Sheet		168
[PADS_TC.H007] Connection of HWCFG[6] pad in QFP-80 and QFP-100 packages – Explanation to Data Sheet history		170
[PMSLE_TC.H001] Typo in SHVL formula for EVRSDCTRL9 in the PMSLE chapter		170
[PMSLE_TC.H002] VSSM/VAGND is not connected to ePad on TQFP-100 or TQFP-80		170
[PMS_TC.H003] $V_{DDPD}$ voltage monitoring limits		171
[PMS_TC.H008] Interaction of interrupt and power management system - Additional information		171
[PMS_TC.H009] Interaction of warm reset and standby mode transitions		173
[PMS_TC.H010] “EVR13” to be replaced by “EVRC” in table titles of TC33x/TC32x Data Sheet - Documentation update		173
[PMS_TC.H011] Supply mode and topology selection - Allowed combinations of VEXT and VDDM - Documentation update		173
[PMS_TC.H015] Update of figures showing VCAP0 and VCAP1 pin labels to correct polarity for EVRC SC-DCDC flying capacitor		174
[PMS_TC.H018] Bit SWDLVL in register EVRSTAT is always 1 when EVRC is OFF		176
[PMS_TC.H019] Limitation of power-cycles - Additional datasheet footnote		176
[PORTS_TC.H018] Misleading footnote on pad driver mode selection table		176

**(table continues...)**

**1 Errata overview**

**Table 4 (continued) Application hints**

Issue title	Change	Page
[QSPI_TC.H008] Details of the baud rate and phase duration control - Documentation update		177
[QSPI_TC.H011] Missing information on SLSI misplaced inactivation enable error		177
[QSPI_TC.H013] Additional parameter value for CMOS and LVDS pads of QSPI module in the datasheet		177
[RESET_TC.H006] Certain registers may have different reset values than documented in TC3xx User's Manual - Documentation update		178
[RESET_TC.H007] Cold Power on Reset Boot Time – Additional information		180
[SAFETY_TC.H013] ESM[SW]:SYS:MCU_FW_CHECK - Access to MC40 FAULTSTS register – Additional information		181
[SAFETY_TC.H017] Safety Mechanisms requiring initialization - Documentation update		181
[SAFETY_TC.H019] SM[HW]:NVM.FSIRAM:REG_MONITOR_TEST should not be considered		183
[SAFETY_TC.H020] Test of SM[HW]:VMT:REG_MONITOR is missing - Documentation update		184
[SCR_TC.H009] RAM ECC Alarms in Standby Mode		184
[SCR_TC.H010] HRESET command erroneously sets RRF flag		184
[SCR_TC.H011] Hang-up when warm PORST is activated during Debug Monitor Mode		184
[SCR_TC.H012] Reaction in case of XRAM ECC Error		185
[SCR_TC.H014] Details on WDT pre-warning period		185
[SCR_TC.H016] SCR current consumption in IDLE mode and 70 kHz clock		185
[SCU_TC.H020] Digital filter on ESRx pins - Documentation update		186
[SCU_TC.H021] LBIST execution affected by TCK/DAP0 state		186
[SCU_TC.H023] Behavior of bit RSTSTAT.PORST after wake-up from standby mode		186
[SCU_TC.H025] Field EEA in register CHIPID - Additional information		186
[SCU_TC.H026] Unexpected alarm ALM0[1] during warm reset		187
[SCU_TC.H027] Bit field INP0 and INP1 in register EICRi - Documentation correction		187
[SCU_TC.H028] ERU configuration changes may lead to ERU reactions		188
[SENT_TC.H006] Parameter $V_{ILD}$ on pads used as SENT inputs		189
[SENT_TC.H007] Range for divider value DIV - Documentation correction		195
[SENT_TC.H009] Unexpected NNI error behavior		195
[SMU_TC.H010] Clearing individual SMU flags: use only 32-bit writes		196
[SMU_TC.H012] Handling of SMU alarms ALM7[1] and ALM7[0]		196
[SMU_TC.H013] Increased Fault Detection for SMU Bus Interface (SMU_CLC Register)		197
[SMU_TC.H016] SMU_stdby restriction for using P33.8 as Emergency Stop input		197

**(table continues...)**

**1 Errata overview****Table 4 (continued) Application hints**

Issue title	Change	Page
[SMU_TC.H017] Handling of ALM21[7] when safety flip-flop self-test is executed		197
[SMU_TC.H020] There is no GTM SSH instances for the alarms ALM6[12:10]		198
[SRI_TC.H001] Using LDMST and SWAPMSK.W instructions on SRI mapped peripheral registers (range 0xF800 0000-0xFFFF FFFF)		198
[SRI_TC.H003] Incorrect information in SRI error capture registers for HSM transactions		199
[SRI_TC.H005] Clarification of effects for setting PECONx.SETPE		199
[SSW_TC.H001] Security hardening measure for the startup behavior		200
[STM_TC.H004] Access to STM registers while STMDIV = 0		201

## 2 Functional deviations

## 2 Functional deviations

### 2.1 [BROM\_TC.013] CAN BSL does not send error message if no valid baudrate is detected

#### Description

If the CAN Bootstrap loader (BSL) is unable to determine the baudrate from the initialization message sent by the host, it does not send the error message as defined in table “Error message (No baudrate detected)” in chapter “AURIX™ TC3xx Platform Firmware” in TC3xx User's Manual, but enters an endless loop with no activity on external pins.

#### Workaround

If the external host does not receive Acknowledgment Message 1 from the CAN BSL within the expected time (~5 ms), it should check the integrity of the connection, and then may reset the TC3xx to restart the boot procedure.

### 2.2 [BROM\_TC.014] Lockstep comparator alarm for CPU0 after warm PORST, system or application reset if lockstep is disabled

#### Description

Lockstep monitoring may be disabled in the Boot Mode Header structure (BMHD) for each CPU<sub>x</sub> with lockstep functionality (including CPU0). The startup software (SSW) will initially re-enable lockstep upon the next reset trigger.

If lockstep is disabled for CPU0, and the next reset is a warm PORST, system or application reset, a lockstep comparator alarm will be raised for CPU0.

**Note:** *This effect does not occur for CPU<sub>x</sub>, x>0.*

#### Workaround

Do not disable lockstep for CPU0, always keep lockstep on CPU0 enabled.  
Non-safety applications may ignore the lockstep comparator alarm for CPU0.

### 2.3 [BROM\_TC.016] Uncorrectable ECC error in Boot Mode Headers

#### Description

If one or more boot mode headers UCB\_BMHD<sub>x</sub>\_ORIG or UCB\_BMHD<sub>x</sub>\_COPY contain an uncorrectable ECC error (4-bit error) in the BMI, BMHDID, STAD, CRCBMHD or CRCBMHD\_N fields, firmware will end up in an irrecoverable state resulting in a device not being able to boot anymore.

This may happen in the following scenarios:

- Power-loss during BMHD reprogramming or erase
- Over-programming of complete BMHD contents

#### Workaround

- Ensure continuous power-supply during BMHD reprogramming and erase using power monitoring including appropriate configuration
- Avoid over-programming of BMHD contents
- Ensure that also in any BMHD<sub>x</sub>\_ORIG or \_COPY unused in the application, the above fields are in a defined ECC-error free state (for example clear them to 0)

## 2 Functional deviations

### 2.4 [CCU\_TC.005] ASC and CAN bootstrap loaders may not work if external clock is missing

#### Description

When using the ASC or CAN bootstrap loader (BSL) with internal clocking ( $f_{\text{BACK}}$ ), and no supply noise or other source of signal level transition is present on the XTAL1 input during device power-up, the device does not respond to the zero byte (ASC BSL) or initialization frame (CAN BSL).

#### Effects

No code download for initial device programming is started.

**Note:** *This problem may only occur for initial start-up of unprogrammed devices. For TC3xx, if automatic start of the external crystal oscillation is programmed in UCB DFLASH, the problem will not occur.*

#### Workaround

Trigger reset and retry if bootstrap loader does not respond.

If connection to the device is possible through a debug tool, use the tool to reconfigure OSCCON.MODE = 00<sub>B</sub> (when using an external crystal), and then trigger reset.

### 2.5 [CPU\_TC.130] Data Corruption when ST.B to local DSPR coincides with external access to same address

#### Description

Under certain conditions, when a CPU accesses its local DSPR using “store byte” (ST.B) instructions, at the same time as stores from another bus master (such as remote CPU or DMA for example) to addresses containing the same byte, the result is the corruption of data in the adjacent byte in the same halfword.

All the following conditions must be met for the issue to be triggered:

- CPU A executes a ST.B targeting its local DSPR
- Remote bus master performs a write of 16-bit or greater targeting CPU A DSPR
- Both internal and external accesses target the same byte without synchronization

**Note:** *Although single 8-bit write accesses by the remote bus master do not trigger the problem, 16-bit bus writes from a remote CPU could occur from a sequence of two 8-bit writes merged by the store buffers into one 16-bit access.*

When the above conditions occur, the value written by the external master to the adjacent byte (to that written by CPU A) is lost, and the prior value is retained.

#### Workaround 1

Ensure mutually exclusive accesses to the memory location. A semaphore or mutex can be put in place in order to ensure that Core A and other bus masters have exclusive access to the targeted DSPR location.

#### Workaround 2

When sharing objects without synchronization between multiple cores, use objects of at least halfword in size.

#### Workaround 3

When two objects, being shared without synchronization between multiple cores, are of byte granularity, locate these objects in a memory which is not a local DSPR to either of the masters (LMU, PSPR, or other DSPR for example).

## 2 Functional deviations

### 2.6 [CPU\_TC.131] Performance issue when MADD or MSUB instructions use E0 or D0 register as accumulator

#### Description

**Note:** Consider the following notes for TC26x, TC27x, TC29x:

- **TC26x:** In TC26x devices, this problem only affects the TC1.6P processor (CPU1). The TC1.6E processor (CPU0) is not affected by this problem.
- **TC27x:** In TC27x devices, this problem only affects the TC1.6P processors (CPU1 and CPU2). The TC1.6E processor (CPU0) is not affected by this problem.
- **TC29x:** In TC29x devices, this problem affects the TC1.6P processors (CPU0, CPU1, and CPU2).

Under certain conditions, when a Multiply (MULx.y) or Multiply-Accumulate (MAC) instruction is followed by a MAC instruction which uses the result of the first instruction as its accumulator input, a performance reduction may occur if the accumulator uses the E0 or D0 register. The accumulator input is that to which the multiplication result is added to (in the case of MADDx.y), or subtracted from (in the case MSUBx.y), in a MAC instruction.

All MAC instructions MADDx.y, MSUBx.y are affected except those that operate on Floating-Point operands (MADD.F, MSUB.F).

The problem occurs where there is a single cycle bubble, or an instruction not writing a result, between these dependent instructions in the Integer Pipeline (IP). When this problem occurs the dependent MAC instruction will take 1 additional cycle to complete execution. If this sequence is in a loop, the additional cycle will be added to every iteration of the loop.

#### Example

```
maddm.h e0, e0, d3, d5u1 ; MUL/MAC writing E0 as result
ld.d e8, [a5] ; Load instruction causing IP bubble
maddm.h e0, e0, d6, d8u1 ; MAC using E0 as accumulator.
                        ; Should be delayed by 1 cycle due to
                        ; dependency to result of previous LD.D,
                        ; but is delayed for 2 cycles
```

**Note:** If there are 2 or more IP instructions, or a single IP instruction writing a result, between the MAC and the previous MUL/MAC, then this issue does not occur.

#### Workaround

Since the issue only affects D0 or E0, it is recommended that to ensure the best performance of an affected sequence as the above example, D0 or E0 is replaced with another register (D1-D15 or E2-E14).

### 2.7 [CPU\_TC.132] Unexpected PSW values used upon Fast Interrupt entry

#### Description

Under certain conditions, unexpected PSW values may be used during the first instructions of an interrupt handler, if the interrupt has been taken as a fast interrupt. For a description of fast interrupts, see the “CPU Implementation-Specific Features” section of the relevant User’s Manual.

## 2 Functional deviations

When the problem occurs, the first instructions of the interrupt handler may be executed using the PSW state from the end of the previous exception handler, rather than that which is being loaded by the fast interrupt entry sequence. The TC1.6E, TC1.6P and TC1.6.2P processors are all affected by this problem as follows:

- TC1.6E (in TC21x..TC27x):
  - Only the first instruction of the ISR is affected
- TC1.6P (in TC26x..TC29x), TC1.6.2P (in TC3xx):
  - Up to 4 instructions at the start of the ISR may be affected.
  - However, if the following pre-condition is not met, then there is no issue for these processor variants: A11 must point to the first instruction of the fast interrupt handler at the end of the previous exception handler, i.e. the return value from the previous exception must be pointing to the very first instruction of the new interrupt handler. Note that this case should not occur normally, unless software updates the A11 register to a value corresponding to the start of an interrupt handler

### Workaround 1

When the PSW fields PSW.PRS, PSW.S, PSW.IO or PSW.GW need to be changed in an exception handler, the change should be wrapped in a function call.

```

_exception_handler:
    CALL _common_handler
    RFE

_common_handler:
    MOV.U d0, #0x0380
    MTCR #(PSW), d0 // PSW.IO updated to User-0 mode
    ...
    RET

```

Note that this workaround assumes SYSCON.TS == SYSCON.IS such that the workaround functions correctly for both traps and interrupts. If this is not the case it is possible for bus accesses to use an incorrect master Tag ID, potentially resulting in an access to be incorrectly allowed, or an unexpected alarm to be generated. In this case it should be ensured that for all interrupt handlers the potentially affected instructions do not produce bus accesses.

### Workaround 2

Do not use any instructions dependent upon PSW settings (for example BISR or ENABLE, dependent on PSW.IO) as the first instruction of an ISR in TC1.6E, or as one of the first 4 instructions in an ISR for TC1.6P or TC1.6.2P.

**Note:** *The workarounds need to be applied in TC1.6P and TC1.6.2P only in case software modifies the A11 register in an exception handler, as described in the pre-conditions above.*

## 2.8 [CPU\_TC.133] Test sequence for DTAG single or double bit errors

### Description

The error injection method described in the section “Error injection and Alarm Triggering” in the MTU chapter of the TC3xx User’s Manual using the ECCMAP method is not sufficient to trigger alarms pertaining to the DTAG RAM of each CPU. In the case of DTAG RAM, an alternate method relying on the Read Data and Bit Flip register (RDBFL) must be used instead.

When using the ECCMAP, the DTAG ECC error detection is disabled when the DTAG memory is mapped in the system address map.

This limitation only affects the testing using ECCMAP for DTAG RAM.

## 2 Functional deviations

During normal operation, where DTAG is used as part of the CPU data cache operation, the ECC error detection functions as intended.

During SSH test mode (used for MBIST) the ECC error detection also operates as intended.

### Workaround

A correct test sequence for DTAG single and double bit error injection must therefore use the RDBFL register without mapping the RAM to the system address space.

### DTAG SRAM test sequence

In order to test the DTAG error injection the following test sequence should be followed:

1. Read an DTAG SRAM location into RDBFL register (see section “Reading a Single Memory Location”)
2. Flip some bit in RDBFL[0]
3. Writeback the content of the RDBFL into the DTAG SRAM (see section “Writing a Single Memory Location”)
4. Read the DTAG SRAM location again

Depending on the number of bits flipped the CE or UCE alarms will be triggered.

## 2.9 [DAP\_TC.005] DAP client\_read: dirty bit feature of Cerberus’ Triggered Transfer Mode

### Description

**Note:** *This problem is only relevant for tool development, not for application development.*

The DAP telegram client\_read reads a certain number of bits from an IOclient (for example Cerberus). The parameter k can be selected to be zero, which is supposed to activate reading of 32 bits plus dirty bit. However, in the current implementation, the dirty bit feature does not work correctly.

It is recommended not to use this dirty bit feature, meaning the number k should not evaluate to “0”.

## 2.10 [DAP\_TC.007] Incomplete client\_blockread telegram in DXCM mode when using the “read CRCup” option

### Description

In DXCM (DAP over CAN Messages) mode, the last parcel containing the CRC32 might be skipped in a client\_blockread telegram using the “read CRCup” option.

### Workaround

Do not use CRCup option with client\_blockread telegrams in DXCM mode.

Instead the CRCup can be read by a dedicated getCRCup telegram.

## 2.11 [DMA\_TC.066] DMA double buffering operations - Update address pointer

### Description

Software may configure a DMA channel for one of the DMA double buffering operations:

- DMA Double Source Buffering Software Switch Only
  - (DMA channel DMA\_ADICRz.SHCT = 1000<sub>B</sub>)

## 2 Functional deviations

- DMA Double Source Buffering Automatic Hardware and Software Switch
  - (DMA channel DMA\_ADICRz.SHCT = 1001<sub>B</sub>)
- DMA Double Destination Buffering Software Switch Only
  - (DMA channel DMA\_ADICRz.SHCT = 1010<sub>B</sub>)
- DMA Double Destination Buffering Automatic Hardware and Software Switch
  - (DMA channel DMA\_ADICRz.SHCT = 1011<sub>B</sub>)

If the software updates a buffer address pointer by BYTE or HALF-WORD writes, the resulting value of the address pointer is corrupted.

### Workaround

If the software updates a buffer address pointer, the software should only use a 32-bit WORD access.

### 2.12 [DMA\_TC.067] DMA Double Buffering Software Switch buffer overflow

#### Description

If a DMA channel is configured for DMA Double Buffering Software Switch Only and the active buffer is emptied or filled, the DMA does not stop. A bug results in the DMA evaluating the state of the FROZEN bit (DMA channel CHCSR.FROZEN). If the FROZEN bit is not set, the DMA continues to service DMA requests in the current buffer. The DMA may perform DMA write moves outside of the address range of the buffer potentially trashing other data.

#### Workaround

Implement one or more of the following to minimize the impact of the bug:

1. Configure access protection across the whole memory map to prevent the trashing data by the DMA channel configured for DMA double buffering. A DMA resource partition may be used to assign a unique master tag identifier to the DMA channel
2. The address generation of the DMA channel configured for DMA double buffering should use a circular buffer aligned to the size of the buffer to prevent the DMA from writing outside the address range of the buffer

### 2.13 [DMA\_TC.068] DMA Double Buffering lost DMA request

#### Description

If a DMA channel is configured for DMA Double Buffering and a buffer switch is performed, no DMA requests shall be lost by the DMA and there shall be no loss, duplication or split of data across two buffers.

A bug results in a software switch clearing a pending DMA request. As a result a DMA transfer is lost without the recording of a TRL event so violating the aforementioned top-level requirements of DMA double buffering.

#### Workaround

The system must ensure that a software switch does not collide with a DMA request. A user program must execute the following steps to switch the buffer:

1. Software must disable the servicing of interrupt service requests by the DMA channel by disabling the corresponding Interrupt Router (IR) Service Request Node (SRN)
  - a. Software shall write IR\_SRCi.SRE = 0<sub>B</sub>
2. Software must halt the DMA channel configured for DMA double buffering
  - a. Software shall write DMA channel TSRc.HLTREQ = 1<sub>B</sub>
  - b. Software shall monitor DMA channel TSRc.HLTACK = 1<sub>B</sub>

## 2 Functional deviations

3. Software must monitor the DMA Channel Transaction Request State
  - a. Software shall read DMA channel TSRc.CH and store the value in a variable SAVED\_CH
4. Software must switch the source or destination buffer
  - a. Software shall write DMA channel CHCSRc.SWB = 1<sub>B</sub>
  - b. Software shall monitor the DMA channel frozen bit CHCSRc.FROZEN
5. When the DMA channel has switched buffers (DMA channel CHCSRc.FROZEN = 1<sub>B</sub>)
  - a. If (SAVED\_CH==1), software shall trigger a DMA software request by writing DMA channel CHCSRc.SCH = 1<sub>B</sub> to restore DMA channel TSRc.CH to the state before the buffer switch
6. Software must un halt the DMA channel
  - a. Software shall write DMA channel TSRc.HLTCLR = 1<sub>B</sub>
7. Software must enable the servicing of interrupt service requests by the DMA channel
  - a. Software shall write IR\_SRCi.SRE = 1<sub>B</sub>

The software must include an error routine.

1. Software must monitor for interrupt overflows (IR\_SRCi.IOV = 1<sub>B</sub>) and lost DMA requests (TSRc.TRL = 1<sub>B</sub>)
2. If software detects an overflow or lost DMA request, the software must execute an error routine and take the appropriate reaction consistent with the application

### 2.14 [DMA\_TC.071] Daisy Chain request is lost when repeat triggers too soon

#### Description

When a DMA channel X is configured for DMA daisy chain request, then when it completes a DMA transaction, it should initiate a DMA transaction on the next lower priority DMA channel X-1 by setting the access pending bit TSRc.CH[X-1].

If a DMA channel N receives a new DMA request or DMA daisy chain request before the completion of an ongoing DMA transaction, the DMA channel N will win the next arbitration slot and may complete its next DMA transaction before the lower priority DMA channel N-1 has processed the pending DMA daisy chain request from the previous iteration. In this case, due to a bug in the DMA controller, the lower priority DMA channel N-1 only processes one DMA daisy chain request, resulting in a lost trigger and transaction. There is no error notification of this lost trigger.

#### Scope

DMA daisy chains, where DMA requests for the initial channel can arrive more frequently than the daisy chain execution time.

#### Effects

DMA transactions on all DMA channels in the DMA daisy chain are expected to complete before the initiator (start of chain) DMA channel in the DMA daisy chain receives a new DMA request. If the application triggers the DMA daisy chain again before any previous execution has completed, DMA daisy chain requests can be lost without any error notification.

#### Workaround

If the application cannot avoid new DMA requests to the initial channel of the DMA daisy chain while a previous iteration of the DMA daisy chain is still executing, the application should not use the DMA daisy chain feature.

The application should instead configure each DMA channel to generate a DMA channel interrupt service request, and further configure the IR to route the DMA channel interrupt service request to the next lower priority DMA channel. This achieves the same functionality provided by the DMA daisy chain, but with a longer trigger latency for the subsequent channels.

## 2 Functional deviations

**Note:** *If the initiator channel request rate is too frequent, the condition may still result in lost transactions but these will be captured by one of the channels `TSRc.TRL` and an error interrupt is generated if its `TSRc.ETRL` is set.*

### 2.15 [FLASH\_TC.053] Erase size limit for PFLASH

#### Description

The device may fail to start up after a primary voltage monitor triggered (cold) PORST if all of the following four conditions are fulfilled at the same time:

- Erase operation is ongoing in PFLASH, AND
- PORST is triggered by one of the primary voltage monitors, AND
- Ambient temperature  $T_A > 60^\circ\text{C}$  OR junction temperature  $T_J > 70^\circ\text{C}$ , AND
- Size of logical sectors  $> 256$  Kbyte is specified in "Erase Logical Sector Range" command

#### Workaround

If it cannot be excluded that all four conditions listed above may occur at the same time:

- Limit the maximum logical sector erase size to 256 Kbyte in the "Erase Logical Sector Range" command

### 2.16 [FLASH\_TC.055] Multi-bit errors detected by PFlash are not communicated to SPB masters

#### Description

Section "PFLASH ECC" in the NVM chapter of the TC3xx User Manual states in bullet points

- "Multi-bit error and not All-0 error" and
- "Multi-bit error and All-0 error"

that a bus error is returned to the reading master.

The same statement is repeated in section "Program Side Memories" in the CPU chapter under the headline "Local Pflash Bank (LPB)" and in the HSM Target Specification.

Effectively the processing of such errors depends on the type of transaction (burst or single) and the path the read transaction takes through the on-chip connectivity with the result that an SPB master (like HSM) gets no information about the detected error as detailed below:

When a CPU reads its local PFlash bank using direct access through its DPI (also called "Fast Path") such errors are directly translated into a PIE trap for instruction fetch and a DIE trap for data read. No bus error is generated as no bus communication is involved.

When any master reads PFlash through the SRI (this includes CPUs reading the PFlash located at another CPU or its local bank with disabled Fast Path) a single transfer with multi-bit error returns a bus error but a burst read is reporting this error using a forced "Transaction ID Error" (concept described in "On-Chip System Connectivity {and Bridges}"). The bus error is always communicated back to the master. The handling of the Transaction ID Error however is master specific.

When a CPU receives the SRI transaction ID error it handles it as bus error and triggers a PSE trap for instruction fetch and a DSE trap for data read.

Also the DMA handles the Transaction ID Error like a bus error, sets the corresponding error flags and triggers the source error interrupt request.

When an SPB master like HSM performs a burst read from a PFlash bank this SRI Transaction ID Error terminates at the SFI\_F2S bridge. The SPB master does not receive a bus error and continues operation with wrong data. The SFI\_F2S bridge signals the error to the XBar for alarm generation.

The SPB master Cerberus acting on behalf of a debug tool issues only single transfers and is therefore correctly informed by a bus-error.

## 2 Functional deviations

### Workaround

Such multi-bit errors are added to the MBAB error buffer in the PFI (documented in the NVM chapter). Filling the MBAB results in sending an alarm “Safety Mechanism: PFlash ECC; Alarm: Multiple Bit Error Detection Tracking Buffer Full” to the SMU.

As described above also SFI\_F2S bridge informs the XBar to generate an alarm “Safety Mechanism: Built-in SRI Error Detection; Alarm: XBAR0 Bus Error Event” to the SMU. With HSM as requesting master the XBAR0 just captures the occurrence of this error but doesn’t capture address or other transaction data in its Error Capture registers.

The application has to take care that the SMU alarm handler informs the SPB master.

### 2.17 [FLASH\_TC.056] Reset value for register HF\_ECCC is 0x0000 0000 - Documentation correction

#### Description

In the register description for register HF\_ECCC (DF0 ECC Control Register) in the TC3xx User’s Manual, the application reset value is documented as C000 0000<sub>H</sub>.

However, this register is cleared by the startup software SSW, and the user software will read the reset value of 0000 0000<sub>H</sub>.

#### Documentation correction

- The application reset value for register HF\_ECCC is 0000 0000<sub>H</sub>

**Note:** *The user must consider that field HF\_ECCC.TRAPDIS is 00<sub>B</sub> after reset, which means a bus error trap is generated if an uncorrectable ECC error occurs upon read from DF0, or read from DF1 when DF1 is configured as not HSM\_exclusive.*

### 2.18 [FlexRay\_AI.087] After reception of a valid sync frame followed by a valid non-sync frame in the same static slot the received sync frame may be ignored

#### Description

If in a static slot of an even cycle a valid sync frame followed by a valid non-sync frame is received, and the frame valid detection (prt\_frame\_decoded\_on\_X) of the DEC process occurs one sclk after valid frame detection of FSP process (fsp\_val\_syncfr\_chx), the sync frame is not taken into account by the CSP process (devte\_xxs\_reg).

#### Scope

The erratum is limited to the case where more than one valid frame is received in a static slot of an even cycle.

#### Effects

In the described case the sync frame is not considered by the CSP process. This may lead to a SyncCalcResult of MISSING\_TERM (error flag SFS.MRCS set). As a result the POC state may switch to NORMAL\_PASSIVE or HALT or the Start-up procedure is aborted.

#### Workaround

Avoid static slot configurations long enough to receive two valid frames.

---

## 2 Functional deviations

### 2.19 [FlexRay\_AI.088] A sequence of received WUS may generate redundant SIR.WUPA/B events

#### Description

If a sequence of wake-up symbols (WUS) is received, all separated by appropriate idle phases, a valid wake-up pattern (WUP) should be detected after every second WUS. The E-Ray detects a valid wake-up pattern after the second WUS and then after each following WUS.

#### Scope

The erratum is limited to the case where the application program frequently resets the appropriate SIR.WUPA/B bits.

#### Effects

In the described case there are more SIR.WUPA/B events seen than expected.

#### Workaround

Ignore redundant SIR.WUPA/B events.

### 2.20 [FlexRay\_AI.089] Rate correction set to zero in case of SyncCalcResult=MISSING\_TERM

#### Description

In case a node receives too few sync frames for rate correction calculation and signals a SyncCalcResult of MISSING\_TERM, the rate correction value is set to zero instead of to the last calculated value.

#### Scope

The erratum is limited to the case of receiving too few sync frames for rate correction calculation (SyncCalcResult=MISSING\_TERM in an odd cycle).

#### Effects

In the described case a rate correction value of zero is applied in NORMAL\_ACTIVE / NORMAL\_PASSIVE state instead of the last rate correction value calculated in NORMAL\_ACTIVE state. This may lead to a desynchronisation of the node although it may stay in NORMAL\_ACTIVE state (depending on gMaxWithoutClockCorrectionPassive) and decreases the probability to re-enter NORMAL\_ACTIVE state if it has switched to NORMAL\_PASSIVE (pAllowHaltDueToClock=false).

#### Workaround

It is recommended to set gMaxWithoutClockCorrectionPassive to 1. If missing sync frames cause the node to enter NORMAL\_PASSIVE state, use higher level application software to leave this state and to initiate a re-integration into the cluster. HALT state can also be used instead of NORMAL\_PASSIVE state by setting pAllowHaltDueToClock to true.

### 2.21 [FlexRay\_AI.090] Flag SFS.MRCS is set erroneously although at least one valid sync frame pair is received

#### Description

If in an odd cycle  $2c+1$  after reception of a sync frame in slot  $n$  the total number of different sync frames per double cycle has exceeded gSyncNodeMax and the node receives in slot  $n+1$  a sync frame that matches with a

---

**2 Functional deviations**

sync frame received in the even cycle 2c, the sync frame pair is not taken into account by CSP process. This may cause the flags SFS.MRCS and EIR.CCF to be set erroneously.

**Scope**

The erratum is limited to the case of a faulty cluster configuration where different sets of sync frames are transmitted in even and odd cycles and the total number of different sync frames is greater than gSyncNodeMax.

**Effects**

In the described case the error interrupt flag EIR.CCF is set and the node may enter either the POC state NORMAL\_PASSIVE or HALT.

**Workaround**

Correct configuration of gSyncNodeMax.

### **2.22 [FlexRay\_AI.091] Incorrect rate and / or offset correction value if second Secondary Time Reference Point (STRP) coincides with the action point after detection of a valid frame**

**Description**

If a valid sync frame is received before the action point and additionally noise or a second frame leads to a STRP coinciding with the action point, an incorrect deviation value of zero is used for further calculations of rate and/or offset correction values.

**Scope**

The erratum is limited to configurations with an action point offset greater than the static frame length.

**Effects**

In the described case a deviation value of zero is used for further calculations of rate and/or offset correction values. This may lead to an incorrect rate and / or offset correction of the node.

**Workaround**

Configure action point offset smaller than static frame length.

### **2.23 [FlexRay\_AI.092] Initial rate correction value of an integrating node is zero if pMicroInitialOffsetA,B = 0x00**

**Description**

The initial rate correction value as calculated in figure 8-8 of protocol spec v2.1 is zero if parameter pMicroInitialOffsetA,B was configured to be zero.

**Scope**

The erratum is limited to the case where pMicroInitialOffsetA,B is configured to zero.

**Effects**

Starting with an initial rate correction value of zero leads to an adjustment of the rate correction earliest 3 cycles later (see figure 7-10 of protocol spec v2.1). In a worst case scenario, if the whole cluster is drifting away too fast, the integrating node would not be able to follow and therefore abort integration.

## 2 Functional deviations

### Workaround

Avoid configurations with pMicroInitialOffsetA,B equal to zero. If the related configuration constraint of the protocol specification results in pMicroInitialOffsetA,B equal to zero, configure it to one instead. This will lead to a correct initial rate correction value, it will delay the start-up of the node by only one micro tick.

### 2.24 [FlexRay\_AI.093] Acceptance of start-up frames received after reception of more than gSyncNodeMax sync frames

#### Description

If a node receives in an even cycle a start-up frame after it has received more than gSyncNodeMax sync frames, this start-up frame is added erroneously by process CSP to the number of valid start-up frames (zStartupNodes). The faulty number of start-up frames is delivered to the process POC. As a consequence this node may integrate erroneously to the running cluster because it assumes that it has received the required number of start-up frames.

#### Scope

The erratum is limited to the case of more than gSyncNodeMax sync frames.

#### Effects

In the described case a node may erroneously integrate successfully into a running cluster.

#### Workaround

Use frame schedules where all start-up frames are placed in the first static slots. gSyncNodeMax should be configured to be greater than or equal to the number of sync frames in the cluster.

### 2.25 [FlexRay\_AI.094] Sync frame overflow flag EIR.SFO may be set if slot counter is greater than 1024

#### Description

If in the static segment the number of transmitted and received sync frames reaches gSyncNodeMax and the slot counter in the dynamic segment reaches the value  $cStaticSlotIDMax + gSyncNodeMax = 1023 + gSyncNodeMax$ , the sync frame overflow flag EIR.SFO is set erroneously.

#### Scope

The erratum is limited to configurations where the number of transmitted and received sync frames equals to gSyncNodeMax and the number of static slots plus the number of dynamic slots is greater or equal than  $1023 + gSyncNodeMax$ .

#### Effects

In the described case the sync frame overflow flag EIR.SFO is set erroneously. This has no effect to the POC state.

#### Workaround

Configure gSyncNodeMax to number of transmitted and received sync frames plus one or avoid configurations where the total of static and dynamic slots is greater than cStaticSlotIDMax.

---

## 2 Functional deviations

### 2.26 [FlexRay\_AI.095] Register RCV displays wrong value

#### Description

If the calculated rate correction value is in the range of  $[-pClusterDriftDamping .. +pClusterDriftDamping]$ , `vRateCorrection` of the CSP process is set to zero. In this case register RCV should be updated with this value. Erroneously `RCV.RCV[11:0]` holds the calculated value in the range  $[-pClusterDriftDamping .. +pClusterDriftDamping]$  instead of zero.

#### Scope

The erratum is limited to the case where the calculated rate correction value is in the range of  $[-pClusterDriftDamping .. +pClusterDriftDamping]$ .

#### Effects

The displayed rate correction value `RCV.RCV[11:0]` is in the range of  $[-pClusterDriftDamping .. +pClusterDriftDamping]$  instead of zero. The error of the displayed value is limited to the range of  $[-pClusterDriftDamping .. +pClusterDriftDamping]$ . For rate correction in the next double cycle always the correct value of zero is used.

#### Workaround

A value of `RCV.RCV[11:0]` in the range of  $[-pClusterDriftDamping .. +pClusterDriftDamping]$  has to be interpreted as zero.

### 2.27 [FlexRay\_AI.096] Noise following a dynamic frame that delays idle detection may fail to stop slot

#### Description

If (in case of noise) the time between 'potential idle start on X' and 'CHIRP on X' (see Protocol Spec. v2.1, Figure 5-21) is greater than `gdDynamicSlotIdlePhase`, the E-Ray will not remain for the remainder of the current dynamic segment in the state 'wait for the end of dynamic slot rx'. Instead, the E-Ray continues slot counting. This may enable the node to further transmissions in the current dynamic segment.

#### Scope

The erratum is limited to noise that is seen only locally and that is detected in the time window between the end of a dynamic frame's DTS and idle detection ('CHIRP on X').

#### Effects

In the described case the faulty node may not stop slot counting and may continue to transmit dynamic frames. This may lead to a frame collision in the current dynamic segment.

#### Workaround

None

### 2.28 [FlexRay\_AI.097] Loop back mode operates only at 10 MBit/s

#### Description

The looped back data is falsified at the two lower baud rates of 5 and 2.5 MBit/s.

---

**2 Functional deviations****Scope**

The erratum is limited to test cases where loop back is used with the baud rate prescaler (PRTC1.BRP[1:0]) configured to 5 or 2.5 MBit/s.

**Effects**

The loop back self test is only possible at the highest baud rate.

**Workaround**

Run loop back tests with 10 MBit/s (PRTC1.BRP[1:0] = 00<sub>B</sub>).

**2.29 [FlexRay\_AI.099] Erroneous cycle offset during start-up after abort of start-up or normal operation****Description**

An abort of start-up or normal operation by a READY command near the macro tick border may lead to the effect that the state INITIALIZE\_SCHEDULE is one macro tick too short during the first following integration attempt. This leads to an early cycle start in state INTEGRATION\_COLDSTART\_CHECK or INTEGRATION\_CONSISTENCY\_CHECK.

As a result the integrating node calculates a cycle offset of one macro tick at the end of the first even/odd cycle pair in the states INTEGRATION\_COLDSTART\_CHECK or INTEGRATION\_CONSISTENCY\_CHECK and tries to correct this offset.

If the node is able to correct the offset of one macro tick ( $pOffsetCorrectionOut \gg gdMacroTick$ ), the node enters NORMAL\_ACTIVE with the first start-up attempt.

If the node is not able to correct the offset error because  $pOffsetCorrectionOut$  is too small ( $pOffsetCorrectionOut \leq gdMacroTick$ ), the node enters ABORT\_STARTUP and is ready to try start-up again. The next (second) start-up attempt is not effected by this erratum.

**Scope**

The erratum is limited to applications where READY command is used to leave STARTUP, NORMAL\_ACTIVE, or NORMAL\_PASSIVE state.

**Effects**

In the described case the integrating node tries to correct an erroneous cycle offset of one macro tick during start-up.

**Workaround**

With a configuration of  $pOffsetCorrectionOut \gg gdMacroTick \cdot (1+cClockDeviationMax)$  the node will be able to correct the offset and therefore also be able to successfully integrate.

**2.30 [FlexRay\_AI.100] First WUS following received valid WUP may be ignored****Description**

When the protocol engine is in state WAKEUP\_LISTEN and receives a valid wake-up pattern (WUP), it transfers into state READY and updates the wake-up status vector CCSV.WSV[2:0] as well as the status interrupt flags SIR.WST and SIR.WUPA/B. If the received wake-up pattern continues, the protocol engine may ignore the first wake-up symbol (WUS) following the state transition and signals the next SIR.WUPA/B at the third instead of the second WUS.

---

## **2 Functional deviations**

### **Scope**

The erratum is limited to the reception of redundant wake-up patterns.

### **Effects**

Delayed setting of status interrupt flags SIR.WUPA/B for redundant wake-up patterns.

### **Workaround**

None

## **2.31 [FlexRay\_AI.101] READY command accepted in READY state**

### **Description**

The E-Ray module does not ignore a READY command while in READY state.

### **Scope**

The erratum is limited to the READY state.

### **Effects**

Flag CCSV.CSI is set. Cold starting needs to be enabled by POC command ALLOW\_COLDSTART (SUCC1.CMD = 1001<sub>B</sub>).

### **Workaround**

None

## **2.32 [FlexRay\_AI.102] Slot status vPOC!SlotMode is reset immediately when entering HALT state**

### **Description**

When the protocol engine is in the states NORMAL\_ACTIVE or NORMAL\_PASSIVE, a HALT or FREEZE command issued by the Host resets vPOC!SlotMode immediately to SINGLE slot mode (CCSV.SLM[1:0] = 00<sub>B</sub>). According to the FlexRay protocol specification, the slot mode should not be reset to SINGLE slot mode before the following state transition from HALT to DEFAULT\_CONFIG state.

### **Scope**

The erratum is limited to the HALT state.

### **Effects**

The slot status vPOC!SlotMode is reset to SINGLE when entering HALT state.

### **Workaround**

None

---

## **2 Functional deviations**

### **2.33 [FlexRay\_AI.103] Received messages not stored in Message RAM when in Loop Back Mode**

#### **Description**

After a FREEZE or HALT command has been asserted in NORMAL\_ACTIVE state, and if state LOOP\_BACK is then entered by transition from HALT state via DEF\_CONFIG and CONFIG, it may happen that acceptance filtering for received messages is not started, and therefore these messages are not stored in the respective receive buffer in the Message RAM.

#### **Scope**

The erratum is limited to the case where Loop Back Mode is entered after NORMAL\_ACTIVE state was left by FREEZE or HALT command.

#### **Effects**

Received messages are not stored in Message RAM because acceptance filtering is not started.

#### **Workaround**

Leave HALT state by hardware reset.

### **2.34 [FlexRay\_AI.104] Missing start-up frame in cycle 0 at coldstart after FREEZE or READY command**

#### **Description**

When the E-Ray is restarted as leading coldstarter after it has been stopped by FREEZE or READY command, it may happen, depending on the internal state of the module, that the E-Ray does not transmit its start-up frame in cycle 0. Only E-Ray configurations with start-up frames configured for slots 1 to 7 are affected by this behavior.

#### **Scope**

The erratum is limited to the case when a coldstart is initialized after the E-Ray has been stopped by FREEZE or READY command. Coldstart after hardware reset is not affected.

#### **Effects**

During coldstart it may happen that no start-up frame is sent in cycle 0 after entering COLDSTART\_COLLISION\_RESOLUTION state from COLDSTART\_LISTEN state.

The next coldstart attempt is no longer affected. Coldstart sequence is lengthened but coldstart of FlexRay system is not prohibited by this behavior.

#### **Workaround**

Use a static slot greater or equal 8 for the start-up/sync message.

## 2 Functional deviations

### 2.35 [FlexRay\_AI.105] RAM select signals of IBF1/IBF2 and OBF1/OBF2 in RAM test mode

#### Description

When accessing Input Buffer RAM 1, 2 (IBF1, 2) or Output Buffer RAM 1, 2 (OBF1, 2) in RAM test mode, the following behavior can be observed when entering RAM test mode after hardware reset.

- Read or write access to IBF2:
  - In this case also IBF1 RAM select `eray_ibf1_cen` is activated initiating a read access of the addressed IBF1 RAM word. The data read from IBF1 is evaluated by the respective parity checker.
- Read or write access to OBF1:
  - In this case also OBF2 RAM select `eray_obf2_cen` is activated initiating a read access of the addressed OBF2 RAM word. The data read from OBF2 is evaluated by the respective parity checker.

If the parity logic of the erroneously selected IBF1 resp. OBF2 detects a parity error, bit `MHDS.PIBF` resp. `MHDS.POBF` in the E-Ray Message Handler Status register is set although the addressed IBF2 resp. OBF1 had not error. The logic for setting `MHDS.PIBF` / `MHDS.POBF` does not distinguish between set conditions from IBF1 or IBF2 resp. OBF1 or OBF2.

Due to the IBF / OBF swap mechanism as described in section 5.11.2 in the E-Ray Specification, the inverted behavior with respect to IBF1, 2 and OBF1, 2 can be observed depending on the IBF / OBF access history.

#### Scope

The erratum is limited to the case when IBF1, 2 or OBF1, 2 are accessed in RAM test mode. The problem does not occur when the E-Ray is in normal operation mode.

#### Effects

When reading or writing IBF1, 2 / OBF1, 2 in RAM test mode, it may happen, that the parity logic of IBF1, 2 / OBF1, 2 signals a parity error.

#### Workaround

For RAM testing after hardware reset, the Input / Output Buffer RAMs have to be first written and then read in the following order: IBF1 before IBF2 and OBF2 before OBF1

### 2.36 [FlexRay\_AI.106] Data transfer overrun for message transfers Message RAM to Output Buffer (OBF) or from Input Buffer (IBF) to Message RAM

#### Description

The problem occurs under the following conditions:

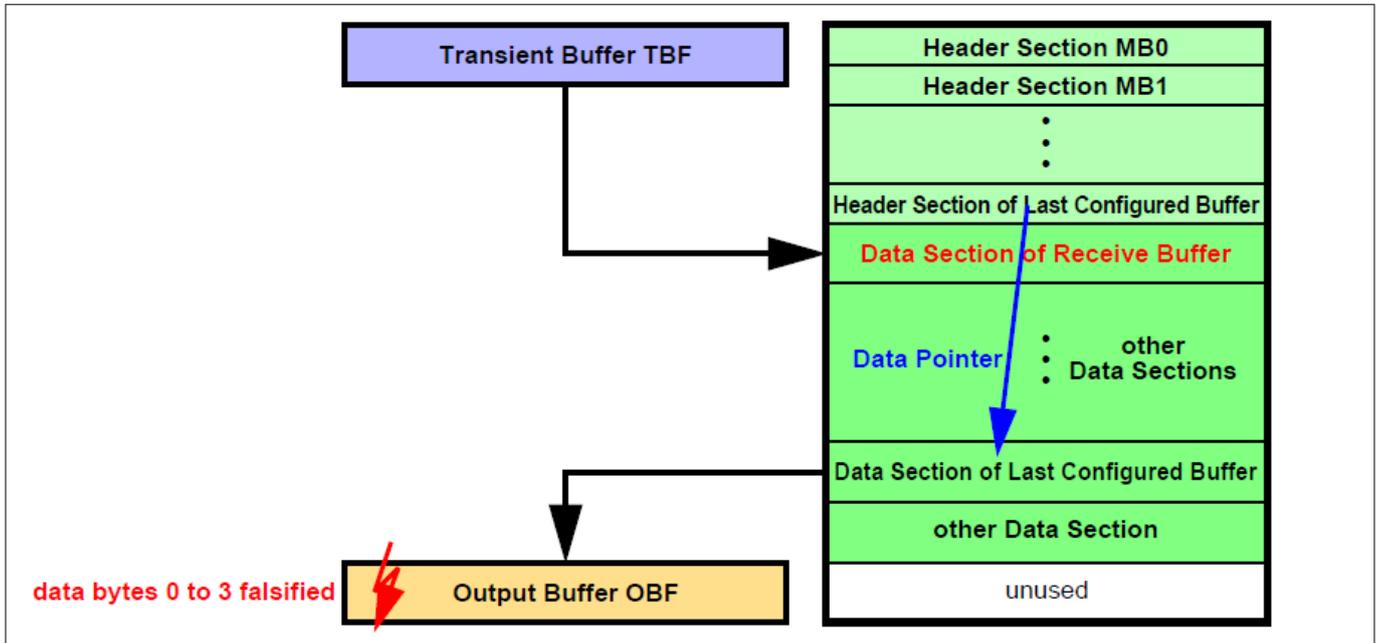
- 1) A received message is transferred from the Transient Buffer RAM (TBF) to the message buffer that has its data pointer pointing to the first word of the Message RAM's Data Partition located directly after the last header word of the Header Partition of the Last Configured Buffer as defined by `MRC.LCB`
- 2) The Host triggers a transfer from / to the Last Configured Buffer in the Message RAM with a specific time relation to the start of the TBF transfer described under 1)

Under these conditions the following transfers triggered by the Host may be affected:

- a) Message buffer transfer from Message RAM to OBF

When the message buffer has its payload configured to maximum length (`PLC = 127`), the OBF word on address `00h` (payload data bytes 0 to 3) is overwritten with unexpected data at the end of the transfer.

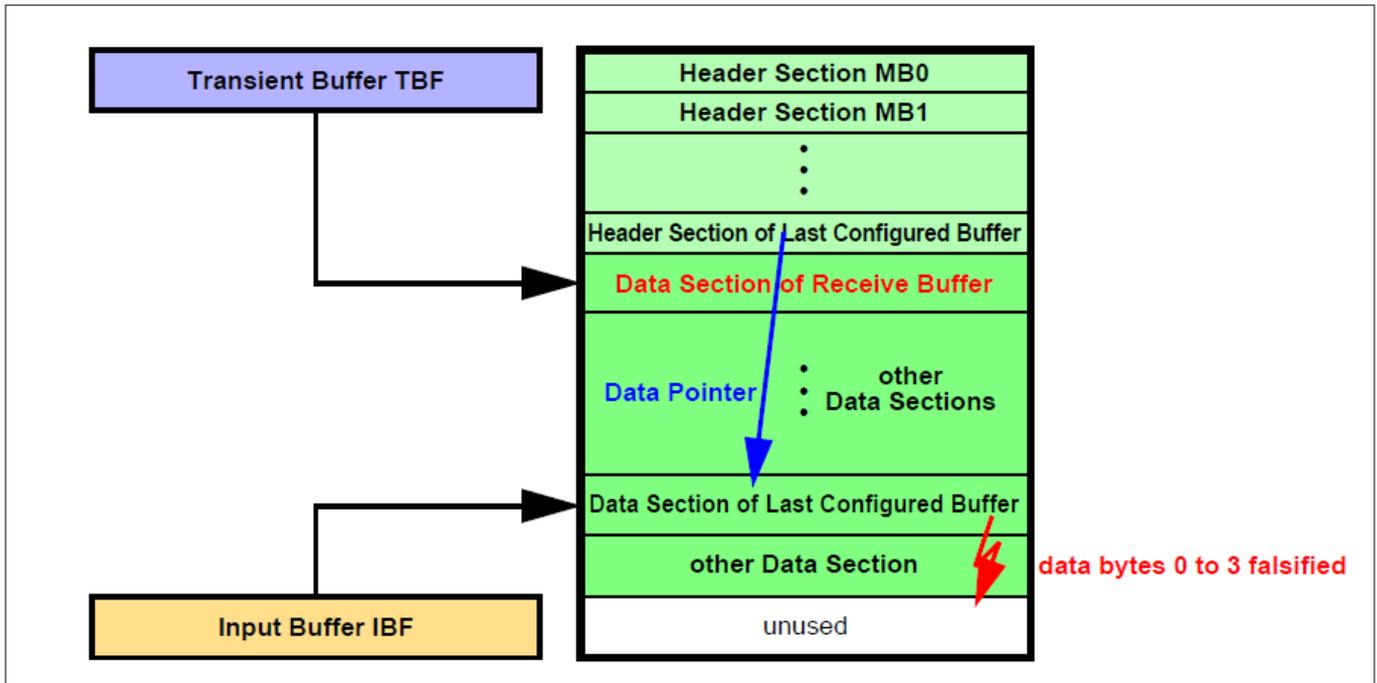
2 Functional deviations



**Figure 1** Message buffer transfer from Message RAM to OBF

b) Message buffer transfer from IBF to Message RAM

After the Data Section of the selected message buffer in the Message RAM has been written, one additional write access overwrites the following word in the Message RAM which might be the first word of the next Data Section.



**Figure 2** Message buffer transfer from IBF to Message RAM

**Scope**

The erratum is limited to the case when (see Figure 3 “Bad Case”):

- 1) The first Data Section in the Data Partition is assigned to a receive buffer (incl. FIFO buffers)

AND

**2 Functional deviations**

2) The Data Partition in the Message RAM starts directly after the Header Partition (no unused Message RAM word in between)

**Effects**

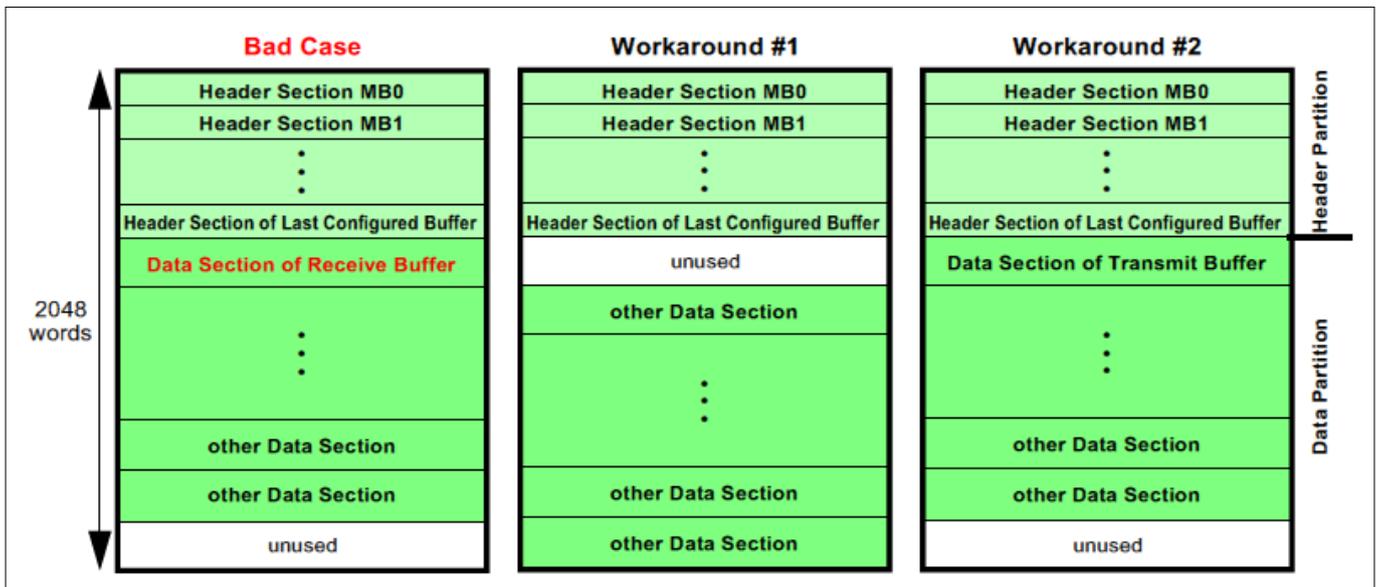
- a) When a message is transferred from the Last Configured Buffer in the Message RAM to the OBF and PLC = 127 it may happen, that at the end of the transfer the OBF word on address 00h (payload data bytes 0 to 3) is overwritten with unexpected data (see [Figure 1](#))
- b) When a message is transferred from IBF to the Last Configured Buffer in the Message RAM, it may happen, that at the end of the transfer of the Data Section one additional write access overwrites the following word, which may be the first word of another message's Data Section in the Message RAM (see [Figure 2](#))

**Workaround 1**

Leave at least one unused word in the Message RAM between Header Section and Data Section.

**Workaround 2**

Ensure that the Data Section directly following the Header Partition is assigned to a transmit buffer.



**Figure 3** Message RAM configurations

**2.37 [GETH\_AI.043] Access type of register bit-fields not correctly marked in UM**

**Description**

Access rights are wrongly documented for hardware modifiable and clear on read and write 1 to clear bits. The following bit-fields inside the GETH specification are not marked as volatile bits, even though they can be changed by hardware during run-time. The definition rw1ch is not used in TC3xx documentation. This is only available inside the text description. To see directly the proper definition, refer to the access right description within this errata. Please be aware that this errata states all available resources in TC39x. As all other devices of the TC3xx family are having less features, the change of the access right is only relevant, if the register exists in the device.

**2 Functional deviations**

**Table 5**

<b>Register name</b>	<b>Bitfield name</b>	<b>Proper access type</b>
MAC_VLAN_Tag_Ctrl	OB	rwh
MAC_VLAN_Incl	BUSY	rh
MAC_Q0_Tx_Flow_Ctrl	FCB_BPA	rwh
MAC_Interrupt_Status	RGSMIIS	rh
	PHYSIS	rch or rw1ch <sup>(1)</sup>
	PMTIS	rh
	LPIIS	rh
	MMCIS	rh
	MMCRXIS	rh
	MMCTXIS	rh
	MMCRXIPIS	rh
	TSIS	rh
	TXSTIS	rh
	RXSTIS	rh
	MDIOIS	rch or rw1ch <sup>(1)</sup>
MAC_Rx_Tx_Status	TJT	rch or rw1ch <sup>(1)</sup>
	NCARR	rch or rw1ch <sup>(1)</sup>
	LCARR	rch or rw1ch <sup>(1)</sup>
	EXDEF	rch or rw1ch <sup>(1)</sup>
	LCOL	rch or rw1ch <sup>(1)</sup>
	EXCOL	rch or rw1ch <sup>(1)</sup>
	RWT	rch or rw1ch <sup>(1)</sup>
MAC_PMT_Control_Status	PWRDWN	rwh
	MGKPRCVD	rch or rw1ch <sup>(1)</sup>
	RWKPRCVD	rch or rw1ch <sup>(1)</sup>
	RWKPTR	rh
	RWKFILTRST	rwh
MAC_RWK_Packet_Filter	WKUPFRMFTR	rwh
MAC_LPI_Control_Status	TLPIEN	rch or rw1ch <sup>(1)</sup>
	TLPIEX	rch or rw1ch <sup>(1)</sup>
	RLPIEN	rch or rw1ch <sup>(1)</sup>
	RLPIEX	rch or rw1ch <sup>(1)</sup>
	TLPIST	rh
	RLPIST	rh
MAC_PHYIF_Control_Status	LNKMOD	rh
	LNKSPEED	rh

**(table continues...)**

**2 Functional deviations**

**Table 5 (continued)**

<b>Register name</b>	<b>Bitfield name</b>	<b>Proper access type</b>
	LNKSTS	rh
MAC_Debug	RPESTS	rh
	RFCFCSTS	rh
	TPESTS	rh
	TFCSTS	rh
MAC_MDIO_Address	GB	rwh
MAC_MDIO_Data	GD	rwh
	RA	rw
MMC_Control	CNTRST	rwh
	CNTPRST	rwh
MMC_Rx_Interrupt	RXGBPCTIS	rch
	RXGBOCTIS	rch
	RXGOCTIS	rch
	RXBCGPIS	rch
	RXMCGPIS	rch
	RXCRCERPIS	rch
	RXALGNERPIS	rch
	RXRUNTPIS	rch
	RXJABERPIS	rch
	RXUSIZEGPIS	rch
	RXOSIZEGPIS	rch
	RX64OCTGBPIS	rch
	RX65T127OCTGBPIS	rch
	RX128T255OCTGBPIS	rch
	RX256T511OCTGBPIS	rch
	RX512T1023OCTGBPIS	rch
	RX1024TMAXOCTGBPIS	rch
	RXUCGPIS	rch
	RXLENERPIS	rch
	RXORANGEPIS	rch
	RXPAUSPIS	rch
	RXFOVPIS	rch
	RXVLANGBPIS	rch
	RXWDOGPIIS	rch
RXRCVERRPIS	rch	
RXCTRLPIS	rch	

**(table continues...)**

2 Functional deviations

Table 5 (continued)

Register name	Bitfield name	Proper access type
	RXLPIUSCIS	rch
	RXLPITRCIS	rch
MMC_Tx_Interrupt	TXGBOCTIS	rch
	TXGBPKTIS	rch
	TXBCGPIS	rch
	TXMCGPIS	rch
	TX64OCTGBPIS	rch
	TX65T127OCTGBPIS	rch
	TX128T255OCTGBPIS	rch
	TX256T511OCTGBPIS	rch
	TX512T1023OCTGBPIS	rch
	TX1024TMAXOCTGBPIS	rch
	TXUCGBPIS	rch
	TXMCGBPIS	rch
	TXBCGBPIS	rch
	TXUFLOWERPIS	rch
	TXSCOLGPIS	rch
	TXMCOLGPIS	rch
	TXDEFPIS	rch
	TXLATCOLPIS	rch
	TXEXCOLPIS	rch
	TXCARERPIS	rch
	TXGOCTIS	rch
	TXGPKTIS	rch
	TXEXDEFPIS	rch
	TXPAUSPIS	rch
TXVLANGPIS	rch	
TXOSIZEGPIS	rch	
TXLPIUSCIS	rch	
TXLPITRCIS	rch	
Tx_Octet_Count_Good_Bad	TXOCTGB	rh
Tx_Packet_Count_Good_Bad	TXPKTGB	rh
Tx_Broadcast_Packets_Good	TXBCASTG	rh
Tx_Multicast_Packets_Good	TXMCASTG	rh
Tx_64Octets_Packets_Good_Bad	TX64OCTGB	rh

(table continues...)

**2 Functional deviations**

**Table 5 (continued)**

<b>Register name</b>	<b>Bitfield name</b>	<b>Proper access type</b>
Tx_65To127Octets_Packets_Good_Bad	TX65_127OCTGB	rh
Tx_128To255Octets_Packets_Good_Bad	TX128_255OCTGB	rh
Tx_256To511Octets_Packets_Good_Bad	TX256_511OCTGB	rh
Tx_512To1023Octets_Packets_Good_Bad	TX512_1023OCTGB	rh
Tx_1024ToMaxOctets_Packets_Good_Bad	TX1024_MAXOCTGB	rh
Tx_Unicast_Packets_Good_Bad	TXUCASTGB	rh
Tx_Multicast_Packets_Good_Bad	TXMCASTGB	rh
Tx_Broadcast_Packets_Good_Bad	TXBCASTGB	rh
Tx_Underflow_Error_Packets	TXUNDRFLW	rh
Tx_Single_Collision_Good_Packets	TXSNGLCOLG	rh
Tx_Multiple_Collision_Good_Packets	TXMULTCOLG	rh
Tx_Deferred_Packets	TXDEFRD	rh
Tx_Late_Collision_Packets	TXLATECOL	rh
Tx_Excessive_Collision_Packets	TXEXSCOL	rh
Tx_Carrier_Error_Packets	TXCARR	rh
Tx_Octet_Count_Good	TXOCTG	rh
Tx_Packet_Count_Good	TXPKTG	rh
Tx_Excessive_Deferral_Error	TXEXSDEF	rh
Tx_Pause_Packets	TXPAUSE	rh
Tx_VLAN_Packets_Good	TXVLANG	rh
Tx_OSize_Packets_Good	TXOSIZG	rh
Rx_Packets_Count_Good_Bad	RXPKTGB	rh
Rx_Octet_Count_Good_Bad	RXOCTGB	rh
Rx_Octet_Count_Good	RXOCTG	rh
Rx_Broadcast_packets_Good	RXBCASTG	rh
Rx_Multicast_packets_Good	RXMCASTG	rh
Rx_CRC_Error_Packets	RXCRCERR	rh
Rx_Alignment_Error_packets	RXALGNERR	rh
Rx_Runt_Error_Packets	RXRUNTERR	rh
Rx_Jabber_Error_Packets	RXJABERR	rh
Rx_Undersize_Packets_Good	RXUNDERSZG	rh
RX_Oversize_Packets_Good	RXOVERSZG	rh

**(table continues...)**

**2 Functional deviations**

**Table 5 (continued)**

<b>Register name</b>	<b>Bitfield name</b>	<b>Proper access type</b>
Rx_64Octets_Packets_Good_Bad	RX64OCTGB	rh
Rx_65To127Octets_Packets_Good_Bad	RX65_127OCTGB	rh
Rx_128To255Octets_Packets_Good_Bad	RX128_255OCTGB	rh
Rx_256To511Octets_Packets_Good_Bad	RX256_511OCTGB	rh
Rx_512To1023Octets_Packets_Good_Bad	RX512_1023OCTGB	rh
Rx_1024ToMaxOctets_Packets_Good_Bad	RX1024_MAXOCTGB	rh
Rx_Unicast_Packets_Good	RXUCASTG	rh
Rx_Length_Error_Packets	RXLENERR	rh
Rx_Out_Of_Range_Type_Packets	RXOUTOFRNG	rh
Rx_Pause_Packets	RXPAUSEPKT	rh
Rx_FIFO_Overflow_Packets	RXFIFOOVFL	rh
Rx_VLAN_Packets_Good_Bad	RXVLANPKTGB	rh
Rx_Watchdog_Error_Packets	RXWDGERR	rh
Rx_Receive_Error_Packets	RXRCVERR	rh
Rx_Control_Packets_Good	RXCTRLG	rh
Tx_LPI_USEC_Cntr	TXLPIUSC	rh
Tx_LPI_Tran_Cntr	TXLPITRC	rh
Rx_LPI_USEC_Cntr	RXLPIUSC	rh
Rx_LPI_Tran_Cntr	RXLPITRC	rh
MMC_IPC_Rx_Interrupt	RXIPV4GPIS	rch
	RXIPV4HERPIS	rch
	RXIPV4NOPAYPIS	rch
	RXIPV4FRAGPIS	rch
	RXIPV4UDSBLPIS	rch
	RXIPV6GPIS	rch
	RXIPV6HERPIS	rch
	RXIPV6NOPAYPIS	rch
	RXUDPGPIS	rch
	RXUDPERPIS	rch
	RXTCPGPIS	rch
	RXTCPERPIS	rch
	RXICMPGPIS	rch

**(table continues...)**

2 Functional deviations

Table 5 (continued)

Register name	Bitfield name	Proper access type
	RXICMPERPIS	rch
	RXIPV4GOIS	rch
	RXIPV4HEROIS	rch
	RXIPV4NOPAYOIS	rch
	RXIPV4FRAGOIS	rch
	RXIPV4UDSBLOIS	rch
	RXIPV6GOIS	rch
	RXIPV6HEROIS	rch
	RXIPV6NOPAYOIS	rch
	RXUDPGOIS	rch
	RXUDPEROIS	rch
	RXTCPGOIS	rch
	RXTCPEROIS	rch
	RXICMPGOIS	rch
	RXICMPEROIS	rch
RxIPv4_Good_Packets	RXIPV4GDPKT	rh
RxIPv4_Header_Error_Packets	RXIPV4HDRERRPKT	rh
RxIPv4_No_Payload_Packets	RXIPV4NOPAYPKT	rh
RxIPv4_Fragmented_Packets	RXIPV4FRAGPKT	rh
RxIPv4_UDP_Checksum_Disabled_Packets	RXIPV4UDSBLPKT	rh
RxIPv6_Good_Packets	RXIPV6GDPKT	rh
RxIPv6_Header_Error_Packets	RXIPV6HDRERRPKT	rh
RxIPv6_No_Payload_Packets	RXIPV6NOPAYPKT	rh
RxUDP_Good_Packets	RXUDPGDPKT	rh
RxUDP_Error_Packets	RXUDPERRPKT	rh
RxTCP_Good_Packets	RXTCPGDPKT	rh
RxTCP_Error_Packets	RXTCPERRPKT	rh
RxICMP_Good_Packets	RXICMPGPKT	rh
RxICMP_Error_Packets	RXICMPGERRPKT	rh
RxIPv4_Good_Octets	RXIPV4GDOCT	rh
RxIPv4_Header_Error_Octets	RXIPV4HDRERROCT	rh
RxIPv4_No_Payload_Octets	RXIPV4NOPAYOCT	rh
RxIPv4_Fragmented_Octets	RXIPV4FRAGOCT	rh
RxIPv4_UDP_Checksum_Disable_Octets	RXIPV4UDSBLOCT	rh

(table continues...)

**2 Functional deviations**

**Table 5 (continued)**

<b>Register name</b>	<b>Bitfield name</b>	<b>Proper access type</b>
RxIPv6_Good_Octets	RXIPV6GDOCT	rh
RxIPv6_Header_Error_Octets	RXIPV6HDRERROCT	rh
RxIPv6_No_Payload_Octets	RXIPV6NOPAYOCT	rh
RxUDP_Good_Octets	RXUDPGDOCT	rh
RxUDP_Error_Octets	RXUDPERROCT	rh
RxTCP_Good_Octets	RXTCPGDOCT	rh
RxTCP_Error_Octets	RXTCPERROCT	rh
RxICMP_Good_Octets	RXICMPGDOCT	rh
RxICMP_Error_Octets	RXICMPERROCT	rh
MAC_Timestamp_Control	TSINIT	rwh
	TSUPDT	rwh
	TSADDREG	rwh
MAC_System_Time_Seconds	TSS	rh
MAC_System_Time_Nanoseconds	TSSS	rh
MAC_System_Time_Higher_Word_Seconds	TSHWR	rwh
MAC_Timestamp_Status	TSSOVF	rch or rw1ch <sup>(1)</sup>
	TSTARGET0	rch or rw1ch <sup>(1)</sup>
	TSTRGTERR0	rch or rw1ch <sup>(1)</sup>
	TSTARGET1	rch or rw1ch <sup>(1)</sup>
	TSTRGTERR1	rch or rw1ch <sup>(1)</sup>
	TSTARGET2	rch or rw1ch <sup>(1)</sup>
	TSTRGTERR2	rch or rw1ch <sup>(1)</sup>
	TSTARGET3	rch or rw1ch <sup>(1)</sup>
	TSTRGTERR3	rch or rw1ch <sup>(1)</sup>
TXTSSIS	rh	
MAC_Tx_Timestamp_Status_Nanoseconds	TXTSSLO	rh
	TXTSSMIS	rch
MAC_Tx_Timestamp_Status_Seconds	TXTSSHI	rh
MAC_PPS_Control	PPSCTRL_PPSCMD	rwh
MAC_PPS0_Target_Time_Nanoseconds	TTSLO	rwh
	TRGTBUSY0	rwh
MTL_Operation_Mode	CNTPRST	rwh
	CNTCLR	rwh
MTL_Interrupt_Status	Q0IS	rh

**(table continues...)**

## 2 Functional deviations

**Table 5** (continued)

Register name	Bitfield name	Proper access type
	Q1IS	rh
	Q2IS	rh
	Q3IS	rh
	Q4IS	rh
MTL_TxQi_Operation_Mode	FTQ	rwh
MTL_TxQi_Underflow	UFFRMCNT	rch
	UFCNTOVF	rch
MTL_TxQi_Debug	TXQPAUSED	rh
	TRCSTS	rh
	TWCSTS	rh
	TXQSTS	rh
	TXSTSFSTS	rh
	PTXQ	rh
	STXSTSF	rh
MTL_TxQi_ETS_Status	ABS	rh
MTL_Qi_Interrupt_Control_Status	TXUNFIS	rw1ch <sup>(1)</sup>
	RXOVFIS	rw1ch <sup>(1)</sup>
	ABPSIS	rw1ch <sup>(1)</sup>
MTL_RxQi_Missed_Packet_Overflow_Cnt	OVFPKTCNT	rch
	OVFCNTOVF	rch
	MISPKTCNT	rch
	MISCNTOVF	rch
MTL_RxQi_Debug	RWCSTS	rh
	RRCSTS	rh
	RXQSTS	rh
	PRXQ	rh
DMA_Mode	SWR	rwh
DMA_Interrupt_Status	DC0IS	rh
	DC1IS	rh
	DC2IS	rh
	DC3IS	rh
	MTLIS	rh
	MACIS	rh
DMA_Debug_Status0	AXWHSTS	rh
	RPS0	rh
	TPS0	rh

(table continues...)

**2 Functional deviations**

**Table 5 (continued)**

Register name	Bitfield name	Proper access type
	RPS1	rh
	TPS1	rh
	RPS2	rh
	TPS2	rh
DMA_Debug_Status1	RPS3	rh
	TPS3	rh
DMA_CHi_Tx_Control	ST	rwh
DMA_CHi_Rx_Control	SR	rwh
DMA_CHi_Slot_Function_Control_Status	RSN	rh
DMA_CHi_Current_App_TxDesc	CURTDESAPTR	rh
DMA_CHi_Current_App_RxDesc	CURRDESAPTR	rh
DMA_CHi_Current_App_TxBuffer	CURTBUFAPTR	rh
DMA_CHi_Current_App_RxBuffer	CURRBUFAPTR	rh
DMA_CHi_Status	TI	rw1ch <sup>(1)</sup>
	TPS	rw1ch <sup>(1)</sup>
	TBU	rw1ch <sup>(1)</sup>
	RI	rw1ch <sup>(1)</sup>
	RPS	rw1ch <sup>(1)</sup>
	RBU	rw1ch <sup>(1)</sup>
	RWT	rwh
	ETI	rw1ch <sup>(1)</sup>
	ERI	rw1ch <sup>(1)</sup>
	FBE	rw1ch <sup>(1)</sup>
	CDE	rw1ch <sup>(1)</sup>
	AIS	rw1ch <sup>(1)</sup>
	NIS	rw1ch <sup>(1)</sup>
	TEB	rh
	REB	rh
DMA_CHi_Miss_Frame_Cnt	MFC	rch
	MFCO	rch

**(1) Write 1<sub>B</sub> to clear, when CSR\_SW\_Ctrl.RCWE = 1<sub>B</sub>**

- Note:**
- *r*: software read
  - *rc*: software read to clear
  - *w*: software write
  - *w1c*: software write 1 to clear
  - *h*: hardware write

---

## **2 Functional deviations**

### **Scope**

Register bit-field access rights.

### **Effects**

Optimization via compiler might be wrong.

### **Workaround**

Use the new access rights as defined in table above.

## **2.38 [GTM\_AI.254] TIM TDU: TDU\_STOP=b101 not functional**

### **Description**

Stop counting of register TO\_CNT on an tdu\_word\_event or stop counting of TO\_CNT1 on a tdu\_frame\_event is not possible.

### **Scope**

TIM

### **Effects**

TO\_CNT1, TO\_CNT can not be stopped counting.

### **Workaround**

No workaround available.

## **2.39 [GTM\_AI.308] TIM, ARU: Limitation that back-to-back TIM data transfers at full ARU clock rate cannot be transferred correctly with ARU dynamic routing feature**

### **Description**

If TIM input signals with signal changes faster or equal than ARU clock rate are processed with the TIM and the results are routed via ARU in dynamic routing mode, it is likely that there is a data loss and only each second data can be transferred.

### **Scope**

ARU Routing, DEBUG signal interface

### **Effects**

- a) If the ARU CADDR is kept stable and data is transferred back-to-back for 2 or more consecutive aru clock cycles while operating in ARU dynamic routing mode, then every second data provided by the TIM module gets lost
- b) Debugging of an ARU data transfer not completely correct. Every second GTM\_DBG\_ARU\_DATAi\_val signal missing

### **Workaround**

Do not use the dynamic routing feature of ARU in the manner that the same ARU caddr is served for multiple cycles with back-to-back data transfers.

## 2 Functional deviations

Ensure that every ARU clock cycle the CADDR address will change.

### **2.40 [GTM\_AI.335] TOM output signal to SPE not functional if up/down counter mode is configured**

#### **Description**

TOM output signal TOM[i]\_CH[x]\_SOUR to SPE not functional if up/down counter mode is configured by setting of TOM[i]\_CH[x]\_CTRL.UDMODE > 0.

#### **Scope**

TOM - SPE interface

#### **Effects**

TOM output signal TOM[i]\_CH[x]\_SOUR to SPE not functional.

#### **Workaround**

No workaround available.

Don't use up/down counter mode together with SPE interface.

### **2.41 [GTM\_AI.340] TOM/ATOM: Generation of TRIG\_CCU0/TRIG\_CCU1 trigger signals skipped in initial phase of A/TOM SOMP one-shot mode**

#### **Description**

Configuration in use:

- A/TOM[i]\_CH[x]\_CTRL.OSM=1
- A/TOM[i]\_CH[x]\_CTRL.OSM\_TRIG=0
- A/TOM[i]\_CH[x]\_CTRL.UDMODE=00
- ATOM[i]\_CH[x]\_CTRL.MODE=10

#### **Expected behavior**

The generation of one-shot pulses in A/TOM can be initiated by a write to CN0. In this case the pulse generation comprises of an initial phase where the signal level at A/TOM output is inactive followed by a pulse. The duration of the initial phase can be controlled by the written value of CN0, where the duration is defined by CM0-CN0. After the counter CN0 reaches the value of CM0-1, the pulse starts with its active edge, CN0 is reset, and starts counting again. When CN0 reaches CM1-1, the inactive edge of the pulse occurs. Due to the fact, that the capture compare units CCU0 and CCU1 compare also in the initial phase of the pulse generation, the trigger conditions for these comparators apply also in this initial phase. Hence, the TRIG\_CCU0 and TRIG\_CCU1 signals also occur in the initial phase of the one-shot pulse. When these trigger signals are enabled in the A/TOM[i]\_CH[x]\_IRQ\_EN, an interrupt signal is generated by A/TOM on the CCU0TC and CCU1TC trigger conditions and the corresponding A/TOM[i]\_CH[x]\_IRQ\_NOTIFY bits are set.

#### **Observed behavior**

For certain start values of CN0 and dependent on the history of pulse generation, the trigger signals TRIG\_CCU0 and TRIG\_CCU1 are skipped. As a consequence, this can led to missing interrupts CCU0TC and CCU1TC on behalf of their missing trigger signals TRIG\_CCU0 and TRIG\_CCU1.

## 2 Functional deviations

For the first pulse generation after enabling the channel, all trigger signals TRIG\_CCU0 and TRIG\_CCU1 appear as expected and described in the section expected behavior. If the channel stays enabled and a new value CN0 is written to trigger a subsequent one-shot pulse.

The TRIG\_CCU0/TRIG\_CCU1 triggers in the initial phases of subsequent one-shot pulses are skipped under the following conditions:

- For TRIG\_CCU0 trigger: if the one-shot pulse is started by writing a value to CN0 greater or equal to CM0-1
- For TRIG\_CCU1 trigger: if the one-shot pulse is started by writing a value to CN0 greater or equal to CM1-1

### Scope

TOM/ATOM

### Effects

Missing TRIG\_CCU0 and TRIG\_CCU1 trigger signals in initial phase of subsequent pulses in A/TOM one-shot mode, when one shot-mode is started with writing to CN0 values greater equal CM0-1 or CM1-1.

### Workaround 1

Disabling, resetting (channel reset), initializing and re-enabling of the channel before starting the next one-shot pulse by writing of CN0 ensures the correct behavior of CCU0TC and CCU1TC interrupt source.

### Workaround 2

Starting a new one-shot pulse by writing twice the counter CN0 whereas the first value, which is written to CN0 should be zero followed by the value which defines the length of the initial phase.

Note that in this case, the total length of the initial phase until the pulse is started, is influenced by the time between the two write accesses to CN0.

## 2.42 [GTM\_AI.341] TOM/ATOM: False generation of TRIG\_CCU1 trigger signal in SOMP one-shot mode with OSM\_TRIG=1 when CM1 is set to value 1

### Description

Configuration in use:

- A/TOM[i]\_CH[x]\_CTRL.OSM=1
- A/TOM[i]\_CH[x]\_CTRL.OSM\_TRIG=1
- A/TOM[i]\_CH[x]\_CTRL.UDMODE=00
- ATOM[i]\_CH[x]\_CTRL.MODE=10

### Expected behavior

The generation of one-shot pulses in A/TOM can be initiated by the trigger event TRIG\_[x-1] from trigger chain or by TIM\_EXT\_CAPTURE(x) trigger event from TIM, whereas the counter CN0 is reset to zero and starts counting. In this case the pulse generation comprises of an initial phase where the signal level at A/TOM output is inactive followed by a pulse. The duration of the initial phase is always as long until the counter CN0 reaches CM0-1.

After the counter CN0 reaches the value of CM0-1, the pulse starts with its active edge, CN0 is reset, and starts counting again. When CN0 reaches CM1-1, the inactive edge of the pulse occurs. Due to the fact, that the capture compare units CCU0 and CCU1 compare also in the initial phase of the pulse generation, the trigger conditions for these comparators apply also in this initial phase. Thus, the TRIG\_CCU0 and TRIG\_CCU1 signals also occur in the initial phase of the one-shot pulse. When these trigger signals are enabled in the A/TOM[i]\_CH[x]\_IRQ\_EN, an interrupt signal is generated by A/TOM on the CCU0TC and CCU1TC trigger conditions and the corresponding A/TOM[i]\_CH[x]\_IRQ\_NOTIFY bits are set.

## 2 Functional deviations

### Observed behavior

If the compare register CM1 is set to 1 and a new one-shot pulse is triggered, two effects can be observed:

- The first observed behavior is that the capture compare unit doesn't generate the TRIG\_CCUI1 trigger signal in the initial phase of the one-shot cycle
- The second observed behavior is that at the end of the operation phase of the one-shot cycle, where CN0 reaches CM0-1 a second time, the capture compare unit generates a TRIG\_CCUI1 trigger signal which is not expected at this point in time

### Scope

TOM/ATOM

### Effects

Missing TRIG\_CCUI1 trigger signal in initial phase of the one-shot cycle and unexpected TRIG\_CCUI1 trigger signal at the end of the operation phase of the one-shot cycle.

### Workaround

Instead of using value 1 for CM1 it could be possible to generate the same pulse length by using a higher CMU\_FXCLK/CMU\_CLK frequency. Then, to get the same pulse length, the value of CM1 has to be multiplied by the difference of the two CMU\_FXCLK/CMU\_CLK frequencies.

Be aware that this workaround is only possible, if you are not already using the CMU\_FXCLK(0) because there is no higher CMU\_FXCLK frequency to select.

**Example for TOM** : Instead of using CMU\_FXCLK(1), which has the divider value  $2^{**4}$ , use CMU\_FXCLK(0), which has the divider value  $2^{**0}$ . In this case, CM1 has to be configured with value  $2^{**4}$  minus  $2^{**0}$  which is equal to  $2^{**4}-1=16$ .

**Hint** : To get the same length of period, which defines the length of the initial phase, the value for the period in CM0 has to be multiplied by the same value.

A second limitation is that the maximum length of the period, which is configured in CM0, is limited. Using a higher CMU\_FXCLK/CMU\_CLK frequency reduces the maximum possible period.

## 2.43 [GTM\_AI.345] SPE: Incorrect behaviour of direction change control via SPE\_CMD.SPE\_CTRL\_CMD bits

### Description

A direction change ("00" <-> "01") via SPE\_CTRL\_CMD disturbs the increment/decrement of the pat\_ptr resulting in incorrect output patterns not corresponding to the input pattern position. Changing the direction bit in SPE\_CTRL\_CMD can also generate invalid IRQs.

### Scope

SPE, TOM

### Effects

Modifying the direction bit ("00" <-> "01") in SPE\_CTRL\_CMD does not provide the correct output pattern to the BLDC motor. Due to a wrong pat\_ptr position incorrect output patterns will be sent to the motor, which are not correlated to the sensor position.

In addition the SPE logic can generate unpredictable IRQs (perr\_irq, dchg\_irq, bis\_irq).

### Workaround

Do not use SPE\_CTRL\_CMD.

## 2 Functional deviations

Instead reprogram the SPE\_OUT\_PAT register to change the direction.

### **2.44 [GTM\_AI.346] ATOM SOMS mode: Shift cycle is not executed correctly in case the reload condition is deactivated with ATOM[i]\_AGC\_GLB\_CTRL.UPEN = 0**

#### **Description**

ATOM is configured to SOMS continuous mode by setting the following configuration bit-fields:

- ATOM[i]\_CH[x]\_CTRL.MODE=11
- ATOM[i]\_CH[x]\_CTRL.OSM=0
- ATOM[i]\_CH[x]\_CTRL.ARU\_EN=0
- ATOM[i]\_AGC\_GLB\_CTRL.UPEN[x]=0b00

#### **Expected behaviour**

After the counter CN0 reaches CM0, no reload cycle is executed due to the configuration of UPEN=0b00. Instead of a reload cycle a shift cycle has to be executed to ensure an continuous shifting.

#### **Observed behaviour**

Neither a reload cycle nor a shift cycle is executed when the counter CN0 reaches CM0. The shifting stops and the shift register CM1 as well as the output ATOM[i]\_CH[x]\_OUT stays unexpectedly stable for two shift clock cycles whereas the counter CN0 continuously counting further on.

#### **Scope**

ATOM

#### **Effects**

After the counter CN0 reaches CM0 the output stays stable for two shift clock cycles before the next shift will be executed.

#### **Workaround**

Increase the number of bits that have to be shifted out inside CM0 register to the maximum value of 23 to ensure an continuous shifting of all bits of the shift register CM1.

### **2.45 [GTM\_AI.347] TOM/ATOM: Reset of (A)TOM[i]\_CH[x]\_CN0 with TIM\_EXT\_CAPTURE are not correctly synchronized to selected CMU\_CLK/CMU\_FXCLK**

#### **Description**

To reset the counter (A)TOM[i]\_CH[x]\_CN0 (SOMP mode in ATOM), the input signal TIM\_EXT\_CAPTURE can be used by configuration of (A)TOM[i]\_CH[x]\_CTRL.EXT\_TRIG=1 and (A)TOM[i]\_CH[x]\_CTRL.RST\_CCU0=1.

The reset of the counter (A)TOM[i]\_CH[x]\_CN0 should happen synchronously to the internal selected CMU clock CMU\_CLK/CMU\_FXCLK. Therefore a synchronisation stage is implemented to synchronize the input signal TIM\_EXT\_CAPTURE to the internal selected CMU clock CMU\_CLK/CMU\_FXCLK.

It can be observed, that the reset of the counter is done immediately with the occurrence of the input signal TIM\_EXT\_CAPTURE and not as expected synchronously to the selected CMU clock enable CMU\_CLK/CMU\_FXCLK.

---

**2 Functional deviations**

As a consequence of this, the output signal for the compare values 0 and 1 of (A)TOM[i]\_CH[x]\_CM1.CM1 and (A)TOM[i]\_CH[x]\_CM0.CM0 will not be set correctly.

**Scope**

ATOM, TOM

**Effects**

The output signal (A)TOM[i]\_CH[x]\_OUT is not set correctly for the compare values 0 and 1 of the operation register bitfields (A)TOM[i]\_CH[x]\_CM1.CM1 and (A)TOM[i]\_CH[x]\_CM0.CM0.

**Workaround 1**

Do not use clock dividing for the affected (A)TOM channels, so the undivided cluster clock is used. For this configure the control registers in the CMU and CCM to generate non-dividing CMU\_FXCLK and/or CMU\_CLK signals. Select within the (A)TOM the non-dividing CMU\_FXCLK0 (for TOM) and/or CMU\_CLK0..7 (for ATOM) via the settings for CLK\_SRC in the (A)TOM[i]\_CH[x]\_CTRL register(s).

**Workaround 2**

Avoid the compare values 0 and 1 for the operation register bitfields (A)TOM[i]\_CH[x]\_CM1.CM1 and (A)TOM[i]\_CH[x]\_CM0.CM0.

**2.46 [GTM\_AI.349] TOM-SPE: OSM-Pulse width triggered by SPE\_NIPD for selected CMU\_FXCLK not correct****Description**

The SPE\_NIPD signal is used to reset TOM\_CH\_CN0 and to generate a one-shot pulse. When the CMU\_FXCLK of the corresponding TOM\_CH is set to a value unequal to 0, there are two effects observed:

1. the first pulse triggered by SPE\_NIPD is generated with the CMU\_FXCLK(0), while any subsequent pulses are generated with the configured CMU\_FXCLK;
2. the pulses generated with the correct CMU\_FXCLK show no determinism. Some pulses end with CCU\_TRIG1, some with CCU\_TRIG0

**Scope**

TOM, SPE

**Effects**

The OSM-Pulse width triggered by SPE\_NIPD are not correct.

**Workaround**

Use SYS\_CLK by selecting CMU\_FXCLK(0) instead of a value unequal to zero for CMU\_FXCLK.

To reach the same pulse width on the output signal, the value for the period (TOM[i]\_CH[x]\_CM0.CM0) and duty cycle (TOM[i]\_CH[x]\_CM1.CM1) has to be scaled due to the relationship between SYS\_CLK and the needed CMU\_FXCLK.

## 2 Functional deviations

### 2.47 [GTM\_AI.350] TOM-SPE: Update of SPE[i]\_OUT\_CTRL triggered by SPE\_NIPD not working for a delay value 1 in TOM[i]\_CH[x]\_CM1

#### Description

When configured in one-shot mode some TOM channels can initiate a delayed change of register SPE\_OUT\_CTRL. The delay can be configured in TOM[i]\_CH[x]\_CM1 register of the corresponding TOM channel.

#### Expected behaviour

The SPE\_OUT\_CTRL register changed its content after a delay of CMU\_FXCLK cycles which are configured in the TOM channel. For CM1=0, no update is expected, for CM1=1, the update is expected with the next CMU\_FXCLK, for CM1=2, a delay of two CMU\_FXCLK clock cycles is expected.

#### Observed behaviour

For CM1=1, there is no change of SPE\_OUT\_CTRL at all, independent of CMU\_FXCLK.

#### Scope

TOM, SPE

#### Effects

The update of SPE\_OUT\_CTRL register is not executed.

#### Workaround

Use SYS\_CLK by selecting CMU\_FXCLK(0) instead of a value unequal to zero for CMU\_FXCLK.

To get the trigger signal from TOM for the delayed update at the same time, the value for the period (TOM[i]\_CH[x]\_CM0.CM0) and duty cycle (TOM[i]\_CH[x]\_CM1.CM1) has to be scaled due to the relationship between SYS\_CLK and the needed CMU\_FXCLK.

### 2.48 [GTM\_AI.352] ATOM: Wrong reload of data from ARU in SOMS and SOMP mode if TIM\_EXT\_CAPTURE(x) or TRIGIN(x) is selected as clock source

#### Description

ATOM configuration:

- SOMP or SOMS mode (ATOM[i]\_CH[x]\_CTRL.MODE=10<sub>B</sub>/11<sub>B</sub>)
- ARU input stream enabled (ATOM[i]\_CH[x]\_CTRL.ARU\_EN=1)
- TRIGIN(x) or TIM\_EXT\_CAPTURE(x) as selected clock source (ATOM[i]\_CH[x]\_CTRL.CLK\_SRC=1101<sub>B</sub>/1110<sub>B</sub>)

#### Expected behavior in SOMS mode

ATOM Channel in SOMS mode shifts all data provided by ARU.

#### Observed behavior in SOMS mode

An ARU read request is initiated and not cancelled after the first data was received from the ARU. The received data overwrites the previously received data in the ATOM[i]\_CH[x]\_SRy.SRy register (y=0,1).

In SOMS continuous mode with ATOM[i]\_CH[x]\_CTRL.OSM=0, an update of the ATOM[i]\_CH[x]\_CMy.CMy register (y=0,1) from last received data in ATOM[i]\_CH[x]\_SRy.SRy register (y=0,1) is executed at the end of the period.

In SOMS one-shot mode with ATOM[i]\_CH[x]\_CTRL.OSM=1, the ATOM channel stops after data is shifted out which was stored in shift register ATOM[i]\_CH[x]\_CM1.CM1 by the CPU. Data which was transferred via ARU

## 2 Functional deviations

stays in shadow register ATOM[i]\_CH[x]\_SR1.SR1 and will not be reloaded into the shift register; instead the channel stops.

### Expected behavior in SOMP continuous mode

Synchronized to the beginning of a new period ATOM Channel requests new data from ARU. The received values from ARU are stored into the shadow registers. If the actual period is ended the stored values are copied from the shadow registers into the operation registers for the new period. At the same time, a new read request to the ARU is started.

### Observed behavior in SOMP continuous mode

ATOM Channel requests new data from ARU without synchronization to the beginning of a new period. The received values are stored into the shadow registers and then copied directly into the operation registers. The next ARU read request is started immediately without synchronization to the actual period.

SOMP one-shot mode together with the reloading of values via the ARU is not supported and is therefore not affected by this ERRATUM.

### Scope

ATOM

### Effects

For the described modes the reloading and update of new values for the shadow registers from ARU is corrupted.

In the SOMS one-shot mode the channel stops.

### Workaround

If TIM\_EXT\_CAPTURE(x) is to be used as clock source, this can be configured within the CCM as clock source for one of the CMU clock sources. This clock source must then be selected in the ATOM itself.

If TRIGIN(x) is to be used as clock source, the output signal of the ATOM channel, which delivers the trigger signal TRIGIN(x), can be routed to TIM input as AUX\_IN signal. Now the TIM\_EXT\_CAPTURE(x) signal from this TIM module can be used with the same workaround as described before for TIM\_EXT\_CAPTURE(x) clock source. An additional clock delay of 3 cluster clocks would need to be considered for the generation of the TRIGIN(x) source.

## 2.49 [GTM\_AI.353] SPEC-ATOM: Specification of the smallest possible PWM period in SOMP mode wrong, when ARU\_EN=1

### Description

Configuration in use:

- ATOM[i]\_CH[x]\_CTRL.MODE=0b10 (SOMP)
- ATOM[i]\_CH[x]\_CTRL.ARU\_EN=1
- ATOM[i]\_AGC\_GLB\_CTRL.UPEN\_CTRLx=1

### Functionality

When ATOM[i]\_CH[x]\_CTRL.ARU\_EN=1 and ATOM[i]\_AGC\_GLB\_CTRL.UPEN\_CTRLx=1 the PWM period and duty cycle (PWM characteristic) can be reloaded via ARU in SOMP mode. The ATOM generates a PWM on the operation registers ATOM[i]\_CH[x]\_CM0.CM0 and ATOM[i]\_CH[x]\_CM1.CM1 while the new values received via ARU are stored in the shadow registers ATOM[i]\_CH[x]\_SR0.SR0 and ATOM[i]\_CH[x]\_SR1.SR1.

Reloading of the ATOM[i]\_CH[x]\_CM0.CM0 and ATOM[i]\_CH[x]\_CM1.CM1 registers with the values from ATOM[i]\_CH[x]\_SR0.SR0 and ATOM[i]\_CH[x]\_SR1.SR1 takes place, when the old PWM period expires

## 2 Functional deviations

(ATOM[i]\_CH[x]\_CN0.CN0 reaches ATOM[i]\_CH[x]\_CM0.CM0 in up counter mode or ATOM[i]\_CH[x]\_CN0.CN0 reaches 0 in up/down counter mode).

Therefore, it is important, that the new PWM characteristic is available in the shadow registers ATOM[i]\_CH[x]\_SR0.SR0 and ATOM[i]\_CH[x]\_SR1.SR1 before ATOM[i]\_CH[x]\_CN0.CN0 reaches ATOM[i]\_CH[x]\_CM0.CM0 (up counter mode) or 0 (up/down counter mode).

### Problem description

The GTM-IP specification defines as minimal possible PWM period, where the PWM characteristic can be reloaded in a predictable manner so that new data is always available in time at the ATOM channel, to be the ARU round trip time of the specific microcontroller device. This is not correct, because the data needs two additional ARU clock cycles to flow through the ARU from a source to the ATOM channel plus one clock cycle for loading the value from the shadow registers ATOM[i]\_CH[x]\_SR0.SR0 and ATOM[i]\_CH[x]\_SR1.SR1 to the registers ATOM[i]\_CH[x]\_CM0.CM0 and ATOM[i]\_CH[x]\_CM1.CM1.

When the PWM period is smaller than the ARU round trip time plus three ARU clock cycles, the PWM output is not correct.

### Scope

SPEC-ATOM

### Effects

When the ATOM channel operates in SOMP mode and receives updates of PWM period and/or duty cycle via ARU, new PWM period and/or duty cycle values get lost, when the PWM Period is smaller than the ARU round trip time plus one or two ARU clock cycles for the given microcontroller device the PWM Period runs on.

### Recommendation for TC2xx

The PWM period has to be larger than ARU round trip time + 3 ARU clock cycles.

### Recommendation for TC3xx

The PWM period has to be larger than ARU round trip time + 3 ARU clock cycles. Alternatively use ARU dynamic routing, or reduce the value of ARU\_CADDR\_END to a value, which fits the PWM period. So, PWM period greater than ARU\_CADDR\_END + 1 + 3 ARU clock cycles.

## 2.50 [GTM\_AI.358] TOM/ATOM: Synchronous update of working register for RST\_CCU0=1 and UDMODE=01<sub>B</sub> not correct

### Description

TOM/ATOM is configured in SOMP mode with ATOM[i]\_CH[x]\_CTRL.MODE="10" (only for ATOM) and up-down counter mode is enabled by setting of (A)TOM[i]\_CH[x]\_CTRL.UDMODE=01<sub>B</sub>. With the additional configuration of (A)TOM[i]\_CH[x]\_CTRL.RST\_CCU0=1, the counter direction from up to down is changed with the trigger signal from a preceding channel TRIGIN[x] or with the TIM\_EXT\_CAPTURE signal from TIM module.

### Expected behaviour

The synchronous update of the working registers (A)TOM[i]\_CH[x]\_CM0 and (A)TOM[i]\_CH[x]\_CM1 in this configuration shall be done only when the channel counter (A)TOM[i]\_CH[x]\_CN0 reaches zero.

### Observed behaviour

Additionally to the update of the working registers (A)TOM[i]\_CH[x]\_CM0 and (A)TOM[i]\_CH[x]\_CM1 when the channel counter (A)TOM[i]\_CH[x]\_CN0 reaches zero, the update is executed with the selected trigger signal TRIGIN[x] or TIM\_EXT\_CAPTURE(x). This is not expected in this configuration with (A)TOM[i]\_CH[x]\_CTRL.UDMODE=01<sub>B</sub>.

## 2 Functional deviations

### Scope

TOM, ATOM

### Effects

The synchronous update of the working register (A)TOM[i]\_CH[x]\_CM0 and (A)TOM[i]\_CH[x]\_CM1 is done unintentionally with the selected trigger signal TRIGIN[x] or TIM\_EXT\_CAPTURE. As a result, depending on the actual output level, an edge to (A)TOM[i]\_CH[x]\_CTRL.SL could occur.

### Workaround

For settings where the PWM phases are longer than the register access times on target system: Ensure to deliver new data to the associated shadow registers (A)TOM[i]\_CH[x]\_SR0 and (A)TOM[i]\_CH[x]\_SR1 only when the channel counter ATOM[i]\_CH[x]\_CN0 is in down counting phase. The down counting phase is reported by the according interrupt.

The described workaround is only possible for ATOM as long as the ARU interface is disabled and the new shadow register values are delivered by configuration interface and not by ARU interface.

## 2.51 [GTM\_AI.359] TOM: Both edges on TOM\_OUT\_T at unexpected times for RST\_CCU0=1 and UDMODE>0

### Description

TOM channel is configured in up-down counter mode by setting of TOM[i]\_CH[x]\_CTRL.UDMODE>0 and the channel is triggered by a preceding channel or by TIM\_EXT\_CAPTURE with configuration of TOM[i]\_CH[x]\_CTRL.RST\_CCU0=1.

### Expected behaviour

In up-counting phase, the output signal TOM\_OUT is set to SL when CN0 >= CM1 and the second output signal TOM\_OUT\_T has to be set to SL when CN0 >= CM0.

In down-counting phase the output signals has to be set to !SL when CN0 < CM1/CM0.

### Observed behaviour

The second output signal TOM\_OUT\_T is set to SL in upcounting phase when CN0 >= CM0 - 1, which is one CMU clock cycle to early.

When the counter is counting down, the output signal TOM\_OUT\_T is set to !SL when CN0 < CM0 - 1, which is one CMU clock cycle too late.

### Scope

TOM

### Effects

The second output signal TOM\_OUT\_T is set one CMU clock cycle too early in up-counting phase and one CMU clock cycle to late in down-counting phase.

### Workaround

The compare value TOM[i]\_CH[x]\_CM0 for the second output signal TOM\_OUT\_T has to be configured with a value which is greater by one (CM0+1).

## 2 Functional deviations

### 2.52 [GTM\_AI.360] SPEC-(A)TOM: PCM mode (BITREV=1) is only available for UDMODE=0

#### Description

If TOM/ATOM channel is configured in PCM mode with (A)TOM[i]\_CH[x]\_CTRL.BITREV=1, the channel may be configured in up-counting mode only with (A)TOM[i]\_CH[x]\_CTRL.UDMODE=0.

Up-down counting mode ((A)TOM[i]\_CH[x]\_CTRL.UDMODE>0) is not supported for PCM mode.

#### Scope

TOM, ATOM

#### Effects

The user is not aware that the combination of PCM mode together with up-down counting mode is not supported and may not be used.

#### Workaround

Do not use the combination of PCM mode together with up-down counting mode.

### 2.53 [GTM\_AI.361] IRQ: Missing pulse in single-pulse interrupt mode on simultaneous interrupt and clear event

#### Description

In single-pulse interrupt mode ([MODULE]\_IRQ\_MODE = 0b11) only the first interrupt event of the interrupt bits of the interrupt notify register inside this module generates a pulse on the output signal IRQ\_line, if the associated interrupt is enabled ([MODULE]\_IRQ\_EN=1). All further interrupt events have no effect on the output signal IRQ\_line until all enabled interrupts are cleared, except when an interrupt and a clear event (HW\_clear or a SW\_clear) occur at the same time.

#### Expected behaviour

On simultaneous occurrence of an interrupt and clear event, a pulse on the output signal IRQ\_line is generated.

#### Observed behaviour

If the associated notify register bit of the interrupt event is not set and another bit of the same notify register is set and this interrupt is enabled, no pulse on the output signal IRQ\_line is generated.

All modules ([MODULE]) are affected by this ERRATUM, which are able to generate interrupts and which have multiple interrupt sources which are ORed to the output. Not affected are the modules DPLL and ARU.

#### Scope

IRQ

#### Effects

Missing pulse on interrupt signal IRQ\_line.

All modules, which deliver an interrupt signal and have more than one internal interrupt source which are ORed are affected. The only exceptions are the modules ARU and DPLL.

## 2 Functional deviations

### Workaround

On a SW clear prevent HW clear events and read the interrupt notify register to check on new interrupts without a received interrupt pulse on IRQ\_line. In this case repeat the SW clear step to enable interrupt generation again.

When disabling the HW clear is not an option refrain from using the single-pulse interrupt mode.

### 2.54 [GTM\_AI.364] ATOM: ARU read request does not start at expected timepoint in UDMODE = 1 and UDMODE = 3

#### Description

ATOM is configured in SOMP continuous up-down counter mode with UDMODE = 1 or 3 and ARU interface is enabled by setting of ARU\_EN = 1.

#### Expected behavior

A new ARU read request has to be started always after the operation registers are updated from their shadow registers. This depends on the UDMODE configuration:

- UDMODE = 1: New ARU read request after CN0 changes the count direction from down to up
- UDMODE = 2: New ARU read request after CN0 changes the count direction from up to down
- UDMODE = 3: New ARU read request in both cases

#### Observed behavior

A new ARU read request is always started when the counter CN0 changes the count direction from up to down, independently from UDMODE configuration:

- UDMODE = 1: New ARU read request after CN0 changes the count direction from up to down
- UDMODE = 2: Works as expected
- UDMODE = 3: New ARU read request after CN0 changes the count direction from up to down

#### Scope

ATOM

#### Effects

The effect depends on the UDMODE configuration:

- UDMODE = 1: The remaining time, from starting a ARU read request until new data from ARU should be received is only half of the defined PWM period instead of the full PWM period
- UDMODE = 3: No new ARU read request is started when the counter CN0 changes the count direction from down to up and therefore no new data can be delivered in this case

#### Workaround

- Workaround for UDMODE = 1  
The PWM period length in up-down counter mode has to be double the length as the ARU round trip cycle (plus 3 ARU clock cycles).
- Workaround for UDMODE = 3  
Use AEI interface for reloading new shadow register values instead of ARU.

## 2 Functional deviations

### 2.55 [GTM\_AI.370] TOM/ATOM: Unexpected reset of CN0 in up-down counter mode and CM0 = 2

#### Description

TOM/ATOM is configured in SOMP mode with  $ATOM[i]_{CH[x]}_{CTRL.MODE} = 10_B$  (only for ATOM) and up-down counter mode is enabled by setting of  $(A)TOM[i]_{CH[x]}_{CTRL.UDMODE} \neq 00_B$ .

#### Expected behavior

In this case, the counter CN0 changes its count direction from up to down either until CN0 reaches CM0-1 for  $RST\_CCU0 = 0$  or with the selected trigger signal TRIGIN ( $EXT\_TRIG = 0$ ) or EXT\_TRIGIN ( $EXT\_TRIG = 1$ ) for  $RST\_CCU0 = 1$ .

#### Observed behavior

There are three different configuration scenarios, where the counter CN0 is unexpectedly reset.

1. In case of  $RST\_CCU0 = 0$ :  
The period value inside CM0 is configured to 2 and then reconfigured to a value greater than 2. After the counter CN0 starts incrementing and reaches value 1, CN0 is once reset to 0 unexpectedly, before it starts incrementing again.
2. In case of  $RST\_CCU0 = 1$  and  $EXT\_TRIG = 0$ :  
The TRIGIN signal from a preceding channel is used to reset the count direction of CN0. After the period value CM0 of the preceding channel is reconfigured from value 2 to a greater value, CN0 of this channel, which is triggered by the preceding channel, is once reset to 0 similar to the first scenario, which happens in the preceding channel.
3. In case of  $RST\_CCU0 = 1$  and  $EXT\_TRIG = 1$ :  
The EXT\_TRIGIN signal from TIM module is used to reset the count direction of CN0. If the EXT\_TRIGIN signal occurs while the counter CN0 is incrementing and reaches the value 1, CN0 is once reset unexpectedly. However, there is already no deterministic dependency between the EXT\_TRIGIN signal and the reset of CN0.

#### Scope

TOM, ATOM

#### Effects

Unexpected reset of the counter CN0.

#### Workaround

No workaround available. The following limitations have to be considered.

For scenario 1 and 2:

- Do not use value 2 for the period, which is configured inside CM0

For scenario 3:

- Do not use EXT\_TRIGIN as trigger signal to change the count direction in up-down counter mode

### 2.56 [GTM\_AI.374] SPEC-ATOM: Statement on timing of duty cycle output level change not correct for SOMP up/down-counter mode

#### Description

The duty cycle output level is determined by  $ATOM[i]_{CH[x]}_{CTRL.SL}$  bit. The specification describes in the section "ATOM Signal output mode PWM (SOMP)" of the GTM chapter in the AURIX™ TC3xx User's Manual, that

## 2 Functional deviations

"the duty cycle output level can be changed during runtime by writing the new duty cycle level into SL bit of the channels configuration register". Further, it is mentioned that "the new signal level becomes active for the next trigger CCU\_TRIGx (since bit SL is written)".

However, the timing specification in the second part of the statement is only valid for the SOMP in up-counter mode. When the ATOM is configured in SOMP up/down-counter mode, the new signal level becomes immediately active, when the ATOM[i]\_CH[x]\_CTRL.SL bit is written.

### Scope

ATOM

### Effects

When the ATOM channel is configured in SOMP up/down-counter mode, a change of bit ATOM[i]\_CH[x]\_CTRL.SL will be visible immediately after the value is written by software and not, as described in the specification, with the next compare match event of one of the CCUx compare units.

### Workaround

No workaround for SOMP up/down-counter mode. Use SOMP up-counter mode, if update of SL-Bit needed during runtime.

## 2.57 [GTM\_AI.375] ATOM: Data from ARU are read only once in SOMC mode even though ARU blocking mode is disabled while FREEZE = 1 and ENDIS = 0

### Description

ATOM is configured in SOMC mode and ARU input stream is enabled and ARU blocking mode is disabled.

Configuration register setting:

ATOM[i]\_CH[x]\_CTRL.MODE == 01<sub>B</sub> (SOMC mode)

ATOM[i]\_CH[x]\_CTRL.ARU\_EN == 1<sub>B</sub> (ARU input stream enabled)

ATOM[i]\_CH[x]\_CTRL.ABM == 0<sub>B</sub> (ARU blocking mode disabled)

### Expected behavior

If the channel gets disabled while ATOM[i]\_CH[x]\_CTRL.FREEZE is set, a pending ARU read request will still be held active, even if the current request is served from ARU with valid data. This is the expected non-blocking behavior.

### Observed behavior

If the channel gets disabled while ATOM[i]\_CH[x]\_CTRL.FREEZE is set and afterward the ARU read request is served by an ARU read valid, the ARU read request is reset and no more data is requested from ARU interface. This corresponds to a blocking behavior.

### Scope

ATOM

### Effects

In SOMC mode and activated FREEZE mode, reading new compare values stops after the first received data instead of continuing data reads.

## 2 Functional deviations

### Workaround

Instead of using the ARU interface for reloading new compare values while the channel is in FREEZE mode, the configuration interface can be used to deliver the new compare values.

If DPLL is used as data source for the ATOM compare values, an MCS channel has to be used to first read the data from DPLL by ARU interface and afterward to write the data via MCS master interface to ATOM. The used MCS module has to be in the same cluster as the ATOM module.

### 2.58 [GTM\_AI.376] TOM/ATOM: Interrupt trigger signals CCU0TC\_IRQ and CCU1TC\_IRQ are delayed by one CMU\_CLK period related to the output signals

#### Description

Interrupt trigger signals CCU0TC\_IRQ and CCU1TC\_IRQ are delayed by one CMU\_CLK period if the following configurations are used:

1. Both CCU0TC\_IRQ and CCU1TC\_IRQ are affected (ATOM: in SOMP mode) when the channel is configured in up-down counter mode ((A)TOM[i]\_CH[x]\_CTRL.UDMODE > 0)
2. CCU1TC\_IRQ only is affected (ATOM: in SOMP mode) when the channel is configured in up-counter mode ((A)TOM[i]\_CH[x]\_CTRL.UDMODE == 0) and (A)TOM[i]\_CH[x]\_CTRL.SR0\_TRIG is enabled

#### Scope

ATOM, TOM

#### Effects

Interrupt signals CCU0TC\_IRQ and CCU1TC\_IRQ are raised with a delay of one CMU\_CLK period.

Depending on the CMU\_CLK period related to system frequency outside of GTM this can be an issue or none at all.

#### Workaround

None.

### 2.59 [GTM\_AI.406] (A)TOM: FREEZE mode has no effect on (A)TOM\_OUT\_T in up-down counter mode with RST\_CCU0 = 1

#### Description

The channel is set into FREEZE mode while it is configured in up-down counter mode and triggered by a preceding channel or by TIM\_EXT\_CAPTURE.

Configuration for TOM:

TOM[i]\_CH[x]\_CTRL.UDMODE > 0

TOM[i]\_CH[x]\_CTRL.RST\_CCU0 = 1

TOM[i]\_CH[x]\_CTRL.FREEZE = 1 (FREEZE mode)

TOM[i]\_TGC[g]\_ENDIS\_STAT.ENDIS\_STAT = 0

Configuration for ATOM:

ATOM[i]\_CH[x]\_CTRL.MODE = 10<sub>B</sub> (SOMP mode)

ATOM[i]\_CH[x]\_CTRL.UDMODE > 0

ATOM[i]\_CH[x]\_CTRL.RST\_CCU0 = 1

ATOM[i]\_CH[x]\_CTRL.FREEZE = 1 (FREEZE mode)

## 2 Functional deviations

ATOM[i]\_AGC\_ENDIS\_STAT.ENDIS\_STAT= 0

### Expected behavior

In FREEZE mode when the channel is disabled, it is expected that the output signal (A)TOM\_OUT as well as (A)TOM\_OUT\_T has to keep its last value.

### Observed behavior

In FREEZE mode when the channel is disabled, the output signal (A)TOM\_OUT\_T is set to the inverted signal level (ATOM[i]\_CH[x]\_CTRL\_SOMP.SL, TOM[i]\_CH[x]\_CTRL.SL) and does not keep its last value.

### Scope

TOM, ATOM

### Effects

Output signal (A)TOM\_OUT\_T is set to inverse signal level (ATOM[i]\_CH[x]\_CTRL\_SOMP.SL, TOM[i]\_CH[x]\_CTRL.SL) and does not keep the value.

### Workaround

None.

## 2.60 [GTM\_AI.408] (A)TOM-RTL: Missing edge on output signal (A)TOM\_OUT when CN0 is reset with force update event

### Description

The channel is configured in continuous up-counter mode. Then a new period is started with a force update event and reset of CN0 is activated.

Configuration for TOM:

TOM[i]\_CH[x]\_CTRL.UDMODE = 0

TOM[i]\_TGC[g]\_FUPD\_CTRL.FUPD\_CTRL[k] = 10<sub>B</sub>

TOM[i]\_TGC[g]\_FUPD\_CTRL.RSTCN0\_CH[k] = 10<sub>B</sub>

Configuration for ATOM:

ATOM[i]\_CH[x]\_CTRL.MODE = 10<sub>B</sub> (SOMP mode)

ATOM[i]\_CH[x]\_CTRL.UDMODE = 0

ATOM[i]\_AGC\_FUPD\_CTRL.FUPD\_CTRL[k] = 10<sub>B</sub>

ATOM[i]\_AGC\_FUPD\_CTRL.RSTCN0\_CH[k] = 10<sub>B</sub>

### Expected behavior

After the counter (A)TOM[i]\_CH[x]\_CN0.CN0 has been reset and therefore a new period has to be started and the output signal (A)TOM\_OUT has to be set immediately to the SL value (ATOM[i]\_CH[x]\_CTRL\_SOMP.SL, TOM[i]\_CH[x]\_CTRL.SL), and after the counter reaches (A)TOM[i]\_CH[x]\_CM1.CM1, an edge on (A)TOM\_OUT to the inverted SL value (ATOM[i]\_CH[x]\_CTRL\_SOMP.SL, TOM[i]\_CH[x]\_CTRL.SL) is expected.

### Observed behavior

An edge on the output signal (A)TOM\_OUT to the SL value (ATOM[i]\_CH[x]\_CTRL\_SOMP.SL, TOM[i]\_CH[x]\_CTRL.SL) at the beginning of the new period does not happen. Instead, the output signal (A)TOM\_OUT holds its last value.

A second observation is in case the SL value (ATOM[i]\_CH[x]\_CTRL\_SOMP.SL, TOM[i]\_CH[x]\_CTRL.SL) changes synchronously together with the force update event, an edge on (A)TOM\_OUT to the inverted SL value

---

**2 Functional deviations**

(ATOM[i]\_CH[x]\_CTRL\_SOMP.SL, TOM[i]\_CH[x]\_CTRL.SL) when (A)TOM[i]\_CH[x]\_CN0.CN0 reaches (A)TOM[i]\_CH[x]\_CM1.CM1 does not happen.

**Scope**

TOM, ATOM

**Effects**

Missing edge and false output signal level on (A)TOM\_OUT.

**Workaround**

None.

### **2.61 [GTM\_AI.410] GTM\_AEI: The AEI bridge might not execute an accepted write transaction**

**Description**

If the AEI Bridge operates in pipeline mode while a soft-reset is issued (writing BRIDGE\_MODE.BRG\_RST = '1'), upcoming write transactions primed in the buffer although accepted may never be actually executed. The maximum number of non-executed transactions depends on the buffer depth (BRIDGE\_BUFF\_DPT).

**Scope**

GTM\_AEI

**Effects**

Write transaction is signaled to be accepted but will never be executed.

**Workaround**

Issue a read access to any address after the soft reset.

### **2.62 [GTM\_AI.411] A change of the BRIDGE\_MODE register might be delayed indefinitely**

**Description**

After a write access to the BRIDGE\_MODE register, the bit-fields BRG\_MODE and BYPASS\_SYNC will not be updated until the transaction buffer is empty. In split mode the bridge allows new transactions to be added to the buffer, even when an update of these bits is pending.

Polling the register in split mode might prevent the buffer from getting empty and as a result prevents the actual update of the described bit-fields.

**Note:** *Bit-field BYPASS\_SYNC is not specified for TC2xx.*

**Scope**

GTM\_AEI

**Effects**

Frequently polling the BRIDGE\_MODE register ends in a deadlock.

## 2 Functional deviations

### Workaround 1

After every failed attempt to read back the new values, increase the wait time before issuing the next read transaction.

### Workaround 2

Use standard mode (which is entered by setting AEI\_PIPE and AEI\_SPLIT at zero while asserting AEI\_SEL) to write and read back the affected bits.

**Note:** *This workaround is only possible in devices without AXIS.*

## 2.63 [GTM\_AI.419] TIM: Potentially wrong capture values

### Description

Configuration: The TIM channel is configured in TIEM, TIPM, TGPS or TSSM mode by setting of  $TIM[i]_{CH[x]}_{CTRL}.TIM\_MODE = \{010_B, 011_B, 101_B, 110_B\}$ . The TIM channel is disabled ( $TIM[i]_{CH[x]}_{CTRL}.TIM\_EN=0$ ) and later enabled again ( $TIM[i]_{CH[x]}_{CTRL}.TIM\_EN = 1$ ).

### Expected behavior for TIEM/TIPM/TGPS mode

The registers  $TIM[i]_{CH[x]}_{CNT}$ ,  $TIM[i]_{CH[x]}_{ECNT}.ECNT[15:1]$ ,  $TIM[i]_{CH[x]}_{GPR0}$  and  $TIM[i]_{CH[x]}_{GPR1}$  are set to their reset values. In case of an input signal edge or an input capture event or an active selected CMU clock (TGPS mode) at the same time as the channel is enabled, this event has to be taken into account and the  $TIM[i]_{CH[x]}_{CNT}$  register must be updated/incremented based on its reset value. Due to this a capture event can happen depending on the configured TIM mode and the register values.

### Expected behavior for TSSM mode

The registers  $TIM[i]_{CH[x]}_{CNT}$ ,  $TIM[i]_{CH[x]}_{ECNT}.ECNT[15:1]$ ,  $TIM[i]_{CH[x]}_{GPR0}$  and  $TIM[i]_{CH[x]}_{GPR1}$  are set to their initial values. The initial value for  $TIM[i]_{CH[x]}_{CNT}$  register depends on  $TIM[i]_{CH[x]}_{CTRL}.ISL$  and  $TIM[i]_{CH[x]}_{CNTS}.CNTS(22)$ . If  $TIM[i]_{CH[x]}_{CNTS}.CNTS(22)$  is set to 0 and  $TIM[i]_{CH[x]}_{CTRL}.ISL$  is set to 0 the initial value of  $TIM[i]_{CH[x]}_{CNT}$  is 0x000000. An input signal event simultaneously to the channel enable is not taken into account.

### Observed behavior for TIEM/TIPM/TGPS mode

If no input signal event or input capture event or active selected CMU clock (TGPS mode) occurs, the registers  $TIM[i]_{CH[x]}_{CNT}$ ,  $TIM[i]_{CH[x]}_{ECNT}.ECNT[15:1]$ ,  $TIM[i]_{CH[x]}_{GPR0}$  and  $TIM[i]_{CH[x]}_{GPR1}$  are set to their reset values as expected.

If an input signal event or an input capture event or an active selected CMU clock (TGPS mode) occurs at same time as the channel gets enabled, the  $TIM[i]_{CH[x]}_{CNT}$  register continues to count (or update) based on the previous (old) value. As a result, a capture could be performed too early and/or with the wrong values. The  $TIM[i]_{CH[x]}_{ECNT}.ECNT[15:1]$  register is set to its reset value as expected.

### Observed behavior for TSSM mode

The register  $TIM[i]_{CH[x]}_{CNT}$  is not set to its initial value of 0x000000 on channel enabling when  $TIM[i]_{CH[x]}_{CNTS}.CNTS(22)$  is set to 0 and  $TIM[i]_{CH[x]}_{CTRL}.ISL$  is set to 0.

**Note:** *The TIM channel modes TPWM, TPIM and TBCM ( $TIM[i]_{CH[x]}_{CTRL}.TIM\_MODE = \{000_B, 001_B, 100_B\}$ ) are not affected.*

## 2 Functional deviations

### Scope

TIM

### Effects

TIM[i]\_CH[x]\_CNT register is not reset and the wrong values could be captured into TIM[i]\_CH[x]\_GPR0 and TIM[i]\_CH[x]\_GPR1 registers.

### Workaround 1

Reset the TIM channel by setting of TIM[i]\_RST.RST\_CH[x] = 1 before enabling the TIM channel.

### Workaround 2

The following sequence has to be executed on the disabled channel, but before the actual enabling of the channel to ensure that the TIM[i]\_CH[x]\_CNT register is set to its reset value when the channel is enabled.

1. Configure TIM[i]\_CH[x]\_CNTS = 0
2. Enable the TIM channel with the following configuration inside the TIM[i]\_CH[x]\_CTRL register:
  - TIM\_EN = 1
  - TIM\_MODE = 101<sub>B</sub> (TGPS)
  - ISL = 1
  - OSM = 1
  - ARU\_EN = 0
  - select a fast CMU\_CLK\_RES, e.g. CLK\_SEL = 000<sub>B</sub>
3. Wait until an edge on the selected CMU\_CLK\_RES occurs. This can be observed on the NEWVAL IRQ notify register. This event sets the TIM[i]\_CH[x]\_CNT register to its reset value
4. Disable TIM channel (TIM[i]\_CH[x]\_CTRL.TIM\_EN = 0)
5. Configure the former TIM channel configuration in TIM[i]\_CH[x]\_CTRL register and enable the TIM channel again

## 2.64 [GTM\_AI.421] GTM\_AEI: Changing BRIDGE\_MODE.MSK\_WR\_RSP in pipeline mode can lead to violation of pipeline protocol

### Description

In pipeline mode, a reconfiguration of the BRIDGE\_MODE.MSK\_WR\_RSP directly after another write transaction can lead to a hang of following write transactions by not setting the AEI\_READY.

**Note:** Please also check on errata GTM\_AI.487 and GTM\_AI.488.

### Scope

GTM\_AEI

### Effects

Transaction not terminated according to protocol, user might be stuck waiting for AEI\_READY to be set.

### Workaround

Make sure the transaction preceding the write of BRIDGE\_MODE.MSK\_WR\_RSP is a read transaction.

---

## 2 Functional deviations

### 2.65 [GTM\_AI.429] TIM: Missing glitch detection interrupt event

#### Description

Configuration:

TIM filter is configured in immediate edge propagation mode by setting `TIM[i]_CH[x]_CTRL.FLT_MODE_RE = 0` or `TIM[i]_CH[x]_CTRL.FLT_MODE_FE = 0`. The filter is enabled by setting `TIM[i]_CH[x]_CTRL.FLT_EN = 1`.

#### Expected behavior

As long as the filter threshold is not reached and the input signal level unexpectedly changes (it is an input glitch occurs), the internal glitch detection interrupt event signal (`TIM_GLITCHDET_IRQ`) should have a HIGH pulse of one cluster clock cycle.

#### Observed behavior

When the input signal glitch occurs at the same time the filter counter reaches its threshold, the internal glitch detection interrupt event signal (`TIM_GLITCHDET_IRQ`) does not occur.

#### Scope

TIM

#### Effects

The `TIM[i]_CH[x]_IRQ_NOTIFY.GLITCHDET` bit is not set. Thus, no interrupt is triggered. Furthermore, the external capture source `EXT_CAPTURE(x)` is not triggered if its source is set to `TIM_GLITCHDET_IRQ`.

#### Workaround

The filter counter threshold can be set to the next higher value. Thus, a former not detected glitch would be detected. In that case, the output signal would be changed (one clock cycle longer) when the input signal is a single cycle pulse.

### 2.66 [GTM\_AI.430] TIM: Unexpected increment of filter counter

#### Description

Configuration: TIM filter is configured in immediate edge propagation mode by setting `TIM[i]_CH[x]_CTRL.FLT_MODE_RE = 0` and/or `TIM[i]_CH[x]_CTRL.FLT_MODE_FE = 0`. The filter is enabled by setting `TIM[i]_CH[x]_CTRL.FLT_EN = 1`. The filter counter threshold is set to zero by setting `TIM[i]_CH[x]_CTRL.FLT_RE.FLT_RE = 0` and/or `TIM[i]_CH[x]_CTRL.FLT_FE.FLT_FE = 0`.

#### Expected behavior

When the input signal level changes, the filter counter should not increment.

#### Observed behavior

When the input signal level changes, the filter counter increments by one and is not reset.

#### Scope

TIM

#### Effects

If an input edge occurs during the acceptance time, the following output signal change will happen one or more selected CMU clock cycles earlier than expected. This depends on the initial configuration and the

---

**2 Functional deviations**

reconfiguration of the filter mode and the filter counter threshold. If the filter mode for both edges is configured to immediate edge propagation and both filter counter thresholds are set to zero, the counter falsely can count up to a higher value than one without resetting. If one or both filter modes and/or thresholds are reconfigured during the application, the higher count of the filter counter can lead to a difference of more than one CMU clock cycle between the expected and actual output signal change at the next occurring input edge. If only one filter counter threshold is set to zero, the difference of the expected and actual output signal change is one CMU clock cycle.

**Workaround**

If acceptable, use a threshold greater than zero. Otherwise there is no workaround available. However, there is a possibility of minimizing the absolute error, deriving from this bug. If possible, a faster CMU clock can be selected. This leads to a shorter absolute time difference between the expected and actual output signal change. Additionally when applying this, the filter counter thresholds need to be assimilated proportionally in order to make the filter work as before.

**2.67 [GTM\_AI.431] TIM: Glitch detection interrupt event of filter is not a single cycle pulse****Description**

Configuration: The TIM filter must be enabled by setting `TIM[i]_CH[x]_CTRL.FLT_EN = 1`.

**Expected behavior**

As long as the filter threshold is not reached and the input signal level changes unexpectedly, the glitch detection interrupt event signal (`TIM_GLITCHDET_IRQ`) should have a single cycle HIGH pulse.

**Observed behavior**

When the input signal level changes unexpectedly for longer than one clock cycle, the glitch detection interrupt event signal (`TIM_GLITCHDET_IRQ`) is HIGH as long as the unexpected signal change is present.

**Scope**

TIM

**Effects**

- Effect 1: The longer lasting HIGH signal of the glitch detection interrupt event signal (`TIM_GLITCHDET_IRQ`) may lead to an unexpected behavior within the GTM only if `TIM_GLITCHDET_IRQ` is used for the external capture signal `EXT_CAPTURE(x)`.
- Effect 2: If the related interrupt notify register (`TIM[i]_CH[x]_IRQ_NOTIFY`) is cleared by software while the `TIM_GLITCHDET_IRQ` signal is still HIGH, the interrupt will unexpectedly retrigger.

**Workaround**

No workaround in hardware.

For the unexpected retrigger of the interrupt directly after an interrupt clear step, the interrupt routine has to consider that the interrupt might be invalid.

## 2 Functional deviations

### 2.68 [GTM\_AI.442] GTM Top Level: GTM\_HALT mode not functional when cluster 0 clock is disabled

#### Description

When entering the halt mode (GTM\_HALT activated by `gtm_halt_req=1`), all functional operation of the GTM shall stop. In case if the cluster 0 is switched off by `GTM_CLS_CLK_CFG.CLS_CLK_DIV0=00B`, the GTM operation will not stop for all clusters > 0 correctly. Operation of GTM functionality will partly continue, which is unexpected.

#### Scope

GTM Top Level

#### Effects

Functional operation of GTM is not completely stopped although `gtm_halt_req` is set and GTM\_HALT mode is started. This will lead to an undefined state of the GTM and the GTM cannot be reliably resumed out of this state.

#### Workaround

No workaround available.

Make sure to turn on the cluster 0 clock before setting `gtm_halt_req=1` to switch to GTM\_HALT mode.

### 2.69 [GTM\_AI.454] (A)TOM: No output if trigger generation feature is used

#### Description

For trigger generation (`((A)TOM[i]_CH[x]_CTRL.SR0_TRIG=1)` in up-down counter mode (`((A)TOM[i]_CH[x]_CTRL.UDMODE>0)`) neither a new PWM on `(A)TOM_OUT` nor an additional trigger output on `(A)TOM_OUT_T` is generated if `(A)TOM[i]_CH[x]_SR0.SR0` register is configured to zero.

#### Scope

TOM, ATOM

#### Effects

No module output signals `(A)TOM_OUT` and `(A)TOM_OUT_T` are generated.

#### Workaround

A second `(A)TOM` channel `z` can be used to generate a trigger signal on `(A)TOM_OUT_T` for `(A)TOM[i]_CH[z]_SR0.SR0=0`. The channel has to be configured in up counter mode (`((A)TOM[i]_CH[z]_CTRL.UDMODE=0)`) with a period value calculated by `(A)TOM[i]_CH[x]_CM0.CM0*2-2` related to the period value of the first channel `x`. Both channels have to be started synchronously via the TGC/AGC mechanisms.

### 2.70 [GTM\_AI.462] (A)TOM: Missing CCU0TC\_IRQ interrupt signal

#### Description

Configuration:

## 2 Functional deviations

The channel is configured in SOMP (ATOM) up-counter mode with up/down counter mode disabled ((A)TOM[i]\_CH[x]\_CTRL.UDMODE=0) or not existing and triggering by a preceding channel with configuration of (A)TOM[i]\_CH[x]\_CTRL.RST\_CCU0=1.

### Expected behavior

When the counter (A)TOM[i]\_CH[x]\_CN0.CN0 reaches the value of (A)TOM[i]\_CH[x]\_CM0.CM0, the interrupt signal CCU0TC\_IRQ must be triggered.

### Observed behavior

In the first period after (A)TOM[i]\_CH[x]\_CM0.CM0 is changed to the value 0 or 1, no CCU0TC\_IRQ interrupt signal is triggered.

**Note:** *When the second period starts after (A)TOM[i]\_CH[x]\_CM0.CM0 is changed to the value 0 or 1 and stays at that value, then the CCU0TC\_IRQ interrupt signal generation works correctly.*

### Scope

TOM, ATOM

### Effects

Interrupt signal CCU0TC\_IRQ is not triggered.

### Workaround

No workaround available.

It needs to be checked if the application can accept the interrupt occurring with the second period.

## 2.71 [GTM\_AI.465] (A)TOM: Missing CCU0TC\_IRQ interrupt signal for UDMODE > 0

### Description

Configuration:

The channel is configured in SOMP (ATOM) up/down counter mode with (A)TOM[i]\_CH[x]\_CTRL.UDMODE>0 and will be triggered by a preceding channel with configuration of (A)TOM[i]\_CH[x]\_CTRL.RST\_CCU0=1.

### Expected behavior

When the counter (A)TOM[i]\_CH[x]\_CN0.CN0 reaches in the up-counting phase the value of (A)TOM[i]\_CH[x]\_CM0.CM0, the interrupt signal CCU0TC\_IRQ must be triggered.

### Observed behavior

In the first period after (A)TOM[i]\_CH[x]\_CM0.CM0 is changed to the value 0, the CCU0TC\_IRQ interrupt signal is triggered but not in the following periods with unchanged value of (A)TOM[i]\_CH[x]\_CM0.CM0=0.

A second observation is that the CCU0TC\_IRQ interrupt signal is not triggered in the first period after the value of (A)TOM[i]\_CH[x]\_CM0.CM0 is changed from 0 to 1.

**Note:** *in this case, the CCU0TC\_IRQ interrupt is triggered in the following periods with unchanged value of 1 for (A)TOM[i]\_CH[x]\_CM0.CM0.*

### Scope

TOM, ATOM

---

## 2 Functional deviations

### Effects

Interrupt signal CCU0TC\_IRQ is not triggered.

### Workaround

No workaround available.

If applicable use the interrupt indication from the preceding channel, which is always generated half a period earlier.

## 2.72 [GTM\_AI.466] TOM: Unexpected behavior of TOM\_OUT\_T for UDMODE>0

### Description

Configuration:

The channel is configured in up-down counter mode with TOM[i]\_CH[x]\_CTRL.UDMODE>0 and will be triggered by a preceding channel with configuration of TOM[i]\_CH[x]\_CTRL.RST\_CCU0=1.

### Expected behavior

The output signal TOM\_OUT\_T has to be set to TOM[i]\_CH[x]\_CTRL.SL value as long as the condition TOM[i]\_CH[x]\_CN0.CN0 >= TOM[i]\_CH[x]\_CM0.CM0 is true.

### Observed behavior for TOM[i]\_CH[x]\_CM0.CM0=0

The output signal TOM\_OUT\_T is set to TOM[i]\_CH[x]\_CTRL.SL value only for one clock period of the selected CMU clock when TOM[i]\_CH[x]\_CN0.CN0 has reached 0. Afterwards TOM\_OUT\_T is set unexpectedly to the inverted value of TOM[i]\_CH[x]\_CTRL.SL.

### Observed behavior for TOM[i]\_CH[x]\_CM0.CM0=1

An unexpected pulse on the output signal TOM\_OUT\_T with the length of one clock period of the selected CMU clock to the inverted value of TOM[i]\_CH[x]\_CTRL.SL can be observed when the trigger input signal TRIGIN occurs and the counter TOM[i]\_CH[x]\_CN0.CN0 starts to count down.

### Scope

TOM

### Effects

Output signal TOM\_OUT\_T behaves not as expected.

### Workaround

No workaround available.

## 2.73 [GTM\_AI.487] GTM\_AEI: Changing BRIDGE\_MODE[2:0] in pipeline mode can lead to violation of pipeline protocol

### Description

The issue from erratum GTM\_AI.421 (“GTM\_AEI: Changing BRIDGE\_MODE.MSK\_WR\_RSP in pipeline mode can lead to violation of pipeline protocol”) not only appears when BRIDGE\_MODE.MSK\_WR\_RSP changes, but also

**2 Functional deviations**

when it stays '1' while the other configuration bit-fields in BRIDGE\_MODE.BYPASS\_SYNC and/or BRIDGE\_MODE.BRG\_MODE change.

Please also check on erratum GTM\_AI.488

**Scope**

GTM\_AEI

**Effects**

Transaction not terminated according to protocol, user might be stuck waiting for AEI\_READY to be set.

**Workaround**

Make sure the transaction preceding the write of the mentioned BRIDGE\_MODE bit-fields is a read transaction. This workaround matches the workaround from GTM\_AI.421.

## **2.74 [GTM\_AI.488] GTM\_AEI: Turning off BRIDGE\_MODE.MSK\_WR\_RSP in asynchronous mode might lead to following transactions being corrupted**

**Description**

If the AEI bridge operates in asynchronous mode and in pipelined protocol, with Mask-Write-Response turned on (BRIDGE\_MODE[2:0]=011<sub>B</sub>) and the BRIDGE\_MODE.MSK\_WR\_RSP is turned off (by writing BRIDGE\_MODE[2:0]=001<sub>B</sub>), the following transaction might be corrupted by the AEI\_READY not being set.

This is an issue like in GTM\_AI.421 and GTM\_AI.487 but a different workaround is needed.

**Scope**

GTM\_AEI

**Effects**

Transaction not terminated according to protocol, user might be stuck waiting for AEI\_READY to be set.

**Workaround**

Change BRIDGE\_MODE.MSK\_WR\_RSP together with setting the software reset (pipeline writing BRIDGE\_MODE[16:0]=10001<sub>H</sub>).

## **2.75 [GTM\_AI.516] SPE-RTL: IRQ raised on disabled inputs**

**Description**

The inputs for the interrupt generation of the SPE[i]\_IRQ\_NOTIFY.SPE\_PERR are not fed by the masked input signals. Hence, an interrupt SPE[i]\_IRQ\_NOTIFY.SPE\_PERR will occur when there is a pattern mismatch detected on the corresponding TIM channels beside active masking (SPE[i]\_CTRL\_STAT.SIE(k)=0).

**Scope**

SPE

**Effects**

An interrupt will be raised on masked input signals instead of ignoring these.

## 2 Functional deviations

### Workaround

Do not toggle (it is not used) the TIM channels that are disabled on the input side of the SPE.

## 2.76 [GTM\_AI.517] (A)TOM: Missing edge on output signal (A)TOM\_OUT

### Description

If an (A)TOM channel is configured to be triggered by a previous channel by setting of (A)TOM[i]\_CH[x]\_CTRL.RST\_CCU0=1 (SOMP mode in ATOM) and there is a pipeline/synchronization register within the trigger chain between the triggering channel and the triggered channel, the edge to the inverse SL at the output signal (A)TOM\_OUT is not generated for (A)TOM[i]\_CH[x]\_CM1.CM1<2 and (A)TOM[i]\_CH[x]\_CM0.CM0>(A)TOM[i]\_CH[x]\_CM1.CM1. The problem only occurs if the selected clock resolution for the triggered channel has a divider factor of more than 1 related to the cluster clock CLS[i]\_CLK.

**Note:** To find the relevant places in the specification versions of TC2xx, search for "trigger chain" instead of "pipeline register" or "synchronization register".

### Scope

ATOM, TOM

### Effects

Missing edge on output signal (A)TOM\_OUT.

### Workaround 1

If available, use channels without a pipeline/synchronization register within the trigger chain between the triggering channel and the triggered channel.

### Workaround 2a

Applicable for the error case with (A)TOM[i]\_CH[x]\_CM1.CM1=1:

- Switch the order of the edges, so that (A)TOM[i]\_CH[x]\_CM0.CM0 defines the first edge and (A)TOM[i]\_CH[x]\_CM1.CM1 the second edge. Additionally invert the SL value to get the same waveform on the output signal (A)TOM\_OUT

**Note:** In this case, to generate 0% duty cycle, use (A)TOM[i]\_CH[x]\_CM1.CM1=0 and (A)TOM[i]\_CH[x]\_CM0.CM0>MAX.

### Workaround 2b

Applicable for the error case with (A)TOM[i]\_CH[x]\_CM1.CM1=0:

- Set (A)TOM[i]\_CH[x]\_CM0.CM0=MAX and (A)TOM[i]\_CH[x]\_SR0.SR0=MAX by writing them before the target period. Set (A)TOM[i]\_CH[x]\_CM1.CM1 to the original application value of (A)TOM[i]\_CH[x]\_CM0.CM0. Additionally, invert the SL value to get the same waveform on the output signal (A)TOM\_OUT

### Workaround 3

Use a clock resolution for the triggered channel with a divider value of 1 related to the cluster clock.

## 2 Functional deviations

### 2.77 [GTM\_AI.522] (A)TOM: Edge at output signal (A)TOM\_OUT does not occur

#### Description

If the channel is configured to be triggered by a preceding channel with (A)TOM[i]\_CH[x]\_CTRL.RST\_CCU0=1<sub>B</sub> (SOMP mode for ATOM) and the duty cycle synchronously switches from 100% duty cycle with (A)TOM[i]\_CH[x]\_CM0.CM0=0<sub>H</sub> and (A)TOM[i]\_CH[x]\_CM1.CM1>MAX to 0% duty cycle with (A)TOM[i]\_CH[x]\_CM0.CM0>MAX for 0% duty cycle, the expected edge on the output signal (A)TOM\_OUT to the inverted (A)TOM[i]\_CH[x]\_CTRL.SL value does not occur. If this switch is performed by setting the corresponding registers asynchronously, the edge on the output signal (A)TOM\_OUT to the inverted (A)TOM[i]\_CH[x]\_CTRL.SL value does occur as expected.

**Note:** *If the setting after synchronously switching to 0% duty cycle is not changed, the edge appears at the beginning of the next period.*

#### Scope

TOM, ATOM

#### Effects

Output signal (A)TOM\_OUT remains at (A)TOM[i]\_CH[x]\_CTRL.SL value.

#### Workaround

In the period before the synchronous change to 0% duty cycle, set the value of (A)TOM[i]\_CH[x]\_CM1.CM1 to MAX instead of greater than MAX. This can be done asynchronously by writing the bit-field (A)TOM[i]\_CH[x]\_CM1.CM1 within the period.

Alternatively, it can be done via the synchronous update mechanism by writing the bit-field (A)TOM[i]\_CH[x]\_SR1.SR1 two periods before switching to 0% duty cycle.

### 2.78 [GTM\_AI.527] GTM-ARCH: CPU bus access is not acknowledged

#### Description

If a cluster is switched off by writing GTM\_CLS\_CLK\_CFG.CLS[j]\_CLK\_DIV = 00<sub>B</sub> while the MCS inside this cluster is executing a bus access, further CPU bus accesses to the switched off cluster are not acknowledged with an AEI\_READY signal and the corresponding AEI\_STATUS = 10<sub>B</sub>.

#### Scope

GTM-ARCH

#### Effects

CPU bus accesses to switched off clusters are not terminated.

#### Workaround

Disable the MCS bus access by writing MCS\_AEM\_DIS.DIS\_CLS[j] = 10<sub>B</sub> before switching off the cluster with GTM\_CLS\_CLK\_CFG.CLS[j]\_CLK\_DIV = 00<sub>B</sub>. Enable the MCS bus access by writing MCS\_AEM\_DIS.DIS\_CLS[j] = 01<sub>B</sub> after switching on the cluster again.

## 2 Functional deviations

### 2.79 [GTM\_AI.538] RTL-TIM: Wrong functionality of the overflow bit TIM[i]\_CH[x]\_IRQ\_NOTIFY.CNTOFL

#### Description

The specification states that if the register TIM[i]\_CH[x]\_CNT produces an overflow (wrap around 0xFFFFF to 0x0 due to increment) during the measurement, the bit TIM[i]\_CH[x]\_IRQ\_NOTIFY.CNTOFL is to be set.

The bit TIM[i]\_CH[x]\_IRQ\_NOTIFY.CNTOFL is set erroneously one (cluster) clock cycle after the register TIM[i]\_CH[x]\_CNT reaches the value 0xFFFFF. It cannot be cleared as long as the counter stays at 0xFFFFF.

#### Scope

TIM

#### Effects

The interrupt TIM\_CNTOFL\_IRQ[x:x] is raised too early and cannot be cleared until the register TIM[i]\_CH[x]\_CNT reaches the value 0x0.

The TIM IRQ output behaves the same (constant level output) in every interrupt mode except the single-pulse mode, where still only one pulse will be generated.

#### Workaround

For applications where the counter counts unconditionally:

- Use no clock divider in the CMU so that the register TIM[i]\_CH[x]\_CNT continuously increments with each (cluster) clock cycle

### 2.80 [GTM\_AI.539] RTL-TIM: Unexpected overflow bit TIM[i]\_CH[x]\_IRQ\_NOTIFY.CNTOFL

#### Description

The specification states that in the TIM modes TIPM, TGPS and TSSM, the register TIM[i]\_CH[x]\_CNT never produces an overflow and the bit TIM[i]\_CH[x]\_IRQ\_NOTIFY.CNTOFL is never set.

This mode dependency is not implemented and the overflow bit TIM[i]\_CH[x]\_IRQ\_NOTIFY.CNTOFL may be activated in all TIM modes.

#### Scope

TIM

#### Effects

The overflow bit TIM[i]\_CH[x]\_IRQ\_NOTIFY.CNTOFL can be set in the TIPM, TGPS and TSSM mode, resulting in the false raising of the interrupt TIM\_CNTOFL\_IRQ[x:x] if the corresponding interrupt enable condition is met.

#### Workaround

Disable the interrupt by setting TIM[i]\_CH[x]\_IRQ\_EN.CNTOFL\_IRQ\_EN and TIM[i]\_CH[x]\_EIRQ\_EN.CNTOFL\_EIRQ\_EN to the value of 0.

## 2 Functional deviations

### 2.81 [GTM\_AI.541] GTM-ARCH: Bit-field GTM\_RST.BRIDGE\_MODE\_WRDIS not writable

#### Description

The bit-field GTM\_RST.BRIDGE\_MODE\_WRDIS cannot be reliably written through the configuration interface if the clock divider for cluster 0 is configured to 1 (GTM\_CLS\_CLK\_CFG.CLS0\_CLK\_DIV = 01<sub>B</sub>) or if cluster 0 clock is switched off (GTM\_CLS\_CLK\_CFG.CLS0\_CLK\_DIV = 00<sub>B</sub>). The reason is that the register GTM\_RST runs always with a 2:1 clock and hence the write will be missed with a 50% probability. No error will be reported.

**Note:** MCS bus write accesses to GTM\_RST.BRIDGE\_MODE\_WRDIS are also affected if a clock divider for cluster 0 is configured to 1 (GTM\_CLS\_CLK\_CFG.CLS0\_CLK\_DIV = 01<sub>B</sub>).

#### Scope

ARCH

#### Effects

A write access to GTM\_RST.BRIDGE\_MODE\_WRDIS will not update the bit-field value and no error will be reported.

#### Workaround

Configure the clock divider for cluster 0 to a 2:1 clock ratio (GTM\_CLS\_CLK\_CFG.CLS0\_CLK\_DIV = 10<sub>B</sub>) before writing to GTM\_RST.BRIDGE\_MODE\_WRDIS.

### 2.82 [GTM\_TC.020] Debug/Normal read access control via bit-field ODA.DRAC

#### Description

A few GTM registers have a different read behavior when accessing them with debug read accesses (see section “GTM Software Debugger Support” in the GTM chapter of the User’s Manual for further details).

Depending on the reading master and the configuration of bit-field DRAC in register GTM\_ODA (OCDS Debug Access Register), the read can be performed in a specific way for debug related read operation.

According to the User’s Manual the read is performed as a debug read operation

- for all masters when ODA.DRAC = 10<sub>B</sub> or 11<sub>B</sub>
- for the Cerberus (OCDS) FPI master when ODA.DRAC = 00<sub>B</sub>

#### Problem Description

In the current implementation the read is performed as debug read operation

- for all masters when ODA.DRAC = 10 or 11<sub>B</sub>
- for the CPU2 FPI master when ODA.DRAC = 00<sub>B</sub>

#### Workaround

The problem described above has 2 aspects:

##### 1. For CPU2 Access to GTM

When the CPU2 FPI master is used to perform a normal read of the GTM registers mentioned above, setting ODA.DRAC = 01<sub>B</sub> is required to avoid an unintended debug read access that would be caused by this issue.

**2 Functional deviations**

**2. For Cerberus (OCDS) Access to GTM**

When ODA.DRAC = 00<sub>B</sub>, due to this problem any read access of the Cerberus (OCDS) FPI master to the registers that by default have a different behavior between normal and debug read will cause the normal read behavior. To get the intended debug read behavior, ODA.DRAC needs to be set to 10<sub>B</sub> or 11<sub>B</sub> before each access of the Cerberus and set back to 00<sub>B</sub> afterwards to not affect the access of other FPI masters on the registers mentioned above.

**2.83 [GTM\_TC.026] Table “GTM IP Application Constraints” #1 (DPLL) - Documentation correction**

**Description**

In table “GTM IP Application Constraints” in the GTM chapter of the AURIX™ TC3xx User’s Manual, in the first row, the unit of the required time for item #1 (Increment duration (time between two valid inputs of the DPLL: TRIGGER/STATE)) is erroneously listed as > 23.4 ms instead of > 23.4 **µs**.

**Documentation correction**

The unit of the required time for item #1 (Increment duration (time between two valid inputs of the DPLL: TRIGGER/STATE)) in table “GTM IP Application Constraints” shall be corrected to > 23.4 **µs**.

**2.84 [GTM\_TC.031]Connections of ADC\_TRIG4 signals - Correction in TC3xx appendix**

**Description**

In the table "Connections of ADC\_TRIGx Signals to ADC/SENT Modules" in the GTM chapter of the product specific appendix to the TC3xx user manual, the following EVADC connections in section "ADC\_TRIG4" are incorrect:

- G\*REQGTL, G\*REQTRL (should be G\*REQGTG, G\*REQTRG)

**Documentation correction**

The EVADC connections in the section "ADC\_TRIG4" in the table "Connections of ADC\_TRIGx Signals to ADC/SENT Modules" in the GTM chapter of the corresponding appendix shall be replaced by the following corrected section, depending on the particular product.

**Note:** *This documentation issue is related to the appendix of the TC3xx user manual for TC3Ex, TC38x, TC37x, TC37xEXT, TC36x and TC33x/TC32x. For TC39x, see erratum GTM\_TC.029.*

**Table 6 Connections of ADC\_TRIGx Signals to ADC/SENT Modules - Correction for ADC\_TRIG4 in TC3Ex and TC38x appendix**

<b>GTM Trigger Signal</b>	<b>EVADC</b>	<b>EDSADC</b>	<b>SENT</b>
<b>ADC_TRIG4</b>			
ADC_TRIG4_[7:0]	-	G[7:0]REQGTG	G[7:0]REQTRG
ADC_TRIG4_[9:8]	FC[1:0]REQTRM	G[9:8]REQGTG	G[9:8]REQTRG
ADC_TRIG4_[11:10]	-	G[11:10]REQGTG	G[11:10]REQTRG

**2 Functional deviations**

**Table 7**                    **Connections of ADC\_TRIGx Signals to ADC/SENT Modules - Correction for ADC\_TRIG4 in TC37x and TC37xEXT appendix**

<b>GTM Trigger Signal</b>	<b>EVADC</b>	<b>EDSADC</b>	<b>SENT</b>
<b>ADC_TRIG4</b>			
ADC_TRIG4_[3:0]	-	G[3:0]REQGTG	G[3:0]REQTRG
ADC_TRIG4_[7:4]	-	-	-
ADC_TRIG4_[9:8]	FC[1:0]REQTRM	G[9:8]REQGTG	G[9:8]REQTRG
ADC_TRIG4_[11:10]	-	G[11:10]REQGTG	G[11:10]REQTRG

**Table 8**                    **Connections of ADC\_TRIGx Signals to ADC/SENT Modules - Correction for ADC\_TRIG4 in TC36x appendix**

<b>GTM Trigger Signal</b>	<b>EVADC</b>	<b>EDSADC</b>	<b>SENT</b>
<b>ADC_TRIG4</b>			
ADC_TRIG4_[3:0]	-	G[3:0]REQGTG	G[3:0]REQTRG
ADC_TRIG4_[7:4]	-	-	-
ADC_TRIG4_[9:8]	FC[1:0]REQTRM	G[9:8]REQGTG	G[9:8]REQTRG
ADC_TRIG4_[11:10]	-	-	-

**Table 9**                    **Connections of ADC\_TRIGx Signals to ADC/SENT Modules - Correction for ADC\_TRIG4 in TC33x/TC32x appendix**

<b>GTM Trigger Signal</b>	<b>EVADC</b>	<b>SENT</b>
<b>ADC_TRIG4</b>		
ADC_TRIG4_[1:0]	G[1:0]REQGTG	G[1:0]REQTRG
ADC_TRIG4_[3:2]	-	-
ADC_TRIG4_[7:4]	-	-
ADC_TRIG4_[9:8]	G[9:8]REQGTG	G[9:8]REQTRG
ADC_TRIG4_[11:10]	-	-

**2.85                    [GTM\_TC.033] Confirmation about 3rd party IPs RWH register type**

**Description**

Access rights are wrongly documented for hardware modifiable bits.

The following bit-fields inside the GTM3 specification are not marked as volatile bits, even though they can be changed by hardware during run-time. In addition, all bit-fields which are marked as write only (w), can be read, but this is not a defined usage of this bit-field. Therefore, the read return value is either specified or undefined.

The definition rw1ch is not used in TC3xx documentation. This is only available inside the text description. To see directly the proper definition, refer to the access right description within this errata.

### 2 Functional deviations

**Note:** *If the Bitfield name is “all”, it excludes the reserved bit fields. If the bit position of a reserved bit is not listed in the table below, the access rights of a reserved bit will remain the same.*

Please be aware that this errata states all available resources in TC39x. As all other devices of the TC3xx family are having less features, the change of the access right is only relevant, if the register exists in the device.

Register name	Bitfield name	Proper access right
GTM_CTRL	AEIM_CLUSTER	rh
CLC	DISS	rh
MON_ACTIVITY_x	all	rw1ch
MON_ACTIVITY_MCSx	all	rw1ch
MCSi_CHx_PC	PC	rwh
MCSi_CHx_ACB	all	rh
MCSi_CHx_IRQ_NOTIFY	all	rw1ch
MCSi_CTRL_STAT	RAM_RST	rwh
MCSi_CTRL_STAT	ERR_SRC_ID	rh
MCSi_CTRG	all	rw1ch
MCSi_STRG	all	rwh
MCSi_CAT	all	rwh
MCSi_CWT	all	rwh
MCSi_ERR	all	rw1ch
MCSi_REG_PROT	all	rwh
CCMi_AEIM_STA	all	rwh
CCMi_ATOM_OUT	all	rh
CCMi_CFG	all	rwh
CCMi_TOM_OUT	all	rh
AFDi_CHx_BUF_ACC	all	rwh
ARU_DATA_H	all	rwh
ARU_DATA_L	all	rwh
ARU_DBG_DATA0_H	all	rh
ARU_DBG_DATA0_L	all	rh
ARU_DBG_DATA1_H	all	rh
ARU_DBG_DATA1_L	all	rh
ATOMi_CHx_CTRL	WR_REQ	rwh
ATOMi_CHx_CTRL	CLK_SRC_SR	rwh
ATOMi_CHx_CTRL	SL	rwh
ATOMi_CHx_STAT	ACBI	rh
ATOMi_CHx_STAT	ACBO	rh
ATOMi_CHx_STAT	DR	rh
ATOMi_CHx_STAT	DV	rh
ATOMi_CHx_STAT	OL	rh

**2 Functional deviations**

<b>Register name</b>	<b>Bitfield name</b>	<b>Proper access right</b>
ATOMi_CHx_STAT	WRF	rw1ch
ATOMi_CHx_SOMP	SL	rwh
ATOMi_CHx_SOMP	CLK_SRC_SR	rwh
ATOMi_CHx_SR0	all	rwh
ATOMi_CHx_SR1	all	rwh
ATOMi_CHx_CM0	all	rwh
ATOMi_CHx_CM1	all	rwh
ATOMi_CHx_CN0	all	rwh
BRIDGE_PTR1	all	rh
BRIDGE_PTR2	all	rh
BRIDGE_MODE	MODE_UP_PGR	rh
BRIDGE_MODE	BUFF_OVL	rw1ch
MCS_AEM_DIS	all	rwh
DATAIn	all	rwh
GTM_IRQ_NOTIFY	AEI_TO_XPT	rw1ch
GTM_IRQ_NOTIFY	AEI_USP_ADDR	rw1ch
GTM_IRQ_NOTIFY	AEI_IM_ADDR	rw1ch
GTM_IRQ_NOTIFY	AEI_USP_BE	rw1ch
GTM_IRQ_NOTIFY	AEIM_USP_ADDR	rw1ch
GTM_IRQ_NOTIFY	AEIM_IM_ADDR	rw1ch
GTM_IRQ_NOTIFY	AEIM_USP_BE	rw1ch
GTM_IRQ_NOTIFY	CLK_EN_ERR	rw1ch
GTM_IRQ_NOTIFY	CLK_PER_ERR	rw1ch
GTM_IRQ_NOTIFY	CLK_EN_ERR_STATE0	rh
GTM_IRQ_NOTIFY	CLK_EN_ERR_STATE1	rh
GTM_IRQ_NOTIFY	CLK_EN_EXP_STATE0	rh
GTM_IRQ_NOTIFY	CLK_EN_EXP_STATE1	rh
TIMi_CHx_IRQ_NOTIFY	all	rw1ch
TOMi_CHx_IRQ_NOTIFY	all	rw1ch
ARU_IRQ_NOTIFY	all	rw1ch
FIFOi_CHz_IRQ_NOTIFY	all	rw1ch
BRC_IRQ_NOTIFY	all	rw1ch
ATOMi_CHx_IRQ_NOTIFY (	all	rw1ch
DPLL_IRQ_NOTIFY	all	rw1ch
SPEi_IRQ_NOTIFY	all	rw1ch
CMP_IRQ_NOTIFY	all	rw1ch
MON_STATUS	ACT_CMUx (x=0-7)	rw1ch
MON_STATUS	ACT_CMUFx(x=0-4)	rw1ch

**2 Functional deviations**

<b>Register name</b>	<b>Bitfield name</b>	<b>Proper access right</b>
MON_STATUS	ACT_CMU8	rw1ch
MON_STATUS	CMP_ERR	rh
MON_STATUS	MCSx_ERR(x=0-9)	rh
TOMi_TGC0_ACT_TB	TB_TRIG	rwh
TOMi_CHx_CTRL	CLK_SRC_SR	rwh
TOMi_CHx_CTRL	SL	rwh
TOMi_CHx_CM0	all	rwh
TOMi_CHx_CM1	all	rwh
TOMi_CHx_CN0	all	rwh
TOMi_CHx_STAT	all	rh
TOMi_TGCx_ENDIS_STAT	all	rwh
TOMi_TGCx_GLB_CTRL	HOST_TRIG	w1c
TOMi_TGCx_GLB_CTRL	RST_CHx (x=0-7)	rw
TOMi_TGCx_OUTEN_STAT	all	rwh
FIFOi_CHx_WR_PTR	all	rh
FIFOi_CHz_RD_PTR	all	rh
FIFOi_CHz_CTRL	FLUSH	rwh
FIFOi_CHz_STATUS	all	rh
FIFOi_CHz_FILL_LEVEL	all	rh
FIFOi_CHz_WR_PTR	all	rh
FIFOi_CHz_RD_PTR	all	rh
AEI_STA_XPT	all	rh
AEI_ADDR_XPT	all	rh
ARU_ACCESS	WREQ	rwh
ARU_ACCESS	RREQ	rwh
ARU_CADDR	all	rh
ARU_x_DYN_ROUTE_LOW	all	rwh
ARU_x_DYN_ROUTE_HIGH	all	rwh
ARU_DBG_DATA0_H	all	rh
ARU_DBG_DATA0_L	all	rh
ARU_DBG_DATA1_H	all	rh
ARU_DBG_DATA1_L	all	rh
F2Ai_CHz_ARU_RD_FIFO	all	rwh
F2Ai_CHz_STR_CFG	all	rwh
F2Ai_ENABLE	all	rwh
F2Ai_CTRL	all	rwh
BRC_SRC_x_ADDR	all	rwh
SPEi_CTRL_STAT	NIP	rh

**2 Functional deviations**

<b>Register name</b>	<b>Bitfield name</b>	<b>Proper access right</b>
SPEi_CMD	SPE_UPD_TRIG	rwh
ICM_IRQG_n (n=0-11)	all	rh
ICM_IRQG_MEI	all	rh
ICM_IRQG_CEl <sub>n</sub> (n=0-4)	all	rh
ICM_IRQG_MCSi_CEl (i=0-9)	all	rh
ICM_IRQG_MCSi_CI (i=0-9)	all	rh
ICM_IRQG_SPE_CEl	all	rh
ICM_IRQG_SPE_CI	all	rh
ICM_IRQG_PSM_0_CEl	all	rh
ICM_IRQG_PSM_0_CI	all	rh
ICM_IRQG_TOM_k_CI (k=0-2)	all	rh
ICM_IRQG_ATOM_k_CI (k=0-2)	all	rh
ICM_IRQG_CLS_k_MEI (k=0-2)	all	rh
CMU_CLK_EN	all	rwh
CMU_GCLK_NUM	all	rwh
CMU_GCLK_DEN	all	rwh
CMU_CLK_[z]_CTRL	all	rwh
CMU_ECLK_[z]_NUM	all	rwh
CMU_ECLK_[z]_DEN	all	rwh
CMU_FXCLK_CTRL	all	rwh
CMU_GLB_CTRL	all	rwh
CMU_CLK_CTRL	all	rwh
TIMi_CHx_CTRL (i=0-7;x=0-7)	TIM_EN	rwh
TIMi_CHx_GPR0 (i=0-7;x=0-7)	GPR0	rwh
TIMi_CHx_GPR1 (i=0-7;x=0-7)	GPR1	rwh
TIMi_CHx_CNT (i=0-7;x=0-7)	all	rh
TIMi_CHx_ECNT (i=0-7;x=0-7)	all	rh
TIMi_CHx_CNTS (i=0-7;x=0-7)	all	rwh
TIMi_CHx_TDUC (i=0-7;x=0-7)	all	rwh
TIMi_INP_VAL (i=0-7)	all	rh
TIMi_IN_SRC (i=0-7)	all	rwh
TBU_CHEN	all	rwh
TBU_CH <sub>n</sub> _CTRL (n=0-2)	all	rwh
TBU_CH3_CTRL	CH_MODE	rh
TBU_CH3_CTRL	USE_CH2	rwh
TBU_CH <sub>n</sub> _BASE	all	rwh
TBU_CH3_BASE_MARK	all	rwh
TBU_CH3_BASE_CAPTURE	all	rh

**2 Functional deviations**

<b>Register name</b>	<b>Bitfield name</b>	<b>Proper access right</b>
CLSi_CDTM_DTMD_CH_CTRL2	all	rwh
CLSi_CDTM_DTMD_CH_CTRL2_SR	SLy_z_SR	rwh
CDTMi_DTMj_CHz_DTV	all	rwh
DPLL_ACB_z	all	rwh
DPLL_ACT_STA	all	rwh
DPLL_ADD_IN_CALn	all	rwh
DPLL_AOSV_2	all	rh
DPLL_CTRL_11	all	rwh
DPLL_CTRL_EXT	all	rwh
DPLL_APT	APT	rwh
DPLL_APT	APT_2B	rwh
DPLL_APT_2C	APT_2C	rwh
DPLL_APT_SYNC	APT_2B_STATUS	rwh
DPLL_APT_SYNC	APT_2B_OLD	rwh
DPLL_APS	APS	rwh
DPLL_APS_EXT	APS	rwh
DPLL_APS_EXT	APS_1C2	rwh
DPLL_APS_1C2	APS_1C2	rwh
DPLL_APS_1C3	APS_1C3	rwh
DPLL_APS_1C3_EXT	APS_1C3	rwh
DPLL_APS_SYNC	APS_1C2_STATUS	rwh
DPLL_APS_SYNC	APT_1C2_OLD	rwh
DPLL_APS_SYNC_EXT	all	rwh
DPLL_CTRL_i_SHADOW_TRIGGER	all	rh
DPLL_CTRL_i_SHADOW_STATE	all	rh
DPLL_AOSV_2	all	rh
DPLL_CDT_SX	CDT_SX	rwh
DPLL_CDT_SX_NOM	CDT_SX_NOM	rwh
DPLL_CDT_TX	CDT_TX	rwh
DPLL_CDT_TX_NOM	CDT_TX_NOM	rwh
DPLL_CTRL_EXT	all	rwh
DPLL_DLAi	DLA	rwh
DPLL_DTAi	DTA	rwh
DPLL_DT_Si	DT_S	rwh
DPLL_DT_S_ACT	DT_S_ACT	rwh
DPLL_DT_T_ACT	DT_T_ACT	rwh
DPLL_EDT_S	EDT_S	rwh
DPLL_EDT_T	EDT_T	rwh

**2 Functional deviations**

<b>Register name</b>	<b>Bitfield name</b>	<b>Proper access right</b>
DPLL_FTV_S	STATE_FT	rwh
DPLL_FTV_T	TRIGGER_FT	rwh
DPLL_ID_PMTR_i	ID_PMTR_X	rwh
DPLL_INC_CNT1	INC_CNT1	rwh
DPLL_INC_CNT2	INC_CNT2	rwh
DPLL_IRQ_MODE	IRQ_MODE	rwh - According to Bosch specification
DPLL_MEDT_S	MEDT_S	rwh
DPLL_MEDT_T	MEDT_T	rwh
DPLL_NAi	all	rwh
DPLL_NMB_S	NMB_S	rwh
DPLL_NMB_S_TAR	NMB_S_TAR	rwh
DPLL_NMB_S_TAR_OLD	NMB_S_TAR_OLD	rwh
DPLL_NMB_T	NMB_T	rwh
DPLL_NMB_T_TAR	NMB_T_TAR	rwh
DPLL_NMB_T_TAR_OLD	NMB_T_TAR_OLD	rwh
DPLL_NTI_CNT	NTI_CNT	rwh
DPLL_NUSC	VSN	rwh
DPLL_NUSC	SYN_S_OLD	rwh
DPLL_NUSC	SYN_S	rwh
DPLL_NUSC	FSS	rwh
DPLL_NUSC	NUSE	rwh
DPLL_NUSC_EXT1	all	rwh
DPLL_NUSC_EXT2	VSN	rwh
DPLL_NUSC_EXT2	FSS	rwh
DPLL_NUSC_EXT2	NUSE	rwh
DPLL_NUTC	VTN	rwh
DPLL_NUTC	SYN_T_OLD	rwh
DPLL_NUTC	SYN_T	rwh
DPLL_NUTC	FST	rwh
DPLL_NUTC	NUTE	rwh
DPLL_OSW	SWON_S	rh
DPLL_OSW	SWON_T	rh
DPLL_OSW	OSS	rwh
DPLL_PDT_i	all	rwh
DPLL_PSAi	PSA	rwh
DPLL_PSACi	PSAC	rwh
DPLL_PSSC	PSSC	rwh

**2 Functional deviations**

<b>Register name</b>	<b>Bitfield name</b>	<b>Proper access right</b>
DPLL_PSSM	PSSM	rwh
DPLL_PSSM_OLD	PSSM_OLD	rwh
DPLL_PSTM	PSTM	rwh
DPLL_PSTM_OLD	PSTM_OLD	rwh
DPLL_PVT	PVT	rwh
DPLL_RAM_INI	INIT_RAM	rwh
DPLL_RAM_INI	INIT_2	rh
DPLL_RAM_INI	INIT_1BC	rh
DPLL_RAM_INI	INIT_1A	rh
DPLL_RCDT_SX	RCDT_SX	rwh
DPLL_RCDT_SX_NOM	RCDT_SX_NOM	rwh
DPLL_RCDT_TX	RCDT_TX	rwh
DPLL_RCDT_TX_NOM	RCDT_TX_NOM	rwh
DPLL_RDT_Si	RDT_S	rwh
DPLL_RDT_S_ACT	S_ACT	rwh
DPLL_RDT_T_ACT	T_ACT	rwh
DPLL_STA	all	rh
DPLL_STATUS	ERR	rh
DPLL_STATUS	LOCK1	rh
DPLL_STATUS	FTD	rh
DPLL_STATUS	FSD	rh
DPLL_STATUS	SYT	rh
DPLL_STATUS	SYS	rh
DPLL_STATUS	LOCK2	rh
DPLL_STATUS	BWD1	rh
DPLL_STATUS	BWD2	rh
DPLL_STATUS	ITN	rh
DPLL_STATUS	ISN	rh
DPLL_STATUS	CAIP1	rh
DPLL_STATUS	CAIP2	rh
DPLL_STATUS	CSVT	rh
DPLL_STATUS	CSVS	rh
DPLL_STATUS	LOW_RES	rh
DPLL_STATUS	RAM_2ERR	rw1ch
DPLL_STATUS	MT	rw1ch
DPLL_STATUS	TOR	rw1ch
DPLL_STATUS	MS	rw1ch
DPLL_STATUS	SOR	rw1ch

## 2 Functional deviations

Register name	Bitfield name	Proper access right
DPLL_STATUS	PSE	rh
DPLL_STATUS	RCT	rh
DPLL_STATUS	RCS	rh
DPLL_STATUS	CRO	rw1ch
DPLL_STATUS	CTO	rw1ch
DPLL_STATUS	CSO	rw1ch
DPLL_STATUS	FPCE	rw1ch
DPLL_STA_FLAG	all	rw1ch
DPLL_TBU_TS0_S	TBU_TS0_S	rwh
DPLL_TBU_TS0_T	TBU_TS0_T	rwh
DPLL_THVAL	THVAL	rwh
DPLL_THVAL2	THVAL	rh
DPLL_TSACi	TSAC	rwh
DPLL_TSF_Si	TSF_S	rwh
DPLL_TS_S	STATE_TS	rwh
DPLL_TS_S_OLD	STATE_TS_OLD	rwh
DPLL_TS_T	TRIGGER_TS	rwh
DPLL_TS_T_OLD	TRIGGER_TS_OLD	rwh

- Note:**
- *r*: read
  - *w*: software write
  - *w1c*: software write 1 clear
  - *h*: hardware write

### Scope

Register access rights.

### Effects

Optimization via compiler might be wrong.

### Workaround

Use the new access rights.

## 2.86 [INT\_TC.003] Wrong number of SRB groups specified

### Description

The following information in the TC33x/TC32x appendix document V2.0, chapter IR are incorrect and must be corrected:

Sub-chapter 'TC33x/TC32x Specific Interrupt Router Configuration':

- Wrong information: Number of SRB Groups is 2
- Correct information: Number of SRB Groups is 1

---

**2 Functional deviations**

Sub-chapter 'TC33x/TC32x Specific Control Registers', Table 'Register Address Space - INT':

- Wrong information 'INT\_SRBx (x=0-1)'
- Correct information 'INT\_SRBx (x=0)'
- Wrong information 'INT\_ACCEN\_SRBx1 (x=0-1)'
- Correct information 'INT\_ACCEN\_SRBx1 (x=0)'

**Scope**

The TC33x/TC32x appendix document V2.0, chapter IR is affected. The topics affected are the overview description and the IR register description.

**Effects**

See the description.

**Workaround**

Not applicable.

**2.87 [MCMCAN\_AI.015] Edge filtering causes mis-synchronization when falling edge at Rx input pin coincides with end of integration phase****Description**

When edge filtering is enabled (CCCRi.EFBI = '1') and when the end of the integration phase coincides with a falling edge at the Rx input pin it may happen, that the MCMCAN synchronizes itself wrongly and does not correctly receive the first bit of the frame. In this case the CRC will detect that the first bit was received incorrectly, it will rate the received FD frame as faulty and an error frame will be send.

The issue only occurs, when there is a falling edge at the Rx input pin within the last time quantum (tq) before the end of the integration phase. The last time quantum of the integration phase is at the sample point of the 11th recessive bit of the integration phase. When the edge filtering is enabled, the bit timing logic of the MCMCAN sees the Rx input signal delayed by the edge filtering. When the integration phase ends, the edge filtering is automatically disabled. This affects the reset of the FD CRC control unit at the beginning of the frame. The Classical CRC control unit is not affected, so this issue does not affect the reception of Classical frames.

In CAN communication, the MCMCAN may enter integrating state (either by resetting CCCRi.INIT or by protocol exception event) while a frame is active on the bus. In this case the 11 recessive bits are counted between the acknowledge bit and the following start of frame. All nodes have synchronized at the beginning of the dominant acknowledge bit. This means that the edge of the following start of frame bit cannot fall on the sample point, so the issue does not occur. The issue occurs only when the MCMCAN is, by local errors, mis-synchronized with regard to the other nodes, or not synchronized at all.

Glitch filtering as specified in ISO 11898-1:2015 is fully functional.

Edge filtering was introduced for applications where the data bit time is at least two tq (of the nominal bit time) long. In that case, edge filtering requires at least two consecutive dominant time quanta before the counter counting the 11 recessive bits for idle detection is restarted. This means edge filtering covers the theoretical case of occasional 1-tq-long dominant spikes on the CAN bus that would delay idle detection. Repeated dominant spikes on the CAN bus would disturb all CAN communication, so the filtering to speed up idle detection would not help network performance.

When this rare event occurs, the MCMCAN sends an error frame and the sender of the affected frame retransmits the frame. When the retransmitted frame is received, the MCMCAN has left integration phase and the frame will be received correctly. Edge filtering is only applied during integration phase, it is never used during normal operation. As integration phase is very short with respect to “active communication time”, the impact on total error frame rate is negligible. The issue has no impact on data integrity.

## 2 Functional deviations

The MCMCAN enters integration phase under the following conditions:

- when CCCRI.INIT is set to '0' after start-up
- after a protocol exception event (only when CCCRI.PXHD = '0')

### Scope

The erratum is limited to FD frame reception when edge filtering is active (CCCRI.EFBI = '1') and when the end of the integration phase coincides with a falling edge at the Rx input pin.

### Effects

The calculated CRC value does not match the CRC value of the received FD frame and the MCMCAN sends an error frame. After retransmission the frame is received correctly.

### Workaround

Disable edge filtering or wait on retransmission in case this rare event happens.

## 2.88 [MCMCAN\_AI.017] Retransmission in DAR mode due to lost arbitration at the first two identifier bits

### Description

When the MCMCAN CAN Node is configured in DAR mode (CANx.CCCRI.DAR = '1') the Automatic Retransmission for transmitted messages that have been disturbed by an error or have lost arbitration is disabled. When the transmission attempt is not successful, the Tx Buffer's transmission request bit (CANx.TXBRPi.TRPz) shall be cleared and its cancellation finished bit (CANx.TXBCFi.CFz) shall be set.

When the transmitted message loses arbitration at one of the first two identifier bits, it may happen, that instead of the bits of the actually transmitted Tx Buffer, the CANx.TXBRPi.TRPz and CANx.TXBCFi.CFz bits of the previously started Tx Buffer (or Tx Buffer 0 if there is no previous transmission attempt) are written (CANx.TXBRPi.TRPz = '0', CANx.TXBCFi.CFz = '1').

If in this case the CANx.TXBRPi.TRPz bit of the Tx Buffer that lost arbitration at the first two identifier bits has not been cleared, retransmission is attempted.

When the CAN Node loses arbitration again at the immediately following retransmission, then actually and previously transmitted Tx Buffer are the same and this Tx Buffer's CANx.TXBRPi.TRPz bit is cleared and its CANx.TXBCFi.CFz bit is set.

### Scope

The erratum is limited to the case when the MCMCAN CAN Node loses arbitration at one of the first two transmitted identifier bits while in DAR mode.

The problem does not occur when the transmitted message has been disturbed by an error.

### Effects

In this case it may happen, that the CANx.TXBRPi.TRPz bit is cleared after the second transmission attempt instead of the first.

Additionally it may happen that the CANx.TXBRPi.TRPz bit of the previously started Tx Buffer is cleared, if it has been set again. As in this case the previously started Tx Buffer has lost MCMCAN internal arbitration against the active Tx Buffer, its message has a lower identifier priority. It would also have lost arbitration on the CAN bus at the same position.

### Workaround

None.

## 2 Functional deviations

### 2.89 [MCMCAN\_AI.018] Tx FIFO message sequence inversion

#### Description

Assume the case that there are two Tx FIFO messages in the output pipeline of the Tx Message Handler (TxMH) and transmission of Tx FIFO message 1 is started:

- Position 1: Tx FIFO message 1 (transmission ongoing)
- Position 2: Tx FIFO message 2
- Position 3: --

Now a non-Tx FIFO message with a higher CAN priority is requested. Due to its priority it will be inserted into the output pipeline. The TxMH performs so called “message-scans” to keep the output pipeline up to date with the highest priority messages from the Message RAM. After the following two message-scans the output pipeline has the following content:

- Position 1: Tx FIFO message 1 (transmission ongoing)
- Position 2: non Tx FIFO message with higher CAN priority
- Position 3: Tx FIFO message 2

If the transmission of Tx FIFO message 1 is not successful (lost arbitration or CAN bus error) it is pushed from the output pipeline by the non-Tx FIFO message with higher CAN priority. The following scan re-inserts Tx FIFO message 1 into the output pipeline at position 3:

- Position 1: non Tx FIFO message with higher CAN priority (transmission ongoing)
- Position 2: Tx FIFO message 2
- Position 3: Tx FIFO message 1

Now Tx FIFO message 2 is in the output pipeline in front of Tx FIFO message 1 and they are transmitted in that order, resulting in a message sequence inversion.

**Note:** *Within the scope of the scenario described above, in case of more than two Tx FIFO messages, the Tx FIFO message that has lost arbitration will be inserted after the next pending Tx FIFO message.*

#### Scope

The erratum describes the case when the MCMCAN uses both, dedicated Tx Buffers and a Tx FIFO (CAN\_TXBCi.TFQM = '0') and the messages in the Tx FIFO do not have the highest internal CAN priority. The described sequence inversion may also happen between two non-Tx FIFO messages (Tx Queue or dedicated Tx Buffers) that have the same CAN identifier and that should be transmitted in the order of their buffer numbers (not the intended use).

#### Effects

In the described case it may happen that two consecutive messages from the Tx FIFO exchange their positions in the transmit sequence.

#### Workaround

When transmitting messages from a dedicated Tx Buffer with higher priority than the messages in the Tx FIFO, choose one of the following workarounds:

##### Workaround 1

Use two dedicated Tx Buffers, for example use Tx Buffers 4 and 5 instead of the Tx FIFO.

The pseudo-code below replaces the function that fills the Tx FIFO.

- Write message to Tx Buffer 4
- Transmit Loop:
  - Request Tx Buffer 4 - write TXBAR.A4
  - Write message to Tx Buffer 5

---

**2 Functional deviations**

- Wait until transmission of Tx Buffer 4 completed – CAN\_IRi.TC, read CAN\_TXBTOi.TO4
- Request Tx Buffer 5 - write CAN\_TXBARI.AR5
- Write message to Tx Buffer 4
- Wait until transmission of Tx Buffer 5 completed – CAN\_IRi.TC, read CAN\_TXBTOi.TO5

**Workaround 2**

Assure that only one Tx FIFO element is pending for transmission at any time. The Tx FIFO elements may be filled at any time with messages to be transmitted, but their transmission requests are handled separately. Each time a Tx FIFO transmission has completed and the Tx FIFO gets empty (CAN\_IRi.TFE = '1') the next Tx FIFO element is requested.

**Workaround 3**

Use only a Tx FIFO. Send the message with the higher priority also from Tx FIFO.

Drawback: The higher priority message has to wait until the preceding messages in the Tx FIFO have been sent.

**2.90 [MCMCAN\_AI.019] Unexpected High Priority Message (HPM) interrupt****Description**

There are two configurations where the issue occurs:

**Configuration A**

- At least one Standard Message ID Filter Element is configured with priority flag set (S0.SFEC = "100"/"101"/"110")
- No Extended Message ID Filter Element configured
- Non-matching extended frames are accepted (GFC.ANFE = "00"/"01")

The HPM interrupt flag IR.HPM is set erroneously on reception of a non-high-priority extended message under the following conditions:

1. A standard HPM frame is received, and accepted by a filter with priority flag set --> Interrupt flag IR.HPM is set as expected
2. Next an extended frame is received and accepted because of GFC.ANFE configuration --> Interrupt flag IR.HPM is set erroneously

**Configuration B**

- At least one Extended Message ID Filter Element is configured with priority flag set (F0.EFEC = "100"/"101"/"110")
- No Standard Message ID Filter Element configured
- Non-matching standard frames are accepted (GFC.ANFS = "00"/"01")

The HPM interrupt flag IR.HPM is set erroneously on reception of a non-high-priority standard message under the following conditions:

1. An extended HPM frame is received, and accepted by a filter with priority flag set --> Interrupt flag IR.HPM is set as expected
2. Next a standard frame is received and accepted because of GFC.ANFS configuration --> Interrupt flag IR.HPM is set erroneously

**Scope**

The erratum is limited to:

---

**2 Functional deviations**

- Configuration A:
  - No Extended Message ID Filter Element configured and non-matching extended frames are accepted due to Global Filter Configuration (GFC.ANFE = "00"/"01")
- Configuration B:
  - No Standard Message ID Filter Element configured and non-matching standard frames are accepted due to Global Filter Configuration (GFC.ANFS = "00"/"01")

**Effects**

Interrupt flag IR.HPM is set erroneously at the reception of a frame with:

- Configuration A: extended message ID
- Configuration B: standard message ID

**Workaround****Configuration A**

Setup an Extended Message ID Filter Element with the following configuration:

- F0.EFEC = "001"/"010" - select Rx FIFO for storage of extended frames
- F0.EFID1 = any value - value not relevant as all ID bits are masked out by F1.EFID2
- F1.EFT = "10" - classic filter, F0.EFID1 = filter, F1.EFID2 = mask
- F1.EFID2 = zero - all bits of the received extended ID are masked out

Now all extended frames are stored in Rx FIFO 0 respectively Rx FIFO 1 depending on the configuration of F0.EFEC.

**Configuration B**

Setup a Standard Message ID Filter Element with the following configuration:

- S0.SFEC = "001"/"010" - select Rx FIFO for storage of standard frames
- S0.SFID1 = any value - value not relevant as all ID bits are masked out by S0.SFID2
- S0.SFT = "10" - classic filter, S0.SFID1 = filter, S0.SFID2 = mask
- S0.SFID2 = zero - all bits of the received standard ID are masked out

Now all standard frames are stored in Rx FIFO 0 respectively Rx FIFO 1 depending on the configuration of S0.SFEC.

**2.91 [MCMCAN\_AI.022] Message order inversion when transmitting from dedicated Tx Buffers configured with same Message ID****Description**

Configuration:

Several Tx Buffers are configured with the same Message ID. Transmission of these Tx Buffers is requested sequentially with a delay between the individual Tx requests.

**Expected behavior**

When multiple Tx Buffers that are configured with the same Message ID have pending Tx requests, they shall be transmitted in ascending order of their Tx Buffer numbers. The Tx Buffer with lowest buffer number and pending Tx request is transmitted first.

---

**2 Functional deviations****Observed behavior**

It may happen, depending on the delay between the individual Tx requests, that in the case where multiple Tx Buffers are configured with the same Message ID the Tx Buffers are not transmitted in order of the Tx Buffer number (lowest number first).

**Scope**

The erratum is limited to the case when multiple Tx Buffers are configured with the same Message ID.

**Effects**

In the case described it may happen that Tx Buffers configured with the same Message ID and pending Tx request are not transmitted with lowest Tx Buffer number first (message order inversion).

**Workaround**

First write the group of Tx messages with same Message ID to the Message RAM and then afterwards request transmission of all these messages concurrently by a single write access to **TXBARI**. Before requesting a group of Tx messages with this Message ID ensure that no message with this Message ID has a pending Tx request.

**2.92 [MCMCAN\_AI.023] Incomplete description in section “Dedicated Tx Buffers” and “Tx Queue” of the M\_CAN documentation in the user manual related to transmission from multiple buffers configured with the same Message ID****Description****Dedicated Tx Buffers**

- Wording user manual

In case that multiple dedicated Tx Buffers are configured with the same Message ID, the Tx Buffer with the lowest buffer number is transmitted first.

- Enhancement - additional text

These Tx Buffers shall be requested in ascending order with lowest buffer number first.

Alternatively all Tx Buffers configured with the same Message ID can be requested simultaneously by a single write access to TXBARI.

**Tx Queue**

- Wording user manual - to be deleted

In case that multiple Queue Buffers are configured with the same Message ID, the Queue Buffer with the lowest buffer number is transmitted first.

- Replacement

If multiple Tx Queue buffers are configured with the same Message ID, the transmission order depends on the numbers of the buffers where the messages were stored for transmission. As these buffer numbers depend on the current states of the Put index, a prediction of the transmission order is not possible.

- Wording user manual - to be deleted

An Add Request cyclically increments the Put Index to the next free Tx Buffer.

- Replacement

The Put Index always points to that free buffer of the Tx Queue with the lowest buffer number.

**Scope**

Use of multiple dedicated Tx Buffers or Tx Queue buffers configured with same Message ID.

---

**2 Functional deviations****Effects**

In case the dedicated Tx Buffers with the same Message ID are not requested in ascending order or at the same time or in the case of multiple Tx Queue buffers with the same Message ID, it cannot be guaranteed, that these messages are transmitted in ascending order with lowest buffer number first.

**Workaround**

In case a defined order of transmission is required the Tx FIFO shall be used for the transmission of messages with the same Message ID. Alternatively dedicated Tx Buffers with same Message ID shall be requested in ascending order with the lowest buffer number first or by a single write access to TXBARi. Alternatively a single Tx Buffer can be used to transmit those messages one after the other.

**2.93 [MCMCAN\_AI.024] Frame transmitted despite confirmed transmit cancellation****Description**

In case the transmission of Tx Buffer z was not successful and is restarted immediately afterwards by automatic retransmission, and the software requests a Tx cancellation for this Tx Buffer by setting the cancellation request bit TXBCRi.CRz during transmission of the first 4 identifier bits, a successful cancellation is wrongly signalled by setting TXBCFi.CFz = '1' and by clearing TXBRPi.TRPz. In addition, the respective transmission occurred bit remains zero (TXBTOi.TOz = '0'), wrongly indicating that the frame was not transmitted on the bus.

Other than signalled by TXBCFi.CFz and TXBTOi.TOz, the transmission continues until the complete frame has been sent on the CAN bus. If the transmission is successful, TXBTOi.TOz will be set.

If in this case new data is written to Tx Buffer z while the transmission is still ongoing, a frame with inconsistent data may appear on the bus.

**Scope**

This problem is limited to the case when automatic retransmission is enabled (CCCRi.DAR = '0'). Erroneous signaling of the relevant status flags (as described in above section) happens irrespective of the CAN frame types and length. The below described effect of inconsistent data in the transmitted frame happens only for CAN FD messages with more than 8 data bytes. Classical CAN and CAN FD messages with less than 8 data bytes are not affected.

**Effects**

When bit TXBRPi.TRPz of Tx Buffer z is reset by an incomplete transmit cancellation, this Tx Buffer is reported to be "free". In case the software now writes new data to this Tx Buffer while a transmission is still ongoing, it may happen that this new data is loaded into the protocol controller, leading to a data inconsistency of the transmitted frame, meaning that the transmitted frame consists partly of the data available at start of frame and data written to the Tx Buffer during the ongoing transmission.

**Workaround**

Do not use transmit cancellation for CAN FD messages with more than 8 data bytes.

Alternatively wait for the duration of the expected transmission time of the cancelled Tx Buffer before writing new data to that Tx Buffer. The duration of the waiting time can be shortened when a new frame is received or transmitted before the end of the expected transmission time of the cancelled Tx Buffer.

**2 Functional deviations**

**2.94 [MCMCAN\_AI.025] Sporadic data corruption (payload) in case acceptance filtering is not finished before reception of data R3 (DB7..DB4) is completed**

**Description**

During frame reception the Rx Handler accesses the external Message RAM for acceptance filtering (read accesses) and for storing accepted messages (write accesses).

The time needed for acceptance filtering and for storing a received message depends on:

- the host clock frequency ( $f_{MCMCANH}$ )
- the worst-case latency of the read and write accesses to the external Message RAM
- the number of configured filter elements
- the workload of the transmit message (Tx) handler in parallel to the receive message (Rx) handler

Received data bytes (DB0..DBm) from the CAN Core are buffered in the cache of the Rx Handler before they are written to the Message RAM (in words of 4 byte). Data words inside the Message RAM are numbered from R2 to Rn ( $n \leq 17$ ).

	31	24 23	16 15	8	7	0	
R0	ESI	XTD	RTR	ID[28:0]			
R1	ANMF	FIDX[6:0]	res	FDF	BRS	DLC[3:0]	RXTS[15:0]
R2		DB3[7:0]		DB2[7:0]		DB1[7:0]	DB0[7:0]
R3		DB7[7:0]		DB6[7:0]		DB5[7:0]	DB4[7:0]
...		...		...		...	...
Rn		DBm[7:0]		DBm-1[7:0]		DBm-2[7:0]	DBm-3[7:0]

**Figure 4 TC3xx - Rx Buffer and FIFO Element (R<sub>i</sub> holds Data Byte DB(x+3)..DBx (with  $x=4*(i-2)$ ))**

Under the following conditions a received message will have corrupted data while the received message is signaled as valid to the host.

1. The data length code (DLC) of the received message is greater than 4 ( $DLC > 4$ )
2. The storage of R<sub>i</sub> of a received message into the Message RAM (after acceptance filtering is done) has not completed before R<sub>(i+1)</sub> is transferred from the CAN Core into the cache of the Rx Handler (where  $2 \leq i \leq 5$ )
3. While condition 1) and 2) apply, a concurrent read of data word R<sub>i</sub> from the cache and write of data word R<sub>(i+1)</sub> into the cache of the Rx handler happens

The data will be corrupted in a way, that in the Message RAM R<sub>(i+1)</sub> has the same content as R<sub>i</sub>.

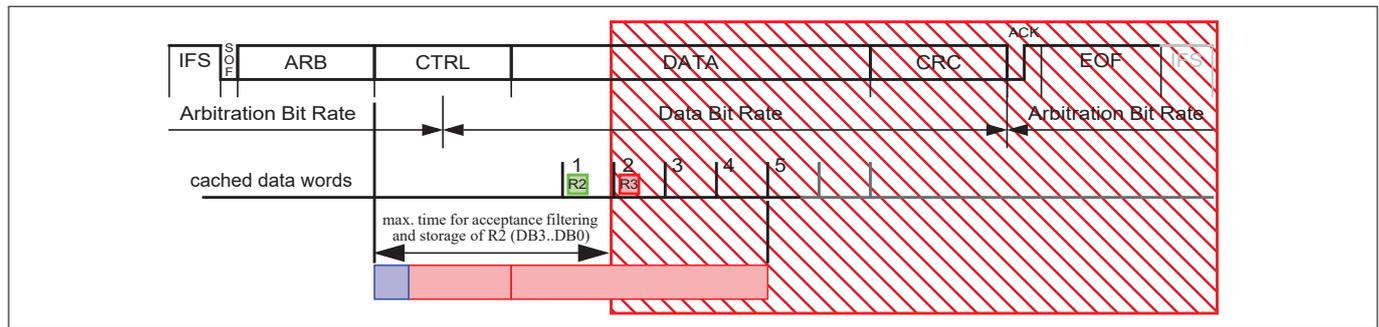
Despite the corrupted data, the M\_CAN signals the storage of a valid frame in the Message RAM:

- Rx FIFO: FIFO put index RXFnS.FnPI is updated
- Dedicated Rx Buffer: New Data flag NDATn.NDxx is set
- Interrupt flag IR.MRAF is not set

The issue may occur in FD Frame Format as well as in Classic Frame Format.

The figure that follows shows how the available time for acceptance filtering and storage is reduced.

**2 Functional deviations**



**Figure 5** CAN Frame with DLC > 4

**TC3xx: Minimum host clock frequency for CAN FD when DLC = 5**

**Table 10** TC3xx: Minimum host clock frequency for CAN FD when DLC = 5

		Arbitration bit rate = 0.5 Mbit/s			Arbitration bit rate = 1 Mbit/s		
No. of configured Active FE <sup>1) 2)</sup>	Number of Active CAN nodes	Data bit rate = 1 Mbit/s	Data bit rate = 2 Mbit/s	Data bit rate = 4 Mbit/s	Data bit rate = 2 Mbit/s	Data bit rate = 4 Mbit/s	Data bit rate = 5 Mbit/s
32/16	2	8	14	23	15	27	32
	3	10	19	32	20	37	44
	4	13	24	41	26	47	57
64/32	2	14	25	44	27	50	60
	3	19	35	61	38	70	84
	4	25	45	78	49	90	108 <sup>3)</sup>
96/48	2	20	37	64	40	74	89
	3	28	52	90	56	103 <sup>3)</sup>	124 <sup>3)</sup>
	4	36	67	116 <sup>3)</sup>	72	133 <sup>3)</sup>	160 <sup>3)</sup>
128/64	2	27	49	85	53	98	118 <sup>3)</sup>
	3	37	68	119 <sup>3)</sup>	74	136 <sup>3)</sup>	164 <sup>3)</sup>
	4	48	88	153 <sup>3)</sup>	95	175 <sup>3)</sup>	211 <sup>3)</sup>

- 1) M\_CAN starts always at filter element #0 and proceeds through the filter list to find a matching element. Acceptance filtering stops at the first matching element and the following filter elements are not evaluated for this message. Therefore the sequence of configured filter elements has a significant impact on the performance of the filtering process.
- 2) Acceptance filtering search for 11-bit IDs and 29-bit IDs filter element is running separately, only one configured filter setting should be considered. Searching for one 29-bit filter element requires double cycles for one 11-bit filter element.
- 3) Frequency not reachable since the maximum host clock frequency for MCMCAN in TC3xx is 100 MHz.

**Scope**

The erratum is limited to the case when the host clock frequency used in the actual device is below the limit shown in section "TC3xx: Minimum host clock frequency for CAN FD when DLC = 5".

## 2 Functional deviations

### Effects

Corrupted data is written to the Rx FIFO element or the dedicated Rx Buffer. The received frame is nevertheless signaled as valid.

### Workaround

Check whether the minimum host clock frequency, that is shown in section "TC3xx: Minimum host clock frequency for CAN FD when DLC = 5", is below the host clock frequency used in the actual device.

If yes, there is no problem with the selected configuration.

If no, use one of the following two workarounds.

#### Workaround 1

Try different configurations by changing the following parameters until ensuring that the actual synchronous clock  $f_{\text{MCANH}}$  frequency is above the minimum host clock frequency shown in section "TC3xx: Minimum host clock frequency for CAN FD when DLC = 5".

- Increase the  $f_{\text{MCANH}}$  in the actual device
- Reduce the CAN-FD data bit rate
- Reduce the number of configured filter elements and use a combination of 11-bit IDs and 29-bit IDs filter elements for one node
- Reduce the number of active M\_CANs

Also, use DLC  $\geq 8$  instead of DLCs 5, 6 and 7 in the CAN environment/system, as they place higher demands on the minimum  $f_{\text{MCANH}}$  (the worst case is DLC=5) or restrict your CAN environment/system to DLC=4.

**Note:** While changing the actual host clock frequency,  $f_{\text{MCANH}}$  must always be equal or higher than  $f_{\text{MCAN}}$  for all configurations.

#### Workaround 2

Due to condition 3) the issue only occurs sporadically. Use an end-to-end (E2E) protection (for example, checksum or CRC covering the data field) and add it to all messages in the CAN system, to detect data corruption in received frames.

## 2.95 [MCMCAN\_TC.006] MCMCAN specific access protection mechanisms

### Description

As described in the section "Registers" of the MCMCAN chapter of the TC3xx user manual, the MCMCAN module provides the following access enable registers:

- ACCEN0: protects the complete address space of MCMCAN including ACCENCTR0 and ACCENNODEi0
- ACCENCTR0: protects the control registers MCR, BUFADR, MECR, and MESTAT
- ACCENNODEi0: protects all the registers of the respective node i (i=0..3) and RAM range defined in the registers STARTADRi (i=0..3) and ENDADRi (i=0..3)

If an access violation takes place to the address space protected by ACCEN0, the access will be blocked and a bus error will be triggered.

Facing an access violation to a protected entity by either ACCENCTR0 or ACCENNODEi0, there will be no bus error triggered, however the access will be blocked.

### Effects

No bus error will be triggered in case of an access violation by using ACCENCTR0 or ACCENNODEi0.

**2 Functional deviations**

**Note:** *This issue has no safety impact since the node-based access protection covered by ACCENCTRL0 and ACCENNODEi0 provides freedom from interference between CAN nodes and triggering a bus error caused by such violation is not required in the safety case for AURIX™ 2nd generation.*

**Workaround**

No workaround is needed to ensure the access protection.

If bus error notification is required upon a MCMCAN access violation, use only the mechanism provided by ACCEN0.

**2.96 [MCMCAN\_TC.007] Incorrect access condition of bit-fields in the user manual**

**Description**

In the MCMCAN chapter of the TC3xx user manual, the access condition mentioned in the column "Type" of the following bit-fields is incorrect:

- TEST[i].LBCK
- TEST[i].TX
- CCCR[i].MON
- PSR[i].TDCV

**Scope**

The issue is related to documentation only.

**Documentation update**

The access condition mentioned in the column "Type" must be corrected as indicated by the following table:

<b>Bit-field</b>	<b>Wrong access condition</b>	<b>Correct access condition</b>
TEST[i].LBCK	rwh	rw
TEST[i].TX	rwh	rw
CCCR[i].MON	rwh	rw
PSR[i].TDCV	r	rh

**2.97 [MEMMAP\_TC.001] Size of PFLASH and DFLASH - Correction to TC33xEXT and TC33x/TC32x Appendix**

**Description**

**Note:** *This issue only affects V1.6.0 and V2.0.0 of the TC33xEXT and the TC33x/TC32x Appendix.*

Versions V1.6.0 and V2.0.0 include incorrect sizes and address ranges for PFLASH (3 Mbyte instead of 2 Mbyte) and DFLASH DF0 (1 Mbyte instead of 128 Kbyte) in table “Address Map as seen by Bus Masters on Bus SRI” in the MEMMAP chapter of the TC33xEXT and TC33x/TC32x Appendix.

Earlier versions (V1.2.0 .. V1.5.0) of the TC33xEXT and TC33x/TC32x Appendix correctly specify the sizes and address ranges for PFLASH (2 Mbyte) and DFLASH DF0 (128 Kbyte).

**2 Functional deviations**

**Documentation correction**

The sizes and address ranges for PFLASH and DFLASH DF0 in table “Address Map as seen by Bus Masters on Bus SRI” in the MEMMAP chapter of the TC33xEXT and TC33x/TC32x Appendix V1.6.0 and V2.0.0 shall be corrected as shown in the following table.

**Table 11 Address Map as seen by Bus Masters on Bus SRI - Corrections**

Address Range		Size	Unit
from	to		
80000000 <sub>H</sub>	801FFFFFF <sub>H</sub>	2 Mbyte	Program Flash (PFI0)
80200000 <sub>H</sub>	8FDFFFFFF <sub>H</sub>	-	Reserved
A0000000 <sub>H</sub>	A01FFFFFF <sub>H</sub>	2 Mbyte	Program Flash (PFI0_NC)
A0200000 <sub>H</sub>	A7FFFFFF <sub>H</sub>	-	Reserved
AF000000 <sub>H</sub>	AF01FFFF <sub>H</sub>	128 Kbyte	Data Flash 0 EEPROM (DF0) and Host Command
AF020000 <sub>H</sub>	AF3FFFFFF <sub>H</sub>	-	Reserved

**2.98 [MTU\_TC.012] Security of CPU cache memories during runtime is limited**

**Description**

MTU chapter “Security Applications” in the User’s Manual describes that selected memories with potentially security relevant content are initialized under certain conditions to prevent reading of their data or supplying manipulated data.

The description is correct, but the initialization of CPU cache and cache tag memories triggered by MBIST enable/disable and when mapping/un-mapping these memories to/from system address space using MEMMAP register is of limited value:

- These memories stay functional as cache in the address mapped state. Therefore software can enable address mapping and afterwards watch cache usage of the application (this is a debug feature). Even manipulation of the cache content is feasible
- It is possible to abort an ongoing memory initialization

The security of memory initialization during startup is not affected. Also protection of FSI0 and HSM memories is not limited.

**Workaround**

Handle security relevant data exclusively inside HSM. Protect the application code by locking external access (for example lock debug interface, prevent boot via serial interface). Consider validation of application code by HSM secure boot.

**2.99 [MTU\_TC.017] Unexpected alarms after application reset**

**Description**

As described in the MTU chapter “Alarms after startup” section, in case of an application reset, there are no SSH alarms or status bits expected to be triggered.

However, this device deviates from this expected behavior, and status flags AG0.SF10 and AG1.SF10 (DMEM Uncorrectable critical error) are set also after an application reset. Correspondingly, the OPERR[0] bits of the following SSHs are also set in the corresponding MCI\_FAULTSTS registers after an application reset:

- MC0 (CPU0\_DMEM)

## 2 Functional deviations

- MC34 (CPU0\_DMEM1), and
- MC35 (CPU1\_DMEM1)

**Note:** *In contrast to alarms resulting from real errors, for these unexpected alarms after application reset  $MCi\_ERRINFO = 0x0$  ( $i = 0, 34, 35$ ).*

### Workaround

The application software may clear the above mentioned alarms and errors after an application reset if  $MCi\_ERRINFO = 0x0$  ( $i = 0, 34, 35$ ), and proceed.

In case these errors occur during normal application run, this shall be considered as a real error.

## 2.100 [MTU\_TC.018] Gated SRAM alarms

### Description

Due to a corner case, SRAM alarms to the SMU for SRAM errors are not correctly generated for the following modules.

- GTM: ALM6[10], ALM6[11]
- DMA, SCR: ALM6[19], ALM6[20]
- CPUx: ALMx[4], ALMx[7], ALMx[10] ( $x = 0..n$ ; n depends on number of CPUs available in product)

### Background

From the SRAMs, the following errors are triggered to the SMU:

- ECC-correctable error: Triggered on a read access to SRAM
- ECC-uncorrectable error: Triggered on a read access to SRAM
- Address error: Triggered on read or write access to SRAM

In case of an error, normally these alarms are triggered appropriately on each read or write access.

However, due to this corner case, for certain SRAMs mentioned above, the alarm is not triggered on the read or write access on which the error is generated, rather, it is generated only on the **next** access to the SRAM or to an SSH register (for example  $MCx\_ECCD$  register).

**Note:** *Only the SMU alarm generation is affected by this issue and not the error triggering to the module. For example, error notification to GTM MCS still works as expected and the MCS may be stopped on an uncorrectable ECC error.*

Additionally, only the alarm propagation is gated in this corner case, that is the error status is still correctly stored in the  $MCx\_ECCD$ ,  $MCx\_FAULTSTS$  registers.

### Workaround

- For GTM & SCR SRAMs:  
Read the  $MCx\_ECCD$  register periodically, depending on application safety considerations, for example within each diagnostic test interval.  
Corresponding SSH instances:
  - GTM: MC53..MC60
  - SCR: MC77, MC78
- For DMA & CPU SRAMs (except  $DLMUx\_STBY$ ):  
No workaround is recommended, because here the issue affects only the address error generation on a write access. In this case, the next read access (when the data would be used) will trigger the error.
- For  $DLMU\_STBY$ :

## 2 Functional deviations

The issue occurs in a corner case just before entering standby mode. Therefore, if standby mode is used and Standby RAM is enabled ( $PMSWCR0.STBYRAMSEL \neq 000_B$ ) - then just before entering standby, perform an additional dummy read to DLMU\_STBY location 0x9000 0000 or 0xB000 0000 (when using CPU0 dLMU RAM) and 0x9001 0000 or 0xB001 0000 (when using CPU1 dLMU RAM). This dummy read triggers the alarm propagation and ensures that no alarms are lost due to standby entry.

### 2.101 [PADS\_TC.011] Pull-ups activate on specific analog inputs upon PORST

#### Description

If  $HWCFG[6] = 1$  or  $PMSWCR5.TRISTREQ = 0$ , respectively, the following analog inputs in the  $V_{DDM}$  domain:

- analog inputs overlaid with general purpose inputs (class S pads) on all pins of P40 and P41<sup>1)</sup>
- analog inputs (class D pads) of channels with multiplexer diagnostics<sup>2)</sup>

will activate internal pull-ups during cold or warm PORST.

When PORST is deasserted and the internal circuitry is reset, the inputs mentioned above will be released to tri-state mode.

**Note:** *This behavior differs from the description in the “Ports” chapter of the User’s Manual (P40/P41 always in tri-state mode during PORST) and the Data Sheet (corresponding pins marked with symbol “HighZ” in columns for buffer/pad type of the pin definition tables).*

### 2.102 [PADS\_TC.013] Buffer type definition for P21.2: no ES functionality - Data Sheet documentation correction

#### Description

As described in section “Exceptions for Emergency Stop Implementation” in the Ports chapter of the User’s Manual and its appendix, the Emergency Stop function is not available for P21.2 (can be used as EMGSTOPB pin).

Erroneously, P21.2 is marked with symbol “ES” (= Supports Emergency Stop) in column “Buffer Type” in the Data Sheet.

#### Correction to Data Sheet

Symbol “ES” shall be removed for P21.2 in column “Buffer Type” in the Data Sheet.

### 2.103 [PADS\_TC.016] Pull-ups active on P33 and P34 pins in standby mode when SCR is disabled and VEXT not supplied

#### Description

In the figure "Standby entry on VEXT ramp-down and wake-up on VEXT ramp-up" in the PMS and PMSLE chapters of the TC3xx user manual, the "Pin behavior" part shows that a pin state during and after wake-up from standby is configurable by bit  $PMSWCR5.TRISTREQ$  as pull-up or tristate.

<sup>1</sup> Availability depends on TC3xy device version, see the product specific Data Sheet.

<sup>2</sup> These channels are explicitly marked with (MD) in table “Analog Input Connections for Product TC3xy” in the EVADC chapter of the product specific appendix to the AURIX™ TC3XX User’s Manual.

## 2 Functional deviations

### Current documentation

For P33 and P34 pins which are supplied by VEVRSB, the following behavior is expected in standby mode as per the description mentioned in the "Pin behavior" part:

- If VEXT pins are not supplied, then VEVRSB supplied SCR pins are under SCR control (implies SCR is enabled)
- If SCR is not enabled and VEXT pins are not supplied, then VEVRSB supplied pins (P33, P34) are in tristate

However, the last part stating that P33 and P34 pins are in tristate during standby mode if the SCR is not enabled and VEXT pins are not supplied, is not correct.

### Actual behavior

When the SCR is not enabled and VEXT pins are not supplied during standby, then pull-ups are active on all P33 and P34 pins in standby mode.

### Workaround 1

When the SCR is not enabled and VEXT pins are not supplied during standby mode, and the application requires a low level on P33 and P34 pins in standby mode, external pull-downs (in the range of  $> 2$  kOhm and  $< 4.7$  kOhm) should be added to the corresponding P33 and P34 pins.

In order to quantify the strength of such an external pull-down, parameter "Pull-up current" ( $I_{PUH}$ ) for the respective pin may be used as the reference, and the value for the external pull-down can be calculated accordingly.

### Workaround 2

Enable the SCR during standby mode.

### Workaround 3

Supply VEXT pins during standby mode.

## 2.104 [PER\_PLL\_TC.002] Peripheral PLL K3 Divider Operation

### Description

The clock output  $f_{PLL2}$  of the K3 divider (K3-DIV) of the Peripheral PLL may be not working (no clock) or having wrong clock frequency (wrong K3 divider setting). This issue may occur after Peripheral PLL power-up, during the restart of the lock detection of the Peripheral PLL (PERPLLCON0.RESLD = 1), or PERPLLCON0.DIVBY change which is typically triggered in the MCU clock initialization phase.

The issue of no clock is detected by the clock alive monitor of  $f_{PLL2}$ . The issue of the wrong K3 divider setting is detected by the clock plausibility safety mechanism (ESM[SW]:CLOCK:PLAUSIBILITY).

### Workaround

To avoid this potential issue which occurs after the Peripheral PLL power-up, during the restart of the lock detection of Peripheral PLL (PERPLLCON0.RESLD = 1), or PERPLLCON0.DIVBY change, K3 divider setting (PERPLLCON1.K3DIV) must be enabled equal to 0x0 or 0x1 only.

### Additional hints for clock init routine

The recommended enable routine for  $f_{PLL2}$  monitor should include time out (recommended value 100  $\mu$ s) for checking the CCUCON3.LCK bit after the  $f_{PLL2}$  monitor is enabled by CCUCON3.PLL2MONEN=0x1.

If the clock.alive\_monitor is triggered or the CCUCON3.LCK bit is locked after the recommended timeout of 100  $\mu$ s, please re-initialize the clock and check again.

The additional indicator for  $f_{PLL2}$  K3 divider issue could be that the PERPLLSTAT.K3-RDY bit is set to 0x0.

## 2 Functional deviations

The application hint for the clock alive monitor mechanism SM[HW]:CLOCK:ALIVE\_MONITOR must be respected to ensure proper operation of the clock alive monitors:

- The application software shall enable the clock alive monitor by setting the corresponding bit in CCUCON3 register after PLLs have been set up and are running

**Note:** *If clock alive monitor alarm and clock plausibility are configured and are not flagging any non-recoverable errors in the customer application, then the customer system is robust for the issue and does not need any additional action.*

### 2.105 [PMS\_TC.005] Voltage rise at P33 and P34 up to $V_{EVRSB}$ during start-up and up to $V_{LVDRSTSB}$ during power-down

#### Description

The HWCFG pins (located in the  $V_{EXT}$  domain) information is evaluated when basic supply and clock infrastructure components are available as the supplies  $V_{EVRSB}$  and  $V_{EXT}$  ramp up. Tristate control information based on HWCFG[6] latched with  $V_{EXT}$  supply ramp can't be used within the  $V_{EVRSB}$  supply domain until both supplies ( $V_{EXT}$  and  $V_{EVRSB}$ ) have reached the minimum threshold value of  $V_{LVDRST5}$  and  $V_{LVDRSTSB}$ , respectively. Therefore, the pad behavior at P33 and P34 pins is “pull-up”, even if pin HWCFG[6] = 0, with the following characteristics:

- the pad voltage level rises to  $V_{EVRSB}$  until the  $V_{LVDRSTSB}$  and  $V_{LVDRST5}$  thresholds of  $V_{EVRSB}$  and  $V_{EXT}$  are reached during the ramp-up phase
- the pad voltage level is below  $V_{LVDRSTSB}$  for the ramp-down phase of the  $V_{EVRSB}$  supply

#### Workaround

If an application requires to ensure the state of P33 and P34 pins within the logical “low” level, then an external pull-down must be used which can overdrive the internal pull-up.

In order to quantify the strength of such an external pull-down, parameter “Pull-up current” ( $I_{PUH}$ , CC) for the respective pin may be used as the reference. There, the values for the internal pull-up resistor (for TTL and AL) can be found via parameter  $R_{MDU}$  in table “VADC 5V” (see footnotes on parameter “Pull-up current” in the Data Sheet).

### 2.106 [PMS\_TC.006] PORST not released during cold power-on reset until $V_{DDM}$ is available

#### Description

Upon a cold power-on reset, the PORST pin may be kept asserted by the PMS until the ADC Analog Supply voltage ( $V_{DDM}$ ) is above 500 mV. This might lead to an additional start-up delay dependent on when  $V_{DDM}$  is available from the external regulator relative to the  $V_{EXT}$ ,  $V_{DDP3}$ , and  $V_{DD}$  supplies. When  $V_{DDM}$  is below 500 mV, the device may not be able to carry out PBIST. As a consequence, the device remains in PORST state until PBIST result gets available and supply voltages are in operating condition. PBIST result is always available when  $V_{DDM}$  is above 500 mV. From PBIST point of view, it is important to have enough  $V_{DDM}$  voltage level, and there is no special timing requirement on when to supply  $V_{DDM}$ .

During operation, if  $V_{DDM}$  drops below the secondary monitor undervoltage threshold, an SMU alarm is generated. If  $V_{DDM}$  further drops below 500 mV, the dedicated ADC of the secondary voltage monitor stops converting and the Secondary Monitor Activity Counter (EVRMONSTAT1.ACTVCNT) freezes at the last value.

#### Workaround

The ADC Analog Supply voltage ( $V_{DDM}$ ) has to be available and needs to be above 500 mV to ensure proper release of PORST during start-up and proper functioning of secondary monitors.

**2 Functional deviations**

User external regulator ramp-up sequence shall be analyzed to avoid unexpected delay or even potential deadlock.

For example:

- User selects EVRC to generate  $V_{DD}$
- User external regulator will only supply  $V_{DDM}$  when its voltage monitor detects that  $V_{DD}$  has reached a certain level
- But when  $V_{DDM}$  is below 500 mV, the device may not be able to carry out PBIST and the device remains in PORST state and accordingly EVRC cannot generate  $V_{DD}$
- Therefore, deadlock could happen

**2.107 [PMS\_TC.007] VDDP3 or VDD Overvoltage during start-up may not be detected by PBIST****Description**

In AURIX™ TC3xx devices, Power Built in Self Test (PBIST) is introduced to ensure that the supply voltages do not exceed absolute maximum limits during the start-up phase.

However, for a VDDP3 or VDD overvoltage event during start-up beyond operational upper limits, the PBIST is not able to detect this overvoltage event.

**Workaround**

Check the VDDP3 overvoltage condition in registers EVRSTAT (flag OV33) and EVRMONSTAT1 (field ADC33V) in software additionally during the start-up phase before enabling the corresponding SMU alarm.

Check the VDD overvoltage condition in registers EVRSTAT (flag OVC) and EVRMONSTAT1 (field ADCCV) in software additionally during the start-up phase before enabling the corresponding SMU alarm.

**2.108 [PMS\_TC.011] VEXT supplied PU2 and PD2 pads always in tristate after standby entry - Documentation correction****Description**

Tristate mode is enabled for VEXT supplied PU2 and PD2 pads (marked PU2 / VEXT and PD2 / VEXT in column “Buffer Type” in the Data Sheet) at the moment of and after entry to standby mode, regardless of the PMSWCR5.TRISTREQ bit setting and the HWCFG[6] pin setting (reflected in the PMSWSTAT register).

For a definition of the buffer types see also chapter “Legend” in the Data Sheet.

**Recommendation**

If the application requires the pull-up state of VEXT supplied PU2 pads (or pull-down state of PD2 pads), then it shall ensure it by means of external pull-up devices (or pull-down devices for PD2 pads) in the event of:

- Standby entry while the VEXT supply ramps down
- Standby entry with the VEXT supply available

**Documentation correction for TC3xx User’s Manual V1.5.0 and following**

In TC3xx User’s Manual V1.5.0 and following versions, the description of this behavior has been included in the PMS and PMSLE chapters. Erroneously, the term “PU1” was used instead of “PU2 and PD2”.

## 2 Functional deviations

In the following sections and sentences in chapter PMS (\*=11) and PMSLE (\*=12), the term “PU1” shall be replaced by “**PU2 and PD2**”:

- Section \*.2.1.1 Supply Mode Selection:
  - „Regardless of the HWCFG[6] setting, the VEXT-buffered PU1 pads (see the PU1 buffer type in the data sheet) are set into tristate ..“ shall be replaced by
  - “Regardless of the HWCFG[6] setting, the VEXT-buffered **PU2 and PD2** pads (see the **PU2 and PD2** buffer type in the data sheet) are set into tristate ..”
- Section \*.2.3.4.8 Entering Standby Mode (only VEVRSB domain supplied):
  - “Regardless of the PMSWCR5.TRISTREQ setting, the VEXT-buffered PU1 pads (see the PU1 buffer type in the data sheet) are set into tristate ..” shall be replaced by
  - “Regardless of the PMSWCR5.TRISTREQ setting, the VEXT-buffered **PU2 and PD2** pads (see the **PU2 and PD2** buffer type in the data sheet) are set into tristate ..”
- Section \*.2.3.4.9 Entering Standby Mode (both VEVRSB and VEXT domain supplied):
  - “Regardless of the PMSWCR5.TRISTREQ setting, the VEXT-buffered PU1 pads (see the PU1 buffer type in the data sheet) are set into tristate ..” shall be replaced by
  - “Regardless of the PMSWCR5.TRISTREQ setting, the VEXT-buffered **PU2 and PD2** pads (see the **PU2 and PD2** buffer type in the data sheet) are set into tristate ..”
- Section \*.2.3.4.10 State during Standby Mode:
  - “Regardless of the PMSWCR5.TRISTREQ setting, the VEXT-buffered PU1 pads (see the PU1 buffer type in the data sheet) are set into tristate ..” shall be replaced by
  - “Regardless of the PMSWCR5.TRISTREQ setting, the VEXT-buffered **PU2 and PD2** pads (see the **PU2 and PD2** buffer type in the data sheet) are set into tristate ..”

See also the corresponding entries in the revision history for PMS chapter V2.2.31 and PMSLE chapter V1.0.4 at the end of each chapter.

### 2.109 [PMS\_TC.014] Parasitic coupling on shared ADC pins depending on supply voltages

#### Description

Bulk diodes exist from the  $V_{EXT}$  supply rail to the  $V_{DDM}$  supply rail through respective shared analog pins of EVADC Group 9 (P00.1 - P0.12).

If  $V_{EXT} > V_{DDM}$  and any of the shared pin voltages ( $V_{INPIN}$ ) is higher than  $V_{DDM}$  by a diode voltage ( $V_{Diode} \sim 0.6V$ ), i.e.

- $V_{INPIN} > (V_{DDM} + V_{Diode})$  OR  $V_{INPIN}$  pulled up to  $V_{EXT}$  by internal/external pull-ups  $> (V_{DDM} + V_{Diode})$

then during start-up and operation, sink currents will flow from the pin to the  $V_{DDM}$  supply. The currents shall be limited by an internal/external pull-up resistor in order to stay within the overload conditions.

#### Behavior during start-up

Only during the start-up phase, when the  $V_{DDM}$  supply voltage is less than the  $V_{DDPPA}$  ( $\sim 1.3V$ ) subthreshold limit, then the shared analog pins within an ADC multiplexer group of EVADC group G9 are internally connected together. The internal connection is high ohmic in nature (current  $< 100 \mu A$ ). Consequently an external pull-up on one pin may be visible on the other pins in the same EVADC multiplexer group until the  $V_{DDM}$  supply is above the  $V_{DDPPA}$  limit and LVD reset limits on  $V_{EXT}$  and  $V_{EVRSB}$  have been reached.

#### Workaround

To avoid any current flow from  $V_{INPIN}/V_{EXT}$  to  $V_{DDM}$  and to prevent parasitic coupling on shared ADC pins:

**2 Functional deviations**

- It needs to be ensured that the shared pin voltages ( $V_{INPIN}$ ) are within the ( $V_{DDM} + V_{Diode}$ ) supply range. Alternatively,  $V_{DDM}$  and  $V_{EXT}$  may be supplied together from the same supply source if the pull-ups on the pins are to the  $V_{EXT}$  rail
- When both  $V_{EXT}$  and  $V_{EVRSB}$  are kept supplied during Standby mode,  $V_{DDM}$  should also be kept supplied if shared analog pins are pulled high

**Note:** *Related to this text module, in TC3xx User’s Manual versions after V1.6, the row for  $V_{DDM}$  in table “5 V Nominal Supply: Voltage variations at independent supply rails during system modes” will be updated accordingly, and a diagram “Parasitic Diode Connectivity between supply rails” will be added.*

**2.110 [PMS\_TC.015] EVRC synchronization – Documentation update for register EVRSDCTRL11 (PMS) and EVRSDCTRL2 (PMSLE)**

**Description**

The formulas for  $d f_{MAXDEV}$  (Maximum Deviation of the Synchronization Input Frequency) and SYNCHYST (Lock Unlock Hysteresis Window) that are documented in the description of fields SYNCMAXDEV and SYNCHYST in register EVRSDCTRL11 (chapter PMS) and EVRSDCTRL2 (chapter PMSLE) of the TC3xx User’s Manual shall be corrected/updated as listed below.

**SYNCMAXDEV in TC3xx User’s Manual V2.0.0 (and earlier versions)**

- $d f_{MAXDEV} = 100 \text{ MHz} * (2 * \text{SYNCMAXDEV}) / (\text{SDFREQ}^2 + \text{SYNCMAXDEV}^2)$
- $\text{SYNCMAXDEV} = \text{round} [ (100 \text{ MHz} / d f_{MAXDEV}) - \text{sqrt}\{ (100 \text{ MHz} / d f_{MAXDEV})^2 - \text{SDFREQ}^2 \} ]$

**Correction to SYNCMAXDEV in register EVRSDCTRL11 (PMS) and EVRSDCTRL2 (PMSLE)**

- $d f_{MAXDEV} = 100 \text{ MHz} * (2 * \text{SYNCMAXDEV}) / (\text{SDFREQ}^2 - \text{SYNCMAXDEV}^2)$
- $\text{SYNCMAXDEV} = \text{round} [ \text{sqrt}\{ (100 \text{ MHz} / d f_{MAXDEV})^2 + \text{SDFREQ}^2 \} - (100 \text{ MHz} / d f_{MAXDEV}) ]$

**SYNCHYST in TC3xx User’s Manual V2.0.0 (and earlier versions)**

- $\text{SYNCHYST} = \text{round} [ d f_{HYST} * (\text{SDFREQ} \pm \text{SYNCMAXDEV})^2 ] / [ d f_{HYST} * (\text{SDFREQ} \pm \text{SYNCMAXDEV}) + 100 \text{ MHz} ]$

**Correction/Update to SYNCHYST in register EVRSDCTRL11 (PMS) and EVRSDCTRL2 (PMSLE)**

- $\text{SYNCHYST} = \text{round} [ d f_{HYST} * (\text{SDFREQ} \pm \text{SYNCMAXDEV})^2 / (100 \text{ MHz} \pm d f_{HYST} * (\text{SDFREQ} \pm \text{SYNCMAXDEV})) ]$
- First hysteresis band:
  - $d f_{HYST} = 100 \text{ MHz} / (\text{SDFREQ} + \text{SYNCMAXDEV} - \text{SYNCHYST}) - 100 \text{ MHz} / (\text{SDFREQ} + \text{SYNCMAXDEV})$
- Second hysteresis band:
  - $d f_{HYST} = 100 \text{ MHz} / (\text{SDFREQ} - \text{SYNCMAXDEV}) - 100 \text{ MHz} / (\text{SDFREQ} - \text{SYNCMAXDEV} + \text{SYNCHYST})$

**2.111 [PORTS\_TC.009] PCSR register incompletely documenting use for EVADC PDD and MD feature - Update to TC33x/TC32x appendix**

**Description**

In ports with analog inputs to the EVADC the PCSR register enables control of pull devices for the Pull Down Diagnostics (PDD) / Multiplexer Diagnostics (MD) feature.

However, in the PORTS chapter of the TC33x/TC32x appendix V2.0.0 (and earlier versions), the following register bits are documented as reserved and their name is **Rx**:

- bit 5 of register P40\_PCSR associated with EVADC G8CH1 on P40.5

---

**2 Functional deviations**

- bit 10 of register P00\_PCSR associated with EVADC G9CH2 on P00.10
- bit 11 of register P00\_PCSR associated with EVADC G9CH1 on P00.11

**Documentation update**

Correct would be to name the following bits as **SELx**:

- bit 5 of register P40\_PCSR (symbolic name SEL5)
- bit 10 of register P00\_PCSR (symbolic name SEL10)
- bit 11 of register P00\_PCSR (symbolic name SEL11)

with the description:

“This bit enables or disables EVADC control of the pulls for Pull Down Diagnostics (PDD) / Multiplexer Diagnostics (MD) feature”

and with documentation of encoding:

0<sub>B</sub> Disable EVADC PDD/MD feature of pin x

1<sub>B</sub> Enable EVADC PDD/MD feature of pin x”

**2.112 [QSPI\_TC.006] Baud rate error detection in slave mode (error indication in current frame)****Description**

According to the specification, a baud rate error is detected if the incoming shift clock supplied by the master has less than half or more than double the expected baud rate (determined by bit-field GLOBALCON.TQ).

However, in this design step, a baud rate error is detected not only if the incoming shift clock has less than half the expected baud rate (as specified), but also already when the incoming shift clock is somewhat (i.e. less than double) higher than the expected baud rate.

In this case, the baud rate error is indicated in the current frame.

**Workaround**

It is recommended not to rely on the baud rate error detection feature, and not to use the corresponding automatic reset enable feature (i.e. keep GLOBALCON.AREN = 0<sub>B</sub>).

The baud rate error detection feature in slave mode is of conceptually limited use and is not related to data integrity. Data integrity can be ensured for example by parity, CRC, etc., while clocking problems of an AURIX™ master are detected by mechanisms implemented in the master.

Protection against the effects of high frequency glitches is provided by the spike detection feature in slave mode.

**2.113 [QSPI\_TC.009] USR Events for PT1=2 (SOF: Start of Frame)****Description**

In master mode, when the interrupt on USR event is associated with Start of Frame (i.e. USREN=1<sub>B</sub>, PT1=2 in register GLOBALCON1, BACON.UINT=1<sub>B</sub>), then flag STATUS.USRF is not set and the interrupt is not triggered for the first frame.

**Workaround**

In the configuration where the interrupt on USR event is associated with Start of Frame (i.e. USREN=1<sub>B</sub>, PT1=2 in GLOBALCON1, BACON.UINT=1<sub>B</sub>), first transmit a “dummy” frame with this configuration. Then, for all subsequent frames, flag USRF will be set and the interrupt on USR event will be generated as expected.

---

## 2 Functional deviations

### **2.114 [QSPI\_TC.010] Move Counter Mode - USR Events for PT1=4 (RBF: Receive Buffer Filled)**

#### **Description**

When a master operates in Move Counter Mode (MCCON.MCEN=1<sub>B</sub>), and the interrupt on USR event is associated with Receive Buffer Filled (i.e. USREN=1<sub>B</sub>, PT1=4 in register GLOBALCON1), the enable signal in BACON.UINT is only evaluated at the start of frame event.

This means in an ongoing frame the status of UINT in the first BACON control word involved determines whether flag STATUS.USRF is set and a user interrupt is generated or not. The status of UINT in following BACON control words in this frames' transmission is not considered.

#### **Workaround**

In case the Receive Buffer Filled event shall only be used as interrupt on USR event for parts of a frame, initialize for example BACON.UINT=1<sub>B</sub> and GLOBALCON.PT1=4 before start of frame, and use GLOBALCON1.USREN to selectively disable/enable the user interrupt during frame transmission.

### **2.115 [QSPI\_TC.013] Slave: No RxFIFO write after transmission upon change of BACON.MSB**

#### **Description**

While a slave transmission is in progress, and if the BACON.MSB configuration is changed for the subsequent frame, then the RxFIFO write of the currently received frame may not occur.

Also in case of a TxFIFO underflow, the RxFIFO write of the currently received frame may not occur.

#### **Workaround**

As a general recommendation, in slave mode the configuration should be done before any transmission starts. In particular to avoid the problem described above, the re-configuration of the BACON has to be done after the RxFIFO write has occurred. This implies the need for a gap between frames if a BACON update occurs.

### **2.116 [QSPI\_TC.014] Slave: Incorrect parity bit upon TxFIFO underflow**

#### **Description**

When a slave TxFIFO underflow occurs, the slave transmits only "ones" in response to a request of the master. If parity is enabled, also the parity bit transmitted by the slave is always set to "1". This may be incorrect, depending on data length and parity type.

#### **Workaround**

If parity is enabled, select even parity if data length is odd, and select odd parity if data length is even.

### **2.117 [QSPI\_TC.016] Master: Move Counter Mode - Counter underflows when data is present in the TXFIFO while in the last TRAIL state of the previous transaction**

#### **Description**

When a master operates in move counter mode (MCCON.MCEN = 1<sub>B</sub>) and is configured for adjacent move counter transactions, the MC.CURRENT counter value underflows when the move counter transaction is in the last TRAIL state of the previous transaction and the TXFIFO is already filled with data for the next move counter

## 2 Functional deviations

transaction. Due to this there is a possibility that the next move counter transaction enters an EXPECT state expecting more frames and stays there until intervened by the software.

Therefore, TXFIFO shall not be filled with the next move counter transaction data before the current transaction is over.

### Workaround

The End of Frame (EOF) phase transition interrupt (i.e. GLOBALCON1.PT1 = 101<sub>B</sub> or GLOBALCON1.PT2 = 101<sub>B</sub>) shall only be used to trigger the CPU/DMA to fill the TXFIFO with the next move counter transaction data.

### 2.118 [QSPI\_TC.017] Slave: Reset when receiving an unexpected number of bits

#### Description

A deactivation of the slave select input (SLSI) by a master is expected to automatically reset the bit counter of the QSPI module when configured as a slave.

This reset should help slaves to recover from messages where faults in the master or glitches on SCLK lead to an incorrect number of clocks on SCLK (= incorrect number of bits per SPI frame).

However, in this design step, the reset of the bit counter is unreliable.

#### Workaround

The slave should enable the Phase Transition interrupt (PT2EN = 1<sub>B</sub> in register GLOBALCON1) to be triggered after the PT2 event “SLSI deselection” (PT2 = 101<sub>B</sub>).

- **TC3xx:** In the interrupt service routine, after ensuring that the receive data has been copied, the software should issue a reset of the bit counter and the state machine via GLOBALCON.RESETS = 01<sub>B</sub>
- **TC2xx:** In the interrupt service routine, after ensuring that the receive data has been copied, the software should issue a reset of the bit counter and the state machine via GLOBALCON.RESETS = 0111<sub>B</sub>

### 2.119 [SAFETY\_TC.023] MCU infrastructure Safety Related Function - Documentation update

#### Description

**Note:** This issue applies to AURIX™ TC3xx Safety Manual version v2.0.

Section 4.3.1 (Introduction) of chapter “Safety Related Functions” in the AURIX™ TC3xx Safety Manual v2.0 mentions in the last bullet point below the table that Safety Related Functions 10, 11 and 12 shall always be correctly implemented in order to reach the ASIL level of the listed Safety Related Functions.

The listed absolute numbers 10, 11, 12 are not correct in this context.

#### Documentation Update

The MCU infrastructure Safety Related functions **12, 13 and 14** are assumed to be always correctly implemented.

### 2.120 [SAFETY\_TC.024] Clock alive monitor for $f_{SPB}$ - Documentation update

#### Description

The AURIX™ TC3xx Safety Manual (v2.0 and earlier versions) states in section 6.37 SM[HW]:CLOCK:ALIVE\_MONITOR that the clock alive monitor for  $f_{SPB}$  is only visible to HSM.

## 2 Functional deviations

This statement is not correct.

### Documentation update

The clock alive monitor for  $f_{SPB}$  is visible to all interfaces in the SMU.

### 2.121 [SAFETY\_TC.025] Wrong alarm listed in safety mechanism SM[HW]:SRI:SRI\_TRANSACTION\_INTEGRITY

#### Description

In SM[HW]:SRI:SRI\_TRANSACTION\_INTEGRITY (section 6.477 in AURIX™ TC3xx Safety Manual v1.04 and higher versions), ALM11[12] “(CONVERTER) Phase Synchronizer Error” is listed as fault identification interface of the SM[HW]:SRI:SRI\_TRANSACTION\_INTEGRITY safety mechanism.

This statement is not correct.

### Documentation update

ALM11[12] must not be considered as fault identification interface of the SM[HW]:SRI:SRI\_TRANSACTION\_INTEGRITY safety mechanism.

### 2.122 [SAFETY\_TC.026] Alarm for SM[HW]:IR:CFG\_MONITOR - Documentation update

#### Description

In SM[HW]:IR:CFG\_MONITOR (section 6.268 in AURIX™ TC3xx Safety Manual v1.04 and higher versions) the alarm “ALM8[22] - EDC Configuration & Data Path Error” is listed as fault identification interface of SM[HW]:IR:CFG\_MONITOR.

This statement is not correct.

### Documentation update

Alarm “ALM10[22] - IR ACCEN Error Event” will be generated if a fault is detected by SM[HW]:IR:CFG\_MONITOR.

### 2.123 [SAFETY\_TC.027] Single point fault detection for lockstep CPUs - Documentation update

#### Description

The following safety mechanisms listed in chapter 6 (Safety Mechanisms) of the AURIX™ TC3xx Safety Manual

- SM[HW]:CPU:CRC
- SM[HW]:CPU:TPS
- SM[HW]:CPU:TPS\_EXCEPTION\_TIME\_MONITOR
- SM[HW]:CPU:CODE\_MPU
- SM[HW]:CPU:DATA\_MPU
- SM[HW]:CPU:UM0
- SM[HW]:CPU:UM1
- SM[HW]:CPU:SV
- SM[HW]:CPU:STI

mention “Single Point Fault Detection = Yes” in column “lockstep CPU” of the table included in the description of the respective safety mechanism.

---

**2 Functional deviations**

This statement is not correct.

The lockstep CPU is protected by SM[HW]:CPU:TRICORE\_LOCKSTEP in any case and all the other SMs listed above are only used for freedom from interference.

**Documentation update**

For the safety mechanisms listed above, the corresponding entry in column “lockstep CPU” shall be corrected to

- “Single Point Fault Detection = N.A.”

**2.124 [SAFETY\_TC.029] Allow software writes to the OLDA address range in a safe system****Description**

In the AURIX™ TC3xx Safety Manual, the sub-chapter "Purpose of the SEooC" lists the features which are to be used only during the system development phase. This list also includes the OLDA (OnLine Data Acquisition) as a development feature, which is incorrect.

**Recommendation**

The OLDA feature can be used in production software as stated in the AURIX™ TC3xx User's Manual in the sub-chapter "OnLine Data Acquisition (OLDA)".

**2.125 [SCR\_TC.015] Bit SCU\_PMCON1.WCAN\_DIS does not disable WCAN PCLK input****Description**

Setting bit SCU\_PMCON1.WCAN\_DIS to 1<sub>B</sub> has no effect – the WCAN clock input (PCLK) is not disabled. Power consumption of the WCAN module will not decrease as expected.

**Workaround**

In order to keep power consumption at a minimum, the WCAN module must not be enabled (WCAN\_CFG.WCAN\_EN = 0<sub>B</sub>).

**2.126 [SCR\_TC.016] DUT response to first telegram has incorrect C\_START value****Description**

**Note:** *This problem is only relevant for tool development, not for application development.*

The C\_START value returned by the SCR OCDS of the DUT (device under test) in response to a first telegram is wrong.

Each monitor processed command starts with sending a telegram containing the CMD (for example READ\_BYTE). The response to this telegram should be a telegram containing the C\_START value of 0x1.

Instead, the value sent by the DUT is a random value.

**Workaround**

Do not evaluate the return value of the first telegram from the DUT. Even though the returned C\_START is wrong, the returned checksum is correct, and should be checked with the theoretical C\_START value of 0x01.

## 2 Functional deviations

### 2.127 [SCR\_TC.018] SSC Receive FIFO not working

#### Description

The receive FIFO of the SSC module is not working properly. An unexpected receive FIFO full indication can be set.

#### Workaround

Do not use the receive FIFO.

Read the received data from the receive buffer register SSC\_RBL each time a receive interrupt event is signaled (flag IRCON1.RIR).

The received data must be read before the next data is received.

### 2.128 [SCR\_TC.019] Accessing the XRAM while SCR is in reset state

#### Description

When accessing the XRAM while the SCR is executing a reset, the following erroneous behavior will occur:

- A read access returns 0 instead of the actual XRAM contents
- A write access has no effect, the data will not be written to the XRAM

#### Workaround

One of the following methods will avoid this problem:

1. Check the SCR reset status bit PMSWSTAT.SCRST before and after any read/write transaction to the XRAM:
  - a. If the bit is set before the transaction, clear bit PMSWSTAT.SCRST and perform the desired XRAM access
  - b. If the bit is set after the transaction, clear bit PMSWSTAT.SCRST and repeat the XRAM read/write access. OR
2. Disable the SCR generated reset sources. OR
3. Disable the entire SCR (no SCR reset can occur) by the following steps:
  - a. Set SCR\_RSTCON.ECCRSTEN = 0<sub>B</sub> disabling double bit ECC reset before an uncorrectable error is happening in XRAM. Bit ECCRSTEN may be set to 0<sub>B</sub> either by explicitly writing to ECCRSTEN, or by triggering a local SCR reset which also clears the ECCRSTEN bit. Alternatively, if ECC reset generation is not disabled and an XRAM ECC error happens later, a periodic reset is triggered to MTU till the reset trigger is serviced. The permanent reset to MTU can be resolved by shortly enabling the SCR and disabling it again to service the pending ECC error triggered reset
  - b. Set PMSWCR0.SCRWKEN = 0<sub>B</sub> – wake-up via SCR disabled
  - c. Set PMSWCR4.SCREN = 0<sub>B</sub> – SCR disabled

### 2.129 [SCR\_TC.020] Stored address in mon\_RETH may be wrong after a break event

#### Description

**Note:** *This problem is only relevant for tool development, not for application development.*

When setting a breakpoint via the SCR debugger connection on address XXFE<sub>H</sub> of an instruction, the stored address in mon\_RETH is wrong if mon\_RETL contains 00<sub>H</sub> (see also section “Calculation of the return address upon a break event” in the SCR chapter). This effect will happen whenever a carry bit should be propagated from the lower 8 bits to the upper 8 bits of the address.

## 2 Functional deviations

### Workaround

If mon\_RETL contains 00<sub>H</sub> after a breakpoint was hit, the debugger tool must increment mon\_RETH by 1 before performing the calculation of the return address as described in section “Calculation of the return address upon a break event” in the SCR chapter.

### 2.130 [SCR\_TC.021] RTC not counting after reset if P33.10 is high

#### Description

The Real-Time Clock (RTC) in the SCR module may not reliably start counting if a high level was present on P33.10 (SCR\_P01.2) during LVD reset. If enabled, the RTC will only start counting after the first high-to-low transition on P33.10 (SCR\_P01.2).

**Note:** Applications using an external (32 or 32.768 kHz) oscillator on P33.10 as clock source for the RTC are not affected.

#### Workaround 1

Ensure a low level on P33.10 (SCR\_P01.2) during LVD reset, for example via a pull-down.

#### Workaround 2

Generate a high-to-low transition on P33.10 (SCR\_P01.2) after LVD reset (by software or external hardware).

### 2.131 [SCR\_TC.022] Effect of application or system reset and warm PORST on MC77\_ECCD and MC78\_ECCD for SCR RAMs

#### Description

Unlike for ECCD registers of other modules, error flags in MC77\_ECCD (for SCR\_XRAM) and MC78\_ECCD (for SCR\_RAMINT) are not cleared upon application or system reset.

As consequence the corresponding alarms ALM6[19], ALM6[20] and ALM6[21] in AG6 are not cleared by an application and system reset (if ECCD is not cleared by SW before triggering the reset).

Furthermore, flags in MC77\_ECCD are not cleared upon warm PORST.

#### Workaround

Clear flags in register MC77\_ECCD and MC78\_ECCD via software by writing '0' to the respective bits.

### 2.132 [SCR\_TC.023] External interrupts EXINT0, EXINT1 may get locked

#### Description

As described in chapter “Interrupt System” of the SCR chapter in the TC3xx User’s Manual, if the external interrupt is positive (negative) edge triggered, the external source must hold the request pin low (high) for at least one CCLK cycle, and then hold it high (low) for at least one CCLK cycle to ensure that the transition is recognized.

However, for external interrupts EXINT0 and EXINT1, respectively, if the time between two triggering edges is shorter than 2 CCLK cycles, no further interrupt request is triggered after the first triggering edge. Further EXINT0 or EXINT1 interrupts are locked until the next application reset.

**Note:** This problem only occurs if interrupt generation on both rising and falling edge is selected, i.e. for EXINT0 if EXICON0.EXINT0 = 10<sub>B</sub>, and for EXINT1 if EXICON0.EXINT1 = 10<sub>B</sub>, respectively.

**2 Functional deviations**

**Workaround**

If using interrupt generation on both edges, ensure that the time between two triggering edges for EXINT0, EXINT1 is  $> 2 \text{ CCLK cycles}$ . To include some margin for clock jitter and external signal slope asymmetries etc., the external source should hold the request pin low (high) for example for  $2.1/f_{\text{CCLK}}$  to ensure that the transitions are correctly recognized.

Otherwise, only use external interrupts EXINT2..15.

**2.133 [SCR\_TC.024] Field ADRES in register ADCOMP\_RES - Documentation correction**

**Description**

In chapter “ADC Comparator Unit (ADCOMP)” of the SCR chapter in the TC3xx User’s Manual, erroneously the term “ADCRES” is used instead of “ADRES” in the text and in figure “ADC Comparator Overview”.

**Documentation correction**

The term “ADCRES” shall be replaced by “ADRES” within the text of chapter “ADC Comparator Unit (ADCOMP)”, and in figure “ADC Comparator Overview”.

In addition, the text in column “Description” for field ADRES in the ADCOMP Result Register ADCOMP\_RES shall be corrected as follows:

**Table 12 Field ADRES in register ADCOMP\_RES - correction**

Field	Bits	Type	Description
ADRES	7:0	rh	<p><b>ADC Conversion Result</b></p> <p>This register shows the current converted ADC result. Software should ensure that the result is read before starting the next conversion.</p> <ul style="list-style-type: none"> <li>For ADRES &gt; 1: <ul style="list-style-type: none"> <li>VIN = [LSB * (ADRES-1)]; LSB = 23.077 mV. Full Range: 5861.54 mV</li> </ul> </li> <li>For ADRES ≤ 1: software shall assume VIN as 0 V</li> </ul>

**2.134 [SCR\_TC.033] [IR] External Interrupts 0 and 1 are not able to exit the Idle Mode of XC800 core**

**Description**

External Interrupts 0 and 1 are not detected in the Idle Mode.

**Scope**

External Interrupts

**Effects**

Idle Mode cannot be exited by asserting External Interrupt 0 or 1.

**Workaround**

To exit the Idle Mode, any other External Interrupt (2 to 15) except 0 and 1 can be used.

2 Functional deviations

**2.135 [SCU\_TC.031] Bits SCU\_STSTAT.HWCFGx (x=1-5) could have an unexpected value in application if pins HWCFGx are left unconnected**

**Description**

An unexpected value for the HWCFGx pin state (x=1-5) may be latched in register field SCU\_STSTAT.HWCFGx after application reset if the corresponding HWCFGx pin is not externally connected to a pull-up or a pull-down and the default reset state of port pins is set to tristate (pin P14.4/HWCFG[6] is pulled to GND).

EVR3 start-up function after cold reset is not affected (HWCFG2).

EVR33 start-up function after cold reset is not affected (HWCFG1).

Only the intended function of HWCFG[3-5] pin configuration options in the corresponding reset cases is affected when BMI.PINDIS=0<sub>B</sub> and DMU\_HF\_PROCONTP.BML=00<sub>B</sub> (application boot defined by HWCFG[3-5] pins).

**Workaround**

Do not leave pins HWCFGx (x=1-5) unconnected if the default reset state of port pins is set to tristate (HWCFG[6] pulled to GND).

**Note:** *This is not a general option for devices in QFP-80 and QFP-100 packages where P14.2/HWCFG2 is internally left unconnected.*

If HWCFG2 is left unconnected, alternatively the application shall not rely on bit SCU\_STSTAT.HWCFG[2] and may check for the correct state in the registers PMSWSTAT.HWCFGEVR or EVRSTAT.EVRC.

**2.136 [SCU\_TC.033] TESTMODE pin shall be held at static level during LBIST**

**Description**

The MISR signatures documented in the product specific TC3xy Appendix to the TC3xx User’s Manual are only valid if the TESTMODE pin (P20.2) is always kept at a static **high** level during LBIST execution. This is the recommended LBIST configuration.

For a stable MISR signature, the level on this pin must not change during LBIST execution.

**Workaround**

For application environments where pin TESTMODE is not held high, but a static **low** level is applied to TESTMODE, a different MISR signature will be received in the LBISTCTRL3.SIGNATURE field, depending on bit LBISTCTRL1.BODY.

**Table 13 Contents of LBISTCTRL3 if TESTMODE is low during LBIST**

Device	Design step	LBISTCTRL3	
		BODY = 0	BODY = 1
TC39x	BA..BC	0x07EC4205	0xFE614D6
	BD	0x935E836A	0x6A14D5B9
TC38x	AA..AD	0x1E1CD10C	0xA7587528
	AE	0x3AD1859B	0x839521BF
TC37xEXT	AA, AB	0x01E51F27	0xEDB9BD01
TC37x	AA	0x62DD6AB1	0xA373D89F

(table continues...)

**2 Functional deviations**

**Table 13** (continued) Contents of LIBISTCTRL3 if  $\overline{\text{TESTMODE}}$  is low during LBIST

Device	Design step	LIBISTCTRL3	
		BODY = 0	BODY = 1
TC36x	AA	0xD833B421	0xF53338B3
TC35x	AB	0xBA38B3CB	0x1CACD3CE
TC33xEXT	AA	0xD1298927	0x1A824479
TC33x/TC32x	AA	0xC12B66CB	0x3F84CD94
TC3Ex	AA	0x4D0F493B	0xE7764C81

**2.137 [SCU\_TC.036] Concurrent reset requests from CERBERUS do not result in all reset requests captured in reset status register**

**Description**

In the RSTSTAT register, there is a remark about a concurrent reset request from OCDS/Cerberus that is supposed to also set bit RSTSTAT[16] =  $\sim$ .PORST.

This does not work as described.

It is possible to set the 3 bits in the OCDS/Cerberus, causing 3 reset requests to the RCU (Reset Control Unit). This also would result in the 3 resets being activated as expected.

But the status after deactivation of all resets is not:

- RSTSTAT[20] =  $\sim$ .CB3 = '1' (application reset request)
- RSTSTAT[19] =  $\sim$ .CB1 = '1' (debug reset request)
- RSTSTAT[18] =  $\sim$ .CB0 = '1' (system reset request)
- RSTSTAT[16] =  $\sim$ .PORST = '1'

as expected, but:

- RSTSTAT[20] =  $\sim$ .CB3 = '1'
- RSTSTAT[19] =  $\sim$ .CB1 = '1'
- RSTSTAT[18] =  $\sim$ .CB0 = '0'
- RSTSTAT[16] =  $\sim$ .PORST = '0'

Because the system reset request from OCDS/Cerberus is asynchronously cleared in the OCDS/Cerberus with the system reset issued on the chip and the other reset requests from OCDS/Cerberus are synchronously cleared and so these reset requests are active longer - till after reset deactivation the SPB-clock is switched on again.

**Scope**

Reset status (SCU\_RSTSTAT.CB\*) inside SCU during a debug session.

**Effects**

The issue described above leads to a new latching of reset requests after reset deactivation and the right RSTSTAT contents is overwritten with the remaining active reset request sources (CB1, CB3) from OCDS/Cerberus.

**Workaround**

No workaround available.

## 2 Functional deviations

### 2.138 [SMU\_TC.012] Unexpected alarms when registers FSP or RTC are written

#### Description TC2xx

Due to a synchronization issue, ALM3[27] is sporadically triggered if the PRE2 field of register FSP is written while the SMU is configured in Time Switching protocol (FSP.MODE = 10<sub>B</sub>) and FSP[0] is toggling with a defined  $T_{SMU\_FFS}$  period.

Also, ALM3[27] is sporadically triggered if the PRE1 or TFSP\_HIGH fields of register FSP are written while the SMU is in the Fault State and  $T_{FSP\_FS}$  has not yet been reached (STS.FSTS=0<sub>B</sub>) (regardless of the FSP.MODE configuration).

In addition, an unexpected ALM2[29] or ALM2[30] is sporadically triggered if field FSP.PRE1 or RTC.RTD is written, and at least one recovery timer is running based on a defined  $T_{SMU\_FS}$  period (regardless of the FSP.MODE configuration).

The alarms can only be cleared with cold or warm Power-On reset.

#### Description TC3xx

Due to a synchronization issue, ALM6[7] and ALM10[21] are sporadically triggered if the PRE2 field of register FSP is written while the SMU is configured either

- in Time Switching protocol (FSP.MODE = 10<sub>B</sub>) and FSP[0] is toggling with a defined  $T_{SMU\_FFS}$  period
- or in Dual Rail protocol (FSP.MODE = 01<sub>B</sub>) and FSP[1:0] are toggling with a defined  $T_{SMU\_FFS}$  period

Also, ALM6[7] and ALM10[21] are sporadically triggered if the PRE1 or TFSP\_HIGH fields of register FSP are written while the SMU is in the Fault State and  $T_{FSP\_FS}$  has not yet been reached (STS.FSTS=0<sub>B</sub>) (regardless of the FSP.MODE configuration).

In addition, an unexpected ALM10[16] or ALM10[17] is sporadically triggered if field FSP.PRE1 or RTC.RTD is written, and at least one recovery timer is running based on a defined  $T_{SMU\_FS}$  period (regardless of the FSP.MODE configuration).

The alarms can only be cleared with cold or warm Power-On reset.

#### Workaround TC2xx

To avoid unexpected alarms, perform the configuration of the PRE1, PRE2 or TFSP\_HIGH fields only when the SMU is not in the Fault State and FSP is in Bi-stable protocol mode (FSP.MODE = 00<sub>B</sub>). Mode switching and configuration shall not be done with the same write access to register FSP.

This means that in the Fault Free State:

- before writing to PRE1, PRE2 or TFSP\_HIGH while Time Switching protocol is enabled:
  - disable Time Switching protocol by setting FSP in Bi-stable protocol mode (FSP.MODE = 00<sub>B</sub>);
  - wait until Bi-stable protocol mode is active (read back register FSP twice);
  - write desired value to PRE1, PRE2 or TFSP\_HIGH;
  - then switch FSP.MODE to the desired protocol (optional step)
- If the mode shall be changed after writing to PRE1, PRE2 or TFSP\_HIGH while in Bi-Stable protocol mode (FSP.MODE = 00<sub>B</sub>):
  - write desired value to PRE1, PRE2 or TFSP\_HIGH;
  - then switch FSP.MODE to Time Switching protocol

If field FSP.PRE1 or RTC.RTD shall be written, make sure no recovery timer is running. It is not allowed to write to the PRE1 or RTD field when at least one recovery timer is running (indicated by bits RTS0 and RTS1 in the STS register).

#### Workaround TC3xx

To avoid unexpected alarms, perform the configuration of the PRE1, PRE2 or TFSP\_HIGH fields only when the SMU is not in the Fault State and FSP is in Bi-stable protocol mode (FSP.MODE = 00<sub>B</sub>). Mode switching and configuration shall not be done with the same write access to register FSP.

## 2 Functional deviations

This means that in the Fault Free State:

- before writing to PRE1, PRE2 or TFSP\_HIGH while Time Switching or Dual Rail protocol is enabled:
  - disable Time Switching or Dual Rail protocol by setting FSP in Bi-stable protocol mode (FSP.MODE = 00<sub>B</sub>);
  - wait until Bi-stable protocol mode is active (read back register FSP twice);
  - write desired value to PRE1, PRE2 or TFSP\_HIGH;
  - then switch FSP.MODE to the desired protocol (optional step)
- If the mode shall be changed after writing to PRE1, PRE2 or TFSP\_HIGH while in Bi-Stable protocol mode (FSP.MODE = 00<sub>B</sub>):
  - write desired value to PRE1, PRE2 or TFSP\_HIGH;
  - then switch FSP.MODE to Time Switching or Dual Rail protocol

If field FSP.PRE1 or RTC.RTD shall be written, make sure no recovery timer is running. It is not allowed to write to the PRE1 or RTD field when at least one recovery timer is running (indicated by bits RTS0 and RTS1 in the STS register).

### 2.139 [SMU\_TC.013] Unexpected setting of Alarm Missed Event bit xAEM in Alarm Executed Status register SMU\_AEX

#### Description

**Note:** *This problem only applies to alarms of Alarm Type: Level (see tables “Alarm Mapping related to ALM\* group” in the product specific Appendix to the TC3xx User’s Manual).*

While servicing an alarm with alarm type Level, request status bit xSTS in the SMU\_AEX register is set. However, the corresponding alarm missed event bit xAEM is also set, 1 cycle after the xSTS bit is set for the same alarm event (x can be any of IRQ0..2, RST0..5, NMI, EMS).

#### Workaround

While clearing the xSTS bit the corresponding xAEM bit should also be cleared for the alarm event.

If the xAEM bit is not cleared while clearing xSTS, only the alarm missed event xAEM functionality will not be available for later alarm events, and it does not impact any alarm action generation and xSTS bit functionality.

### 2.140 [SMU\_TC.015] SMU alarm emulation might trigger unwanted active alarm reaction

#### Description

While the SMU is in START state, a level alarm is raised by the hardware and stays active. Since the SMU is in START state, the corresponding configured reaction, for example RESET\_REQ, is not triggered. In this context, another alarm for which there is no SMU reaction configured, is triggered by the software using the alarm emulation function of SMU. The expected behavior is that SMU does not react to the software triggered alarms, however, the alarm reaction of previously set alarm, for example RESET\_REQ, would unexpectedly be triggered.

#### Scope

Alarm emulation

#### Effects

Unintended alarm reaction. The actual reaction will depend on the configuration.

---

**2 Functional deviations****Workaround**

While the SMU is in START state, all alarms must be cleared in SMU and at the source safety mechanism before using the alarm emulation function.

**2.141 [SMU\_TC.017] Alarm type indication (level or pulse) for SBCU SPB and EBCU BBB bus error event****Description**

ALM7[20] SBCU SPB Error detection and ALM7[21] EBCU BBB Error detection are documented as pulse alarms, meaning that the alarm line is asserted with a pulse, and alarm handling through the SMU control interface. However, the respective alarm lines to the SMU are controlled by the SBCU\_ALSTATx and EBCU\_ALSTATx registers, resulting in a level alarm behavior. These registers must be cleared through SBCU\_ALCLR<sub>x</sub> and EBCU\_ALCLR<sub>x</sub> respectively. Only after this, the alarms can also be cleared in the SMU.

**Scope**

SBCU SPB and EBCU BBB alarm type.

SBCU and with that ALM7[20] is available in all devices of the TC3x family.

EBCU and with that ALM7[21] is only available in some devices of the TC3x family.

**Effects**

The handling of pulse alarms does not work to clear the alarm status of these alarms.

**Workaround**

Treat the alarms ALM7[20] and ALM7[21] as level alarms and clear the alarm source in the SBCU\_ALSTATx and EBCU\_ALSTATx registers through SBCU\_ALCLR<sub>x</sub> and EBCU\_ALCLR<sub>x</sub> respectively.

3 Parametric deviations

3 Parametric deviations

3.1 [ADC\_TC.P017] Increased RMS noise for TC33x/32x devices

Description

**Note:** This problem depends on the package type and affects the variants TC333\*/TC323\* (devices in TQFP-100 package), TC334\*/TC324\* (devices in TQFP-144 package), TC337\*/TC327\* (devices in BGA-292 package), and TC336\* (devices in BGA-180 package).

For these devices, on some or all channels the specified RMS noise ( $EN_{RMS}$ ) increases, independent of the noise reduction mode, as listed in Table 14 below.

**Table 14 RMS noise for TC33x and TC32x, independent of noise reduction mode**

Device Variant	Package	Analog channel ANx	$EN_{RMS}$ (Max.)
TC333*, TC323*	TQFP-100	AN37, AN38, AN39	1.0 LSB
TC334*, TC324*	TQFP-144	AN0, AN39	1.0 LSB
TC337*, TC327*	BGA-292	all ANx	1.0 LSB
TC336*	BGA-180	AN0	1.2 LSB
		AN2, AN3, AN5, AN6, AN39	1.0 LSB

3.2 [CCU\_TC.P001] Back-up clock accuracy after trimming - Disregard datasheet footnote

Description

The following text in the footnote on parameter “Back-up clock accuracy after trimming” in table “Back-up Clock” of the current TC3xx datasheets cannot be met under all operating conditions:

- A short term trimming providing the accuracy required by LIN communication is possible by periodic trimming every 2 ms for temperature and voltage drifts up to temperatures of 125° Celsius

This footnote shall be disregarded.

3.3 [FLASH\_TC.P003] Program Flash Erase Time per Multi-Sector Command

Description

The maximum value for parameter “Program Flash Erase Time per Multi-Sector Command” can be

- $t_{MERP} \leq 0.52$  s (instead of 0.5 s as specified in the Data Sheet)

Consequently, the maximum value for parameter “Complete Device Flash Erase Time PFlash and DFlash” can also increase by 0.04 s/Mbyte, resulting in

- **TC39x:**  $t_{ER\_Dev} \leq 19.14$  s (instead of 18.5 s as specified in the Data Sheet)
- **TC38x:**  $t_{ER\_Dev} \leq 11.9$  s (instead of 11.5 s as specified in the Data Sheet)
- **TC3Ex:**  $t_{ER\_Dev} \leq 14.98$  s (instead of 14.5 s as specified in the Data Sheet)
- **TC37x, TC37xEXT:**  $t_{ER\_Dev} \leq 7.24$  s (instead of 7 s as specified in the Data Sheet)
- **TC35x, TC36x:**  $t_{ER\_Dev} \leq 5.16$  s (instead of 5 s as specified in the Data Sheet)
- **TC33xEXT, TC33x/TC32x:**  $t_{ER\_Dev} \leq 3.08$  s (instead of 3 s as specified in the Data Sheet)

The increased values should be considered for example when defining erase timeout limits.

### 3 Parametric deviations

#### 3.4 [PADS\_TC.P014] Electrical characteristics for P20.2/TESTMODE

##### Description

The buffer type for P20.2/TESTMODE is defined in column "Buffer Type" of table "Port 20 Functions" in the Data Sheet as "class S/PU / VEXT".

The electrical characteristics specified in table "Class S 5V" and "Class S 3.3V" (if present in the Data Sheet) literally only apply to class S pads on P40 (and P41 if available), as they are connected to  $V_{DDM}$ .

##### Recommendation

Tables "Class S 5V" and "Class S 3.3V" apply analogously to the electrical characteristics of P20.2/TESTMODE, with  $V_{DDM}$  replaced by  $V_{EXT}$ .

**Note:** *The current Data Sheets for TC35x and TC33xEXT do not include tables for class S pads (they do not have P40). Therefore, please see the Data Sheet for one of the other TC3xx devices.*

**Note:** *The current Data Sheet for TC36x does not include table "Class S 3.3V". Therefore, please see the Data Sheet for one of the other TC3xx devices.*

#### 3.5 [PORST\_TC.P002] $V_{IH}$ and $V_{IL}$ definition for PORST pad - Additional Data Sheet footnote

##### Description

The following footnote shall be added in column "Note/Test Condition" of Data Sheet table "PORST Pad" to the parameters "Input high voltage level" (symbol  $V_{IH}$ ) and "Input low voltage level" (symbol  $V_{IL}$ ):

**Note:** *The levels defined are valid within the operating conditions of  $V_{EXT} = 5 V \pm 10\%$  or  $V_{EXT} = 3.3 V \pm 10\%$ , respectively.*

#### 3.6 [PWR\_TC.P015] Power pattern definition - Documentation update to TC33x/TC32x Data Sheet V1.1

##### Description

The TC33x/TC32x includes one TC1.6.2P core with lockstep (checker) core.

Therefore, the bullet points related to these cores in the description of the real and max power pattern in chapter "Power Supply Current" of the TC33x/TC32x Data Sheet shall be changed as follows:

##### Documentation update

The real (realistic) power pattern defines the following conditions:

- one core is active with lockstep core (IPC=0.6) (instead of "one cores is active without lockstep core (IPC=0.6)")

The max power pattern defines the following conditions:

- one core is active with lockstep core (IPC=1.2) (instead of "all cores are active including one lockstep core (IPC=1.2)")

---

### 3 Parametric deviations

#### 3.7 [VEVRSB\_TC.P001] Bonding of VEVRSB pad on LQFP packages - Data Sheet documentation correction

##### Description

Column “Note/Test Condition” in table “Operating Conditions” for parameter “Digital external supply voltage for EVR and during Standby mode” (symbol  $V_{\text{EVRSB}}$ ) of the Data Sheet contains the following statement:

- VEVRSB is bonded together with VEXT supply pin in smaller LQFP packages

This statement does not apply and shall be deleted.

**Note:** *Pin VEVRSB is not bonded together with VEXT supply pin and must be supplied accordingly. See also figure “TC33x Supply Pins and Module Connectivity” in the PMSLE chapter of the TC3xx User’s Manual which includes the statement “External supply is separated into sub-domains not connected internally: VEXT, VEVRSB, VFLEX, VDDM & VEBU”.*

## 4 Application hints

### 4 Application hints

#### 4.1 [ADC\_TC.H026] Additional waiting phase in slow standby mode

##### Description

When a conversion is requested while slow standby mode is configured and the respective converter currently is in standby state, the extended wakeup time  $t_{WU}$  must be added to the intended sample time (see section “Analog Converter Control” in the TC3xx User’s Manual).

While idle precharge is disabled ( $GxANCFG.IPE = 0_B$ ), an additional waiting phase of  $1.6 \mu s$  ( $@f_{ADC} = 160 \text{ MHz}$ ) is inserted automatically. Operation starts after this phase.

However, if the slow standby state is left after just 1 clock cycle, this waiting phase is omitted.

##### Recommendation

It is, therefore, recommended to add the specified extended wakeup time ( $t_{WU}$ ) when leaving the standby state in all cases, to ensure proper operation.

#### 4.2 [ADC\_TC.H032] ADC accuracy parameters - Definition

##### Description

Chapter “VADC Parameters” in the Data Sheet contains the following introduction section:

“The accuracy of the converter results depends on the reference voltage range. The parameters in the table below are valid for a reference voltage range of  $(V_{AREF} - V_{AGND}) \geq 4.5 \text{ V}$ . If the reference voltage range is below  $4.5 \text{ V}$  by a factor of  $k$  (e.g.  $3.3 \text{ V}$ ), the accuracy parameters increase by a factor of  $1.1/k$  (e.g.  $1.1 \times 4.5 / 3.3 = 1.5$ ).”

Accuracy parameters in the context of the statement above are:

- Total Unadjusted Error ( $TUE$ )
- INL Error ( $EA_{INL}$ )
- DNL Error ( $EA_{DNL}$ )
- Gain Error ( $EA_{GAIN}$ )
- Offset Error ( $EA_{OFF}$ )
- RMS Noise ( $EN_{RMS}$ )
- Converter diagnostics voltage accuracy ( $dV_{CSD}$ )
- Deviation of IVR output voltage  $V_{DDK}$  ( $dV_{DDK}$ )

#### 4.3 [ADC\_TC.H033] Basic initialization sequence for primary and secondary EVADC groups

##### Description

For consistency, to ensure that the maximum value for the settling time of the analog module is always considered in the basic initialization sequence, the start-up calibration should be started **after** a waiting time equal or higher than the extended wakeup time ( $t_{WU}$ ). The related basic initialization sequence is described in the following execution scheme.

**Note:** Compared to the sequence listed in chapter “Basic Initialization Sequence” in the EVADC chapter of TC3xx User’s Manual V1.2.0 and earlier versions, step “WAIT” (third step below) has been shifted **before** the begin of the start-up calibration.

## 4 Application hints

```
EVADC_GxANCFG = 0x00300000
;Analog clock frequency is 160 MHz / 4 = 40 MHz (example)
;CALSTC = 00
EVADC_GxARBCFG = 0x00000003 ;Enable analog block
WAIT ;Pause for extended wakeup time (≥ 5 μs)
;(other operations not related to EVADC can be executed in the meantime)
EVADC_GLOBCFG = 0x80000000 ;Begin start-up calibration
EVADC_GxARBPR = 0x01000000 ;Enable arbitration slot 0
EVADC_GxQMR0 = 0x00000001 ;Enable request source 0
EVADC_GxICLASS0 = 0x00000002
;Select 4 clocks for sampling time: 4 / 40 MHz = 100 ns
;The default setting stores results in GxRES0,
;service requests are issued on GxSR0
EVADC_GxRCR0 = 0x80000000
;Enable result service requests, if required
EVADC_GxQINR0 = 0x00000020
;Request channel 0 in auto-repeat mode
WAIT ;Wait for start-up calibration to complete *)
;(other operations not related to EVADC can be executed in the meantime)
;=> This starts continuous conversion of the channel
*)time tSUCAL or flag GxARBCFG.CAL=0
```

### 4.4 [ADC\_TC.H035] Effect of input leakage current on Broken Wire Detection

#### Description

The Broken Wire Detection (BWD) feature uses the sample capacitor of the ADC input to discharge (BWG: Broken Wire Detection against  $V_{AGND}$ ) or to charge (BWR: Broken Wire Detection against  $V_{AREF}$ ) the input node of the ADC.

This mechanism can be seen as small current sink (BWG) or current source (BWR). When the BWD feature is enabled, in case the ADC input is not connected to an external voltage source (i.e. the wire is broken), the ADC input voltage is drifting down or up. When a defined voltage level (i.e. the detection threshold) is reached, “broken wire detected” is claimed.

Broken Wire Detection currents  $I_{BWG}$ ,  $I_{BWR}$  are quite small and must overwhelm input leakage currents  $I_{OZ}$  on the same node. Input leakage currents depend exponentially on junction temperature.

It is therefore required to check whether an application using Broken Wire Detection can deal with leakage currents also under worst case conditions.

#### Application considerations

1. Get the input leakage current ( $I_{OZ}$ ) limits from the Data Sheet, depending on used ADC pins and maximum junction temperature  $T_J$  of your application
2. Compare this limit against the Broken Wire Detection currents  $I_{BWG}$ ,  $I_{BWR}$ , which can be calculated as follows:
  - Broken Wire Detection against  $V_{AGND}$  (BWG):
    - $I_{BWG} = V_{AIN} * C_{AINS} * CR$
  - Broken Wire Detection against  $V_{AREF}$  (BWR):
    - $I_{BWR} = (V_{AIN} - V_{AREF}) * C_{AINS} * CR$

## 4 Application hints

where

- $V_{AIN}$ : ADC input voltage at the detection threshold (typ. 10% of full scale for BWD, 80% of full scale for BWR)
- $C_{AINS}$ : ADC input sampling capacitance, typ. 2.5 pF
- $CR$ : Conversion Rate, that is number of conversions per second per input

### Recommendation

The absolute value of the Broken Wire Detection current ( $I_{BWG}$  or  $I_{BWR}$ ) at the BWD threshold shall be at least 2x the maximum input leakage current  $I_{OZ}$  (absolute value).

### Examples

#### 1. Typical case example

Assuming that  $T_J \leq 150^\circ\text{C}$  (max) and ADC inputs are used in a configuration where  $I_{OZ} \leq 150$  nA (see Data Sheet),  $I_{BWG}$  should be  $\geq 300$  nA according to the recommendation above.

With  $C_{AINS} = 2.5$  pF and  $V_{AIN} = 0.5$  V (10% of full scale) it can be calculated from the formula for  $I_{BWG}$  above that  $CR$  should be  $\geq 240\,000$  samples per second and input.

#### 2. Worst case example

Assuming that  $T_J \leq 170^\circ\text{C}$  (max) and ADC inputs are used in a configuration where  $I_{OZ} \leq 800$  nA (see Data Sheet),  $I_{BWG}$  should be  $\geq 1600$  nA according to the recommendation above.

With  $C_{AINS} = 2.5$  pF and  $V_{AIN} = 0.5$  V (10% of full scale) it can be calculated from the formula for  $I_{BWG}$  above that  $CR$  should be  $\geq 1\,280\,000$  samples per second and input.

### Recommendations for increasing the Broken Wire Detection current

In order to increase the Broken Wire Detection current,

1. Relax the detection threshold, for example for BWG from 10% to 20% of the full scale voltage
2. Increase the conversion rate  $CR$  per input by introducing additional conversions

## 4.5 [ADC\_TC.H040] Selection of masters for synchronization groups - Documentation update to TC33x/TC32x Appendix

### Description

The 4 converter kernels of the TC33x/TC32x can be connected to synchronization groups to achieve parallel conversion of several input channels.

Figure “Synchronization via ANON and Ready Signals” in the EVADC chapter of the TC3xx User’s Manual shows how a master is selected via bit-field  $GxSYNCTR.STSEL$ . In this figure, kernels ADC0 and ADC1 relate to converter group G0 and G1, while ADC2 and ADC3 relate to G8 and G9 of the TC33x/TC32x, respectively.

However, the mapping in the rows for G8 and G9 in the corresponding table “Synchronization Groups” in the EVADC chapter of the TC33x/TC32x Appendix V2.0.0 and earlier versions is incorrect and shall be updated as follows.

### Documentation Update

The following table summarizes which kernels can be synchronized for parallel conversions in the TC33x/TC32x (corrections in rows for G8 and G9 shown in **bold**).

**4 Application hints**

**Table 15 Synchronization Groups of TC33x/TC32x**

ADC Kernel	Synchr. Group	Master selected by control input C1x <sup>1)</sup>			
		C10 <sup>2)</sup>	C11	C12	C13
G0 (Prim.)	A	G0	G1	G8	G9
G1 (Prim.)	A	G1	G0	G8	G9
G8 (Sec.)	A	G8	<b>G0</b>	<b>G1</b>	<b>G9</b>
G9 (Sec.)	A	G9	<b>G0</b>	<b>G1</b>	<b>G8</b>

- 1) The control input is selected by bit-field STSEL in register GxSYNCTR. Select the corresponding ready inputs accordingly by bits EVALRx.
- 2) Control input C10 always selects the own control signals of the corresponding ADC kernel. This selection is meant for the synchronization master or for stand-alone operation.

**4.6 [ADC\_TC.H043] Information on supervision signal  $V_{ANACOMM}$  not relevant - Documentation update**

**Description**

The functionality of supervision signal  $V_{ANACOMM}$  listed in table “Supervision Signals” of the EVADC chapter in the TC3xx User’s Manual V2.0.0 (and earlier versions) can only be used during the Infineon production test. It cannot be used in a customer application.

**Documentation update**

Disregard the first line about  $V_{ANACOMM}$  in table “Supervision Signals” in the EVADC chapter of the TC3xx User’s Manual.

**4.7 [ADC\_TC.H044] Start-up calibration timing in synchronized mode - Documentation update**

**Description**

The formula for the start-up calibration duration  $t_{SUCAL}$  in the EVADC chapter of the TC3xx user manual is not valid for all of the different configurations:

- In the synchronized mode (GLOBCFG.USC=0, default after reset), each calibration step is waiting for a valid starting point and this prolongs the total calibration time

**Documentation update**

In section “Start-Up Calibration Timing” in the EVADC chapter of the TC3xx user manual, a second footnote <sup>2)</sup> shall be added to the formula for  $t_{SUCAL}$ :

- <sup>2)</sup> In the synchronized mode (USC=0),  $t_{SUCAL}$  can increase up to 200%

**4.8 [ADC\_TC.H045] Level selection for broken wire detection feature**

**Description**

The broken wire detection (BWD) feature uses the sample capacitor of the ADC input to discharge (BWD against  $V_{AGND}$ ) or to charge (BWD against  $V_{AREF}$ ) the input node of the EVADC. The level ( $V_{AREF}$  or  $V_{AGND}$ ) is selected in the bit-field BWDCH as documented in the description of register GxCHCTRY in the EVADC chapter of the TC3xx user manual.

## 4 Application hints

However, the text in the EVADC chapter only describes BWD against  $V_{AGND}$ , it does not explicitly describe BWD against  $V_{AREF}$ .

### Documentation update

The description related to broken wire detection (BWD) in the following parts of the EVADC chapter shall be updated to reflect both BWD against  $V_{AGND}$  and BWD against  $V_{AREF}$ :

- Section "Safety Features"

Broken wire detection (BWD) preloads the converter network with a selectable level before sampling the input channel. The result will then reflect the preload value if the input signal is no more connected

- Chapter "Broken Wire Detection"

To test the proper connection of an external analog sensor to its input pin, the converter's capacitor can be precharged to a selectable value before the regular sample phase. If the connection to the sensor is interrupted, the subsequent conversion value will rather represent the precharged value than the expected sensor result. By using a precharge voltage outside the expected result range (broken wire detection uses  $V_{AGND}$  or  $V_{AREF}$ ) a valid measurement (sensor connected) can be distinguished from a failure (sensor detached)

Broken wire detection can be enabled for each channel separately by the bit-field BW DEN in the corresponding channel control register (GxCHCTRY). The bit-field BWDCH selects the level for the preparation phase

## 4.9 [ADC\_TC.H046] Incorrect number of EVADC kernels in TC33x/TC32x Datasheet

### Description

In the TC33x/TC32x Datasheet, the section "Summary of features" contains the following statement:

- Cluster of 2 independent ADC kernels

The number of ADC kernels listed in the bullet point is incorrect. TC33x and TC32x have 2 primary and 2 secondary groups, as described in the table "Platform Feature Overview" in the TC33x/TC32x Datasheet.

### Documentation update

For TC33x and TC32x, the number of independent ADC kernels mentioned in the section "Summary of Features" must be corrected to 4 instead of 2.

## 4.10 [ADC\_TC.H048] EVADC sampling time setting below 300 ns lead to $V_{DDK}$ signal conversion inaccuracy

### Description

EVADC: Timing exception for measuring on-chip supervision signals.

### Scope

$V_{DDK}$  monitoring

### Effects

$V_{DDK}$  measurement results can become unreliable when the sampling time is too short.

### Recommendation

A minimum sampling time of 300 ns for the whole supply range should be considered.

## 4 Application hints

### 4.11 [ASCLIN\_TC.H001] Bit field FRAMECON.IDLE in LIN slave tasks

#### Description

For LIN performing slave tasks, bit-field FRAMECON.IDLE has to be set to 000<sub>B</sub> (default after reset), i.e. no pause will be inserted between transmission of bytes.

If FRAMECON.IDLE > 000<sub>B</sub>, the inter-byte spacing of the ASCLIN module is not working properly in all cases in LIN slave tasks (no bit errors are detected by the ASCLIN module within the inter-byte spacing).

### 4.12 [ASCLIN\_TC.H006] Sample point position when using three samples per bit

#### Description

As documented in the description of field BITCON.SAMPLEPOINT, "... if three sample points at position 7, 8, 9 are required, this bit-field would contain 9".

In general, if three samples per bit are selected (BITCON.SM = 1<sub>B</sub>), field BITCON.SAMPLEPOINT defines the position of the last sample point.

#### Documentation update

The text related to three sample points in figure "ASCLIN Bit Structure" in the ASCLIN chapter of the user manual should be updated as follows:

- 16x Oversampling, 3 sample points, relevant sample position 7, 8, 9 (BITCON.OVERSAMPLING = 16, BITCON.SM = 1, BITCON.SAMPLEPOINT = 9)
  - instead of "16x Oversampling, 3 sample points, relevant sample position 8"
- 8x Oversampling, 3 sample points, relevant sample position 3, 4, 5 (BITCON.OVERSAMPLING = 8, BITCON.SM = 1, BITCON.SAMPLEPOINT = 5)
  - instead of "8x Oversampling, 3 sample points, relevant sample position 4"

### 4.13 [ASCLIN\_TC.H007] Handling TxFIFO and Rx FIFO interrupts in single move mode

#### Present description for Tx FIFO single move mode

As described in section "Single Move Mode" of chapter "TxFIFO interrupt generation" in the user manual, the purpose of the Single Move Mode is to keep the Tx FIFO as full as possible, refilling the Tx FIFO by writing to it as soon as there is a free element. The single move mode supports primarily a DMA operation using single move per Tx FIFO interrupt.

See also the note at the end of this section in the user manual.

"In Single Move Mode multiple software writes or block DMA moves would lead to multiple interrupts and (false) transaction lost events. Therefore they should be avoided - only single moves should be used."

To complement the above description, the following two sentences shall be added to the section before the reference to figure "Interrupt generation in the single move mode" in the user manual.

#### Documentation update for Tx FIFO single move mode

If Tx FIFO can handle new data, it generates an interrupt but expects just one data of the defined frame width. The DMA or the user should not write multiple data at once to avoid unexpected behavior.

#### Present description for Rx FIFO single move mode

As described in section "Single Move Mode" of chapter "Rx FIFO interrupt generation" in the user manual, the purpose of the Single Move Mode is to keep the Rx FIFO as empty as possible, by fetching the received elements

## 4 Application hints

one by one as soon as possible. The single move mode supports primarily a DMA operation using single move per RxFIFO interrupt.

See also the note at the end of this section in the user manual.

"In Single Move Mode multiple software reads or block DMA moves lead to multiple interrupts and (false) transaction lost events. Therefore they should be avoided - only single moves should be used."

To complement the above description, the following two sentences shall be added to the section before the reference to figure "RXFIFO - Interrupt Triggering in the Single Move Mode" in the user manual.

### Documentation update for RxFIFO single move mode

If RxFIFO can handle new data, it generates an interrupt but fetches just one data of the defined frame width. The DMA or the user should not read multiple data at once to avoid unexpected behavior.

## 4.14 [ASCLIN\_TC.H008] SPI master timing – Additional information to Data Sheet characteristics

### Description

The following note shall be added to chapter "ASCLIN SPI Master Timing" in the Data Sheet:

**Note:** *The specified timings describe the pad capabilities for the respective driver strength configuration. For the maximum achievable baud rate in a given application, the MRST input timings need to be considered in particular.*

### Background information

Chapter "ASCLIN SPI Master Timing" in the Data Sheet contains separate tables for different output driver configurations. As can be seen from these tables, the master output timings directly depend on the selected driver strength. The corresponding parameters are marked as controller characteristics with symbol "CC".

The setup and hold timings for input data received from the slave are marked as system requirements with symbol "SR". They must be provided by the system in which the device is designed in.

In a given application, the maximum rate at which data can be received from a slave on the master receive input MRST may be limited by the required setup time  $t_{52}$  (MRST setup to ASCLKO latching edge). As data is shifted by the slave on one edge of ASCLKO and latched by the master on the opposite edge, one phase of ASCLKO must always be greater than the minimum required MRST setup time (assuming the sampling point is in the middle). This means the ASCLKO period  $t_{50}$  must be  $> 2 \times t_{52}$ .

## 4.15 [ASCLIN\_TC.H012] Unexpected collision detection flag raised when soft suspend request is raised and ACK has not arrived

### Description

For ASCLIN SPI half-duplex or LIN communication, in debug mode, an unexpected collision detection flag is detected when a soft suspend request between a transmission trigger and start of transmission is raised. A subsequent interrupt could be triggered if the collision detection interrupt is enabled.

### Recommendation

In SPI half duplex or LIN mode, when a transmission is triggered, and if soft suspend is requested before the transmission has started (first bit sent out), the CE flag might be set (if collision detection is enabled by setting bit-field FRAMECON.CEN). This flag must be ignored in this particular use-case.

## 4 Application hints

### 4.16 [BROM\_TC.H009] Re-enabling lockstep via BMHD

#### Description

For all CPUs with lockstep option, the lockstep functionality is controlled by Boot Mode Headers (BMHD) loaded during boot upon a reset trigger.

If lockstep is disabled for a CPUx with lockstep functionality, re-enabling (for example via a different BMHD) is not reliably possible if warm PORST, System or Application reset is executed.

#### Recommendation

Use cold PORST if lockstep is disabled and shall be re-enabled upon the reset trigger.

### 4.17 [BROM\_TC.H014] SSW behavior in case of wrong state or uncorrectable error in UCBs - Documentation Update

#### Description

The boot sequence terminates and the device is put into error state (endless loop) in the following cases:

- **Wrong state** - i.e. different from CONFIRMED or UNLOCKED (in case an UCB has ORIGINAL and COPY: wrong state of the both) – for the following UCBs:
  - UCB\_BMHDx, UCB\_SWAP, UCB\_SSW, UCB\_USER, UCB\_PFLASH, UCB\_DFLASH, UCB\_DBG, UCB\_HSM, UCB\_HSMCOTP0...1, UCB\_HSMCFG, UCB\_ECPRIO, UCB\_OTP0...7, UCB\_REDSEC, UCB\_TEST, UCB\_RETEST
- **Uncorrectable ECC error** within the used locations when state valid (CONFIRMED or UNLOCKED) – for the following UCBs:
  - UCB\_SSW, UCB\_PFLASH, UCB\_DFLASH, UCB\_DBG, UCB\_HSM, UCB\_HSMCOTP0...1, UCB\_ECPRIO, UCB\_OTP0...7, UCB\_REDSEC, UCB\_RETEST
- For UCB\_SWAP ORIGINAL/COPY – according to the descriptions in user manual

#### Recommendation

Instructions to be followed for UCB-reprogramming (in order to avoid unexpected boot termination):

- Always verify the changed contents before confirming the UCB state
- Strictly follow the sequence in section “UCB Confirmation” in the “Non Volatile Memory (NVM)” chapter of the user manual

### 4.18 [BROM\_TC.H020] Processing in case no valid BMHD found

#### Description

The section "Processing in case no valid BMHD found" in the Firmware chapter of the TC3xx user manual states in step 4 as follows:

- 4. instal the address with offset 0x0020 in logical sector S40 in PFLASH0 (I.e. 0xA000A020) as user code start address into BOOT\_ADDR

The absolute address specified in the above sentence is incorrect. It must be 0xA00A0020 instead of 0xA000A020.

#### Documentation update

The absolute address specified in step 4 in the section “Processing in case no valid BMHD found” in the Firmware chapter of the TC3xx user manual shall be corrected as follows:

- 4. Install the address with offset 0x0020 in logical sector S40 in PFLASH0 (that is 0xA00A0020) as user code start address into BOOT\_ADDR

**4 Application hints**

**4.19 [CCU6\_TC.H001] CCU6 module clock source information - Documentation Update**

**Description**

In the CCU6 module chapter of the AURIX™ TC3xx User’s manual, the CCU6 module clock source information is missing.

**Documentation update**

The CCU6 module is clocked with the SPB clock, so  $f_{CC6} = f_{SPB}$ .

See also figures “Clocking System example” in the Clocking System chapter of the TC3xx User’s manual, where the CCU6 module is connected to  $f_{SPB}$ .

**4.20 [CCU\_TC.H012] Configuration of the Oscillator- Documentation Update**

**Description**

As described in chapter „Configuration of the Oscillator” in the CCU chapter of the User’s Manual, configuration of the oscillator is always required before an external crystal / ceramic resonator can be used as clock source.

Depending on the supply voltage ramp-up characteristics the behavior described in the following note may be observed:

**Note:** *If VEXT is present then the oscillator could start oscillating (crystal/resonator connected). As soon as Cold PORST of AURIX™ is released, the oscillator is set to External Input Mode and the oscillation decays. This characteristic behavior has no impact on the oscillator start-up as initiated by software.*

**4.21 [CLC\_TC.H001] Description alignment for bits DISR, DISS, EDIS in register CLC - Documentation Update**

**Description**

For the description of bits DISR, DISS, and EDIS (if available) in register CLC (and CLC1 for I2C), different styles are used in the current version of the TC3xx user manual.

For the following modules, the function of these bits depending on their status ( $0_B$  or  $1_B$ ) is not explicitly described:

- ASCLIN, CIF, E-RAY, FCE, GETH, GTM, HSPDM, HSSL (incl. HSCT), I2C, MCMCAN, MSC, PSI5, PSI5-S, QSPI, SDMMC, SENT, STM

For these modules, the missing parts of the bit description can be taken from the following general description:

**Table 16 General description of bits DISR, DISS, EDIS in register CLC**

Field	Bits	Type	Description
<b>DISR</b>	0	rw	<b>Module Disable Request Bit</b> Used for enable/disable control of the module $0_B$ No disable requested $1_B$ Disable requested
<b>DISS</b> (table continues...)	1	rh	<b>Module Disable Status Bit</b>

**4 Application hints**

**Table 16 (continued) General description of bits DISR, DISS, EDIS in register CLC**

Field	Bits	Type	Description
			Bit indicates the current status of the module 0 <sub>B</sub> Module is enabled 1 <sub>B</sub> Module is disabled
<b>EDIS</b>	3	rw	<b>Sleep Mode Enable Control</b> Used for module Sleep Mode control 0 <sub>B</sub> Sleep Mode request is regarded. Module is enabled to go into Sleep Mode on a request. 1 <sub>B</sub> Sleep Mode request is disregarded: Sleep Mode cannot be entered on a request.

**Notes:**

1. Bit EDIS is not implemented for the following of the modules listed above: CIF, GETH, I2C, SDMMC
2. In the FCE module, the bit at position of EDIS is of type 'rw', but without function and not shown in the user manual
3. In the EDSADC, GTM, STM modules bit DISS is of type 'rh', but shown as 'r' in the user manual

**4.22 [CPU\_TC.H022] Store buffering and the effect of bit SMACON.IODT**

**Description**

To increase performance, the TC1.3.1, TC1.6.\* , and TC1.8 CPUs implement store buffering. In normal operation, with bit SMACON.IODT=0<sub>B</sub> (default after reset), non-dependent loads may bypass store operations. To improve the performance of load operations, the CPU will snoop the content of the store buffer for a matching address. If a match is found, the load data is retrieved from the store buffer before the store is committed to memory. For further details, see the chapter "Store Buffers" in the TC3xx user manual.

In this context, the following statements included in the CPU chapter of TC3xx user manual may be misleading:

- In the chapter "Store Buffers": Store buffer operation may be disabled by setting the SMACON.IODT bit
- In the description of register SMACON in the chapter "Memory Integrity Registers" for IODT=1<sub>B</sub>: In-order operation, loads always flush preceding stores, processor store buffer disabled

**Recommendation**

Effectively, setting SMACON.IODT=1<sub>B</sub> results in memory operations to be performed in program order, where loads always flush preceding stores.

As described in the user manual, setting SMACON.IODT=1<sub>B</sub> should not be done in normal execution, but should only be performed by test routines at start-up or shut-down, as it will severely limit performance.

If there is a requirement that data is written to local memory prior to execution of a subsequent instruction then a DSYNC instruction may be used to flush the store buffers.

**4.23 [CPU\_TC.H023] CPU\_SYSCON register safety protection description clarification**

**Description**

The usage of SAFETY\_ENDINIT protection of the CPU system control register (CPU\_SYSCON) is configurable by the application software. From system reset, as well as after any kernel reset, the register is not subject to

## 4 Application hints

SAFETY\_ENDINIT protection. This information is not conveyed in the register overview table which states that the register is always subject to SAFETY\_ENDINIT protection. Clearing the compatibility control register safety protection field (COMPAT.SP=0<sub>B</sub>) enables the SAFETY\_ENDINIT protection of CPU\_SYSCON[31:1].

**Note:** CPU\_SYSCON[0] is never subject to SAFETY\_ENDINIT protection.

### Scope

This problem is limited to the CPU\_SYSCON register.

### Effects

After kernel reset, the CPU\_SYSCON register is not subject to the SAFETY\_ENDINIT protection; which is compatible with the earlier devices.

### Workaround

To ensure that CPU\_SYSCON[31:1] is subject to SAFETY\_ENDINIT protection, the application software must clear COMPAT.SP.

## 4.24 [CPU\_TC.H024] Usage of atomic instructions SWAPMSK.W and LDMST to access registers with bit-fields that can also be updated by hardware (rwh)

### Description

Atomic instructions like SWAPMSK.W and LDMST in the AURIX™ microcontroller provide atomicity and bit-wise operations to the targeted memory locations or peripheral registers. They are also referred to as Read-Modify-Write (RMW) instructions. The bit-manipulation functionality allows software to update individual bits in a register without affecting other, selecting the bits to be read and written through a mask in the instruction.

**Note:** Please refer to the TriCore™ Architecture Manual for further information about these instructions and their formats.

**Note:** Additionally, please refer to the dedicated note in the User Manual indicating the usage of instructions SWAPMSK and LDMST for specific registers containing the "rwh" field, indicating that this field can be accessed and modified by software and hardware.

As a general remark, consider that when implementing time-critical or safety-critical systems, operations involving registers that can be updated by both hardware and software, poses a risk of conflicts between hardware and software updates. In particular when using atomic instructions like SWAPMSK.W or LDMST to access registers with "rwh" fields. If the bit-field is accessed simultaneously by the hardware and the software, the hardware update events may be lost, and the bit-field updated by the hardware may be overwritten by the software write operation performed via the atomic instruction.

### Recommendation

LDMST or SWAPMSK.W instructions should be used only with bit mask enabled for registers containing rwh bits. It is essential to consider that an atomic instruction can take multiple cycles to complete, during which any hardware event can occur and be lost. For example, a SWAPMSK.W operation, used for reading (READ) a "rwh" flag from a register (for example, GTM\_IRQ\_NOTIFY) and then for writing (WRITE) to clear flags of the register, takes around 6 clock cycles. It is possible that when the read operation is performed, no flag is set by hardware. However, within the 6-clock-cycle time interval, a hardware event can set a flag after the READ but before the WRITE of the atomic operation. At that point, the event flag was not captured by the READ operation but could be cleared by the WRITE operation. This is especially true if the bit-field is used for both status reporting and for clearing itself (for example, read value 1 indicates the hardware event was raised, while writing 1 to the same

## 4 Application hints

bit-field clears it). To generalize, if software and hardware update bit(s) in the same cycle, the software wins and the hardware event is lost.

### 4.25 [CPU\_TC.H025] Avoiding unbounded delays in store buffer residency

#### Description

In a multiprocessor system, sharing variables between different cores is the basis for implementing synchronization for programs running concurrently on multiple processors.

In a multiprocessor system, the following mechanism may be used to pass information between different cores:

- CPU\_A
  - Writes to a shared variable V
- CPU\_B, or other masters
  - Waits for a change in the value of the shared variable V

Usually, a wait by CPU\_B may be implemented as a polling loop.

System performance may be adversely impacted if the visibility by CPU\_B of the write to the shared variable is delayed. In an extreme case, an unbounded delay may obstruct system progress.

#### Scenario 1: Store operation with possibly unbounded delay scenario 1

Consider a scenario where:

- CPU\_A
  - Writes to a shared variable V and then
  - Performs a task that involves repeated loads from the same memory module
- CPU\_B, or other masters
  - Waits for a change in the value of the shared variable V

When CPU\_A executes a store instruction targeting V, the information is recorded in a Store Buffer (see Store Buffers paragraph in the CPU chapter of AURIX™ TC2xx, TC3xx and TC4xx User Manuals). CPU\_A will then attempt to complete the store information by starting the write access to the target memory where V is allocated.

This write access is subject to arbitration with other concurrent accesses to the same target memory. If other accesses such as loads from the same CPU\_A, or accesses from other masters like additional CPUs, DMA, etc., have higher priority, the write access may be delayed. This delays CPU\_B's visibility of the update to V. In an extreme case, if the competing accesses are repeated indefinitely, the visibility will be delayed indefinitely, potentially leading to a significant disruption in CPU\_B's operation.

To prevent such delay, a DSYNC<sup>3)</sup> instruction should be inserted in the CPU\_A code following the write operation to V, and preceding the execution of repeated loads by CPU\_A. This ensures timely visibility of the update to V by other CPUs.

#### Example code: Store operation with possibly unbounded delay

The following code snippet illustrates a scenario where a store operation may be delayed due to access arbitration with other loads. In this example, CPU\_A executes a store instruction (ST.W) to variable V, followed by a generic instruction (INSTR\_X) and a loop (L1) of load instructions (LD.W) that access variables R, S, and T.

<sup>3</sup> For devices with the TC1.8 core (excepting the TC49x A steps), the LSYNC instruction can be used as an alternative to DSYNC to ensure correctness. LSYNC is sufficient to prevent the delay to V's visibility and has a lower execution cost.

**4 Application hints**

These load instructions share access arbitration with the store operation to variable V, potentially causing a delay in the completion of the store operation. The loop L1 terminates when a specific value of variable R is read. The variable R is expected to be updated by another CPU (or DMA) in the system.

```

CPU_A_code:
    ST.W [V], Dv
    INSTR_X
L1:
    LD.W Dr, [R] 4)
    LD.W Ds, [S] 4)
    LD.W Dt, [T] 4)
    JNZ Dr, L1
    
```

The following table illustrates the execution pipeline of CPU\_A, highlighting the store operation to variable V and the subsequent loop of load instructions that access variables R, S, and T. The table also shows the contents of the store buffer, access arbitration, and memory access.

**Table 17 Example execution: Store operation with unbounded delay**

<b>CPU_A: Start of execution pipeline</b>	<b>CPU_A: End of execution pipeline<sup>1)</sup></b>	<b>CPU_A: Store Buffer content</b>	<b>Access arbitration</b>	<b>Memory access</b>
ST.W, store to variable V	-	-	-	-
INSTR_X	-	-	-	-
LD.W, load from variable R	ST.W [V]	Store to V	Store to V Load from R (higher priority)	-
LD.W, load from variable S	INSTR_X	Store to V	Store to V Load from S (higher priority)	Load from R
LD.W, load from variable T JNZ	LD.W [R]	Store to V	Store to V Load from T (higher priority)	Load from S
LD.W, load from variable R	LD.W [S]	Store to V	Store to V Load from R (higher priority)	Load from T
LD.W, load from variable S	LD.W [T] Conditional jump	Store to V	Store to V Load from S (higher priority)	Load from R
LD.W, load from variable T JNZ	LD.W [R]	Store to V	Store to V Load from T (higher priority)	Load from S

**(table continues...)**

<sup>4</sup> Assumption: access to the variables R, S, and T share access arbitration with the access to the variable V

**4 Application hints**

**Table 17 (continued) Example execution: Store operation with unbounded delay**

CPU_A: Start of execution pipeline	CPU_A: End of execution pipeline <sup>1)</sup>	CPU_A: Store Buffer content	Access arbitration	Memory access
...	...	Store to V	Store to V Load from ... (higher priority)	Load from T
...	...	...	...	...

1) The number of cycles in the CPU execution pipeline is implementation dependent, see the core User Manual for details.

**Key points:**

- The store operation to variable V is delayed due to access arbitration with the load instructions
- The load instructions from variables R, S, and T have higher priority and win arbitration, causing the store operation to be delayed
- The store buffer content remains unchanged until the store operation is completed
- The memory access column shows the actual memory access that occurs, which is delayed due to access arbitration

It is noteworthy that the "Store to V" operation is not reflected in the Memory access column. This indicates that the store operation to variable V has not been completed, and the new value has not been written to memory.

While the iterations of loop L1 continue, the value in memory of V will not be updated by CPU\_A. Consequently, if another CPU (CPU\_B) were to access variable V, it would read the previous value of variable V. If CPU\_B awaits a change in V's value, it may experience an unbounded delay due to the delayed completion of the store operation.

**Scenario 1: Resolution with DSYNC instruction added**

In this example, the DSYNC instruction is added after the store operation to variable V. The DSYNC instruction ensures the visibility of the update to variable V as the store buffer is emptied before commencing the load from variable R.

**Example code: with DSYNC instruction added**

The following code snippet illustrates the use of the DSYNC instruction to ensure that a store operation is visible before proceeding with subsequent memory instructions.

```

CPU_A code:
  ST.W [V], Dv
  DSYNC
  INSTR_X
L1:
  LD.W Dr, [R]
  LD.W Ds, [S]
  LD.W Dt, [T]
  JNZ Dr, L1
    
```

The following table illustrates the execution pipeline of CPU\_A, highlighting the use of the DSYNC instruction to ensure the visibility of the store operation to variable V, before the subsequent loop of load instructions that access variables R, S, and T.

**4 Application hints**

**Table 18 Example execution: with DSYNC instruction added**

<b>CPU_A: Start of execution pipeline</b>	<b>CPU_A: End of execution pipeline<sup>1)</sup></b>	<b>CPU_A: Store Buffer content</b>	<b>Access arbitration</b>	<b>Memory access</b>
ST.W, store to variable V	-	-	-	-
DSYNC	-	-	-	-
(DSYNC) – waits for store buffer empty	-	-	-	-
(DSYNC) – waits for store buffer empty	ST.W [V]	Store to V	Store to V	-
INSTR_X	-	-	-	Store to V
LD.W load from variable R	DSYNC	-	Load from R	-
LD.W load from variable S	INSTR_X	-	Load from S	Load from R
LD.W load from variable T JNZ	LD.W [R]	-	Load from T	Load from S
...	...	-	Load from ...	Load from T
...	...	...	...	...

1) The number of cycles in the CPU execution pipeline is implementation dependent, see the core User Manual for details.

**Key points:**

- The store operation to variable V is followed by the DSYNC instruction, which ensures that the store operation is no longer in the store buffer
- The DSYNC instruction waits for the store buffer to be drained
- The load loop execution is delayed until the store operation to variable V is commenced, ensuring that CPU\_B can observe the updated value of variable V in a timely manner

It is noteworthy that the "Store to V" operation is now reflected in the Memory access column. This indicates that the store operation to variable V has been commenced, and will be written to memory. Consequently, this ensures that CPU\_B can observe the updated value of the variable V in a timely manner.

**Scenario 2: Store operation with possibly unbounded delay (minimum two CPUs)**

Consider a scenario where:

- CPU\_A
  - Writes to a shared variable V and then
  - Performs a task that involves repeated loads from the same memory module
- CPU\_B
  - Polls the shared variable V in a loop using a Read-Modify-Write (RMW) instruction (for example CMPSWAP.W), waiting for a change in V value

**Example code: Store operation with possibly unbounded delay**

The following code snippets illustrate the interaction between CPU\_A and CPU\_B, highlighting the potential issues that can arise when accessing shared variables. The loop L1 terminates when a specific value of variable

**4 Application hints**

R is read. The variable R is expected to be updated by CPU\_B. The loop L2 terminates when a specific value of variable V; expected to be updated by CPU\_A, is observed.

```

CPU_A code:
  ST.W [V], Dv
  INSTR_X
L1:
  LD.W Dr, [R] 5)
  INSTR_Y 6)
  INSTR_Z 6)
  JNZ Dr, L1

CPU_B code:
L2:
  [...]
  CMPSWAP.W [V], Ex
  JNE Dx, Dz, L2
    
```

In this example, CPU\_A executes a store instruction to variable V, followed by a loop (L1) containing a load instruction from variable R. The load instruction from variable R shares access arbitration with the accesses to variable V from both CPU\_A and CPU\_B. In the same loop (L1), CPU\_A also executes two other instructions, INSTR\_Y and INSTR\_Z, which are not store instructions.

Meanwhile, CPU\_B executes a loop (L2) containing a compare-and-swap atomic instruction (CMPSWAP.W) on variable V and registers Ex = [Dx, Dx+1], which checks if the content of V is equal to the content of register Dx+1 and if they are equal then swaps the contents of V with the register Dx. Register Dx is unconditionally updated with the contents of V. At the tail of the loop, if the value of register Dx and the value in register Dz are not equal CPU\_B jumps back to label L2. If CPU\_B is waiting for the store from CPU\_A to update V to the value in Dz, it may experience an unbounded delay due to the delayed completion of the store operation.

The following table illustrates the execution pipeline of CPU\_A, highlighting the store operation to variable V and the subsequent loop of the load instruction that accesses variable R. The table also shows the contents of the store buffer, CPU\_B access to variable V, access arbitration, and memory access.

**Table 19 Example execution: Store operation with unbounded delay**

CPU_A, start of execution pipeline	CPU_A, end of execution pipeline <sup>1)</sup>	CPU A, Store Buffer contents	CPU_B, access request	Access arbitration	Memory access
ST.W, store to variable V	-	-	-	-	-
INSTR_X	-	-	-	-	-
LD.W load from variable R	ST.W [V]	Store to V	-	Store to V Load from R (higher priority)	-
INSTR_Y	INSTR_X	Store to V	RMW, read from V (CMPSWAP.W)	Store to V RMW, read from V (higher priority)	CPU_A, Load from R

**(table continues...)**

<sup>5</sup> Assumption: access to the variable R shares access arbitration with the access to the variable V

<sup>6</sup> Assumption: neither INSTR\_Y nor INSTR\_Z is a store instruction

**4 Application hints**

**Table 19 (continued) Example execution: Store operation with unbounded delay**

CPU_A, start of execution pipeline	CPU_A, end of execution pipeline <sup>1)</sup>	CPU A, Store Buffer contents	CPU_B, access request	Access arbitration	Memory access
INSTR_Z	LD.W [R]	Store to V	-	Store to V Locked by RMW <sup>2)</sup>	CPU_B, RMW read from V (previous value of V)
JNZ	INSTR_Y	Store to V	RMW, write to V (CMPSWAP.W)	Store to V Locked by RMW <sup>2)</sup> RMW, write to V	-
LD.W load from variable R	INSTR_Z	Store to V	-	Store to V Load from R (higher priority)	CPU_B, RMW, write to V
INSTR_Y	Conditional jump	Store to V	RMW, read from V (CMPSWAP.W)	Store to V RMW, read from V (higher priority)	CPU_A, Load from R
INSTR_Z	LD.W [R]	Store to V	-	Store to V Locked by RMW <sup>2)</sup>	CPU_B, RMW read from V (previous value of V)
JNZ	INSTR_Y	Store to V	RMW, write to V (CMPSWAP.W)	Store to V Locked by RMW <sup>2)</sup> RMW, write to V	-
...	...	...	...	...	...

- 1) The number of cycles in the CPU execution pipeline is implementation dependent, see the core User Manual for details.  
 2) RMW operation holds exclusive access to the memory for multiple cycles.

**Key Points:**

- In this example, CPU\_B may wait forever for variable V to be updated by CPU\_A because CPU\_A's store to variable V may be continuously blocked by the combination of the CMPSWAP.W memory access from CPU\_B and the load access from CPU\_A itself
- This may lead to an unbounded delay in the progress of CPU\_B which could lead to system unavailability

**Scenario 2: Resolution with DSYNC instruction added (minimum two CPUs)**

In this example, the DSYNC<sup>7)</sup> instruction is added after the store operation to variable V. The DSYNC instruction ensure the visibility of the update to the variable V as the store buffer is emptied before commencing the load from variable R.

The CPU\_B code remains unchanged.

<sup>7)</sup> For devices with the TC1.8 core (excepting the TC49x A steps), the LSYNC instruction can be used as an alternative to DSYNC to ensure correctness. LSYNC is sufficient to prevent the delay to V's visibility and has a lower execution cost.

**4 Application hints**

**Example code: with DSYNC instruction added**

```

CPU_A_code:
    ST.W [V], Dv
    DSYNC
    INSTR_X
L1:
    LD.W Dr, [R]8)
    INSTR_Y
    INSTR_Z
    JNZ Dr, L1

CPU_B_code:
    [no change]
    
```

The following table shows the start and end of the execution pipeline for each instruction of CPU\_A, as well as the contents of the store buffer, access requests from CPU\_B to variable V, access arbitration, and memory access, highlighting the use of the DSYNC instruction to ensure that a store operation is completed before proceeding with subsequent instructions.

**Table 20 Example execution: with DSYNC instruction added**

<b>CPU_A, start of execution pipeline</b>	<b>CPU_A, end of execution pipeline <sup>1)</sup></b>	<b>CPU A, Store Buffer contents</b>	<b>CPU_B, access request</b>	<b>Access arbitration</b>	<b>Memory access</b>
ST.W, store to variable V	-	-	-	-	-
DSYNC	-	-	RMW, read from V (CMPSWAP.W)	RMW, read from V	-
(DSYNC) – waits for store buffer empty	ST.W [V]	Store to V	-	Store to V Locked by RMW <sup>2)</sup>	CPU_B, RMW read from V (previous value of V)
(DSYNC) – waits for store buffer empty	-	Store to V	RMW, write to V (CMPSWAP.W)	Store to V Locked by RMW <sup>2)</sup> RMW, write to V	-
(DSYNC) – waits for store buffer empty	-	Store to V	-	Store to V	CPU_B, RMW, write to V
INSTR_X	-	-	RMW, read from V (CMPSWAP.W)	RMW, read from V	CPU_A, Store to V
LD.W load from variable R	DSYNC	-	-	Load from R Locked by RMW <sup>2)</sup>	CPU_B, RMW read from V (updated value of V)

**(table continues...)**

<sup>8</sup> Assumption: access to the variable R share access arbitration with the access to the variable V

**4 Application hints**

**Table 20 (continued) Example execution: with DSYNC instruction added**

CPU_A, start of execution pipeline	CPU_A, end of execution pipeline <sup>1)</sup>	CPU A, Store Buffer contents	CPU_B, access request	Access arbitration	Memory access
(LD.W) – waits for access	INSTR_X	-	-	Load from R Locked by RMW <sup>2)</sup> RMW, write to V	-
(LD.W) – waits for access	-	-	-	Load from R	CPU_B, RMW, write to V
INSTR_Y	-	-	-	-	CPU_A, Load from R
INSTR_Z	LD.W [R]	-	-	-	-
JNZ	INSTR_Y	-	-	-	-

- 1) The number of cycles in the CPU execution pipeline is implementation dependent, see the core User Manual for details.  
 2) RMW operation holds exclusive access to the memory for multiple cycles.

In this example, the visibility of the store to the variable V is enforced by using a DSYNC instruction before CPU\_A proceeds with the load loop instructions.

**Key Points:**

- The store operation to variable V is followed by the DSYNC instruction, which ensures that the store operation is no longer in the store buffer
- The DSYNC instruction waits for the store buffer to be drained
- The load loop execution is delayed until the store operation to variable V is commenced, ensuring that CPU\_B can observe the updated value of variable V in a timely manner

**Conditions for unbounded delay for store completion**

The completion of a store may have an unbounded delay if all of the following conditions are met.

**Condition set:**

- The store is followed by sequence of load instructions such as a load or loads executed in a loop **AND**
- There is neither a DSYNC instruction nor an atomic memory operation executed after the store operation **AND**
- The each load operation competes in internal CPU arbitration against the store operation **AND**
- There are no store<sup>9)</sup> operations executed between those loads

**Note:** *This applies not only to the store (ST\*) instruction but also to any instruction that involves a store operation. In particular, as described in the TriCore™ Architecture Manual (Context Switching for Function Calls), a CALL instruction stores registers to a CSA, therefore it qualifies as a “store operation” with respect to these conditions.*

The following code examples demonstrate the conditions for delayed store completion. These examples are designed to aid in determining whether a particular code sequence is vulnerable to potential excessive delay in store completion. Examples are given of both code sequences with potential delay (requiring a DSYNC addition, or other modification for timely completion), and of code sequences where no excessive delay occurs (no code modification is required).

**Example 1: code with potential delay of store completion**

<sup>9</sup> Store operations refers not only to ST\* instructions, but also to any instruction that involves a store operation. In particular, as described in the TriCore™ Architecture Manual (Context Switching for Function Calls), a CALL instruction stores registers to a CSA, therefore it qualifies as a “store operation” with respect to these conditions.

#### 4 Application hints

The following code example illustrates a scenario where the completion of a store operation may be delayed due to the conditions outlined in the main section.

Code example:

```
ST.W [V], Dv
MOVH Ax, ...
LEA Ax, Ax, ...
L1:
LD.W Dr, [R]
LD.W Ds, [S]
LD.W Dt, [T]
JNZ Dr, L1
```

The location of the variables in this case is:

- V, R, S, and T are in the CPU's own DSPR **OR**
- V, R, S, and T are in the CPU's own DLMU **OR**
- V, R, S, and T are outside the CPU's own DSPR/DLMU

In this example, a store operation is executed to variable V, followed by a sequence of load instructions that access variables R, S, and T. The load instructions are executed in a loop, and the load operations compete in internal CPU arbitration against the store operation.

All the conditions listed in the [condition set](#) are true in this example. Due to these conditions, the completion of the store operation to variable V may be delayed.

#### **Example 1 recommended solution: Using DSYNC instruction**

The following code example illustrates the recommended solution to ensure timely completion of store operations, particularly in scenarios where the conditions outlined in the [condition set](#) are true.

Code example:

```
ST.W [V], Dv
DSYNC
MOVH Ax, ...
(alternative DSYNC location)
LEA Ax, Ax, ...
(alternative DSYNC location)
L1:
LD.W Dr, [R]
LD.W Ds, [S]
LD.W Dt, [T]
JNZ Dr, L1
```

The DSYNC instruction, inserted after the store operation, ensures the store buffer is empty of all the preceding stores. The store to V is completed in a timely manner. The DSYNC instruction can be inserted at alternative locations, such as after the MOVH or LEA instructions, to ensure that the store buffer is empty before proceeding with subsequent instructions.

#### **Example 2: code with no unbounded delay**

The following code example illustrates a scenario where the completion of a store operation is not affected by the conditions outlined in the [condition set](#); to be clear there is no unbounded delay.

#### 4 Application hints

Code example:

```
ST.W [V], Dv
MOVH Ax, ...
LEA Ax, Ax, ...
L1:
LD.W Dr, [R]
LD.W Ds, [S]
LD.W Dt, [T]
JNZ Dr, L1
```

The location of the variables in this case is:

- R, S, and T are in the CPU's own DSPR, and
- V is in another CPU's DSPR

In this example, a store operation is executed to variable V, followed by a sequence of load instructions that access variables R, S, and T. However, the loads from R, S, and T do not compete in arbitration against the store to V, as R, S, and T are located in different DSPRs. As a result, the completion of the store operation to variable V is not affected by the loads from R, S, and T. The store operation is completed in a timely manner.

#### **Example 3: code with no unbounded delay**

The following code example illustrates a scenario where a store operation is executed to variable V, followed by a sequence of load instructions that access variables R, S, and T. However, the sequence of repeated operations also contains a store instruction to variable U.

Code example:

```
ST.W [V], Dv
MOVH Ax, ...
LEA Ax, Ax, ...
L1:
LD.W Dr, [R]
LD.W Ds, [S]
LD.W Dt, [T]
ST.B [U], Du
JNZ Dr, L1
```

In this case example the sequence of repeated operations contains a store instruction. The store buffer quickly fills up when stores are executed in a loop. Once the store buffer is full, the CPU waits for at least one store to exit the store buffer before executing the next store instruction. Therefore, in this case there is no unbounded delay in the completion of stores.

#### **Example 4: code with no unbounded delay**

The following code example illustrates a scenario where a store operation is executed to variable V, followed by a sequence of load instructions that access variables R, S, and T. However, the sequence of repeated operations also contains a function call.

## 4 Application hints

Code example:

```
ST.W [V], Dv
MOVH Ax, ...
LEA Ax, Ax, ...
L1:
LD.W Dr, [R]
LD.W Ds, [S]
LD.W Dt, [T]
CALL foo
JNZ Dr, L1
```

In this example the sequence of repeated operations contains a function call (which requires a CSA save operation). This will cause the store buffer to fill up and as in the previous example, the store operation to V will exit the store buffer in a timely manner.

### Example 5: code with unbounded delay in store completion

The following code example illustrates another scenario where the completion of a store operation may be delayed due to the conditions outlined in the [condition set](#).

Code example:

```
ST.W [V], Dv
MOVH Ax, ...
LEA Ax, Ax, ...
L1:
LD.W Dr, [R]
ADD Dx, 1
JNZ Dr, L1
```

The location of variables in this case is:

- V and R located in CPU's own DSPR **OR**
- V and R located in CPU's own DLMU

**Note:** *It is assumed that R could be updated by an event (originating from an external source to the CPU\_A itself).*

In this example, a store operation is executed to variable V, followed by a sequence of load instructions that access variable R. The sequence of repeated operations is executed in a loop, and the the load operations compete in internal CPU arbitration against the store operation. All the conditions listed in 1.4 are true, consequently the completion of the store to V may be delayed. In this case an unbounded delay may occur if there are concurrent accesses from other CPUs (or DMAs) accessing the same memory.

In general, as the exact pattern of behavior of other CPUs, DMA, etc. may be difficult to ensure, it is recommended to add a DSYNC instruction to such code sequences. This will ensure that the store operation exits the store buffer before proceeding with subsequent instructions, eliminating the delay to visibility of the updated value.

### Recommendation

In summary, the Store Buffer mechanism is an important feature of TriCore™ and its existence is mostly transparent to uni-processor code. However, when multiprocessor synchronization is to be performed, it requires careful consideration and management to prevent unexpected behavior. As described above, by using the DSYNC instruction to flush the Store Buffer and ensure the completion of store operations, developers can guarantee that shared resources are updated correctly and in a timely manner, ensuring the correct behavior of the system.

## 4 Application hints

### 4.26 [CPU\_TC.H027] Performance impact depending on the enabling of the Core MPU protection

#### Description

In TriCore™ TC1.6.2 the protection registers set (PRS) value associated with the function return address is stored in the context save area (CSA). Therefore, when a return (RET) instruction is encountered, the TC1.6.2 CPU does not immediately know which PRS value to use for the instruction fetch operation from the return address.

In an attempt to reduce the delay, TC1.6.2 CPU predicts the PRS value and performs a speculative instruction fetch for the return address using the predicted PRS value. If the MPU is enabled and the PRS prediction is not correct, any result from the speculative fetch operation is ignored, and an instruction fetch is performed using the correct PRS value.

#### Example of instruction fetch with incorrect PRS prediction

The following code snippet assumes that the function is called with a value of PRS=M, but executes the corresponding RET instruction with a different value of PRS.

```
Os_PrivilegedFunction:
    ..
    CALL Os_SupportFunction
    <return target>

Os_SupportFunction:
    ..
    Change PRS value to N
    ..
    RET                ; Return to Os_PrivilegedFunction
```

Assume the case where function `Os_PrivilegedFunction` calls `Os_SupportFunction` (with PRS=M). When the corresponding RET instruction is executed, the CPU (speculatively) fetches from the return target with the current PRS value (PRS=N). After the CSA is read, the actual PRS value is determined to be M. As this does not match the PRS value used for the speculative fetch, the result from the speculative fetch is ignored, and a re-fetch from the return target address with PRS=M is performed.

When the MPU is disabled (PROTEN==0), the PRS value is irrelevant and the mismatch is ignored.

When the MPU is enabled (PROTEN==1), the PRS misprediction may result in extra cycles when executing the return instruction. This performance impact is only visible when executing from the Program ScratchPad RAM (PSPR) memory or from an address already present in Program Cache.

There is no functional impact (no CPU state change) resulting from the PRS misprediction.

Under some circumstances, the PRS prediction may use inappropriate PRS values resulting in an excessive number of PRS mispredictions.

#### Common case for PRS misprediction

The PRS prediction scheme is modal. It toggles on an exception entry. The exception PRS prediction mode may not be suitable if there is a significant amount of function calls.

The normal mode can be restored by clearing the interrupt stack bit (PSW.IS); the PSW.IS bit can then be restored to its previous value.

**Note:** *Interrupts must be disabled before attempting to update the PSW.IS bit.*

## 4 Application hints

Example of execution sequence:

```
Exception entry
..
CALL foo      ; foo() function call
..
; Start of foo()
..
RET          ; foo() function return will likely cause a PRS mispredict
```

To restore normal PRS prediction, the following code can be used:

```
Exception entry
MOV D10, 0x880
MFCR D11, PSW
MTCR PSW, D10 ; clear PSW.IS
MTCR PSW, D11 ; restore PSW
CALL foo      ; foo() function call
..           ; foo() function return target, PRS prediction operates as expected
RFE
```

Alternative solution:

```
foo(...)
{
    // interrupts need to be disabled at this point
    uint32_t psw_value = mfcr(PSW);
    mtcrc(PSW, psw_value & 0xFFFFDFFF); // clear PSW.IS
    mtcrc(PSW, psw_value);              // restore PSW.IS
    // interrupts can be re-enabled here if required
    [...]
}
```

### Recommendation

When the MPU is enabled (PROTEN==1), the PRS prediction may use inappropriate PRS values resulting in an excessive number of PRS mispredictions. If the application contains performance relevant code executed in an interrupt handler or trap handler (for example: in a SYSCALL handler) it is recommended to add the PSW. IS modification code (as detailed above) at the start of the handler, to avoid the performance degradation when the MPU is enabled.

## 4.27 [DMA\_TC.H018] Maximum size of circular buffers is 32 Kbytes

### Description

Section "Circular Buffer" in the DMA chapter of the user manual states:

- "Possible buffer sizes of the circular buffers can be  $2^{\text{CBL S}}$  or  $2^{\text{CBL D}}$  bytes (= 1, 2, 4, 8, 16, ... up to 64k bytes)"  
The maximum size specified in that sentence is incorrect.

## 4 Application hints

### Documentation update

The maximum size of the circular buffers is 32 Kbytes. The corresponding sentence in the user manual shall be corrected as follows:

- Possible buffer sizes of the circular buffers can be  $2^{\text{CBL5}}$  or  $2^{\text{CBLD}}$  bytes (= 1, 2, 4, 8, 16, ... up to 32 Kbytes)

### 4.28 [DTS\_TC.H002] Unexpected alarms after start-up/wake-up when temperature is close to lower/upper limit

#### Description

The result of the first temperature measurement received from the Die Temperature Sensor (DTS) after start-up from cold PORST or wake-up from standby mode is inaccurate due to parallel processing of sensor trimming.

#### Effect

If temperature is close ( $< 10$  K) to the thresholds defined in register DTSLIM, alarms ALM9[0] or ALM9[1] in SMU\_core and ALM21[9] or ALM21[8] in SMU\_stdby can be triggered falsely indicating lower temperature limit underflow (ALM9[1], ALM21[8]) or upper limit overflow (ALM9[0], ALM21[9]). Also, the corresponding flag DTSLIM.LLU or DTSLIM.UOF is set.

#### Recommendation

The application software shall clear the respective flag in DTSLIM and **afterwards** clear related SMU alarms. In case alarms are retriggered, application SW shall consider these as real alarms, and trigger a reaction within the FTTI (Fault Tolerant Time Interval) of the respective application.

### 4.29 [EVR\_TC.H001] External input capacitor value - Additional Data Sheet footnote

#### Description

The following footnote shall be added to parameter “External input capacitor value” (symbol  $C_{\text{IN}}$ ) in table “EVRC SMPS External components” in the TC3yx Data Sheets:

**Note:** *From EVRC view there is no defined hard limit for the maximum value of the input capacity, the specified upper limit is determined by the measurement setup. From EVRC view, the typical value of  $C_{\text{IN}}$  can be up to ~4x higher compared to the value listed in the Data Sheet.*

### 4.30 [FLASH\_TC.H021] Flash Wait State configuration

#### Description

Configuring flash wait states in your application is critical for correct operation.

Refer to these parts of the documentation of the respective TC3\*x design step for guidance on avoiding data read errors over the lifetime of the device:

- Data Sheet, chapter “Flash Target Parameters”:
  - minimum access times  $t_{\text{PF}} / t_{\text{PF ECC}}$  for PFLASH
  - and  $t_{\text{DF}} / t_{\text{DF ECC}}$  for DFLASH
- AURIX™ TC3xx User’s Manual, NVM chapter “Configuring Flash Read Access Cycles” (6.5.2.1.2 in TC3xx User's Manual V2.0.0)
- Application Note AP32381 (AURIX™ 2nd Generation startup and initialisation)

## 4 Application hints

When **increasing** the SRI and FSI clock frequencies: first set registers HF\_PWAIT and HF\_DWAIT to the correct values, and then change the clock configuration.

When **decreasing** the SRI and FSI clock frequencies: first change the clock configuration, and then set registers HF\_PWAIT and HF\_DWAIT to the correct values.

**Note:** *Applications that omit configuration of HF\_PWAIT and HF\_DWAIT may work in the development phase, but encounter data read errors in the field.*

### 4.31 [FLASH\_TC.H024] PFLASH erase and program time is affected by time slicing but not clearly documented

#### Description

In the NVM chapter in the TC3xx user manual, the sub-section "Time Slice Control and Flash Parameters" describes the erase and program time increase due to time slicing. It is mentioned only for DFLASH in the user manual. However, this is applicable for PFLASH also.

#### Documentation update

The timing penalty of time slicing described in the sub-section "Time Slice Control and Flash Parameters" of the NVM chapter in the TC3xx user manual is applicable not only for DFLASH but it is also applicable for PFLASH operations when running concurrently to HSM operations.

### 4.32 [FLASH\_TC.H026] Additional information about Test Pass Marker

#### Description

In the UCB\_TEST of TC2xx and TC3xx devices, there are 4 bytes reserved for Test Pass Marker.

A production device with a Test Pass Marker value different than 80658383<sub>H</sub>, is specified to be discarded. This is to prevent any possibility of an unknown fault in Infineon production or customer procurement process, resulting in an invalid device used by the customer. In such cases Infineon appreciates a notification.

In accordance with Infineon actual assessment, devices that are in the field without checked Test Pass Marker are not subject to a field recall.

**Note:** *For TC2xx, it is referred as UCB\_IFX in the TC2xx user manual.*

### 4.33 [FlexRay\_AI.H004] Only the first message can be received in External Loop Back mode

#### Description

If the loop back (TXD to RXD) will be performed via external physical transceiver, there will be a large delay between TXD and RXD.

A delay of two sample clock periods can be tolerated from TXD to RXD due to a majority voting filter operation on the sampled RXD.

Only the first message can be received, due to this delay.

To avoid that only the first message can be received, a start condition of another message (idle and sampling '0' -> low pulse) must be performed.

The following procedure can be applied at one or both channels:

- wait for no activity (TEST1.AOx = 0 -> bus idle)
- set Test Multiplexer Control to I/O Test Mode (TEST1.TMC = 2), simultaneously TXDx = TXENx = 0

#### **4 Application hints**

- wait for activity (TEST1.AOx = 1 -> bus not idle)
- set Test Multiplexer Control back to Normal signal path (TEST1.TMC = 0)
- wait for no activity (TEST1.AOx = 0 -> bus idle)

Now the next transmission can be requested.

#### **4.34 [FlexRay\_AI.H005] Initialization of internal RAMs requires one eray\_bclk cycle more**

##### **Description**

The initialization of the E-Ray internal RAMs as started after hardware reset or by CHI command CLEAR\_RAMs (SUCC1.CMD[3:0] = 1100<sub>B</sub>) takes 2049 eray\_bclk cycles instead of 2048 eray\_bclk cycles as described in the E-Ray Specification.

Signalling of the end of the RAM initialization sequence by transition of MHDS.CRAM from 1<sub>B</sub> to 0<sub>B</sub> is correct.

#### **4.35 [FlexRay\_AI.H006] Transmission in ATM/Loopback mode**

##### **Description**

When operating the E-Ray in ATM/Loopback mode there should be only one transmission active at the same time. Requesting two or more transmissions in parallel is not allowed.

To avoid problems, a new transmission request should only be issued when the previously requested transmission has finished. This can be done by checking registers TXRQ1/2/3/4 for pending transmission requests.

#### **4.36 [FlexRay\_AI.H007] Reporting of coding errors via TEST1.CERA/B**

##### **Description**

When the protocol engine receives a frame that contains a frame CRC error as well as an FES decoding error, it will report the FES decoding error instead of the CRC error, which should have precedence according to the non-clocked SDL description.

This behavior does not violate the FlexRay protocol conformance. It has to be considered only when TEST1.CERA/B is evaluated by a bus analysis tool.

#### **4.37 [FlexRay\_AI.H009] Return from test mode operation**

##### **Description**

The E-Ray FlexRay IP-module offers several test mode options

- Asynchronous Transmit Mode
- Loop Back Mode
- RAM Test Mode
- I/O Test Mode

To return from test mode operation to regular FlexRay operation we strongly recommend to apply a hardware reset via input eray\_reset to reset all E-Ray internal state machines to their initial state.

**Note:** *The E-Ray test modes are mainly intended to support device testing or FlexRay bus analyzing. Switching between test modes and regular operation is not recommended.*

## 4 Application hints

### 4.38 [FlexRay\_AI.H010] Driver software must launch CLEAR\_RAMs command before reading from E-Ray RAMs

#### Description

After a Power-on-Reset, the RAMs used by the E-Ray module must be written once. Reading from RAM locations before at least writing once to them may cause a Parity Error Trap (AUDO\* microcontroller family) or an ECC Error Trap.

#### Recommendation

The recommended solution is to trigger a CLEAR\_RAMs command (via register SUCC1). CLEAR\_RAMs fills a defined value into all memory locations. A safe initialization sequence of the E-Ray RAM blocks using the CLEAR\_RAMs command is described in section “CLEAR\_RAMs Command” of the E-Ray chapter in the corresponding AURIX™ user manual.

An alternate solution is to write explicitly by software to all RAM locations, which are intended to be read later, for example by writing the complete configuration and writing into all allocated message buffers, including receive buffers. The latter activity may be required if buffers are configured to store frames sent in the dynamic segment. The sent frames may be smaller than the configured buffer size. If the software reads the amount of configured data (not the amount of received data), it may read from non-activated RAM locations.

For AURIX™ devices, as a further option, the MBIST auto-initialization algorithm may be used. See section “Filling a Memory with Defined Contents” in the corresponding AURIX™ user manual.

### 4.39 [FlexRay\_AI.H011] Behavior of interrupt flags in FlexRay™ Protocol Controller (E-Ray)

#### Description

In the corner case described below, the actual behavior of the interrupt flags of the FlexRay™ Protocol Controller (E-Ray) differs from the expected behavior.

**Note:** *This behaviour only applies to E-Ray interrupts INT0 and INT1. All other E-Ray interrupts are not affected.*

#### Expected behavior

When clearing an interrupt flag by software, the resulting value of the flag is expected to be zero.

A hardware event that occurs afterwards then leads to a zero to one transition of the flag, which in turn leads to an interrupt service request.

#### Actual behavior in corner case

When the interrupt flag is being cleared by software in the same clock cycle as a new hardware event sets the flag again, then the hardware event wins and the flag remains set without being cleared.

As interrupt requests are generated only upon zero to one transitions of the flag, no interrupt request will be generated for this flag until the flag is successfully cleared by software later on.

#### Workaround

After clearing the flag, the software shall read the flag and repeat clearing until the flag reads zero.

## 4 Application hints

### 4.40 [FlexRay\_TC.H003] Initialization of E-Ray RAMs - Documentation update

#### Description

After Power-On reset, the E-Ray RAMs hold arbitrary values which causes ECC errors (MHDS) when a read operation is performed on an E-Ray RAM location. Hence the E-Ray RAMs should be initialized always after a Power-On reset.

#### Recommendation

The E-Ray RAMs initialization can be performed using the CLEAR\_RAMs command of the E-Ray module. A safe initialization sequence of the E-Ray RAM blocks using the CLEAR\_RAMs command is described in section “CLEAR\_RAMs Command” of chapter “FlexRay™ Protocol Controller (E-Ray)” in the AURIX™ TC3xx User’s Manual.

#### Documentation update

**Note:** *In order to ensure proper FlexRay communication, RAM test mode must be explicitly disabled via TEST1.TMC = 00b at the end of the initialization sequence.*

Therefore, Step16 in section “CLEAR\_RAMs Command” of the TC3xx User’s Manual must be updated from

- 16. Switch off Test Mode: TEST1.WRTEN = 0b
- to
- 16. Switch off Test Mode: **TEST1.TMC = 00b and** TEST1.WRTEN = 0b

### 4.41 [FlexRay\_TC.H004] Bit WRECC in register TEST2 has no function

#### Description

In the AURIX™ implementation of the E-Ray module, bit WRECC in register TEST2 has no function.

#### Recommendation

The value read from WRECC should not be evaluated by software, the value written (0<sub>B</sub> or 1<sub>B</sub>) to it is irrelevant. For new software projects, keep bit WRECC at its reset value (0<sub>B</sub>) for easier migration to future AURIX™ generations.

### 4.42 [FlexRay\_TC.H005] E-Ray OTGB2 trigger set active even if disabled

#### Description

The trigger set TS32\_SCSC of the E-Ray IP-module is associated with OTGB2. An internal “valid” signal should be asserted only in case the trigger set is selected via OTSS.OTGB2.

#### Expected behavior

The OTGB2 trigger set valid signal should be gated by the bit-field OTSS.OTGB2.

#### Actual behavior

The E-Ray IP does not gate the valid signal with the OTSS.OTGB2 state, but only the data are gated. Meaning the OTGB2 trigger valid signal is only dependent on the slot counter and transfer buffer state changes, irrespective of the OTSS.OTGB2 value.

## 4 Application hints

### Recommendation

Ignore all OTGB2 E-Ray triggers when data reported is only 0s.

### 4.43 [FPI\_TC.H003] Burst write access may lead to data corruption and to a stalling issue

#### Description

For the FPI slave modules listed below, if a write burst access is aborted on the last beat or more than one burst access is executed in a fast sequence, this may lead to data corruption of all future accesses. No error is generated when the burst access is aborted and the device and the bus can be stalled which makes a normal working impossible. For example, a CPU performing the burst write accesses may appear to be permanently blocked. It is worth noting that a burst write access can be generated from the CPU by a 64-bit write access (BTR2).

This problem only affects the following modules:

- CONVCTRL, EVADC, PMS, SCR XRAM

#### Recommendation

It is advised not to perform burst accesses to registers in CONVCTRL, EVADC, PMS, and SCR XRAM.

### 4.44 [GPT12\_TC.H002] Bits TxUD and TxUDE in incremental interface mode - Additional information

#### Description

The present description of the incremental interface mode for timers T2, T3, T4 in the User's Manual, including figures and tables, implicitly refers to the following configuration of bits TxUD and TxUDE ( $x = 2, 3, 4$ ):

- TxUD = 0<sub>B</sub>
- TxUDE = 1<sub>B</sub>

This is the recommended and validated setting for these bits in incremental interface mode.

#### Additional information

When bit TxUD = 1<sub>B</sub>, the count direction of timer Tx is inverted compared to the setting with TxUD = 0<sub>B</sub> in incremental interface mode.

The setting of bit TxUDE is irrelevant in incremental interface mode, the behavior of Tx for TxUDE = 0<sub>B</sub> and TxUDE = 1<sub>B</sub> is identical. The figures related to incremental interface mode shall be interpreted as if TxUDE is permanently tied to 1<sub>B</sub>.

### 4.45 [GTM\_AI.H480] SPEC-TIM: Wrong action description for TPIM mode

#### Description

In TIM Pulse Integration Mode (TPIM) with External Capture (TIM[i]\_CH[x]\_CTRL.EXT\_CAP\_EN = 1), the capture is done only with the external capture signal and not with the rising or falling edge of the TIM input signal.

Therefore in the chapter describing the TPIM mode the action description in the table "Operation depending ..." (GTM4.1 spec.: TIM\_2138) rows 2 and 4 are wrong.

In both rows it is described that if inc\_cnt == true, a capture as well as a TIM\_NEWVAL\_IRQ has to be executed. But this is not the case and has to be removed. Only the last line in both rows of the table ("inc\_cnt = false") is correct.

**4 Application hints**

**Table 21** TC3xx - Operation depending on CMU clock, DSL and the input signal value (inc\_cnt = false if TIM channel is enabled)

Input signal F_OUTx	Selected CMU clock	External capture	ISL	DSL	Action description
Falling edge	-	0	-	0	inc_cnt = true
Rising edge	-	0	-	0	inc_cnt = false
Rising edge	-	0	-	1	inc_cnt = true
Falling edge	-	0	-	1	inc_cnt = false
-	1	0	-	-	If inc_cnt == true then CNT++;
-	-	Rising edge	-	-	Do capture GPRx, CNTS; issue NEWVAL_IRQ; CNT=0
-	0	0	-	-	No

**Scope**

TIM

**Effects**

Contrary to the description, neither a capture nor an interrupt is triggered by a rising or falling edge of the input signal.

**Recommendation**

Consider the information in the corresponding table above.

**4.46 [GTM\_AI.H481] SPEC-TIM: Wrong description for TBCM mode**

**Description**

**Note:** Register names in the text follow the TC3xx syntax conventions. Correlation of register names:

- **TC3xx:** TIM[i]\_CH[x]\_CTRL
- **TC2xx:** TIMi\_CHx\_CTRL

In TIM Bit Compression Mode with External Capture (TIM[i]\_CH[x]\_CTRL.EXT\_CAP\_EN=1), the capture is done only with the external capture signal without dependency to the input signal level. Therefore the bit-field TIM[i]\_CH[x]\_CTRL.ISL must be set to 1. The value 0 for TIM[i]\_CH[x]\_CTRL.ISL is prohibited. The bit-field TIM[i]\_CH[x]\_CTRL.DSL is not relevant.

The following parts in section "External capture Bit Compression Mode (TBCM)" in the TBCM chapter have to be adapted as follows:

- In the prose text

**4 Application hints**

"If external capture is enabled, capturing is done for TIM[i]\_CH[x]\_CTRL.ISL=1 as defined in the next table. The value 0 for TIM[i]\_CH[x]\_CTRL.ISL is prohibited."

- In the table
  - In the action description of row 1 the part "TIM[i]\_CH[x]\_CNT++" has to be removed
  - All rows starting with row 3 have to be replaced with only one row where the content for the column of TIM[i]\_CH[x]\_CTRL.ISL has to be filled with "0 - prohibited". All other columns in row 3 have to be marked with "-" (don't care)

**Table 22 Resulting table for TC3xx and TC2xx**

- TC3xx - Capturing depended on the DSL, ISL and the input signal value, if external capture is enabled
- TC2xx - TIM Input Event Mode

Input signal F_OUTx	External capture	ISL	DSL	Action description
-	Rising edge	1	-	do capture; issue NEWVAL_IRQ
-	0	1	-	No
-	-	0 - prohibited	-	-

**Scope**

TIM

**Effects**

The input signal level defined by TIM[i]\_CH[x]\_CTRL.DSL with TIM[i]\_CH[x]\_CTRL.ISL = 0 is not taken into account.

**Recommendation**

Consider the information given above. Do not configure TIM[i]\_CH[x]\_CTRL.ISL to 0.

**4.47 [GTM\_AI.H482] SPEC-TIM: Wrong description in TBCM mode regarding TIM[i]\_CH[x]\_CTRL.GPR1\_SEL bit field**

**Description**

In TIM Bit Compression Mode it is described that the bit field TIM[i]\_CH[x]\_CTRL.GPR1\_SEL is not applicable. That is not the case and therefore the sentence mentioning that the bit field TIM[i]\_CH[x]\_CTRL.GPR1\_SEL " is not applicable in TBCM mode." in the section "TIM Bit Compression Mode" (GTM4.1 spec.: TIM\_1055) has to be ignored.

**Scope**

TIM

**Effects**

The captured value depends on the bit field TIM[i]\_CH[x]\_CTRL.GPR1\_SEL in contrast to the notion in the specification.

**Recommendation**

The value of the bit field TIM[i]\_CH[x]\_CTRL.GPR1\_SEL must be taken into account.

## 4 Application hints

### 4.48 [GTM\_AI.H497] SPEC-SPE wiring in figure is wrong

#### Description

In the GTM chapter of the corresponding user manual, the usage of the SIE inputs SIE0 and SIE2 must be swapped in the following figure:

- TC3xx: figure "SPE[i]\_IN\_PAT register representation"

In the GTM chapter of the corresponding user manual, the usage of the SIE inputs is missing in the following figure:

- TC2xx: figure "SPE[i]\_IN\_PAT register representation"

#### Scope

SPE

#### Effects

The enabling of the affected TIM input signals of the SPE is not working as expected when considering the figure as basis.

#### Recommendation

For the correct functionality, see the description of bits SIE0..2 in the following register:

- TC3xx, TC2xx: SPEi\_CTRL\_STAT

### 4.49 [GTM\_AI.H512] SPEC-SPE: Wrong signal names

#### Description

In the SPE submodule chapter, the figure "SPE Submodule Architecture" contains wrong signal names. The following replacements shall be done in the SPE\_PAT register table ("Pattern bits"):

- TIM\_CHx[48] should be replaced by NIP\_MASK[0]
- TIM\_CHy[48] should be replaced by NIP\_MASK[1]
- TIM\_CHz[48] should be replaced by NIP\_MASK[2]

#### Scope

SPE

#### Effects

A user may draw wrong conclusions on how the masking of the SPE pattern works.

#### Workaround

No workaround is available.

### 4.50 [GTM\_AI.H519] SPEC-(A)TOM: Misleading description of Continuous Counting Up Mode

#### Description

In the third list item of the paragraph, where some statements are given for Continuous Counting Up Mode with RST\_CCU0=1, the following statement for the case (A)TOM[i]\_CH[x]\_CM0.CM0=(A)TOM[i]\_CH[x]\_CM1.CM1 is given:

## 4 Application hints

"If (A)TOM[i]\_CH[x]\_CM0.CM0=(A)TOM[i]\_CH[x]\_CM1.CM1, the output switches to (A)TOM[i]\_CH[x]\_CTRL\_SOMP.SL if (A)TOM[i]\_CH[x]\_CN0.CN0=(A)TOM[i]\_CH[x]\_CM0.CM0=(A)TOM[i]\_CH[x]\_CM1.CM1 ((A)TOM[i]\_CH[x]\_CM0.CM0 has higher priority."

or in the older specification versions (before GTM3 generations):

"If (A)TOM[i]\_CH[x]\_CM0.CM0=(A)TOM[i]\_CH[x]\_CM1.CM1, the output is 100% (A)TOM[i]\_CH[x]\_CTRL.SL ((A)TOM[i]\_CH[x]\_CM0.CM0 has higher priority."

Both statements are misleading and have to be replaced by the following statement:

"As soon as (A)TOM[i]\_CH[x]\_CN0.CN0 reaches the value of (A)TOM[i]\_CH[x]\_CM0.CM0 while (A)TOM[i]\_CH[x]\_CM1.CM1 is equal to (A)TOM[i]\_CH[x]\_CM0.CM0, an edge to (A)TOM[i]\_CH[x]\_CTRL.SL is generated at the output or the output remains at (A)TOM[i]\_CH[x]\_CTRL.SL level, depending on the former level of the output ((A)TOM[i]\_CH[x]\_CM0.CM0 has higher priority)."

**Note:** *The above configuration is not suitable for generating 100% duty cycle.*

### Scope

ATOM, TOM

### Effects

Textual description can be erroneously interpreted as the configuration of (A)TOM[i]\_CH[x]\_CM0.CM0=(A)TOM[i]\_CH[x]\_CM1.CM1 is suitable to generate 100% duty cycle for the current PWM period.

This is because the potential value change to SL will happen as soon as (A)TOM[i]\_CH[x]\_CN0.CN0 reaches the value of (A)TOM[i]\_CH[x]\_CM0.CM0 while (A)TOM[i]\_CH[x]\_CM1.CM1 is equal to (A)TOM[i]\_CH[x]\_CM0.CM0.

### Recommendation

For a setup of 100% duty cycle for Continuous Counting Up Mode with RST\_CCU0=1, the following setting must be used:

- ATOM[i]\_CH[x]\_CM0.CM0 = 0 and ATOM[i]\_CH[x]\_CM1.CM1 > MAX

## 4.51 [GTM\_AI.H520] SPEC-(A)TOM: Description for 100% duty cycle incorrect

### Description

In the list of characteristics for Continuous Counting Up Mode with (A)TOM[i]\_CH[x]\_CTRL\_RST\_CCU0=1<sub>B</sub>, it is stated that for 100% duty cycle, (A)TOM[i]\_CH[x]\_CM1.CM1 must be set to MAX value. This is incorrect.

Instead, (A)TOM[i]\_CH[x]\_CM1.CM1 must be set to a value greater than MAX. Therefore, the statement must be corrected as follows:

"If ATOM[i]\_CH[x]\_CM0.CM0 = 0 and ATOM[i]\_CH[x]\_CM1.CM1 > MAX, the output is a pulse of ATOM[i]\_CH[x]\_CTRL\_SOMP.SL for the period MAX (i.e. 100% duty cycle PWM signal)."

### Scope

TOM, ATOM

### Effects

Output behavior for 100% duty cycle is not correct.

### Workaround

No workaround is available.

## 4 Application hints

### 4.52 [GTM\_AI.H521] SPEC-ATOM: Missing information for SOMB mode

#### Description

If the channel is configured in SOMB mode and controlled over configuration interface, it is mandatory to always write both bit-fields ATOM[i]\_CH[x]\_SR0.SR0 and ATOM[i]\_CH[x]\_SR1.SR1 regardless of the compare strategy. Therefore, the following note must be added inside the chapter "SOMB controlled over configuration interface":

**Note:** *It is mandatory to always write both bit-fields ATOM[i]\_CH[x]\_SR0.SR0 and ATOM[i]\_CH[x]\_SR1.SR1 regardless of the compare strategy.*

#### Scope

ATOM

#### Effects

In case only one of the bit-fields (ATOM[i]\_CH[x]\_SR0.SR0 and ATOM[i]\_CH[x]\_SR1.SR1) is written, the behavior of the operation register bit-fields ATOM[i]\_CH[x]\_CM0.CM0, ATOM[i]\_CH[x]\_CM1.CM1 and ATOM[i]\_CH[x]\_STAT.ACBI will be unexpected.

#### Workaround

No workaround is available.

### 4.53 [GTM\_AI.H525] SPEC-(A)TOM: Description for 0% duty cycle incorrect

#### Description

In the list of characteristics for Continuous Counting Up Mode with (A)TOM[i]\_CH[x]\_CTRL.RST\_CCU0=1<sub>B</sub>, it is stated that configuration of 0% duty cycle is independent of (A)TOM[i]\_CH[x]\_CM1.CM1. This is incorrect and must be modified.

Instead, (A)TOM[i]\_CH[x]\_CM1.CM1 must be set to 0. Therefore, the statement must be corrected as follows:  
"If (A)TOM[i]\_CH[x]\_CM0.CM0 > MAX and (A)TOM[i]\_CH[x]\_CM1.CM1 = 0, the output is !  
(A)TOM[i]\_CH[x]\_CTRL\_SOMP.SL (i.e. 0% duty cycle PWM signal)."

#### Scope

ATOM, TOM

#### Effects

The configuration of a value for (A)TOM[i]\_CH[x]\_CM1.CM1 not equal to 0 does not lead to 0% duty cycle on the output signal.

#### Workaround

No workaround is available.

## 4 Application hints

### 4.54 [GTM\_AI.H526] SPEC-(A)TOM: Missing information for SOMP mode

#### Description

The following information about the priority between the CCU0 and CCU1 compare match (configured by (A)TOM[i]\_CH[x]\_CM0.CM0 and (A)TOM[i]\_CH[x]\_CM1.CM1) is missing in the specification and has to be added to the list of characteristics for Continuous Counting Up Mode with (A)TOM[i]\_CH[x]\_CTRL.RST\_CCU0 = 1<sub>B</sub>:

For ATOM:

"If ATOM[i]\_CH[x]\_CM0.CM0 is set to MAX, an edge to ATOM[i]\_CH[x]\_CTRL\_SOMP.SL is generated at the output at the very end of the current period or the output remains at the ATOM[i]\_CH[x]\_CTRL\_SOMP.SL level, depending on the previous level of the output."

Be aware that this applies even if the value of ATOM[i]\_CH[x]\_CM1.CM1 is updated to 0 for the new period (ATOM[i]\_CH[x]\_CM0.CM0 has higher priority).

For TOM:

"If TOM[i]\_CH[x]\_CM0.CM0 is set to MAX, an edge to TOM[i]\_CH[x]\_CTRL.SL is generated at the output at the very end of the current period or the output remains at the TOM[i]\_CH[x]\_CTRL.SL level, depending on the previous level of the output."

Be aware that this applies even if the value of TOM[i]\_CH[x]\_CM1.CM1 is updated to 0 for the new period (TOM[i]\_CH[x]\_CM0.CM0 has higher priority).

#### Scope

ATOM, TOM

#### Effects

The user may not be aware that the setting of (A)TOM[i]\_CH[x]\_CM0.CM0=MAX will definitely lead to a setting of the output to (A)TOM[i]\_CH[x]\_CTRL.SL at the very end of the period.

#### Recommendation

To reach a 0% duty cycle after a period where ATOM[i]\_CH[x]\_CM0.CM0 is set to MAX, the following applies:

- Hint : At the same time as ATOM[i]\_CH[x]\_SR0.SR0 and ATOM[i]\_CH[x]\_SR1.SR1 are updated before the unwanted output edge, ATOM[i]\_CH[x]\_CM0.CM0 must be set to the same value as ATOM[i]\_CH[x]\_SR0.SR0. This will prevent the priority decision

### 4.55 [GTM\_AI.H528] Spec-(A)TOM: Missing priority information

#### Description

There is a missing priority information in the figures "ATOM channel architecture" (ATOM) and "TOM Channel ... architecture" (TOM). For the output driving storing element "SOUR" in the mentioned figures, CCU0 has priority over CCU1. It is when TRIG\_CCU0=1, this is dominant over TRIG\_CCU1.

#### Scope

TOM, ATOM

#### Effects

Unclear behavior.

#### Workaround

No workaround is available.

## 4 Application hints

### 4.56 [GTM\_AI.H540] SPEC-TIM: Wrong update condition in TSSM mode

#### Description

In TIM TSSM mode ("Signal Generation with TIM Serial Shift Mode"), it is stated that the update of TIM[i]\_CH[x]\_CNT will only take place once on a trigger "...if the TIM[i]\_CH[x]\_GPRO was written by the CPU". But the update of TIM[i]\_CH[x]\_CNT does not depend on whether a value has been written by CPU. Therefore, this update condition shall be removed from specification.

#### Scope

TIM

#### Effects

A user assumes an incorrect update condition.

#### Workaround

No workaround available.

### 4.57 [GTM\_AI.H543] SPEC-TIM: Wrong clock name referenced

#### Description

In the "Timer Input Module (TIM)" specification chapter, it is stated that the incoming input signals are synchronized with the clock SYS\_CLK, resulting in a delay of two SYS\_CLK periods for the incoming signals. The reference to the clock signal SYS\_CLK is wrong. Instead, it has to reference the internal cluster clock cls\_clk.

#### Scope

TIM

#### Effects

Assumption of an incorrect clock for input synchronization, and also a differing latency.

#### Workaround

No workaround available.

### 4.58 [GTM\_AI.H803] SPEC-(A)TOM: Missing priority information for register update

#### Description

The following information is missing in the specification and has to be placed inside the TGC Sub-unit/AGC Sub-unit chapter:

- Inside ATOM chapter: The trigger condition has always priority over the bus write access to the ATOM[i]\_AGC\_OUTEN\_STAT and ATOM[i]\_AGC\_ENDIS\_STAT registers, even if ATOM[i]\_AGC\_OUTEN\_CTRL.OUTEN\_CTRL[k] / ATOM[i]\_AGC\_ENDIS\_CTRL.ENDIS\_CTRL[k] is set to 00<sub>B</sub>. This means that the bus write access to ATOM[i]\_AGC\_OUTEN\_STAT and ATOM[i]\_AGC\_ENDIS\_STAT register is ignored in the clock cycle when the trigger condition is active
- Inside TOM chapter: The trigger condition has always priority over the bus write access to the TOM[i]\_TGC[g]\_OUTEN\_STAT and TOM[i]\_TGC[g]\_ENDIS\_STAT registers, even if TOM[i]\_TGC[g]\_OUTEN\_CTRL.OUTEN\_CTRL[k] / TOM[i]\_TGC[g]\_ENDIS\_CTRL.ENDIS\_CTRL[k] is set to 00<sub>B</sub>.

#### 4 Application hints

This means that the bus write access to TOM[i]\_TGC[g]\_OUTEN\_STAT and TOM[i]\_TGC[g]\_ENDIS\_STAT register is ignored in the clock cycle when the trigger condition is active

**Note:** *The trigger override does not happen if the trigger is a HOST\_TRIG, as this is initiated by a bus write itself and cannot happen at the same time as another bus write to the register.*

**Note:** *This AppHint is published as GTM-IP-523 by Bosch.*

#### Scope

TOM, ATOM

#### Effects

In (A)TOM the bus write access to the "OUTEN\_STAT" and "ENDIS\_STAT" registers is overridden by a trigger update and the desired values are not written into the register.

#### Recommendation

To nevertheless ensure that the desired value is actually stored in the target register, consider one of the following hints:

1. Write first the channel k within "ENDIS\_CTRL" ("OUTEN\_CTRL") and write the desired channel k in "ENDIS\_STAT" ("OUTEN\_STAT") afterward. For the first write accesses to "ENDIS\_CTRL" ("OUTEN\_CTRL"), it has to be ensured that all unused channels within "ENDIS\_CTRL" ("OUTEN\_CTRL") are set to 11<sub>B</sub>. This way, either the asynchronous write or the synchronous write becomes effective
2. If recommendation 1 is not the case, then read back the value of the "OUTEN\_STAT" and "ENDIS\_STAT" registers to ensure the written value is actually present in the register

### 4.59 [GTM\_TC.H010] Trigger Selection for EVADC and EDSADC

#### Description

If the GTM output selection in the SELz bit-fields for ADC triggers (registers ADCTRIGxOUTy, DSADCOUTSELxy) is changed during SW runtime, multi-bit changes may lead to unintended ADC triggering.

#### Recommendation

Before changing the trigger source in the GTM output selection fields SELz, ensure that the ADCs at the trigger destination will not react on intermediate state changes of the trigger signals.

### 4.60 [GTM\_TC.H019] Register GTM\_RST - Documentation Update

#### Description

In the current documentation, bit 0 in register GTM\_RST is described as

- **Type:** r
- **Description:** Reserved - Read as zero, should be written as zero

#### Documentation Update

Actually, bit 0 in register GTM\_RST is implemented as follows:

- **Type:** rw
- **Description:** Reserved - Read as zero, **shall** be written as zero

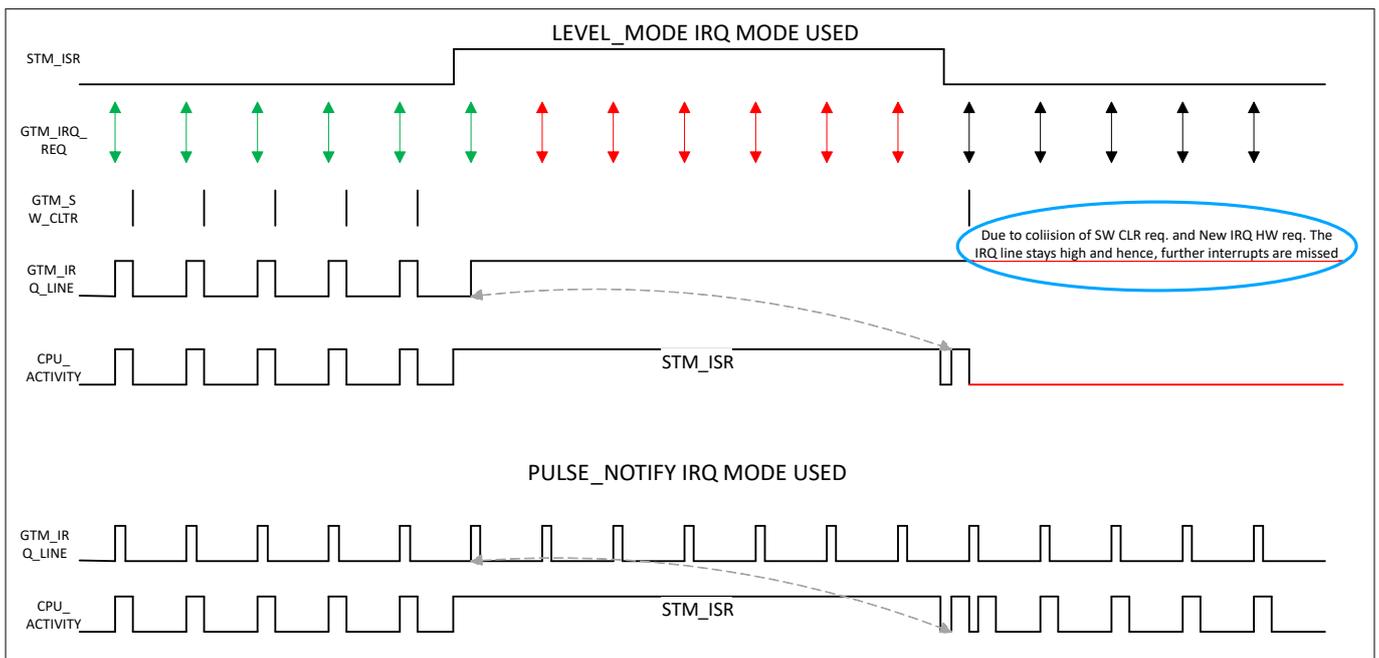
**4 Application hints**

**Note:** This Application Hint relates to problem GTM-IP-316 reported by the GTM IP supplier. On this AURIX™ TC3xx device step, the reported problem has no effect, independent of the value written to bit GTM\_RST.0. However, GTM\_RST.0 shall always be written with 0<sub>B</sub> as documented in the register description to ensure compatibility with future versions.

**4.61 [GTM\_TC.H021] Interrupt strategy mode selection in IRQ\_MODE**

**Description**

The default setting for field IRQ\_MODE in register IRQ\_MODE is Interrupt Level mode (00<sub>B</sub> ). Figure “GTM interrupts” in chapter “GTM Implementation” of the TC3xx User’s Manual shows how the interrupt signal (GTM\_IRQ\_XXX) triggers an interrupt towards the Interrupt Router (IR), depending on IRQ\_MODE. As described in the text below this figure, while using Level mode, if more internal “interrupt” events are generated (i.e. two TOM channels generating a CCU0 interrupt), just one interrupt signal is sent to the IR, and no more interrupts are triggered until the SW clears the GTM\_IRQ\_XXX line towards the IR. Hence, in Level Mode, in some scenarios where another interrupt request is generated by GTM while the ISR handle also requests a SW clear, then, as the interrupt event is dominant over the clear event (for simultaneous interrupt and clear events), GTM\_IRQ\_XXX is not cleared and remains high. As a consequence, the IR observes no transition on GTM\_IRQ\_XX. Thus, any forthcoming interrupt events in this scenario are lost as there is no chance to release the CPU IRQ when a collision happens as shown in Figure below.



**Figure 6 Interrupt Level vs. Pulse-Notify mode - Corner Case Example**

If Pulse-Notify mode is selected, every internal trigger will be forwarded to the IR, irrespective of the time of occurrence and the clear event, as the pulse-notify leads to set and reset of GTM\_IRQ\_XXX as compared to only setting of the GTM\_IRQ\_XXX line in Level mode.

**Recommendation**

Therefore, it is recommended to use the Pulse-Notify mode to ensure that none of the interrupts might be lost by the IR even in corner timing cases.

As described above, this scenario in Level mode is only a corner case due to the timing of the SW and ISR handle. If using a different IRQ\_MODE setting, evaluate your system performance for sufficient timing margin.

## 4 Application hints

### 4.62 [GTM\_TC.H027] Register ODA (OCDS Debug Access) - Documentation update

#### Description

In the GTM chapter of the TC3xx User's Manual, the following note is located below the description of register ODA in section "OCDS Debug Access Register":

- 6. TIM[i]\_CH[x]\_GPR0/1: Reading these register will reset the ECNT counter

#### Documentation update

Note 6. (see above) does not apply and shall be deleted.

### 4.63 [GTM\_TC.H035] Type of bit-fields for xxx\_IRQ\_NOTIFY registers should be marked as rwh

#### Description

In the GTM chapter of the TC3xx user manual, the value "rw" mentioned in the column "Type" of bit-fields for the following registers is incorrect.

- GTM\_IRQ\_NOTIFY
- ARU\_IRQ\_NOTIFY
- BRC\_IRQ\_NOTIFY
- FIFO[i]\_CH[z]\_IRQ\_NOTIFY
- TIM[i]\_CH[x]\_IRQ\_NOTIFY
- TOM[i]\_CH[x]\_IRQ\_NOTIFY
- ATOM[i]\_CH[x]\_IRQ\_NOTIFY
- MCS[i]\_CH[x]\_IRQ\_NOTIFY
- DPLL\_IRQ\_NOTIFY
- SPE[i]\_IRQ\_NOTIFY
- CMP\_IRQ\_NOTIFY

#### Scope

The issue is related to documentation only.

#### Documentation update

The value mentioned in the column "Type" must be corrected as "rwh" instead of "rw" of bit-fields for the following registers:

- GTM\_IRQ\_NOTIFY
- ARU\_IRQ\_NOTIFY
- BRC\_IRQ\_NOTIFY
- FIFO[i]\_CH[z]\_IRQ\_NOTIFY
- TIM[i]\_CH[x]\_IRQ\_NOTIFY
- TOM[i]\_CH[x]\_IRQ\_NOTIFY
- ATOM[i]\_CH[x]\_IRQ\_NOTIFY
- MCS[i]\_CH[x]\_IRQ\_NOTIFY
- DPLL\_IRQ\_NOTIFY
- SPE[i]\_IRQ\_NOTIFY
- CMP\_IRQ\_NOTIFY

**4 Application hints**

**4.64 [INT\_TC.H006] Number of SRNs supporting external interrupt/service requests – Documentation update**

**Description**

SCU chapter “Output Gating Unit” of the TC3xx User’s Manual (V1.6 and newer versions) contains the following statement:

- “Interrupt/service requests can be generated only by the ERU OGU[0-3] via its outputs ERU\_IOUT[0-3].”

Actually, all ERU OGUs [0-7] can generate interrupts/service requests by outputs ERU\_IOUT[0-7]. As only 4 interrupt nodes are available for ERU interrupts/service requests, multiple sources share interrupt nodes as shown in [Table 23](#) below.

The following statement in the IR chapter “External Interrupts” of the TC3xx User’s Manual is conflicting with the description above:

- “Eight SRNs (Int\_SCUSRC[7:0]) are reserved to handle external interrupts.”

**Documentation update**

The statement in the IR chapter shall be changed as follows:

- “**Four** SRNs (SRC\_SCUERUx (x=0-3)) are implemented to handle external interrupts.”

Table “OGU to SRC connection” in the SCU chapter shall be changed as follows:

**Table 23 OGU to SRC connection**

<b>OGUy.ERU_IOUTy (OGU I-output signal)</b>	<b>SRC_SCUERUx (interrupt SRC register)</b>
OGU0.ERU_IOUT0, OGU4.ERU_IOUT4	SRC_SCUERU0
OGU1.ERU_IOUT1, OGU5.ERU_IOUT5	SRC_SCUERU1
OGU2.ERU_IOUT2, OGU6.ERU_IOUT6	SRC_SCUERU2
OGU3.ERU_IOUT3, OGU7.ERU_IOUT7	SRC_SCUERU3

In the SCU chapter, all references regarding the relation between ERU\_IOUT\*, ERU\_INT\* and SRC\_SCUERU\* shall be interpreted according to the table above:

The ERU\_IOUTi and ERU\_IOUTi+4 outputs are signaled through the ERU\_INTi signals to the service request control registers SRC\_SCUERUi in the interrupt router module (IR); i=0-3.

**4.65 [INT\_TC.H007] Interrupt router SRC\_xxx register is not always read with correct value**

**Description**

If the SRC register is accessed from different CPUs or other masters in the chip, it is possible that the second access receives an outdated data when it is a read access. The problem occurs if several masters access the same SRC register too soon after each other.

If a write access to the SRC register is followed immediately by a read access to the same SRC register, then the read access can be executed in the Interrupt Router (IR) before the change caused by the previous write access is visible in the SRC register. This can also happen if two masters access one SRC register with instructions that are executed in the system as Read Modify Write (RMW) accesses.

Then, it is possible that the read of the second RMW is executed before the write of the first RMW access becomes visible in the SRC register.

**Recommendation**

We recommend to access each SRC register from only one master at a time. Or, when using two masters to access the same SRC register, it is recommended to use the byte accesses in order to avoid cases in which a

#### 4 Application hints

write operation is followed by a read operation to the same address. Additionally, when using byte accesses, both masters must access different bytes within the register.

### 4.66 [ISTANDBY\_TC.H001] Characteristics of standby current $I_{\text{STANDBY}}$ on TC33x/TC32x in QFP-80 and QFP-100 packages

#### Description

**Note:** *This effect only applies to TC332 and TC322 in QFP-80 and TC333\* and TC323\* in QFP-100 packages, and only affects the use case where  $V_{\text{EXT}}$  is not switched off during Standby mode.*

As pin P33.12 is unavailable on TC33x/TC32x in QFP-80 and QFP-100 packages, and the internal pad is floating in Standby mode in these variants and no pull devices are active when  $V_{\text{EXT}}$  is not switched off, during Standby entry the Standby current value  $I_{\text{STANDBY}}$  as specified in the Data Sheet is reached after a transient delay of 10 - 100 ms (typical).

The Standby current has a higher value during the transient phase of 200  $\mu\text{A}$  – 600  $\mu\text{A}$  @-40°C and <170  $\mu\text{A}$  @25°C after which it settles to the  $I_{\text{STANDBY}}$  target value as specified in the Data Sheet.

**Note:** *If  $V_{\text{EXT}}$  is switched off during Standby mode the internal pull-up at pad P33.12 is activated, this means no floating state at this pad and therefore no additional cross current.*

**Note:** *If pin P33.12 is available in higher pin-count packages, the pin level can be driven either using external pull devices or internal port IO functions and also be controlled by the PMSWCR5.TRISTREQ function.*

### 4.67 [LBIST\_TC.H003] Update reset behavior of LBISTCTRL0 and LBISTCTRL3 register - Additional information

#### Description

Even though the LBISTCTRL3.[31:0] and LBISTCTRL0.28 register bits are cleared by a power-on reset they will automatically recover their values from stored contents of the central LBIST controller in the TCU (Test Control Unit) afterwards.

So on first software access the user will never see the initial reset values, but the updated LBIST done status and MISR result from the TCU LBIST controller.

The stored LBIST done status and MISR result in the central TCU LBIST controller will be cleared only through an externally applied warm power-on reset, during any cold power-on reset (triggered from EVR voltage monitors), or through the LBISTCTRL0.1 register bit (LBISTRES).

### 4.68 [LBIST\_TC.H005] Effects of LBIST execution on P33.8

#### Description

As described in the chapter "LBIST Support" in the SCU chapter of the TC3xx user manual, the static GPIO behavior during LBIST execution is selectable through the LBISTCTRL1.BODY bit. This bit allows to select between tristate ( $\text{BODY}=1_{\text{B}}$ ) and a weak pull-up ( $\text{BODY}=0_{\text{B}}$ ).

**Note:** *P33.8 will be pulled up if  $\text{BODY}=0_{\text{B}}$  during LBIST execution, which is different from the behavior during cold reset where P33.8 is always in tristate.*

## 4 Application hints

### Recommendation

If bit BODY=0<sub>B</sub>, and a low level is required on P33.8 during LBIST execution, add an external pull-down (in the range of > 2 kOhm and < 4.7 kOhm) to this pin.

In order to quantify the strength of such an external pull-down, the parameter "Pull-up current" ( $I_{PUH}$ ) for the respective pin may be used as a reference, and the value for the external pull-down can be calculated accordingly.

### 4.69 [MBIST\_TC.H001] Destructive MBIST requires DSPR0 initialization

#### Description

When performing a destructive MBIST, the DSPR0 content might be corrupted (meaning it contains ECC uncorrectable errors). For this reason, after the execution of that test, the user has to completely initialize DSPR0 with ECC correct data.

However, if a power interruption happens during the execution of a destructive MBIST and DSPR0 is configured to be not initialized during startup (see "RAM Configuration" register HF\_PROCONRAM in TC3xx User's Manual), the device may hang and it will not be able to execute user code; ESR0 pin remains permanently low. From this state, it can only be unblocked by a power-off/-on cycle.

#### Recommendation

Ensure when performing a destructive MBIST of DSPR0 that this memory is under all possible conditions initialized before being used by software. This can be achieved by applying the following measures:

- Take care that all error conditions cause either a cold or warm power-on reset. Configure DSPR0 initialization by SSW after cold and warm power-on-reset (see "RAM Configuration" register HF\_PROCONRAM in TC3xx User's Manual)
- After finishing the MBIST the software can perform the RAM initialization by itself or can trigger a cold or warm power-on reset to let SSW perform the needed initialization

### 4.70 [MBIST\_TC.H002] Time for 4N non-destructive test

#### Description

Section "Usage of GANGs" in the MTU chapter of the TC3xx user manual mentions that the total time for the 4N non-destructive test is specified in the datasheet of each device.

#### Documentation update

This information can be found in the corresponding device datasheet in the table "Module Core Current Consumption" in column "Note/Test Condition" for parameter " $I_{DD}$  core dynamic current added by MBIST" (symbol  $I_{DDMBIST}$ ).

### 4.71 [MCMCAN\_AI.H001] Behavior of interrupt flags in CAN Interface (MCMCAN)

#### Description

In the corner case described below, the actual behavior of the interrupt flags of the CAN Interface (MCMCAN) differs from the expected behavior as follows.

#### Expected Behavior

When clearing an interrupt flag by software, the resulting value of the flag is expected to be zero.

## 4 Application hints

A hardware event that occurs afterwards then leads to a zero to one transition of the flag, which in turn leads to an interrupt service request.

### Actual Behavior in Corner Case

When the interrupt flag is being cleared by software in the same clock cycle as a new hardware event sets the flag again, then the hardware event wins and the flag remains set without being cleared.

As interrupt requests are generated only upon zero to one transitions of the flag, no interrupt request will be generated for this flag until the flag is successfully cleared by software later on.

**Note:** *This behavior applies to all Interrupt flags of MCMCAN, with the exception of the receive timeout event (flag NTRTRi.TE).*

### Workaround

After clearing the flag, the software shall read the flag and repeat clearing until the flag reads zero.

## 4.72 [MCMCAN\_AI.H002] Bus off recovery

### Description

**Note:** *The following text is copied from Application Note M\_CAN\_AN004 V1.1 by Robert Bosch GmbH and describes the bus off recovery handling in the MCMCAN module used in AURIX™ devices.*

The M\_CAN enters bus off state according to CAN protocol conditions. The bus off state is reported by setting PSRi.BO. Additionally, the M\_CAN sets CCCRi.INIT to stop all CAN operation.

To restart CAN operation, the application software needs to clear CCCRi.INIT.

After CCCRi.INIT is cleared, the M\_CAN's CAN state machine waits for the completion of the bus off recovery sequence according to CAN protocol (at least 128 occurrences of Bus Idle condition, which is the detection of 11 consecutive recessive bits).

In the MCMCAN chapter of the user manual the description of bus off recovery states that "Once CCCR.INIT has been cleared by the CPU, the device will then wait for 129 occurrences of Bus Idle (129 \* 11 consecutive recessive bits) before resuming normal operation. At the end of the Bus\_Off recovery sequence, the Error Management Counters will be reset". See Note in description of field LEC and footnote 1) in description of bit BO in Protocol Status Register (PSRi).

The M\_CAN uses its Receive Error Counter to count the occurrences of the Bus Idle condition. If need be, that can be monitored at ECRi.REC. Additionally, each occurrence of the Bus Idle condition is flagged by PSRi.LEC = 5 = Bit0Error, which triggers an interrupt (IRi.PEA) when IEi.PEAE is enabled.

While the bus off recovery proceeds, the CAN activity is reported as "Synchronizing", PSRi.ACT = 0 and PSRi.BO remains set. The time from resetting CCCRi.INIT to the clearing of PSRi.BO will be (in the absence of dominant bits on the CAN bus) 1420 (11 \* 129 + 1) CAN bit times plus synchronization delay between clock domains.

The M\_CAN does not receive messages while the bus off recovery proceeds.

The M\_CAN does not start transmissions while the bus off recovery proceeds. When a transmission is requested while the bus off recovery proceeds, it will be started after the recovery has completed and CAN activity entered Idle state, PSRi.ACT = 1.

When the Busoff Recovery has completed, PSRi.BO, ECRi.TEC, and ECRi.REC are cleared, and one CAN bit time later PSRi.ACT is set to Idle.

After PSRi.ACT reaches Idle, it will remain in Idle for at least one CAN bit time. The M\_CAN's CAN state machine will become receiver (PSRi.ACT = 2) when it samples a dominant bit during idle state or it will become transmitter (PSRi.ACT = 3) when it detects a pending transmission request during idle state.

## 4 Application hints

### 4.73 [MCMCAN\_TC.H001] Behavior of undefined data bytes read from Receive Buffer

#### Description

During CAN reception, the corresponding Receive Buffer in MCMCAN RAM is written only with number of bytes specified in DLC of the received frame, while the remaining bytes of the receive buffer retain the old/undefined values. Unlike MultiCAN+ in AURIX™ TC2xx devices, the additional bytes of the receive buffer are not overwritten with zeros.

#### Recommendation

When reading from the Receive Buffer of the MCMCAN RAM, the application software should ensure that only the number of bytes as specified in DLC of the received CAN frame are considered as valid data. The remaining bytes read should be ignored.

### 4.74 [MCMCAN\_TC.H006] Unintended behavior of receive timeout interrupt

#### Description

On following conditions:

1. Receive timeout feature is enabled (NTRTRi.RELOAD != 0), and
2. Received CAN frames are stored in RxFIFO 0/1 or dedicated Rx buffers, and
3. Respective New CAN frame received interrupts are disabled (i.e. bits IEi.RF0NE, IEi.RF1NE or IEi.DRXE are 0)

then an unintended receive timeout interrupt (if enabled, NTRTRi.TEIE = 1) is triggered, although a valid CAN frame is newly received and stored in the respective RxFIFO 0/1 or dedicated Rx buffers.

#### Recommendation

Enable the corresponding receive interrupt via bits IEi.RF0NE, IEi.RF1NE, or IEi.DRXE, depending on the usage of RxFIFO0/1 or dedicated Rx buffers for proper function of the receive timeout interrupt.

Example:

If RxFIFO 0 is used, set IEi.RF0NE = 1.

### 4.75 [MCMCAN\_TC.H007] Delayed time triggered transmission of frames

#### Description

The value written in the bit-field RELOAD of register NTATTRi, NTBTRi, NTCTTRi represents the reload counter value for the timer used for triggered transmission of message objects (Classical CAN or CAN FD frames).

The timer source and the prescaler value is defined in the NTCCRi register.

Once a value is written to bit-field RELOAD with bit STRT = 1 the timer starts counting. This timer counts one value more than the written value in bit-field RELOAD, then it triggers the transmission of a message object.

#### Effect

The message object transmission is delayed by one counter cycle with respect to the desired count time written in bit-field RELOAD.

#### Recommendation

In order to transmit a message object at a specific time, when using one of these registers:

- NTATTRi, NTBTRi, NTCTTRi

**4 Application hints**

set bit-field RELOAD to one value less than the calculated counter value.

**4.76 [MCMCAN\_TC.H008] Parameter “CAN Frequency” - Documentation update to symbol in Data Sheet**

**Description**

As described in chapter “Clocking System” of the AURIX™ TC3xx User’s Manual,

- $f_{MCMCANH}$  defines the frequency for the internal clocking of the MCMCAN module
- $f_{MCMCAN}$  defines the basic frequency for the MCMCAN module used for the baud rate generation

**Documentation Update**

For consistency with the description in the TC3xx User’s Manual, the symbol for parameter “CAN frequency” in table “Operating Conditions” in the Data Sheet shall be changed from “ $f_{CAN}$ ” to “ $f_{MCMCAN}$ ” as shown below:

**Table 24 Operating Conditions - CAN Frequency: symbol update**

Parameter	Symbol	Values			Unit	Note / Test Condition
		Min.	Typ.	Max.		
CAN Frequency	$f_{MCMCAN}$ SR	-	-	80	MHz	

**4.77 [MTU\_TC.H015] ALM7[0] may be triggered after cold PORST**

**Description**

During firmware start-up after cold PORST, alarm status flag AG7.SF0 (correctable SRAM error) may erroneously be set to 1, although no error occurred. This is due to a dummy read to an uninitialized SRAM by firmware.

**Note:** *No entry into any of the ETRR registers is made due to this issue.*

**Recommendation**

As alarms for correctable errors are uncritical in general, no action is required (alarm can be ignored). The application may only react on the error overflow.

In addition, to ensure that SMU alarm ALM7[0] does not correspond to a real SRAM correctable error, the user may refer to the ESM MCU\_FW\_CHECK described in the Safety Manual.

**4.78 [MTU\_TC.H016] MCI\_FAULTSTS.OPERR[2] may be triggered at power-up in case LBIST is not run**

**Description**

After power-up and before initialization by the SSW the safety flip-flops in the SSH can indicate a fault since some internal registers are not initialized. As a consequence MCI\_FAULTSTS.OPERR[2] could be set and result in an alarm.

This is not a real error. LBIST does initialize the internal registers and clears the error.

**Recommendation**

Alarms resulting from MCI\_FAULTSTS.OPERR[2] should be ignored during start-up and cleared right after execution of the SSW in case LBIST was not run.

**4 Application hints****Note/Documentation correction**

In the corresponding text at the beginning of section “SSH Safety Faults Status Register” in the MTU chapter of the TC3xx User Manual (V1.2.0 and later versions), the term MCI\_FAULTSTS.MISCERR[2] shall be replaced by MCI\_FAULTSTS.OPERR [2].

**4.79 [MTU\_TC.H019] Application reset value of register SRC\_MTUDONE different to documentation****Description**

The application reset value for the MTU Done Service Request control register SRC\_MTUDONE is defined as 00000000<sub>H</sub> in the Interrupt Router (IR) chapter of the appendix to the TC3xx user manual.

For design reasons, after reset, the value of bit SRC\_MTUDONE.SRR is 1<sub>B</sub>, although there was no MTUDONE interrupt event in the respective MTU.

**Recommendation**

Software must clear bit SRC\_MTUDONE.SRR when configuring the SRC register before enabling the service request.

**4.80 [MTU\_TC.H020] SIQ 53 - ALM7[1] unexpectedly raised after an application reset****Description**

The table "MTU, SSH and SRAMs Reset Domains" in the chapter "Memory Test Unit (MTU)" of the TC3xx user manual has an exception. For the SSH(40), system reset must be used wherever application reset applies in the table.

**Scope**

Reset domains.

**Effects**

If the system reset is not used, the alarms for SSH(40) cannot be cleared.

**Workaround**

Use the system reset instead of the application reset to clear the alarms for SSH(40).

**4.81 [NVM\_TC.H001] References to DMU\_HP\_PROCONTP – Typo in TC3xx user manual****Description**

The register name DMU\_HP\_PROCONTP mentioned in the TC3xx user manual must be replaced with DMU\_HF\_PROCONTP.

**Documentation Update**

The term DMU\_HP\_PROCONTP must be replaced by the correct register name DMU\_HF\_PROCONTP in the following parts of the NVM chapter:

- "Safety Endinit protection" ("Safety Measures" chapter)

## 4 Application hints

- "Handling Errors During Operation" (DMU chapter)
- "UCB\_OTPy\_ORIG and UCB\_OTPy\_COPY (y = 0 - 7)" (UCB chapter)

### 4.82 [OCDS\_TC.H014] Avoiding failure of key exchange command due to overwrite of COMDATA by firmware

#### Description

**Note:** *This problem is only relevant for tool development, not for application development.*

After PORST the UNIQUE\_CHIP\_ID\_32BIT is written to the COMDATA register by firmware (time point T1). Then, firmware evaluates whether a key exchange request (CMD\_KEY\_EXCHANGE) is contained inside of the COMDATA register at a time point (T2). If yes, firmware will expect the 8 further words (password) from the COMDATA. If no, firmware will write again the UNIQUE\_CHIP\_ID\_32BIT value for external tools to identify the device.

If the key exchange request cannot arrive between time points T1 and T2, firmware will skip the unlock procedure and will not unlock the device. For example, the device is locked and the external tool writes the CMD\_KEY\_EXCHANGE value to COMDATA before T1. Then, this value is overwritten by firmware at T1. After this, firmware doesn't see the CMD\_KEY\_EXCHANGE value and skips the unlock procedure. The device stays locked.

#### Recommendation

The external tool shall write the CMD\_KEY\_EXCHANGE to the COMDATA register between T1 and T2. As different derivatives and firmware configurations may have different execution time, it is recommended to poll the content of COMDATA after PORST until the UNIQUE\_CHIP\_ID\_32BIT is available. Then, the external tool shall write the CMD\_KEY\_EXCHANGE immediately. In this way, the overwrite of key exchange request by firmware can be avoided.

When LBIST is activated during startup, the execution time stays the same after the PORST triggered by LBIST. Therefore, the end of LBIST should be detected by the external tool. This can be achieved by polling the device state via JTAG/DAP. During LBIST, the debug interface is disabled and no response can be received. After LBIST, the response can be received normally. This symptom can be utilized to determine whether LBIST is done. The details are described in the section "Halt after PORST with DAP" in the OCDS chapter of the device documentation.

### 4.83 [OCDS\_TC.H015] System or Application Reset while OCDS and lockstep monitoring are enabled

#### Description

After a System or Application Reset the Lockstep Alarm ALMx[0] gets activated if all of the following conditions are met (x = index of CPU with checker core):

1. Lockstep monitoring is enabled by BMI.LSENAx = 1<sub>B</sub> for CPUx, AND
2. Debug System is enabled (CBS\_OSTATE.OEN = 1<sub>B</sub>), AND
3. a) CPUx is halted (either in boot-halt state or stopped by debugger tool or in idle mode) when reset is triggered, OR b) CPUx Performance Counters are enabled and CPUx Clock Cycle Count register CCNT is read

#### Recommendation

To avoid the unintended ALMx[0] under the conditions described above, either:

- Keep the debug system disabled. OR
- Ensure all CPUs that have lockstep monitoring enabled are out of halted state AND CPUx Performance Counters are disabled before executing a System or Application reset. OR
- Use PORST instead of a System or Application reset

## 4 Application hints

### 4.84 [OCDS\_TC.H016] Release of application reset via OJCONF may fail

#### Description

**Note:** *This problem is only relevant for tool development, not for application development.*

The OJCONF.OJC7 bit-field can be used to send an application reset request to the SCU. The tool sets the bit to request an application reset and has to clear the bit to release the request otherwise the device will remain in reset state.

If JTAG is used in the above case and the frequency of JTAG is very low, there is a risk that the tool is not able to release the application reset request. If DAP is used, there is a low risk that the first release of reset request may fail but the second will always work.

#### Recommendation

It is recommended to run JTAG above 1 MHz and execute the following instructions back to back:  
IO\_SUPERVISOR + IO\_SET\_OJCONF (release) + IO\_SUPERVISOR + IO\_SET\_OJCONF (release).  
This double releasing ensures that the reset request is released reliably.

### 4.85 [OCDS\_TC.H018] Unexpected stop of Startup Software after system or application reset

#### Description

**Note:** *This problem is only relevant for tool development, not for application development.*

As documented in the TriCore™ Architecture Manual, the settings in the Debug Status Register (DBGSR) are only cleared upon a debug or power-on reset. This may lead to unexpected behavior in the following scenario:  
If CPU0 is in HALT mode, and a system or application reset is triggered, the Startup Software (SSW) starts execution on CPU0, but it is stopped again (due to the settings in DBGSR) before the SSW has finished the boot procedure.

#### Recommendation

The tool should switch the device from HALT to RUN mode through the DBGSR register.  
Alternatively, a power-on reset may be performed instead of a system or application reset.

### 4.86 [OSC\_TC.H002] Split the external crystal mode and the external input clock mode parameters of MHz oscillator in the TC3xx datasheet

#### Description

The parameters described in the table "OSC\_XTAL" in the sub-chapter "MHz Oscillator" of the TC3xx datasheet shall be split into External Crystal Mode and External Input Clock Mode. This mixing of parameters leads to a misunderstanding in the MHz oscillator configuration and usage for a user.

#### Documentation update

The parameters described in the table "OSC\_XTAL" in the sub-chapter "MHz Oscillator" of the TC3xx datasheet shall be split into External Crystal Mode and External Input Clock Mode.

**4 Application hints**

**Table 25 OSC\_XTAL**

Parameter	Note/Test Condition
Input current at XTAL1	$V_{IN} > 0\text{ V}$ ; $V_{IN} < V_{EXT}$ ; External Input Clock Mode; measured with DC input voltage
Oscillator frequency	External Input Clock Mode selected if shaper is not bypassed
	External Crystal Mode selected
Oscillator start-up time	$20\text{ MHz} \leq f_{OSC}$ and 8 pF load capacitance; External Crystal Mode
Input voltage at XTAL1 <sup>2)</sup>	External Crystal Mode or External Input Clock Mode if shaper is not bypassed
Input amplitude (peak to peak) at XTAL1	External Crystal Mode or External Input Clock Mode if shaper is not bypassed; $f_{OSC} > 25\text{ MHz}$
	External Crystal Mode or External Input Clock Mode if shaper is not bypassed; $f_{OSC} \leq 25\text{ MHz}$
Internal load capacitor	External Crystal Mode; enabled via bit OSCCON.CAP0EN
Internal load capacitor	External Crystal Mode; enabled via bit OSCCON.CAP1EN
Internal load capacitor	External Crystal Mode; enabled via bit OSCCON.CAP2EN
Internal load capacitor	External Crystal Mode; enabled via bit OSCCON.CAP3EN
Internal load stray capacitor between XTAL1 and XTAL2	External Crystal Mode
Internal load stray capacitor between XTAL1 and ground	External Crystal Mode or External Input Clock Mode
Duty cycle at XTAL1 <sup>3)</sup>	$V_{XTAL1} = 0.5 * V_{PPX}$ ; External Input Clock Mode
Absolute RMS jitter at XTAL1 <sup>3)</sup>	10 KHz to $f_{OSC}/2$ ; External Input Clock Mode
Slew rate at XTAL1 <sup>3)</sup>	Maximum 30% difference between rising and falling slew rate; External Input Clock Mode

**4.87 [OSC\_TC.H003] XTAL1 SR parameter change for compatibility with BGT60ATR24AIP**

**Description**

Can the XTAL1 slew rate requirement can be reduced to  $> 0.1\text{ V/ns}$  without any performance limitation of the Microcontroller. Validation and simulation ensures this so

**Recommendation**

Reduce the slew rate of the XTAL1 from  $0.3\text{V/ns}$  to  $0.1\text{V/ns}$

4 Application hints

4.88 [PACKAGE\_TC.H003] Exposed pad dimensions and package outlines for QFP packages - Updates to TC33x/TC32x Data Sheet

Description

In the scope of the harmonization of the package drawings, the drawings for the TQFP packages of the TC33x/TC32x have been updated. No change of form, fit or function is implied.

The dimensions for the exposed diepads are included in the respective figures.

Furthermore, for the exposed die pads, the maximum boundary of the structural corner protrusions to be considered during system design and integration has been added.

This information shall substitute the corresponding information in the TC33x/TC32x AA-step Data Sheet V1.1.

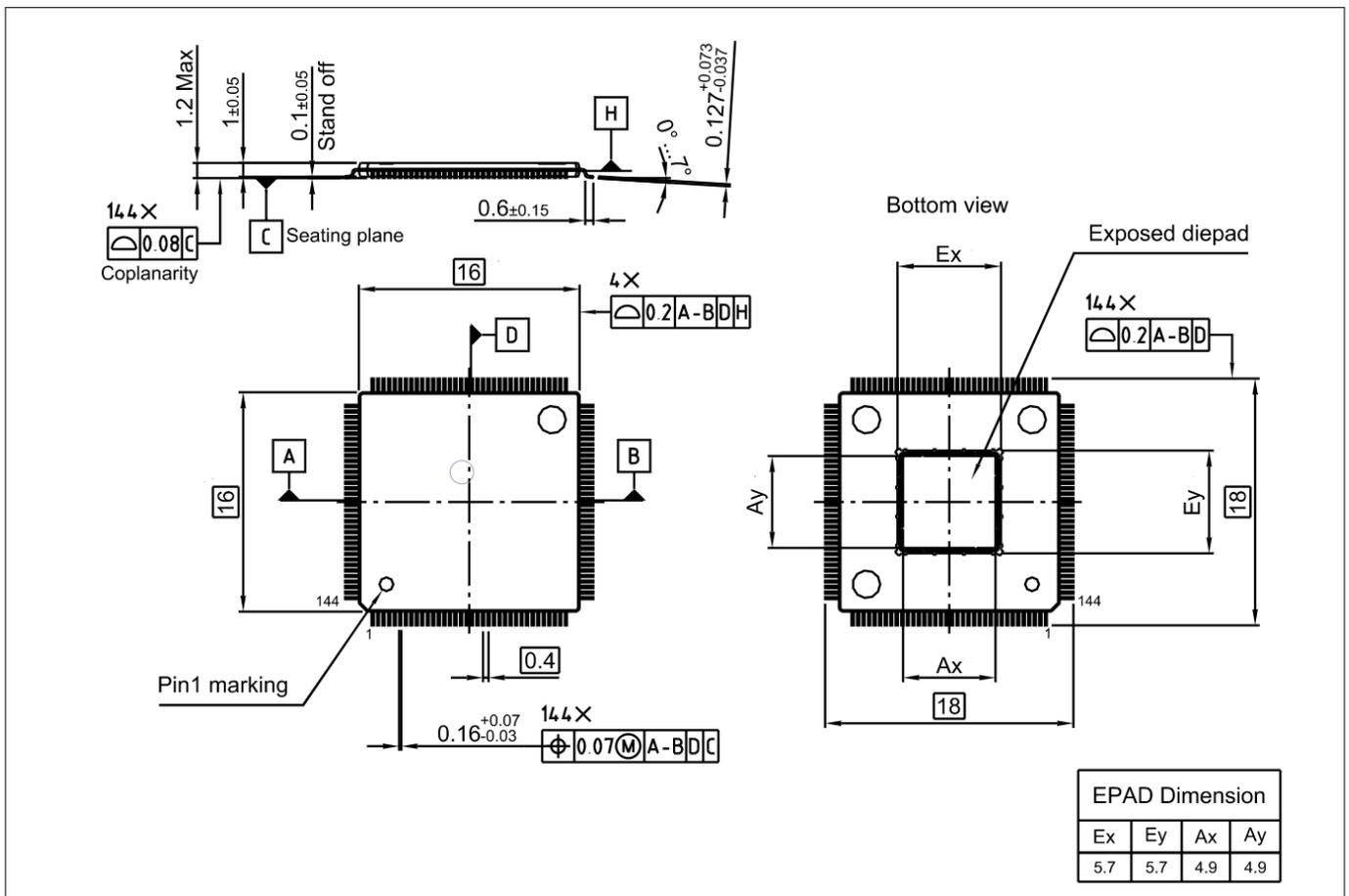
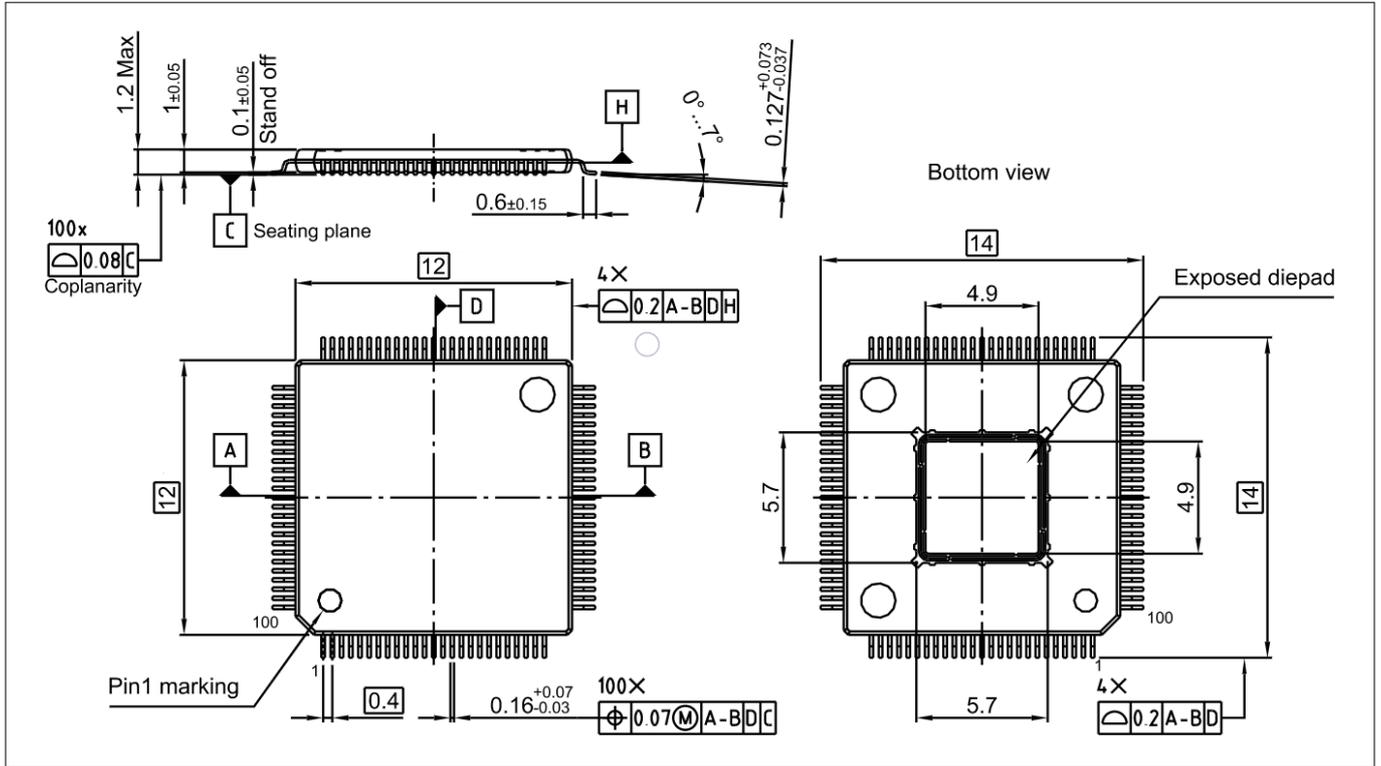


Figure 7 Package Outlines TQFP-144 for TC33x/TC32x

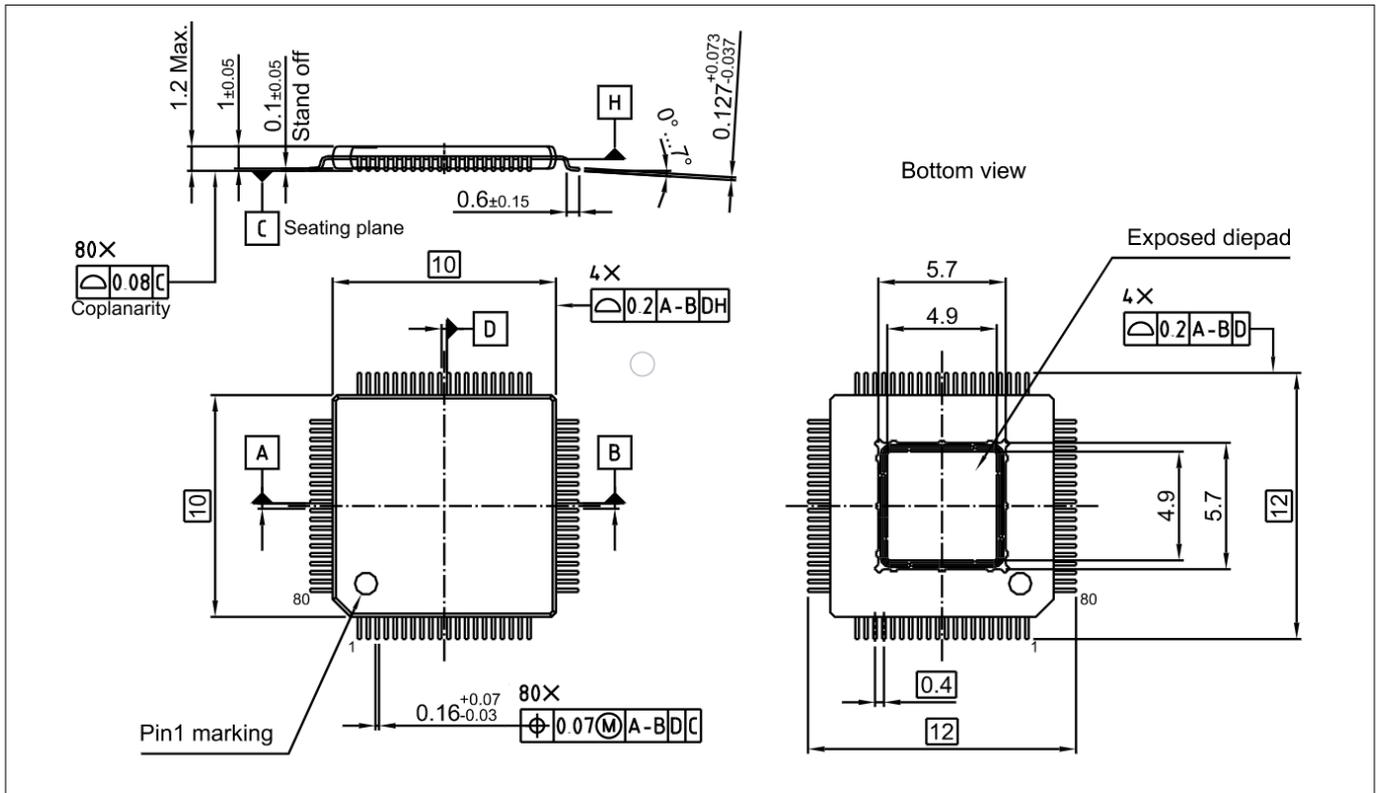
**Note:** For the exposed diepad of the TQFP-144 package of the TC33x/TC32x, structural corner protrusions have to be considered for purposes of system design and integration with a maximum boundary of 6.2 mm.

**4 Application hints**



**Figure 8 Package Outlines TQFP-100 for TC33x/TC32x**

**Note:** For the exposed diepad of the TQFP-100 package of the TC33x/TC32x, structural corner protrusions have to be considered for purposes of system design and integration with a maximum boundary of 6.3 mm.



**Figure 9 Package Outlines TQFP-80 for TC33x/TC32x**

**4 Application hints**

**Note:** For the exposed diepad of the TQFP-80 package of the TC33x/TC32x, structural corner protrusions have to be considered for purposes of system design and integration with a maximum boundary of 6.2 mm.

You can find all of our packages, sorts of packing and others on our Infineon Internet Page <http://www.infineon.com/products>

**4.89 [PADS\_TC.H007] Connection of HWCFG[6] pad in QFP-80 and QFP-100 packages – Explanation to Data Sheet history****Description**

In QFP80 and QFP100 packages, HWCFG[2] and HWCFG[6] pins are not available. Internally, the corresponding pads are handled as follows:

- HWCFG[2] is tied to 1 (via internal pull-up) to ensure EVRC is enabled;
- HWCFG[6] is tied to 0 (connected to  $V_{SS}$  via E-PAD) to ensure pins are in tristate

This is also documented in Note 2.) in figure “Hardware Configuration (HWCFG) pins” in the PMSLE chapter of the TC3xx User’s Manual in version V1.3.0 and later versions.

In V0.6 of the TC33x/TC32x Data Sheet, the E-PAD was listed as VSS pin 81 for QFP-80, pin 101 for QFP-100, and pin 145 for QFP-144 packages, respectively. The E-PAD is explicitly listed in the supply tables in TC33x/TC32x Data Sheet V0.7 and following (see also chapter “History” in corresponding Data Sheets).

**4.90 [PMSLE\_TC.H001] Typo in SHVL formula for EVRSDCTRL9 in the PMSLE chapter****Description**

For the register EVRSDCTRL9 in the PMSLE chapter of the TC3xx user manual, the formula " $V_{OUT} + SHVL \times 5 \text{ mV}$ " mentioned in the description of bit-field SHVL is incorrect.

**Documentation update**

For EVRSDCTRL9, the formula mentioned in the description of bit-field SHVL must be corrected as " $V_{OUT} - SHVL \times 5 \text{ mV}$ " instead of " $V_{OUT} + SHVL \times 5 \text{ mV}$ ".

**4.91 [PMSLE\_TC.H002] VSSM/VAGND is not connected to ePad on TQFP-100 or TQFP-80****Description**

VAGND pin in the packages TQFP-100 and TQFP-80 is tied to the VSSM pin on the package level.

In the packages TQFP-100 and TQFP-80, the analog ground pin VSSM and analog reference pin VAGND are connected together on the package level. This common ADC ground VSSM/VAGND pin is not connected to the main VSS (E-PAD) to avoid noise coupling into the ADC.

**Recommendation**

It is mandatory to connect the common ADC ground VSSM/VAGND pin through direct and nearest via to the ground plane.

## 4 Application hints

### 4.92 [PMS\_TC.H003] $V_{DDPD}$ voltage monitoring limits

#### Description

The EVR pre-regulator (EVRPR) generates the internal  $V_{DDPD}$  voltage. Its upper and lower threshold limits are monitored by the  $V_{DDPD}$  secondary monitor, while the minimum  $V_{LVD RSTC}$  voltage (LVD reset level) is monitored by the  $V_{DDPD}$  detector with built-in reference.

The secondary voltage monitor's upper and lower voltage thresholds for the  $V_{DDPD}$  channel may be adapted in software for better centering across the nominal set point with sufficient margin accounting for static regulation and dynamic response of the  $V_{DDPD}$  internal voltage regulator.

**Note:** *The PREOVVAL and PREUVAL values of  $EB_H$  and  $C7_H$ , respectively, mentioned in column "Note/Test Conditions" for  $V_{DDMON}$  in the Data Sheet are only examples used to characterize the  $V_{DDMON}$  accuracy under the specified conditions and shall not be used for the configuration of the EVROVMON2.PREOVVAL and EVRUVMON2.PREUVAL fields in an application.*

#### Recommendation

- The over-voltage alarm threshold setting in EVROVMON2.PREOVVAL needs not to be modified. The register reset value 0xFE = 1.460 V is appropriate (as well as the next lower value 0xFD = 1.454 V)
- For the under-voltage alarm threshold setting in EVRUVMON2.PREUVAL:
  - The register reset value 0xBC = 1.079 V (typical) may be kept. It matches the LVD reset level ( $V_{LVD RSTC}$ ) which is at 1.074 V (typical). In this case, the reset will occur concurrently with the alarm, therefore either the reset, or the alarm and the reset will be triggered
  - The threshold value might be set higher to the value 0xC4 = 1.125 V (typical), in order for the software to have some time to react on the alarm before the reset occurs

In V1.4.0 and newer versions of the TC3xx User's Manual, the part for the  $V_{DDPD}$  voltage monitor in figure "Voltage Monitoring - VEVR SB, VDDM & VDDPD" in the PMS and PMSLE chapter has been updated accordingly:

- PREOVVAL range = 1.43 V - 1.48 V
  - Register reset value: SMU alarm generated at PREOVVAL ~ 1.46 V
- PREUVAL range = 1.1 V - 1.15 V
  - Register reset value: SMU alarm generated at PREUVAL ~ 1.08 V

### 4.93 [PMS\_TC.H008] Interaction of interrupt and power management system - Additional information

#### Description

- **TC2xx:** The description of steps to enter Idle, Sleep and Standby Mode in chapter "Power Management Overview" of the PMC chapters in the current TC2xx User's Manuals is not comprehensive in explaining the dependency on pending interrupts as well as received interrupts. Hence, more explanation is provided here.
- **TC3xx:** The description of steps to enter Idle, Sleep and Standby Mode in chapter "Power Management Overview" of the PMS and PMSLE chapters in the current TC3xx User's Manual is not comprehensive in explaining the dependency on pending interrupts as well as received interrupts. Hence, more explanation is provided here.

For a CPU to enter Idle Mode, it must have no interrupts pending. If it is in Idle Mode it will stay in Idle Mode until one of the specified wake-up events occurs – one of these is to have a pending interrupt.

Any SRN targeting a specific CPU (i.e. TOS set to that CPU), which is enabled, i.e. has SRE set, and has received a trigger event, i.e. has SRR set (whether by a received trigger from a peripheral or a master using the SETR control bit in the SRN) is a pending interrupt. Thus, even if a peripheral is shut down by having its clocks gated off, if it has presented a trigger event to the IR, and the SRE bit for that SRN is set, there will be a pending interrupt to the specified CPU.

#### 4 Application hints

It is not necessary for the priority of the pending interrupt to allow it to be taken, nor is it necessary for the CPU to have interrupt servicing enabled. It is possible and valid for Idle Mode to be entered with interrupts disabled, and to only re-enable interrupt acceptance subsequent to resuming execution. Equally, the CPU's priority may well dictate that the interrupt cannot be serviced immediately on re-enabling interrupts.

There may be some interrupts in a system that a CPU will be required to service and must exit Idle Mode (or Sleep Mode) or prevent entry to Idle Mode (or Sleep or Standby Mode) on their arrival. If one of these interrupts is raised prior to, or just as Idle Mode, Sleep Mode or Standby Mode is requested then that mode will not be entered.

The description for the REQSLP field states

- “In Idle Mode or Sleep Mode, these bits are cleared in response to an interrupt for the CPU, or when bit 15 of the corresponding CPU Watchdog Timer register (bit WDTCPUsSR.TIM[15]) changes from 0 to 1.”

For clarity, this also means, if a write to PMCSRx.REQSLP occurs while the IR has a pending interrupt for CPUx the write data will be ignored and the REQSLP value will remain as 00<sub>B</sub> “Run Mode”.

For the system to enter Sleep or Standby Mode by writing to PMCSRx.REQSLP (as opposed through an external low voltage condition), all CPUs must be in Idle Mode. Typically, first other CPUs will be brought into Idle Mode and then the master CPU will be the last to enter to Idle Mode as a transitional state of the request for the system mode Sleep or Standby. Consequently any pending interrupts for any CPU will prevent the entry into Sleep or Standby Mode.

#### Recommendation

To ensure the transition to a power save mode, for a CPU intended to enter Idle Mode or for a system entering Sleep or Standby mode, all interrupts that are not intended to cause Run Mode to be re-entered or retained, should either have the SRE bit cleared in the respective SRN or be guaranteed to have the SRR bit clear.

- **TC2xx:** If modifying the SRE bit of an SRN, to ensure the new state is reflected in IR arbitration information conveyed to the PMC and CPUs, sufficient time for an arbitration must have elapsed. Hence, a subset of the synchronisation described in subsection “Changing the SRN configuration” of the IR chapter in the corresponding TC2xx User's Manual is required.
- **TC3xx:** If modifying the SRE bit of an SRN, to ensure the new state is reflected in IR arbitration information conveyed to the PMS and CPUs, sufficient time for an arbitration must have elapsed. Hence, a subset of the synchronisation described in subsection “Changing the SRN configuration” of the IR chapter in the TC3xx User's Manual is required.

After the last SRN (for CPUx) has been updated

- Read back the last SRN
- Read the LWSRx register

Clearing the SRR bit or disabling the source of the trigger can also be used if there are no timing hazards; i.e. no risk of a trigger being raised just before reconfiguring the peripheral (to not raise triggers), or no risk of an SRN that has had SRR cleared being set again while other SRNs are accessed. If the timing behaviour of these interrupt sources allows them to be disabled at source or in the SRN these are also valid methods. So long as the SRE bit and SRR bit are not both set, there will not be a pending interrupt. If the SRR bits are cleared, after the last SRN is modified there also needs to be a synchronisation step for the IR outputs to reflect the update before the PMCSRx is written.

Once there are no pending interrupts, request the power saving mode by writing to the respective PMCSRx.

**Note:** *TC2xx: There will still be several system clock cycles till the power saving mode is enabled by the PMC during which the CPU will continue to execute instructions.*

**Note:** *TC3xx: There will still be several system clock cycles till the power saving mode is enabled by the PMS during which the CPU will continue to execute instructions.*

To ensure a deterministic boundary for execution to end after the power saving mode request, the write to PMCSRx should be followed by a DSYNC and a WAIT instruction.

## 4 Application hints

### 4.94 [PMS\_TC.H009] Interaction of warm reset and standby mode transitions

#### Description

Chapter “Power Management” in the PMS and PMSLE chapters of the TC3xx User’s Manual in general describes how the standby mode transitions are performed from the AURIX™ system point of view (see also figure “Power down modes and transitions”).

This application hint addresses the specific use cases

- when a standby request by VEXT ramp down is issued during warm reset, or
- when a warm reset is triggered when a standby mode transition is ongoing

The PMS and PMSLE modules have a separate state machine operating independently from the rest of the AURIX™ system. The PMS and PMSLE modules and states are not affected by warm resets (for example application reset). Table “Effect of Reset on Device Functions” in the SCU chapter of the TC3xx User’s Manual shows how the AURIX™ modules are affected by different reset types. The PMS and PMSLE modules behave in the same way as the EVR module listed in this table.

Therefore standby mode entry is achieved even in the reset state of the AURIX™ system modes.

### 4.95 [PMS\_TC.H010] “EVR13” to be replaced by “EVRC” in table titles of TC33x/TC32x Data Sheet - Documentation update

#### Description

For consistency with the User’s Manual, the term “EVR13” shall be replaced by “EVRC” in the table titles of the following two tables in the EVR chapter of the TC33x/TC32x Data Sheet V1.1 (and earlier versions):

- Title of table “EVR13 SMPS” shall be replaced by “EVRC SMPS”
- Title of table “EVR13 SMPS external components” shall be replaced by “EVRC SMPS external components”

### 4.96 [PMS\_TC.H011] Supply mode and topology selection - Allowed combinations of VEXT and VDDM - Documentation update

#### Description

Tables “Allowed Combinations of Nominal External Supply Voltages between Voltage Rails” in the PMS and PMSLE chapters of the TC3xx User’s Manual define the allowed combinations of the external supply voltages for the target applications and verified use cases of the TC3xx family.

These tables do not include the combination VEXT = 5V and VDDM = 3.3V.

#### Documentation update

For consistency, in tables “Supply Mode and Topology selection” in the same chapters, for the configurations with “VEXT & VEVR SB = **5.0V** external supply” in column “Supply Pin Voltage/Level”, the term “VDDM = VAREF<sub>x</sub> = 5V **or** 3.3V ” shall be replaced by

- VDDM = VAREF<sub>x</sub> = 5V

**4 Application hints**

**4.97 [PMS\_TC.H015] Update of figures showing VCAP0 and VCAP1 pin labels to correct polarity for EVRC SC-DCDC flying capacitor**

**Description**

In the following figures included in the PMSLE chapter of the TC3xx user manual V2.0 (and earlier versions)

- Figure "VCAP behavior during start-up when EVRC regulator is used"
- Figure "EVRC Step down regulator"

there is a mismatch between the figures and the actual HW implementation: the VCAP0 and VCAP1 pin labels are inverted.

In the figures listed above, VCAP0 is shown as the higher voltage node (at the flying capacitor), whereas in actual HW and package implementation, VCAP1 is the higher voltage node.

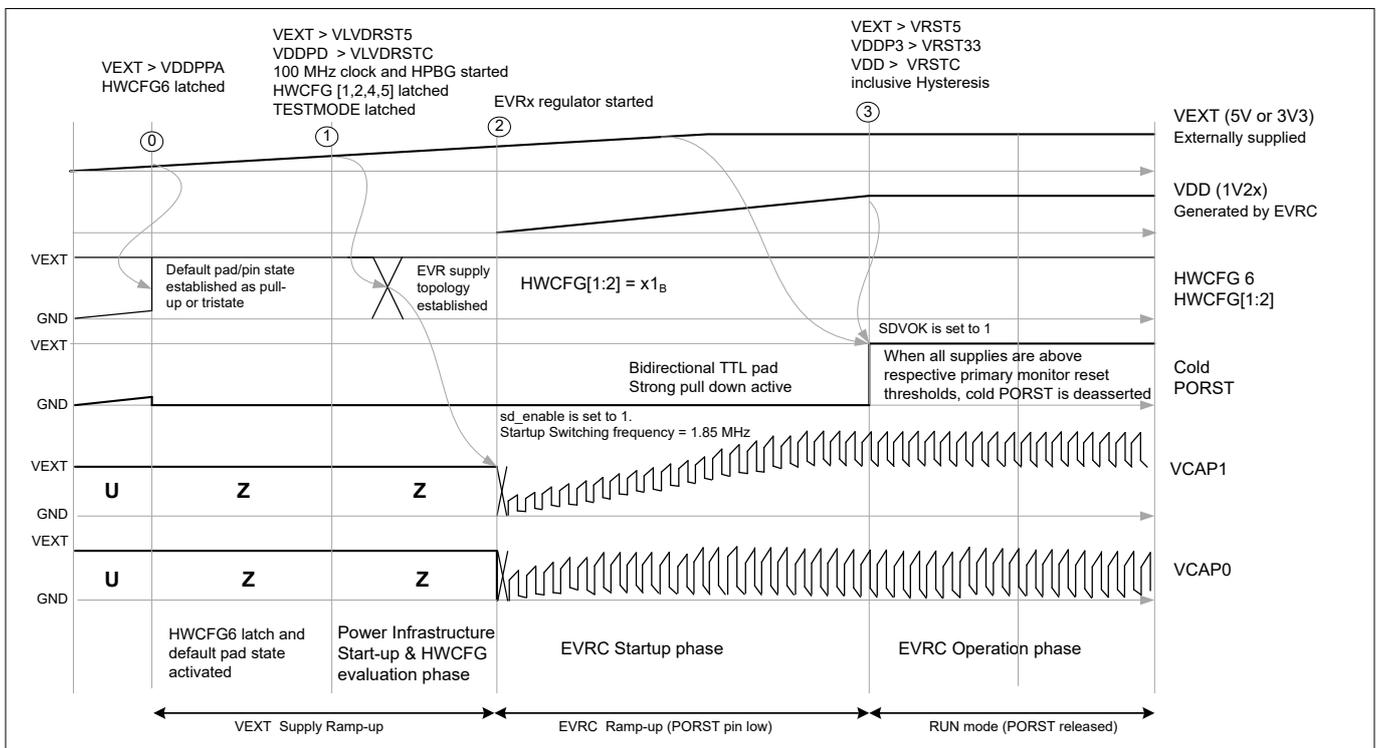
The incorrect labeling of the VCAP pins has no impact on functionality (the flying capacitor has no polarity restriction, as this is a ceramic MLCC component).

**Note:** In the following supply pin package figures in section "EVRC Supply Pins", the description is correct:

- Figure "EVRC SC-DCDC Supply Pins for QFP Packages"
- Figure "EVRC SC-DCDC Supply Pins for BGA Package (LFBGA292)"

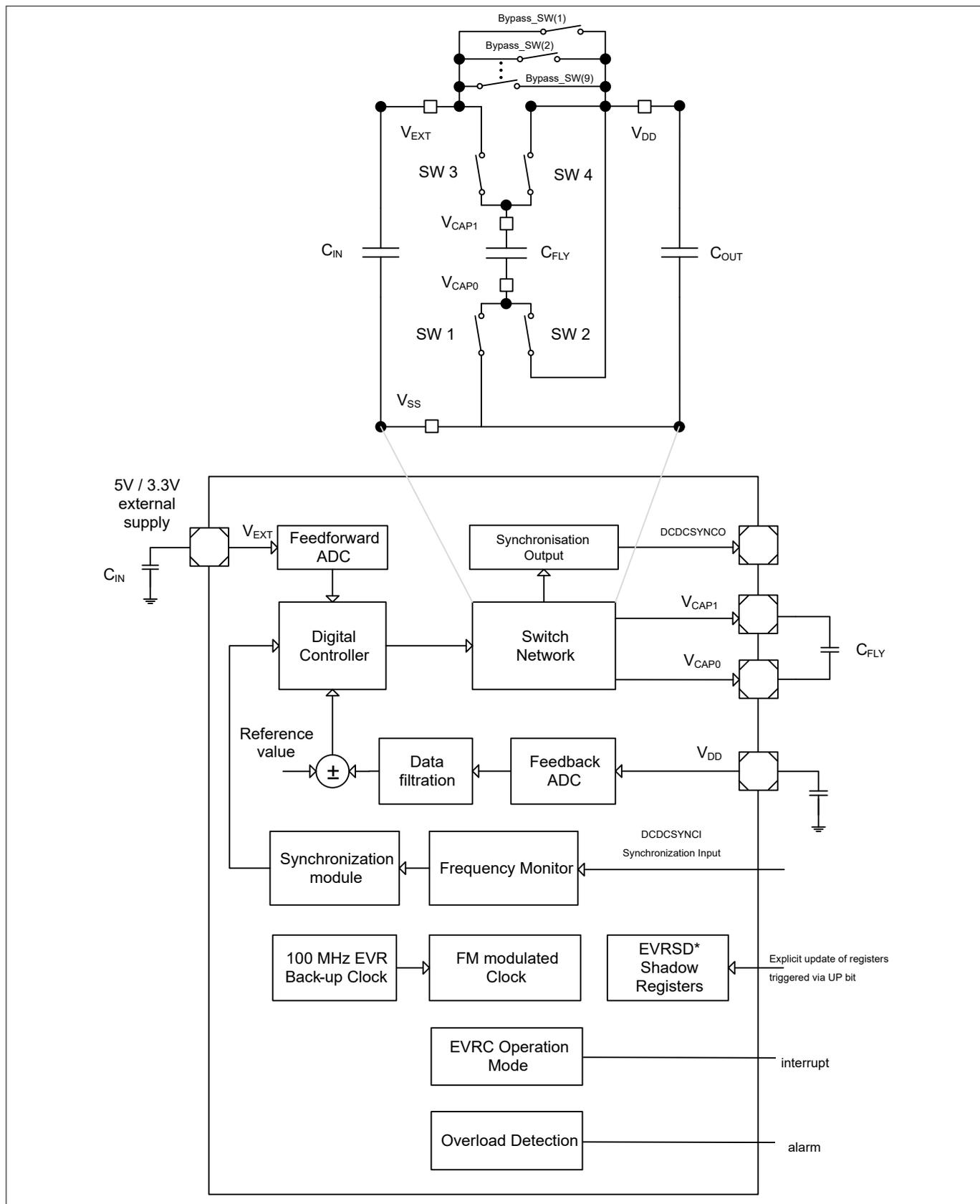
**Documentation update**

See corrected figures included below.



**Figure 10 VCAP behavior during start-up when EVRC regulator is used - corrected**

**4 Application hints**



**Figure 11** EVRC Step down regulator - corrected

## 4 Application hints

### 4.98 [PMS\_TC.H018] Bit SWDLVL in register EVRSTAT is always 1 when EVRC is OFF

#### Description

The text in the section "EVR Status Register" (EVRSTAT) in the PMS and PMSLE chapters of the TC3xx user manual states in the description of the bit SWDLVL (VEXT external supply level status):

- "This bit indicates that the VEXT voltage has dropped below ~4 V to indicate EVRC parameter switch to differentiate 5 V or 3.3 V external supply. A hysteresis of ~120 mV is implemented on this detector.
  - 0<sub>B</sub> VEXT external supply is above the threshold
  - 1<sub>B</sub> VEXT external supply is below the threshold"

The statement "0<sub>B</sub> VEXT external supply is above the threshold" is incorrect when EVRC is disabled and the VEXT voltage is above 4 V threshold.

#### Documentation update

When the EVRC is disabled by HWCFG pin and VEXT voltage is above 4 V threshold, the bit SWDLVL will be 1<sub>B</sub>, which means the bit takes the reset state as the EVRC is inactive.

### 4.99 [PMS\_TC.H019] Limitation of power-cycles - Additional datasheet footnote

#### Description

In the table "Supply Ramp" in the sub-chapter EVR of the TC3xx datasheet, a statement is described as:

- "Up to 1000000 power-cycles, matching the limits defined in the table "Supply Ramp", are allowed for TC3xx, without any restriction to reliability"

The statement with reference to the limitation of power-cycles up to 1000000 can be omitted if the device is operated within the nominal operating conditions during the device start-up phase.

#### Documentation update

In the table "Supply Ramp" in the sub-chapter EVR or PMS of the TC3xx datasheet, the following footnote shall be added to the statement "Up to 1000000 power-cycles, matching the limits defined in the table "Supply Ramp", are allowed for TC3xx, without any restriction to reliability":

- <sup>1)</sup>If the MCU supply voltages are within the nominal operating conditions as specified in the datasheet during the device start-up phase, then there is no restriction on the number of power-cycles for all TC3xx devices in the latest design steps (TC39x\_BC, TC39x\_BD, TC38x\_AE, TC3Ex\_AA, TC37x\_AA, TC37xEXT\_AB, TC35x\_AB, TC36x\_AA, TC33xEXT-AA and TC33x\_TC32x\_AA)

### 4.100 [PORTS\_TC.H018] Misleading footnote on pad driver mode selection table

#### Description

The table "Pad Driver Mode Selection for Slow Pads" in the PORTS chapter of the TC3xx user manual contains the following footnote:

1) This setting is marked "sm" as the electrical characteristics are identical to the strong driver medium edge setting. The Data Sheet contains also only common "sm" tables.
--

This footnote is inaccurate. Please read the recommendation that follows.

## 4 Application hints

### Recommendation

The electrical characteristics of the medium driver with sharp setting are explicitly documented in the datasheet. They are generally not identical to the characteristics of the strong driver with medium setting. However, the documentation of timing parameters for communication interfaces in the datasheet has "sm" tables that are valid for both types of output driver settings.

### 4.101 [QSPI\_TC.H008] Details of the baud rate and phase duration control - Documentation update

#### Description

To enhance readability, the last part of the second paragraph in the QSPI chapter "Details of the Baud Rate and Phase Duration Control", starting with "Variations in the baud rates of the slaves ..", shall be rephrased as shown below.

For further details see also the formulas in the chapter mentioned above and in the figures in chapter "Calculation of the Baud Rates and the Delays" in the User's Manual.

#### Documentation update

Variations in the baud rates of slaves of one module are supported by the ECONz.Q and the ECONz.A/B/C bit-field settings allowing for a flexible bit time variation between the channels in one module.

### 4.102 [QSPI\_TC.H011] Missing information on SLSI misplaced inactivation enable error

#### Description

Missing information for error interrupt "SLSI misplaced inactivation" in the Status Register.

#### Recommendation

The documentation will be updated as follows:

- SLSI misplaced inactivation error interrupt is raised when SLSI is deactivated by the master while the data transfer is still ongoing

### 4.103 [QSPI\_TC.H013] Additional parameter value for CMOS and LVDS pads of QSPI module in the datasheet

#### Description

In the table "Master Mode Strong Sharp (ss) output pads" in the sub-chapter "QSPI Timings, Master and Slave Mode" of TC3xx datasheet, the following information is missing for the parameter "SCLKO clock period":

- $t_{50\text{ CC}} = 20\text{ ns}$  at  $C_L = 25\text{ pF}$ . This is valid only for simplex mode

Similarly, in the table "Master Mode Timing, LVDS output pads for data and clock" in the sub-chapter "QSPI Timings, Master and Slave Mode" of TC3xx datasheet, the following information is missing for the parameter "SCLKO clock period":

- $t_{50\text{ CC}} = 50\text{ ns}$  at  $C_L = 25\text{ pF}$
- $t_{50\text{ CC}} = 20\text{ ns}$  at  $C_L = 25\text{ pF}$ . This is only valid for simplex mode

#### Scope

Maximum achievable QSPI baud rate in different modes.

**4 Application hints**

**Effects**

Duplex and half-duplex modes in QSPI are supported only upto 20 MBaud. Simplex mode is supported upto 50 MBaud.

**Documentation update**

The parameter "SCLKO clock period" described in the tables "Master Mode Strong Sharp (ss) output pads" and "Master Mode Timing, LVDS output pads for data and clock" in the sub-chapter "QSPI Timings, Master and Slave Mode" of TC3xx datasheet should be updated as follows:

**Table 26 Master Mode timing for strong sharp (ss) output pads**

Parameter	Symbol	Values			Unit	Note or test condition
		Min.	Typ.	Max.		
SCLKO clock period	$t_{50\ CC}$	20	-	-	ns	$C_L = 25\ \text{pF}$ ; only valid for simplex mode
		50	-	-	ns	$C_L = 25\ \text{pF}$

**Table 27 Master Mode timing, LVDS output pads for data and clock**

Parameter	Symbol	Values			Unit	Note or test condition
		Min.	Typ.	Max.		
SCLKO clock period	$t_{50\ CC}$	20 <sup>1)</sup>	-	-	ns	$C_L = 25\ \text{pF}$ ; only valid for simplex mode
		50 <sup>1)</sup>	-	-	ns	$C_L = 25\ \text{pF}$

**4.104 [RESET\_TC.H006] Certain registers may have different reset values than documented in TC3xx User's Manual - Documentation update**

**Description**

The following registers may show different reset values compared to those documented in the TC3xx User's Manual or TC3yx appendix. During device start-up, the initial hardware reset values of certain registers may be updated. Consequently, user software may read different values. Please refer to the table below for further details.

**Note:** *The TC3xx User's Manual chapters and/or register bit-field descriptions may contain information in addition to reset values/tables.*

**Note:** *The registers listed in the following table apply to TC39x, TC38x, TC37x, TC37xEXT, TC36x, TC35x, TC33xEXT and TC3Ex. For TC33x/32x see separate table below. Presence of CPU\*\_PCON1 registers depends on number of available CPUs.*

**4 Application hints**

**Table 28 TC39x, TC38x, TC37x, TC37xEXT, TC36x, TC35x, TC33xEXT and TC3Ex registers that may have different reset values than documented in TC3xx User's Manual**

Register	Initial reset value	Reset value defined in User's Manual	Remark
P20_IOCRO	0x0010 0000	0x0000 0000 (HWCFG6 = tri-state)	TESTMODE pin is PU (input pull-up), even with HWCFG6 = tri-state, as described in Data Sheet.
EMEM_TILECONFIG	0x0000 0000	0x5555 5555	This is a write-only ("w") register. For tile mode information do not read EMEM_TILECONFIG; instead, read EMEM_TILESTATE.
SBCU_DBCNTL	0x0000 7002	0x0000 7003	Bit EO is "Status of BCU Debug Support Enable" and only set after reset when OCDS is enabled. This bit is controlled by Cerberus.
CPU1_PCON1	0x0000 0001	0x0000 0000	Bit PCINV of PCON1 is set when CPU is in boot halt mode, it is cleared when CPU starts execution
CPU2_PCON1	0x0000 0001	0x0000 0000	
CPU3_PCON1	0x0000 0001	0x0000 0000	
CPU4_PCON1	0x0000 0001	0x0000 0000	
CPU5_PCON1	0x0000 0001	0x0000 0000	
HSSL0_MFLAGS	0xA000 0000	0x8000 0000	Bit TEI indicates the state of CTS (Clear To Send) signal from HSCT module. The default state of this bit is 1.
HF_OPERATION	0x0000 0X00	0x0000 0000	RES bits shall be ignored.
PMS_EVRSDCTRL0	0x3039 0001	0xF039 0001	LCK and UP bits are cleared.
PMS_EVRSDCTRL1	0x0669 0708	0x8669 0708	LCK bit is cleared.
PMS_EVRSDCTRL6	0x0023 1C94	0x8023 1C94	LCK bit is cleared.
PMS_EVRSDCTRL7	0x0000 00FE	0x8000 00FE	LCK bit is cleared.
PMS_EVRSDCTRL8	0x1121 048E	0x9121 048E	LCK bit is cleared.
PMS_EVRSDCTRL9	0x0000 0434	0x8000 0434	LCK bit is cleared.
PMS_EVRSDCTRL11	0x1207 0909	0x9207 0909	LCK bit is cleared.
PMS_EVRSDCOEFF0	0x3508 73B6	0xB508 73B6	LCK bit is cleared.
PMS_EVRSDCOEFF1	0x2294 6C46	0xA294 6C46	LCK bit is cleared.
PMS_EVRSDCOEFF6	0x0097 1802	0x8097 1802	LCK bit is cleared.
PMS_EVRSDCOEFF7	0x0000 D8F7	0x8000 D8F7	LCK bit is cleared.
PMS_EVRSDCOEFF8	0x0017 1002	0x8017 1002	LCK bit is cleared.
PMS_EVRSDCOEFF9	0x0000 A0AF	0x8000 A0AF	LCK bit is cleared.
SCU_OSCCON	0x0000 0258 for UCB_DFLASH.OSCCFG = 0; 0x0XX0 XXXX otherwise	0x0000 0X1X	SCU_OSCCFG setting is recovered from UCB_DFLASH

**Note:** The registers listed in the following table apply to TC33x/TC32x.

**4 Application hints**

**Table 29 TC33x/32x registers that may have different reset values than documented in TC3xx User's Manual**

Register	Initial reset value	Reset value defined in User's Manual	Remark
P20_IOCRO	0x0010 0000	0x0000 0000 (HWCFG6 = tri-state)	TESTMODE pin is PU (input pull-up), even with HWCFG6 = tri-state, as described in Data Sheet.
SBCU_DBCNTL	0x0000 7002	0x0000 7003	Bit EO is "Status of BCU Debug Support Enable" and only set after reset when OCDS is enabled. This bit is controlled by Cerberus.
HF_OPERATION	0x0000 0X00	0x0000 0000	RES bits shall be ignored.
PMS_EVRSDCTRL0	0x0000 0003	0xC000 0003	LCK and UP bits are cleared.
PMS_EVRSDCTRL1	0x0012 0012	0x8012 0012	LCK bit is cleared.
PMS_EVRSDCTRL2	0x140A 0909	0x940A 0909	LCK bit is cleared.
PMS_EVRSDCTRL3	0x0000 0001	0x8000 0001	LCK bit is cleared.
PMS_EVRSDCTRL4	0x2000 2209	0xA000 2209	LCK bit is cleared.
PMS_EVRSDCTRL5	0x001B 7566	0x801B 7566	LCK bit is cleared.
PMS_EVRSDCTRL6	0x0081 0000	0x8081 0000	LCK bit is cleared.
PMS_EVRSDCTRL7	0x2061 0400	0xA061 0400	LCK bit is cleared.
PMS_EVRSDCTRL8	0x1070 0000	0x9070 0000	LCK bit is cleared.
PMS_EVRSDCTRL9	0x0000 4040	0x8000 4040	LCK bit is cleared.
PMS_EVRSDCTRL10	0x130C 0719	0x930C 0719	LCK bit is cleared.
PMS_EVRSDCOEFF0	0x0052 0083	0x8052 0083	LCK bit is cleared.
PMS_EVRSDCOEFF1	0x17B2 6996	0x97B2 6996	LCK bit is cleared.
PMS_EVRSDCOEFF2	0x4924 8BD9	0xC924 8BD9	LCK bit is cleared.
PMS_EVRSDCOEFF3	0x0780 00A3	0x8780 00A3	LCK bit is cleared.
SCU_OSCCON	0x0000 0258 for UCB_DFLASH.OSCCFG = 0; 0x0XX0 XXXX otherwise	0x0000 0X1X	SCU_OSCCFG setting is recovered from UCB_DFLASH

**4.105 [RESET\_TC.H007] Cold Power on Reset Boot Time – Additional information**

**Description**

Parameter "Cold Power on Reset Boot Time" (symbol  $t_{BP}$ ) in chapter "Reset Timing" of the Data Sheet is specified as a "Max." value.

**Additional information**

The term "Firmware execution time" used in column "Note/Test Condition" for this parameter includes the execution time of both Startup Software (SSW) and Checker Software (CHSW).

## **4 Application hints**

### **4.106 [SAFETY\_TC.H013] ESM[SW]:SYS:MCU\_FW\_CHECK - Access to MC40 FAULTSTS register – Additional information**

#### **Description**

The FSI RAM is used to configure the PFLASH. For security related reason, the access to this RAM is restricted. Therefore, in order to avoid accesses to this RAM through its SSH, the MBIST Controller 40 (MC40) is not disclosed in the AURIX™ TC3xx User's Manual.

However, according to Appendix A of the Safety Manual, for SSH(40) register MC40\_FAULTSTS must be compared to an expected value by ESM[SW]:SYS:MCU\_FW\_CHECK after reset.

#### **Recommendation**

When implementing ESM[SW]:SYS:MCU\_FW\_CHECK, the register address listed below has to be used to access the FAULTSTS register of MBIST Controller 40:

- MC40\_FAULTSTS (0xF006 38F0)

### **4.107 [SAFETY\_TC.H017] Safety Mechanisms requiring initialization - Documentation update**

#### **Description**

In chapter “Safety Mechanisms” of the AURIX™ TC3xx Safety Manual, safety mechanisms that need to be initialized by Application SW have a link in the “Init Conditions” field to a Safety Mechanism Configuration (SMC). This SMC provides a description of what has to be implemented to activate the respective safety mechanism (SM).

This is not valid for all safety mechanisms. Some of them have no SMC as “Init Conditions”, although they need to be activated.

#### **Documentation update**

Following is the list of safety mechanisms that have to be activated with the respective “Init Conditions”, in addition to the SMs that are already listed with a link to a SMC in field “Init Conditions” in the Safety Manual:

#### **SM[HW]:CLOCK:ALIVE\_MONITOR**

##### **Init conditions**

The application SW shall enable the clock alive monitoring by setting the corresponding bit in CCUCON3 register after PLLs have been set up and are running.

#### **SM[HW]:CPU:TPS**

##### **Init conditions**

The application SW shall enable the temporal protection system (configure CPU\_SYSCON.TPROTEN = 1<sub>B</sub>).

#### **SM[HW]:CPU:TPS\_EXCEPTION\_TIME\_MONITOR**

##### **Init conditions**

The application SW shall enable the temporal protection system (configure CPU\_SYSCON.TPROTEN = 1<sub>B</sub>).

#### **SM[HW]:CPU:CODE\_MPU**

##### **Init conditions**

The application SW shall configure the Code MPU according to TriCore™ TC1.6.2 Core Architecture Manual Volume 1 V1.2.2 - Chapter 10 “Memory Protection System”.

---

#### **4 Application hints**

##### **SM[HW]:CPU:DATA\_MPU**

###### **Init conditions**

The application SW shall configure the Data MPU according to TriCore™ TC1.6.2 Core Architecture Manual Volume 1 V1.2.2 - Chapter 10 “Memory Protection System”.

##### **SM[HW]:CPU:UM0**

###### **Init conditions**

The application SW shall configure the CPU access privilege level control to User-0 Mode (CPU\_PSW.IO = 00<sub>B</sub>).

##### **SM[HW]:CPU:UM1**

###### **Init conditions**

The application SW shall configure the CPU access privilege level control to User-1 Mode (CPU\_PSW.IO = 01<sub>B</sub>).

##### **SM[HW]:CPU:SV**

###### **Init conditions**

The application SW shall configure the CPU access privilege level control to Supervisor Mode (CPU\_PSW.IO = 10<sub>B</sub>). (Default configuration).

##### **SM[HW]:CPU:STI**

###### **Init conditions**

The application SW shall configure the safe task identifier (CPU\_PSW.S = 1<sub>B</sub>).

##### **SM[HW]:DMA:TIMESTAMP**

###### **Init conditions**

The application SW shall enable the appendage of a DMA timestamp (configure DMA\_ADICRc.STAMP = 1<sub>B</sub>).

##### **SM[HW]:EMEM:CONTROL\_REDUNDANCY**

###### **Init conditions**

The application SW shall enable the EMEM module SRI control redundancy logic (EMEM\_SCTRL.LSEN = 10<sub>B</sub>).

##### **SM[HW]:EMEM:READ\_WRITE\_MUX**

###### **Init conditions**

The application SW shall configure the mode of the EMEM tiles via EMEM\_TILECONFIG and enable access to the EMEM tiles via EMEM\_TILEECC and EMEM\_TILELECT.

##### **SM[HW]:LMU:CONTROL\_REDUNDANCY**

###### **Init conditions**

The application SW shall enable the LMU control redundancy logic (LMU\_SCTRL.LSEN = 10<sub>B</sub>).

##### **SM[HW]:NVM.PFLASH:ERROR\_CORRECTION**

###### **Init conditions**

The application SW shall enable the ECC error correction (CPUx\_FLASHCON2.ECCCORDIS = 10<sub>B</sub>).  
Enabled after reset.

##### **SM[HW]:NVM.PFLASH:ERROR\_MANAGEMENT**

###### **Init conditions**

## 4 Application hints

The application SW shall enable address buffer recording (CPU<sub>x</sub>\_FLASHCON2.RECDIS = 10<sub>B</sub>).  
Enabled after reset.

### **SM[HW]:NVM.PFLASH:FLASHCON\_MONITOR**

#### **Init conditions**

The application SW shall initialize CPU<sub>x</sub>\_FLASHCON2.

### **SM[HW]:SPU:REDUNDANCY\_SCC**

#### **Init conditions**

SM[HW]:SPU:REDUNDANCY\_SCC is enabled when either of SM[HW]:SPU:PARTIAL\_REDUNDANCY or SM[HW]:SPU:REDUNDANCY are enabled.

### **SM[HW]:SCU:EMERGENCY\_STOP**

#### **Init conditions**

By default after reset, the Synchronous mode is selected; in this mode, the application SW shall enable (via EMSR.ENON = 1<sub>B</sub>) the setting of the Emergency stop flag (EMSR.EMSF) on an inactive-to-active level transition of the port input.

Alternatively, the application SW can:

- Select the Asynchronous mode (EMSR.MODE = 1); in this mode the occurrence of an active level at the port input immediately activates the emergency stop signal
- Configure Alarm(s) in SMU to trigger an Emergency Stop

### **SM[HW]:SMU:RT**

#### **Init conditions**

The application SW shall enable the Recovery Timers (RT<sub>y</sub>, where y = 0,1) via RTC.RTyE = 1<sub>B</sub>.

Recovery Timers (RT<sub>y</sub>, where y = 0,1) are enabled after Application Reset to service the WDT timeout alarms.

### **SM[HW]:SMU:FSP\_MONITOR**

#### **Init conditions**

FSP Monitor is enabled after Power-on Reset. The application SW shall ensure that the FSP is in the Fault Free State (SMU\_ReleaseFSP() ) before entering the RUN state with the SMU\_Start() command.

### **SM[HW]:PMS:VDDM\_MONITOR**

#### **Init conditions**

SMC[SW]:PMS:Vx\_MONITOR\_CFG

## **4.108 [SAFETY\_TC.H019] SM[HW]:NVM.FSIRAM:REG\_MONITOR\_TEST should not be considered**

### **Description**

The SM[HW]:NVM.FSIRAM:REG\_MONITOR\_TEST is defined in the AURIX™ TC3xx Safety Manual as a self-test mechanism for the FSI.RAM SFFs.

### **Recommendation**

The system integrator has to consider the FSI.RAM SFFs as not testable, as in fact there are no means to trigger this test.

## 4 Application hints

**Note:** *SM[HW]:NVM.FSIRAM:REG\_MONITOR\_TEST is not part of the FMEDA and will not impact the FIT rate.*

### 4.109 [SAFETY\_TC.H020] Test of SM[HW]:VMT:REG\_MONITOR is missing - Documentation update

#### Description

The “Tests” field of SM[HW]:VMT:REG\_MONITOR (section 6.500 in AURIX™ TC3xx Safety Manual v1.04 and higher versions) is empty. Users may think that the safety flip-flops mechanism is not testable, which is not true.

#### Documentation update

SM[HW]:VMT:REG\_MONITOR is testable through SMU by ESM[SW]:SMU:REG\_MONITOR\_TEST.

### 4.110 [SCR\_TC.H009] RAM ECC Alarms in Standby Mode

#### Description

During Standby mode, every ECC error in the RAMs of the Standby Controller (SCR) can be detected but the respective alarm signal is not propagated and not triggered by the SMU (ALM6[19], ALM6[20] and ALM6[21]).

**Note:** *If not in Standby mode, alarm signals for ECC errors from the SCR RAMs are propagated and triggered by the SMU.*

#### Recommendation

ECC errors from the RAMs of SCR can be checked by the application software via bit SCRECC of PMS register PMSWCR2 (Standby and Wake-up Control Register).

### 4.111 [SCR\_TC.H010] HRESET command erroneously sets RRF flag

#### Description

**Note:** *This problem is only relevant for tool development, not for application development.*

The HRESET command (to reset the SCR including its OCDS) erroneously sets the RRF flag (which signals received data to the FW).

#### Recommendation

With the following three additional commands (1-3) after an HRESET, the issues with the HRESET command can be solved:

- Execute HRESET
  1. Execute HSTATE to remove reset bit from shift register
  2. Perform JTAG tool reset to remove flag RRF (receive register flag)
  3. Execute HCOMRST to remove flag TRF (transmit register flag)

### 4.112 [SCR\_TC.H011] Hang-up when warm PORST is activated during Debug Monitor Mode

#### Description

**Note:** *This problem is only relevant for debugging.*

## 4 Application hints

When a debugger is connected and the device is in Monitor Mode (MMODE), the activation of a warm PORST will result in a hang-up of the SCR controller.

### Recommendation

Perform an LVD reset (power off/on) to terminate this situation.

### 4.113 [SCR\_TC.H012] Reaction in case of XRAM ECC Error

#### Description

When the double-bit ECC reset is enabled via bit ECCRSTEN in register SCR\_RSTCON, and a RAM double-bit ECC error is detected, bit RSTST.ECCRST in register SCR\_RSTST is set, but no reset is performed.

#### Recommendation

The reset of the SCR module in case of a double-bit ECC error must be performed via software.

The following steps need to be done:

- Enable the double-bit ECC reset by setting bit ECCRSTEN in register SCR\_RSTCON to 1<sub>B</sub>
- Enable the RAM ECC Error for NMI generation by setting bit NMIRAMECC in register SCR\_NMICON to 1<sub>B</sub>

When a RAM double-bit ECC error is detected, an NMI to the TriCore™ is generated, and bit RSTST.ECCRST in register SCR\_RSTST is set.

The TriCore™ software first has to check the cause of the NMI wakeup by checking register SCR\_RSTST. If bit ECCRST is set, a double-bit ECC error has occurred. In this case, do the following steps:

- Fill the XRAM memory with 0
- Check whether an ECC error has occurred
- If no ECC error has occurred after filling the XRAM with 0, then:
  - Reload the contents of the XRAM
  - Perform a reset of the SCR module: Set bit SCRSTREQ in register PMSWCR4 to 1<sub>B</sub>

### 4.114 [SCR\_TC.H014] Details on WDT pre-warning period

#### Description

The pre-warning interrupt request (FNMIWDT) of the SCR Watchdog Timer (WDT) means that a WDT overflow has just occurred, and in 32 cycles of the SCR WDT clock there will be a reaction to this overflow – a reset of the SCR.

After this pre-warning interrupt it is not possible to stop the WDT, as it has already overflowed, and it is not possible to stop this reaction (reset).

### 4.115 [SCR\_TC.H016] SCR current consumption in IDLE mode and 70 kHz clock

#### Description

The data sheet specifies SCR current values for the different modes like STANDBY and IDLE. Additionally, the SCR CPU clock speed can be configured and is another parameter for the variance of the module current consumption.

The value for  $I_{SCRIDLE}$  specified in the data sheet is based on the real power pattern and covers the use case  $f_{SYS\_SCR} = 20$  MHz,  $T_J = 150^\circ\text{C}$  and SCR CPU in IDLE mode.

**4 Application hints**

**Documentation update**

For the use case with  $f_{\text{SYS\_SCR}} = 70 \text{ kHz}$ ,  $T_J = 25^\circ\text{C}$  and SCR CPU in IDLE mode, we estimate  $I_{\text{SCRIDLE}_70}$  CC of 90  $\mu\text{A}$ .

**4.116 [SCU\_TC.H020] Digital filter on ESRx pins - Documentation update**

**Description**

As described in the SCU and PMS chapters of the TC3xx User’s Manual, the input signals  $\overline{\text{ESR0}}$  /  $\overline{\text{ESR1}}$  can be filtered. The filter for  $\overline{\text{ESRx}}$  is enabled via bit PMSWCR0.ESRxDEN = 1<sub>B</sub> (default after reset).

If the digital filter is enabled then pulses less than 30 ns will not result in a trigger.

For pulses longer than 100 ns, the following dependency on  $f_{\text{SPB}}$  should be noted:

**Note:** *Pulses longer than 100 ns will always result in a trigger for  $f_{\text{SPB}} \geq 20 \text{ MHz}$  in RUN mode.*

**4.117 [SCU\_TC.H021] LBIST execution affected by TCK/DAPO state**

**Description**

The TCK/DAPO pad includes an internal pull down (marked “PD2” in column “Buffer Type” in table “System I/O of the Data Sheet”).

If TCK/DAPO is pulled up by an external device, LBIST execution will be stalled.

**Recommendation**

TCK/DAPO pad shall be left open or pulled down if no tool is connected.

**4.118 [SCU\_TC.H023] Behavior of bit RSTSTAT.PORST after wake-up from standby mode**

**Description**

After cold-power on (power up from no power supply), bit RSTSTAT.PORST is always set independent of PORST pad level (pulled high or low by user).

After wake-up from standby, bit RSTSTAT.PORST indicates if the PORST pad was asserted after the wake-up trigger.

**Recommendation**

If the user expects that bit RSTSTAT.PORST is always set after wake-up from standby, the PORST pad should be kept low externally until all supplies are in operating condition.

**4.119 [SCU\_TC.H025] Field EEA in register CHIPID - Additional information**

**Description**

In the SCU chapter of the TC3xx User’s Manual, field EEA in register CHIPID is described as follows:

**Table 30 Field EEA in register CHIPID**

Field	Bits	Type	Description
EEA	16	rh	Emulation or ADAS Extension Available

(table continues...)

**4 Application hints**

**Table 30 (continued) Field EEA in register CHIPID**

Field	Bits	Type	Description
			Indicates if the emulation or ADAS extension hardware is available or not. $0_B$ EEC is not available (SAK-TC3xxxU or SAK-TC3xxxP) $1_B$ EEC is available (SAK-TC3xxxE or SAK-TC3xxxF)

The product names/feature packages (SAK-TC3xxxU, ...) mentioned in this description shall only be understood as examples; they may not exist for each TC3yx device variant.

**Recommendation**

For a summary of the functionality available in each TC3yx device variant see the TC3yx Data Sheet Addendum. As can be seen from the Chip ID values defined in this document, bit EEA =  $1_B$  in TC39x, TC37xEXT, TC35x and TC33xEXT devices, and bit EEA =  $0_B$  in TC3Ex, TC38x, TC37x, TC36x and TC33x/TC32x devices.

For a summary of the functionality of TC3xx emulation devices and their Chip ID values see the TC3xx\_ED Data Sheet.

**4.120 [SCU\_TC.H026] Unexpected alarm ALM0[1] during warm reset**

**Description**

For any warm reset, the shutdown request handler described in section “Shutdown request handler” in the Firmware chapter of the TC3xx User’s manual requires access to a specific region in the DSPR of CPU0.

For the following configuration

- access to a specific region of CPU0 DSPR (see recommendation below) is disabled for one or all n of the implemented CPUs (CPUx, x = 0..n-1) in registers SPR\_SPROT\_RGNACCENAi\_W and SPR\_SPROT\_RGNACCENAi\_R, which means the access enable bits for the corresponding master TAG IDs are ‘0’

an unexpected alarm ALM0[1] (CPU0 Bus-level MPU violation/Access Protection violation) will occur when an application reset, system reset or warm PORST is requested, and the corresponding flag DF1 in register SMU\_AD0 will remain set after the reset if it was a system or application reset.

**Recommendation**

To avoid these effects, enable read and write access for all available CPUs in the address range 0x70000200 - 0x700003EF in registers SPR\_SPROT\_RGNACCENAi\_W and SPR\_SPROT\_RGNACCENAi\_R.

**4.121 [SCU\_TC.H027] Bit field INP0 and INP1 in register EICRi - Documentation correction**

**Description**

In the SCU chapter of the current user manual, for settings INP0 =  $100_B$  to  $111_B$  and INP1 =  $100_B$  to  $111_B$  in the description of register EICRi, the last index y of signal TRxy is erroneously shown a 0.

In the description for INP0, the enable bit is erroneously referenced as EIEN(2i) instead of EICRi.EIEN0, and as EIEN(2i+1) instead of EICRi.EIEN1 in the description for INP1.

**Documentation correction**

The last index y of signal TRxy shall be identical to the OGUy index. The corrected description for INP0 and settings INP0 =  $100_B$  to  $111_B$  and for INP1 and settings INP1 =  $100_B$  to  $111_B$  is shown in the following table.

**4 Application hints**

**Table 31**                    **Field INP0 and INP1 in register EICRi (i=0-3) - Correction**

Field	Bits	Type	Description
INP0	14:12	rw	<b>Input Node Pointer</b>
			This bit-field determines the destination (output channel) for trigger event (2i) (if enabled by <b>EICRi.EIEN0</b> ).
			100 <sub>B</sub> An event from input ETL 2i triggers output OGU4 (signal TR(2i) <b>4</b> )
			101 <sub>B</sub> An event from input ETL 2i triggers output OGU5 (signal TR(2i) <b>5</b> )
			110 <sub>B</sub> An event from input ETL 2i triggers output OGU6 (signal TR(2i) <b>6</b> )
			111 <sub>B</sub> An event from input ETL 2i triggers output OGU7 (signal TR(2i) <b>7</b> )
INP1	30:28	rw	<b>Input Node Pointer</b>
			This bit-field determines the destination (output channel) for trigger event (2i+1) (if enabled by <b>EICRi.EIEN1</b> ).
			100 <sub>B</sub> An event from input ETL 2i+1 triggers output OGU4 (signal TR(2i+1) <b>4</b> )
			101 <sub>B</sub> An event from input ETL 2i+1 triggers output OGU5 (signal TR(2i+1) <b>5</b> )
			110 <sub>B</sub> An event from input ETL 2i+1 triggers output OGU6 (signal TR(2i+1) <b>6</b> )
			111 <sub>B</sub> An event from input ETL 2i+1 triggers output OGU7 (signal TR(2i+1) <b>7</b> )

**Note:**                    *In the table above, only rows that include corrections are shown.*

**4.122**                    **[SCU\_TC.H028] ERU configuration changes may lead to ERU reactions**

**Description**

The External Request Unit (ERU) may react on changes of control registers even if there is no edge at its inputs. For example, if one of the inputs of an input channel x is '1' and this is switched to another input of this channel (by EICRy.EXISz) that is '0', then ERU recognizes an edge if configured for this input channel x and the corresponding EIFR.INTFx is set and the trigger is propagated to the ERU output as configured.

**Recommendation**

Clear EIFR.INTFx bits after (re-)configuration.

If an ERU reaction is to be suppressed on configuration changes (and you suspect there might be two different levels at the two ERU inputs to be switched), then:

- Clear bits EICRy.RENz, EICRy.FENz without changing EICRy.EXISz (so potential edges are swallowed at the 'Detect Event (edge)' block)
- With a 2<sup>nd</sup> write access to EICRy set bits EICRy.EXISz as needed without changing the EICRy.RENz, EICRy.FENz
- Wait long enough  
The wait time depends on the ERU input filter setting  
In case the filter is active, the 3<sup>rd</sup> access to EICRy has to happen after EIFILT.DEPTH \* (EIFILT.FILTDIV + 1) SPB (100 MHz) clock cycles, otherwise the edge is still traveling through the filter and has not arrived at the 'Detect Event (edge)' block yet, to be swallowed as intended
- Then with a 3<sup>rd</sup> write access set EICRy.RENz, EICRy.FENz as needed without changing the EICRy.EXISz

**4 Application hints**

**4.123 [SENT\_TC.H006] Parameter  $V_{ILD}$  on pads used as SENT inputs**

**Description**

Some port pins may have restrictions when used as SENT inputs, depending on the number of active neighbor pins (on the pad frame) and their output driver setting.

In the implementation of the SENT module and product integration within Infineon Technologies products there are never negative values for  $V_{ILD}$ , so  $V_{ILDmin}$  is 0 mV. Considering the same tolerance as the SENT standard  $V_{ILDmax}$  is 100 mV.

**Note:** All SENT port pins not listed in the tables below have no restrictions on their application usage as SENT inputs.

**Table 32 SENT input pads and considered neighbors**

Device	Considered left neighbors		SENT input		Considered right neighbors	
			Pad	Channel		
<b>TC39x</b>	P15.1	P15.3	P15.4	11D	P15.6	P20.9
	P31.6	P31.7	P31.8	20C	P31.9	P31.10
	P31.7	P31.8	P31.9	21C	P31.10	P31.11
	P31.8	P31.9	P31.10	22C	P31.11	P31.14
	P31.9	P31.10	P31.11	23C	P31.14	P31.12
	P31.11	P31.14	P31.12	24C	P31.13	P31.15
<b>TC38x</b>	P15.0	P15.2	P15.4	11D	P15.1	P15.3
	P31.6	P31.7	P31.8	20C	P31.10	P31.9
	P31.8	P31.10	P31.9	21C	P31.12	P31.11
	P31.7	P31.8	P31.10	22C	P31.9	P31.12
	P02.13	P02.11	P02.12	23B	P02.4	P02.15
	P31.9	P31.12	P31.11	23C	P31.14	P31.13
	P31.10	P31.9	P31.12	24C	P31.11	P31.14
<b>TC3Ex</b>	P02.6	P02.7	P02.8	0C	P02.9	P02.10
	P02.4	P02.6	P02.7	1C	P02.8	P02.9
	P02.11	P02.4	P02.6	2C	P02.7	P02.8
	P15.0	P15.2	P15.4	11D	P15.1	P15.3
	P02.3	P02.11	P02.4	12B	P02.6	P02.7
	P02.1	P02.5	P02.3	13B	P02.11	P02.4
	P15.1	P15.3	P14.0	17D	P15.6	P15.7

**(table continues...)**

4 Application hints

Table 32 (continued) SENT input pads and considered neighbors

Device	Considered left neighbors		SENT input		Considered right neighbors	
			Pad	Channel		
TC37x and TC37xEXT	P02.6	P02.7	P02.8	0C	P02.9	P02.10
	P00.0	P00.1	P00.2	1B	P00.3	P00.4
	P02.5	P02.6	P02.7	1C	P02.8	P02.9
	P02.4	P02.5	P02.6	2C	P02.7	P02.8
	P02.3	P02.4	P02.5	3C	P02.6	P02.7
	P15.0	P15.1	P15.2	10D	P15.3	P15.4
	P15.2	P15.3	P15.4	11D	P15.5	P15.6
	P02.2	P02.3	P02.4	12B	P02.5	P02.6
	P02.1	P02.2	P02.3	13B	P02.4	P02.5
	P02.0	P02.1	P02.2	14B	P02.3	P02.4
TC36x	P02.8	P00.0	P00.1	0B	P00.2	P00.3
	P02.6	P02.7	P02.8	0C	P00.0	P00.1
	P00.0	P00.1	P00.2	1B	P00.3	P00.4
	P02.5	P02.6	P02.7	1C	P02.8	P00.0
	P02.4	P02.5	P02.6	2C	P02.7	P02.8
	P02.3	P02.4	P02.5	3C	P02.6	P02.7
TC33xEXT	P02.8	P00.0	P00.1	0B	P00.2	P00.7
	P02.7	P02.6	P02.8	0C	P00.0	P00.1
	P00.0	P00.1	P00.2	1B	P00.7	P00.3
	P02.2	P02.3	P02.7	1C	P02.6	P02.8
	P02.4	P02.1	P02.5	3C	P02.2	P02.3
TC33x/TC32x	P02.8	P00.0	P00.1	0B	P00.2	P00.3
	P02.6	P02.6	P02.8	0C	P00.0	P00.1
	P00.0	P00.1	P00.2	1B	P00.3	P00.4
	P02.5	P02.6	P02.7	1C	P02.8	P00.0
	P02.4	P02.5	P02.6	2C	P02.7	P02.8
	P02.3	P02.4	P02.5	3C	P02.6	P02.7
	P33.4	P33.5	P33.6	4C	P33.7	P33.8

**Note:** The table above is sorted by SENT channel numbers in ascending order. The same sorting is also used in the tables below.

The following tables summarize the results of the  $V_{ILD}$  measurements of the SENT input pads potentially exceeding the  $V_{ILD}$  limits with different neighbor (2N/4N) and different edge strength/driver strength configurations.

- **VILD(DIST4N):**  $V_{ILD}$  measurements with four neighbor pads (two on the left and two on the right hand side of the SENT input) used in output mode alongside the SENT input pad on the pad frame
- **VILD(DIST2N):**  $V_{ILD}$  measurements with two neighbor pads (one on the left and one on the right hand side of the SENT input) used in output mode alongside the SENT input pad on the pad frame

**4 Application hints**

**Table 33 Effect of Driver Settings Fss, Sms, Sm on SENT inputs**

Device	SENT Channel			Neighbors: Fast pads configured as Fss, others Sms/Sm	
	Name	Number	Pin	VILD(DIST4N)	VILD(DIST2N)
<b>TC39x</b>	SENT:SENT11D	11D	P15.4	x	x
	SENT:SENT20C	20C	P31.8	x	x
	SENT:SENT21C	21C	P31.9	x	x
	SENT:SENT22C	22C	P31.10	x	x
	SENT:SENT23C	23C	P31.11	x	x
	SENT:SENT24C	24C	P31.12	x	x
<b>TC38x</b>	SENT:SENT11D	11D	P15.4	x	OK
	SENT:SENT20C	20C	P31.8	x	x
	SENT:SENT21C	21C	P31.9	x	OK
	SENT:SENT22C	22C	P31.10	x	x
	SENT:SENT23B	23B	P02.12	x	OK
	SENT:SENT23C	23C	P31.11	x	x
	SENT:SENT24C	24C	P31.12	x	x
<b>TC3Ex</b>	SENT:SENT0C	0C	P02.8	x	x
	SENT:SENT1C	1C	P02.7	x	OK
	SENT:SENT2C	2C	P02.6	x	x
	SENT:SENT11D	11D	P15.4	x	OK
	SENT:SENT12B	12B	P02.4	x	OK
	SENT:SENT13B	13B	P02.3	x	OK
	SENT:SENT17D	17D	P14.0	x	OK
<b>TC37x and TC37xEXT</b>	SENT:SENT0C	0C	P02.8	x	OK
	SENT:SENT1B	1B	P00.2	x (TC37xEXT) OK (TC37x)	OK
	SENT:SENT1C	1C	P02.7	x	OK
	SENT:SENT2C	2C	P02.6	x	x (TC37xEXT) OK (TC37x)
	SENT:SENT3C	3C	P02.5	x	OK
	SENT:SENT10D	10D	P15.2	x (TC37xEXT) OK (TC37x)	OK
	SENT:SENT11D	11D	P15.4	x	OK
	SENT:SENT12B	12B	P02.4	x	OK
	SENT:SENT13B	13B	P02.3	x	x (TC37xEXT) OK (TC37x)
	SENT:SENT14B	14B	P02.2	x	OK

**(table continues...)**

**4 Application hints**

**Table 33 (continued) Effect of Driver Settings Fss, Sms, Sm on SENT inputs**

Device	SENT Channel			Neighbors: Fast pads configured as Fss, others Sms/Sm	
	Name	Number	Pin	VILD(DIST4N)	VILD(DIST2N)
TC36x	SENT:SENT0B	0B	P00.1	x	OK
	SENT:SENT0C	0C	P02.8	x	OK
	SENT:SENT1B	1B	P00.2	x	OK
	SENT:SENT1C	1C	P02.7	x	OK
	SENT:SENT2C	2C	P02.6	x	x
	SENT:SENT3C	3C	P02.5	x	x
TC33xEXT	SENT:SENT0B	0B	P00.1	x	x
	SENT:SENT0C	0C	P02.8	x	OK
	SENT:SENT1B	1B	P00.2	x	OK
	SENT:SENT1C	1C	P02.7	x	OK
	SENT:SENT3C	3C	P02.5	x	OK
TC33x/TC32x	SENT:SENT0B	0B	P00.1	x	OK
	SENT:SENT0C	0C	P02.8	x	OK
	SENT:SENT1B	1B	P00.2	x	OK
	SENT:SENT1C	1C	P02.7	x	OK
	SENT:SENT2C	2C	P02.6	x	x
	SENT:SENT3C	3C	P02.5	x	x
	SENT:SENT4C	4C	P33.6	x	OK

**Table 34 Effect of Driver Settings Fsm, Fm, Sms, Sm on SENT inputs**

Device	SENT Channel			Neighbors: Fast pads configured as Fsm or Fm, others Sms/Sm	
	Name	Number	Pin	VILD(DIST4N)	VILD(DIST2N)
TC39x	SENT:SENT11D	11D	P15.4	OK	OK
	SENT:SENT20C	20C	P31.8	x	OK
	SENT:SENT21C	21C	P31.9	x	OK
	SENT:SENT22C	22C	P31.10	x	OK
	SENT:SENT23C	23C	P31.11	x	OK
	SENT:SENT24C	24C	P31.12	x	OK

(table continues...)

**4 Application hints**

**Table 34 (continued) Effect of Driver Settings Fsm, Fm, Sms, Sm on SENT inputs**

Device	SENT Channel			Neighbors: Fast pads configured as Fsm or Fm, others Sms/Sm	
	Name	Number	Pin	VILD(DIST4N)	VILD(DIST2N)
<b>TC38x</b>	SENT:SENT11D	11D	P15.4	OK	OK
	SENT:SENT20C	20C	P31.8	x	OK
	SENT:SENT21C	21C	P31.9	x	OK
	SENT:SENT22C	22C	P31.10	x	OK
	SENT:SENT23B	23B	P02.12	OK	OK
	SENT:SENT23C	23C	P31.11	x	OK
	SENT:SENT24C	24C	P31.12	x	OK
<b>TC3Ex</b>	SENT:SENT0C	0C	P02.8	OK	OK
	SENT:SENT1C	1C	P02.7	OK	OK
	SENT:SENT2C	2C	P02.6	OK	OK
	SENT:SENT11D	11D	P15.4	OK	OK
	SENT:SENT12B	12B	P02.4	OK	OK
	SENT:SENT13B	13B	P02.3	OK	OK
	SENT:SENT17D	17D	P14.0	OK	OK
<b>TC37x and TC37xEXT</b>	SENT:SENT0C	0C	P02.8	OK	OK
	SENT:SENT1B	1B	P00.2	OK	OK
	SENT:SENT1C	1C	P02.7	OK	OK
	SENT:SENT2C	2C	P02.6	OK	OK
	SENT:SENT3C	3C	P02.5	OK	OK
	SENT:SENT10D	10D	P15.2	OK	OK
	SENT:SENT11D	11D	P15.4	OK	OK
	SENT:SENT12B	12B	P02.4	OK	OK
	SENT:SENT13B	13B	P02.3	OK	OK
	SENT:SENT14B	14B	P02.2	OK	OK
<b>TC36x</b>	SENT:SENT0B	0B	P00.1	OK	OK
	SENT:SENT0C	0C	P02.8	OK	OK
	SENT:SENT1B	1B	P00.2	OK	OK
	SENT:SENT1C	1C	P02.7	OK	OK
	SENT:SENT2C	2C	P02.6	OK	OK
	SENT:SENT3C	3C	P02.5	OK	OK

**(table continues...)**

**4 Application hints**

**Table 34 (continued) Effect of Driver Settings Fsm, Fm, Sms, Sm on SENT inputs**

Device	SENT Channel			Neighbors: Fast pads configured as Fsm or Fm, others Sms/Sm	
	Name	Number	Pin	VILD(DIST4N)	VILD(DIST2N)
TC33xEXT	SENT:SENT0B	0B	P00.1	OK	OK
	SENT:SENT0C	0C	P02.8	OK	OK
	SENT:SENT1B	1B	P00.2	OK	OK
	SENT:SENT1C	1C	P02.7	OK	OK
	SENT:SENT3C	3C	P02.5	OK	OK
TC33x/TC32x	SENT:SENT0B	0B	P00.1	OK	OK
	SENT:SENT0C	0C	P02.8	OK	OK
	SENT:SENT1B	1B	P00.2	OK	OK
	SENT:SENT1C	1C	P02.7	OK	OK
	SENT:SENT2C	2C	P02.6	OK	OK
	SENT:SENT3C	3C	P02.5	OK	OK
	SENT:SENT4C	4C	P33.6	OK	OK

**Table 35 Effect of Driver Settings Fm, Sms, Sm on SENT inputs**

Device	SENT Channel			Neighbors: Fast pads configured as Fm, others Sms/Sm	
	Name	Number	Pin	VILD(DIST4N)	VILD(DIST2N)
TC39x	SENT:SENT11D	11D	P15.4	OK	OK
	SENT:SENT20C	20C	P31.8	OK	OK
	SENT:SENT21C	21C	P31.9	OK	OK
	SENT:SENT22C	22C	P31.10	OK	OK
	SENT:SENT23C	23C	P31.11	OK	OK
	SENT:SENT24C	24C	P31.12	OK	OK
TC38x	SENT:SENT11D	11D	P15.4	OK	OK
	SENT:SENT20C	20C	P31.8	OK	OK
	SENT:SENT21C	21C	P31.9	OK	OK
	SENT:SENT22C	22C	P31.10	OK	OK
	SENT:SENT23B	23B	P02.12	OK	OK
	SENT:SENT23C	23C	P31.11	OK	OK
	SENT:SENT24C	24C	P31.12	OK	OK

**Table 36 Abbreviations used for pad configuration**

Symbol	Pad type	Driver Strength / Edge Mode
Fss	Fast	strong driver, sharp edge
Fsm	Fast	strong driver, medium edge
Fm	Fast	medium driver

**(table continues...)**

**4 Application hints**

**Table 36 (continued) Abbreviations used for pad configuration**

Symbol	Pad type	Driver Strength / Edge Mode
Sms	Slow	medium driver, sharp edge
Sm	Slow	medium driver

**Recommendation**

From the tables above, following is the conclusion based on the measured  $V_{ILD}$  values for each pad in different configurations:

**Table 37 Conclusion for SENT application usage**

Symbol	Conclusion for SENT application usage
OK	$V_{ILD}$ is below the standard threshold (100mV) and hence pin can be used in the mentioned configuration.
x	<p><math>V_{ILD}</math> is above the standard threshold (100mV) and hence pin cannot be used in the mentioned configuration. Following are possible alternatives to use the SENT pad (marked as “OK” in the tables above):</p> <ul style="list-style-type: none"> <li>• Configure the neighboring pads have to weaker edge mode / driver strength (Fsm or Fm instead of Fss)</li> <li>• Use SENT input with 2N neighbors instead of 4N</li> </ul>

**4.124 [SENT\_TC.H007] Range for divider value DIV - Documentation correction**

**Description**

In section “Baud Rate Generation” and in the description of register CFDRx in the SENT chapter of the TC3xx User’s Manual, the range for the divider value DIV is documented as

- $DIV = [2200, 49100]$

The upper limit of this range is incorrect.

**Documentation correction**

The correct range that can be used for the divider value DIV is

- $DIV = [2200, 52428]$

**4.125 [SENT\_TC.H009] Unexpected NNI error behavior**

**Description**

The NNI interrupt is triggered when the actual number of transmitted nibbles exceeds the expected count predefined in RCRx.FRL. Specifically, when  $IEP = 0$  and no pause pulse is used, NNI interrupt performs as expected. However, when  $IEP = 1$  and a pause pulse is used, the interrupt is not triggered if the number of transmitted nibbles surpasses the expected value by one nibble. In this case, the NNI interrupt is only triggered when the number of nibbles transmitted surpasses the expected value by two or more nibbles.

**Recommendation**

Due to this issue, SENT messages could be missed. This can be detected by implementing timeout or message rate checking mechanisms.

## 4 Application hints

### 4.126 [SMU\_TC.H010] Clearing individual SMU flags: use only 32-bit writes

#### Description

The SMU registers shall only be written via 32-bit word accesses (i.e. ST.W instruction), as mentioned in table “Registers Overview” of the SMU chapter in the User’s Manual.

If any other instruction such as LDMST or SWAPMSK.W is used to modify only a few bits in the 32-bit register, then this may have the effect of modifying/clearing unintended bits.

#### Recommendation (Examples in C Language)

- **Example 1:** To clear status flag SF2 in register AG0, use:
  - SMU\_AG0.U = 0x0000 0004;
- **Example 2:** To clear status flags EF2 in register RMEF and RMSTS, use:
  - SMU\_RMEF.U = 0xFFFF FFFB;
  - SMU\_RMSTS.U = 0xFFFF FFFB;

Here the <REGISTER>.U implies writing to the register as an unsigned integer, which normally results in a compiler translation into an ST.W instruction.

#### Safety Considerations

As long as software uses only 32-bit writes to the SMU registers, there is no risk of malfunction.

In case the software does not use 32-bit writes (and for example uses bit-wise operations such as LDMST instructions instead) – then potentially unintended flags may be written and modified in the SMU registers. Depending on the application, this may potentially have an impact on safety and/or diagnostics.

**Note:** *The SMU reaction itself (for example alarm action triggering) is not affected even if the software unintentionally clears additional bits by not using a 32-bit write as recommended.*

### 4.127 [SMU\_TC.H012] Handling of SMU alarms ALM7[1] and ALM7[0]

#### Description

The FSI RAM is used to configure the PFLASH. For security related reason, the access to this RAM is restricted. Therefore, in order to avoid accesses to this RAM through its SSH, the MBIST Controller 40 is not disclosed in the AURIX™ TC3xx User’s Manual.

However, the SMU alarms ALM7[1] and ALM7[0] are set intentionally after PORST and system reset and shall be cleared by the application SW (cf. ESM[SW]:SYS:MCU\_FW\_CHECK in Safety Manual).

Also, in order to clear the SMU alarms ALM7[1] and ALM7[0], it is necessary to clear the alarms within this MC40.

#### Recommendation

The following register addresses have to be written to clear the FSI RAM Fault Status and ECC Detection Register:

MCi\_FAULTSTS (i=40, 0xF00638F0) = (16-bit write) 0x0

MCi\_ECCD (i=40, 0xF0063810) = (16-bit write) 0x0

## 4 Application hints

### 4.128 [SMU\_TC.H013] Increased Fault Detection for SMU Bus Interface (SMU\_CLC Register)

#### Description

Transient faults can possibly affect the SMU\_CLC register and lead to disabling the SMU\_core. This unintended switching off of SMU\_core cannot be detected if the FSP protocol is not used at all or used in FSP bi-stable mode.

#### Recommendation

In order to increase the capability of the microcontroller to detect such faults it is recommended to:

- Option 1:  
Use FSP Dynamic dual-rail or Time-switching protocol only, don't use FSP bi-stable protocol
- Option 2:  
In case FSP protocol is not used at all or Recommendation Option 1 is not possible, the [Application SW] shall read periodically, once per FTTI, the SMU\_CLC register to react on unintended disabled SMU

### 4.129 [SMU\_TC.H016] SMU\_stdby restriction for using P33.8 as Emergency Stop input

#### Description

SMU\_core can be configured to report the presence of faults on port pin P33.8 using the Fault Signaling Protocol. However, SMU\_stdby in combination with SMU\_core can also be configured for handling common cause faults in a diverse way.

On the detection of faults, the SMU\_STDBY sets P33.8 in high impedance state (fault state). In order to recognize the high impedance state of the P33.8 as the fault state, an external pull-down is needed if bi-stable FSP is used (see for example section "Interface to the Pads (ErrorPin)" in the SMU chapter of the User's Manual). When the SMU\_stdby sets P33.8 in high impedance state (fault state), the SMU\_STS.FSP[0] bit-field as well as P33\_IN.P8 bit-field do not reflect the actual logic level of the pin P33.8.

While SMU\_stdby sets P33.8 to high impedance the port triggered emergency stop function on this port will be disabled and no emergency stop event will be generated as consequence.

#### Recommendation

Therefore it is recommended to use port pin P21.2 (PORT B) as Emergency Stop input when SMU\_stdby is configured to report fault on P33.8.

### 4.130 [SMU\_TC.H017] Handling of ALM21[7] when safety flip-flop self-test is executed

#### Description

After execution of the Safety flip-flop self-test when the PMS or PMSLE module is enabled (SMU\_RMCTL[5] = 1), the alarm ALM21[7] might not be reported. This is due to the fact that the clock of the SMU\_stdby (where the ALM21[7] belongs to) is slower than the alarm pulse, therefore the alarm might not be reported.

This happens only during the Safety flip-flop self-test, because in case of a real error during run mode, the alarm signal will persist longer and it will be caught also by a slower clock.

Furthermore, flags used to signal the end of the test (SMU\_RMSTS[5]) or a failure in the test (SMU\_RMEF[5]) are working properly.

A more detailed view shows that the error signal coming out from the PMS/PMSLE Safety flip-flop is connected to both SMU\_core and SMU\_stdby for diverse processing during run mode. During the Safety flip-flop self-test

**4 Application hints**

of the PMS/PMSLE module, it is expected that the error signal should trigger an alarm in the SMU\_core only, which means that the ALM21[7] is superfluous.

**Recommendation**

When the Safety flip-flop self-test is executed and the PMS/PMSLE module is enabled (SMU\_RMCTL[5] = 1), ignore alarm ALM21[7] and clear it when the test is completed.

**4.131 [SMU\_TC.H020] There is no GTM SSH instances for the alarms ALM6[12:10]**

**Description**

The specific alarm mapping table in the TC33x/TC32x appendix of the TC3xx user manual states that the corresponding alarms ALM6[12:10] are mapped to GTM related SRAMs. This is incorrect and must be ignored. There are no associated GTM related SRAMs. Configuration of those alarm nodes will not result in any further action by SMU.

**Scope**

The issue is related to documentation only.

**Documentation update**

The alarm mapping table must be corrected as follows:

**Table 38 Alarm Mapping related to ALM6 group**

<b>Alarm Index</b>	<b>Module</b>	<b>Safety Mechanism &amp; Alarm Indication</b>
ALM6[10]	Reserved	Reserved
ALM6[11]	Reserved	Reserved
ALM6[12]	Reserved	Reserved

**4.132 [SRI\_TC.H001] Using LDMST and SWAPMSK.W instructions on SRI mapped peripheral registers (range 0xF800 0000-0xFFFF FFFF)**

**Description**

The LDMST and SWAPMSK.W instructions in the AURIX™ microcontrollers are intended to provide atomicity as well as bit-wise operations to a targeted memory location or peripheral register. They are also referred to as Read-Modify-Write (RMW) instructions.

The bit-manipulation functionality is intended to provide software a mechanism to write to individual bits in a register, without affecting other bits. The bits to be written can be selected through a mask in the instruction. Please refer to the TriCore™ Architecture Manual for further information about these instructions and their formats.

**Restrictions for SRI mapped Peripherals**

The bit-manipulation functionality is supported only on registers accessed via the SPB bus, and is not supported on the SRI mapped peripheral range, that is address range 0xF800 0000 to 0xFFFF FFFF

## 4 Application hints

The SRI mapped peripheral range includes the following units (if available):

- In **TC2xx**: EBU, PMU0, SRI Crossbar, LMU, DAM, FFT, CPUx SFRs and CSFRs, MCDS, miniMCDS; see table “On Chip Bus Address Map of Segment 15” in chapter “Memory Map”
- In **TC3xx**: DMU, LMU, EBU, DAM, SRI Crossbar, SPU, CPUx SFRs and CSFRs, AGBT, miniMCDS, ...; see table “On Chip Bus Address Map of Segment 15” in chapter “Memory Map”

On the SRI mapped peripherals, usage of these instructions always results in all the bits of a register being written, and not just specific individual bits.

**Note:** *The instructions are still executed atomically on the bus – that is, the SRI is locked between the READ and the WRITE transaction.*

### 4.133 [SRI\_TC.H003] Incorrect information in SRI error capture registers for HSM transactions

#### Description

Details of SRI errors resulting from HSM transactions are not captured in the ERRx register, although PESTAT is updated. Since the ERRx registers are not updated, they could hold a value that was sampled previously due to another non HSM error.

#### Scope

The problem is limited to the details of HSM error capture mechanism as implemented in the SRI.

#### Effects

For all transactions from the HSM master that are errored by a slave connected on SCIx, the corresponding PESCLx bit in PESTAT is set to signal the occurrence of an error, if enabled by the corresponding PEENx. However, the HSM transaction data will not be captured in the DOMy\_ERRx, DOMy\_ERRADDRx as indicated by DOMy\_PCONx.PEACK registers. In general, within the SRI, transaction details of errors resulting from transactions initiated by the HSM are not captured. The content of these registers must be considered as invalid as it does not relate to the erroneous HSM transaction, and therefore must be ignored. The above SRI transaction error capture behavior in relation to HSM is an imposed security restriction and hence, a hardwired configuration which cannot be modified during runtime.

#### Workaround

The information in the registers DOMy\_ERRx, DOMy\_ERRADDRx must be ignored if DOMy\_PCONx.PEACK is not set, even if PESTAT.PESCLx is set as described in the user manual.

### 4.134 [SRI\_TC.H005] Clarification of effects for setting PECONx.SETPE

#### Description

TC3xx devices have a register field PECONx.SETPE, where x stands for the SCI index. On each SCI instance of each SRI instance, the corresponding bit-field SETPE can be set to create a notification, both an interrupt and alarm, while not necessarily capturing the transaction information nor locking the ERRADDRx and ERRx registers. Under usual error circumstances, the occurrence of a protocol error leads to capture of the transaction information in ERRADDRx, ERRx, sets the PECONx.PEACK lock bit and sticky PESTAT.PESCLx bit and the signaling of alarms and interrupts. In contrast, setting PECONx.SETPE by software only signals alarms and interrupts, but does not lock the error capture registers.

#### Scope

This AppHint is to enhance the documentation, as the results for setting bit-field SETPE are not described completely in the user manual.

---

**4 Application hints****Effects**

When using PECONx.SETPE to simulate a protocol error detection condition, interrupts and alarms can be signaled but it has no affect on the information in the error capture registers.

PECONx.SETPE is set (temporarily for one cycle) as a result of the write to PECON (set protocol error) if the requisite slave arbiter PESTATx.PESCI is not already set. The setting of PECONx.SETPE, generates both an interrupt and an alarm. If the ERRx and ERRADDRx registers are locked, as indicated by PECONx.PEACK then this will be the result of an earlier erroneous transaction. If not locked, then they contain the address of the last transaction (erroneous or not), however, the values are not valid unless PECONx.PEACK is set.

**4.135 [SSW\_TC.H001] Security hardening measure for the startup behavior****Description**

In order to increase the robustness of the debug protection mechanism against malicious attacks, it is strongly suggested to always apply another layer of protection in combination with it.

**Recommendation**

On top of the debug protection mechanism, enabled via UCB\_DBG through the HF\_PRONCONDBG.DBGIFLCK bit using a 256-bit password, you must set the global PFLASH or DFLASH read protection.

Both protections can be enabled individually or together. It is not mandatory to set both protections at the same time.

In most cases PFLASH will be the preferred option since standard drivers for DFLASH (for example for EEPROM emulation) do not support DFLASH protection.

In order to enable the global PFLASH read protection, HF\_PROCONPF.RPRO has to be set to 1 inside the UCB\_PFLASH\_ORIG/COPY.

In order to enable the global DFLASH read protection, HF\_PROCONDF.RPRO has to be set to 1 inside the UCB\_DFLASH\_ORIG/COPY.

Be aware that the global read protection will apply also a write protection over the entire PFLASH or DFLASH memory respectively.

The enabled read protection is always effective for the startup hardening. For the Flash read access by CPUs it has only an effect in case the device is not booting from internal Flash.

In case a software update is needed, the write protection, inherited as side effect from the global read protection, can be temporarily disabled executing the “Disable Protection” command sequence.

The PFLASH write protection is also contained in the same UCB\_PFLASH\_ORIG/COPY, so this leads to have only one password (different from the Debug password) to disable write and read protection mechanisms at the same time.

If you remove the global PFLASH read protection this will remove also the PFLASH write protection at the same time.

Same for the DFLASH write protection, which is included in the UCB\_DFLASH\_ORIG/COPY. Another single password is used to disable write and read protection over Data Flash 0 at the same time. Data Flash 1 and HSM PFLASH sectors are protected with another security mechanism through “exclusive protection”.

The disabled protection is valid until the next reset or executing the “Resume Protection” command sequence.

For further details please refer to AP32399 “TC3xx debug protection (with HSM)” or to chapter “Non Volatile Memory (NVM) Subsystem“ in the AURIX™ TC3xx User’s Manual.

## 4 Application hints

### 4.136 [STM\_TC.H004] Access to STM registers while STMDIV = 0

#### Description

If accesses to STM kernel registers are performed while bit-field STMDIV = 0<sub>H</sub> in the corresponding CCU Clock Control register (that is, clock  $f_{\text{STM}}$  is stopped),

- the SPB bus gets locked after the first access until a timeout (defined in BCU Control register field SBCU\_CON.TOUT) occurs;
- after the second access the STM slave will answer with RTY (retry) until the STM is clocked again with STMDIV > 0<sub>H</sub>

The corresponding CCU Clock Control register including STMDIV is:

- CCUCON1 in **TC2xx**
- CCUCON0 in **TC3xx**

#### Recommendation

- In **TC2xx**, do not access any STM kernel register while CCUCON1.STMDIV = 0<sub>H</sub>
- In **TC3xx**, do not access any STM kernel register while CCUCON0.STMDIV = 0<sub>H</sub>

**Revision history**

**Revision history**

<b>Document version</b>	<b>Date of release</b>	<b>Description of changes</b>
1.0	2020-01-17	First version for TC33x step AA
1.1	2020-03-31	Update: <ul style="list-style-type: none"> <li>• TC32x included</li> <li>• New/updated text modules see column “Change” in tables 3..5 of errata sheet V1.1</li> <li>• Removed: <ul style="list-style-type: none"> <li>- CPU_TC.H016 (List of OS and I/O Privileged Instructions - Documentation Update): updated description in TriCore™ TC1.6.2 Core Architecture Manual V1.2.1, Vol. 2 Instruction Set, table 14</li> <li>- GTM_TC.H020 (GTM can cause unintended bus errors after enabling when SPB or GTM frequency is very low): does not apply to this design step</li> </ul> </li> </ul>
1.2	2020-07-06	Update: <ul style="list-style-type: none"> <li>• New/updated text modules see column “Change” in tables 3..5 of errata sheet V1.2</li> </ul>
1.3	2020-10-23	Update: <ul style="list-style-type: none"> <li>• New/updated text modules see column “Change” in tables 3..5 of errata sheet V1.3</li> <li>• Text module SCR_TC.022 (Effect of application or system reset and warm PORST on MC77_ECCD and MC78_ECCD for SCR RAMs) moved from chapter “Application Hints” to chapter “Functional Problems”</li> <li>• Removed: <ul style="list-style-type: none"> <li>- SAFETY_TC.H001 (Features intended for development only – Documentation update to Safety Manual)</li> <li>- SAFETY_TC.H003 (ESM[SW]:EDSADC: VAREF_PLAUSIBILITY and ESM[SW]: EVADC:VAREF_PLAUSIBILITY – Additional information)</li> <li>- SAFETY_TC.H004 (ESM[HW]:PMS: VEXT_VEVR SB_OVERVOLTAGE – Wording update)</li> </ul> </li> </ul> <p>&gt;&gt; updated description in TC3xx Safety Manual V1.11</p>
1.4	2021-01-22	Update: <ul style="list-style-type: none"> <li>• New/updated text modules see column “Change” in tables 3..5 of errata sheet V1.4</li> </ul>
1.5	2021-04-22	Update: <ul style="list-style-type: none"> <li>• New/updated text modules see column “Change” in tables 3..5 of errata sheet V1.5 <ul style="list-style-type: none"> <li>- Text modules already published in TC3xx Errata Advance Information 2021_02: MCMCAN_AI.022, SMU_TC.H012</li> <li>- Text modules already published in TC3xx Errata Advance Information 2021_03: FlexRay_TC.H004, SCR_TC.023, SENT_TC.H007</li> </ul> </li> <li>• Removed SCU_TC.032, included description in update of SCU_TC.031 (Bits SCU_STSTAT.HWCFGx (x=1-5) could have an unexpected value in application if pins HWCFGx are left unconnected)</li> </ul>

**Revision history**

<b>Document version</b>	<b>Date of release</b>	<b>Description of changes</b>
1.6	2021-07-23	<p>Update:</p> <ul style="list-style-type: none"> <li>• New/updated text modules see column “Change” in tables 3..5 of errata sheet V1.6                             <ul style="list-style-type: none"> <li>- Text modules already published in TC3xx Errata Advance Information 2021_05: GTM_AI.364, GTM_AI.370, GTM_AI.374..376</li> <li>- Text modules already published in TC3xx Errata Advance Information 2021_06: FLASH_TC.055, MCMCAN_TC.H008, MTU_TC.018, PMS_TC.015</li> </ul> </li> </ul>
1.7	2021-11-04	<p>Update:</p> <ul style="list-style-type: none"> <li>• New/updated text modules see column “Change” in tables 3..5 of errata sheet V1.7                             <ul style="list-style-type: none"> <li>- Text modules already published in TC3xx Errata Advance Information 2021-09: FLASH_TC.056, GTM_AI.358, MCMCAN_AI.023, MEMMAP_TC.001, PADS_TC.H007, SAFETY_TC.023, SAFETY_TC.024</li> </ul> </li> <li>• Table 2 (History List) of errata sheet V1.6: corrected “GTM.AI.*” to “GTM_AI.*”</li> </ul>
1.8	2022-03-25	<p>Update:</p> <ul style="list-style-type: none"> <li>• New/updated text modules see column “Change” in tables 3..5 of errata sheet V1.8                             <ul style="list-style-type: none"> <li>- Text modules already published in TC3xx Errata Advance Information 2021-12: ASCLIN_TC.012, CCU_TC.005, GTM_TC.H026, GTM_TC.H027, PMS_TC.H011, SAFETY_TC.H019, SAFETY_TC.H020, SCR_TC.024, SCU_TC.033</li> <li>- Text modules already published in TC3xx Errata Advance Information 2022-01: ADC_TC.H033, ASCLIN_TC.H007, GTM_AI.406, GTM_AI.408, OCDS_TC.H015, SMU_TC.H016</li> </ul> </li> <li>• Removed SCU_TC.H016 (RSTSTAT reset values - documentation update): updated in TC3xx User’s Manual V1.2.0 and following</li> </ul>

**Revision history**

Document version	Date of release	Description of changes
1.9	2022-07-29	<p>Update:</p> <ul style="list-style-type: none"> <li>• Documentation reference changed to Safety Manual v2.0 (see table 1)</li> <li>• New/updated text modules see column “Change” in tables 3..5 of errata sheet V1.9                             <ul style="list-style-type: none"> <li>- Text modules already published in TC3xx Errata Advance Information 2022-05: ADC_TC.H043, FLASH_TC.H021, MTU_TC.H016, SCU_TC.H025</li> <li>- Text modules already published in TC3xx Errata Advance Information 2022-06: GTM_AI.454, INT_TC.H006, LBIST_TC.H003, MCMCAN_AI.024, MCMCAN_AI.H002, SAFETY_TC.026, SAFETY_TC.027</li> </ul> </li> <li>• Removed due to updates in Safety Manual v1.12 and following:                             <ul style="list-style-type: none"> <li>- SAFETY_TC.002 (section 6.325 SM[HW]:NVM.PFLASH:FLASHCON_MONITOR),</li> <li>- SAFETY_TC.006 (section 6.428 SM[HW]:SMU:CCF_MONITOR)</li> <li>- SAFETY_TC.007 (6.346 SM[HW]:PMS:VDDM_MONITOR)</li> <li>- SAFETY_TC.H002 (section 6.97 SM[HW]:CPU.PTAG:ERROR_DETECTION)</li> <li>- SAFETY_TC.H006 (section 6.349 SM[HW]:PMS:VDD_MONITOR)</li> <li>- SAFETY_TC.H007 (section 6.43 SM[HW]:CLOCK:PLL_LOSS_OF_LOCK_DETECTION)</li> <li>- SAFETY_TC.H008 (section 6.47 SM[HW]:CONVCTRL:PHASE_SYNC_ERR)</li> <li>- SAFETY_TC.H015 (section 6.336 SM[HW]:NVM:STARTUP_PROTECTION)</li> <li>- SAFETY_TC.H016 (section 5.32 ESM[SW]:CPU:SOFTERR_MONITOR)</li> </ul> </li> </ul>

**Revision history**

Document version	Date of release	Description of changes
1.10	2022-11-11	<p>Update:</p> <ul style="list-style-type: none"> <li>• Documentation reference changed to TC3xx User’s Manual V2.0.0 (see table 1)</li> <li>• New/updated text modules see column “Change” in tables 3..5 of errata sheet V1.10 <ul style="list-style-type: none"> <li>- Text modules already published in TC3xx Errata Advance Information 2022-08: FlexRay_TC.H005, SCR_TC.019</li> <li>- Text modules already published in TC3xx Errata Advance Information 2022-09: INT_TC.H006, MBIST_TC.H001</li> </ul> </li> <li>• Removed: <ul style="list-style-type: none"> <li>- ASCLIN_TC.012 (not applicable to design implementation in AURIX™ family)</li> <li>- CPU_TC.H021 (does not apply to TC33x/TC32x (only one CPU core))</li> </ul> </li> <li>• Removed due to updates in TC3xx documentation (see also table 1): <ul style="list-style-type: none"> <li>- ADC_TC.P014 (updated figure “Equivalent Circuitry for Analog Inputs” included in TC33x/TC32x AA Data Sheet V0.7 (and newer versions))</li> <li>- ADC_TC.H034 (updated footnote on QCONV in table “VADC 5V” in TC33x/TC32x AA Data Sheet V0.7 (and newer versions))</li> <li>- BROM_TC.H017 (corrected in footnote 1 to table ““ALL CHECKS PASSED” indication by CHSW” in Firmware chapter of product specific Appendix V1.3.0 (and newer versions))</li> <li>- CCU_TC.004 (updated in description of register OSCCON in Clocking System chapter of TC3xx UM V1.6.0 (and newer versions))</li> <li>- CPU_TC.H020 (updated in tables “Register Overview – SPR”, “Register Overview – DLMU”, sections “Scratch Pad SRAMs”, “DLMU SRAMs”, “Register access enable Protection”, “Safety Protection registers” in CPU chapter of TC3xx UM V1.5.0 (and newer versions))</li> <li>- FLASH_TC.H019 (updated in section “Write Burst Once” in NVM chapter of TC3xx UM V1.4.0 (and newer versions))</li> <li>- GTM_TC.021 (updated in description of registers GTM_CANOUTSEL0, GTM_CANOUTSEL1 in GTM chapter of product specific Appendix V1.3.0 (and newer versions))</li> <li>- GTM_TC.022 (updated in description of register ATOMi_AGC_ENDIS_STAT in GTM chapter of TC3xx UM V2.0.0)</li> <li>- GTM_TC.H022 (updated in description of register ATOMi_AGC_ENDIS_CTRL in GTM chapter of TC3xx UM V1.4.0 (and newer versions))</li> <li>- GTM_TC.H026 (updated in Table 208 “Assignment of TOUTSEL Registers to TOUTy Outputs” in TC33x/TC32x Appendix to TC3xx User Manual V2.0)</li> <li>- MTU_TC.019 (updated in description of register MCI_MCONTROL in MTU chapter of TC3xx UM V1.4.0 (and newer versions))</li> <li>- MTU_TC.H017 (updated in section “SRAM Error Detection &amp; Correction (EDC/ECC)” in MTU chapter of TC3xx UM V2.0.0)</li> <li>- PINNING_TC.H002 (notes added at the beginning of “Package Variant Pin Configuration” chapters for TQFP-144, TQFP-100 and TQFP-80 packages in TC33x/TC32x AA Data Sheet V1.0 (and newer versions))</li> </ul> </li> </ul>

**Revision history**

Document version	Date of release	Description of changes
		<ul style="list-style-type: none"> <li>- PMS_TC.012 (updated in section “Primary under-voltage monitors and Cold PORST” in PMS/PMSLE chapter of TC3xx UM V1.6.0/V1.5.0 (and newer versions))</li> <li>- PMS_TC.H005 (updated in section “Standby Controller (SCR) Interface” in PMS/PMSLE chapter of TC3xx UM V1.6.0 (and newer versions))</li> <li>- PMS_TC.H006 (updated in table “EVR33 External Component Reference” in PMSLE chapter of TC3xx UM V1.6.0 (and newer versions) and in table “EVR33 LDO” in TC33x/TC32x Data Sheet V1.0 (and newer versions))</li> <li>- RESET_TC.P003 (updated in table “Reset” in TC33x/TC32x AA Data Sheet V0.7 (and newer versions))</li> <li>- SAFETY_TC.H018 (updated in Safety Manual v2.0, sections 4.3.1 (Safety Related Functions, table in Introduction), 4.3.2.6 (Digital Acquisition ASIL B/D), 4.3.2.7 (Digital Actuation ASIL B/D))</li> <li>- SCR_TC.H013 (notes added after the description of register RCT_CON in SCR chapter of TC3xx UM V1.1.0 (and newer versions))</li> <li>- SCR_TC.H015 (updated in table “WUF Configuration Registers Address Map” in SCR chapter of TC3xx UM V2.0.0)</li> <li>- SCU_TC.H022 (updated in section “Functional Description” of chapter “LBIST Support” in SCU chapter of TC3xx UM V2.0.0)</li> <li>- SMU_TC.H015 (updated in figure “Reference clocks for FSP timings” and in description of register FSP in SMU chapter of TC3xx UM V1.6.0 (and newer versions))</li> </ul>

**Revision history**

Document version	Date of release	Description of changes
2.0	2023-03-15	<p>Update to latest errata sheet document template and aligned with AURIX™ TC4xx errata sheet flow (details see below).</p> <p>For new and changed errata see also column "Change" in tables 2, 3, and 4.</p> <p>New:</p> <ul style="list-style-type: none"> <li>• GTM_AI.H482, PADS_TC.P014, PMS_TC.H015</li> <li>• ADC_TC.H044, CCU6_TC.H001, FlexRay_AI.H010, PORTS_TC.009, RESET_TC.H007, SCU_TC.H026, SCU_TC.H027 (Errata already published in TC3xx Errata Advance Information 2022-12)</li> <li>• CCU_TC.P001, GTM_AI.421, GTM_AI.H480, GTM_AI.H481, GTM_AI.487, GTM_AI.488 (Errata already published in TC3xx Errata Advance Information 2023-01)</li> </ul> <p>Changed:</p> <ul style="list-style-type: none"> <li>• GTM_AI.352, PMS_TC.006 (Errata already published in TC3xx Errata Advance Information 2022-12)</li> <li>• GTM_AI.408, SCR_TC.016, SCU_TC.H026 (Errata already published in TC3xx Errata Advance Information 2023-01)</li> </ul> <p>Removed:</p> <ul style="list-style-type: none"> <li>• ADC_TC.H036 - Updated in section "Buffer for the Analog Input" in EVADC chapter of TC3xx user manual V1.5.0 (and newer versions)</li> <li>• ADC_TC.H037 - Information included in section "Result FIFO buffer timing" with TC3xx user manual V2.0.0</li> <li>• ADC_TC.H039 - Information included in section "Result FIFO buffer timing" with TC3xx user manual V2.0.0</li> </ul> <p>When an erratum is used by different families or devices, the erratum is now identical in all errata sheets. Differences between the different families or devices are clearly highlighted in the erratum:</p> <ul style="list-style-type: none"> <li>• CPU_TC.131, FLASH_TC.P003, FlexRay_AI.104, FlexRay_AI.105, FlexRay_AI.106, GTM_AI.353, MCMCAN_AI.023, PMS_TC.H008, QSPI_TC.017, SENT_TC.H006, SMU_TC.012, SRI_TC.H001, STM_TC.H004</li> </ul> <p>Following editorial changes were applied to several (not all) errata (examples):</p> <ul style="list-style-type: none"> <li>• Misspellings, typos, and case sensitivity</li> <li>• Aligned with latest Infineon writing guidelines</li> <li>• Added 'Description' section title when missing</li> <li>• Changed literals: 0b -&gt; subscripted B, 0x -&gt; subscripted H</li> <li>• Added '™' where missing (e.g. TriCore™)</li> <li>• Standard footnote numbers are incremented over the entire document (and not per erratum). Table footnotes are numbered per table</li> </ul> <p>In order to increase readability and comprehensibility and to standardize, some errata texts have been slightly changed. These are not changes in content. Below are some examples:</p> <ul style="list-style-type: none"> <li>• ADC_TC.H040 - Erratum is moved from chapter 'Parametric deviations' to chapter 'Application hints'</li> <li>• GTM_AI.364 - "...UDMODE=1,3" --&gt; "...UDMODE = 1 or 3 ..."</li> <li>• SCR_TC.H010 - Changed ordered list from 'a, b, c' to '1, 2, 3'</li> </ul>

**Revision history**

<b>Document version</b>	<b>Date of release</b>	<b>Description of changes</b>
2.1	2023-08-21	<p>For new and changed errata see also column "Change" in tables 2, 3, and 4.</p> <p>New:</p> <ul style="list-style-type: none"> <li>FLASH_TC.H024, GTM_AI.H497, GTM_TC.031, SCR_TC.H016</li> <li>DMA_TC.H018, LBIST_TC.H005, MBIST_TC.H002, MCMCAN_AI.025, MCMCAN_TC.006, MTU_TC.H019, NVM_TC.H001, PADS_TC.016 (is replacing PADS_TC.H009), PORTS_TC.H018, SCU_TC.H028 (Errata already published in TC3xx Errata Advance Information 2023-05)</li> <li>ADC_TC.H045, BROM_TC.H020, CPU_TC.H022, (Errata already published in TC3xx Errata Advance Information 2023-06)</li> </ul> <p>Changed:</p> <ul style="list-style-type: none"> <li>GTM_AI.353 (Erratum already published in TC3xx Errata Advance Information 2023-06)</li> </ul> <p>Removed:</p> <ul style="list-style-type: none"> <li>BROM_TC.H008 - Updated in section "CAN BSL flow" of chapter "AURIX™ TC3xx Platform Firmware" in TC3xx UM V1.1.0 (and newer versions)</li> <li>PADS_TC.H009 - Replaced by PADS_TC.016</li> </ul>
2.2	2023-12-11	<p>For new and changed errata see also column "Change" in tables 2, 3, and 4.</p> <p>New:</p> <ul style="list-style-type: none"> <li>GTM_AI.H803</li> <li>ADC_TC.H046, CPU_TC.H023, GTM_AI.442, PMS_TC.H018, PMSLE_TC.H001 (Errata already published in TC3xx Errata Advance Information 2023-09)</li> <li>GTM_AI.410, GTM_AI.516, GTM_AI.517, PMS_TC.H019, SMU_TC.015 (Errata already published in TC3xx Errata Advance Information 2023-10)</li> </ul> <p>Changed:</p> <ul style="list-style-type: none"> <li>GTM_AI.516 (Update of technical content in comparison to TC3xx Errata Advance Information 2023-10), SCU_TC.H028</li> </ul> <p>Removed:</p> <ul style="list-style-type: none"> <li>GTM_AI.H511: Initially GTM_AI.H511 was published with TC3xx Errata Advance Information 2023-10. It is now removed as this AppHint is not applicable to TC3xx devices</li> </ul> <p>TC4xx family specific content has been removed or erratum has been editorial re-written for better readability. No update of technical content.</p> <ul style="list-style-type: none"> <li>GTM_AI.H480, GTM_AI.H481, GTM_AI.H482, GTM_AI.H497, MCMCAN_AI.023, MCMCAN_AI.025, MCMCAN_AI.H002, MCMCAN_TC.H006, MCMCAN_TC.H007, QSPI_TC.017, SRI_TC.H001, STM_TC.H004</li> </ul>

**Revision history**

Document version	Date of release	Description of changes
2.3	2023-04-08	<p>For new and changed errata see also column "Change" in tables 2, 3, and 4.</p> <p>New:</p> <ul style="list-style-type: none"> <li>• GTM_AI.H526, INT_TC.H007, SMU_TC.H020</li> <li>• PER_PLL_TC.002, PMSLE_TC.H002, QSPI_TC.H011, SCU_TC.036, SENT_TC.H009, SRI_TC.H003 (Errata already published in TC3xx Errata Advance Information 2024-01)</li> <li>• GTM_AI.H519, GTM_AI.H520, GTM_AI.H521, GTM_AI.522, GTM_AI.H525 (Errata already published in TC3xx Errata Advance Information 2024-02)</li> </ul> <p>Changed:</p> <ul style="list-style-type: none"> <li>• GTM_AI.H803, SRI_TC.H003 (Update of technical content in comparison to TC3xx Errata Advance Information 2024-01)</li> <li>• GTM_AI.358 (Change of erratum already published in TC3xx Errata Advance Information 2024-01)</li> <li>• GTM_AI.517, PMS_TC.H019 (Change of erratum already published in TC3xx Errata Advance Information 2024-02)</li> </ul>
2.4	2024-09-09	<p>For new and changed errata see also column "Change" in tables 2, 3, and 4.</p> <p>New:</p> <ul style="list-style-type: none"> <li>• ADC_TC.H048, OSC_TC.H002, SRI_TC.H005</li> <li>• Errata already published in TC3xx Errata Advance Information 2024-05 <ul style="list-style-type: none"> <li>- DMA_TC.071</li> <li>- FLASH_TC.H026</li> <li>- MTU_TC.H020</li> </ul> </li> <li>• Errata already published in TC3xx Errata Advance Information 2024-06 <ul style="list-style-type: none"> <li>- GTM_AI.527</li> <li>- GTM_AI.H512</li> <li>- GTM_AI.H528</li> <li>- GTM_TC.H035</li> </ul> </li> </ul> <p>Changed:</p> <ul style="list-style-type: none"> <li>• GTM_AI.H803, SMU_TC.015</li> <li>• Change of erratum already published in TC3xx Errata Advance Information 2024-05 <ul style="list-style-type: none"> <li>- GTM_AI.410</li> <li>- GTM_AI.517</li> </ul> </li> <li>• Change of erratum already published in TC3xx Errata Advance Information 2024-06 <ul style="list-style-type: none"> <li>- FLASH_TC.H026 (Update of technical content in comparison to TC3xx Errata Advance Information 2024-05)</li> <li>- PER_PLL_TC.002</li> </ul> </li> </ul>

**Revision history**

<b>Document version</b>	<b>Date of release</b>	<b>Description of changes</b>
2.5	2024-12-10	<p>For new and changed errata see also column "Change" in tables 2, 3, and 4.</p> <p>New:</p> <ul style="list-style-type: none"> <li>• CPU_TC.H024, MCMCAN_TC.007, QSPI_TC.H013</li> <li>• Errata already published in TC3xx Errata Advance Information 2024-10                             <ul style="list-style-type: none"> <li>- ASCLIN_TC.H012</li> <li>- SAFETY_TC.029</li> <li>- SCR_TC.033</li> </ul> </li> </ul> <p>Changed:</p> <ul style="list-style-type: none"> <li>• SCR_TC.033 (Update of technical content in comparison to TC3xx Errata Advance Information 2024-10)</li> <li>• Change of erratum already published in TC3xx Errata Advance Information 2024-10                             <ul style="list-style-type: none"> <li>- GTM_AI.517</li> <li>- GTM_AI.522</li> </ul> </li> </ul>
2.6	2025-06-30	<p>For new and changed errata see also column "Change" in tables 2, 3, and 4.</p> <p>New:</p> <ul style="list-style-type: none"> <li>• Errata already published in TC3xx Errata Advance Information 2025-01                             <ul style="list-style-type: none"> <li>- SMU_TC.017</li> </ul> </li> <li>• Errata already published in TC3xx Errata Advance Information 2025-04                             <ul style="list-style-type: none"> <li>- GTM_TC.033</li> </ul> </li> <li>• Errata already published in TC3xx Errata Advance Information 2025-05                             <ul style="list-style-type: none"> <li>- CPU_TC.H025</li> <li>- INT_TC.003</li> </ul> </li> </ul> <p>Changed:</p> <ul style="list-style-type: none"> <li>• GTM_AI.340 (Update of technical content in comparison to TC3xx Errata Advance Information 2025-06)</li> <li>• Change of erratum already published in TC3xx Errata Advance Information 2025-02                             <ul style="list-style-type: none"> <li>- FPI_TC.H003</li> </ul> </li> </ul>
2.7	2025-12-10	<p>For new and changed errata see also column "Change" in tables 2, 3, and 4.</p> <p>New:</p> <ul style="list-style-type: none"> <li>• OSC_TC.H003</li> <li>• Errata already published in TC3xx Errata Advance Information 2025-08                             <ul style="list-style-type: none"> <li>- GTM_AI.538, GTM_AI.539, GTM_AI.541, GTM_AI.H540</li> </ul> </li> <li>• Errata already published in TC3xx Errata Advance Information 2025-10                             <ul style="list-style-type: none"> <li>- GETH_AI.043, CPU_TC.H027, GTM_AI.H543</li> </ul> </li> </ul> <p>Changed:</p> <ul style="list-style-type: none"> <li>• Change of erratum already published in TC3xx Errata Advance Information 2025-08                             <ul style="list-style-type: none"> <li>GTM_AI.374, GTM_AI.522, GTM_AI.H803, GTM_AI.H526</li> </ul> </li> <li>• Change of erratum already published in TC3xx Errata Advance Information 2025-10                             <ul style="list-style-type: none"> <li>- GTM_AI.541</li> </ul> </li> </ul>

## Trademarks

All referenced product or service names and trademarks are the property of their respective owners.

**Edition 2025-12-10**

**Published by**

**Infineon Technologies AG**

**81726 Munich, Germany**

**© 2025 Infineon Technologies AG**

**All Rights Reserved.**

**Do you have a question about any aspect of this document?**

**Email: [erratum@infineon.com](mailto:erratum@infineon.com)**

**Document reference**

**IFX-xci1636536039829**

## Important notice

The information contained in this application note is given as a hint for the implementation of the product only and shall in no event be regarded as a description or warranty of a certain functionality, condition or quality of the product. Before implementation of the product, the recipient of this application note must verify any function and other technical information given herein in the real application. Infineon Technologies hereby disclaims any and all warranties and liabilities of any kind (including without limitation warranties of non-infringement of intellectual property rights of any third party) with respect to any and all information given in this application note.

The data contained in this document is exclusively intended for technically trained staff. It is the responsibility of customer's technical departments to evaluate the suitability of the product for the intended application and the completeness of the product information given in this document with respect to such application.

## Warnings

Due to technical requirements products may contain dangerous substances. For information on the types in question please contact your nearest Infineon Technologies office.

Except as otherwise explicitly approved by Infineon Technologies in a written document signed by authorized representatives of Infineon Technologies, Infineon Technologies' products may not be used in any applications where a failure of the product or any consequences of the use thereof can reasonably be expected to result in personal injury.