

# AURIX™ TC37xEXT errata sheet

**Marking/Step: (E)ES-AB, AB**

10468AERRA

## 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™ TC37xEXT Appendix to User's Manual	V2.0.0	2021-02
TC37xEXT AB-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 ED Target Specification (distribution under NDA, only relevant for tool development not for application development)	V3.2.0	2020-08
AURIX™ TC3xx Safety Manual	V2.0	2021-05-03

Newer versions replace older versions, unless specifically stated otherwise.

Please always refer to the corresponding documentation for this device available in the category 'Documents' at [www.infineon.com/AURIX™](http://www.infineon.com/AURIX™) and [www.myInfineon.com](http://www.myInfineon.com).

### 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™

### About this document

- [Type] = category of deviation
  - [none] = Functional deviation
  - 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> .....	<b>1</b>
	<b>Table of contents</b> .....	<b>3</b>
<b>1</b>	<b>Errata overview</b> .....	<b>4</b>
<b>2</b>	<b>Functional deviations</b> .....	<b>18</b>
<b>3</b>	<b>Parametric deviations</b> .....	<b>141</b>
<b>4</b>	<b>Application hints</b> .....	<b>144</b>
	<b>Revision history</b> .....	<b>216</b>
	<b>Disclaimer</b> .....	<b>225</b>

## 1 Errata overview

### 1 Errata overview

List of errata referenced in this document.

**Table 2 Functional deviations**

Issue title	Change	Page
[ADC_TC.095] Ramp trigger ignored when ramp ends		18
[BROM_TC.013] CAN BSL does not send error message if no valid baudrate is detected		18
[BROM_TC.014] Lockstep comparator alarm for CPU0 after warm PORST, system or application reset if lockstep is disabled		18
[BROM_TC.016] Uncorrectable ECC error in Boot Mode Headers		19
[CCU_TC.005] ASC and CAN bootstrap loaders may not work if external clock is missing		19
[CPU_TC.130] Data Corruption when ST.B to local DSPR coincides with external access to same address		19
[CPU_TC.131] Performance issue when MADD or MSUB instructions use E0 or D0 register as accumulator		20
[CPU_TC.132] Unexpected PSW values used upon Fast Interrupt entry		21
[CPU_TC.133] Test sequence for DTAG single or double bit errors		22
[DAP_TC.005] DAP client_read: dirty bit feature of Cerberus' Triggered Transfer Mode		22
[DAP_TC.007] Incomplete client_blockread telegram in DXCM mode when using the "read CRCup" option		23
[DAP_TC.010] Performance when accessing EMEM in UWM and WM modes		23
[DMA_TC.059] ACCEN Protection not implemented for ERRINTRr		23
[DMA_TC.066] DMA double buffering operations - Update address pointer		24
[DMA_TC.067] DMA Double Buffering Software Switch buffer overflow		24
[DMA_TC.068] DMA Double Buffering lost DMA request		24
[EMEM_TC.001] Integrity of EMEM contents when using standby locked mode		25
[FLASH_TC.053] Erase size limit for PFLASH		27
[FLASH_TC.055] Multi-bit errors detected by PFlash are not communicated to SPB masters		27
[FLASH_TC.056] Reset value for register HF_ECCC is 0x0000 0000 - Documentation correction		28
[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		28
[FlexRay_AI.088] A sequence of received WUS may generate redundant SIR.WUPA/B events		29
[FlexRay_AI.089] Rate correction set to zero in case of SyncCalcResult=MISSING_TERM		29
[FlexRay_AI.090] Flag SFS.MRCS is set erroneously although at least one valid sync frame pair is received		29

**(table continues...)**

## 1 Errata overview

**Table 2** (continued) **Functional deviations**

Issue title	Change	Page
[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		30
[FlexRay_AI.092] Initial rate correction value of an integrating node is zero if pMicroInitialOffsetA,B = 0x00		30
[FlexRay_AI.093] Acceptance of start-up frames received after reception of more than gSyncNodeMax sync frames		31
[FlexRay_AI.094] Sync frame overflow flag EIR.SFO may be set if slot counter is greater than 1024		31
[FlexRay_AI.095] Register RCV displays wrong value		32
[FlexRay_AI.096] Noise following a dynamic frame that delays idle detection may fail to stop slot		32
[FlexRay_AI.097] Loop back mode operates only at 10 MBit/s		32
[FlexRay_AI.099] Erroneous cycle offset during start-up after abort of start-up or normal operation		33
[FlexRay_AI.100] First WUS following received valid WUP may be ignored		33
[FlexRay_AI.101] READY command accepted in READY state		34
[FlexRay_AI.102] Slot status vPOC!SlotMode is reset immediately when entering HALT state		34
[FlexRay_AI.103] Received messages not stored in Message RAM when in Loop Back Mode		35
[FlexRay_AI.104] Missing start-up frame in cycle 0 at coldstart after FREEZE or READY command		35
[FlexRay_AI.105] RAM select signals of IBF1/IBF2 and OBF1/OBF2 in RAM test mode		36
[FlexRay_AI.106] Data transfer overrun for message transfers Message RAM to Output Buffer (OBF) or from Input Buffer (IBF) to Message RAM		36
[GETH_AI.001] Packets with Destination Address (DA) mismatch are delayed until EOP is received in threshold (cut-through) mode		38
[GETH_AI.008] Application Error Along with Start-of-Packet Can Corrupt the FCS Field of the Previous Frame in the MAC Pipeline		39
[GETH_AI.009] Corrupted Rx Descriptor Write Data		39
[GETH_AI.010] Fatal Bus Error Interrupt Might Be Generated for Incorrect DMA Channel		40
[GETH_AI.011] Receive Queue Overflow at End of Frame Along with SPRAM Read-Write Conflict Can Cause Data Loss		40
[GETH_AI.012] Incorrect Flexible PPS Output Interval When Fine Time Correction Method is Used		41
[GETH_AI.013] False Dribble and CRC Error Reported in RMII PHY 10 Mbps Mode		42

**(table continues...)**

## 1 Errata overview

**Table 2** (continued) **Functional deviations**

Issue title	Change	Page
[GETH_AI.014] Receive DMA Channel Generates Spurious Receive Watchdog Timeout Interrupt		42
[GETH_AI.015] MAC Receive VLAN Tag Hash Filter Always Operates in Default Mode		43
[GETH_AI.016] Receive DMA Header Split Function Incorrectly Overruns the Allocated Header Buffer		44
[GETH_AI.017] Carrier-Sense Signal Not Generated When False Carrier Detected in RGMII 10/100 Mbps Mode		45
[GETH_TC.002] Initialization of RGMII interface		45
[GTM_AI.254] TIM TDU: TDU_STOP=b101 not functional		46
[GTM_AI.262] SPEC-DPLL: PSSC/PSTC behavior description incorrect		47
[GTM_AI.263] DPLL: DPLL_STATUS.LOCK1 flag (0 ->1) delayed after direction change when DPLL operating in DPLL_CTRL_0.RMO = 1		48
[GTM_AI.304] MCS: Scheduling modes Single Prioritization and Multiple Prioritization are not functional		49
[GTM_AI.306] DPLL: DPLL_NUTC.syn_t_old, DPLL_NUSC.syn_s_old not updated according specification		49
[GTM_AI.307] IRQ: AEI_IM_ADDR is not set in GTM_IRQ_NOTIFY register if cluster 0 is disabled		50
[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		50
[GTM_AI.318] MCS: NARD(I) instruction terminates unexpectedly		51
[GTM_AI.319] (A)TOM: Unexpected (A)TOM_CCUI1TCx_IRQ in up/down counter mode		51
[GTM_AI.320] ATOM: Unexpected restart of a SOMS oneshot cycle while ATOM[i]_CH[x]_CM0 is zero		52
[GTM_AI.322] DPLL: PSTC, PSSC not updated correctly after fast pulse correction completed (DPLL_CTRL1.PCM1/2 = 0)		52
[GTM_AI.323] DPLL: Registers DPLL_NUTC.SYN_T and DPLL_NUSC.SYN_S are updated by the profile (ADT_T.NT/ADT_S.NS) before the DPLL is synchronized (DPLL_STATUS.SYT/S=0)		53
[GTM_AI.325] TIM: Bits ACB[2:1] lost on interface to ARU (always zero)		53
[GTM_AI.326] TIM: ARU bit ACB[0] (signal level) incorrect in case a second ARU request occurs while the actual request is just acknowledged		54
[GTM_AI.329] Interference of MCS to AEI/ADC and CPU to AEI traffic within the same cluster could result in incorrect MCS program execution		55
[GTM_AI.331] GTM_TIM[i]_AUX_IN_SRC and GTM_EXT_CAP_EN_[i] register: wrong status 2 by AEI write access if cluster 0 is disabled		61
[GTM_AI.332] Access to registers GTM_TIM[i]_AUX_IN_SRC and GTM_EXT_CAP_EN_[i] via legacy address space: read data always 0 for AEI read access while cluster 0 is disabled		62

**(table continues...)**

## 1 Errata overview

**Table 2 (continued) Functional deviations**

Issue title	Change	Page
[GTM_AI.333] MCS bus master interface: a not word aligned address access to DPLL ram region can cause incorrect execution of MCS channel code		62
[GTM_AI.334] DPLL RAM content of single address can be corrupted after leaving debug mode		63
[GTM_AI.335] TOM output signal to SPE not functional if up/down counter mode is configured		63
[GTM_AI.336] GTM Bus Bridge: Incorrect AEI access execution in case the previous AEI access was aborted with the access timeout abort function		64
[GTM_AI.339] DPLL: Control bits DPLL_CTRL_11.PCMF1 and DPLL_CTRL_11.PCMF2 are not reset to 0 after a pulse correction is completed		64
[GTM_AI.340] TOM/ATOM: Generation of TRIG_CCU0/TRIG_CCU1 trigger signals skipped in initial phase of A/TOM SOMP one-shot mode		65
[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		66
[GTM_AI.344] DPLL: Incorrect AEI_STATUS on internal MCS2DPLL interface on valid and implemented address accesses		67
[GTM_AI.345] SPE: Incorrect behaviour of direction change control via SPE_CMD.SPE_CTRL_CMD bits		68
[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		68
[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		69
[GTM_AI.348] DPLL: Correction of missing pulses delayed after start of pulse generation		69
[GTM_AI.349] TOM-SPE: OSM-Pulse width triggered by SPE_NIPD for selected CMU_FXCLK not correct		70
[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		71
[GTM_AI.351] MAP: Disable of input lines by MAP_CTRL register not implemented for input signals TSPP0 TIM0_CHx(48) (x=0..2) and TSPP1 TIM0_CHx(48) (x=3..5)		71
[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		72
[GTM_AI.353] SPEC-ATOM: Specification of the smallest possible PWM period in SOMP mode wrong, when ARU_EN=1		73
[GTM_AI.354] MCS: Unresolved hazard resulting from RAW (Read After Write) dependency		74
[GTM_AI.357] MCS: instructions XCHB, SETB, and CLRB do not suppress register write		74
[GTM_AI.358] TOM/ATOM: Synchronous update of working register for RST_CCU0=1 and UDMODE=0b01 not correct		75

**(table continues...)**

**1 Errata overview**

**Table 2 (continued) Functional deviations**

Issue title	Change	Page
[GTM_AI.359] TOM: Both edges on TOM_OUT_T at unexpected times for RST_CCU0=1 and UDMODE>0		76
[GTM_AI.360] SPEC-(A)TOM: PCM mode (BITREV=1) is only available for UDMODE=0		76
[GTM_AI.361] IRQ: Missing pulse in single-pulse interrupt mode on simultaneous interrupt and clear event		77
[GTM_AI.362] MCS: Using wrong WURM mask during execution of instruction WURMX or WURCX		77
[GTM_AI.364] ATOM: ARU read request does not start at expected timepoint in UDMODE = 1 and UDMODE = 3		78
[GTM_AI.367] MCS: Instructions WURMX and WURCX implement invalid extended register set for argument A		79
[GTM_AI.370] TOM/ATOM: Unexpected reset of CN0 in up-down counter mode and CM0 = 2		79
[GTM_AI.371] MCS: Instruction MWRIL applies unexpected address offset calculation		80
[GTM_AI.374] SPEC-ATOM: Statement on timing of duty cycle output level change not correct for SOMP up/down-counter mode		81
[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		81
[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		82
[GTM_AI.387] DPLL: Wrong calculation of pulse generator frequency for DPLL_CTRL_0.AMT/S=1 and DPLL_CTRL_11.ADT/S=1 when number of pulses (DPLL_CTRL_0.MLT or DPLL_MLS1/2.MLS1/2) is too small		82
[GTM_AI.398] DPLL: Incorrect DPLL_THVAL calculation leading to a false direction decision in case tbu_ts0 wraps around		83
[GTM_AI.400] MCS-RTL: Division instruction may produce unexpected memory overflow and wrong results		83
[GTM_AI.404] MCS-RTL: Division instruction reports unrelated ECC error		84
[GTM_AI.406] (A)TOM: FREEZE mode has no effect on (A)TOM_OUT_T in up-down counter mode with RST_CCU0 = 1		84
[GTM_AI.408] (A)TOM-RTL: Missing edge on output signal (A)TOM_OUT when CN0 is reset with force update event		85
[GTM_AI.409] DPLL: Flags of register DPLL_STA_FLAG are not set		86
[GTM_AI.410] GTM_AEI: The AEI bridge might not execute an accepted write transaction	new	86
[GTM_AI.411] A change of the BRIDGE_MODE register might be delayed indefinitely		87
[GTM_AI.419] TIM: Potentially wrong capture values		87
[GTM_AI.421] GTM_AEI: Changing BRIDGE_MODE.MSK_WR_RSP in pipeline mode can lead to violation of pipeline protocol		89

**(table continues...)**



## 1 Errata overview

**Table 2** (continued) **Functional deviations**

Issue title	Change	Page
[GTM_AI.422] DPLL: Wrong DPLL_RDT_S_ACT/DPLL_RDT_T_ACT value in case of overflow correction		89
[GTM_AI.428] DPLL: Pulse correction is executed twice		90
[GTM_AI.429] TIM: Missing glitch detection interrupt event		90
[GTM_AI.430] TIM: Unexpected increment of filter counter		91
[GTM_AI.431] TIM: Glitch detection interrupt event of filter is not a single cycle pulse		92
[GTM_AI.441] DPLL: Missing pulse correction in case of DPLL_CTRL_1.SMC=1		92
[GTM_AI.442] GTM Top Level: GTM_HALT mode not functional when cluster 0 clock is disabled	new	93
[GTM_AI.450] DPLL: Stored time stamp values do not consider filter delays		93
[GTM_AI.451] DPLL: Wrong measured position stamps in RAM		94
[GTM_AI.454] (A)TOM: No output if trigger generation feature is used		94
[GTM_AI.456] DPLL: No action calculation		95
[GTM_AI.458] DPLL: Missing TOR interrupt and status flag	changed	95
[GTM_AI.461] MCS: Unexpected behavior of instruction WUCE		95
[GTM_AI.462] (A)TOM: Missing CCU0TC_IRQ interrupt signal		96
[GTM_AI.463] DPLL: DPLL_PVT not cleared after direction change		97
[GTM_AI.464] DPLL: Pulse correction executed twice when DPLL_CTRL_11.INCF1/2 is activated		97
[GTM_AI.465] (A)TOM: Missing CCU0TC_IRQ interrupt signal for UDMODE > 0		98
[GTM_AI.466] TOM: Unexpected behavior of TOM_OUT_T for UDMODE>0		98
[GTM_AI.474] DPLL: DPLL_PSTC, DPLL_PSSC erroneously modified		99
[GTM_AI.475] DPLL: Incorrect values of DPLL_RCDT_TX, DPLL_RCDT_SX		100
[GTM_AI.476] MCS: Unexpected instruction execution while disabling of MCS channel		100
[GTM_AI.477] DPLL: DPLL_DCGI interrupt not triggered		101
[GTM_AI.478] DPLL: Incorrect calculation of DPLL_THVAL, DPLL_THVAL2		101
[GTM_AI.483] DPLL: Malfunction on changing DPLL_PVT.PVT	new	102
[GTM_AI.487] GTM_AEI: Changing BRIDGE_MODE[2:0] in pipeline mode can lead to violation of pipeline protocol		102
[GTM_AI.488] GTM_AEI: Turning off BRIDGE_MODE.MSK_WR_RSP in asynchronous mode might lead to following transactions being corrupted		103
[GTM_AI.490] TOP: Interrupt from DPLL not detected in MCS0		103
[GTM_AI.492] DPLL: Wrong value of DPLL_INC_CNT1.INC_CNT1 upon switching to normal mode		104
[GTM_AI.507] DPLL: Irregular pulse generation and wrong PMT results	new	105

**(table continues...)**

## 1 Errata overview

**Table 2** (continued) **Functional deviations**

Issue title	Change	Page
[GTM_AI.516] SPE-RTL: IRQ raised on disabled inputs	new	105
[GTM_AI.517] (A)TOM: Missing edge on output signal (A)TOM_OUT	new	105
[GTM_TC.018] DPLLRAM trace data can be wrong		106
[GTM_TC.019] ARU can not be traced if GTM cluster 5 is disabled		107
[GTM_TC.020] Debug/Normal read access control via bit-field ODA.DRAC		107
[GTM_TC.025] Register DPLL_IRQ_NOTIFY - Documentation update for bits SORI and DCGI		108
[GTM_TC.026] Table “GTM IP Application Constraints” #1 (DPLL) - Documentation correction		108
[GTM_TC.028] Incorrect MCS behavior when SSH registers are accessed while MCS is running		108
[GTM_TC.031] Connections of ADC_TRIG4 signals - Correction in TC3xx appendix		109
[HSCT_TC.012] HSCT sleep mode not supported		110
[HSCT_TC.013] Internal loopback mode not reliable		110
[MCDS_TC.052] TriCore™ wrap around write access causes redundant MCDS message		111
[MCDS_TC.064] ACCEN0 register write not supervisor protected		111
[MCDS_TC.065] Selection of SRI trace sources		111
[MCDS_TC.066] Selection of CPU trace sources		112
[MCDS_TC.067] MCDS kernel reset shall not be used		112
[MCMCAN_AI.015] Edge filtering causes mis-synchronization when falling edge at Rx input pin coincides with end of integration phase		112
[MCMCAN_AI.017] Retransmission in DAR mode due to lost arbitration at the first two identifier bits		113
[MCMCAN_AI.018] Tx FIFO message sequence inversion		114
[MCMCAN_AI.019] Unexpected High Priority Message (HPM) interrupt		115
[MCMCAN_AI.022] Message order inversion when transmitting from dedicated Tx Buffers configured with same Message ID		117
[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		117
[MCMCAN_AI.024] Frame transmitted despite confirmed transmit cancellation		118
[MCMCAN_AI.025] Sporadic data corruption (payload) in case acceptance filtering is not finished before reception of data R3 (DB7..DB4) is completed		119
[MCMCAN_TC.006] MCMCAN specific access protection mechanisms		121
[MTU_TC.012] Security of CPU cache memories during runtime is limited		122
[MTU_TC.017] Unexpected alarms after application reset		122

**(table continues...)**

## 1 Errata overview

**Table 2** (continued) **Functional deviations**

Issue title	Change	Page
[MTU_TC.018] Gated SRAM alarms		123
[PADS_TC.011] Pull-ups activate on specific analog inputs upon PORST		124
[PADS_TC.013] Buffer type definition for P21.2: no ES functionality - Data Sheet documentation correction		124
[PADS_TC.016] Pull-ups active on P33 and P34 pins in standby mode when SCR is disabled and VEXT not supplied		124
[PMS_TC.005] Voltage rise at P33 and P34 up to $V_{EVRSB}$ during start-up and up to $V_{LVDRSTSB}$ during power-down		125
[PMS_TC.006] PORST not released during cold power-on reset until $V_{DDM}$ is available		125
[PMS_TC.007] VDDP3 or VDD Overvoltage during start-up may not be detected by PBIST		126
[PMS_TC.011] VEXT supplied PU2 and PD2 pads always in tristate after standby entry - Documentation correction		126
[PMS_TC.014] Parasitic coupling on shared ADC pins depending on supply voltages		127
[PMS_TC.015] EVRC synchronization – Documentation update for register EVRSDCTRL11 (PMS) and EVRSDCTRL2 (PMSLE)		128
[PSI5_TC.005] Incorrect read pointer upon two consecutive RDFn read operations if two or more channels are configured		128
[QSPI_TC.006] Baud rate error detection in slave mode (error indication in current frame)		129
[QSPI_TC.009] USR Events for PT1=2 (SOF: Start of Frame)		129
[QSPI_TC.010] Move Counter Mode - USR Events for PT1=4 (RBF: Receive Buffer Filled)		130
[QSPI_TC.013] Slave: No RxFIFO write after transmission upon change of BACON.MSB		130
[QSPI_TC.014] Slave: Incorrect parity bit upon TxFIFO underflow		130
[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		130
[QSPI_TC.017] Slave: Reset when receiving an unexpected number of bits		131
[SAFETY_TC.023] MCU infrastructure Safety Related Function - Documentation update		131
[SAFETY_TC.024] Clock alive monitor for $f_{SPB}$ - Documentation update		131
[SAFETY_TC.025] Wrong alarm listed in safety mechanism SM[HW]:SRI:SRI_TRANSACTION_INTEGRITY		132
[SAFETY_TC.026] Alarm for SM[HW]:IR:CFG_MONITOR - Documentation update		132
[SAFETY_TC.027] Single point fault detection for lockstep CPUs - Documentation update		132
[SCR_TC.015] Bit SCU_PMC0N1.WCAN_DIS does not disable WCAN PCLK input		133
[SCR_TC.016] DUT response to first telegram has incorrect C_START value		133
[SCR_TC.018] SSC Receive FIFO not working		133

**(table continues...)**

## 1 Errata overview

**Table 2 (continued) Functional deviations**

Issue title	Change	Page
[SCR_TC.019] Accessing the XRAM while SCR is in reset state		134
[SCR_TC.020] Stored address in mon_RETH may be wrong after a break event		134
[SCR_TC.021] RTC not counting after reset if P33.10 is high		134
[SCR_TC.022] Effect of application or system reset and warm PORST on MC77_ECCD and MC78_ECCD for SCR RAMs		135
[SCR_TC.023] External interrupts EXINT0, EXINT1 may get locked		135
[SCR_TC.024] Field ADRES in register ADCOMP_RES - Documentation correction		136
[SCU_TC.031] Bits SCU_STSTAT.HWCFGx (x=1-5) could have an unexpected value in application if pins HWCFGx are left unconnected		136
[SCU_TC.033] TESTMODE pin shall be held at static level during LBIST		137
[SDMMC_AI.001] Timeout error when BOOT ACK driven on all data lines in SD/eMMC mode		137
[SMU_TC.012] Unexpected alarms when registers FSP or RTC are written		138
[SMU_TC.013] Unexpected setting of Alarm Missed Event bit xAEM in Alarm Executed Status register SMU_AEX		139
[SMU_TC.015] SMU alarm emulation might trigger unwanted active alarm reaction	new	140

**Table 3 Parametric deviations**

Issue title	Change	Page
[ADC_TC.P009] Increased TUE for G10 when using Alternate Reference		141
[ADC_TC.P015] Increased RMS noise for TC374* and TC375* devices		141
[CCU_TC.P001] Back-up clock accuracy after trimming - Disregard datasheet footnote		142
[FLASH_TC.P003] Program Flash Erase Time per Multi-Sector Command		142
[PADS_TC.P014] Electrical characteristics for P20.2/ <u>TESTMODE</u>		143
[PORST_TC.P002] $V_{IH}$ and $V_{IL}$ definition for PORST pad - Additional Data Sheet footnote		143

**Table 4 Application hints**

Issue title	Change	Page
[ADC_TC.H026] Additional waiting phase in slow standby mode		144
[ADC_TC.H032] ADC accuracy parameters - Definition		144
[ADC_TC.H033] Basic initialization sequence for primary and secondary EVADC groups		144
[ADC_TC.H035] Effect of input leakage current on Broken Wire Detection		145
[ADC_TC.H043] Information on supervision signal $V_{ANACOMM}$ not relevant - Documentation update		146
[ADC_TC.H044] Start-up calibration timing in synchronized mode - Documentation update		146

**(table continues...)**

## 1 Errata overview

**Table 4** (continued) **Application hints**

Issue title	Change	Page
[ADC_TC.H045] Level selection for broken wire detection feature		147
[AGBT_TC.H004] Configuration of registers PYCR2 and PACR2		147
[AGBT_TC.H005] Availability of AGBT depending on feature package		147
[ASCLIN_TC.H001] Bit field FRAMECON.IDLE in LIN slave tasks		148
[ASCLIN_TC.H006] Sample point position when using three samples per bit		148
[ASCLIN_TC.H007] Handling Tx FIFO and Rx FIFO interrupts in single move mode		148
[ASCLIN_TC.H008] SPI master timing – Additional information to Data Sheet characteristics		149
[BROM_TC.H009] Re-enabling lockstep via BMHD		149
[BROM_TC.H011] Assertion of ALM7[14] after cold or warm PORST		150
[BROM_TC.H012] Availability of $V_{\text{DSD}}$ during start-up		150
[BROM_TC.H014] SSW behavior in case of wrong state or uncorrectable error in UCBs - Documentation Update		150
[BROM_TC.H015] Different initial values for CPU0_PMEM SSH registers in MTU after cold PORST if SOTA/SWAP is enabled		151
[BROM_TC.H020] Processing in case no valid BMHD found		151
[CCU6_TC.H001] CCU6 module clock source information - Documentation Update		152
[CCU_TC.H012] Configuration of the Oscillator- Documentation Update		152
[CLC_TC.H001] Description alignment for bits DISR, DISS, EDIS in register CLC - Documentation Update		152
[CPU_TC.H019] Semaphore handling for shared memory resources		153
[CPU_TC.H021] Resource update failure despite correct SW synchronization upon retried FPI write transactions by CAN and E-Ray modules		155
[CPU_TC.H022] Store buffering and the effect of bit SMACON.IODT		156
[CPU_TC.H023] CPU_SYSCON register safety protection description clarification	new	156
[DAM_TC.H002] Triggering DAM MEMCON.RMWERR and INTERR flags		157
[DMA_TC.H018] Maximum size of circular buffers is 32 Kbytes		157
[DTS_TC.H002] Unexpected alarms after start-up/wake-up when temperature is close to lower/upper limit		157
[EDSADC_TC.H001] Auxiliary filter cleared with start of integration window - Additional information		158
[EDSADC_TC.H004] CIC3 filter properties - Documentation update		158
[EDSADC_TC.H005] Incorrect configuration in the section "Stopping the Integration Window"		158
[EMEM_TC.H006] Triggering EMEM MEMCON.RMWERR and INTERR flags		159
[EMEM_TC.H007] Access restrictions to EMEM for tools		159

**(table continues...)**

## 1 Errata overview

**Table 4** (continued) **Application hints**

Issue title	Change	Page
[EVR_TC.H001] External input capacitor value - Additional Data Sheet footnote		160
[FLASH_TC.H021] Flash Wait State configuration		160
[FLASH_TC.H024] PFLASH erase and program time is affected by time slicing but not clearly documented		160
[FlexRay_AI.H004] Only the first message can be received in External Loop Back mode		161
[FlexRay_AI.H005] Initialization of internal RAMs requires one eray_bclk cycle more		161
[FlexRay_AI.H006] Transmission in ATM/Loopback mode		161
[FlexRay_AI.H007] Reporting of coding errors via TEST1.CERA/B		162
[FlexRay_AI.H009] Return from test mode operation		162
[FlexRay_AI.H010] Driver software must launch CLEAR_RAMs command before reading from E-Ray RAMs		162
[FlexRay_AI.H011] Behavior of interrupt flags in FlexRay™ Protocol Controller (E-Ray)		163
[FlexRay_TC.H003] Initialization of E-Ray RAMs - Documentation update		163
[FlexRay_TC.H004] Bit WRECC in register TEST2 has no function		164
[FlexRay_TC.H005] E-Ray OTGB2 trigger set active even if disabled		164
[FPI_TC.H002] Write access to register ACCEN1		164
[FPI_TC.H003] Burst write access may lead to data corruption		164
[GETH_AI.H001] Preparation for Software Reset		165
[GETH_AI.H003] Undefined behavior when LD bit is set and buffer length B1L or B2L is zero - Additional information		165
[GETH_AI.H004] MAC address 0 configuration sequence		166
[GETH_TC.H002] Stopping and Starting Transmission - Additional information		166
[GPT12_TC.H002] Bits TxUD and TxUDE in incremental interface mode - Additional information		167
[GTM_AI.H425] MCS: Instructions BRDI and BWRI evaluate unused address bits		167
[GTM_AI.H473] SPEC-FIFO: Wrong description of FIFO flush operation		168
[GTM_AI.H480] SPEC-TIM: Wrong action description for TPIM mode		169
[GTM_AI.H481] SPEC-TIM: Wrong description for TBCM mode		170
[GTM_AI.H482] SPEC-TIM: Wrong description in TBCM mode regarding TIM[i]_CH[x]_CTRL.GPR1_SEL bit field		171
[GTM_AI.H497] SPEC-SPE wiring in figure is wrong		171
[GTM_AI.H502] SPEC-DPLL input selection for SUB_INC1 is incomplete		171
[GTM_AI.H515] SPEC-MCS: Incomplete usage of term CPU in MCS chapter	new	172
[GTM_AI.H803] Missing feature aspect in the user manual	new	172
[GTM_TC.H010] Trigger Selection for EVADC and EDSADC		174

**(table continues...)**

## 1 Errata overview

**Table 4** (continued) **Application hints**

Issue title	Change	Page
[GTM_TC.H019] Register GTM_RST - Documentation Update		174
[GTM_TC.H021] Interrupt strategy mode selection in IRQ_MODE		174
[GTM_TC.H027] Register ODA (OCDS Debug Access) - Documentation update		175
[GTM_TC.H034] Correction to figure GTM to MSC Connections, 1st Level Muxes Overview		175
[HSCT_TC.H009] High speed dividers five phase clock sequence ordering		176
[HSCT_TC.H010] Interface control command timing on the LVDS ports		177
[I2C_TC.H008] Handling of RX FIFO Overflow in Slave Mode		178
[I2C_TC.H009] Connections of Serial Clock Inputs		178
[INT_TC.H006] Number of SRNs supporting external interrupt/service requests – Documentation update		178
[LBIST_TC.H003] Update reset behavior of LBISTCTRL0 and LBISTCTRL3 register - Additional information		179
[LBIST_TC.H005] Effects of LBIST execution on P33.8		179
[MBIST_TC.H001] Destructive MBIST requires DSPR0 initialization		179
[MBIST_TC.H002] Time for 4N non-destructive test		180
[MCDS_TC.H007] Program trace of CPUx (x > 0) program start not correct		180
[MCMCAN_AI.H001] Behavior of interrupt flags in CAN Interface (MCMCAN)		181
[MCMCAN_AI.H002] Bus off recovery		181
[MCMCAN_TC.H001] Behavior of undefined data bytes read from Receive Buffer		182
[MCMCAN_TC.H006] Unintended behavior of receive timeout interrupt		182
[MCMCAN_TC.H007] Delayed time triggered transmission of frames		182
[MCMCAN_TC.H008] Parameter “CAN Frequency” - Documentation update to symbol in Data Sheet		183
[MSC_TC.H014] Symbol $T_A$ in specification of FCLPx clock period in Data Sheet - Additional information		183
[MTU_TC.H015] ALM7[0] may be triggered after cold PORST		183
[MTU_TC.H016] MCi_FAULTSTS.OPERR[2] may be triggered at power-up in case LBIST is not run		184
[MTU_TC.H019] Application reset value of register SRC_MTUDONE different to documentation		184
[NVM_TC.H001] References to DMU_HP_PROCONTP – Typo in TC3xx user manual		184
[OCDS_TC.H014] Avoiding failure of key exchange command due to overwrite of COMDATA by firmware		185
[OCDS_TC.H015] System or Application Reset while OCDS and lockstep monitoring are enabled		185

**(table continues...)**

## 1 Errata overview

**Table 4** (continued) **Application hints**

Issue title	Change	Page
[OCDS_TC.H016] Release of application reset via OJCONF may fail		186
[OCDS_TC.H017] CIF not reset by application reset if OCDS is enabled		186
[OCDS_TC.H018] Unexpected stop of Startup Software after system or application reset		186
[PACKAGE_TC.H005] Exposed pad dimensions and package outlines for QFP packages - Updates to TC37xEXT Data Sheet		187
[PADS_TC.H008] Overload coupling for LVDS RX pads – Additional information		188
[PMS_TC.H003] $V_{DDPD}$ voltage monitoring limits		189
[PMS_TC.H008] Interaction of interrupt and power management system - Additional information		190
[PMS_TC.H009] Interaction of warm reset and standby mode transitions		191
[PMS_TC.H011] Supply mode and topology selection - Allowed combinations of VEXT and VDDM - Documentation update		192
[PMS_TC.H018] Bit SWDLVL in register EVRSTAT is always 1 when EVRC is OFF	new	192
[PMS_TC.H019] Limitation of power-cycles - Additional datasheet footnote	new	192
[PORTS_TC.H018] Misleading footnote on pad driver mode selection table		193
[PSI5_TC.H001] No communication error in case of payload length mismatch		193
[QSPI_TC.H008] Details of the baud rate and phase duration control - Documentation update		193
[RESET_TC.H006] Certain registers may have different reset values than documented in TC3xx User's Manual - Documentation update		194
[RESET_TC.H007] Cold Power on Reset Boot Time – Additional information		196
[SAFETY_TC.H013] ESM[SW]:SYS:MCU_FW_CHECK - Access to MC40 FAULTSTS register – Additional information		196
[SAFETY_TC.H017] Safety Mechanisms requiring initialization - Documentation update		197
[SAFETY_TC.H019] SM[HW]:NVM.FSIRAM:REG_MONITOR_TEST should not be considered		199
[SAFETY_TC.H020] Test of SM[HW]:VMT:REG_MONITOR is missing - Documentation update		199
[SCR_TC.H009] RAM ECC Alarms in Standby Mode		200
[SCR_TC.H010] HRESET command erroneously sets RRF flag		200
[SCR_TC.H011] Hang-up when warm PORST is activated during Debug Monitor Mode		200
[SCR_TC.H012] Reaction in case of XRAM ECC Error		200
[SCR_TC.H014] Details on WDT pre-warning period		201
[SCR_TC.H016] SCR current consumption in IDLE mode and 70 kHz clock		201
[SCU_TC.H019] Connection on ERU input E_REQ7(5)		201

**(table continues...)**



## 1 Errata overview

**Table 4** (continued) **Application hints**

Issue title	Change	Page
[SCU_TC.H020] Digital filter on ESRx pins - Documentation update		202
[SCU_TC.H021] LBIST execution affected by TCK/DAP0 state		202
[SCU_TC.H023] Behavior of bit RSTSTAT.PORST after wake-up from standby mode		202
[SCU_TC.H025] Field EEA in register CHIPID - Additional information		202
[SCU_TC.H026] Unexpected alarm ALM0[1] during warm reset		203
[SCU_TC.H027] Bit field INP0 and INP1 in register EICRi - Documentation correction		203
[SCU_TC.H028] ERU configuration changes may lead to ERU reactions	changed	204
[SDMMC_TC.H001] Idle State of SDMMC0_CLK		204
[SENT_TC.H006] Parameter $V_{ILD}$ on pads used as SENT inputs		205
[SENT_TC.H007] Range for divider value DIV - Documentation correction		211
[SMU_TC.H010] Clearing individual SMU flags: use only 32-bit writes		211
[SMU_TC.H012] Handling of SMU alarms ALM7[1] and ALM7[0]		212
[SMU_TC.H013] Increased Fault Detection for SMU Bus Interface (SMU_CLC Register)		212
[SMU_TC.H016] SMU_stdb restriction for using P33.8 as Emergency Stop input		213
[SMU_TC.H017] Handling of ALM21[7] when safety flip-flop self-test is executed		213
[SRI_TC.H001] Using LDMST and SWAPMSK.W instructions on SRI mapped peripheral registers (range 0xF800 0000-0xFFFF FFFF)		213
[SSW_TC.H001] Security hardening measure for the startup behavior		214
[STM_TC.H004] Access to STM registers while STMDIV = 0		215

## **2 Functional deviations**

## **2 Functional deviations**

### **2.1 [ADC\_TC.095] Ramp trigger ignored when ramp ends**

#### **Description**

The Fast Compare Channels can automatically generate ramps (see section “Ramp Mode” in the EVADC chapter). A ramp can be started by a hardware trigger.

- A trigger that occurs while the ramp is running will restart the ramp from its defined starting level
- A trigger that occurs exactly at the time when the ramp is completed (corner case) will be ignored and lead to no action

#### **Workaround**

Make sure the ramp trigger is activated while the ramp is not running (this will be the usual case). Avoid trigger intervals with the same duration as the ramp itself.

### **2.2 [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.

### **2.3 [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 CPUx 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 CPUx, 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 Functional deviations**

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

#### **Description**

If one or more boot mode headers UCB\_BMHDx\_ORIG or UCB\_BMHDx\_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 BMHDx\_ORIG or \_COPY unused in the application, the above fields are in a defined ECC-error free state (for example clear them to 0)

### **2.5 [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.

### **2.6 [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

## 2 Functional deviations

**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).

## 2.7 [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, d5ul ; MUL/MAC writing E0 as result
ld.d e8, [a5] ; Load instruction causing IP bubble
maddm.h e0, e0, d6, d8ul ; 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
```

## 2 Functional deviations

**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).

## 2.8 [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.

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.

## **2 Functional deviations**

### **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.*

## **2.9 [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.

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.

## **2.10 [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 Functional deviations**

### **2.11 [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.

### **2.12 [DAP\_TC.010] Performance when accessing EMEM in UWM and WM modes**

#### **Description**

**Note:** *This problem is only relevant for development tools and their device connection.*

The read bandwidth of DAP for accessing EMEM is about 25 Mbyte/s in Unidirectional Wide Mode (UWM) and Wide Mode (WM) if the DAP is clocked at 160 MHz. This is lower than the target value of 30 Mbyte/s. The bandwidth is limited by the long path from DAP to EMEM.

If the DAP frequency is below 125 MHz, this effect is hidden behind other delays. Therefore below this frequency the bandwidth will be proportional to the DAP frequency. At 125 MHz a bandwidth of 23.5 Mbyte/s was measured in simulation.

The details of WM and UWM are described in the section “DAP Modes and Options” in the OCDS chapter of the device documentation.

**Note:** *The bandwidth measurement is conducted with the BBB frequency of 150 MHz.*

**Note:** *In TC39x, UWM should not be used (see DAP\_TC.008 “DAP Unidirectional Wide Mode (UWM) not working”).*

### **2.13 [DMA\_TC.059] ACCEN Protection not implemented for ERRINTRr**

#### **Description**

In table “Register Overview” of the DMA chapter in the TC3xx user manual, the debug feature Error Interrupt Set Register ERRINTRr for Resource Partition r (r = 0..3) is specified as access enable protected (symbol “Pr” in column Access Mode/Write).

However, register ERRINTRr (r = 0..3) is not implemented as access enable protected.

#### **Workaround**

None.

## **2 Functional deviations**

### **2.14 [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>)
- 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.

### **2.15 [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

### **2.16 [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.



## 2 Functional deviations

### 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 = 0B`
2. Software must halt the DMA channel configured for DMA double buffering
  - a. Software shall write DMA channel `TSRc.HLTREQ = 1B`
  - b. Software shall monitor DMA channel `TSRc.HLTACK = 1B`
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 = 1B`
  - b. Software shall monitor the DMA channel frozen bit `CHCSRc.FROZEN`
5. When the DMA channel has switched buffers (DMA channel `CHCSRc.FROZEN = 1B`)
  - a. If (`SAVED_CH == 1`), software shall trigger a DMA software request by writing DMA channel `CHCSRc.SCH = 1B` to restore DMA channel `TSRc.CH` to the state before the buffer switch
6. Software must unhalt the DMA channel
  - a. Software shall write DMA channel `TSRc.HLTCLR = 1B`
7. Software must enable the servicing of interrupt service requests by the DMA channel
  - a. Software shall write `IR_SRCi.SRE = 1B`

The software must include an error routine.

1. Software must monitor for interrupt overflows (`IR_SRCi.IOV = 1B`) and lost DMA requests (`TSRc.TRL = 1B`)
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

### 2.17 [EMEM\_TC.001] Integrity of EMEM contents when using standby locked mode

#### Description

**Note:** *This problem is only relevant if any data has to be retained in EMEM standby locked mode; otherwise there is no issue.*

If the EMEM is supplied by  $V_{DDSB}$  during standby locked mode while the other parts of the device are unsupplied, at some EMEM addresses the previous content might not be reliable anymore after powering up the device again.

The root cause of this problem is that redundancy information is cleared upon cold PORST.

#### Workaround

For applications that use the standby locked mode of EMEM it is possible to reliably restore the previous contents of the respective EMEM locations via software.

This reliable proven workaround sequence must be executed as soon as standby locked mode is exited (before using the EMEM data in the application or for debug/ calibration purpose).

For each MCx (x = 44,45,46,47 in TC39x, x = 44,45,46 in TC37xEXT):

- Enter MEMTEST mode
- Disable all error notifications in ALMSRCS and ECCS except ECE i.e. only single bit error **correction** is enabled by setting `ECE = 1`, rest of error notifications (`CENE`; `UCENE`; `MENE`) are disabled. ==> This is to prevent additional errors in ECCD or ERRINFO due to the test sequence

## 2 Functional deviations

- Copy the RANGE register to a temporary variable, so that its default value can be restored at the end of the sequence
- Read 8 x 16-bit words from addresses  $MCx\_BASE + 0x20 + 2*i$   $/(i=0-7)$

```
//for x=44-47 (TC39x), x=44-46 (TC37xEXT), for i=0-7
uint32 addr_tmpi = (MCx_BASE + 0x20 + 2*i)
tmp_uint16i = *(volatile uint16*)(addr_tmpi);
```

- If bit 15 = 1:

```
if (tmp_uint16i & 0x8000)
    //NOTE: if bit 15 is 0, then nothing to do
```

- Clear the value:

```
*(volatile uint16*)(addr_tmpi) = tmp_uint16i & 0x7FFF;
```

**Note:** Should be in SV mode with Safety Endinit disabled, ACCEN allowing the master to write to SSH registers; and the access should be a 16-bit write.

- Perform a single read to memory location as explained in the MTU chapter “Reading a Single Memory Location” in the TC3xx User’s Manual:

```
MCx_RANGE.ADDR = tmp_uint16i & 0x7FFF;
```

**Note:** The RANGE.ADDR field to be used for this READ operation should be taken from bits 0-14 of the read value.

The RDBFL registers are updated. Don’t modify the contents since they are used in the later step.

- Set the bit-15 back:

```
*(volatile uint16*)(addr_tmpi) = tmp_uint16i | 0x8000;
```

**Note:** Should be in SV mode with Safety Endinit disabled, ACCEN allowing the master to write to SSH registers; and the access should be a 16-bit write.

- Perform a single write to memory location as explained in the MTU chapter “Writing to a Single Memory Location” in the TC3xx User’s Manual **without modifying the RDBFL register contents and RANGE.ADDR**
- Restore RANGE register from the temporary variable saved above
- Restore the alarm /error notifications in ECCS and ALMSRCS
- Exit MEMTEST mode

## **2 Functional deviations**

### **2.18 [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

### **2.19 [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.

#### **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.

## **2 Functional deviations**

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.

### **2.20 [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.*

### **2.21 [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**

### **2.22 [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.

### **2.23 [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.

### **2.24 [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.

## **2.25 [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**

### **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.

## **2.26 [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.

### **2.27 [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.

### **2.28 [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**

### **2.29 [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.

### **2.30 [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

### **2.31 [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>).

## **2.32 [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.

## **2.33 [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

## **2.34 [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

## **2.35 [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**

### **2.36 [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.

### **2.37 [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**

### **2.38 [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

### **2.39 [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:

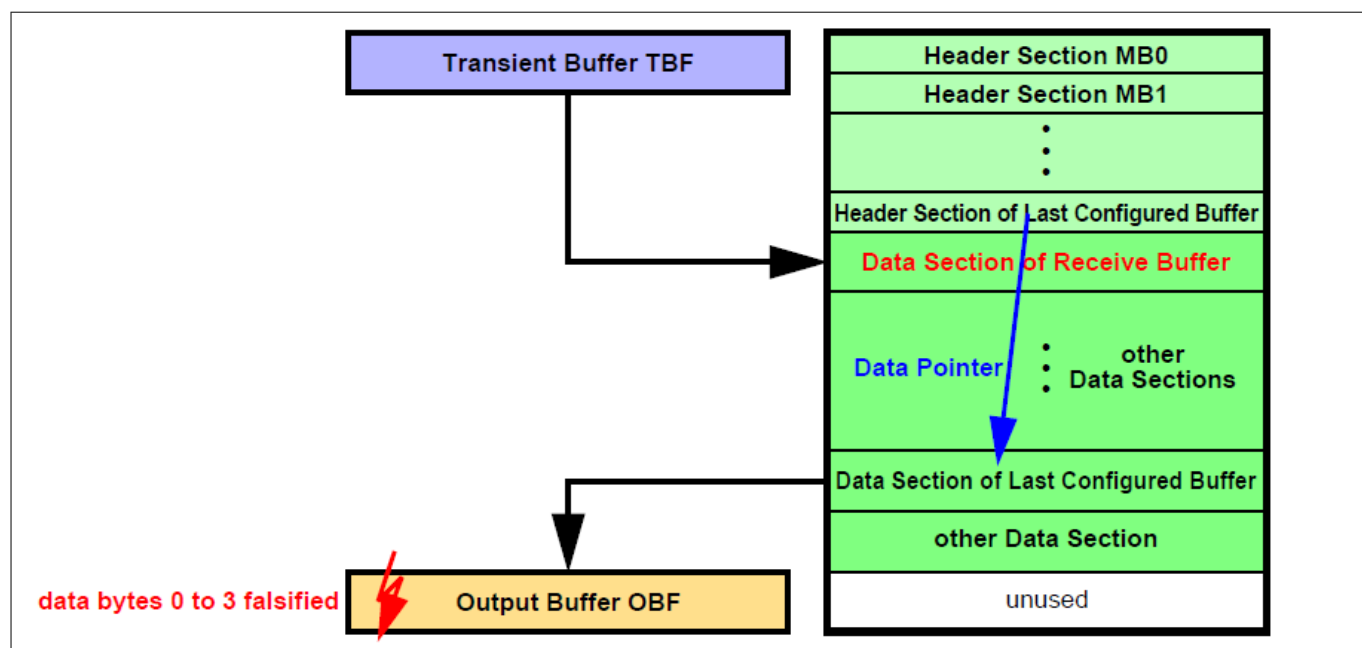
- 1) A received message is transferred from the Transient Buffer RAM (TBF) to the message buffer that has its data pointer pointing to the first word of the Message RAM's Data Partition located directly after the last header word of the Header Partition of the Last Configured Buffer as defined by `MRC.LCB`
- 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.

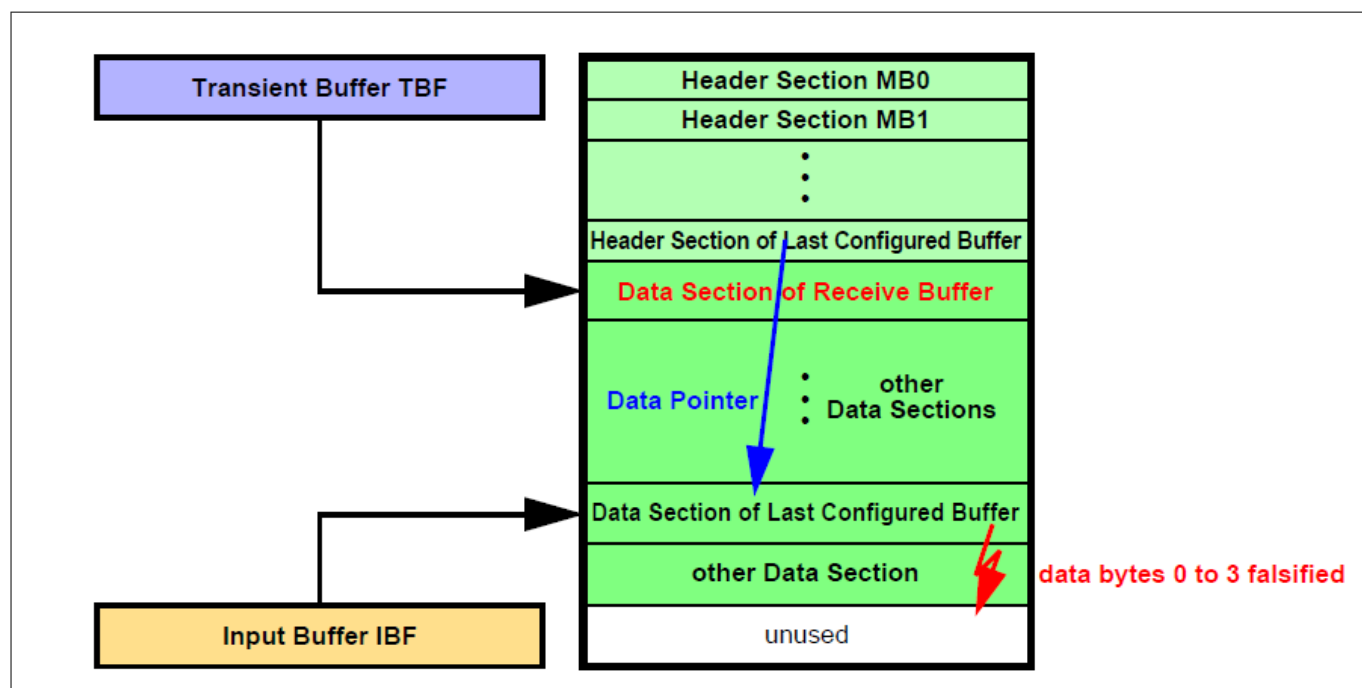
## 2 Functional deviations



**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](#))

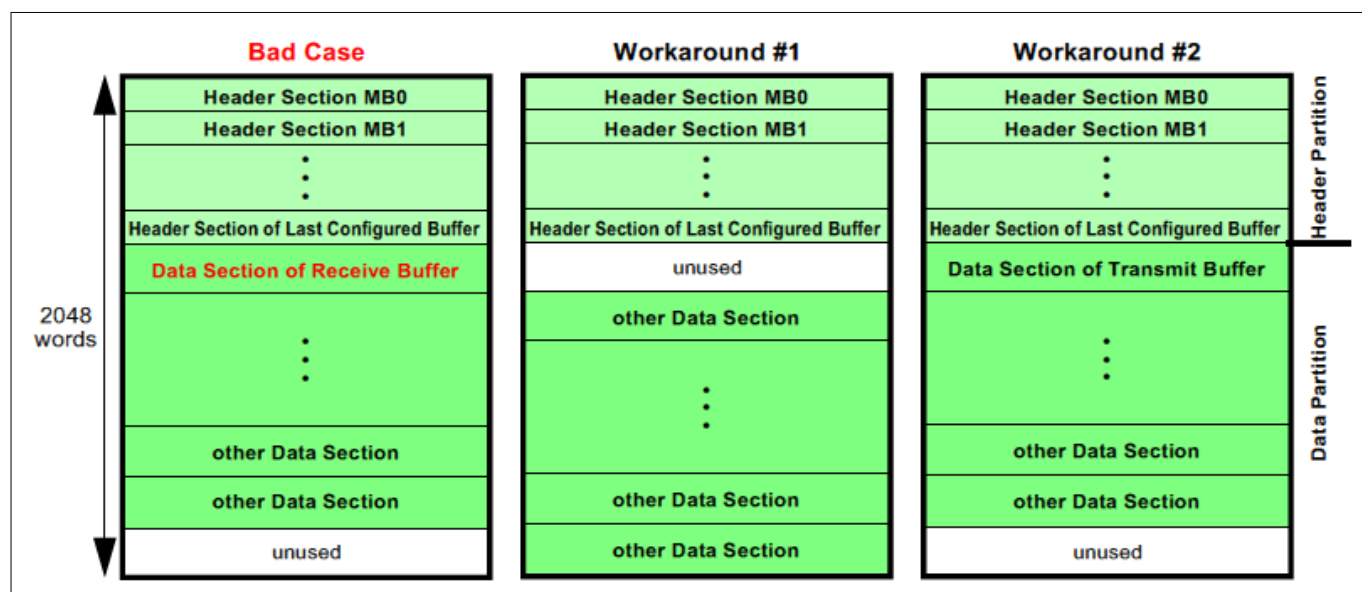
b) When a message is transferred from IBF to the Last Configured Buffer in the Message RAM, it may happen, that at the end of the transfer of the Data Section one additional write access overwrites the following word, which may be the first word of another message's Data Section in the Message RAM (see [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

## 2.40 [GETH\_AI.001] Packets with Destination Address (DA) mismatch are delayed until EOP is received in threshold (cut-through) mode

### Description

For each received packet, Header status is created by the MAC receiver based on the parsing of the Ethernet/VLAN/IP Header fields and forwarded to the DMA. This header status includes information about the size of the L2/L3/L4 header data (SPLIT\_HDR\_EN configurations) and/or the DMA Channel (NUM\_DMA\_RX\_CH>1 configuration) which will forward the packet to host memory. The DA match result would provide DMA channel information based on the DCS field in the corresponding MAC Address Register that matched the DA field.

Due to this defect, instead of waiting for the DA match operation to complete, the design was waiting for a successful DA match to happen. If a DA match did not happen, the Header Status was being generated at the time of receiving the End of Packet (EoP).

The MTL Rx Queue controller waits for the Header status, stores it before it forwards the packet to target Rx DMA. Since the packets without a successful DA match were not getting the header status until the EoP, MTL Rx Queue controller forwards the packet only after the EoP is received in cut through mode.

## **2 Functional deviations**

### **Impacted use cases**

The DA mismatch packets will be forwarded only when Receive All or Promiscuous mode is set. In other use-cases, packets with DA mismatch will get dropped by the MTL Rx Queue controller and never reach the RxDMA.

### **Consequence**

Additional/un-necessary latency is introduced in the transfer of received packets with DA mismatch in the MTL Rx Queue operating in threshold (cut-through) mode. Effectively, it operates in store and forward mode for such packets.

### **Method of Reproducing**

1. Enable Receive All or Promiscuous mode for the receiver by programming MAC\_Packet\_Filter register
2. Enable Threshold (Cut-through) mode and program the threshold value by writing to RSF and RTC fields of MTL\_RxQ<n>\_Operation\_Mode
3. When a packet with a packet length greater than threshold value is received, and a DA match does not happen, the packet will be read out of MTL Rx FIFO only after the EoP is received, while the expected behavior would have been to read the packet after the threshold is crossed

### **Workaround**

None.

## **2.41 [GETH\_AI.008] Application Error Along with Start-of-Packet Can Corrupt the FCS Field of the Previous Frame in the MAC Pipeline**

### **Description**

On the MAC Transmit Interface (MTI) if an application error indication is asserted along with the Start of Packet of a new packet while the MAC is transmitting a packet, the error indication can corrupt the FCS field of the packet being transmitted. This defect manifests because the error indication is inadvertently passed to the MAC transmitter logic directly when sampled along with the Start of Packet indication.

The scenario that causes the problem is:

- Bus error on the first beat of frame data read from the application

### **Impacted use cases**

This issue occurs when Bus Error is received from the system along with the first beat of new packet data, manifesting as error indication and Start of Packet indication asserted simultaneously during an ongoing packet transmission.

### **Consequence**

The packet in transmission is sent with corrupted FCS and therefore the remote end discards it.

### **Workaround**

Discard pending data on bus error and re-init the GETH.

## **2.42 [GETH\_AI.009] Corrupted Rx Descriptor Write Data**

### **Description**

Packets received by DWC\_ether\_qos are transferred to the system memory address space as specified in the receive descriptor prepared by the software. After transferring the packet to the system memory, DWC\_ether\_qos updates the descriptor with the packet status.

## **2 Functional deviations**

However, due to a defect in the design, the Rx packet status gets corrupted when the MTL Rx FIFO status becomes empty during the packet status read. This can happen only when the MTL Rx FIFO is in Threshold (cut through) mode and Frame based arbitration is enabled on the receive.

### **Impacted use cases**

The defect is applicable when the Rx FIFO is in Threshold (Cut-through) mode and Frame based arbitration is enabled in the Rx FIFO.

MTL Rx FIFO working in cut-through mode (bit[5], RSF in MTL\_RxQ[n]\_Operation\_Mode register is set to 0, the default value) and

MTL Rx FIFO is enabled to work in Frame Based Arbitration (bit[3], RXQ\_FRM\_ARBIT in MTL\_RxQn\_Control register is set to 1.

### **Consequence**

The Rx packet status written into the descriptor for the affected packet is corrupt. All subsequent frames are processed as expected.

### **Workaround**

Do not use cut through OR/AND do not use RX arbitration.

## **2.43 [GETH\_AI.010] Fatal Bus Error Interrupt Might Be Generated for Incorrect DMA Channel**

### **Description**

When a bus error occurs, the status reflects the associated RX DMA channel number.

When the current burst or packet transfer is about to end, the MTL arbiter might grant access to another Rx DMA channel for the next burst or packet transfer (with ari\_chnum signal indicating the channel number of Rx DMA that is granted latest access).

However due this defect, when bus error occurs towards end of current burst, the DMA might associate it with Rx DMA channel of next burst (based on the ari\_chnum) and provide the incorrect Rx DMA channel number in the status register.

### **Impacted use cases**

Cases where the MTL arbiter has already granted access to another Rx DMA channel for next burst transfer and bus error occurs for current burst.

### **Consequence**

A wrong Rx DMA channel number is reported for the Fatal Bus Error interrupt.

### **Workaround**

Discard pending data on bus error and re-init the GETH. Debugger can not rely on DMA Status register after bus error of a RX Burst.

## **2.44 [GETH\_AI.011] Receive Queue Overflow at End of Frame Along with SPRAM Read-Write Conflict Can Cause Data Loss**

### **Description**

Read and write operations can conflict, based on the address being read and written. During a conflict in the MTL Receive FIFO, the read operation gets priority and the write operation is retried in the subsequent cycle.



## **2 Functional deviations**

When End of Frame (EoF) is received, the MTL Receiver computes FIFO overflow condition based on the anticipated space needed to write End of Frame (EoF) and RxStatus. When EoF is received on MRI interface and a read-write conflict occurs in the SPRAM for the EoF write along with a FIFO overflow computation, it causes the MTL Receive FSM to malfunction.

### **Impacted use cases**

This issue occurs when the MTL Receive FIFO has a read-write conflict and the Rx FIFO computes an overflow condition upon receiving EoF in the MRI interface.

### **Consequence**

The packet that causes MTL FIFO overflow is handled correctly. However due to the malfunctioning of MTL receive FSM, the subsequent packet loses a part of the data at the beginning of the frame.

### **Workaround**

Discard pending data on bus error and re-init the GETH.

## **2.45 [GETH\_AI.012] Incorrect Flexible PPS Output Interval When Fine Time Correction Method is Used**

### **Description**

The MAC provides programmable option, fine and coarse, for correcting the IEEE 1588 internal time reference. When coarse correction method is used, the correction is applied in one shot and does not affect the flexible PPS output.

When fine correction method is used, the correction is applied uniformly and continuously to the IEEE 1588 internal time reference as well as to the flexible PPS output.

However, due to this defect, when fine correction method is used and the drift in the frequency of the clock that drives the IEEE 1588 internal time reference is large (when compared with the grandmaster source clock), the flexible PPS output interval is incorrect. This does not impact the IEEE 1588 internal time reference updates.

The internal PPS counter used for generating the PPS interval is incorrectly reset earlier than expected, resulting in the next PPS cycle starting incorrectly, earlier than expected.

### **Impacted use cases**

The Flexible PPS Output feature is used in Pulse Train mode and the Fine Correction method is used for correcting the IEEE 1588 internal time reference due to drift in the frequency of the clock that drives it.

### **Consequence**

The incorrect Flexible PPS Output Interval from the MAC can cause the external devices, that are synchronized with flexible PPS trigger outputs, to go out of synchronization.

### **Workaround**

The application can use coarse method for correcting the IEEE 1588 internal time reference. Because, in the coarse correction method, as the time correction is applied in a single shot, timestamp captured for at the most one packet is impacted. This might be the case when current cycle of time-synchronization related packet-exchanges coincides with the coarse time correction of previous cycle. This discrepancy is corrected in the next time-synchronization correction cycle.

## **2 Functional deviations**

### **2.46 [GETH\_AI.013] False Dribble and CRC Error Reported in RMII PHY 10 Mbps Mode**

#### **Description**

The MAC receiver clock is derived synchronously from RMII REF\_CLK, the frequency is 2.5 MHz in 10 Mbps speed mode and 25 MHz in 100 Mbps speed mode. In 10 Mbps mode, the 2-bit RMII data is captured every 10 cycles of RMII REF\_CLK, combined and provided as 4-bit data on the MAC receiver clock. As per RMII protocol, the RMII CRS\_DV is asserted asynchronously with RMII REF\_CLK, which also implies that it is asynchronous to the MAC receiver clock. The MAC correctly captures the received packet irrespective of the phase relation between RMII CRS\_DV assertion and MAC receiver clock.

However due to this defect, in the 10 Mbps speed mode, when the RMII CRS\_DV is asserted two RMII REF\_CLK rising edges ahead of MAC receiver clock, the MAC reports false dribble and CRC error in the Receive status. The dribble error is reported when MAC receives odd number of nibbles (4-bit words) and CRC error is additionally reported. In this case the additional nibble captured is a repetition of the last valid nibble. The MAC forwards only the data received on byte boundaries to the software and ignores the extra nibble. Therefore, there is no data loss or corruption of packet forwarded to the software. However, if error-packet drop is enabled (FEP bit in MTL\_RxQ(#i)\_Operation\_Mode register is set to 0), MAC drops the packets, causing packet loss and impacts the performance.

#### **Impacted use cases**

The RMII PHY interface is enabled for 10Mbps operation and RMII CRS\_DV is asserted two RMII REF\_CLK rising edges ahead of MAC receiver clock.

#### **Consequence**

The MAC reports false dribble and CRC error in Receive status. If error-packet drop is enabled, (FEP bit in MTL\_RxQ(#i)\_Operation\_Mode register is set to 0), MAC drops the packets causing packet loss and impacts the performance. If the error-packet drop is disabled (FEP bit in MTL\_RxQ(#i)\_Operation\_Mode register is set to 1), MAC forwards the packet to the software, up to the byte boundary, and there is no data loss or corruption.

#### **Workaround**

If error-packet drop is enabled (FEP bit in MTL\_RxQ(#i)\_Operation\_Mode register is set to 0), software can disable it and take the dropping decision based on the Rx status. If the dropping of error packets is disabled (FEP bit in MTL\_RxQ(#i)\_Operation\_Mode register is set to 1), software can ignore the dribble and CRC error and accept packets that have both these errors together. The occurrence of real dribble error is rare and happens when there are synchronization issues due to faulty clock recovery.

### **2.47 [GETH\_AI.014] Receive DMA Channel Generates Spurious Receive Watchdog Timeout Interrupt**

#### **Description**

Programming the RWT field in DMA\_CH(#i)\_Rx\_Interrupt\_Watchdog\_Timer register to a non-zero value enables the Receive DMA for generating the Receive Watchdog Timeout Interrupt. The RWTU field in the same register is used for selecting the units (in terms of number of system clock cycles) for the value programmed in the RWT field. The Receive Watchdog timer starts when the RWT field is programmed to a non-zero value, and if the Receive descriptors corresponding to the packet does not have the completion interrupt enabled. When the timer equals the programmed number of system clock cycles, Receive DMA sets the Receive Interrupt status (RI bit in DMA\_CH(#i)\_Status register). The interrupt is generated on sbd\_intr\_o or sbd\_perch\_rx\_intr\_o[i] based on the INTM field in DMA\_Mode register, when both RIE and NIE bits in DMA\_CH(#i)\_Interrupt\_Enable register are set to 1.

However due to the defect, when the non-zero value programmed in the RWTU field (timer programmed to count in units of 512, 1024, or 2048 system clock cycles) reaches the timer logic earlier than the value programmed in RWT field, a spurious Receive Watchdog Timeout Interrupt is generated. This is because the

## 2 Functional deviations

logic incorrectly checks for concatenation of RWTU and RWT fields to be non-zero instead of checking only the RWT field; this triggers the comparison of RWT field with timer bits shifted left by the value in the RWTU field. As the timer has not started, its initial value of zero matches the default value of zero of the RWT field, which incorrectly sets the Receive Interrupt status (RI bit in DMA\_CH(#i)\_Status register). The interrupt is generated on `sbd_intr_o` or `sbd_perch_rx_intr_o[i]` based on INTM field in DMA\_Mode register when both RIE and NIE bits in DMA\_CH(#i)\_Interrupt\_Enable register are set to 1.

The delay in the programmed value of RWT field reaching the timer logic with respect to programmed value in RWTU field can be due to following reasons:

1. The software performs a byte-wide write with byte containing RWTU field written first
2. The software performs a 32-bit wide write access, but two separate writes are performed, first one to program RWTU field and second one to program RWT field. This may not be an efficient use case and is not widely used
3. The software performs a 32-bit wide write access and writes both RWTU and RWT fields together, but there is different synchronization delay from CSR clock domain to system clock domain on both these paths (the configurations in which `DWC_EQOS_CSR_SLV_CLK` is selected)

The issue is not observed when:

1. A zero value is written in RWTU field (timer programmed to count in units of 256 system clock cycles) and non-zero value is written in RWT field
2. A single write access with non-zero value written in RWTU field (timer programmed to count in units of 256 system clock cycles) and non-zero value written in RWT field

This issue does not have any functional impact; on receiving the spurious Receive Watchdog Timeout Interrupt the software triggers processing of received packets, it does not find any Receive descriptor closed by the Receive DMA and exits the Interrupt Service Routine (ISR). This has a very insignificant impact on the software performance.

### Impacted Use Cases

The completion interrupt is not enabled in Receive Descriptors and periodic Receive Watchdog Timeout Interrupt is enabled (timer programmed to count in units of 512, 1024, or 2048 system clock cycles) by software for bulk processing of the received packets.

### Consequence

The software enters the Interrupt Service Routine (ISR) to process the received packets. But it does not find any received packet to process and exits, which has very insignificant impact on the software performance.

### Workaround

1. When the software performs a byte-wide write, the byte containing RWT field must be written prior to the byte containing RWTU field
2. When the software performs a 32-bit wide write access, but two separate writes are performed to program RWTU field and RWT field, the RWT field must be written prior to the RWTU field
3. When the software performs a 32-bit wide write access and writes both RWTU and RWT fields together, two separate writes must be performed; RWT field must be written prior to the RWTU field

## 2.48 [GETH\_AI.015] MAC Receive VLAN Tag Hash Filter Always Operates in Default Mode

### Description

The ETV, DOVLTC, and ERSVLM bits of the MAC\_VLAN\_Tag (Extended Receive VLAN filtering is not selected) or MAC\_VLAN\_Tag\_Ctrl (Extended Receive VLAN filtering is selected) register are used to program the mode of operation of the Receive VLAN Hash Filtering. The ETV bit is used to enable computation of Hash for only 12 bits of VLAN Tag. The DOVLTC bit is used to disable VLAN Type Check for VLAN Hash filtering. The ERSVLM bit is used to enable VLAN Hash filtering for S-VLAN Type.

## **2 Functional deviations**

However, due to this defect, the Receive VLAN Hash filter always operates in default mode, that is, VLAN Hash is computed for 16-bits (ETV=0) of C-VLAN Tag (DOVLTC=0 and ERSVLM=0). Therefore, programming of ETV, DOVLTC, or ERSVLM bits to 1 do not take effect because these bits have incorrect read-only attribute.

As a result, unintended packets might be forwarded to the application due to incorrect filter pass or bypass results/status. Also, packets might be dropped in the MAC due to incorrect filter fail result.

### **Impacted use cases**

The defect is applicable when non-default VLAN Hash filtering modes are programmed, that is, one or more of ETV, DOVLTC, and ERSVLM bits are set to 1.

### **Consequence**

Forwarding unintended packets to the application can lead to performance degradation in the software stack due to additional processing overhead. Dropping unintended packets results in packet loss requiring retransmission, which never succeeds. This again leads to performance degradation. This is a static issue and the software can avoid it by following the procedure mentioned in the Workaround section.

### **Workaround**

The software should disable the Receive VLAN Hash filtering by setting the VTHM bit of the MAC\_VLAN\_Tag\_Ctrl register to 0 (when non-default VLAN Hash filtering mode is required) and use the alternative filtering methods available in the hardware (perfect or inverse VLAN filtering) or perform filtering in software.

## **2.49 [GETH\_AI.016] Receive DMA Header Split Function Incorrectly Overruns the Allocated Header Buffer**

### **Description**

When the Header Split function is enabled, the DWC\_ether\_qos identifies the boundary between Header (L2 layer Header or TCP/IP Header) and the payload and stores the header and payload data in separate buffers in the host memory. The size of the allocated Header buffer depends on the HDSMS field in the MAC\_Ext\_Configuration register expressed in terms of Data-width. When the buffer address start address is not aligned to the Data-width, the Receive DMA writes that many lesser bytes in the allocated Header buffer. If the Header cannot be accommodated in allocated Header buffer, the Receive DMA indicates in the status that the packet data is not split into header and payload buffer.

However, due to this defect, when the Header buffer start address is not aligned to the Data-width the Receive DMA Header Split function incorrectly overruns the allocated Header buffer. The overrun happens only when the Header size in the received packet is equal or less (by up to the number of bytes which could not be written due to buffer start offset) than the HDSMS field in MAC\_Ext\_Configuration register.

### **Impacted use cases**

The Header Split function is enabled and the Header buffer start address is not aligned to the Data-width.

### **Consequence**

The bytes written beyond the allocated buffer corrupts the data at that location in memory.

### **Method of Reproducing**

Program the SPH bit in DMA\_CH(#i)\_Control register to 1, to enable the Split Header function in Receive DMA.  
Program the HDSMS field in MAC\_Ext\_Configuration register to 0, to enable splitting of headers up to 64 bytes.  
Set up Receive descriptor with Header buffer start address offset of 1.  
Generate and send receive packet with a header size of 64 bytes.  
Observe that the last byte of the header is written beyond allocated Header buffer.

## **2 Functional deviations**

### **Workaround**

The software should always allocate header buffers with start address aligned to Data-width (64 bit) or the HDSMS field in MAC\_Ext\_Configuration register should be programmed to a value larger than the largest expected Header size in receive packet by a number of bytes equal or more than one Data-width (aligned to 64 bit).

### **2.50 [GETH\_AI.017] Carrier-Sense Signal Not Generated When False Carrier Detected in RGMII 10/100 Mbps Mode**

#### **Description**

The RGMII PHY interface generates the carrier-sense signal (CRS) when a packet is transmitted, or when a packet, carrier extension, carrier extend error, carrier sense, or false carrier is received. The CRS is used for generating the COL (Collision) signal in half-duplex mode, when transmission and reception occur simultaneously, or for deferring the transmission.

However, due to the defect, when false carrier is detected in RGMII 10/100 Mbps mode, CRS is not generated. This is because the logic incorrectly checks for alternate 0xE and 0x0 pattern as expected in 1000 Mbps mode instead of the expected continuous 0xE pattern in 10/100 Mbps mode.

#### **Impacted use cases**

The RGMII PHY interface is enabled and operated in 10/100 Mbps speed mode.

#### **Consequence**

In Full-Duplex mode, when ECRSFD bit of the MAC\_Configuration register is programmed to 1, MAC does not defer the transmit packet when false carrier is received. This can result in loss of transmitted packet, requiring retransmission.

In Half-Duplex mode, the MAC does not defer the transmit packet because CRS is not generated when false carrier is received. This results in collision; as COL signal is not generated, the MAC transmitter incorrectly considers successful transmission of the packet. The corresponding MMC counters are incorrectly updated in configurations where MMC counters are selected.

#### **Method of reproducing**

Enable RGMII PHY interface (GPCTL.EPR = 001<sub>B</sub>).

Enable 100 Mbps mode by programming both PS and FES bits of the MAC\_Configuration register to 1'b1.

In Full-Duplex transmission mode, enable Carrier Sense by programming ECRSFD field of the MAC\_Configuration register to 1'b1.

Generate and send multiple back-to-back packets from MAC transmitter.

Send false carrier (0xE) pattern to the MAC receiver while packet transmission is in progress.

Observe that the MAC transmitter does not defer the packet transmission when false carrier pattern is received.

#### **Workaround**

None required; false carrier error occurs rarely. The application layer detects the loss of packet and triggers retransmission.

### **2.51 [GETH\_TC.002] Initialization of RGMII interface**

#### **Description**

If RGMII mode (GETH\_GPCTL.EPR = 001) is configured and GREFCLK (Gigabit Reference Clock) is running during initialization (including a kernel reset), a persistent communication failure may occur due to an internal

## 2 Functional deviations

synchronization issue, resulting in a phase shift of the Transmit Clock TXCLK relative to TXD/TXCTL of  $\pm 180^\circ$  (@1000Mbit/s), or  $\pm 36^\circ$  (@100Mbit/s), or  $\pm 3.6^\circ$  (@10Mbit/s).

**Note:** For MII and RMII see Application Hint GETH\_AI.H001.

### Workaround

After the required I/O settings have been configured (see also section “IO Interfaces” in the GETH chapter of the TC3xx User’s Manual) and the module clock is enabled and GREFCLK and RXCLK are running, follow the initialization sequence listed below:

- Finish active transfers and make sure that transmitters and receivers are set to stopped state:
  - Check the RPSx and TPSx status bit-fields in register DMA\_DEBUG\_STATUS0/1
  - Check that MTL\_RXQ0\_DEBUG, MTL\_RXQi\_DEBUG, MTL\_TXQ0\_DEBUG and MTL\_TXQi\_DEBUG register content is equal to zero.  
Note: it may be required to wait  $70 f_{SPB}$  cycles after the last reset before checking if RXQSTS in MTL\_RXQ0\_DEBUG and MTL\_RXQi\_DEBUG are zero
  - Wait until a currently running interrupt is finished and globally disable GETH interrupts
- Ensure GETH\_GPCTL.EPR = 000<sub>B</sub>
- Ensure GETH\_SKEWCTL = 0x0
- Apply a kernel reset to the GETH module:
  - Deactivate Endinit protection, as registers KRST0/1 and KRSTCLR can only be written in Supervisor Mode and when Endinit protection is not active.  
Write to corresponding RST bits of KRST0/1 registers to request a kernel reset. The reset status flag KRST0.RSTSTAT may be cleared afterwards by writing to bit CLR in the KRSTCLR register.  
Re-activate Endinit protection
  - Wait  $35 f_{SPB}$  cycles
- Set GETH\_GPCTL.EPR = 001<sub>B</sub>(RGMII)
- Setup GETH\_SKEWCTL if required
- Perform a software reset by writing to the DMA\_MODE.SWR bit. Wait  $4 f_{SPB}$  cycles, then check if DMA\_MODE.SWR = 0<sub>B</sub>
- Configure remaining GMAC registers according to application requirements

## 2.52 [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 Functional deviations

### 2.53 [GTM\_AI.262] SPEC-DPLL: PSSC/PSTC behavior description incorrect

#### Description

When changing from normal to emergency mode (DPLL\_CTRL\_0.RMO = 0->1), the RAM1b.PSSC value is not calculated as specified.

When changing from emergency to normal mode (DPLL\_CTRL\_0.RMO = 1->0), the RAM1b.PSTC value is not calculated as specified.

- In the specification it is written:
  - For changing from normal mode to emergency mode at the following TRIGGER slope (according to the RMO value in the shadow register)<sup>1)</sup> the PSSC value is calculated by  $PSSC = PSSM + \text{correction value}$  (forward direction) or  $PSSC = PSSM - \text{correction value}$  (backward direction) with the correction value =  $\text{inc\_cnt1} - \text{nmb\_t}$ .
  - For changing from emergency mode to normal mode at the following STATE slope (according to the RMO value in the shadow register)<sup>2)</sup> the PSTC value is calculated by  $PSTC = PSTM + \text{correction value}$  (forward direction) or  $PSTC = PSTM - \text{correction value}$  (backward direction) with the correction value =  $\text{inc\_cnt1} - \text{nmb\_s}$ . In case no further TRIGGER or STATE events the CPU has to perform the above corrections.
- Instead the following behavior should be specified:
  - For changing from normal mode to emergency mode at the following STATE slope (according to the RMO value in the shadow register) the PSSC value is calculated by  $PSSC = PSSM + \text{correction value}$  (forward direction) or  $PSSC = PSSM - \text{correction value}$  (backward direction) with the correction value =  $\text{inc\_cnt1} - \text{nmb\_s}$ .
  - For changing from emergency mode to normal mode at the following TRIGGER slope (according to the RMO value in the shadow register) the PSTC value is calculated by  $PSTC = PSTM + \text{correction value}$  (forward direction) or  $PSTC = PSTM - \text{correction value}$  (backward direction) with the correction value =  $\text{inc\_cnt1} - \text{nmb\_t}$ . In case no further TRIGGER or STATE events the CPU has to perform the above corrections.

#### Scope

DPLL

#### Effects

The PSSC value is not updated at the following trigger slope but at the following STATE slope with the value  $PSSC = PSSM \pm \text{correction value}$  with correction value =  $\text{inc\_cnt1} - \text{nmb\_s}$ . This might lead to an unexpected behavior of the GTM IP.

The PSTC value is not updated at the following STATE slope but at the following TRIGGER slope with the value  $PSTC = PSTM \pm \text{correction value}$  with correction value =  $\text{inc\_cnt1} - \text{nmb\_t}$ . This might lead to an unexpected behavior of the GTM IP.

#### Workaround 1

If possible leave PSSC or PSTC as is.

If a different value for PSSC/PSTC is necessary, the value could be written by CPU interface as already written in the specification. Starting with version 3.1.5 the modification could be done by MCS0 as well.

#### Workaround 2

The application can disable the DPLL through DPLL\_CTRL\_1.DEN=0 and then re-start the DPLL in emergency or normal mode with the following sequence:

Setting up the DPLL modes as desired, for example (not mandatory values):

<sup>1</sup> Stored in an independent shadow register for an active TRIGGER event and for DEN = 1

<sup>2</sup> Stored in an independent shadow register for an active STATE event and for DEN = 1

## **2 Functional deviations**

GTM\_DPLL\_CTRL\_1.B.PIT = 1; -- only as an example  
GTM\_DPLL\_CTRL\_1.B.DMO = 0; -- only as an example  
GTM\_DPLL\_CTRL\_1.B.COA = 0; -- only as an example  
GTM\_DPLL\_CTRL\_1.B.SYSF = 1; -- only as an example  
GTM\_DPLL\_CTRL\_1.B.TSL = 1; -- only as an example  
GTM\_DPLL\_CTRL\_1.B.SSL = 3; -- only as an example  
GTM\_DPLL\_CTRL\_0.B.SEN/TEN = 1; -- only as an example  
Then switch into emergency/normal mode and enable the DPLL again.  
GTM\_DPLL\_CTRL\_0.B.RMO = 1 or 0; -- only as an example  
GTM\_DPLL\_CTRL\_0.B.SEN/TEN = 1; -- only as an example  
GTM\_DPLL\_CTRL\_1.B.DEN = 1; -- only as an example  
In this case the behavior for PSSC/PSTC is different:  
For DPLL\_CTRL\_0.RMO = 0->1: PSSC=PSSM.  
For DPLL\_CTRL\_0.RMO = 1->0: PSTC=PSTM.

If in this case nevertheless a different value for PSSC/PSTC is necessary the value could be written by the CPU interface as already written in the specification. Starting with version 3.1.5 the modification could be done by MCS0 as well.

### **2.54 [GTM\_AI.263] DPLL: DPLL\_STATUS.LOCK1 flag (0 ->1) delayed after direction change when DPLL operating in DPLL\_CTRL\_0.RMO = 1**

#### **Description**

The DPLL\_STATUS.LOCK1 flag does not behave like requested in the specification:  
When the DPLL is operating in emergency mode (DPLL\_CTRL\_0.RMO = 1) and a direction change happens the lock1 flag is reset to "0" as described in the specification.  
The problem is, that the LOCK1 bit is set to "1" again after 4 detected gaps (missing state irq, ms -flag) and not as requested after 2 subsequent gaps.

#### **Scope**

DPLL

#### **Effects**

When the DPLL is operating in emergency mode (DPLL\_CTRL\_0.RMO = 1) and a direction change happens the LOCK1 flag is reset to "0" as described in the specification.  
The problem is, that the LOCK1 bit is set to "1" again after 4 detected gaps (missing state irq, ms -flag) and not as requested after 2 subsequent gaps.

#### **Workaround**

If you need to use this information one could observe the missing state interrupt to check for the correct point in time when the LOCK1 flag should be set again.  
If the use of the LOCK1 flag is not time critical it could be used as is with the latency described above.



## **2 Functional deviations**

### **2.55 [GTM\_AI.304] MCS: Scheduling modes Single Prioritization and Multiple Prioritization are not functional**

#### **Description**

If an MCS instance is configured with the Single or Multiple Prioritization Scheduling mode and the last non-suspended and prioritized MCS channel (CLP) is entering its suspended state (which means that the MCS starts scheduling the remaining non-prioritized channels with accelerated scheduling scheme) and if the suspended state of CLP is resumed five clock cycles after it was entering the suspended state the MCS channel CLP is not executing the instruction that is following the suspending instruction.

#### **Scope**

MCS

#### **Effects**

The program execution of a prioritized MCS channel can skip an instruction that is directly following a suspending instruction.

#### **Workaround**

Add an additional NOP instruction after all suspending instructions (WURM, WURMX, WURCX, WUCE, ARD, ARDI, NARD, NARDI, AWR, AWRI, BRD, BRDI, BWR, and BWRI) in a prioritized MCS program.

### **2.56 [GTM\_AI.306] DPLL: DPLL\_NUTC.syn\_t\_old, DPLL\_NUSC.syn\_s\_old not updated according specification**

#### **Description**

The DPLL specification defines for DPLL\_NUTC.WSYN=1 that an update of register DPLL\_NUTC allows writing of the bits DPLL\_NUTC.syn\_t while DPLL\_NUTC.syn\_t\_old inherits the previous value of DPLL\_NUTC.syn\_t.

Differing from the specified behavior the actual hardware does not update the value of DPLL\_NUTC.syn\_t\_old with the previous value of DPLL\_NUTC.syn\_t but instead updates DPLL\_NUTC.syn\_t\_old according to the corresponding bits of the write operation executed by the CPU.

The DPLL specification defines for DPLL\_NUSC.WSYN=1 that an update of register DPLL\_NUSC allows writing of the bits DPLL\_NUSC.syn\_s while DPLL\_NUSC.syn\_s\_old inherits the previous value of DPLL\_NUSC.syn\_s.

Differing from the specified behavior the actual hardware does not update the value of DPLL\_NUSC.syn\_s\_old with the previous value of DPLL\_NUSC.syn\_s but instead updates DPLL\_NUSC.syn\_s\_old according to the corresponding bits of the write operation executed by the CPU.

#### **Scope**

DPLL

#### **Effects**

The registers bits DPLL\_NUTC.syn\_t\_old are not updated with the previous value of DPLL\_NUTC.syn\_t but by the bits of the input data word.

The registers bits DPLL\_NUSC.syn\_s\_old are not updated with the previous value of DPLL\_NUSC.syn\_s but by the bits of the input data word.

#### **Workaround**

If the update of syn\_t/s\_old shall be done like described in the specification the register DPLL\_NU(T/S)C.syn\_t/s must be read first, then the DPLL\_NU(T/S)C.syn\_(t/s) can be used to modify the bits which are written to DPLL\_NU(T/S)C.syn\_(t/s)\_old.

## **2 Functional deviations**

As the current behavior of DPLL\_NUT/SC.syn\_s/t\_old is in use by and can be advantageous for certain applications, there is no intend to change the current hardware behavior at this point in time. Instead a specification update to align the specification with the current hardware behavior is planned for future GTM generations.

### **2.57 [GTM\_AI.307] IRQ: AEI\_IM\_ADDR is not set in GTM\_IRQ\_NOTIFY register if cluster 0 is disabled**

#### **Description**

Bit 2 - AEI\_IM\_ADDR - in register GTM\_IRQ\_NOTIFY is not set and the related error interrupt (if enabled) does not occur if cluster 0 is disabled and

- an FPI read or write access to a cluster 0 register OR
- an illegal FPI write access to any enabled cluster is done

In case BRIDGE\_MODE.MSK\_WR\_RSP = 0x0 (not recommended) an FPI bus error is issued for all write accesses.

#### **Scope**

IRQ

#### **Effects**

See description above.

#### **Workaround**

- Do not disable cluster 0
- If cluster 0 is disabled: after each WRITE access check register GTM\_AEI\_STA\_XPT; if the read value is >0, it signs an access error
- If cluster 0 is disabled: do not read from cluster 0 register or any other disabled cluster register

### **2.58 [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

## **2 Functional deviations**

### **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.

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

### **2.59 [GTM\_AI.318] MCS: NARD(I) instruction terminates unexpectedly**

#### **Description**

If the bit-field CFG\_CLK\_RATE of register CCM[i]\_HW\_CONF is set and an MCS runs on a cluster i while bit-field CLSc\_CLK\_DIV of register GTM\_CLS\_CLK\_CFG is set to 1, the execution of a NARD or NARDI instruction might signalize an unsuccessful data transfer (bit-field SAT of internal MCS register STA is cleared) although the data source has data available for sending. The unexpected behavior depends on the current state of the internal ARU counter.

#### **Scope**

MCS

#### **Effects**

The available data from the ARU source is not sent via ARU.

#### **Workaround**

None. However, typical applications that are polling data sources with the NARD or NARDI instruction do not have to concern about this errata since the polling loop just requires more iterations.

### **2.60 [GTM\_AI.319] (A)TOM: Unexpected (A)TOM\_CCU1TCx\_IRQ in up/down counter mode**

#### **Description**

If the up-down counter mode is activated (bit-field UDMODE of register (A)TOM[i]\_CH[x]\_CTRL is set to a value greater than zero) and the interrupt (A)TOM\_CCU1TCx\_IRQ is enabled (bit-field CCU1TC\_IRQ\_EN of register (A)TOM[i]\_CH[x]\_IRQ\_EN is set), the interrupt signal (A)TOM\_CCU1TCx\_IRQ will be set unexpectedly directly after the interrupt (A)TOM\_CCU0TCx\_IRQ was set and indicates that the counter (A)TOM[i]\_CH[x]\_CN0 reaches (A)TOM[i]\_CH[x]\_CM0.

#### **Scope**

TOM/ATOM

#### **Effects**

Interrupt signal (A)TOM\_CCU1TCx\_IRQ is set unexpectedly.

#### **Workaround**

If the interrupt (A)TOM\_CCU1TCx\_IRQ is needed, it can be disabled by the first occurrence of itself and enabled again with the interrupt (A)TOM\_CCU0TCx\_IRQ. With the following occurrence of the interrupt (A)TOM\_CCU1TCx\_IRQ it will be disabled again and so on.

## **2 Functional deviations**

### **2.61 [GTM\_AI.320] ATOM: Unexpected restart of a SOMS oneshot cycle while ATOM[i]\_CH[x]\_CM0 is zero**

#### **Description**

If ATOM is set to SOMS oneshot mode (bit-field MODE of ATOM[i]\_CH[x]\_CTRL is set to 0b11 and bit-field OSM in register ATOM[i]\_CH[x]\_CTRL is set) a oneshot cycle is started immediately by writing a value unequal to zero to ATOM[i]\_CH[x]\_SR0 register while the value of ATOM[i]\_CH[x]\_CM0 register is zero.

#### **Scope**

ATOM

#### **Effects**

Restarting of a oneshot cycle starts immediately while ATOM[i]\_CH[x]\_CM0 is zero and a write access to ATOM[i]\_CH[x]\_SR0 is executed with a value unequal to zero.

#### **Workaround**

Avoid value 0 in ATOM[i]\_CH[x]\_CM0 register if SOMS oneshot mode is enabled (bit-field OSM in register ATOM[i]\_CH[x]\_CTRL).

### **2.62 [GTM\_AI.322] DPLL: PSTC, PSSC not updated correctly after fast pulse correction completed (DPLL\_CTRL1.PCM1/2 = 0)**

#### **Description**

When additional pulses are requested using DPLL\_CTRL\_11.PCMF1/2=1 AND PCMF1/2\_INCCNT\_B=0 the PSTC/ PSSC parameters as well as NMB\_T/S\_TAR are not updated correctly, because either the amount of additional pulses (MPVAL1/2) are not incremented or NMB\_T/S\_TAR is set to a wrong value.

#### **Scope**

DPLL

#### **Effects**

After the pulse correction is performed the fields NMB\_T/S\_TAR are set to wrong values such that after a new input event the parameters PSTC/PSSC are not updated correctly.

Incorrect PSTC/PSSC values are ending up in wrong NA[i] parameters. These wrong NA[i] values are leading to incorrect PMT calculations.

The pulse generation itself (register DPLL\_INC\_CNT1/2 and the status of the angle clocks TBU\_TS1/2) is correct and not affected by this issue.

#### **Workaround**

Implement the workaround in the TISI interrupt, i.e. start the workaround at the arrival of inactive edge. This ensures that swon\_t/swon\_s is stable and the incorrect nmb\_t\_tar/nmb\_s\_tar has already been generated. This is to ensure the following:

- Start the workaround after the incorrect nmb\_t\_tar/nmb\_s\_tar has been generated and swon\_t/swon\_s is not toggling anymore
- Workaround should be finished before the arrival of next active edge

Workaround steps are as follows:

## 2 Functional deviations

1. Check swon\_t/swon\_s,
  - If swon\_t/swon\_s = 1, save & use nmb\_t\_tar/nmb\_s\_tar for further corrections
  - Else save & use nmb\_t\_tar\_old/nmb\_s\_tar\_old for further corrections
2. Set PCM1=1 to trigger the fast pulse correction with PCMF1 already set to 1
3. Wait for PCM1 to reset to 0
4. Overwrite the nmb\_t\_tar/nmb\_s\_tar or nmb\_t\_tar\_old/nmb\_s\_tar\_old with the correct value based on swon\_t/swon\_s, similarly based on the choice in step 1

After the next active edge, the PSTC/PSSC values are corrected.

### 2.63 [GTM\_AI.323] DPLL: Registers DPLL\_NUTC.SYN\_T and DPLL\_NUSC.SYN\_S are updated by the profile (ADT\_T.NT/ADT\_S.NS) before the DPLL is synchronized (DPLL\_STATUS.SYT/S=0)

#### Description

The registers DPLL\_NUTC.SYN\_T and DPLL\_NUSC.SYN\_S as well as the corresponding \*\_OLD registers are updated unexpectedly by the profile (ADT\_T.NT/ADT\_S.NS) before the DPLL is synchronized (DPLL\_STATUS.SYT/S=0).

This is not a problem for the calculation of the number of pulses (nmb\_t/s,...), due to the fact that the correct value of SYN\_T/S for the internal use is determined by the signal DPLL\_STATUS.SYT/S. The microtick generation of the DPLL is not affected by this bug.

This problem is only relevant if the SYN\_T/S values are read from other consumers than the DPLL.

#### Scope

DPLL

#### Effects

When the DPLL is enabled and before the DPLL is synchronized (by writing to the relevant pointers (DPLL\_APT\_2c/DPLL\_APS\_1C3) the DPLL\_NUTC.SYN\_T/DPLL\_NUSC.SYN\_S registers are unexpectedly updated by the profile.

Because the SYN\_T\_OLD and SYN\_S\_OLD registers are updated by SYN\_T, SYN\_S they are affected as well.

The DPLL internal processes of calculation of the number of microticks for the next increment is not affected by that bug.

#### Workaround

When DPLL\_NUTC.SYN\_T/\_OLD, DPLL\_NUSC\_SYN\_S/\_OLD values are needed outside the DPLL it must be checked that the DPLL is already synchronized (DPLL\_STATUS.SYT/SYS). When the relevant DPLL channel (TRIGGER/STATE) is not synchronized yet the SYN\_T/S values should be taken into account as "1".

### 2.64 [GTM\_AI.325] TIM: Bits ACB[2:1] lost on interface to ARU (always zero)

#### Description

In case of CFG\_CLOCK\_RATE=1 (see register CCM[i]\_HW\_CONF) some cluster can be configured to run with fast clock frequency (200 MHz) while the ARU always runs with slow clock frequency (100 MHz).

For this configuration of a cluster running with fast clock frequency (200 MHz) also the TIM module in this cluster is running with fast clock frequency (200 MHz).

In this case and if a TIM channel is configured to send data to ARU (ARU\_EN bit in TIM[i]\_CH[x]\_CTRL register), the bits ACB[2:1] will not be transferred with any ARU transfer request, they are always 0.

## **2 Functional deviations**

Due to this any master receiving ARU data is not able to use the ACB bit information received from a fast running TIM.

- ACB[1]: additional ARU request event received while actual request not finished
- ACB[2]: Timeout event occurred

### **Scope**

TIM, ARU transfers

### **Effects**

ARU bits ACB[2:1] sent by TIM are always zero.

### **Workaround 1**

For applications running within a single cluster, use AEI communication (MCS-TIM) instead of ARU communication.

### **Workaround 2**

Use half clock rate for the cluster that contains the TIM module read out via ARU.

### **Workaround 3**

If the data from TIM channel should be routed over the ARU to the MCS module, then the MCS can read the information (TIM[i]\_CH[x]\_IRQ\_NOTIFY[4:3]) directly over the AEI bus master interface instead of routing it through the ARU.

**Note:** For this workaround the TIM channel and the MCS channel have to be in the same cluster.

### **Workaround 4**

If the data from TIM channel should be routed over the ARU to the FIFO or BRC module, then the MCS can read the information (TIM[i]\_CH[x]\_IRQ\_NOTIFY[4:3]) directly over the AEI bus master interface and forward the data via its ARU interface to FIFO or BRC.

**Note:** For this workaround the TIM channel and the MCS channel have to be in the same cluster.

### **Workaround 5**

Depending on the application it might be possible by comparing the actual ARU data with the previous received ARU data to “reconstruct” the ACB bits [2:1].

## **2.65 [GTM\_AI.326] TIM: ARU bit ACB[0] (signal level) incorrect in case a second ARU request occurs while the actual request is just acknowledged**

### **Description**

An issued ARU request will be served at least after the ARU round trip time.

If one GTM clock cycle before the ARU request is acknowledged a new capture event occurs (overflow condition due to for example input change) the bit ACB[0] will not show the new value.

The overflow bit ACB[1] and the ARU data words selected by (E)GPRy\_SEL) will show the correct behavior, only the ACB[0] will show the previous state.

## 2 Functional deviations

### Scope

TIM, ARU transfers

### Effects

ARU bit ACB[0] not consistent with data transferred in ARU data words.

### Workaround 1

Ensure that events which trigger a ARU request occur with a greater timely distance than the ARU round trip time.

### Workaround 2

Use the signal level information embedded in the ARU data words (selectable by ECNT/TIM\_INP\_VAL). This data will show the correct signal level.

## 2.66 [GTM\_AI.329] Interference of MCS to AEI/ADC and CPU to AEI traffic within the same cluster could result in incorrect MCS program execution

### Description

Operations of the MCS via its AEI master port on the AEI bus can be categorized into 3 different types of operations based on the response time required by an addressed resource to complete the operation on the bus. As operations from MCS to ADC are also handled via the MCS AEI master port, ADC operations are also relevant regarding the bus traffic scenarios.

The vast majority of register accesses via AEI as well as ADC reads complete with zero wait states (N=0) on the AEI bus and fall into the first category. The second category is defined by register operations to a small set of special registers that require 1 wait cycle (N=1) on the AEI interface to complete while the third category covers AEI accesses to memories (e.g. DPLL memory, MCS memory or FIFO memory) as well as 2 special registers in MCS that require multiple wait cycles (N>1) on the AEI interface to complete.

Certain interferences between accesses from MCS to the AEI/ADC interface and AEI accesses from CPU within the same cluster can result in bus traffic situations that impact the correct program execution of MCS channels. These rare but critical traffic conditions must be avoided to ensure the correct execution of MCS code.

Further the dynamic usage of GTM\_MCS\_AEM\_DIS to temporarily disable a MCS AEI master port (AEI and ADC communication path) must be avoided. This switch can only be used for the permanent disablement of the MCS AEI master port.

MCS AEI master port usage scenarios proven to avoid the problematic traffic conditions under all circumstances include the usage scenarios described in the workaround section of this erratum. Usage of MCS to AEI or ADC communication not covered by tested scenarios must be avoided.

Communication from MCS to any GTM resource via ARU is not impacted and has no influence on the problems and scenarios described within this erratum.

- GTM resources with 1 AEI wait cycle (N=1):

**Table 5 Memory locations critical from MCS (N=1 wait cycle)**

Register name or memory location
----------------------------------

GTM_RST
---------

GTM_CLS_CLK_CFG
-----------------

BRC_RST
---------

TIM[i]_RST
------------

**(table continues...)**

## 2 Functional deviations

**Table 5 (continued) Memory locations critical from MCS (N=1 wait cycle)**

Register name or memory location
TOM[i]_TGC0_GLB_CTRL
TOM[i]_TGC1_GLB_CTRL
DPLL_CTRL_1
ATOM[i]AGC_GLB_CTRL

- GTM resources with more than 1 AEI wait cycle (N>1):

**Table 6 Memory locations critical from CPU/DMA and MCS (N>1 wait cycles)**

Register name or memory location	Type	Comment
AFD[i]_[0-7]_BUFF_ACC		
FIFO[i]_MEMORY	RAM address space	Not accessible from MCS
DPLL_RAM1A	RAM address space (DPLL_RR1)	
DPLL_RAM1B	RAM address space (DPLL_RR1)	
DPLL_RAM1C	RAM address space (DPLL_RR1)	
DPLL_RAM2	RAM address space (DPLL_RR2)	
MCS[i]_MEM	RAM address space	Not accessible from MCS
MCS[i]_CTRG		
MCS[i]_STRG		

### Technical Background

AEI bus access to all modules within a given cluster is granted by the AEIMux which arbitrates between accesses from the external CPU and accesses from MCS (via AEIM – “AEI master of MCS”). By default AEI access requests from MCS have higher priority than access requests from the external CPU to ensure the determinism of MCS code execution.

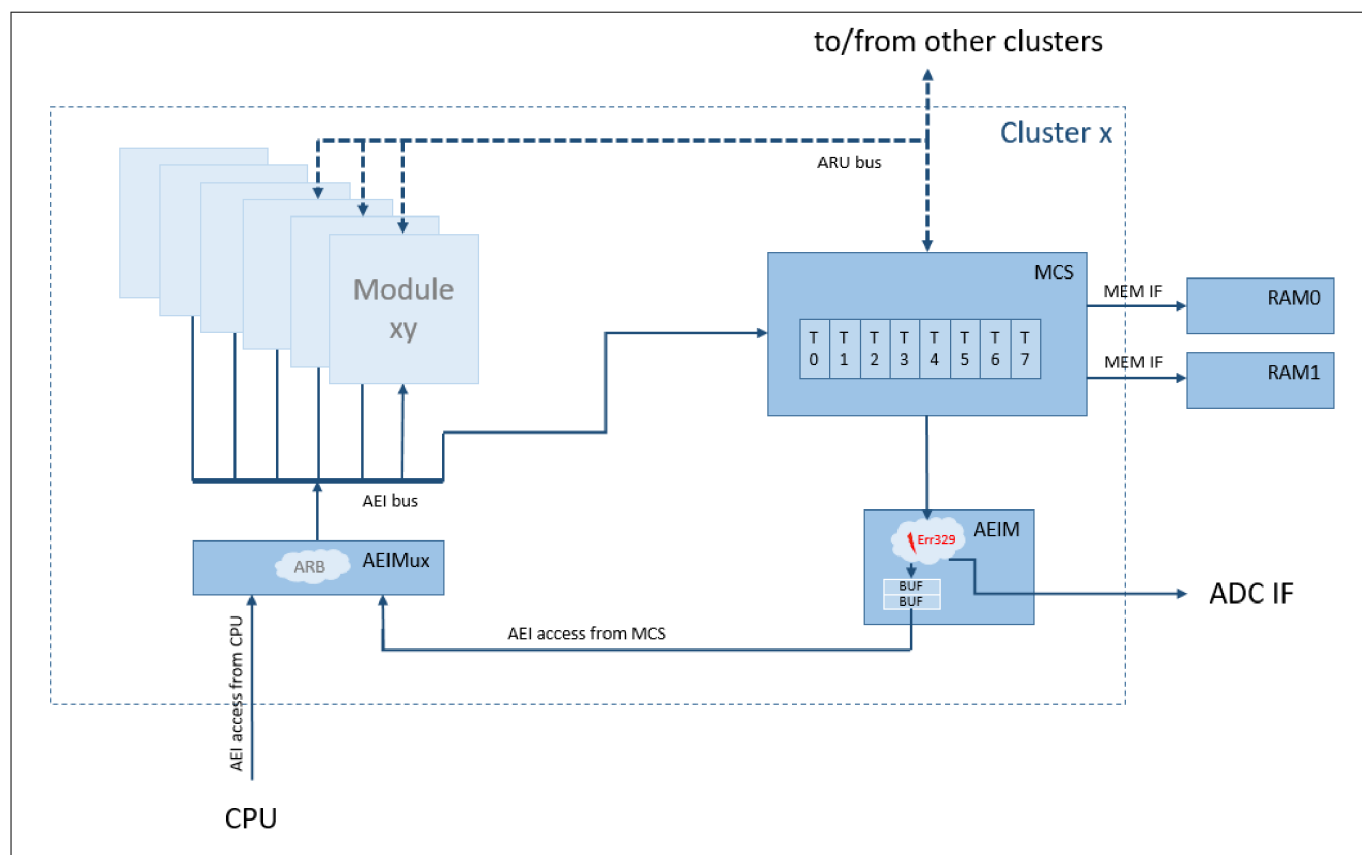
Depending on the addressed target of an AEI bus access, the access will complete either within one bus cycle (N=0 wait cycle), within 2 bus cycles (N=1 wait cycle) or multiple bus cycles (N>1 wait cycles). The vast majority of AEI accessible resources will respond with N=0 wait cycles. For a list of resources with N=1 or N>1 see [Table 5](#) and [Table 6](#) above.

As an AEI bus access of a given MCS thread can be delayed either due to an ongoing AEI bus access from CPU or a multi cycle AEI access from another MCS thread, the AEIM contains buffers to store MCS bus accesses to AEI that cannot be served immediately.

As accesses to ADC from MCS point of view are not different from AEI bus accesses, these ADC accesses are forwarded to the AEIM in the same way and on the same interfaces as MCS accesses to the AEI bus. The AEIM will identify the ADC accesses by their targeted address space and forward them to the GTM external ADC. As ADC accesses will always be served with zero wait time, these ADC accesses are not routed through the AEIM buffers.



## 2 Functional deviations



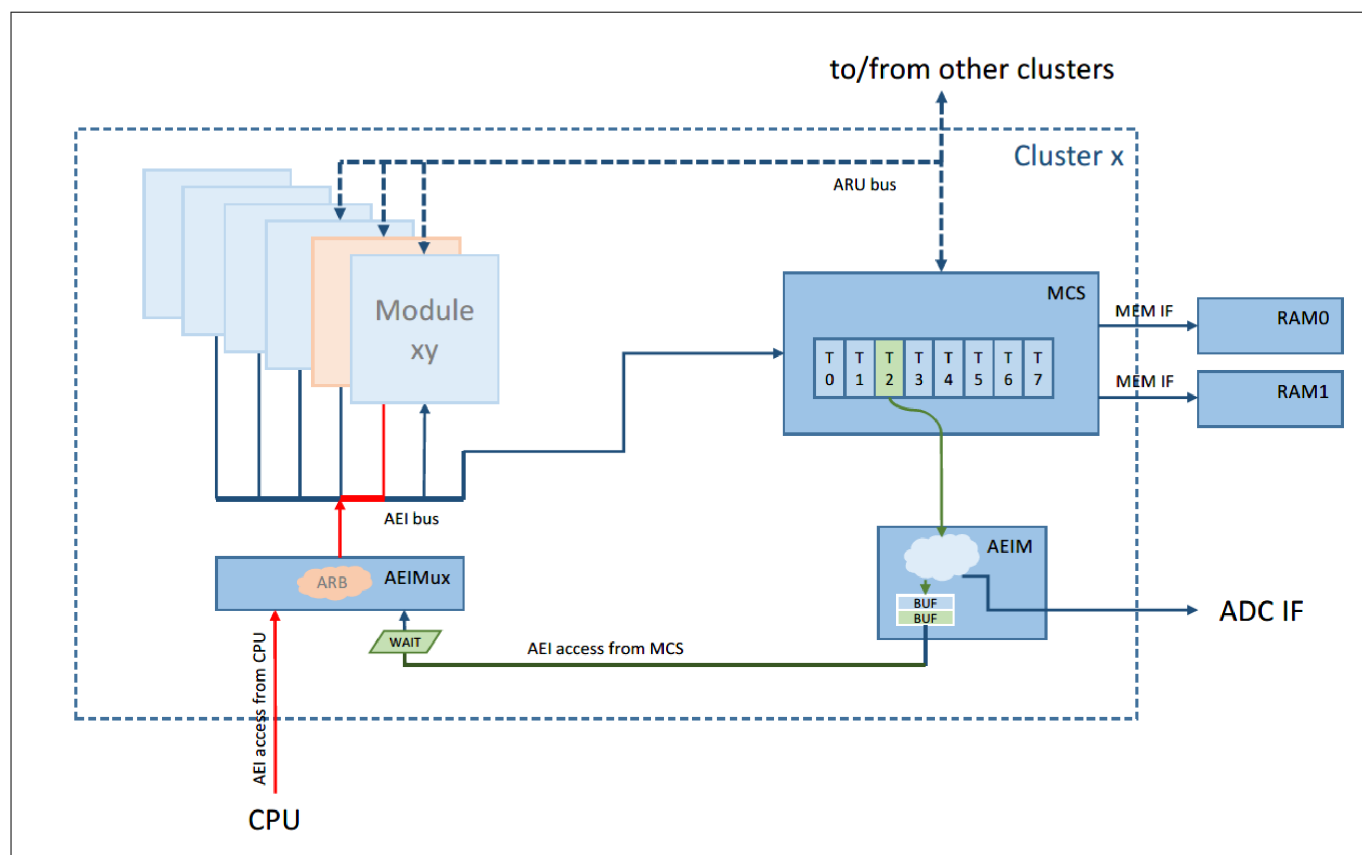
**Figure 4 Cluster-internal AEI network**

Problem GTM\_AI.329 is related to a potential incorrect handling of MCS access requests to the AEI bus at the AEIM entry stage in case that one AEIM buffer is already filled. If in such a case a new access request from MCS enters the AEIM logic during a very specific time window (related to progress on the AEI bus), the AEIM logic might signalize incorrect parameters back to MCS that could result in incorrect data, the loss of data or a repetitive execution of MCS accesses to AEI.

To avoid the potential occurrence of problem GTM\_AI.329 it has to be ensured that not more than 1 AEIM buffer gets filled due to backpressure in the AEI bus system.

Figure 5 shows an uncritical traffic situation. Here an access of a MCS thread to the AEI bus (green access path) is temporary delayed due to an ongoing AEI bus access from the CPU (red access path). The MCS access to the AEI bus will be safely buffered at the AEIM (green buffer). As soon as the CPU access to the AEI bus completes, the buffered AEI bus access from MCS will be executed.

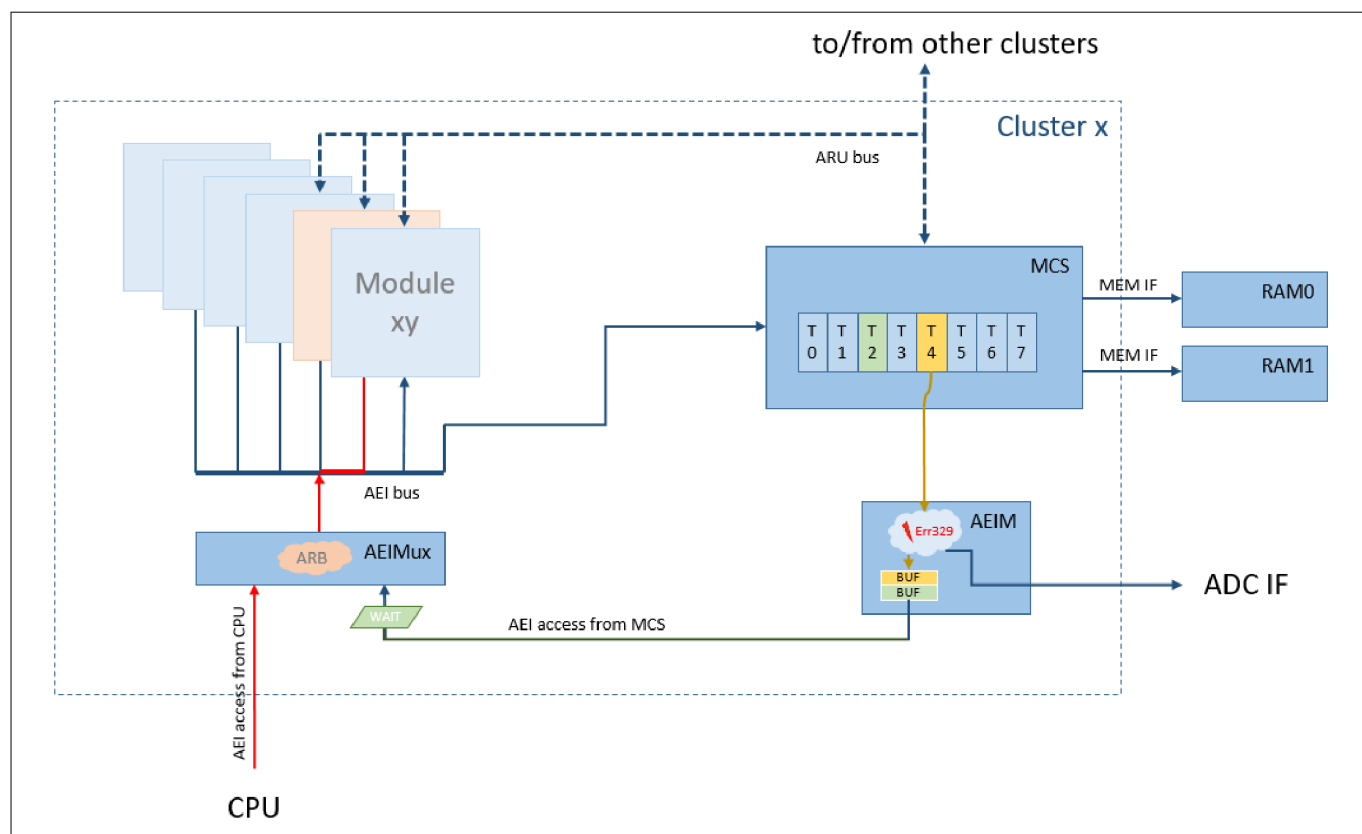
## 2 Functional deviations



**Figure 5 MCS AEI master access path**

In [Figure 6](#), a potential risk for the occurrence of problem GTM\_AI.329 is illustrated. Here the first buffer of the AEIM (green) is still waiting for the access to the AEI bus while a second AEI bus access from MCS arrives at the AEIM (yellow access path). Under certain, but for the user unpredictable timing conditions, this second AEI bus access arriving at the AEIM can trigger the misbehavior described in problem GTM\_AI.329.

## 2 Functional deviations



**Figure 6 Critical MCS AEI master scenario**

The workarounds described in the 'Workaround' section will ensure that not more than one AEIM buffer will be filled and therefore the occurrence of problem GTM\_AI.329 is avoided. All workarounds have been tested and proven correct.

### Scope

Usage of MCS AEI master port (AEI and ADC communication from MCS); MCS channel code execution; Dynamic usage of GTM\_MCS\_AEM\_DIS.

### Effects

Incorrect MCS channel code execution (skipping execution of instructions or repetitive execution of instructions) or processing of incorrect read data from AEI or ADC interface by MCS channel code.

### Workaround

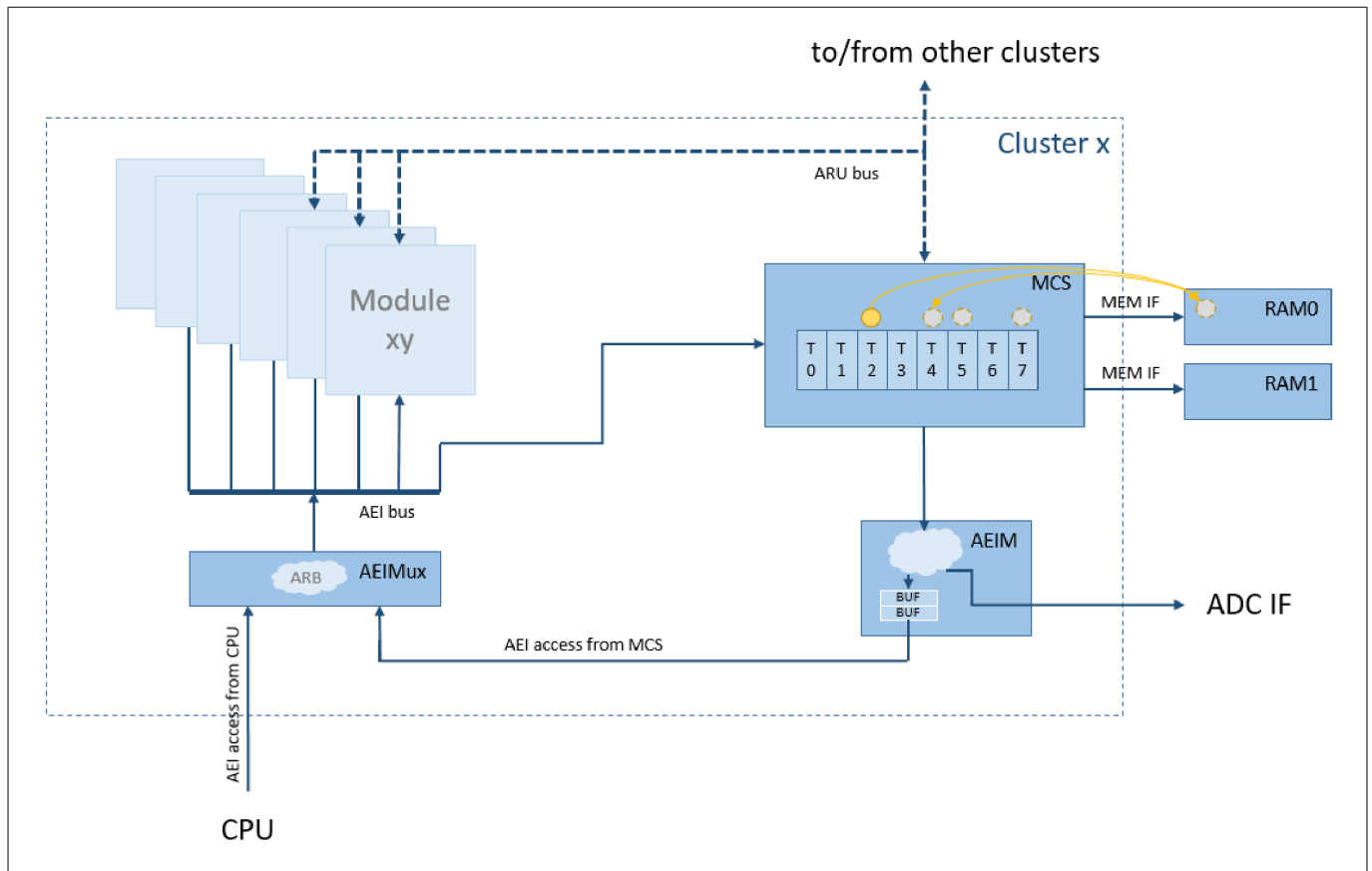
To ensure that a correct execution of MCS channel code is not influenced by certain traffic scenarios on the MCS AEI/ADC bus master interface, only proven usage scenarios are allowed for MCS to AEI/ADC communication. The most common usage scenarios tested to be safe include:

#### Option 1:

Limit the usage of the MCS AEI master port (ADC and AEI communication) to one MCS channel per MCS at a time. ARU communication is available for all MCS channels and there are no limitations for the CPU access path in this usage model.

In case multiple MCS channels want to use the AEI master port for AEI or ADC communication, establish a mechanism that ensures that only one channel uses the AEI master at a time (e.g. exchange a token between channels or use trigger registers to hand over the AEI master port ownership between MCS channels).

## 2 Functional deviations



**Figure 7 MCS token mechanism**

An application note (AN020 – MCS Mutex implementation) describing a token exchange mechanism between MCS threads is available at Bosch. Such a token mechanism allows multiple MCS threads to safely access the AEI bus system by exchanging an AEI bus access token via MCS memory.

If multiple MCS threads have the need to access the AEI bus, only the MCS thread that currently owns the token is allowed to access the AEI bus via AEIM. The token must not be returned before the AEI bus access completed. This will ensure that not more than one AEI bus access is active at the same time.

### Option 2:

Limit the usage of the MCS AEI master port to ADC communication only. The usage of the MCS AEI master port for AEI communication must be avoided for all channels. ARU communication is available for all MCS channels and there are no limitations for the CPU access path in this usage model.

### Option 3:

Limit the usage of the MCS AEI master port to ADC as well as AEI communication with zero wait cycles ( $N=0$ ) only. AEI communication from MCS to resources with  $N>0$  must be avoided.

Further the access from CPU to this cluster has to be limited to accesses with zero or one wait cycle ( $N=0$  and  $N=1$ ) only. Memories or registers with  $N>1$  within the given clusters cannot be accessed by the CPU in this usage model.

If the CPU has to access these resources in this cluster, the number of MCS threads using the MCS AEI master port to access AEI or ADC temporarily has to be limited to one thread while all other MCS threads accessing AEI or ADC have to be suspended while the CPU accesses  $N>1$  resources.

## 2 Functional deviations

**Table 7**                      **Summary of problem GTM\_AI.329**

MCS			Access from CPU cluster		
MCS AEI Master Port	Channels active performing BRD/BWR all at the same time	Wait cycles	No access	Wait cycles N≤1	Wait cycles N>1
AEI or ADC	>1	N=0	Erratum does not apply	Erratum does not apply	Erratum applies; No issue if all channels issuing BRD/BWR instructions except one MCS channel are in suspend state (disabled) while the CPU/DMA access is active
		N>0	Erratum does not apply	Erratum applies	Erratum applies
ADC only	≥1	N=0	Erratum does not apply	Erratum does not apply	Erratum does not apply

### 2.67                      **[GTM\_AI.331] GTM\_TIM[i]\_AUX\_IN\_SRC and GTM\_EXT\_CAP\_EN[i] register: wrong status 2 by AEI write access if cluster 0 is disabled**

#### Description

AEI write access to the register GTM\_TIM[i]\_AUX\_IN\_SRC and GTM\_EXT\_CAP\_EN[i] via the legacy address space responds with status 2 even though the write operation is correctly executed and the register contains the correct new value.

#### Scope

Register access via legacy address.

#### Effects

If status 2 is responded an interrupt bit in GTM\_IRQ\_NOTIFY is set and the write address will be caught in register GTM\_AEI\_STA\_XPT.

Any further AEI access with responded status >0 will not be caught in GTM\_AEI\_STA\_XPT until GTM\_AEI\_STA\_XPT is reset by AEI read to GTM\_AEI\_STA\_XPT.

#### Workaround

Do not use the legacy addresses of the listed registers while cluster 0 is disabled.

## 2 Functional deviations

### 2.68 [GTM\_AI.332] Access to registers GTM\_TIM[i]\_AUX\_IN\_SRC and GTM\_EXT\_CAP\_EN\_[i] via legacy address space: read data always 0 for AEI read access while cluster 0 is disabled

#### Description

AEI read access to registers via legacy address space which are not in any cluster will respond always with read value 0 if cluster 0 is disabled.

- Impacted registers are:
  - GTM\_TIM[i]\_AUX\_IN\_SRC
  - GTM\_EXT\_CAP\_EN\_[i]

#### Scope

Register access via legacy address.

#### Effects

If cluster 0 is disabled, register data unequal to 0 would not be read from any register which is not part of a cluster (see list of impacted register sections) via its legacy address.

#### Workaround

Do not access the impacted registers via their legacy address while cluster 0 is disabled.

### 2.69 [GTM\_AI.333] MCS bus master interface: a not word aligned address access to DPLL ram region can cause incorrect execution of MCS channel code

#### Description

MCS accesses to the DPLL ram regions with not correctly aligned address while concurrently CPU accesses to the same cluster occur could result in incorrect execution of MCS channel code.

#### Scope

MCS bus interface; MCS program execution.

#### Effects

MCS channel program execution incorrect. Instructions might be executed multiple times or might be skipped. MCS BRD\* instruction reads wrong data.

#### Workaround

Ensure that address used in BWR\* and BRD\* instructions is correctly aligned.

**Note:** *If the bus master addresses as provided in table “MCS Master Interface Address Map” are used along with BWR\* and BRD\* then this issue will not occur.*

## **2 Functional deviations**

### **2.70 [GTM\_AI.334] DPLL RAM content of single address can be corrupted after leaving debug mode**

#### **Description**

Assume a MCS RAM write access to DPLL RAM address x in RAM1a, RAM1bc or RAM2 is executed at the point in time when the GTM is switched to debug mode (gtm\_halt\_req=1). Any following write access to DPLL address space while in debug mode will corrupt the data in memory location x when the restore operation which is executed while leaving debug mode (gtm\_halt\_req=0) is processed.

Read operations to DPLL address space while in debug mode will not corrupt the DPLL memory content.

#### **Scope**

GTM Debug

#### **Effects**

Data in RAM might be corrupted

#### **Workaround**

If only READ accesses to DPLL address space are performed while in debug mode the described effect will never occur.

When write accesses to DPLL address space are performed while in debug mode the following workaround has to be considered:

1. Determine with the debugger whether a BWR instruction to DPLL RAM was executed just before the HALT occurred
2. The active address and the data of this instruction has to be written again with a debug access directly before leaving debug mode

### **2.71 [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 Functional deviations**

### **2.72 [GTM\_AI.336] GTM Bus Bridge: Incorrect AEI access execution in case the previous AEI access was aborted with the access timeout abort function**

#### **Description**

In case the GTM internal AEI access timeout abort function is in use (GTM\_CTRL.TO\_VAL != 0 and GTM\_CTRL.TO\_MODE=1), a following AEI access can be corrupted:

- a) A write access might not be executed (register/ memory not written to the specified value)
- b) A read access can return random data (read value does not reflect the content of the addressed register / memory).

Hint: As a timeout based abort of a GTM register access is assumed to be an error scenario, the internal state of the GTM might be exposed. To ensure the proper behavior after such a severe incident, the GTM IP should be re-initialized as part of a recovery action on system level.

#### **Scope**

CPU interface accesses

#### **Effects**

Read access returns random data.

Write access does not change the content of the target address.

#### **Workaround**

Do not use the AEI access abort mode, use the observe mode instead (Set GTM\_CTRL.TO\_MODE=0).

Enable additionally the timeout observe IRQ by setting GTM\_IRQ\_EN.AEI\_TO\_XPT\_IRQ=1 to invoke higher level recovery mechanisms for GTM re-initialization.

(e.g. abort the pending access to the GTM and re-initialize the GTM\_IP from hardware reset).

### **2.73 [GTM\_AI.339] DPLL: Control bits DPLL\_CTRL\_11.PCMF1 and DPLL\_CTRL\_11.PCMF2 are not reset to 0 after a pulse correction is completed**

#### **Description**

In DPLL specification it is written in the description of field PCMF1 in register DPLL\_CTRL\_11: "When taken the MPVAL1 value to RPCUx and INC\_CNT1 the PCMF1 bit is reset immediately and after that also the PCMF1 bit."

The implemented behavior of the DPLL is that the PCMF1 bit is not reset after the PCMF1 bit is reset to 0. In mode DPLL\_CTRL\_1.SMC=1, the same is true for the signal DPLL\_CTRL\_11.PCMF2.

#### **Scope**

DPLL

#### **Effects**

After a pulse correction is executed by writing to DPLL\_CTRL\_1.PCM1=1 and this signal is reset to 0 again, the signal DPLL\_CTRL\_11.PCMF1 is not reset back to 0.

After a pulse correction is executed by writing to DPLL\_CTRL\_1.PCM2=1 and this signal is reset to 0 again, the signal DPLL\_CTRL\_11.PCMF2 is not reset back to 0.



## **2 Functional deviations**

### **Workaround**

Before a following pulse correction is executed this signal must be set to 0 again if needed. When a sequence of pulse corrections with the same configuration of DPLL\_CTRL\_11.PCMF1 or DPLL\_CTRL\_11.PCMF2 is executed no modification of DPLL\_CTRL\_11.PCMF1 or DPLL\_CTRL\_11.PCMF2 is necessary.

When reset of DPLL\_CTRL\_11.PCMF1 or DPLL\_CTRL\_11.PCMF2 is needed this can be done by writing to register DPLL\_CTRL\_11.PCMF1/2.

### **2.74 [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. 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.

#### **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.

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.

## 2 Functional deviations

### Workaround 1

Disabling, resetting (channel reset), re-enabling and initializing of the channel between each one-shot pulse will ensure 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.

Be aware 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.

## 2.75 [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.

### 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\_CCU1 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\_CCU1 trigger signal which is not expected at this point in time

### Scope

TOM/ATOM

## 2 Functional deviations

### Effects

Missing TRIG\_CCUI trigger signal in initial phase of the one-shot cycle and unexpected TRIG\_CCUI 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  $2^{**4}-16$ .

**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.

## 2.76 [GTM\_AI.344] DPLL: Incorrect AEI\_STATUS on internal MCS2DPLL interface on valid and implemented address accesses

### Description

The status signal on the MCS2DPLL interface is always responding with "0b11" independent if an available or an unavailable address with correct byte alignment of that interface is accessed.

### Scope

DPLL, MCS0

### Effects

When the master interface of the MCS is accessing any address of the MCS2DPLL interface the DPLL always responds by setting the internal signal `mcs_aeim_status` = "0b11". When this happens the register `CCM0_AEIM_STA` is storing the `mcs_aeim_status` of "0b11" and additionally storing the address of the access. Although the MCS2DPLL interface is operating correctly it is not possible to check for invalid accesses under the described conditions.

If the register `MCS[0]_CTRL_STAT.HLT_AEIM_ERR=0b1` the MCS0 channel which executed the bus master access is halted.

### Workaround

The register bit-field `MCS0_CTRL_STAT.HLT_AEIM_ERR` must be set to "0b0" to prevent the MCS0 channels from halt.

For the `mcs_aeim_status` there is no workaround possible. The master AEI interface of the MCS is operating correctly under the above configuration, but it is not possible to check for invalid address accesses via the `CCM0_AEIM_STA` register when the MCS is accessing any address of the MCS2DPLL interface.

## **2 Functional deviations**

### **2.77 [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.

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

### **2.78 [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.

## 2 Functional deviations

### 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.

### 2.79 [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\_CC0=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.

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.

### 2.80 [GTM\_AI.348] DPLL: Correction of missing pulses delayed after start of pulse generation

#### Description

The described erratum occurs in the DPLL configuration DPLL\_CTRL\_1.DMO=0 (Automatic end mode) and DPLL\_CTRL\_1.COA=0 (Fast pulse correction). When after the start of pulse generation (DPLL\_CTRL\_1.SGE1/2=0-->1) not all pulses scheduled could be generated, repeating the pulses at fast speed is not executed at the second TRIGGER/STATE input event.

#### Scope

DPLL

## 2 Functional deviations

### Effects

When the pulse generation has been started by setting DPLL\_CTRL\_1.SGE1/2 and not all scheduled pulses could be generated there is no fast pulse correction after the second active input signal. Beyond that the DPLL internal pulse counter DPLL\_ICNT1/2 is incremented correctly so that no pulse is getting lost. After the third input event the pulse correction is working as specified.

### Workaround 1

DPLL must be in direct load mode (DPLL\_CTRL\_1.DLM1/2 = 1). Set DPLL\_ADD\_IN\_LD1/2.ADD\_IN\_LD1/2=0 for the first two increments after the DPLL pulse generation has been started by DPLL\_CTRL\_1.SGE1/2=1 (all GTM Versions)

### Workaround 2

Do nothing: If there is no need to do the pulse correction for the second input signal after start of pulse generation. With the third input signal the pulse correction is starting to work.

### Workaround 3

**Note:** Workaround 3 is applicable for all GTM versions used in TC3xx devices. It is not applicable for TC2xx devices.

Use pulse correction mechanism triggered by DPLL\_CTRL\_1.PCM1/2:

- Set DPLL\_MPVAL1/2.MPVAL1/2 to the desired number of pulses which has to be sent out fast
- Set DPLL\_CTRL\_11.PCMF1/2=1 AND DPLL\_CTRL\_11.PCMF1/2\_INCCNT\_B=1
- Trigger the fast pulses by setting DPLL\_CTRL\_1.PCM1/2=1

## 2.81 [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**

### **2.82 [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.

### **2.83 [GTM\_AI.351] MAP: Disable of input lines by MAP\_CTRL register not implemented for input signals TSPP0 TIM0\_CHx(48) (x=0..2) and TSPP1 TIM0\_CHx(48) (x=3..5)**

#### **Description**

The Control bits TSPP0\_I0V, TSPP0\_I1V, TSPP0\_I2V, TSPP1\_I0V, TSPP1\_I1V, TSPP1\_I2V of register MAP\_CTRL are not operating as specified. The specified gating functions of the input signals TIM0\_CH0(48), TIM0\_CH1(48), TIM0\_CH2(48) of TSPP0 submodule and the input signals TIM0\_CH3(48), TIM0\_CH4(48), TIM0\_CH5(48) of TSPP1 submodule are not implemented, hence the input signals cannot be disabled.

#### **Scope**

MAP

#### **Effects**

The specified disable function of the input signals TIM0\_CH0(48), TIM0\_CH1(48), TIM0\_CH2(48) of TSPP0 submodule and the input signals TIM0\_CH3(48), TIM0\_CH4(48), TIM0\_CH5(48) of TSPP1 submodule are not implemented, hence the input signals cannot be disabled.

#### **Workaround**

The combined TRIGGER or STATE output signals to the DPLL module can be disabled by using the control signals DPLL\_CTRL\_0.TEN(TRIGGER, TSPP0) and DPLL\_CTRL\_0.SEN (STATE, TSPP1).



## 2 Functional deviations

No workaround exists for switching off the level input signals of the TSPP0 and TSPP1 submodules individually.

### 2.84 [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 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.



## **2 Functional deviations**

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.

### **2.85 [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 (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.

## 2 Functional deviations

### 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.

### 2.86 [GTM\_AI.354] MCS: Unresolved hazard resulting from RAW (Read After Write) dependency

#### Description

If an MCS instruction sequence has any RAW (read after write) data dependency, which involves one of the following SFRs mentioned below, the read access is executed before the write access if the latency of these instructions is one or two clock cycles.

The involved SFRs are: GMI0, GMI1, DSTA, DSTAX or AXIMI.

#### Example

Assume that following sequence

- MOV GMI0, R0 // write GMI0
- MOV R1, GMI0 // read GMI0

is executed in two subsequent clock cycles (w/o any additional wait cycles), read access of GMI0 is executed before the write access to GMI0.

#### Scope

MCS

#### Effects

The executed order of the program sequence is not as specified in the program code.

#### Workaround

Ensure that the delay between such RAW dependencies is always greater than 2 clock cycles.

For example:

1. Chose round robin scheduling mode, in which the situation will never occur
2. Reformulate the sequence in a way that there are at least two instructions between the critical RAW dependency. For example:
  - MOV GMI0, R0
  - NOP
  - NOP
  - MOV R1, GMI0

### 2.87 [GTM\_AI.357] MCS: instructions XCHB, SETB, and CLRB do not suppress register write

#### Description

According to the specification, the instructions XCHB CLRB, and SETB perform a specific bit operation on the B[4:0]-th bit of register A, but only if B[4:0] is less than 24. If B[4:0] is greater than or equal to 24, the content of A shall not be modified.

## **2 Functional deviations**

However, the current RTL implementation of these instructions always reads register A and it is always followed by a write back to register A, independently of the value B[4:0]. But the read content of A is only modified if B[4:0] is less than 24.

Thus, the functional behavior of this implementation is correct in the case that A is a register which only has non-volatile register bits. However, if A is a register that has volatile bits, the result might also be modified if B[4:0] is greater than or equal to 24, since the write access to this register might modify its content.

### **Scope**

MCS

### **Effects**

If B[4:0] is greater than or equal to 24, unexpected write accesses to the referred SFR of A can occur.

### **Workaround**

MCS program must ensure that B[4:0] is always in the range of 0 to 23, at least if volatile SFRs are used as argument A in the instructions XCHB, SETB, or CLRB.

## **2.88 [GTM\_AI.358] TOM/ATOM: Synchronous update of working register for RST\_CCU0=1 and UDMODE=0b01 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=0b01. 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=0b01.

### **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 unintendedly with the selected trigger signal TRIGIN[x] or TIM\_EXT\_CAPTURE.

### **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.

## 2 Functional deviations

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.

### 2.89 [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  $CN0 \geq CM1$  and the second output signal TOM\_OUT\_T has to be set to SL when  $CN0 \geq CM0$ .

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  $CN0 \geq CM0 - 1$ , which is one CMU clock cycle too 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 too 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 ( $CM0+1$ ).

### 2.90 [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.

## **2 Functional deviations**

### **Workaround**

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

### **2.91 [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.

### **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.

### **2.92 [GTM\_AI.362] MCS: Using wrong WURM mask during execution of instruction WURMX or WURCX**

#### **Description**

If a WURM mask defined in R6 for usage with the instruction WURMX or WURCX is updated exactly one clock cycle before the associated instruction is executed, the WURMX or WURCX instruction is using the old (not yet updated) value of R6 as WURM mask for exactly one cluster clock cycle.

Example:

Assume that the sequence

MOVL R6, 0x2

...

## 2 Functional deviations

MOVL R6, 0x1

WURMX R0, STRG

is executed with Accelerated Scheduling Mode and the scheduler does not apply any delay between both instructions, the WURMX instruction is using the old value 0x2 as WURM mask for the very first cluster clock cycle. In subsequent cluster clock cycles the correct value 0x2 is used as WURM mask.

### Scope

MCS

### Effects

WURMX and WURCX instructions are sensitive to the wrong volatile bit to be observed (for example interrupt or trigger bit) for one cluster clock cycle.

### Workaround

Ensure that delay between update of the WURM mask R6 and its associated WURM instruction is greater than one cluster clock cycle. For example:

1. Insert NOP instruction or another useful instruction between update of R6 and the associated WURMX or WURCX instruction
2. Use Round Robin Scheduling Mode

## 2.93 [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:

## 2 Functional deviations

- 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.

## 2.94 [GTM\_AI.367] MCS: Instructions WURMX and WURCX implement invalid extended register set for argument A

### Description

Current implementation uses register set XOREG for the argument A of instructions WURMX and WURCX. However, the specification only allows the usage of the register set OREG, which is a subset of XOREG. In detail, the current implementation is evaluating a don't care bit of its instruction code (bit position 14) for determination of the register address A.

### Scope

MCS

### Effects

If the SW tool chain is expanding a '1' for the don't care bit at position 14, the WURMX and WURCX instruction will use a wrong register A.

### Workaround

The SW tool chain must ensure that the unused don't care bits of these instructions are always expanded as '0'.

## 2.95 [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  $RST\_CCU0 = 0$ :

## **2 Functional deviations**

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

## **2.96 [GTM\_AI.371] MCS: Instruction MWRIL applies unexpected address offset calculation**

### **Description**

The MCS instruction MWRIL applies an address offset calculation by evaluation of bits 2 to 15 of the corresponding instruction code, although these bits are defined as don't care bits.

### **Scope**

MCS

### **Effects**

If the don't care bits 2 to 15 of the instruction code are set unequal to zero, a wrong address is calculated for the memory access.

### **Workaround**

Ensure that the don't care bits 2 to 15 of the instruction word are set to zero.



## 2 Functional deviations

### 2.97 [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 section 15.3.3 “ATOM Signal output mode PWM (SOMP)” of the GTM chapter in the AURIX™ TC3xx User’s Manual, that “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” (section 15.3.3.4). Further, it is mentioned: “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.

### 2.98 [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

## **2 Functional deviations**

### **Effects**

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

### **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.

### **2.99 [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.

### **2.100 [GTM\_AI.387] DPLL: Wrong calculation of pulse generator frequency for DPLL\_CTRL\_0.AMT/S=1 and DPLL\_CTRL\_11.ADT/S=1 when number of pulses (DPLL\_CTRL\_0.MLT or DPLL\_MLS1/2.MLS1/2) is too small**

#### **Description**

When the number of pulses per increment DPLL\_CTRL\_0.MLT is smaller than 127, or DPLL\_MLS1/2.MLS1/2 is smaller than 128 and the correction of physical deviations is used (DPLL\_CTRL\_0.AMT/AMS = 1 and DPLL\_CTRL\_11.ADT/ADS = 1), the calculation of internal values such as DPLL\_DT\_T/S\_ACT.DT\_T/S\_ACT, DPLL\_RDT\_T/S\_ACT.RDT\_T/S\_ACT, and DPLL\_ADD\_IN\_CAL1/2.ADD\_IN\_CAL\_1/2 are wrong. The resulting frequency of the generated sub-increment pulses of the DPLL is too small.

## **2 Functional deviations**

### **Scope**

DPLL

### **Effects**

The frequency of the generated sub increment pulses of the DPLL is too small. This leads to an unbalanced generation of micro ticks.

### **Workaround**

1. Do not use pulse numbers  $DPLL\_CTRL\_0.MLT < 127$  and/or  $DPLL\_MLS1/2.MLS1/2 < 128$ , when using correction of physical deviation ( $DPLL\_CTRL\_11.ADT/ADS = 1$  when  $DPLL\_CTRL\_0.AMT/AMS = 1$ )
2. When workaround 1 cannot be applied use configuration  $DPLL\_CTRL\_11.ADT/ADS = 0$  when  $DPLL\_CTRL\_0.AMT/AMS = 1$  is used

## **2.101 [GTM\_AI.398] DPLL: Incorrect DPLL\_THVAL calculation leading to a false direction decision in case tbu\_ts0 wraps around**

### **Description**

When

- a) The inactive edge of TRIGGER input signal is used for detection of the direction ( $DPLL\_CTRL\_1.IDDS = 1$ ) and
- b) The input delay information is used to correct time stamps ( $DPLL\_CTRL\_0.IDT = 1$ ) and
- c) In between the active input signal edge and the inactive input signal edge on TRIGGER `tbu_ts0` wraps around then the calculation of  $DPLL\_THVAL.THVAL$  is incorrect incurring a false direction decision.

### **Scope**

DPLL

### **Effects**

Wrong value of  $DPLL\_THVAL$  and false direction decision.

### **Workaround**

Do not use  $DPLL\_CTRL\_0.IDT = 1$  when evaluating direction with  $DPLL\_CTRL\_1.IDDS = 1$ .

## **2.102 [GTM\_AI.400] MCS-RTL: Division instruction may produce unexpected memory overflow and wrong results**

### **Description**

Assume that a division instruction (DIVU or DIVS) is located in the MCS memory within the address range `[MP1-4*6, ..., MP1-4]`. If this instruction is executed with an Accelerated Scheduling mode or a Prioritized Scheduling mode the associated MCS channel potentially stops its execution and signalizes a memory overflow. In this case the calculated results of the instruction are wrong.

### **Scope**

MCS

## **2 Functional deviations**

### **Effects**

An MCS channel stops with a memory overflow error and the calculated results of the division instruction are wrong.

### **Workaround**

Re-order program sequence in a way that any division instruction is located outside the critical address range [MP1-4\*6, ..., MP1-4] of the MCS memory.

## **2.103 [GTM\_AI.404] MCS-RTL: Division instruction reports unrelated ECC error**

### **Description**

If a sequential division instruction is executed at memory location  $x$ , and a read access to a subsequent memory location  $x+y$  with  $1 \leq y \leq 6$  has an ECC error, the former instruction incorrectly reports an ECC error.

### **Scope**

MCS

### **Effects**

Unexpected signaling of an ECC error.

### **Workaround**

None

## **2.104 [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)

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.

## 2 Functional deviations

### 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.

## 2.105 [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 (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

## **2 Functional deviations**

### **Effects**

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

### **Workaround**

None.

## **2.106 [GTM\_AI.409] DPLL: Flags of register DPLL\_STA\_FLAG are not set**

### **Description**

The flags of register DPLL\_STA\_FLAG (STA\_FLAG\_T, STA\_FLAG\_S, INC\_CNT1\_FLAG, INC\_CNT2\_FLAG) are not set when one of these flags is cleared by a write operation of MCS register DSTAX (STA\_FLAG\_T, STA\_FLAG\_S, INC\_CNT1\_FLAG, INC\_CNT2\_FLAG) within the same cluster0 clock cycle.

### **Scope**

DPLL, MCS

### **Effects**

The event represented by the flags will be missed and not set in the registers. This affects the DPLL\_STA\_FLAG register as well as the MCS register DSTAX with the bit-fields: STA\_FLAG\_T, STA\_FLAG\_S, INC\_CNT1\_FLAG, INC\_CNT2\_FLAG.

### **Workaround**

If DPLL\_CTRL\_0.RMO = 0, only use DPLL\_STA\_FLAG.STA\_FLAG\_T or DPLL\_STA\_FLAG.INC\_CNT1\_FLAG.

If DPLL\_CTRL\_0.RMO = 1 and DPLL\_CTRL\_SMC = 0, only use DPLL\_STA\_FLAG.STA\_FLAG\_S or DPLL\_STA\_FLAG.INC\_CNT1\_FLAG.

If DPLL\_CTRL\_0.RMO = 1 and DPLL\_CTRL\_SMC = 1, only use one of DPLL\_STA\_FLAG.STA\_FLAG\_S, DPLL\_STA\_FLAG.INC\_CNT2\_FLAG, DPLL\_STA\_FLAG.STA\_FLAG\_T, or DPLL\_STA\_FLAG.INC\_CNT1\_FLAG.

## **2.107 [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 after the soft reset.

## **2 Functional deviations**

### **2.108 [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.

#### **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.*

### **2.109 [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

## 2 Functional deviations

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<sub>B</sub>, 001<sub>B</sub>, 100<sub>B</sub>}) are not affected.

### 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 Functional deviations

### 2.110 [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.

### 2.111 [GTM\_AI.422] DPLL: Wrong DPLL\_RDT\_S\_ACT/DPLL\_RDT\_T\_ACT value in case of overflow correction

#### Description

The wrong overflow correction occurs for DPLL\_RDT\_S\_ACT when the DPLL is in normal mode (DPLL\_CTRL\_0.RMO=0, DPLL\_CTRL\_1.SMC=0), or for DPLL\_RDT\_T\_ACT when the DPLL is in emergency mode (DPLL\_CTRL\_0.RMO=1, DPLL\_CTRL\_1.SMC=0).

Instead of 0xFFFFF the value 0x000000 is written in both cases. A problem in the calculation of pulse frequency (settling behavior) or PMT values may occur when the mode in DPLL\_CTRL\_0.RMO is switched (normal mode <-> emergency mode). If the overflow value was not yet overwritten (due to engine revolution happening before mode's switch) the wrong value might come into use for the described calculations.

#### Scope

DPLL

#### Effects

Wrong value in either DPLL\_RDT\_T\_ACT (emergency mode) or DPLL\_RDT\_S\_ACT (normal mode) after detection of overflow condition. With the next active input signal slope a potentially wrong value of DPLL\_RDT\_T\_ACT is stored to DPLL\_RDT\_T of RAM2 and DPLL\_RDT\_S\_ACT is stored to DPLL\_RDT\_S of RAM1bc. This might lead to different settling behavior of the sub increments and wrong results for PMT calculations if these values are actually used.

#### Workaround

Modification of DPLL\_RDT\_T\_ACT (emergency mode) or DPLL\_RDT\_S\_ACT (normal mode) after detection of overflow condition is not possible but does not cause any negative effects on pulse generation or PMT calculation at all.

The values stored to DPLL\_RDT\_T of RAM2 or DPLL\_RDT\_S of RAM1bc need to be corrected by following workaround sequence:

1. Check if relevant overflow on either DPLL\_RDT\_T\_ACT or DPLL\_RDT\_S\_ACT occurred. This can be done by observation of DPLL\_STATUS.CRO when interrupt DPLL\_IRQ\_NOTIFY.EI occurred.

## **2 Functional deviations**

2. Check which of the interrupts DPLL\_IRQ\_NOTIFY.TASI/SASI has occurred next and based on that DPLL\_RDT\_T or DPLL\_RDT\_S has to be corrected.
3. For DPLL\_CTRL\_0.RMO = 0 and DPLL\_CTRL\_1.SMC = 0, DPLL\_RDT\_S[DPLL\_APS.APS -1] has to be written with 0xFFFFFFFF.  
For DPLL\_CTRL\_0.RMO = 1 and DPLL\_CTRL\_1.SMC = 0, DPLL\_RDT\_T[DPLL\_APT.APT -1] has to be written with 0xFFFFFFFF.

### **2.112 [GTM\_AI.428] DPLL: Pulse correction is executed twice**

#### **Description**

If DPLL\_CTRL\_1.PCM1/2 is set during DPLL\_CTRL\_1.DEN = 0 or DPLL\_CTRL\_1.DEN changes from 1 to 0, these values are immediately transferred to the respective shadow registers DPLL\_CTRL\_1\_TRIGGER\_SHADOW and DPLL\_CTRL\_1\_STATE\_SHADOW. Since DPLL\_CTRL\_1.DEN = 0, the DPLL\_CTRL\_1.PCM1/2 bits are not cleared when transferred to the shadow registers.

When DPLL\_CTRL\_1.DEN = 1 and the next relevant input signal on either STATE or TRIGGER arrives, the pulse corrections are executed due to the state of the named shadow registers.

As DPLL\_CTRL\_1.PCM1/2 were not cleared this is leading to an additional pulse correction for the next following input signal (TRIGGER / STATE).

#### **Scope**

DPLL

#### **Effects**

One additional pulse correction with DPLL\_MPVAL1/2 pulses with impact on the angle base CCM[0]\_TBU\_TS1 or CCM[0]\_TBU\_TS2.

#### **Workaround**

Avoid setting DPLL\_CTRL\_1.PCM1/2 when DPLL\_CTRL\_1.DEN = 0.

### **2.113 [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

## **2 Functional deviations**

### **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.

## **2.114 [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]\_FLT\_RE.FTL\_RE = 0 and/or TIM[i]\_CH[x]\_FLT\_FE.FTL\_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 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 Functional deviations**

### **2.115 [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.

### **2.116 [GTM\_AI.441] DPLL: Missing pulse correction in case of DPLL\_CTRL\_1.SMC=1**

#### **Description**

If `DPLL_CTRL_1.SMC=1` and `DPLL_CTRL_0.RMO=1` and `DPLL_CTRL_11.PCMF1=0` no pulse correction on `CCM[0]_TBU_TS1` is executed.

#### **Scope**

DPLL

#### **Effects**

Missing pulse correction via sub increment generator 1 for `CCM[0]_TBU_TS1` under the described configuration.

#### **Workaround 1**

Use `DPLL_CTRL_0.RMO=0` when `DPLL_CTRL_1.SMC=1`. In this case fast pulse corrections via `DPLL_CTRL_11.PCMF1` are possible.

## **2 Functional deviations**

### **Workaround 2**

Use DPLL\_CTRL\_11.PCMF1=1 for pulse corrections. In this case no negative values for DPLL\_MPVAL1 can be used.

### **2.117 [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=00<sub>B</sub>, 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.

### **2.118 [GTM\_AI.450] DPLL: Stored time stamp values do not consider filter delays**

#### **Description**

For the case where the filter delay values should be considered (DPLL\_CTRL\_0.IDT/IDS=1) the data values of the time stamp fields in RAM1c (DPLL\_TSF\_S) and RAM2 (DPLL\_TSF\_T) actually do not take them into account for the input signals TRIGGER/STATE.

#### **Scope**

DPLL

#### **Effects**

The missing correction of the stored time stamp values does lead to inaccuracies in DPLL PMT calculations.

#### **Workaround**

The entry of DPLL\_TSF\_T[p]/\_S[p] can be read, corrected (by DPLL\_FTV\_T/\_S), and written back.

The correction needs to be done after the DPLL has received new input data. For this reason it is necessary to read and store the filter value of the last but one DPLL input signal, which then will be used for the correction.

## 2 Functional deviations

### 2.119 [GTM\_AI.451] DPLL: Wrong measured position stamps in RAM

#### Description

For the synchronous motor control (DPLL\_CTRL\_1.SMC=1) in normal mode (DPLL\_CTRL\_0.RMO=0) wrong values are stored in RAM1b for DPLL\_PSSM and DPLL\_PSSM\_OLD. The entries are not derived from CCM[0]\_TBU\_TS2 at the point of time when the active input signal arrives but they are derived erroneously from CCM[0]\_TBU\_TS1 instead.

#### Scope

DPLL

#### Effects

Wrong values for DPLL\_PSSM and DPLL\_PSSM\_OLD stored in memory. Controlling of angle clock cannot rely on these values.

#### Workaround

Configure relevant TIM channels which are used to define the STATE input signal, such that CCM[0]\_TBU\_TS2 is captured in each one of the TIM[0]\_CH[x]\_GPR1.GPR1 registers.

After a STATE input signal has arrived, wait until the point in time when the DPLL should have calculated the DPLL\_PSSM or DPLL\_PSSM\_OLD value. This is fulfilled when the content of the bit-field DPLL\_STA.STA\_S has passed the value 28<sub>H</sub>.

Then write DPLL\_PSSM or DPLL\_PSSM\_OLD with the value of the TIM[0]\_CH[x]\_GPR1.GPR1 register of the corresponding TIM channel causing the captured input signal edge of STATE input. Which of the DPLL\_PSSM or DPLL\_PSSM\_OLD values has to be written might be figured out by using the content of the bit-field DPLL\_OSW.SWON\_S.

### 2.120 [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 Functional deviations

### 2.121 [GTM\_AI.456] DPLL: No action calculation

#### Description

If DPLL\_CTRL\_1.SMC=1 and DPLL\_CTRL\_0.RMO=0 no action calculation is done in STATE processing unit for action channels NOAC/2 to NOAC-1 (NOAC: number of action channels).

**Note:** Starting with GTM V4.1.\* NOAC=32, while in previous versions NOAC=32 for TC3xx and TC29x, and NOAC=24 for TC27x and TC26x.

#### Scope

DPLL

#### Effects

No action calculation of channels NOAC/2 to NOAC-1.

#### Workaround

None

### 2.122 [GTM\_AI.458] DPLL: Missing TOR interrupt and status flag

#### Description

**Note:** Register names in the text follow the TC3xx syntax conventions. Correlation of register names:

- **TC2xx:** DPLL\_, DPLL\_TS\_T\_0, DPLL\_TS\_T\_1, DPLL\_TS\_T\_OLD\_0, DPLL\_TS\_T\_OLD\_1
- **TC3xx:** DPLL\_, DPLL\_TS\_T, DPLL\_TS\_T\_OLD

If DPLL\_CTRL\_0.RMO=1 and DPLL\_CTRL\_1.SMC=0, the TOR interrupt (DPLL\_IRQ\_NOTIFY.TORI) is not triggered and the status flag (DPLL\_STATUS.TOR) is not set on encountering an out of range TRIGGER.

#### Scope

DPLL

#### Effects

The TOR interrupt (DPLL\_IRQ\_NOTIFY.TORI) is not triggered and the status flag (DPLL\_STATUS.TOR) is not set with the described configuration.

#### Workaround

No workaround available in hardware.

Nevertheless the application can detect the trigger out of range interrupt by observing TBU\_TS0:

If the current TRIGGER time stamp (DPLL\_TS\_T.TRIGGER\_TS/DPLL\_TS\_T\_OLD.TRIGGER\_TS\_OLD) + DPLL\_DT\_T\_ACT.DT\_T\_ACT\* DPLL\_TLR.TLR > TBU\_TS0 and no active TRIGGER input was encountered, the CPU/MCS can force a TOR interrupt by writing a one to DPLL\_IRQ\_FORCINT.TRG\_TORI.

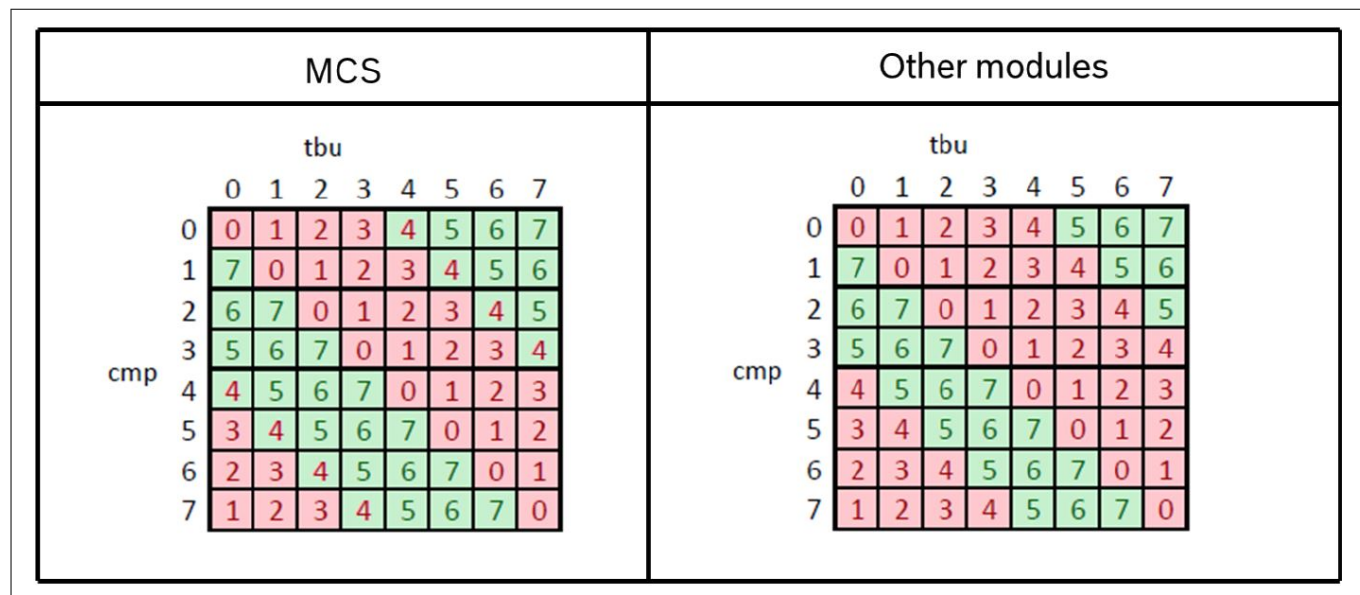
### 2.123 [GTM\_AI.461] MCS: Unexpected behavior of instruction WUCE

#### Description

The GTM specification for the Cyclic Event Compare was not detailed enough to describe the complete range including the boundaries between future and past.

## 2 Functional deviations

With the revised specification of GTM 4.1 it was revealed that the implementation for the WUCE instruction is not correct for the boundaries. The implementation of the WUCE instruction has an inverted behavior for both boundaries between future and past.



**Figure 8** TBU value evaluation MCS against other modules. Red (past) and green (future).

**Note:** Typical applications do not operate directly with these boundary values.

### Scope

MCS

### Effects

The behavior of the WUCE instruction for both boundary values does not follow the formal definition of Cyclic Event Compare functionality. A point in time in the past might be considered as future on the MCS and vice versa.

### Workaround

No workaround.

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

### Description

Configuration:

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.



## **2 Functional deviations**

### **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.

## **2.125 [GTM\_AI.463] DPLL: DPLL\_PVT not cleared after direction change**

### **Description**

For settings of DPLL\_CTRL\_1.SMC=1 or alternatively DPLL\_CTRL\_1.SMC=0 and DPLL\_CTRL\_1.IDDS=1 the direction change on TRIGGER channel is done via DPLL input port "TDIR" (generated via the control path SPE or TIM to MAP to DPLL).

If there is a direction change the RAM parameter DPLL\_PVT is not cleared as specified.

### **Scope**

DPLL

### **Effects**

DPLL\_PVT is not cleared and the PVT check is not suppressed under the described conditions. The PVT violation interrupt (DPLL\_PWI\_IRQ) could be unexpectedly triggered.

### **Workaround**

Reset DPLL\_PVT by CPU or MCS0 write operation, when direction change is detected via DPLL\_IRQ\_NOTIFY.DCGI interrupt.

## **2.126 [GTM\_AI.464] DPLL: Pulse correction executed twice when DPLL\_CTRL\_11.INCF1/2 is activated**

### **Description**

DPLL\_MPVAL1/2.MPVAL1/2 is incremented or decremented twice in subsequent input signal events (only in the current increment and the following) when DPLL\_CTRL\_11.INCF1/2 is switching to 1 and a pulse correction is triggered via DPLL\_CTRL\_1.PCM1/2=1.

### **Scope**

DPLL

## 2 Functional deviations

### Effects

Pulse correction is executed twice. The angle clocks CCM[0]\_TBU\_TS1/2 are misaligned. DPLL\_PSTM, DPLL\_PSSM, DPLL\_PSTM\_OLD, and DPLL\_PSSM\_OLD will have wrong values.

### Workaround

Do not change DPLL\_CTRL\_11.INCF1/2 to 1 while pulse correction is ongoing.

If this cannot be avoided, additional pulse correction might be needed to counteract the double correction of this erratum.

## 2.127 [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

### 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.

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

### Description

Configuration:

## 2 Functional deviations

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.

## 2.129 [GTM\_AI.474] DPLL: DPLL\_PSTC, DPLL\_PSSC erroneously modified

### Description

If a direction change happens while the TRIGGER processing unit is not yet synchronized (`DPLL_STATUS.SYT=0`) then `DPLL_PSTC` is erroneously overwritten.

If a direction change happens while the STATE processing unit is not yet synchronized (`DPLL_STATUS.SYS=0`) then `DPLL_PSSC` is erroneously overwritten.

### Scope

DPLL

### Effects

Update of `DPLL_PSTC` and/or `DPLL_PSSC` after direction change. These values are unreliable then.

### Workaround

Store the `DPLL_PSTC`, `DPLL_PSSC` values outside the DPLL, each time a TRIGGER/STATE input occurs.

If a direction change is detected, overwrite the newly calculated value by the value stored earlier. This is necessary as long as the DPLL is not yet synchronized (`DPLL_STATUS.SYT=0` for `DPLL_PSTC` and/or `DPLL_STATUS.SYS=0` for `DPLL_PSSC`).

## 2 Functional deviations

### 2.130 [GTM\_AI.475] DPLL: Incorrect values of DPLL\_RCDT\_TX, DPLL\_RCDT\_SX

#### Description

If during the reciprocal value calculation an overflow happens then the parameters DPLL\_RCDT\_TX.RCDT\_TX and DPLL\_RCDT\_SX.RCDT\_SX are set erroneously to 0x000000. The specified value is 0xFFFFF.

#### Scope

DPLL

#### Effects

Wrong value is stored in either DPLL\_RCDT\_TX (normal mode) or DPLL\_RCDT\_SX (emergency mode) after a detection of a reciprocal overflow condition. The derived parameters DPLL\_RCDT\_TX\_NOM.RCDT\_TX\_NOM and DPLL\_RCDT\_SX\_NOM.RCDT\_SX\_NOM are diverging accordingly. This leads to a different calculation of the pulse generator frequencies (DPLL\_ADD\_IN\_CAL1.ADD\_IN\_CAL1 or DPLL\_ADD\_IN\_CAL2.ADD\_IN\_CAL2 in dependence of the configured DPLL mode), which might lead to a different settling behavior of the generated angle clocks in such cases.

The diverging settling behavior is not necessarily malicious.

#### Workaround

If a different settling behavior of the DPLL control loop is acceptable no specific countermeasure is necessary.

### 2.131 [GTM\_AI.476] MCS: Unexpected instruction execution while disabling of MCS channel

#### Description

A disable request initiated by a write access MCS[i]\_CH[x]\_CTRL.EN = 0 might cause the following unexpected side effects if the MCS is not configured in Round Robin Scheduling mode and the following conditions are met:

1. Assume that an MCS channel x is disabled after the execution of an instruction instr1. If a potential successor instruction instr2 of instruction instr1 is a memory instruction executing a parallel memory write access and the delay between instr2 and instr1 is up to 3 cluster clock cycles, the write access of instruction instr2 might be executed unexpectedly after the MCS channel is already disabled
2. Assume that an MCS channel x is disabled after the execution of an instruction instr1. If a potential successor instruction instr2 of instruction instr1 is a bus master instruction executing a bus access and the delay between instr2 and instr1 is up to 2 cluster clock cycles, the access of instruction instr2 might be executed unexpectedly after the MCS channel is already disabled

#### Scope

MCS

#### Effects

Unexpected write access to MCS memory or read/write access to the GTM sub module.

#### Workaround

Provide a disabling feature by MCS program, for example:

1. Reserve a memory cell in MCS RAM and define value 1 as a request for an MCS channel disable
2. Instead of writing MCS[i]\_CH[x]\_CTRL.EN = 0 write value 1 to reserved memory cell
3. Poll the reserved memory cell during idle time of MCS program and switch off the MCS channel with instruction MOVL STA 0x0 if the reserved memory cell contains value 1

## 2 Functional deviations

### 2.132 [GTM\_AI.477] DPLL: DPLL\_DCGI interrupt not triggered

#### Description

When synchronous motor control mode is active (DPLL\_CTRL\_1.SMC=1):

If a first direction change together with an input signal change (active edge) has happened, then for a consecutive direction change together with the next following input signal change the interrupt DPLL\_DCGI does not occur.

#### Scope

DPLL

#### Effects

The interrupt DPLL\_DCGI does not occur.

#### Workaround

When a direction change is detected by DPLL\_IRQ\_NOTIFY.DCGI the register DPLL\_STATUS.BWD1 can be checked after the next relevant input signal edge on TRIGGER. If a second direction change is detected with the very next relevant input signal, the DPLL\_DCGI can be set by writing DPLL\_IRQ\_FORCINT.TRG\_DCGI=1. The next relevant input signal edge is the next input signal edge for DPLL\_CTRL\_1.SMC=1 (in contrast to the next inactive input signal edge when DPLL\_CTRL\_1.SMC=0).

### 2.133 [GTM\_AI.478] DPLL: Incorrect calculation of DPLL\_THVAL, DPLL\_THVAL2

#### Description

**Note:** Register names in the text follow the TC3xx syntax conventions. Correlation of register names:

- **TC2xx:** DPLL\_CTRL\_, DPLL\_THVAL
- **TC3xx:** DPLL\_CTRL\_, DPLL\_THVAL, DPLL\_THVAL2

In case of LOW\_RES=1, DPLL\_CTRL\_1.SMC=0, DPLL\_CTRL\_0.IDT=1, and DPLL\_CTRL\_1.TS0\_HRT=0 the values of DPLL\_THVAL, DPLL\_THVAL2 are calculated incorrectly because the filter values are not divided by 8 as specified.

#### Scope

DPLL

#### Effects

Under the described conditions the values of DPLL\_THVAL, DPLL\_THVAL2 are incorrect. The divergence is small, but in theory this can still lead to a wrong direction decision as the THVAL is used for the evaluation of the direction change.

Example:

In case of a 45/90 µs sensor input signal for this failure to happen means to have a difference of the filter values between active and inactive input signal edge on TRIGGER larger than 450 clock cycles in case of a 20 MHz TBU\_TS0 clock configuration.

#### Workaround

If a negative effect on the direction decision is not expected no workaround is necessary. If a negative effect cannot be excluded, the use of the filter values can be switched off by setting DPLL\_CTRL\_0.IDT=0.

## 2 Functional deviations

### 2.134 [GTM\_AI.483] DPLL: Malfunction on changing DPLL\_PVT.PVT

#### Description

If the DPLL is operated in normal mode with DPLL\_CTRL\_11.INCF1 = 1, the calculation of the PVT check starts in the current increment (in the DPLL state DPLL\_STA.STA\_T = 0x58). The decision of whether to admit or to reject the upcoming active TRIGGER is then evaluated upon encountering it.

If DPLL\_PVT.PVT changes after DPLL\_STA.STA\_T = 0x58, the DPLL behaves as follows:

1. If the old DPLL\_PVT.PVT (the one written before DPLL\_STA.STA\_T = 0x58) causes rejection and the new one causes admission: The upcoming TRIGGER will be (partially) rejected in the sense that DPLL\_INC\_CNT1.INC\_CNT1 will not be updated as expected and therefore no micro ticks will be generated. However, erroneously the pointers (DPLL\_APT.APT, DPLL\_APT.APT\_2B, DPLL\_APT\_2C.APT\_2C) will be incremented and the respective memory structures (DPLL\_RDT\_T[p], DPLL\_TSF\_T[p], DPLL\_ADT\_T[p], DPLL\_DT\_T[p]) in RAM Region 2 will be updated. The DPLL\_PWI interrupt will not be triggered although it should be
2. If the old DPLL\_PVT.PVT causes admission and the new one causes rejection:  
The old one is the decisive one. That is, the upcoming TRIGGER will be admitted as expected
3. If both cause rejection:  
The upcoming TRIGGER is rejected (as expected)
4. If both cause admission:  
The upcoming TRIGGER is admitted (as expected)

#### Scope

DPLL

#### Effects

Rejecting a TRIGGER only partially will ultimately lead to a wrong angle clock (CCM[0]\_TBU\_TS1) and wrong PMT calculations.

#### Workaround

Write DPLL\_PVT.PVT only between DPLL\_STA.STA\_T = 0x8 and DPLL\_STA.STA\_T = 0x58. In order to do that timely, MCS[0] can be programmed as follows:

1. Suspend until DPLL\_STA.STA\_T = 0x8 (by means of WURM instruction)
2. Write DPLL\_PVT.PVT on resumption

### 2.135 [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 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.

## **2 Functional deviations**

### **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.

### **2.136 [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>).

### **2.137 [GTM\_AI.490] TOP: Interrupt from DPLL not detected in MCS0**

#### **Description**

Some of the DPLL interrupts can be detected in the internal registers DSTA and DSTAX of MCS0. If the clock divider of cluster 0 is configured to 10<sub>B</sub> (2:1 clock ratio) inside GTM\_CLS\_CLK\_CFG.CLS0\_CLK\_DIV and DPLL\_IRQ\_MODE.IRQ\_MODE is configured to 01<sub>B</sub>, 10<sub>B</sub>, or 11<sub>B</sub>, the interrupt pulse cannot be detected inside MCS0 due to a connectivity failure on GTM\_IP top level and gets lost.

#### **Scope**

MCS, DPLL, TOP

#### **Effects**

Interrupt from DPLL cannot be detected in MCS0.

#### **Workaround 1**

Use interrupt level mode in DPLL by setting of DPLL\_IRQ\_MODE.IRQ\_MODE = 00<sub>B</sub>.

#### **Workaround 2**

Configure the cluster clock divider of cluster 0 to 1 by setting of GTM\_CLS\_CLK\_CFG.CLS0\_CLK\_DIV = 01<sub>B</sub>.

## 2 Functional deviations

### 2.138 [GTM\_AI.492] DPLL: Wrong value of DPLL\_INC\_CNT1.INC\_CNT1 upon switching to normal mode

#### Description

**Note:** Register names in the text follow the TC3xx syntax conventions. Correlation of register and bit names:

- **TC2xx:** DPLL\_, ICM\_IRQG\_1.DPLL\_TAS\_IRQ
- **TC3xx:** DPLL\_, ICM\_IRQG\_1.DPLL\_TASI\_IRQ

DPLL\_CTRL\_0.RMO: 1 -> 0:

Upon switching from emergency to normal mode (with DPLL\_CTRL\_1.SGE1 set to 1), DPLL\_INC\_CNT1.INC\_CNT1 increments by DPLL\_MLS1.MLS1 micro ticks every time an active STATE input is encountered till the first active TRIGGER input is encountered. The extra micro ticks accumulated in DPLL\_INC\_CNT1.INC\_CNT1 will only be generated after encountering the first active TRIGGER.

The described behavior is not intended because the STATE input is not supposed to contribute to the pulse generation in normal mode.

#### Scope

DPLL

#### Effects

##### 1. DPLL\_CTRL\_0.RMO: 1 -> 0

The generation of the extra micro ticks accumulated in DPLL\_INC\_CNT1.INC\_CNT1 after encountering the first active TRIGGER input ultimately leads to wrong angle clock (manifests in wrong CCM[0]\_TBU\_TS1) and wrong PMT calculations due to incorrect DPLL\_PSTC.PSTC.

Further effect, which is only applicable to GTM v4.1.0 devices:

The value of DPLL\_INC\_CNT1.INC\_CNT1 is not assigned to DPLL\_MP\_T.MP\_T on the first active TRIGGER input in contrast to what is specified in MP\_T description in "DPLL\_MP\_T" (GTM4.1 specs: DPLL\_16159).

Further observations without malicious effects:

The value of the current position stamp is not assigned to DPLL\_PSSM.PSSM at the active STATE input in contrast to what is specified in "State description of the State Machine Table" step 21 (GTM4.1 spec.: DPLL\_6908). This is, however, insignificant because DPLL\_PSSM.PSSM is deemed invalid in normal mode (see description of bit-field PSSM in the register DPLL\_PSSM (GTM4.1 spec.: DPLL\_6370)) and therefore should not be used and relied on.

##### 2. DPLL\_CTRL\_0.RMO: 0 -> 1

Upon switching from normal to emergency mode (with DPLL\_CTRL\_1.SGE1 set to 1), the value of the current position stamp is not assigned to DPLL\_PSTM.PSTM at the active TRIGGER input "State description of the State Machine Table" step 1 (GTM4.1 spec.: DPLL\_6908). This is, however, insignificant because DPLL\_PSTM.PSTM is deemed invalid in emergency mode (see description of bit-field PSTM in the register DPLL\_PSTM (GTM4.1 spec.: DPLL\_6360)) and therefore should not be used and relied on.

No effect is associated with not assigning the current position stamp to DPLL\_PSSM.PSSM in normal mode and DPLL\_PSTM.PSTM in emergency mode.

#### Workaround

Two possible workarounds for DPLL\_CTRL\_0.RMO: 1 -> 0

1. Defer setting DPLL\_CTRL\_1.SGE1 to 1 till the first DPLL\_TASI interrupt is encountered (signaling the arrival of the first active TRIGGER)
2. Make sure that DPLL\_MLS1.MLS1 is set to zero upon switching the mode. The user may then alter it on encountering the first DPLL\_TASI interrupt



## **2 Functional deviations**

### **2.139 [GTM\_AI.507] DPLL: Irregular pulse generation and wrong PMT results**

#### **Description**

With the configuration of  $DPLL\_CTRL\_NUTC.NUTE = 2 * (DPLL\_CTRL\_0.TNU + 1) - 1$  for TRIGGER or  $DPLL\_CTRL\_NUSC.NUSE = 2 * (DPLL\_CTRL\_0.SNU + 1) - 1$  for STATE the prediction of the pulse generation frequency is incorrect.

This results in irregular pulse generation and wrong PMT results.

This problem occurs on either CCM[0]\_TBU\_TS1 or CCM[0]\_TBU\_TS2 or both depending on the DPLL operation mode (normal or emergency mode, or synchronous motor control).

#### **Scope**

DPLL

#### **Effects**

1. Wrong pulse generation frequency because of incorrect prediction. This leads to irregular pulse distributions
2. Wrong PMT results

#### **Workaround**

Avoid these configurations:  $DPLL\_CTRL\_NUTC.NUTE = 2 * (DPLL\_CTRL\_0.TNU + 1) - 1$  or  $DPLL\_CTRL\_NUSC.NUSE = 2 * (DPLL\_CTRL\_0.SNU + 1) - 1$ .

Instead, use for example full scale configurations as described in the specification.

### **2.140 [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.

#### **Workaround**

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

### **2.141 [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 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  $(A)TOM[i]_{CH[x]_{CM1.CM1}}=1$  and  $(A)TOM[i]_{CH[x]_{CM0.CM0}}>1$ .

## 2 Functional deviations

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.

### Scope

TOM, ATOM

### Effects

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

### Workaround 1

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

### Workaround 2

Switch the order of the edges, so that (A)TOM[i]\_CH[x]\_CM1.CM0.CM0 defines the first edge and (A)TOM[i]\_CH[x]\_CM1.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, do not use (A)TOM[i]\_CH[x]\_CM1.CM1=(A)TOM[i]\_CH[x]\_CM0.CM0. Instead, use (A)TOM[i]\_CH[x]\_CM1.CM1=1 and (A)TOM[i]\_CH[x]\_CM0.CM0=MAX. However, this only works in the non-fixed (relevant) releases.*

### Workaround 3

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

## 2.142 [GTM\_TC.018] DPLLRAM trace data can be wrong

### Description

**Note:** *This problem only has an effect during debugging.*

The OCDS Trigger Bus Interface supports routing of various trigger sets to MCDS on OTGBM0 and OTGBM1 (see table “Trigger Set Mapping Options” in the GTM chapter).

Tracing of addresses or data of a DPLL RAM on one part of the OTGBM interface (OTGBM0 or OTGBM1, respectively) can create wrong DPLLRAM trace data when any other source (including data or addresses of a different DPLL RAM) is configured for the other part of the OTBGM interface (OTGBM1 or OTGBM0, respectively).

### Workaround

- If DPLLRAM addresses are configured for OTGBM0 (bit-field OTSS.OTGBM0 = 2 or 3 or 4):
  - only DPLLRAM data of the same DPLL RAM may be selected for OTGBM1 (i.e. OTSS.OTGBM1 must be equal to OTSS.OTGBM0);
  - otherwise “No Trigger Set” must be selected for OTGBM1 (OTSS.OTGBM1 = 0)
- If DPLLRAM data are configured for OTGBM1 (bit-field OTSS.OTGBM1 = 2 or 3 or 4):
  - only DPLLRAM addresses of the same DPLL RAM may be selected for OTGBM0 (i.e. OTSS.OTGBM1 must be equal to OTSS.OTGBM0);
  - otherwise “No Trigger Set” must be selected for OTGBM0 (OTSS.OTGBM0 = 0)

## 2 Functional deviations

### 2.143 [GTM\_TC.019] ARU can not be traced if GTM cluster 5 is disabled

#### Description

**Note:** *This problem only has an effect during debugging.*

Tracing of the ARU is controlled by the GTM cluster 5 clock. If cluster 5 is disabled, the ARU can not be traced anymore.

#### Workaround

Do not disable GTM cluster 5 if ARU shall be traced.

### 2.144 [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. 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 Functional deviations

### 2.145 [GTM\_TC.025] Register DPLL\_IRQ\_NOTIFY - Documentation update for bits SORI and DCGI

#### Description

In the description of field SORI (State out of range interrupt) in register DPLL\_IRQ\_NOTIFY in the GTM chapter of the TC3xx User's Manual, the sentence "The interrupt occurs at line number 0" is incorrect.

#### Documentation update

In the description of register DPLL\_IRQ\_NOTIFY, the sentence "The interrupt occurs at line number 0" shall be moved from bit SORI to bit DCGI, as shown below.

**Table 8 Register DPLL\_IRQ\_NOTIFY - Documentation update for bits SORI and DCGI**

Field	Bits	Type	Description
<b>SORI</b>	26	rw	<b>STATE out of range</b> The interrupt occurs at line number 0.
<b>DCGI</b>	27	rw	<b>Direction change interrupt</b> The interrupt occurs at line number 0.

The rest of the description for register DPLL\_IRQ\_NOTIFY remains unchanged.

**Note:** *The assignment of DPLL interrupts to interrupt line numbers DPLL\_IRQ[n] is correctly described in table "ICM Interrupt Signals" in the GTM chapter of the TC3xx User's Manual. According to this table, SORI is assigned to DPLL\_IRQ[26].*

### 2.146 [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  $\mu$ s.

#### 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  $\mu$ s.

### 2.147 [GTM\_TC.028] Incorrect MCS behavior when SSH registers are accessed while MCS is running

#### Description

When performing a read/write access to the GTM MCS SSH registers while the related MCS module is active, the functional MCS SRAM accesses (like program code fetching by MCS) can be disturbed by interfering with the SSH register access. This problem occurs if GTMDIV > 1 is configured in the clock control unit (CCU).

**Note:** *The SSH instances (MCx) and corresponding SSH registers (MTU\_MCi\_\*) are described in the MTU chapter of the product specific appendix to the TC3xx User's Manual.*

## 2 Functional deviations

### Scope

Access to GTM MCS SSH registers

### Effects

The functional MCS SRAM accesses (like program code fetching by MCS) can be disturbed, which will cause the MCS program execution to fail.

### Workaround

Use only setting GTMDIV = 1.

## 2.148 [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 9 Connections of ADC\_TRIGx Signals to ADC/SENT Modules - Correction for ADC\_TRIG4 in TC3Ex and TC38x appendix**

GTM Trigger Signal	EVADC	EDSADC	SENT
<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

**Table 10 Connections of ADC\_TRIGx Signals to ADC/SENT Modules - Correction for ADC\_TRIG4 in TC37x and TC37xEXT appendix**

GTM Trigger Signal	EVADC	EDSADC	SENT
<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

(table continues...)

## 2 Functional deviations

**Table 10** (continued) **Connections of ADC\_TRIGx Signals to ADC/SENT Modules - Correction for ADC\_TRIG4 in TC37x and TC37xEXT appendix**

GTM Trigger Signal	EVADC	EDSADC	SENT
<b>ADC_TRIG4</b>			
ADC_TRIG4_[11:10]	-	G[11:10]REQGTG	G[11:10]REQTRG

**Table 11** **Connections of ADC\_TRIGx Signals to ADC/SENT Modules - Correction for ADC\_TRIG4 in TC36x appendix**

GTM Trigger Signal	EVADC	EDSADC	SENT
<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 12** **Connections of ADC\_TRIGx Signals to ADC/SENT Modules - Correction for ADC\_TRIG4 in TC33x/TC32x appendix**

GTM Trigger Signal	EVADC	SENT
<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]	-	-

### 2.149 [HSCT\_TC.012] HSCT sleep mode not supported

#### Description

Due to unreliability of the wake-up functionality, sleep mode for the HSCT is no longer supported and shall not be used.

Do not set bit SLEEPCTRL.SLPEN = 1<sub>B</sub>.

### 2.150 [HSCT\_TC.013] Internal loopback mode not reliable

#### Description

The internal loopback mode used for looping the HSCT TX data back internally to HSCT RX is not reliable to work under all operating conditions.

Therefore, do not use the internal loopback mode (i.e. do not set bit TESTCTRL.LLOPTXRX = 1<sub>B</sub>).

## **2 Functional deviations**

### **Workaround**

Use external loopback by configuring another external device (slave) by sending an interface control command with payload value = 0xFF (Turn on payload loopback) from the master interface.

### **2.151 [MCDS\_TC.052] TriCore™ wrap around write access causes redundant MCDS message**

#### **Description**

**Note:** *This problem is only relevant for development tools. This problem corresponds to problem miniMCDS\_TC.005 for devices with miniMCDS.*

This effect will only occur for Circular Addressing Mode when there is an unaligned write access at the wrap around boundary. There is no known case where such an access would be generated for normal compiled code. When TriCore™ performs such an access, MCDS generates a redundant message.

#### **Example**

TriCore™ writes 0x87654321 to address 0xAF01F90E, but due to wrap around it first writes 0x4321 to 0xAF01F90E and then writes 0x8765 to 0xAF01F900.

MCDS outputs the following messages:

- DTU\_<AorB>\_TCX.DTW 0x8765 0xAF01F900
  - i.e. writing HWORD(8765) to AF01F900
- DTU\_<AorB>\_TCX.DTW 0x87654321 0xAF01F90E
  - i.e. writing WORD(87654321) to AF01F90E

#### **Workaround**

No workaround possible.

### **2.152 [MCDS\_TC.064] ACCEN0 register write not supervisor protected**

#### **Description**

**Note:** *This problem is only relevant for development tools.*

The write access terms for register ACCEN0 are defined as “SV, SE” (access only when Supervisor Mode/Safety Endinit is active) in table “Registers Overview” in the MCDS and MCDSLlight chapter of the TC3XX\_ED documentation.

Actually, the implementation of register ACCEN0 in the MCDS and MCDSLlight modules does not have Supervisor Mode protection (“SV”), it only has Safety Endinit protection (“SE”).

### **2.153 [MCDS\_TC.065] Selection of SRI trace sources**

#### **Description**

**Note:** *This problem is only relevant for development tools. It corresponds to problem miniMCDS\_TC.006 for devices with miniMCDS.*

The MCDS/MCDSLlight provides the capability to trace accesses to the SRI Slaves CPUx\_MEMSlave, LMU0, OLDA, to trace accesses initiated by the SRI Master DMA, and to trace SPU output events.

## **2 Functional deviations**

The signal timing at these interfaces is compliant to the Infineon internal SRI bus protocol. The bus protocol consists of an address and data phase. It is not possible to select one of the above trace sources while transactions are on-going. This can lead to permanently corrupted MCDS trace messages.

### **Workaround**

Switch to the trace source only in a time frame when no transactions are on-going.

### **2.154 [MCDS\_TC.066] Selection of CPU trace sources**

#### **Description**

**Note:** *This problem is only relevant for development tools. It corresponds to problem miniMCDS\_TC.007 for devices with miniMCDS.*

The MCDS/MCDSlight provides the capability to trace CPU read/write accesses to registers and memories. The signal protocol at the CPU trace interface has separated address and data phases. If another CPU is selected as trace source while the CPU is running, this can lead to the effect that no MCDS data trace messages are generated anymore.

### **Workaround**

Switch to another CPU trace source only at a time when no CPU transactions are ongoing (e.g. CPU halted).

### **2.155 [MCDS\_TC.067] MCDS kernel reset shall not be used**

#### **Description**

**Note:** *This problem is only relevant for development tools. It corresponds to problem miniMCDS\_TC.008 for devices with miniMCDS.*

If an MCDS/MCDSlight kernel reset is triggered via bit CT.SETR, this can confuse the data trace generation with the duplex DTUs. In this case data trace messages are missing and wrong data trace messages are generated where the data part and the address part are from different data transactions. This confused state of the DTU is then permanent.

### **Workaround**

Do not use the MCDS/MCDSlight kernel reset.

In case of reprogramming set all used MCDS/MCDSlight configuration registers to new values or to their reset values.

### **2.156 [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



## **2 Functional deviations**

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  $t_q$  (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- $t_q$ -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.

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.

## **2.157 [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.

## **2 Functional deviations**

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.

## **2.158 [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.*

## **2 Functional deviations**

### **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
  - 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.

## **2.159 [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")

## 2 Functional deviations

- 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:

- 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

## **2 Functional deviations**

- 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.

### **2.160 [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.

#### **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.

### **2.161 [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

## **2 Functional deviations**

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.

### **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.

## **2.162 [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.

## 2 Functional deviations

### 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.

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

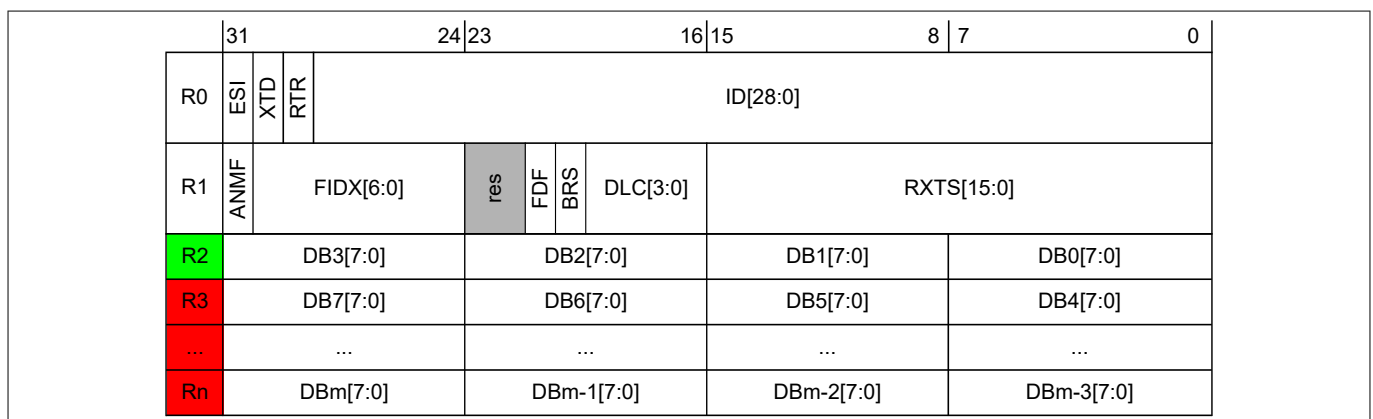
### Description

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 ( $f_{MCANH}$ )
- 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 ( $n \leq 17$ ).



**Figure 9 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



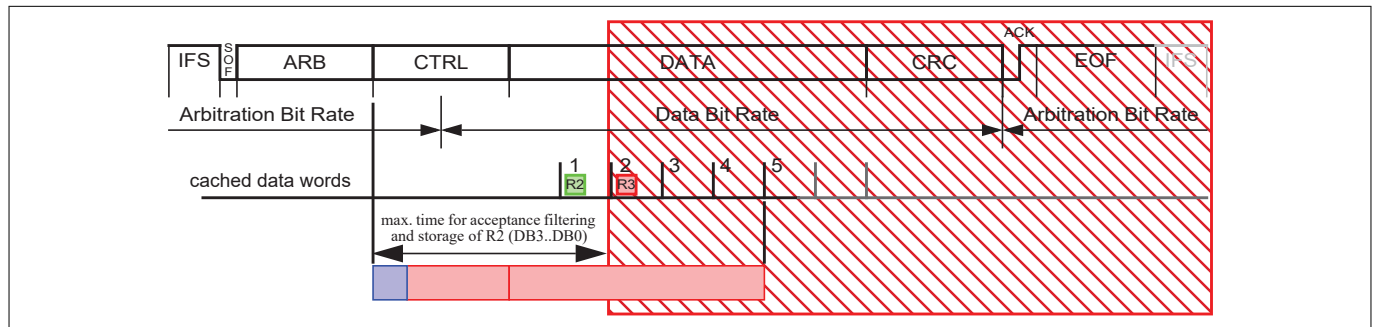
## 2 Functional deviations

The data will be corrupted in a way, that in the Message RAM  $R_{(i+1)}$  has the same content as  $R_i$ .  
 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.



**Figure 10** CAN Frame with DLC > 4

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

**Table 13** TC3xx: Minimum host clock frequency for CAN FD when DLC = 5

No. of configured Active FE <sup>1) 2)</sup> 11-bit IDs/ 29-bit IDs	Number of Active CAN nodes	Arbitration bit rate = 0.5 Mbit/s			Arbitration bit rate = 1 Mbit/s		
		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.



## 2 Functional deviations

### 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".

### 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.

## 2.164 [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, MECDR, 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.

## **2 Functional deviations**

### **Effects**

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

**Note:** *This issue has no safety impact since the node-based access protection covered by ACCENCTR0 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.

## **2.165 [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.

## **2.166 [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\_DMEN)
- MC34 (CPU0\_DMEN1), and
- MC35 (CPU1\_DMEN1)

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

## 2 Functional deviations

### 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.

### 2.167 [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:  
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 ≠ 000B`) - 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 Functional deviations

### 2.168 [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>3)</sup>
- analog inputs (class D pads) of channels with multiplexer diagnostics<sup>4)</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).*

### 2.169 [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.

### 2.170 [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.

#### 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.

<sup>3</sup> Availability depends on TC3xy device version, see the product specific Data Sheet.

<sup>4</sup> These channels are explicitly marked with (MD) in table “Analog Input Connections for Product TC3yx” in the EVADC chapter of the product specific appendix to the AURIX™ TC3XX User’s Manual.

## 2 Functional deviations

### 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.

## 2.171 [PMS\_TC.005] Voltage rise at P33 and P34 up to $V_{EVR SB}$ during start-up and up to $V_{LVDRST SB}$ 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  $V_{EVR SB}$  and  $V_{EXT}$  ramp up. Tristate control information based on HWCFG[6] latched with  $V_{EXT}$  supply ramp can't be used within the  $V_{EVR SB}$  supply domain until both supplies ( $V_{EXT}$  and  $V_{EVR SB}$ ) have reached the minimum threshold value of  $V_{LVDRST5}$  and  $V_{LVDRST SB}$ , 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  $V_{EVR SB}$  until the  $V_{LVDRST SB}$  and  $V_{LVDRST5}$  thresholds of  $V_{EVR SB}$  and  $V_{EXT}$  are reached during the ramp-up phase
- the pad voltage level is below  $V_{LVDRST SB}$  for the ramp-down phase of the  $V_{EVR SB}$  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).

## 2.172 [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

## 2 Functional deviations

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.

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

### 2.173 [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.

### 2.174 [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



## 2 Functional deviations

### 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".

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.

### 2.175 [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 ( $V_{DDM} + V_{Diodc}$ ) 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.*

### 2.176 [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 SYNCMADEV and SYNCHYST in register EVRSDCTRL11 (chapter PMS) and EVRSDCTRL2 (chapter PMSLE) of the TC3xx User's Manual shall be corrected/updated as listed below.

#### SYNCMADEV in TC3xx User's Manual V2.0.0 (and earlier versions)

- $d f_{MAXDEV} = 100 \text{ MHz} * (2 * SYNCMADEV) / (SDFREQ^2 + SYNCMADEV^2)$
- $SYNCMADEV = \text{round} [ (100 \text{ MHz} / d f_{MAXDEV}) - \sqrt{(100 \text{ MHz} / d f_{MAXDEV})^2 - SDFREQ^2} ]$

#### Correction to SYNCMADEV in register EVRSDCTRL11 (PMS) and EVRSDCTRL2 (PMSLE)

- $d f_{MAXDEV} = 100 \text{ MHz} * (2 * SYNCMADEV) / (SDFREQ^2 - SYNCMADEV^2)$
- $SYNCMADEV = \text{round} [\sqrt{(100 \text{ MHz} / d f_{MAXDEV})^2 + SDFREQ^2} - (100 \text{ MHz} / d f_{MAXDEV}) ]$

#### SYNCHYST in TC3xx User's Manual V2.0.0 (and earlier versions)

- $SYNCHYST = \text{round} [ d f_{HYST} * (SDFREQ \pm SYNCMADEV)^2 ] / [ d f_{HYST} * (SDFREQ \pm SYNCMADEV) + 100 \text{ MHz} ]$

#### Correction/Update to SYNCHYST in register EVRSDCTRL11 (PMS) and EVRSDCTRL2 (PMSLE)

- $SYNCHYST = \text{round} [ d f_{HYST} * (SDFREQ \pm SYNCMADEV)^2 / (100 \text{ MHz} \pm d f_{HYST} * (SDFREQ \pm SYNCMADEV)) ]$
- First hysteresis band:
  - $d f_{HYST} = 100 \text{ MHz} / (SDFREQ + SYNCMADEV - SYNCHYST) - 100 \text{ MHz} / (SDFREQ + SYNCMADEV)$
- Second hysteresis band:
  - $d f_{HYST} = 100 \text{ MHz} / (SDFREQ - SYNCMADEV) - 100 \text{ MHz} / (SDFREQ - SYNCMADEV + SYNCHYST)$

### 2.177 [PSI5\_TC.005] Incorrect read pointer upon two consecutive RDFn read operations if two or more channels are configured

#### Description

Whenever two (or more) channels (CHm, CHn) are configured, the read pointer REP of CHn is not incremented correctly if the CHn FIFO read access (two consecutive accesses to RDFn) is performed one cycle before new data of CHm is written into its FIFO.



## 2 Functional deviations

### Scope

FIFO usage

### Effects

Incorrect read pointer.

### Workaround

1. Perform additional plausibility check every time there is an attempt to read the data from the receive data memory. Plausibility check verifies that the REP value is incremented after the two consecutive read operations by 2
2. If 1. is not the case, then read the stored sensor data through RDML/Hny registers, and once all the previously stored data is read, flush the FIFO buffer structure by setting bit RFCn.FLU (this will reset the WRP and REP values). From this point on, data could be read from the receive data memory through RDFn register

## 2.178 [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.

## 2.179 [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**

### **2.180 [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.

### **2.181 [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 Tx FIFO 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.

### **2.182 [QSPI\_TC.014] Slave: Incorrect parity bit upon Tx FIFO underflow**

#### **Description**

When a slave Tx FIFO 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.

### **2.183 [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.

### 2.184 [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>

### 2.185 [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.

### 2.186 [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.

### **2.187 [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.

### **2.188 [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.

### **2.189 [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.”

### **2.190 [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>).

### **2.191 [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.

### **2.192 [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 Functional deviations

### 2.193 [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

### 2.194 [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.

#### 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.

### 2.195 [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).

## 2 Functional deviations

**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).

## 2.196 [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.

## 2.197 [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.

### Workaround

If using interrupt generation on both edges, ensure that the time between two triggering edges for EXINT0, EXINT1 is > 2 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  $2.1/f_{CCLK}$  to ensure that the transitions are correctly recognized.

Otherwise, only use external interrupts EXINT2..15.



## 2 Functional deviations

### 2.198 [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 14** Field ADRES in register ADCOMP\_RES - correction

Field	Bits	Type	Description
ADRES	7:0	rh	<b>ADC Conversion Result</b>
			<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>

### 2.199 [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).

EVRC 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 Functional deviations

### 2.200 [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 15** 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
TC36x	AA	0xD833B421	0xF53338B3
TC35x	AB	0xBA38B3CB	0x1CACD3CE
TC33xEXT	AA	0xD1298927	0x1A824479
TC33x/TC32x	AA	0xC12B66CB	0x3F84CD94
TC3Ex	AA	0x4D0F493B	0xE7764C81

### 2.201 [SDMMC\_AI.001] Timeout error when BOOT ACK driven on all data lines in SD/eMMC mode

#### Description

Defect summary:

In eMMC mode during Boot operation, when the device responds with BOOT ACK in all the data lines for 4/8 bit data widths, timeout/CRC error is reported by the controller because it erroneously detects the start bit of BOOT ACK as start bit of the BOOT Data.

#### Impacted configurations

Configurations with the following register settings:

- BOOT\_CTRL\_R.BOOT\_ACK\_ENABLE=1
- HOST\_CTRL1\_R.DAT\_XFER\_WIDTH=1 or HOST\_CTRL1\_R.EXT\_DAT\_XFER=1

#### Consequence(s)

- Unexpected event requiring software interrupt

## 2 Functional deviations

- System benignly self-recovers
- Persistent data corruption/loss

### Method of reproducing

Enabling BOOT ACK with 4/8 bit data widths, and driving BOOT ACK on all the data lines will reproduce this issue.

### Workaround 1

Drive BOOT ACK on D0 only.

### Workaround 2

Disable the BOOT ACK feature during BOOT.

## 2.202 [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);

## 2 Functional deviations

- 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.

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).

## 2.203 [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 Functional deviations**

### **2.204 [SMU\_TC.015] SMU alarm emulation might trigger unwanted active alarm reaction**

#### **Description**

While the SMU is in START state, an alarm is raised by the hardware. 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 either by raising a software alarm or by 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.

#### **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, or before triggering a software alarm.

### 3 Parametric deviations

## 3 Parametric deviations

### 3.1 [ADC\_TC.P009] Increased TUE for G10 when using Alternate Reference

#### Description

When using the alternate reference (G10CH0) for conversions on channels of converter group G10, the Total Unadjusted Error (TUE) of the conversion results may increase with temperature

- up to  $\pm 12 \text{ LSB}_{12}$  for operation with converter reference clock  $f_{\text{ADCI}} = 32 \text{ MHz}$
- up to  $\pm 25 \text{ LSB}_{12}$  for operation with converter reference clock  $f_{\text{ADCI}} = 40 \text{ MHz}$
- up to  $\pm 46 \text{ LSB}_{12}$  for operation with converter reference clock  $f_{\text{ADCI}} = 53.33 \text{ MHz}$

**Note:** This problem will not occur in the lower range of the ADC analog supply voltage ( $V_{\text{DDM}} < 4.5 \text{ V}$ ), as  $f_{\text{ADCI}}$  is limited to 26.67 MHz in this case (see Data Sheet).

#### Recommendation

- Do not use the alternate reference of converter group G10 for  $f_{\text{ADCI}} > 26.67 \text{ MHz}$
- Use a different converter group Gx ( $x \neq 10$ ) when an alternate reference voltage is required for conversions with  $f_{\text{ADCI}} > 26.67 \text{ MHz}$

### 3.2 [ADC\_TC.P015] Increased RMS noise for TC374\* and TC375\* devices

#### Description

**Note:** This problem only affects TC374\* and TC375\* devices (devices in PG-QFP-144 and -176 packages). For TC375TI see section 2, for all other TC374\* and TC375\* devices see section 1.

#### 1. For TC374\* and TC375\* except TC375TI

On every analog channel ANx, the specified RMS noise ( $EN_{\text{RMS}}$ ) increases, independent of the noise reduction mode.

- For analog channel AN0 and AN1:
  - $EN_{\text{RMS}} (\text{Max.}) \leq 2.7 \text{ LSB}_{12}$  in the temperature range  $T_A > -40^\circ\text{C}$
- For analog channels AN2 to AN8:
  - $EN_{\text{RMS}} (\text{Max.}) \leq 2 \text{ LSB}_{12}$  in the temperature range  $-40^\circ\text{C} < T_A \leq 0^\circ\text{C}$
  - $EN_{\text{RMS}} (\text{Max.}) \leq 1.7 \text{ LSB}_{12}$  for  $T_A > 0^\circ\text{C}$
- For all other analog channels ANx ( $x > 8$ ):
  - $EN_{\text{RMS}} (\text{Max.}) \leq 1.2 \text{ LSB}_{12}$  in the temperature range  $T_A > -40^\circ\text{C}$

#### 2. For TC375TI

On several analog input channels the specified RMS noise ( $EN_{\text{RMS}}$ ) increases, independent of the noise reduction mode. The level of deviation is dependent on the ambient temperature ( $T_A$ ) and the EVRC switching frequency (EVRSDCTRL0.SDFREQ).

- EVRC switching frequency 1.8 MHz (EVRSDCTRL0.SDFREQ=37<sub>H</sub>)
  - For analog channel AN0:
    - $EN_{\text{RMS}} (\text{Max.}) \leq 1.6 \text{ LSB}_{12}$  in the temperature range  $-40^\circ\text{C} < T_A \leq 25^\circ\text{C}$
    - $EN_{\text{RMS}} (\text{Max.}) \leq 1.4 \text{ LSB}_{12}$  in the temperature range  $T_A > 25^\circ\text{C}$

### 3 Parametric deviations

- For analog channel AN1:
  - $EN_{RMS}$  (Max.)  $\leq 1.4 \text{ LSB}_{12}$  in the temperature range  $-40^{\circ}\text{C} < T_A \leq 25^{\circ}\text{C}$
  - $EN_{RMS}$  (Max.)  $\leq 1.2 \text{ LSB}_{12}$  in the temperature range  $T_A > 25^{\circ}\text{C}$
- For analog channels AN2 and AN3:
  - $EN_{RMS}$  (Max.)  $\leq 1.2 \text{ LSB}_{12}$  in the temperature range  $-40^{\circ}\text{C} < T_A \leq 25^{\circ}\text{C}$
  - $EN_{RMS}$  (Max.)  $\leq 1 \text{ LSB}_{12}$  in the temperature range  $T_A > 25^{\circ}\text{C}$
- For all other analog channels ANx (x>3):
  - $EN_{RMS}$  (Max.)  $\leq 1 \text{ LSB}_{12}$  in the temperature range  $T_A > -40^{\circ}\text{C}$
- EVRC switching frequency 0.8 MHz (EVRSDCTRL0.SDFREQ=7D<sub>H</sub>)
  - For analog channel AN0:
    - $EN_{RMS}$  (Max.)  $\leq 1.4 \text{ LSB}_{12}$  in the temperature range  $-40^{\circ}\text{C} < T_A \leq 25^{\circ}\text{C}$
    - $EN_{RMS}$  (Max.)  $\leq 1.2 \text{ LSB}_{12}$  in the temperature range  $T_A > 25^{\circ}\text{C}$
  - For analog channel AN1:
    - $EN_{RMS}$  (Max.)  $\leq 1.2 \text{ LSB}_{12}$  in the temperature range  $-40^{\circ}\text{C} < T_A \leq 25^{\circ}\text{C}$
    - $EN_{RMS}$  (Max.)  $\leq 1 \text{ LSB}_{12}$  in the temperature range  $T_A > 25^{\circ}\text{C}$
  - For all other analog channels ANx (x>1):
    - $EN_{RMS}$  (Max.)  $\leq 1 \text{ LSB}_{12}$  in the temperature range  $-40^{\circ}\text{C} < T_A \leq 25^{\circ}\text{C}$
    - $EN_{RMS}$  (Max.)  $\leq 0.8 \text{ LSB}_{12}$  in the temperature range  $T_A > 25^{\circ}\text{C}$

### 3.3 [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.

### 3.4 [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

- $t_{MERP} \leq 0.52 \text{ s}$  (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:**  $t_{ER\_Dev} \leq 19.14 \text{ s}$  (instead of 18.5 s as specified in the Data Sheet)
- **TC38x:**  $t_{ER\_Dev} \leq 11.9 \text{ s}$  (instead of 11.5 s as specified in the Data Sheet)
- **TC3Ex:**  $t_{ER\_Dev} \leq 14.98 \text{ s}$  (instead of 14.5 s as specified in the Data Sheet)
- **TC37x, TC37xEXT:**  $t_{ER\_Dev} \leq 7.24 \text{ s}$  (instead of 7 s as specified in the Data Sheet)
- **TC35x, TC36x:**  $t_{ER\_Dev} \leq 5.16 \text{ s}$  (instead of 5 s as specified in the Data Sheet)
- **TC33xEXT, TC33x/TC32x:**  $t_{ER\_Dev} \leq 3.08 \text{ s}$  (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

#### 3.5 [PADS\_TC.P014] Electrical characteristics for P20.2/ $\overline{\text{TESTMODE}}$

##### Description

The buffer type for P20.2/ $\overline{\text{TESTMODE}}$  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  $V_{\text{DDM}}$ .

##### Recommendation

Tables "Class S 5V" and "Class S 3.3V" apply analogously to the electrical characteristics of P20.2/ $\overline{\text{TESTMODE}}$ , with  $V_{\text{DDM}}$  replaced by  $V_{\text{EXT}}$ .

**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.

#### 3.6 [PORST\_TC.P002] $V_{\text{IH}}$ and $V_{\text{IL}}$ 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  $V_{\text{IH}}$ ) and "Input low voltage level" (symbol  $V_{\text{IL}}$ ):

**Note:** The levels defined are valid within the operating conditions of  $V_{\text{EXT}} = 5 \text{ V} \pm 10\%$  or  $V_{\text{EXT}} = 3.3 \text{ V} \pm 10\%$ , respectively.

## 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):
    - $I_{BWG} = V_{AIN} \cdot C_{AINS} \cdot CR$
  - Broken Wire Detection against  $V_{AREF}$  (BWR):
    - $I_{BWR} = (V_{AIN} - V_{AREF}) \cdot C_{AINS} \cdot CR$

## 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.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.

## 4.6 [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 (GLOB\_CFG.USC=0, default after reset), each calibration step is waiting for a valid starting point and this prolongs the total calibration time

## 4 Application hints

### 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  $t_{\text{SUCAL}}$ :

- <sup>2)</sup> In the synchronized mode ( $\text{USC}=0$ ),  $t_{\text{SUCAL}}$  can increase up to 200%

## 4.7 [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  $V_{\text{AGND}}$ ) or to charge (BWD against  $V_{\text{AREF}}$ ) the input node of the EVADC. The level ( $V_{\text{AREF}}$  or  $V_{\text{AGND}}$ ) is selected in the bit-field BWDCH as documented in the description of register GxCHCTRY in the EVADC chapter of the TC3xx user manual.

However, the text in the EVADC chapter only describes BWD against  $V_{\text{AGND}}$ , it does not explicitly describe BWD against  $V_{\text{AREF}}$ .

### 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  $V_{\text{AGND}}$  and BWD against  $V_{\text{AREF}}$ :

- 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  $V_{\text{AGND}}$  or  $V_{\text{AREF}}$ ) 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 BWDEN in the corresponding channel control register (GxCHCTRY). The bit-field BWDCH selects the level for the preparation phase

## 4.8 [AGBT\_TC.H004] Configuration of registers PYCR2 and PACR2

### Description

Settings different to the reset configuration are required for the following fields of registers PYCR2 and PACR2:

- Register PYCR2.[26:24] = 100<sub>B</sub>(CKRXTERM)
- Register PYCR2.[12:8] = 01101<sub>B</sub>(TXOCDSLE)
- Register PACR2.[26:24] = 010<sub>B</sub>(PICAPSEL)

## 4.9 [AGBT\_TC.H005] Availability of AGBT depending on feature package

### Description

The Data Sheets of TC39x, TC37xEXT, TC35x and TC33xEXT in general refer to all variants of the respective product family, including their specific emulation devices. Therefore, table “Platform Feature Overview” also includes the AGBT module.

## 4 Application hints

**Note:** *The Aurora Gigabit Trace Module (AGBT) is a trace interface intended for development use only (not to be used in series production). It is only available on the specific emulation devices with feature package E, T, and on ADAS devices with feature package A, H of the TC39x, TC37xEXT, TC35x, TC33xEXT. AGBT I/O functions are only available for packages with 292 or more pins.*

For details on AGBT parameters see the “TC3xx Emulation Devices” Data Sheet.

### 4.10 [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).

### 4.11 [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”

### 4.12 [ASCLIN\_TC.H007] Handling TxFIFO and Rx FIFO interrupts in single move mode

#### Present description for TxFIFO 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 TxFIFO as full as possible, refilling the TxFIFO 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 TxFIFO 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.

## **4 Application hints**

### **Documentation update for TxFIFO single move mode**

If TxFIFO 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 RxFIFO single move mode**

As described in section “Single Move Mode” of chapter “RxFIFO interrupt generation” in the user manual, the purpose of the Single Move Mode is to keep the RxFIFO as empty as possible, by fetching the received elements 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.

## **4.13 [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  $> 2 \times t_{52}$ .

## **4.14 [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.

## 4 Application hints

### Recommendation

Use cold PORST if lockstep is disabled and shall be re-enabled upon the reset trigger.

### 4.15 [BROM\_TC.H011] Assertion of ALM7[14] after cold or warm PORST

#### Description

After a cold or warm PORST, if no global PFLASH or DFLASH read protection is set and no Halt After Reset has been requested, the startup software checks for a 64-bit signature at 3 predefined locations within EMEM as part of the prolog code handling (see chapter “Startup with Prolog Code in EMEM” in the TC3xxED documentation).

This may trigger ALM7[14] (EMEM ECC Error) if the EMEM is not ECC-clean initialized at that time.

**Note:** *No entry into the ETRR registers is made due to this issue.*

### Recommendation

Ignore ALM7[14] after cold or warm PORST if the corresponding ETRR registers contain no entry.

### 4.16 [BROM\_TC.H012] Availability of $V_{DDSB}$ during start-up

#### Description

Devices with integrated EMEM have a separate EMEM SRAM Standby Power Supply ( $V_{DDSB}$ ).

As documented e.g. in chapter “Power Supply Concept” in the TC3xxED documentation

- $V_{DDSB}$  has to be supplied when  $V_{DD}$  is supplied and the EMEM is unlocked
- $V_{DDSB}$  can be unsupplied when  $V_{DD}$  is supplied and PORST is active or the EMEM is locked
- $V_{DDSB}$  can be supplied when  $V_{DD}$  is unsupplied and PORST is active (EMEM standby mode)

**Note:** *If  $V_{DDSB}$  is not supplied at PORST release and the EMEM is unlocked, cross current from  $V_{DD}$  domain could lead to device damage.*

Software may check availability of  $V_{DDSB}$  via bit SBRCTR.STBPON when EMEM is unlocked.

During start-up, however, there are situations where the EMEM is unlocked for a short time without checking SBRCTR.STBPON before.

### Recommendation

The external system must ensure that  $V_{DDSB}$  is within the active operation range ( $1.25\text{ V} \pm 10\%$ , see Data Sheet) after PORST release during startup.

### 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

## 4 Application hints

- **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.H015] Different initial values for CPU0\_PMEM SSH registers in MTU after cold PORST if SOTA/SWAP is enabled

### Description

If SOTA/SWAP functionality is enabled via the SOTA Mode Enable (UCB\_OTP.PROCONT.PSWAPEN), and a cold PORST is performed, registers ECCD, ETRRx and ERRINFOx in MC2 (associated with CPU0\_PMEM) may contain “false-positive signatures”, indicating correctable or uncorrectable ECC errors. However, these are no real errors but result from firmware side effects (prefetches to - at that time - uninitialized memory).

### Recommendation

Execute the following code sequence during startup (e.g. before MBIST or other safe application startup routines) in order to reverse this effect:

```
Ifx_MTU_MC *mc = &MODULE_MTU.MC[IfxMtu_MbistSel_cpu0Pspr];  
mc->ECCD.B.TRC = 1; // Clears EOv, VAL bits plus the ETRR and ERRINFO registers  
mc->ECCD.B.CERR = 0; // Clears CERR and enables further alarms to be forwarded to SMU  
---
```

**Note:** Resulting signatures are matching with AURIX™ TC3xx Safety Manual Appendix A.

## 4.19 [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. install 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

### 4.20 [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  $f_{SPB}$ .

### 4.21 [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.*

### 4.22 [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<sub>B</sub> or 1<sub>B</sub>) 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 <sub>B</sub> No disable requested 1 <sub>B</sub> 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

## 4.23 [CPU\_TC.H019] Semaphore handling for shared memory resources

### Description

In a multiprocessor system, sharing state between different cores is generally guarded by semaphores or mutexes.

In AURIX™ TC3xx and TC2xx devices specific synchronization steps are needed to achieve specific results for programs running concurrently on multiple processors.

Special care needs to be taken in software when guarded state and semaphores are located in different memory modules.

When the paths from two CPUs to common memory resources are not the same for both CPUs, the effect of two generic stores from one CPU can appear in the opposite order to two generic loads from the other CPU if correct synchronization steps are not taken. This can happen when the master releasing the semaphore has a different access path to a shared resource than to its associated semaphore. In this case, it is possible for another master to observe the semaphore update prior to the final update of the guarded state.

In order to guarantee that the guarded state update is globally visible, both correct sequence and correct synchronization are required. A master must first acquire the semaphore to ensure correct synchronization. It is also required to include a DSYNC in the semaphore acquire and release methods. DSYNC waits until the store buffer is empty and then DSYNC completes ensuring correct sequence. In a multi-domain crossbar where one of the paths from the master to the shared resource involves an SRI extender, additional steps are required to ensure correct sequence. In such a case it is highly recommended to locate the semaphore and shared buffer in the same memory module.

### Operational Details

From a CPU's point of view, resources can be accessed in different ways:

- Local resource to the CPU
  - Local DSPR
  - Local DLMU (AURIX™ TC3xx)

#### 4 Application hints

- SRI accessed resource
  - Any resource accessed via the SRI on the local crossbar
- SRI accessed resource via SRI bridge (AURIX™ TC3xx)
  - Any resource located behind an SRI to SRI bridge in a multi-domain crossbar (relative to the accessing master)

In the case of multi-domain crossbars connected by SRI to SRI bridges there may be multiple paths of different latency from masters to shared resources potentially involving different bridges. When the guarded state is a shared memory location, the sequence observed by each master is guaranteed to be the same as long as the semaphore and guarded state are located in the same memory module. If semaphore and guarded state are not located in the same memory module then a load from the module is required prior to releasing the semaphore.

In order to achieve correct synchronization between the different masters, correct semaphore handling is required.

#### Acquiring and Releasing semaphores - Recommendations

In order to ensure correct sequence and synchronization a DSYNC instruction should be used as part of the semaphore acquire and release sequences. Additionally, a typical use case always requires the acquisition of the semaphore prior to accessing the guarded resource. The DSYNC waits until the store buffer is empty and then completes.

- Acquiring semaphores: A sequence of atomic compare and swap followed by a DSYNC
- Releasing semaphores: A sequence of DSYNC followed by the clearing of the semaphore

#### Examples

The following examples refer to memory accesses to non-peripheral regions (i.e. segments 0<sub>H</sub> ..D<sub>H</sub>). These examples are just describing the memory operations and not the complete sequence of operations

#### Example 1a: Out of order memory access due to different access paths to semaphore and shared resource

In this example, the semaphore is local to CPU<sub>x</sub> and the resource is local to CPU<sub>y</sub>. CPU<sub>x</sub> already owns the semaphore at the start of the described sequence. CPU<sub>y</sub> has not acquired the semaphore prior to accessing the resource.

**Table 17**      **Example 1a: Out of order memory access due to different access paths to semaphore and shared resource**

CPU <sub>x</sub>	CPU <sub>y</sub>	Memory Access Sequence
st-1 (resource-update)	ld-1 (semaphore-check)	
st-2 (semaphore-release)	ld-2 (resource-read)	
		st-2 (semaphore-release)
		ld-1 (semaphore-check)
		ld-2 (resource-read) "stale data"
		st-1 (resource-update)

#### Example 1b: Access order is enforced by correct semaphore handling

**Table 18**      **Example 1b: Access order is enforced by correct semaphore handling**

CPU <sub>x</sub>	CPU <sub>y</sub>	Memory Access Sequence
st-1 (resource-update)	CMPSWAP.W (semaphore-acquire)	
DSYNC	DSYNC	

(table continues...)

## 4 Application hints

**Table 18** (continued) **Example 1b: Access order is enforced by correct semaphore handling**

CPUx	CPUy	Memory Access Sequence
st-2 (semaphore-release)	ld-1(resource-read)	
		st-1 (resource-update)
		st-2 (semaphore-release)
		CMPSWAP.W (semaphore-acquire)
		ld-1(resource-read)

A master may only access a resource if the associated semaphore is acquired successfully.

**Note:** *CMPSWAP.W is only used here as an example. TriCore™ provides several other instructions supporting the implementation of semaphore operations*

### 4.24 [CPU\_TC.H021] Resource update failure despite correct SW synchronization upon retried FPI write transactions by CAN and E-Ray modules

#### Description

**Note:** *Module names in the text follow the TC3xx syntax conventions. Correlation of module names:*

- **TC3xx:** MCMCAN
- **TC2xx:** MultiCAN+

In a multiprocessor system, sharing the same resource between different CPUs is generally guarded by semaphores or mutexes. A DSYNC instruction is used in the semaphore's acquire and release methods in order to guarantee correct synchronization between the CPUs.

In certain situations, peripherals including MCMCAN and E-Ray may not immediately accept some write operations and they remain pending and will be retried. Ordinarily this behavior is invisible to the system as the CPU's subsequent FPI transactions will be delayed till the operation is complete.

In this scenario, CPUx's (which has the semaphore) execution of DSYNC incorrectly views the pending store operation as complete and itself completes too early. CPUx then releases the semaphore, allowing the other CPU (CPUy) to acquire the semaphore and commence accessing the shared resource.

Under certain circumstances, the pending write operation by CPUx may be retried multiple times by the module and may still not have completed. This can lead to CPUy accessing the MCMCAN or E-Ray module before the state intended by CPUx has been established. An example sequence in the table below shows the incorrect behavior.

**Table 19** **Possible SW/HW interaction producing incorrect state**

CPUx	CPUy	Incorrect resource access sequence
Resource update (store instruction)	Semaphore check	CPUx – Resource update first attempt (becomes pending FPI write)
DSYNC	...	CPUx - DSYNC incorrectly completes
Semaphore release	...	CPUx – Semaphore release
...	Semaphore acquire	CPUy – Semaphore acquire
...	Resource read (load instruction)	CPUy – Resource read (incorrect state)
...	...	CPUx - Pending FPI write succeeds (Resource update completed)

## **4 Application hints**

### **Scope**

This problem is limited to software running in different CPUs using the same shared resource of the peripherals MCMCAN or E-Ray.

### **Recommended sequence (workaround)**

A read operation to the shared resource must be performed by the first CPU (CPU<sub>x</sub>) before execution of the DSYNC instruction.

Hence, the modified sequence of CPU<sub>x</sub> for the example given above must be:

- Resource update (store instruction)
- Resource read (load instruction)
- DSYNC
- Semaphore release

## **4.25 [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.

## **4.26 [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 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.

## **4 Application hints**

### **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.

## **4.27 [DAM\_TC.H002] Triggering DAM MEMCON.RMWERR and INTERR flags**

### **Description**

The MEMCON.INTERR error flag is linked to the MEMCON.RMWERR error flag and both flags are set for the same error condition.

This error condition can be triggered by inserting an uncorrectable ECC error into a RAM location and then making a write of 32 bits or less to the same RAM location.

**Note:** For devices with EMEM see also Application Hint EMEM\_TC.H006

## **4.28 [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  $2^{CBLS}$  or  $2^{CBLD}$  bytes (= 1, 2, 4, 8, 16, ... up to 64k bytes)"
- The maximum size specified in that sentence is incorrect.

### **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  $2^{CBLS}$  or  $2^{CBLD}$  bytes (= 1, 2, 4, 8, 16, ... up to 32 Kbytes)

## **4.29 [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.

## **4 Application hints**

### **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.

### **4.30 [EDSADC\_TC.H001] Auxiliary filter cleared with start of integration window - Additional information**

#### **Description**

**Note:** *The following information is an extension to the description included in section "Starting the Integration Window" in the EDSADC chapter of the TC3xx User's Manual.*

To support deterministic integration results in the case where the integration window is controlled by a hardware signal ( $DICFGx.ITRMODE = 01_B$  or  $10_B$ ), the filter chain can be cleared with the start of the integration window if bit  $IWCTRx.FRC = 0_B$ . This means, every non-recursive filter element of the filter chain (CIC3, FIR0, FIR1, integrator stage) is cleared to zero and the related decimation counters are loaded with their start values.

In this TC3xx design implementation, simultaneously also the CIC filter of the Auxiliary Filter is cleared to zero and the related decimation counter is set to its initial value.

Clearing of the described filter elements can be avoided if bit FRC is set to  $1_B$ , which means no filter element inside the filter chain nor the Auxiliary Filter is cleared.

**Note:** *There is no EDSADC configuration supported where the filter elements of the filter chain are cleared and the Auxiliary Filter keeps its value. For this particular configuration, the characteristic of the TC3xx EDSADC is not compatible with the TC2xx DSADC module of the AURIX™ product family.*

### **4.31 [EDSADC\_TC.H004] CIC3 filter properties - Documentation update**

#### **Description**

Table "CIC Filter Properties" in the EDSADC chapter of the AURIX™ TC3xx User's Manual contains obsolete information and is user irrelevant.

#### **Workaround**

Table "CIC Filter Properties" shall be ignored.

### **4.32 [EDSADC\_TC.H005] Incorrect configuration in the section "Stopping the Integration Window"**

#### **Description**

The EDSADC chapter in TC3xx user manual (V1.5.0 and newer versions) contains the following statement in the section "Stopping the Integration Window".

For this purpose, the result FIFO has to use one of the following configurations:

- $DICFGx.ITRMODE=10_B$ ,  $FCFGMx.SRGM=10_B$ ,  $RFCx.SRLVL=00_B$
- $DICFGx.ITRMODE=01_B$ ,  $FCFGMx.SRGM=10_B$ ,  $RFCx.SRLVL=00_B$

The setting  $FCFGMx.SRGM=10_B$  in the first configuration item above is incorrect.

## **4 Application hints**

### **Documentation update**

The part describing the configurations in the "Stopping the Integration Window" section in the EDSADC chapter shall be corrected as:

For this purpose, the result FIFO has to use one of the following configurations:

- DICFGx.ITRMODE=10<sub>B</sub>, FCFGx.SRGM=01<sub>B</sub>, RFCx.SRLVL=00<sub>B</sub>
- DICFGx.ITRMODE=01<sub>B</sub>, FCFGx.SRGM=10<sub>B</sub>, RFCx.SRLVL=00<sub>B</sub>

### **4.33 [EMEM\_TC.H006] Triggering EMEM MEMCON.RMWERR and INTERR flags**

#### **Description**

The following methods can be used to set the relevant bits in the MEMCON register of the EMEM module:

#### **Method 1**

The MEMCON.**RMWERR** bit is set when an EMEM memory address is written using a non-BTR4 access after the ECC of that word has been written via the MTU interface so that an uncorrectable ECC will occur.

#### **Method 2**

The MEMCON.**INTERR** bit is set when the SCTRL.GED bit in an EMEM module is written with a 1 and then an EMEM memory address in the same module is written using a non-BTR4 access.

**Note:** For devices with DAM see also Application Hint DAM\_TC.H002

### **4.34 [EMEM\_TC.H007] Access restrictions to EMEM for tools**

#### **Description**

Access to any EMEM tile configured in common-tool mode from the BBB can be blocked by continuous write accesses to the same tile from the SRI. The blocked BBB access results in an FPI Timeout condition that is logged as an error at the EBCU and/or IOClient.

#### **Details**

In common-tool mode, an EMEM tile may be accessed from both SRI and BBB interfaces, with arbitration logic always giving the SRI access higher priority.

Writes to EMEM must be carried out as an internal read-modify-write sequence because of the 32-byte word width of the SRAM tiles used (unless the write is a 32-byte item such as BTR4 on SRI). During this internal read-modify-write sequence, the tile arbitration logic of EMEM does not allow a BBB access to the same tile.

Therefore, a sequence of pipelined writes from SRI can keep the tile interface permanently busy, completely blocking access from BBB.

**Note:** Accesses to any two different tiles from the BBB and SRI can operate in parallel with complete independence, not causing the problem described above. Also, the problem does not occur for EMEM tiles used in common-application mode (ADAS application) and EMEM tiles used in trace mode (MCDS trace).



## 4 Application hints

### Recommendation

The problem can be avoided by modifying the SRI master behavior, to perform writes to EMEM using only BTR4 transactions. The 32-byte BTR4 write is carried out as a single write at the EMEM tile and no internal read-modify-write sequence is necessary. This allows sufficient idle cycles for any BBB access to take place.

The problem can also be avoided by modifying the SRI master behavior, to perform some alternative tile access along with the original writes to the affected tile. For example, a dummy read to any other tile in the same megabyte module of EMEM will allow cycles for a blocked BBB access to take place.

### 4.35 [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  $C_{IN}$ ) 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  $C_{IN}$  can be up to ~4x higher compared to the value listed in the Data Sheet.*

### 4.36 [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  $t_{PF} / t_{PFECC}$  for PFLASH
  - and  $t_{DF} / t_{DFECC}$  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)

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.*

### 4.37 [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.



## 4 Application hints

### 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.

### 4.38 [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
- 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.

### 4.39 [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.

### 4.40 [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 Application hints**

### **4.41 [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.

### **4.42 [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.*

### **4.43 [FlexRay\_AI.H010] Driver software must launch CLEAR\_RAM 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\_RAM command (via register SUCC1). CLEAR\_RAM fills a defined value into all memory locations. A safe initialization sequence of the E-Ray RAM blocks using the CLEAR\_RAM command is described in section "CLEAR\_RAM 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 Application hints

### 4.44 [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.

### 4.45 [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 Application hints**

### **4.46 [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.

### **4.47 [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.

#### **Recommendation**

Ignore all OTGB2 E-Ray triggers when data reported is only 0s.

### **4.48 [FPI\_TC.H002] Write access to register ACCEN1**

#### **Description**

The ACCEN1 (Access Enable Register 1) registers in the AURIX™ devices are reserved for future expansion. The bits in the ACCEN1 registers are described as "Reserved", read-only. There is no need for software to configure (write to) the ACCEN1 registers.

**Note:** For a write access to the ACCEN1 registers in the following modules, a bus error will be generated:

- **TC2xx:** MTU, SMU, ETH, I2C, FFT, CIF
- **TC3xx:** MTU, SMU (documented in TC3xx user manual), CIF

### **4.49 [FPI\_TC.H003] Burst write access may lead to data corruption**

#### **Description**

For the FPI slave modules listed below, if a write burst access is aborted on the last beat, this may lead to data corruption of all future accesses. No error is generated when the burst access is aborted.

This problem only affects the following modules:

- CONVCTRL, EVADC, PMS, SCR XRAM

## 4 Application hints

### Recommendation

Do not perform burst accesses to registers in CONVCTRL, EVADC, PMS, and to SCR XRAM.

### 4.50 [GETH\_AI.H001] Preparation for Software Reset

#### Description

**Note:** This application hint applies to MII and RMII. For RGMII see GETH\_TC.002.

When a kernel reset or software reset (via bit DMA\_MODE.SWR) shall be performed, the GETH module must be clocked (MII: RXCLK and TXCLK; RMII: REFCLK) and be in a defined state to avoid unpredictable behavior. Therefore, it is recommended to use the defined sequence listed below if frame transactions took place before setting bit SWR:

1. Finish running transfers and make sure that transmitters and receivers are set to stopped state:
  - a. Check the RPSx and TPSx status bit fields in register DMA\_DEBUG\_STATUS0/1
  - b. Check that MTL\_RXQ0\_DEBUG, MTL\_RXQi\_DEBUG, MTL\_TXQ0\_DEBUG and MTL\_TXQi\_DEBUG register content is equal to zero.  
**Note:** It may be required to wait  $70 f_{SPB}$  cycles after the last reset before checking if RXQSTS in MTL\_RXQ0\_DEBUG and MTL\_RXQi\_DEBUG are zero.
2. Wait until a currently running interrupt is finished and globally disable interrupts
3. Apply kernel reset to GETH module:
  - a. Deactivate Endinit protection, as registers KRST0/1 and KRSTCLR can only be written in Supervisor Mode and when Endinit protection is not active. Write to corresponding RST bits of KRST0/1 registers to request a kernel reset. The reset status flag KRST0.RSTSTAT may be cleared afterwards by writing to bit CLR in the KRSTCLR register. Re-activate Endinit protection
  - b. Wait  $70 f_{SPB}$  cycles, then check if RXQSTS in MTL\_RXQ0\_DEBUG and MTL\_RXQi\_DEBUG are zero
4. Configure the same mode as before (MII, RMII) in bit field GPCTL.EPR
5. Apply software reset by writing to the DMA\_MODE.SWR bit. Wait  $4 f_{SPB}$  cycles, then check if DMA\_MODE.SWR = 0<sub>B</sub>

If coming directly from Power-on Reset (i.e. no frame transaction took place yet), it is sufficient to follow the simplified sequence:

1. Configure the desired mode (MII, RMII) in bit field GPCTL.EPR
2. Apply software reset by writing to the DMA\_MODE.SWR bit. Wait  $4 f_{SPB}$  cycles, then check if DMA\_MODE.SWR = 0<sub>B</sub>

### 4.51 [GETH\_AI.H003] Undefined behavior when LD bit is set and buffer length B1L or B2L is zero - Additional information

#### Description

The description for bit LD (Last Descriptor) in table “TDES3 Normal Descriptor (Read Format)” in the GETH chapter of the User’s Manual contains the following sentence:

- “When this bit is set, it indicates that the buffer contains the last segment of the packet. When this bit is set, the B1L or B2L field should have a non-zero value.”

#### Additional information

If BL1 or BL2 (in TDES2) are zero when bit LD is set, Tx and Rx DMA operation may stop, resulting in undefined behavior of the GETH module.

## **4 Application hints**

### **Recommendation**

To ensure proper transmission of a packet and the next packet, you must specify a non-zero buffer size for the Transmit descriptor that has the Last Descriptor (TDES3[28]) set.

See also the note at the end of section “Transmit Packet Processing”.

## **4.52 [GETH\_AI.H004] MAC address 0 configuration sequence**

### **Description**

The MAC\_Address0 registers are double-synchronized to the (G)MII clock domains, and the synchronization is triggered only when bits [31:24] of the MAC\_Address0\_Low SFR are written. As a consequence, MAC\_Address0 SFRs are wrongly configured when a user writes MAC\_Address0\_Low SFR first and then the MAC\_Address0\_High SFR.

### **Effects**

Improper update of MAC\_Address0 SFRs leads to unintended receive filtering which causes the unintended loss of receive packets.

### **Documentation Update**

For the correct configuration of MAC\_Address0 SFRs, perform the following steps:

1. Write MAC\_Address0\_High SFR first and then
2. Write MAC\_Address0\_Low SFR

## **4.53 [GETH\_TC.H002] Stopping and Starting Transmission - Additional information**

### **Description**

Section “Stopping and Starting Transmission” in chapter “Programming Sequences” of the GETH chapter in the TC3xx User’s Manual shall be extended by additional information in steps 3, 4, and 5.

**Note:** *The following text is copied from the current version of the TC3xx User’s Manual, with the additional information added in steps 3a), 4a), and 5a).*

### **Stopping and Starting Transmission**

You can pause transmission by disabling the Transmit DMA, waiting for previous frame transmissions to complete, disabling the MAC transmitter and receiver, and disabling the Receive DMA.

#### **Notes:**

1. *Do not change the configuration (such as duplex mode, speed, port, or loop back) when the MAC is actively transmitting or receiving. These parameters are changed by software only when the MAC transmitter and receiver are not active*
2. *Similarly, do not change the DMA-related configuration when Transmit and Receive DMA are active*

Complete the following steps to pause the transmission for some time. The steps are provided for Channel 0.

### **Steps**

1. Disable the Transmit DMA (if applicable) by clearing Bit 0 (ST) of DMA\_CH0 Register
2. Wait for any previous frame transmissions to complete. You can check this by reading the appropriate bits of MTL\_TxQ0\_Debug Register (TRCSTS is not 01 and TXQSTS=0)

#### 4 Application hints

3. Disable the MAC transmitter and MAC receiver by clearing Bit 0 (RE) and Bit 1 (TE) of the MAC\_Configuration Register
  - a. The receiver will be stopped after the register got written (verified by reading back the register) in approximately  $3 \cdot f_{GETH} + 6 \cdot RXCLK$  cycles if no future packet is in receive state (see description of bit RE)
4. Disable the Receive DMA (if applicable), after making sure that the data in the Rx FIFO is transferred to the system memory (by reading the appropriate bits of MTL\_RxQ0\_Debug Register, PRXQ=0 and RXQSTS=00)
  - a. In case received data of a queue is not intended to be processed anymore set bit DMA\_CHI\_RX\_CONTROL.RPF of each DMA channel moving out packets from this queue to flush all packets inside the queue after stopping the receive process
5. Make sure that both Tx Queue and Rx Queue are empty (TXQSTS is 0 in MTL\_TxQ0\_Debug Register and RXQSTS is 0 in MTL\_RxQ0\_Debug Register)
  - a. In case a late packet arrived, either forward to the system memory by re-enabling the RX DMAs (see step 6) or flush data out of the queue (see step 4.a)
6. To restart the operation, first start the DMAs, and then enable the MAC Transmitter and Receiver

#### 4.54 [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>.

#### 4.55 [GTM\_AI.H425] MCS: Instructions BRDI and BWRI evaluate unused address bits

##### Description

Bus master instructions with indirect addressing (BRDI and BWRI) use the bits 2 to 15 of register B for defining their target address. However, if the bit slice B[1:0] is unequal to 0, the current implementation of the GTM-IP behaves as follows:

bit-field CCM[i]\_AEIM\_STA.AEIM\_XPT\_STA is updated with value 1 and CCM[i]\_AEIM\_STA.AEIM\_XPT\_ADDR is updated with the address of the selected register B. Further, if bit-field MCS[i]\_CTRL\_STAT.HLT\_AEIM\_ERR is set, then the associated MCS channel stops and indicates an unexpected bus master error.

##### Scope

MCS



## 4 Application hints

### Effects

The word alignment for the address of indirect bus master instructions is checked unexpectedly leading to a false error indication.

BRD and BRDI accesses of any MCS channel in the affected MCS instance that are following the non-wordaligned access might receive no data or wrong data. The received data for a BRD or BRDI instruction might also be stored in another register of the same MCS channel. Further, the MCS program execution of any channel might be stopped.

### Workaround

The use of the lower address bits is forbidden for indirect addressing. Hence this error needs to be avoided by the application.

## 4.56 [GTM\_AI.H473] SPEC-FIFO: Wrong description of FIFO flush operation

### Description

**Note:** Register names in the text follow the TC3xx syntax conventions. Correlation of register names:

- **TC2xx:** FIFO*i*\_CH*x*\_
- **TC3xx:** FIFO[i]\_CH[x]\_

FLUSH bit-field description of register FIFO[i]\_CH[x]\_CTRL (GTM4.1 spec.: FIFO\_585):

The specification describes that the FIFO[i]\_CH[x]\_FILL\_LEVEL.LEVEL, the FIFO[i]\_CH[x]\_RD\_PTR.ADDR, and FIFO[i]\_CH[x]\_WR\_PTR.ADDR will be reset to their initial values.

This is valid for FIFO[i]\_CH[x]\_FILL\_LEVEL.LEVEL but not for FIFO[i]\_CH[x]\_RD\_PTR.ADDR and FIFO[i]\_CH[x]\_WR\_PTR.ADDR, which are set to the value of FIFO[i]\_CH[x]\_START\_ADDR.ADDR on a FIFO flush operation.

Also it should be mentioned in the specification that the status bits EMPTY, FULL, LOW\_WM and UP\_WM of register FIFO[i]\_CH[x]\_STATUS are set to EMPTY=1, FULL=0, LOW\_WM and UP\_WM depending on the values programmed into FIFO[i]\_CH[x]\_LOWER\_WM.ADDR and FIFO[i]\_CH[x]\_UPPER\_WM.ADDR.

UP\_WM bit-field description of register FIFO[i]\_CH[x]\_STATUS (GTM4.1 spec.: FIFO\_628):

The condition for the UP\_WM bit-field of the register FIFO[i]\_CH[x]\_STATUS is not correct in case the register for FIFO[i]\_CH[x]\_UPPER\_WM.ADDR is programmed to 0 and afterward a FIFO flush is requested. In this case the bit UP\_WM will signal 0 in RTL, but the evaluation due to the specification expects a 1.

To overcome this inconsistency between RTL and the specification, the value 0 for FIFO[i]\_CH[x]\_UPPER\_WM.ADDR has to be excluded in the specification (see Note in the ADDR bit-field description of register FIFO[i]\_CH[x]\_UPPER\_WM, GTM4.1 spec.: FIFO\_609), as this value does not make sense from an application point of view.

Prose text in the Overview chapter of FIFO (GTM4.1 spec.: FIFO\_836):

It is mentioned that the read and write pointer and also the fill level of the corresponding FIFO channel will be reset.

Here the word reset is used in context with the flush. Typically reset is combined with setting the initial values, but this is not true here. Instead of "reset" another term should be used, for example: "set to previously configured values".

Following note is missing in the specification:

A FIFO flush operation does not influence the state of the FIFO[i]\_CH[x]\_IRQ\_NOTIFY register.

### Scope

FIFO



## 4 Application hints

### Effects

1. The values of FIFO[i]\_CH[x]\_RD\_PTR.ADDR and FIFO[i]\_CH[x]\_WR\_PTR.ADDR are not set to the initial value as described in the specification
2. False value of FIFO[i]\_CH[x]\_STATUS.UP\_WM after a flush request in case FIFO[i]\_CH[x]\_UPPER\_WM.ADDR is programmed to 0

### Recommendation

Please apply either 1 or 2:

1. Configure FIFO[i]\_CH[x]\_START\_ADDR.ADDR to its initial value before executing the flush operation
2. Do not configure value 0 for FIFO[i]\_CH[x]\_UPPER\_WM.ADDR

## 4.57 [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.

**Table 20 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.

## 4 Application hints

### Recommendation

Consider the information in the corresponding table above.

## 4.58 [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  
 "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 21 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 Application hints**

### **4.59 [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.

### **4.60 [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

### **4.61 [GTM\_AI.H502] SPEC-DPLL input selection for SUB\_INC1 is incomplete**

#### **Description**

The bit field DPLL\_CTRL\_0\_SHADOW\_TRIGGER.RMO is updated on active TRIGGER input, whereas DPLL\_CTRL\_0\_SHADOW\_STATE.RMO is updated on active STATE input. The active TRIGGER input and active STATE input might arrive at different times. Therefore, on a switch from normal mode to emergency mode (or

#### **4 Application hints**

the other way around), the two bit fields might be updated at different points in time leading to them having different values.

The SPEC deficiency:

The coding for DPLL\_CTRL\_0\_SHADOW\_TRIGGER.RMO (GTM4.1 spec.: DPLL\_16133) and for bit field DPLL\_CTRL\_0\_SHADOW\_STATE.RMO (GTM4.1 spec.: DPLL\_16132) does not take into account that the two bit fields may have different values. As a consequence it is ambiguous which input contributes to the calculation of INC\_CNT1.

In order to fix the specification:

Add a note to bit field DPLL\_CTRL\_0\_SHADOW\_TRIGGER.RMO and also to bit field DPLL\_CTRL\_0\_SHADOW\_STATE.RMO:

"The coding is applicable if DPLL\_CTRL\_0\_SHADOW\_TRIGGER.RMO = DPLL\_CTRL\_0\_SHADOW\_STATE.RMO. If not (they are unequal) the DPLL\_CTRL\_0.RMO defines which input is evaluated: STATE is selected if DPLL\_CTRL\_0.RMO=1 else TRIGGER is selected".

#### **Scope**

DPLL

#### **Effects**

DPLL behaves in an unspecified manner if DPLL\_CTRL\_0\_SHADOW\_TRIGGER.RMO ≠ DPLL\_CTRL\_0\_SHADOW\_STATE.RMO.

#### **Recommendation**

None.

### **4.62 [GTM\_AI.H515] SPEC-MCS: Incomplete usage of term CPU in MCS chapter**

#### **Description**

The MCS Spec describes in several parts that a register or a bit-field can only be accessed by CPU although it can be accessed by CPU and MCS AEI bus master interface. This erratum applies to the following sections in the specifications:

- The register MCS[i]\_CH[x]\_IRQ\_NOTIFY in the section "MCS Configuration Register Description" of the GTM chapter in the user manual

#### **Scope**

MCS

#### **Effects**

The description in the specification erroneously limits the accessibility of registers and bit-fields to CPU only.

#### **Workaround**

No workaround available.

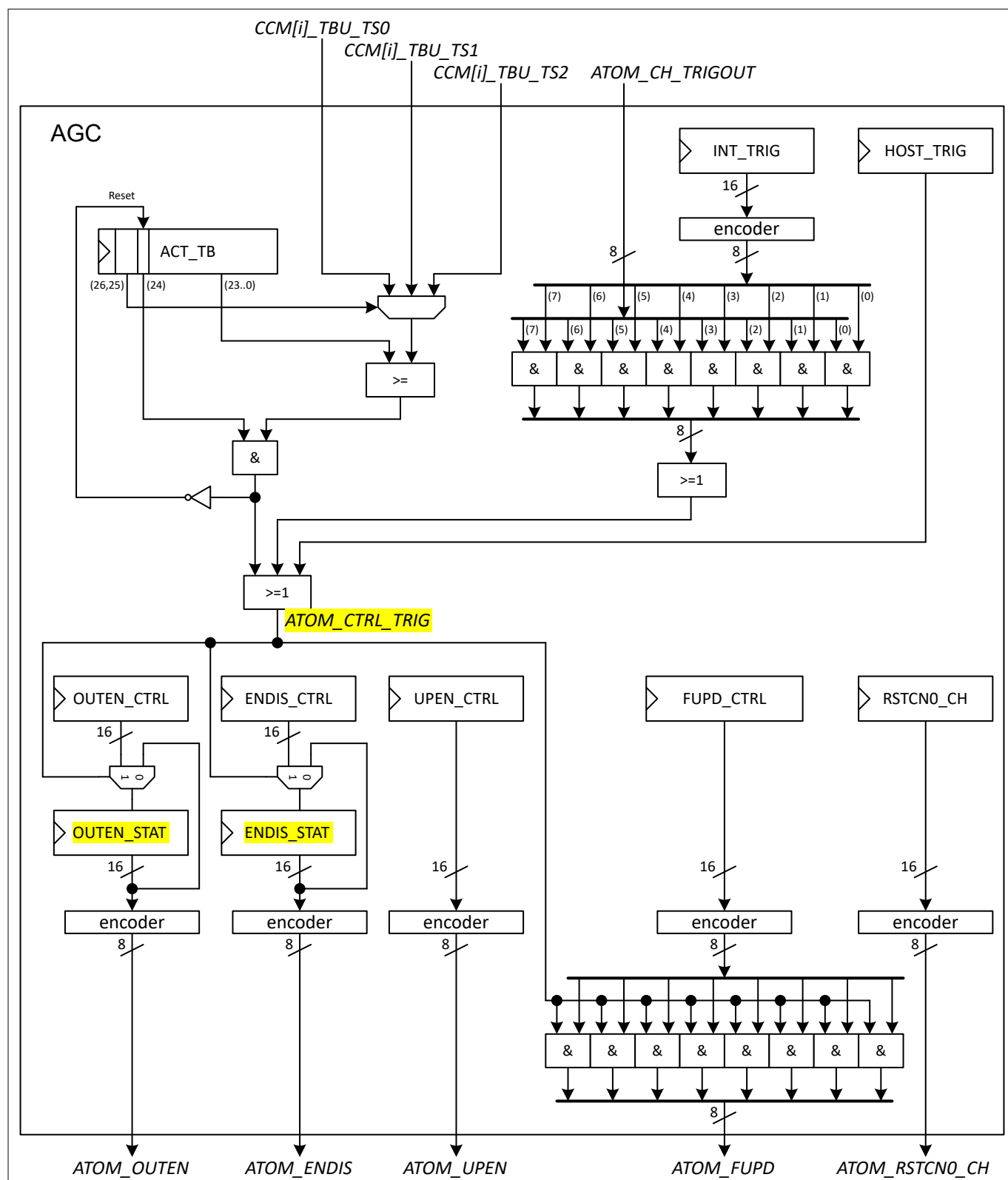
### **4.63 [GTM\_AI.H803] Missing feature aspect in the user manual**

#### **Description**

The following behavior is part of the ATOM and also TOM. In the clock cycle in which ATOM\_CTRL\_TRIG (see yellow highlight in the attached figure) is active, a simultaneous bus write access to register

## 4 Application hints

ATOM[i]\_AGC\_ENDIS\_STAT or also ATOM[i]\_AGC\_OUTEN\_STAT is ignored, regardless of the update value from the associated registers ATOM[i]\_AGC\_ENDIS\_CTRL/ATOM[i]\_AGC\_OUTEN\_CTRL.



**Figure 11** ATOM Global channel control mechanism

## Scope

TOM, ATOM

## 4 Application hints

### Effects

The write access to ATOM[i]\_AGC\_ENDIS\_STAT (respectively ATOM[i]\_AGC\_OUTEN\_STAT) is ignored for the clock cycle, where ATOM\_CTRL\_TRIG is active, even if ATOM[i]\_AGC\_ENDIS\_CTRL=0 (respectively ATOM[i]\_AGC\_OUTEN\_CTRL=0).

### Workaround

No workaround available.

## 4.64 [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.

## 4.65 [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

**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.*

## 4.66 [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>).

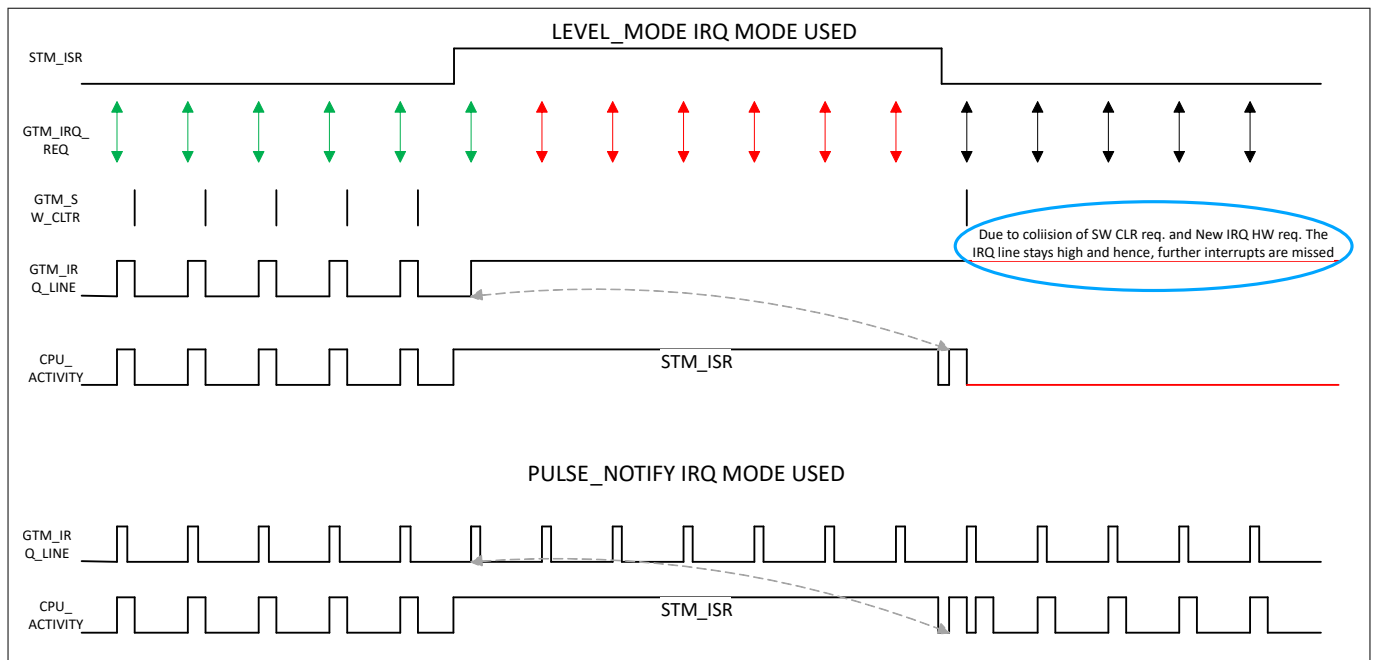
Figure “GTM interrupts” in chapter “GTM Implementation” of the TC3xx User’s Manual shows how the interrupt signal (GTM\_IRQ\_XXX) triggers an interrupt towards the Interrupt Router (IR), depending on IRQ\_MODE.

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

## 4 Application hints

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 12** 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.

## 4.67 [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.

## 4.68 [GTM\_TC.H034] Correction to figure GTM to MSC Connections, 1st Level Muxes Overview

### Description

In the registers MSCSETiCONj, bit-fields SELk define the GTM timer source configured as Set i signal k out.

## 4 Application hints

In the appendix of the TC3xx user manual (V2.0.0 and earlier versions), the figure “GTM to MSC Connections, 1st Level Muxes Overview” shows the correlation between the registers MSCSETiCONj, bit-fields SELk and signals SETi[k]. In this figure, the values for index j (j=0-3) and for index k (k=0-3) shown for the registers MSCSET0CONj, MSCSET1CONj and MSCSET8CONj are incorrect.

### Documentation Update

For the correct definition, see the description of the registers MSCSETiCONj in the TC3xx user manual.

In general, the bit-fields SELk in the registers MSCSETiCONj control the signals SETi[k]. Therefore, in the appendix of the TC3xx user manual (V2.0.0 and earlier versions), the figure “GTM to MSC Connections, 1st Level Muxes Overview” must be corrected as follows:

- MSCSET0CON0.SEL3..SEL0 must be corrected as MSCSET0CON3.SEL15..SEL12
- MSCSET0CON1.SEL3..SEL0 must be corrected as MSCSET0CON2.SEL11..SEL8
- MSCSET0CON2.SEL3..SEL0 must be corrected as MSCSET0CON1.SEL7..SEL4
- MSCSET0CON3.SEL3..SEL0 must be corrected as MSCSET0CON0.SEL3..SEL0
- MSCSET1CON1.SEL3..SEL0 must be corrected as MSCSET1CON1.SEL7..SEL4
- MSCSET1CON2.SEL3..SEL0 must be corrected as MSCSET1CON2.SEL11..SEL8
- MSCSET1CON3.SEL3..SEL0 must be corrected as MSCSET1CON3.SEL15..SEL12

In the appendix of the TC39x user manual (V2.0.0 and earlier versions), the figure “GTM to MSC Connections, 1st Level Muxes Overview” must be corrected as follows:

- MSCSET8CON1.SEL3..SEL0 must be corrected as MSCSET8CON1.SEL7..SEL4
- MSCSET8CON2.SEL3..SEL0 must be corrected as MSCSET8CON2.SEL11..SEL8
- MSCSET8CON3.SEL3..SEL0 must be corrected as MSCSET8CON3.SEL15..SEL12

In the appendix of the TC38x, TC3Ex user manual (V2.0.0 and earlier versions), the figure “GTM to MSC Connections, 1st Level Muxes Overview” must be corrected as follows:

- MSCSET6CON1.SEL3..SEL0 must be corrected as MSCSET6CON1.SEL7..SEL4
- MSCSET6CON2.SEL3..SEL0 must be corrected as MSCSET6CON2.SEL11..SEL8
- MSCSET6CON3.SEL3..SEL0 must be corrected as MSCSET6CON3.SEL15..SEL12

In the appendix of the TC37x, TC37xEXT, TC36x user manual (V2.0.0 and earlier versions), the figure “GTM to MSC Connections, 1st Level Muxes Overview” must be corrected as follows:

- MSCSET3CON1.SEL3..SEL0 must be corrected as MSCSET3CON1.SEL7..SEL4
- MSCSET3CON2.SEL3..SEL0 must be corrected as MSCSET3CON2.SEL11..SEL8
- MSCSET3CON3.SEL3..SEL0 must be corrected as MSCSET3CON3.SEL15..SEL12

## 4.69 [HSCT\_TC.H009] High speed dividers five phase clock sequence ordering

### Description

For correct operation of the phase correlator and avoiding degradation of BER during operation, the five phase clock generated by high speed dividers must remain in correct sequence.

To prevent the sequence of the five phase clock from being disordered, certain requirements have to be fulfilled when powering on/off of the HSCT physical layer.

### Recommendation

Powering on:

- Make sure the Peripheral PLL clock is already locked and a correct clock is provided before the HSCT physical layer is powered on by setting bit CONFIGPHY.PON to 1<sub>B</sub>



## 4 Application hints

Powering off:

- Note that, according to section “Disable Request (Power-Off)” in the HSCT chapter of the TC3xx User’s Manual, high speed dividers need to be disabled (INIT.RXHD = 000<sub>B</sub> and INIT.TXHD = 000<sub>B</sub>) before the physical layer is powered off by setting CONFIGPHY.PON to 0<sub>B</sub>

### 4.70 [HSCT\_TC.H010] Interface control command timing on the LVDS ports

#### Description

As described in section “Interface Control” of the HSCT chapter in the user manual, a HSCT master device is sending interface control commands to a slave device by setting the command in register IFCTRL.IFCVS and triggering IFCTRL.SIFCV.

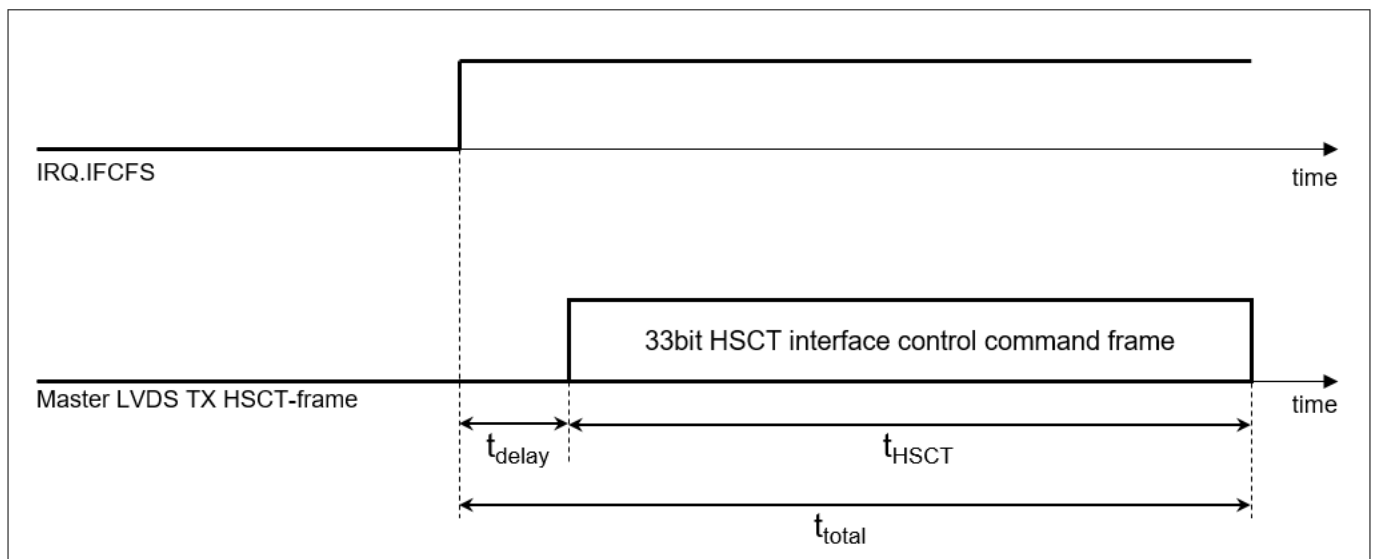
Once triggered, the interface command is scheduled for take over into the transmission FIFO for sending. Only when the interface command has been taken over into the FIFO, sending of the next interface command must be triggered by software. Therefore, software must monitor the takeover by a transition of IRQ.IFCFS from 0 to 1.

As flag IRQ.IFCFS only indicates

- takeover of the interface command into the FIFO
- readiness for the next interface command to be triggered

the user might falsely assume that also the actual sending on the LVDS TX port has already occurred once the IRQ.IFCFS flag is set, which is not true.

Instead the timing shown in the following figure applies.



**Figure 13 Timing of LVDS TX interface command sending in relation to IRQ.IFCFS**

#### Recommendation

Before changing interface configurations, software must guarantee not having transfers active on the interface. Therefore the time of  $t_{\text{total}}$  has to be taken into account:

- $t_{\text{total}} = t_{\text{delay}} + t_{\text{HSCT}}$

While  $t_{\text{HSCT}}$  of the HSCT control command frame is determined and can be calculated from the actual baud rate, the additional time  $t_{\text{delay}}$  has to be taken into account with:

- $t_{\text{delay}} \geq 10 \mu\text{s}$

This value is valid for  $f_{\text{SRI}} \geq 100 \text{ MHz}$ ,  $f_{\text{SPB}} \geq 100 \text{ MHz}$  and baud rate  $\geq 5 \text{ MBaud}$ .

## **4 Application hints**

### **4.71 [I2C\_TC.H008] Handling of RX FIFO Overflow in Slave Mode**

#### **Description**

If the I2C kernel has detected a RX FIFO overflow in slave mode, a RX\_OFL\_srq request is generated, the incoming character is discarded, and the kernel puts a not-acknowledge on the bus and changes to listening state.

However, it does not generate an EORXP\_ind signal, so that the remaining characters in the FIFO can not be moved out by means of data transfer requests.

#### **Recommendation**

Upon an RX FIFO overflow in slave mode, received data may be invalid. However, they may be read from the FIFO for example for analysis if required.

In order to flush the FIFO and correctly resume communication

- set bit RUNCTRL.RUN = 0<sub>B</sub> (switch to configuration mode)
- set bit RUNCTRL.RUN = 1<sub>B</sub> (participate in I2C communication)

### **4.72 [I2C\_TC.H009] Connections of Serial Clock Inputs**

#### **Description**

In tables “Connections of I2C0” and “Connections of I2C1” (for TC39x, TC38x TC3Ex only) in the respective product specific appendix to the TC3xx User’s Manual, only the serial clock output connections are shown. The serial clock input connections (located on the same port pins) are not shown.

#### **Recommendation**

See the corresponding TC3xy Data Sheet for the available I2C serial clock input connections (I2C0:SCLA/SCLB/SCLC, I2C1:SCLA/SCLB) on the respective port pins.

### **4.73 [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 22](#) 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\_SCUERU<sub>x</sub> (x=0-3)) are implemented to handle external interrupts.”

Table “OGU to SRC connection” in the SCU chapter shall be changed as follows:

## 4 Application hints

**Table 22 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.

### 4.74 [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).

### 4.75 [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 (BODY=1<sub>B</sub>) and a weak pull-up (BODY=0<sub>B</sub>).

**Note:** P33.8 will be pulled up if BODY=0<sub>B</sub> during LBIST execution, which is different from the behavior during cold reset where P33.8 is always in tristate.

#### 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.

### 4.76 [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.

#### 4 Application hints

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

### 4.77 [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  $I_{DDMBIST}$ ).

### 4.78 [MCDS\_TC.H007] Program trace of CPUx ( $x > 0$ ) program start not correct

#### Description

**Note:** *This problem is only relevant for development tools.*

All CPUs - except CPU0 - need to be started by user software from another CPU. This user software writes the PC and starts the CPU.

If this phase is traced with MCDS, the program trace for the first two executed instructions is not correct. The start PC is not visible and depending on the trace mode, the address of the second instruction is shown twice in the trace and there can be a wrong IPI message. However these effects are limited to the first two instructions. Nevertheless this is confusing for the user and it can be an issue for trace based code coverage tools.

#### Recommendations

- A trace tool can ignore the generated trace messages for the first two instructions and replace it with proper messages
- A trace tool can notify the user that the initial trace sequence for the first two instructions at startup is not correct

## 4 Application hints

### 4.79 [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.  
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.

### 4.80 [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 CCCRi.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.

## **4 Application hints**

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.

### **4.81 [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.

### **4.82 [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.

### **4.83 [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.

## 4 Application hints

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, NTBTTi, NTCTTRi

set bit-field RELOAD to one value less than the calculated counter value.

## 4.84 [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,

- $f_{\text{MCANH}}$  defines the frequency for the internal clocking of the MCMCAN module
- $f_{\text{MCAN}}$  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 “ $f_{\text{CAN}}$ ” to “ $f_{\text{MCAN}}$ ” as shown below:

**Table 23 Operating Conditions - CAN Frequency: symbol update**

Parameter	Symbol	Values			Unit	Note / Test Condition
		Min.	Typ.	Max.		
CAN Frequency	$f_{\text{MCAN}}$ SR	-	-	80	MHz	

## 4.85 [MSC\_TC.H014] Symbol $T_A$ in specification of FCLPx clock period in Data Sheet - Additional information

### Description

The tables in chapter “MSC Timing” in the Data Sheet use symbol “ $T_A$ ” in the specification of parameter “FCLPx clock period”.

In this context,  $T_A = 1/f_A$ , where  $f_A$  is the input clock frequency of the MSC ABRA block (see also the MSC chapter in the User’s Manual).

## 4.86 [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.



## **4 Application hints**

### **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.

### **4.87 [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.

#### **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].

### **4.88 [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.

### **4.89 [NVM\_TC.H001] References to DMU\_HP\_PROCONTTP – Typo in TC3xx user manual**

#### **Description**

The register name DMU\_HP\_PROCONTTP mentioned in the TC3xx user manual must be replaced with DMU\_HF\_PROCONTTP.



## 4 Application hints

### 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)
- "Handling Errors During Operation" (DMU chapter)
- "UCB\_OTPy\_ORIG and UCB\_OTPy\_COPY (y = 0 - 7)" (UCB chapter)

### 4.90 [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.

### 4.91 [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:

#### **4 Application hints**

- 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

#### **4.92 [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.

#### **4.93 [OCDS\_TC.H017] CIF not reset by application reset if OCDS is enabled**

##### **Description**

**Note:** *This problem only has an effect during debugging and if the CIF module is used.*

If OCDS is enabled, the CIF module will not be reset (initialised) on an application reset.

##### **Recommendation**

During debugging use power-on reset or disable OCDS before triggering the application reset to ensure the reset of the CIF module.

#### **4.94 [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.

## 4 Application hints

Alternatively, a power-on reset may be performed instead of a system or application reset.

### 4.95 [PACKAGE\_TC.H005] Exposed pad dimensions and package outlines for QFP packages - Updates to TC37xEXT Data Sheet

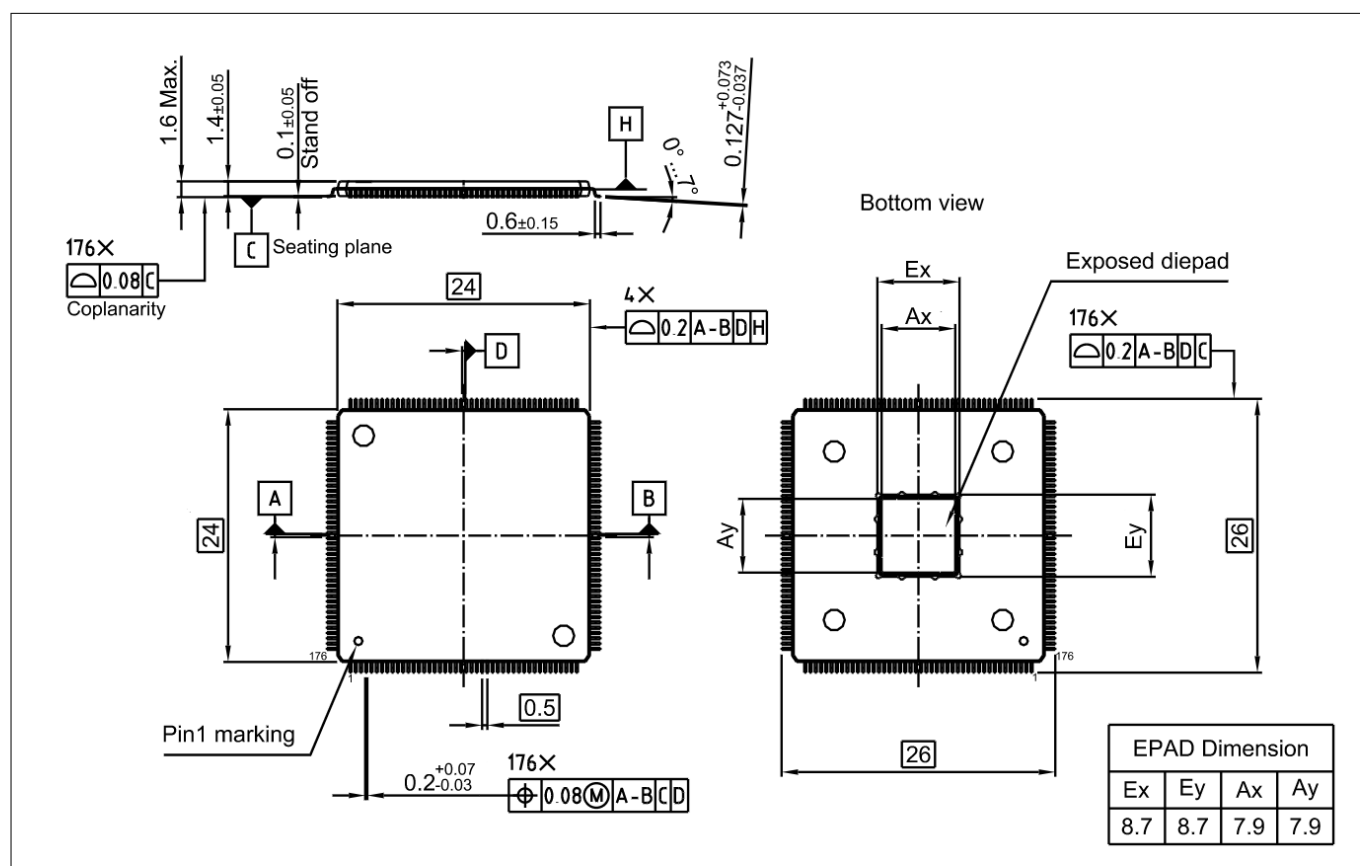
#### Description

In the scope of the harmonization of the package drawings, the drawings for the LQFP packages of the TC37xEXT 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.

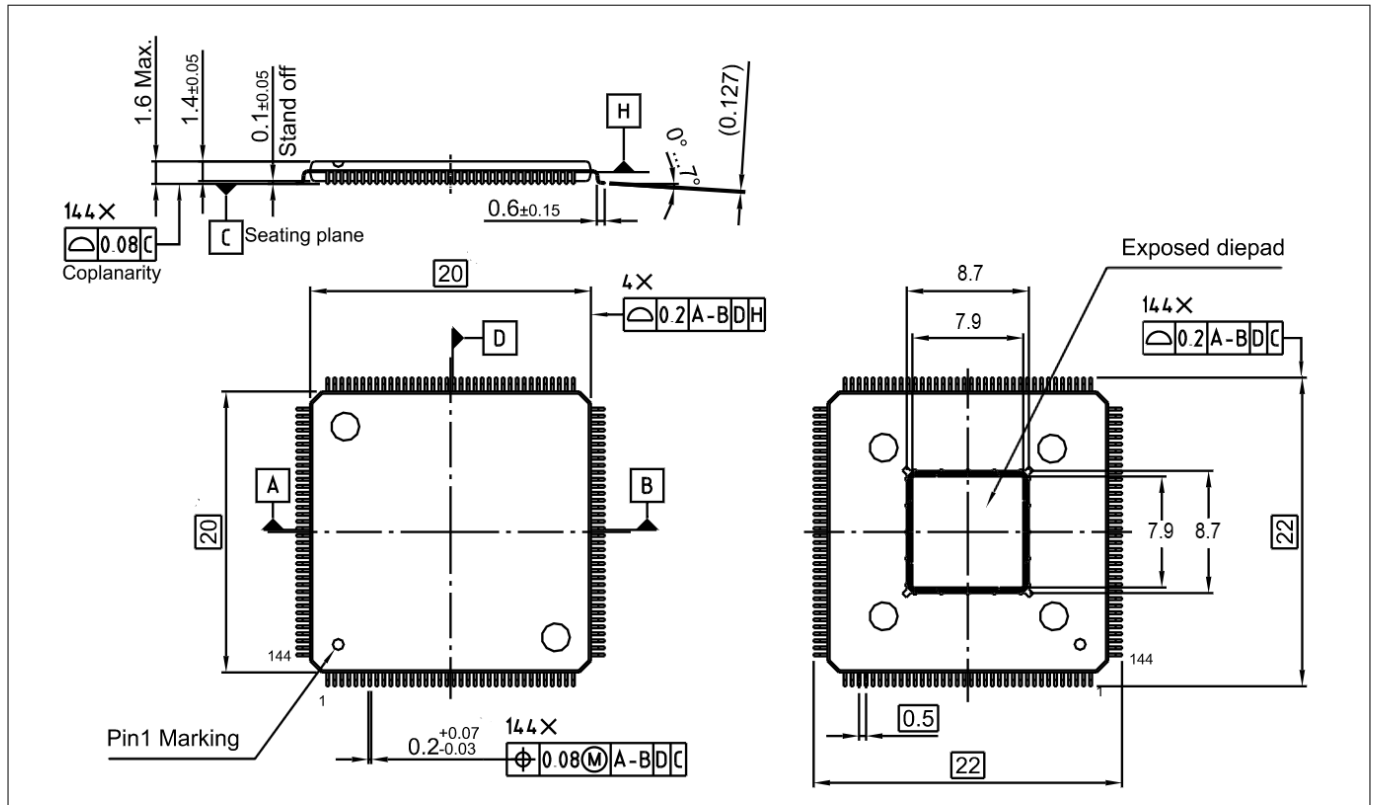
This information shall substitute the corresponding information in the TC37xEXT AB-step Data Sheet V1.1.



**Figure 14** Package Outlines LQFP-176 for TC37xEXT

**Note:** For the exposed diepad of the LQFP-176 package of the TC37xEXT, structural corner protrusions have to be considered for purposes of system design and integration with a maximum boundary of 9.4 mm.

#### 4 Application hints



**Figure 15** Package Outlines LQFP-144 for TC37xEXT

**Note:** For the exposed diepad of the LQFP-144 package of the TC37xEXT, structural corner protrusions have to be considered for purposes of system design and integration with a maximum boundary of 9.4 mm.

You can find all of our packages, sorts of packing and others on our Infineon Internet Page <http://www.infineon.com/products>

### 4.96 [PADS\_TC.H008] Overload coupling for LVDS RX pads – Additional information

#### Description

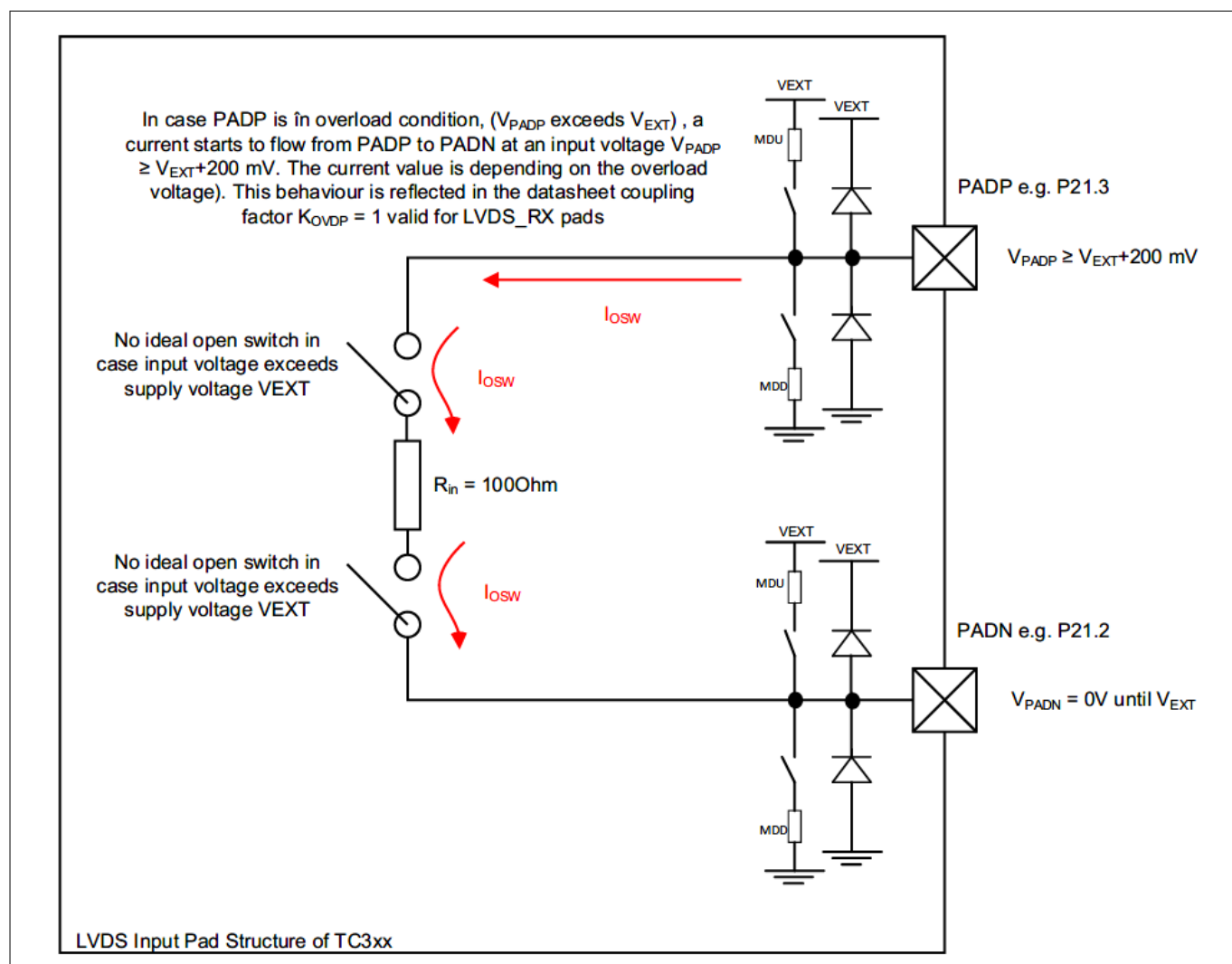
Due to the internal circuit structure, for pads with buffer type “LVDS\_RX” (see column “buffer type” in port function tables in the Data Sheet), the specified overload coupling factors  $K_{OVDP}$  and  $K_{OVDN}$  to the neighbor LVDS pad are relatively large (see table “Overload Parameters” in the Data Sheet).

**Note:** In this context, the term “neighbor LVDS pad” of a P-pad (N-pad) refers to the corresponding N-pad (P-pad) within the same P-/N-pad pair. If available on the package and implemented as buffer type LVDS\_RX, these are P14.9/10, P21.0/1, P21.2/3.

The following figure illustrates the effect for a positive overload condition ( $V_{PADP} > V_{EXT}$ ) on the P-pad (PADP) of a pin pair where both the P-pad and the corresponding N-pad (NPAD, with  $0V \leq V_{NPAD} \leq V_{EXT}$ ) are used as digital CMOS inputs.

The overload current  $I_{OSW}$  through the non-ideal open switch significantly increases when  $V_{PADP} \geq V_{EXT} + 200$  mV.

#### 4 Application hints



**Figure 16** Positive overload on P-pad of a LVDS RX pin pair (both used as inputs)

**Note:** In general, the effect will also occur if the N-pad is used as output, or for a negative overload condition on either the P-pad or the N-pad.

#### 4.97 [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 PREUVVAL 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.PREUVVAL fields in an application.

## 4 Application hints

### 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.PREUVVAL:
  - The register reset value 0xBC = 1.079 V (typical) may be kept. It matches the LVD reset level ( $V_{LVDRSTC}$ ) 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
- PREUVVAL range = 1.1 V - 1.15 V
  - Register reset value: SMU alarm generated at PREUVVAL ~ 1.08 V

## 4.98 [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.

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 WDTCPUsR.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

## 4 Application hints

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.

## 4.99 [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.



#### **4 Application hints**

Therefore standby mode entry is achieved even in the reset state of the AURIX™ system modes.

### **4.100 [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

### **4.101 [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.

### **4.102 [PMS\_TC.H019] Limitation of power-cycles - Additional datasheet footnote**

#### **Description**

In the table "Supply Ramp" in the sub-chapter EVR of the TC39x datasheet, a statement is described as:

- "Up to 1000000 power-cycles, matching the limits defined in the table "Supply Ramp", are allowed for TC39x, 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. This AppHint is applicable to all TC3xx corresponding device datasheet.



## **4 Application hints**

### **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 TC39x, 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

### **4.103 [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.

#### **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.

### **4.104 [PSI5\_TC.H001] No communication error in case of payload length mismatch**

#### **Description**

When the payload of a frame is higher than the set payload size PDL<sub>xy</sub> for channel x and slot y, then neither the CRC error nor any other error flag is reliably set in all cases.

When less data is received than the set payload size PDL<sub>xy</sub>, there are error flags (NBI) that can handle this scenario.

#### **Recommendation**

The payload data received should match the configured payload size PDL<sub>xy</sub> for channel x and slot y (register/field RCRA<sub>x</sub>.PDL<sub>y</sub>).

### **4.105 [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.

## 4 Application hints

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.

## 4.106 [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.

**Table 24 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.

(table continues...)

#### 4 Application hints

**Table 24** (continued) 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
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.

**Table 25** 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.

**(table continues...)**

#### 4 Application hints

**Table 25** (continued) 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
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

#### 4.107 [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).

#### 4.108 [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 Application hints**

### **4.109 [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”.

#### **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>).

#### **4 Application hints**

##### **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\_TILEECT.

##### **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**

The application SW shall enable address buffer recording (CPUx\_FLASHCON2.RECDIS = 10<sub>B</sub>).  
Enabled after reset.

##### **SM[HW]:NVM.PFLASH:FLASHCON\_MONITOR**

###### **Init conditions**

The application SW shall initialize CPUx\_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.

## **4 Application hints**

### **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

## **4.110 [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.

**Note:** SM[HW]:NVM.FSIRAM:REG\_MONITOR\_TEST is not part of the FMEDA and will not impact the FIT rate.

## **4.111 [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 Application hints**

### **4.112 [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).

### **4.113 [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)

### **4.114 [SCR\_TC.H011] Hang-up when warm PORST is activated during Debug Monitor Mode**

#### **Description**

**Note:** *This problem is only relevant for debugging.*

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.

### **4.115 [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.



## **4 Application hints**

### **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>

### **4.116 [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).

### **4.117 [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.

#### **Documentation update**

For the use case with  $f_{SYS\_SCR} = 70$  kHz,  $T_J = 25^\circ\text{C}$  and SCR CPU in IDLE mode, we estimate  $I_{SCRIDLE\_70}$  CC of 90  $\mu\text{A}$ .

### **4.118 [SCU\_TC.H019] Connection on ERU input E\_REQ7(5)**

#### **Description**

Table “Connections of SCU” in the TC37x and TC37xEXT specific Appendix to the AURIX™ TC3xx User’s Manual V2.0.0 and earlier versions includes the following connection

- SCU:E\_REQ7(5) from GTM:SCU\_TRIG(3)

As TOM3 is not present in the GTM implementation in TC37x and TC37xEXT, this connection is not functional in the TC37x and TC37xEXT and therefore should not be used.

## 4 Application hints

### 4.119 [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  $\overline{\text{ESR}x}$  is enabled via bit PMSWCR0.ESRxDFEN = 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.

### 4.120 [SCU\_TC.H021] LBIST execution affected by TCK/DAP0 state

#### Description

The TCK/DAP0 pad includes an internal pull down (marked "PD2" in column "Buffer Type" in table "System I/O of the Data Sheet).

If TCK/DAP0 is pulled up by an external device, LBIST execution will be stalled.

#### Recommendation

TCK/DAP0 pad shall be left open or pulled down if no tool is connected.

### 4.121 [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.

### 4.122 [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 26** Field EEA in register CHIPID

Field	Bits	Type	Description
EEA	16	rh	<b>Emulation or ADAS Extension Available</b> Indicates if the emulation or ADAS extension hardware is available or not. 0 <sub>B</sub> EEC is not available (SAK-TC3xxxU or SAK-TC3xxxP) 1 <sub>B</sub> EEC is available (SAK-TC3xxxE or SAK-TC3xxxF)

## 4 Application hints

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<sub>B</sub> in TC39x, TC37xEXT, TC35x and TC33xEXT devices, and bit EEA = 0<sub>B</sub> 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.

## 4.123 [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.

## 4.124 [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<sub>B</sub> to 111<sub>B</sub> and INP1 = 100<sub>B</sub> to 111<sub>B</sub> 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<sub>B</sub> to 111<sub>B</sub> and for INP1 and settings INP1 = 100<sub>B</sub> to 111<sub>B</sub> is shown in the following table.

**Table 27 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> )

(table continues...)

## 4 Application hints

**Table 27 (continued) Field INP0 and INP1 in register EICRi (i=0-3) - Correction**

Field	Bits	Type	Description
			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> )
<b>INP1</b>	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.

### 4.125 [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

### 4.126 [SDMMC\_TC.H001] Idle State of SDMMC0\_CLK

#### Description

The idle level of the card clock output SDMMC0\_CLK is high while the card clock is not enabled (bit CLK\_CTRL.SD\_CLK\_EN = 0<sub>B</sub>).

## 4 Application hints

### 4.127 [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 28 SENT input pads and considered neighbors**

Device	Considered left neighbors		SENT input		Considered right neighbors	
			Pad	Channel		
TC39x	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
TC38x	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
TC3Ex	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 28 (continued) SENT input pads and considered neighbors**

Device	Considered left neighbors		SENT input		Considered right neighbors	
			Pad	Channel		
<b>TC37x and TC37xEXT</b>	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
<b>TC36x</b>	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
<b>TC33xEXT</b>	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
<b>TC33x/TC32x</b>	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 29 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 29 (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 30 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 30 (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 30 (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 31 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 32 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 32 (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 33 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>

#### 4.128 [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]

#### 4.129 [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.

## 4 Application hints

### 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.*

## 4.130 [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`

## 4.131 [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:

#### **4 Application hints**

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

### **4.132 [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.

### **4.133 [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 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.

### **4.134 [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.

#### 4 Application hints

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

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.*

#### 4.135 [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.

---

**4 Application hints**

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.

**4.136 [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

Document version	Date of release	Description of changes
1.0	2019-03-29	This is the first version for TC37xEXT step AB
1.1	2019-07-19	Update: <ul style="list-style-type: none"> <li>New/updated text modules see column “Change” in tables 4..6 of errata sheet V1.1</li> <li>Removed: <ul style="list-style-type: none"> <li>SRI_TC.H002 (CPU Accesses from PFlash: Stall cycles for local and SRI resources - Documentation update): corrected in table “CPU Accesses: Stall cycles for local and SRI resources” in the CPU chapter of TC3xx User’s Manual V1.1.0 and following</li> </ul> </li> </ul>
1.2	2019-12-02	Update: <ul style="list-style-type: none"> <li>New/updated text modules see column “Change” in tables 4..6 of errata sheet V1.2</li> <li>Removed: <ul style="list-style-type: none"> <li>CCU_TC.H013 (Field SHBY in register OSCCON - Documentation Update): updated description of OSCCON.SHBY in TC3xx User’s Manual V1.1.0 and following</li> <li>CCU_TC.H014 (System Reset value of CCUCON0; Clock Ramp Example - Documentation Updates): updated CCUCON0 description in TC3xx User’s Manual V1.1.0 and following</li> <li>CPU_TC.H017 (MSUB.Q does not match MUL.Q+SUB.Q - Documentation Update): updated description in TriCore™ TC1.6.2 Core Architecture Manual V1.1, Vol. 2 Instruction Set</li> <li>FLASH_TC.H017 (UCB_SWAP MARKERHx and CONFIRMATIONHx - Documentation Update): updated description in NVM chapter “UCB_SWAP_ORIG and UCB_SWAP_COPY” of TC3xx User’s Manual V1.1.0 and following</li> <li>FLASH_TC.H018 (Sequence for Programming/Erasing - Documentation Update): updated description in DMU chapter “Performing Flash Operations” in TC3xx User’s Manual V1.2.0 and following</li> <li>LVDS_TC.H001 (Driver ground potential difference - Additional information): Note included in chapter “High performance LVDS pads” in TC37xEXT AB Data Sheet V1.0 and following</li> <li>STM_TC.H003 (Suspend control for STMx - Documentation Update): updated OCS register description in STM chapter of TC3xx User’s Manual V1.2.0 and following</li> </ul> </li> </ul>
1.3	2020-03-31	Update: <ul style="list-style-type: none"> <li>New/updated text modules see column “Change” in tables 4..6 of errata sheet V1.3</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> </ul> </li> </ul>
1.4	2020-07-06	Update: <ul style="list-style-type: none"> <li>New/updated text modules see column “Change” in tables 4..6 of errata sheet V1.4</li> </ul>



**Revision history**

<b>Document version</b>	<b>Date of release</b>	<b>Description of changes</b>
1.5	2020-10-23	<p>Update:</p> <ul style="list-style-type: none"> <li>New/updated text modules see column “Change” in tables 4..6 of errata sheet V1.5</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> </ul>
1.6	2021-01-22	<p>Update:</p> <ul style="list-style-type: none"> <li>New/updated text modules see column “Change” in tables 4..6 of errata sheet V1.6</li> <li>Table 1: reference for Safety Manual changed from V1.06 to V1.05</li> </ul>
1.7	2021-04-22	<p>Update:</p> <ul style="list-style-type: none"> <li>New/updated text modules see column “Change” in tables 4..6 of errata sheet V1.7 <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: EMEM_TC.H007, FlexRay_TC.H004, GETH_TC.H003, HSCT_TC.H010, 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.HWCFCGx (x=1-5) could have an unexpected value in application if pins HWCFCGx are left unconnected)</li> </ul>
1.8	2021-07-23	<p>Update:</p> <ul style="list-style-type: none"> <li>New/updated text modules see column “Change” in tables 4..6 of errata sheet V1.8 <ul style="list-style-type: none"> <li>Text modules already published in TC3xx Errata Advance Information 2021_05: GTM_AI.362, GTM_AI.364, GTM_AI.367, GTM_AI.370/371, GTM_AI.374..376</li> <li>Text modules already published in TC3xx Errata Advance Information 2021_06: FLASH_TC.055, GTM_AI.H004 (update see this errata sheet), MCMCAN_TC.H008, MTU_TC.018, PMS_TC.015</li> </ul> </li> </ul>
1.9	2021-11-04	<p>Update:</p> <ul style="list-style-type: none"> <li>New/updated text modules see column “Change” in tables 4..6 of errata sheet V1.9 <ul style="list-style-type: none"> <li>Text modules already published in TC3xx Errata Advance Information 2021-09: FLASH_TC.056, GTM_AI.358, GTM_AI.387, MCMCAN_AI.023, RESET_TC.H006, SAFETY_TC.023, SAFETY_TC.024</li> </ul> </li> <li>Table 2 (History List) of errata sheet V1.8: corrected “GTM.AI.*” to “GTM_AI.*”</li> <li>Removed <ul style="list-style-type: none"> <li>GTM_AI.H004: replaced by GTM_AI.387</li> <li>GTM_AI.262, GTM_AI.263: only apply to TC39x step AA</li> </ul> </li> </ul>

## Revision history

Document version	Date of release	Description of changes
1.10	2022-03-25	<p>Update:</p> <ul style="list-style-type: none"> <li>New/updated text modules see column “Change” in tables 4..6 of errata sheet V1.10 <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_AI.398, GTM_AI.400, GTM_TC.025, GTM_TC.H027, I2C_TC.H009, 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, GETH_AI.H003, GTM_AI.406, GTM_AI.408, OCDS_TC.H015, SMU_TC.H016</li> <li>Note: GTM_AI.263 and GTM_AI.262 re-included (also apply to TC37xEXT), with updated workaround 2 for GTM_AI.262</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
1.11	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 4..6 of errata sheet V1.11 <ul style="list-style-type: none"> <li>Text modules already published in TC3xx Errata Advance Information 2022-05: ADC_TC.H043, CPU_TC.H021, FLASH_TC.H021, GTM_AI.H425, MTU_TC.H016, SCU_TC.H025</li> <li>Text modules already published in TC3xx Errata Advance Information 2022-06: GTM_AI.441, GTM_AI.450, GTM_AI.451, GTM_AI.454, INT_TC.H006, LBIST_TC.H003, MCMCAN_AI.024, MCMCAN_AI.H002, PSI5_TC.005, SAFETY_TC.026, SAFETY_TC.027</li> </ul> </li> <li>Removed GTM_AI.425: replaced by GTM_AI.H425</li> <li>Removed due to updates in Safety Manual v1.06 and following: <ul style="list-style-type: none"> <li>SAFETY_TC.021 (section 6.283 SM[HW]:LMU:CFG_AS_AP)</li> </ul> </li> <li>Removed due to updates in Safety Manual v1.11 and following: <ul style="list-style-type: none"> <li>SAFETY_TC.004 (section 5.1 ESM[HW]:MCU:LBIST_MONITOR)</li> <li>SAFETY_TC.H001 (section 3.2.1 Purpose of the SEooC)</li> <li>SAFETY_TC.H003 (section 5.42 ESM[SW]:EDSADC:VAREF_PLAUSIBILITY and 5.51 ESM[SW]:EVADC:VAREF_PLAUSIBILITY)</li> <li>SAFETY_TC.H004 (section 5.3 ESM[HW]:PMS:VEXT_VEVRSB_OVERVOLTAGE)</li> <li>SAFETY_TC.H009 (Appendix A for TC39x, TC37xEXT, TC35x, TC33xEXT)</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.H014 (section 4.2.3.1.2 Monitoring Concept)</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
1.12	2022-11-11	<p>Update:</p> <ul style="list-style-type: none"> <li>Documentation reference changed to TC3xx User's Manual V2.0.0 and TC3xx ED Target Specification V3.2.0 (see table 1)</li> <li>New/updated text modules see column "Change" in tables 4..6 of errata sheet V1.12 <ul style="list-style-type: none"> <li>Text modules already published in TC3xx Errata Advance Information 2022-08: EDSADC_TC.H004, FlexRay_TC.H005, GTM_TC.028, SCR_TC.019</li> <li>Text modules already published in TC3xx Errata Advance Information 2022-09: CPU_TC.H021, 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> </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 TC37xEXT AB Data Sheet V1.1)</li> <li>ADC_TC.H034 (updated footnote on QCONV in table "VADC 5V" in TC37xEXT AB Data Sheet V1.1)</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>EDSADC_TC.003 (corrected in table "Settling Time Summary" in EDSADC chapter of TC3xx UM V1.6.0 (and newer versions))</li> <li>EDSADC_TC.P002 (updated footnote on table "DSADC 5V" in TC37xEXT AB Data Sheet V1.1)</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>GETH_AI.018 (updated in section "Description of the Transmit Checksum Offload Engine" in GETH chapter of TC3xx UM V2.0.0)</li> <li>GETH_AI.H002 (updated in section "Registers" in GETH chapter of TC3xx UM V1.4.0 (and newer versions))</li> <li>GETH_TC.001 (updated in table "Clock Lines of Ethernet MAC" in GETH chapter of product specific Appendix V1.3.0 (and newer versions))</li> <li>GETH_TC.P001 (corrected in chapter "Operating Conditions" in TC37xEXT AB Data Sheet V1.1)</li> <li>GETH_TC.H003 (updated in table "ETH MII Signal Timing Parameters" and "ETH RMII Signal Timing Parameters" in TC37xEXT AB Data Sheet V1.1)</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> </ul> </li> </ul>

**Revision history**

Document version	Date of release	Description of changes
		<ul style="list-style-type: none"> <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>- 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>- PADS_TC.P011 (updated in note below table “LVDS - IEEE standard LVDS general purpose link (GPL)” in TC37xEXT AB Data Sheet V1.1)</li> <li>- PINS_TC.P001 (updated in chapter “Pin Reliability in Overload” and table “Example Inactive Lifetime Temperature Profile” in TC37xEXT AB Data Sheet V1.1)</li> <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.H004 (updated in table “Supply Monitors” in column “Note/ Test Conditions” in rows for VEXTMON = 5.4V (typ) in TC37xEXT AB Data Sheet V1.1)</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>- RESET_TC.P003 (updated in table “Reset” in TC37xEXT AB Data Sheet V1.1)</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>- SDMMC_TC.H002 (updated figure “SDIO Card Interrupt Sequence” included in SDMMC chapter of TC3xx UM V1.6 (and newer versions))</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</li> <li>ADC_TC.H044, CCU6_TC.H001, FlexRay_AI.H010, 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.H473, GTM_AI.H480, GTM_AI.H481, GTM_AI.487, GTM_AI.488, GTM_AI.490, GTM_AI.492 (Errata already published in TC3xx Errata Advance Information 2023-01)</li> </ul> <p>Changed:</p> <ul style="list-style-type: none"> <li>GTM_AI.456</li> <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>SAFETY_TC.H011 - Updated in section 6.245 SM[HW]:GTM:TOM_TOM_MONITORING_WITH_IOM in safety manual v1.11 and newer versions</li> <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>ADC_TC.P015, CPU_TC.131, DSADC_TC.H010, 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>GTM_AI.362 - "WURMX or WURCX instruction is..." --&gt; "WURMX and WURCX instructions are..."</li> </ul>

## Revision history

Document version	Date of release	Description of changes
		<ul style="list-style-type: none"> <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>
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, FPI_TC.H002, GTM_AI.461, GTM_AI.H497, GTM_AI.H502, GTM_TC.031, GTM_TC.H034, 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, EDSADC_TC.H005, GETH_AI.H004 (Errata already published in TC3xx Errata Advance Information 2023-06)</li> </ul> <p>Changed:</p> <ul style="list-style-type: none"> <li>GTM_AI.458, GTM_AI.477, GTM_AI.478, GTM_AI.490, GTM_AI.492, GTM_AI.H473</li> <li>GTM_AI.262 (Erratum already published in TC3xx Errata Advance Information 2023-05)</li> <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>DSADC_TC.H010 - Description for the use case where two or more DSADC channels have to provide synchronous results added to section "Basic Initialization Sequence" in EDSADC chapter of TC3xx UM V2.0.0</li> <li>EDSADC_TC.H003 - Description included in section "Stopping the Integration Window" in EDSADC chapter of TC3xx UM V1.5.0 (and newer versions)</li> <li>PADS_TC.H009 - Replaced by PADS_TC.016</li> </ul>

**Revision history**

Document version	Date of release	Description of changes
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.H515, GTM_AI.H803</li> <li>CPU_TC.H023, GTM_AI.442, PMS_TC.H018 (Errata already published in TC3xx Errata Advance Information 2023-09)</li> <li>GTM_AI.410, GTM_AI.483, GTM_AI.507, 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> <li>GTM_AI.458 (Change of erratum already published in TC3xx Errata Advance Information 2023-10)</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.458, GTM_AI.478, GTM_AI.492, GTM_AI.H473, GTM_AI.H480, GTM_AI.H481, GTM_AI.H482, GTM_AI.H497, GTM_AI.H502, 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>

**Table 34 Errata fixed in this step (Reference = TC37xEXT step AA)**

Errata	Short description	Change
BROM_TC.H016	CHSW fails check of PMS_EVR registers after software triggered LBIST	Fixed
PER_PLL_TC.001	Peripheral PLL weakness for 25 MHz input clock when using Divider Bypass	Fixed



## Trademarks

All referenced product or service names and trademarks are the property of their respective owners.

**Edition 2023-12-11**

**Published by**

**Infineon Technologies AG**  
**81726 Munich, Germany**

**© 2023 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-egt1636535921475**

## Important notice

The information given in this document shall in no event be regarded as a guarantee of conditions or characteristics ("Beschaffenhheitsgarantie").

With respect to any examples, hints or any typical values stated herein and/or any information regarding the application of the product, 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.

In addition, any information given in this document is subject to customer's compliance with its obligations stated in this document and any applicable legal requirements, norms and standards concerning customer's products and any use of the product of Infineon Technologies in customer's applications.

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.