

# AURIX™ TC33xEXT errata sheet

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

10472AERRA

## 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™ TC33xEXT Appendix to User's Manual	V2.0.0	2021-02
TC33xEXT AA-Step Data Sheet	V1.1	2021-03
TriCore™ TC1.6.2 Core Architecture Manual:		
• Core Architecture (Vol. 1)	V1.2.2	2020-01-15
• Instruction Set (Vol. 2)	V1.2.2	2020-01-15
AURIX™ TC3xx 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> .....	1
	<b>Table of contents</b> .....	3
<b>1</b>	<b>Errata overview</b> .....	4
<b>2</b>	<b>Functional deviations</b> .....	12
<b>3</b>	<b>Parametric deviations</b> .....	61
<b>4</b>	<b>Application hints</b> .....	63
	<b>Revision history</b> .....	119
	<b>Disclaimer</b> .....	125

## 1 Errata overview

### 1 Errata overview

List of errata referenced in this document.

**Table 2 Functional deviations**

Issue title	Change	Page
[BROM_TC.013] CAN BSL does not send error message if no valid baudrate is detected		12
[BROM_TC.014] Lockstep comparator alarm for CPU0 after warm PORST, system or application reset if lockstep is disabled		12
[BROM_TC.016] Uncorrectable ECC error in Boot Mode Headers		12
[CCU_TC.005] ASC and CAN bootstrap loaders may not work if external clock is missing		13
[CPU_TC.130] Data Corruption when ST.B to local DSPR coincides with external access to same address		13
[CPU_TC.131] Performance issue when MADD or MSUB instructions use E0 or D0 register as accumulator		14
[CPU_TC.132] Unexpected PSW values used upon Fast Interrupt entry		14
[CPU_TC.133] Test sequence for DTAG single or double bit errors		15
[DAP_TC.005] DAP client_read: dirty bit feature of Cerberus' Triggered Transfer Mode		16
[DAP_TC.007] Incomplete client_blockread telegram in DXCM mode when using the "read CRCup" option		16
[DAP_TC.010] Performance when accessing EMEM in UWM and WM modes		16
[DMA_TC.066] DMA double buffering operations - Update address pointer		17
[DMA_TC.067] DMA Double Buffering Software Switch buffer overflow		17
[DMA_TC.068] DMA Double Buffering lost DMA request		18
[FLASH_TC.053] Erase size limit for PFLASH		18
[FLASH_TC.055] Multi-bit errors detected by PFlash are not communicated to SPB masters		19
[FLASH_TC.056] Reset value for register HF_ECCC is 0x0000 0000 - Documentation correction		19
[GETH_AI.001] Packets with Destination Address (DA) mismatch are delayed until EOP is received in threshold (cut-through) mode		20
[GETH_AI.008] Application Error Along with Start-of-Packet Can Corrupt the FCS Field of the Previous Frame in the MAC Pipeline		21
[GETH_AI.009] Corrupted Rx Descriptor Write Data		21
[GETH_AI.010] Fatal Bus Error Interrupt Might Be Generated for Incorrect DMA Channel		22
[GETH_AI.011] Receive Queue Overflow at End of Frame Along with SPRAM Read-Write Conflict Can Cause Data Loss		22
[GETH_AI.012] Incorrect Flexible PPS Output Interval When Fine Time Correction Method is Used		23
[GETH_AI.013] False Dribble and CRC Error Reported in RMII PHY 10 Mbps Mode		23

**(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		24
[GETH_AI.015] MAC Receive VLAN Tag Hash Filter Always Operates in Default Mode		25
[GETH_AI.016] Receive DMA Header Split Function Incorrectly Overruns the Allocated Header Buffer		26
[GETH_AI.017] Carrier-Sense Signal Not Generated When False Carrier Detected in RGMII 10/100 Mbps Mode		26
[GETH_TC.002] Initialization of RGMII interface		27
[MCDS_TC.052] TriCore™ wrap around write access causes redundant MCDS message		28
[MCDS_TC.064] ACCEN0 register write not supervisor protected		28
[MCDS_TC.065] Selection of SRI trace sources		29
[MCDS_TC.066] Selection of CPU trace sources		29
[MCDS_TC.067] MCDS kernel reset shall not be used		29
[MCMCAN_AI.015] Edge filtering causes mis-synchronization when falling edge at Rx input pin coincides with end of integration phase		30
[MCMCAN_AI.017] Retransmission in DAR mode due to lost arbitration at the first two identifier bits		31
[MCMCAN_AI.018] Tx FIFO message sequence inversion		31
[MCMCAN_AI.019] Unexpected High Priority Message (HPM) interrupt		33
[MCMCAN_AI.022] Message order inversion when transmitting from dedicated Tx Buffers configured with same Message ID		34
[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		34
[MCMCAN_AI.024] Frame transmitted despite confirmed transmit cancellation		35
[MCMCAN_AI.025] Sporadic data corruption (payload) in case acceptance filtering is not finished before reception of data R3 (DB7..DB4) is completed		36
[MCMCAN_TC.006] MCMCAN specific access protection mechanisms		39
[MEMMAP_TC.001] Size of PFLASH and DFLASH - Correction to TC33xEXT and TC33x/TC32x Appendix		40
[MTU_TC.012] Security of CPU cache memories during runtime is limited		40
[MTU_TC.017] Unexpected alarms after application reset		41
[MTU_TC.018] Gated SRAM alarms		41
[PADS_TC.011] Pull-ups activate on specific analog inputs upon PORST		42
[PADS_TC.013] Buffer type definition for P21.2: no ES functionality - Data Sheet documentation correction		42

**(table continues...)**

## 1 Errata overview

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

Issue title	Change	Page
[PADS_TC.016] Pull-ups active on P33 and P34 pins in standby mode when SCR is disabled and VEXT not supplied		43
[PMS_TC.005] Voltage rise at P33 and P34 up to $V_{EVR\overline{S}B}$ during start-up and up to $V_{LVDR\overline{S}T\overline{S}B}$ during power-down		43
[PMS_TC.006] PORST not released during cold power-on reset until $V_{DDM}$ is available		44
[PMS_TC.007] VDDP3 or VDD Overvoltage during start-up may not be detected by PBIST		44
[PMS_TC.011] VEXT supplied PU2 and PD2 pads always in tristate after standby entry - Documentation correction		45
[PMS_TC.015] EVRC synchronization – Documentation update for register EVRSDCTRL11 (PMS) and EVRSDCTRL2 (PMSLE)		46
[QSPI_TC.006] Baud rate error detection in slave mode (error indication in current frame)		46
[QSPI_TC.009] USR Events for PT1=2 (SOF: Start of Frame)		47
[QSPI_TC.010] Move Counter Mode - USR Events for PT1=4 (RBF: Receive Buffer Filled)		47
[QSPI_TC.013] Slave: No RxFIFO write after transmission upon change of BACON.MSB		47
[QSPI_TC.014] Slave: Incorrect parity bit upon TxFIFO underflow		47
[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		48
[QSPI_TC.017] Slave: Reset when receiving an unexpected number of bits		48
[RIF_TC.004] External ramp feature not reliable		48
[RIF_TC.005] RIF LVDS calibration must not be used		49
[SAFETY_TC.023] MCU infrastructure Safety Related Function - Documentation update		49
[SAFETY_TC.024] Clock alive monitor for $f_{SPB}$ - Documentation update		49
[SAFETY_TC.025] Wrong alarm listed in safety mechanism SM[HW]:SRI:SRI_TRANSACTION_INTEGRITY		50
[SAFETY_TC.026] Alarm for SM[HW]:IR:CFG_MONITOR - Documentation update		50
[SAFETY_TC.027] Single point fault detection for lockstep CPUs - Documentation update		50
[SCR_TC.015] Bit SCU_PMCON1.WCAN_DIS does not disable WCAN PCLK input		51
[SCR_TC.016] DUT response to first telegram has incorrect C_START value		51
[SCR_TC.018] SSC Receive FIFO not working		51
[SCR_TC.019] Accessing the XRAM while SCR is in reset state		52
[SCR_TC.020] Stored address in mon_RETH may be wrong after a break event		52
[SCR_TC.021] RTC not counting after reset if P33.10 is high		52

**(table continues...)**

## 1 Errata overview

**Table 2 (continued) Functional deviations**

Issue title	Change	Page
[SCR_TC.022] Effect of application or system reset and warm PORST on MC77_ECCD and MC78_ECCD for SCR RAMs		53
[SCR_TC.023] External interrupts EXINT0, EXINT1 may get locked		53
[SCR_TC.024] Field ADRES in register ADCOMP_RES - Documentation correction		54
[SCU_TC.031] Bits SCU_STSTAT.HWCFGx (x=1-5) could have an unexpected value in application if pins HWCFGx are left unconnected		54
[SCU_TC.033] $\overline{\text{TESTMODE}}$ pin shall be held at static level during LBIST		55
[SDMMC_AI.001] Timeout error when BOOT ACK driven on all data lines in SD/eMMC mode		55
[SMU_TC.012] Unexpected alarms when registers FSP or RTC are written		56
[SMU_TC.013] Unexpected setting of Alarm Missed Event bit xAEM in Alarm Executed Status register SMU_AEX		57
[SMU_TC.015] SMU alarm emulation might trigger unwanted active alarm reaction	new	58
[SPU_TC.019] ACCEN0 register description - Correction		58
[SPU_TC.020] SPU Power Sensitivity to IDM_RM_IOLR.ILR setting		58
[SPU_TC.023] Write to register SPU_STAT: use only 32-bit writes		59
[SPU_TC.024] Register CRC: assume PACTR.RST bit as 0 <sub>B</sub> - Documentation Update		60

**Table 3 Parametric deviations**

Issue title	Change	Page
[CCU_TC.P001] Back-up clock accuracy after trimming - Disregard datasheet footnote		61
[FLASH_TC.P003] Program Flash Erase Time per Multi-Sector Command		61
[PADS_TC.P014] Electrical characteristics for P20.2/ $\overline{\text{TESTMODE}}$		61
[PORST_TC.P002] $V_{IH}$ and $V_{IL}$ definition for PORST pad - Additional Data Sheet footnote		62

**Table 4 Application hints**

Issue title	Change	Page
[ADC_TC.H026] Additional waiting phase in slow standby mode		63
[ADC_TC.H032] ADC accuracy parameters - Definition		63
[ADC_TC.H033] Basic initialization sequence for primary and secondary EVADC groups		63
[ADC_TC.H035] Effect of input leakage current on Broken Wire Detection		64
[ADC_TC.H043] Information on supervision signal $V_{ANACOMM}$ not relevant - Documentation update		65
[ADC_TC.H044] Start-up calibration timing in synchronized mode - Documentation update		65
[ADC_TC.H045] Level selection for broken wire detection feature		66

**(table continues...)**

## 1 Errata overview

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

Issue title	Change	Page
[AGBT_TC.H004] Configuration of registers PYCR2 and PACR2		66
[AGBT_TC.H005] Availability of AGBT depending on feature package		66
[ASCLIN_TC.H001] Bit field FRAMECON.IDLE in LIN slave tasks		67
[ASCLIN_TC.H006] Sample point position when using three samples per bit		67
[ASCLIN_TC.H007] Handling TxFIFO and RxFIFO interrupts in single move mode		67
[ASCLIN_TC.H008] SPI master timing – Additional information to Data Sheet characteristics		68
[BROM_TC.H009] Re-enabling lockstep via BMHD		68
[BROM_TC.H011] Assertion of ALM7[14] after cold or warm PORST		69
[BROM_TC.H012] Availability of $V_{DDSB}$ during start-up		69
[BROM_TC.H014] SSW behavior in case of wrong state or uncorrectable error in UCBs - Documentation Update		69
[BROM_TC.H020] Processing in case no valid BMHD found		70
[CCU6_TC.H001] CCU6 module clock source information - Documentation Update		70
[CCU_TC.H012] Configuration of the Oscillator- Documentation Update		70
[CCU_TC.H017] XTAL1 input signal compatibility with Infineon CTRX81xx devices	new	71
[CCU_TC.H018] Encoding of bit-field ADASDIV in register CCUCON5		71
[CLC_TC.H001] Description alignment for bits DISR, DISS, EDIS in register CLC - Documentation Update		71
[CPU_TC.H019] Semaphore handling for shared memory resources		72
[CPU_TC.H021] Resource update failure despite correct SW synchronization upon retried FPI write transactions by CAN and E-Ray modules		74
[CPU_TC.H022] Store buffering and the effect of bit SMACON.IODT		75
[CPU_TC.H023] CPU_SYSCON register safety protection description clarification	new	76
[DMA_TC.H018] Maximum size of circular buffers is 32 Kbytes		76
[DTS_TC.H002] Unexpected alarms after start-up/wake-up when temperature is close to lower/upper limit		76
[EMEM_TC.H006] Triggering EMEM MEMCON.RMWERR and INTERR flags		77
[EMEM_TC.H007] Access restrictions to EMEM for tools		77
[EVR_TC.H001] External input capacitor value - Additional Data Sheet footnote		78
[FLASH_TC.H021] Flash Wait State configuration		78
[FLASH_TC.H024] PFLASH erase and program time is affected by time slicing but not clearly documented		78
[FPI_TC.H003] Burst write access may lead to data corruption		79
[GETH_AI.H001] Preparation for Software Reset		79

**(table continues...)**



## 1 Errata overview

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

Issue title	Change	Page
[GETH_AI.H003] Undefined behavior when LD bit is set and buffer length B1L or B2L is zero - Additional information		80
[GETH_AI.H004] MAC address 0 configuration sequence		80
[GETH_TC.H002] Stopping and Starting Transmission - Additional information		80
[GPT12_TC.H002] Bits TxUD and TxUDE in incremental interface mode - Additional information		81
[HSPDM_TC.H001] Accesses to specific HSPDM registers while 160/320 MHz clocks are disabled		82
[HSPDM_TC.H002] Gap between stop/start of bit streaming		82
[INT_TC.H006] Number of SRNs supporting external interrupt/service requests – Documentation update		82
[LBIST_TC.H003] Update reset behavior of LBISTCTRL0 and LBISTCTRL3 register - Additional information		83
[LBIST_TC.H005] Effects of LBIST execution on P33.8		83
[MBIST_TC.H001] Destructive MBIST requires DSPR0 initialization		83
[MBIST_TC.H002] Time for 4N non-destructive test		84
[MCDS_TC.H007] Program trace of CPUx (x > 0) program start not correct		84
[MCMCAN_AI.H001] Behavior of interrupt flags in CAN Interface (MCMCAN)		85
[MCMCAN_AI.H002] Bus off recovery		85
[MCMCAN_TC.H001] Behavior of undefined data bytes read from Receive Buffer		86
[MCMCAN_TC.H006] Unintended behavior of receive timeout interrupt		86
[MCMCAN_TC.H007] Delayed time triggered transmission of frames		86
[MCMCAN_TC.H008] Parameter “CAN Frequency” - Documentation update to symbol in Data Sheet		87
[MTU_TC.H015] ALM7[0] may be triggered after cold PORST		87
[MTU_TC.H016] MCI_FAULTSTS.OPERR[2] may be triggered at power-up in case LBIST is not run		88
[MTU_TC.H019] Application reset value of register SRC_MTUDONE different to documentation		88
[NVM_TC.H001] References to DMU_HP_PROCONTP – Typo in TC3xx user manual		88
[OCDS_TC.H014] Avoiding failure of key exchange command due to overwrite of COMDATA by firmware		89
[OCDS_TC.H015] System or Application Reset while OCDS and lockstep monitoring are enabled		89
[OCDS_TC.H016] Release of application reset via OJCONF may fail		90
[OCDS_TC.H018] Unexpected stop of Startup Software after system or application reset		90

**(table continues...)**

## 1 Errata overview

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

Issue title	Change	Page
[PADS_TC.H008] Overload coupling for LVDS RX pads – Additional information		90
[PMS_TC.H003] $V_{DDPD}$ voltage monitoring limits		91
[PMS_TC.H008] Interaction of interrupt and power management system - Additional information		92
[PMS_TC.H009] Interaction of warm reset and standby mode transitions		93
[PMS_TC.H011] Supply mode and topology selection - Allowed combinations of VEXT and VDDM - Documentation update		94
[PMS_TC.H018] Bit SWDLVL in register EVRSTAT is always 1 when EVRC is OFF	new	94
[PMS_TC.H019] Limitation of power-cycles - Additional datasheet footnote	new	94
[PORTS_TC.H018] Misleading footnote on pad driver mode selection table		95
[QSPI_TC.H008] Details of the baud rate and phase duration control - Documentation update		95
[RESET_TC.H006] Certain registers may have different reset values than documented in TC3xx User's Manual - Documentation update		95
[RESET_TC.H007] Cold Power on Reset Boot Time – Additional information		98
[RIF_TC.H007] Initialization sequence for RIF		98
[SAFETY_TC.H013] ESM[SW]:SYS:MCU_FW_CHECK - Access to MC40 FAULTSTS register – Additional information		98
[SAFETY_TC.H017] Safety Mechanisms requiring initialization - Documentation update		98
[SAFETY_TC.H019] SM[HW]:NVM.FSIRAM:REG_MONITOR_TEST should not be considered		101
[SAFETY_TC.H020] Test of SM[HW]:VMT:REG_MONITOR is missing - Documentation update		101
[SCR_TC.H009] RAM ECC Alarms in Standby Mode		101
[SCR_TC.H010] HRESET command erroneously sets RRF flag		102
[SCR_TC.H011] Hang-up when warm PORST is activated during Debug Monitor Mode		102
[SCR_TC.H012] Reaction in case of XRAM ECC Error		102
[SCR_TC.H014] Details on WDT pre-warning period		103
[SCR_TC.H016] SCR current consumption in IDLE mode and 70 kHz clock		103
[SCU_TC.H020] Digital filter on ESRx pins - Documentation update		103
[SCU_TC.H021] LBIST execution affected by TCK/DAP0 state		103
[SCU_TC.H023] Behavior of bit RSTSTAT.PORST after wake-up from standby mode		104
[SCU_TC.H025] Field EEA in register CHIPID - Additional information		104
[SCU_TC.H026] Unexpected alarm ALM0[1] during warm reset		104
[SCU_TC.H027] Bit field INP0 and INP1 in register EICRi - Documentation correction		105
[SCU_TC.H028] ERU configuration changes may lead to ERU reactions	changed	105

**(table continues...)**

## 1 Errata overview

**Table 4** (continued) Application hints

Issue title	Change	Page
[SDMMC_TC.H001] Idle State of SDMMC0_CLK		106
[SENT_TC.H006] Parameter $V_{ILD}$ on pads used as SENT inputs		106
[SENT_TC.H007] Range for divider value DIV - Documentation correction		113
[SMU_TC.H010] Clearing individual SMU flags: use only 32-bit writes		113
[SMU_TC.H012] Handling of SMU alarms ALM7[1] and ALM7[0]		113
[SMU_TC.H013] Increased Fault Detection for SMU Bus Interface (SMU_CLC Register)		114
[SMU_TC.H016] SMU_stdby restriction for using P33.8 as Emergency Stop input		114
[SMU_TC.H017] Handling of ALM21[7] when safety flip-flop self-test is executed		115
[SMU_TC.H019] TC33xEXT: Incorrect ALM1[0] and ALM1[2] descriptions	new	115
[SPU_TC.H011] In Place FFT with RIF data as input		115
[SPU_TC.H013] CFAR and Local Max function when spectrum extension is disabled - Additional information		116
[SRI_TC.H001] Using LDMST and SWAPMSK.W instructions on SRI mapped peripheral registers (range 0xF800 0000-0xFFFF FFFF)		116
[SSW_TC.H001] Security hardening measure for the startup behavior		117
[STM_TC.H004] Access to STM registers while STMDIV = 0		118

## 2 Functional deviations

## 2 Functional deviations

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

#### Description

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

#### Workaround

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

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

#### Description

Lockstep monitoring may be disabled in the Boot Mode Header structure (BMHD) for each 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.

### 2.3 [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 Functional deviations**

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

#### **Description**

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

#### **Effects**

No code download for initial device programming is started.

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

#### **Workaround**

Trigger reset and retry if bootstrap loader does not respond.

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

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

#### **Description**

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

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

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

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

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

#### **Workaround 1**

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

#### **Workaround 2**

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

#### **Workaround 3**

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

## 2 Functional deviations

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

#### Description

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

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

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

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

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

#### Example

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

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

#### Workaround

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

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

#### Description

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

## 2 Functional deviations

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

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

### Workaround 1

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

```
_exception_handler:
    CALL _common_handler
    RFE

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

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

### Workaround 2

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

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

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

### Description

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

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

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

## **2 Functional deviations**

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

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

### **Workaround**

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

### **DTAG SRAM test sequence**

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

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

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

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

### **Description**

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

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

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

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

### **Description**

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

### **Workaround**

Do not use CRCup option with client\_blockread telegrams in DXCM mode.  
Instead the CRCup can be read by a dedicated getCRCup telegram.

## **2.11 [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.



## 2 Functional deviations

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”).*

### 2.12 [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.

### 2.13 [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 Functional deviations

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

#### Description

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

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

#### Workaround

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

1. Software must disable the servicing of interrupt service requests by the DMA channel by disabling the corresponding Interrupt Router (IR) Service Request Node (SRN)
  - a. Software shall write  $IR\_SRCi.SRE = 0_B$
2. Software must halt the DMA channel configured for DMA double buffering
  - a. Software shall write DMA channel  $TSRc.HLTREQ = 1_B$
  - b. Software shall monitor DMA channel  $TSRc.HLTACK = 1_B$
3. Software must monitor the DMA Channel Transaction Request State
  - a. Software shall read DMA channel  $TSRc.CH$  and store the value in a variable  $SAVED\_CH$
4. Software must switch the source or destination buffer
  - a. Software shall write DMA channel  $CHCSRc.SWB = 1_B$
  - b. Software shall monitor the DMA channel frozen bit  $CHCSRc.FROZEN$
5. When the DMA channel has switched buffers (DMA channel  $CHCSRc.FROZEN = 1_B$ )
  - a. If ( $SAVED\_CH == 1$ ), software shall trigger a DMA software request by writing DMA channel  $CHCSRc.SCH = 1_B$  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  $TSRc.HLTCLR = 1_B$
7. Software must enable the servicing of interrupt service requests by the DMA channel
  - a. Software shall write  $IR\_SRCi.SRE = 1_B$

The software must include an error routine.

1. Software must monitor for interrupt overflows ( $IR\_SRCi.IOV = 1_B$ ) and lost DMA requests ( $TSRc.TRL = 1_B$ )
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

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

#### Description

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

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

#### Workaround

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

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

## **2 Functional deviations**

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

#### **Description**

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

- “Multi-bit error and not All-0 error” and
- “Multi-bit error and All-0 error”

that a bus error is returned to the reading master.

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

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

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

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

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

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

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

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

#### **Workaround**

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

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

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

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

#### **Description**

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

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

## 2 Functional deviations

### Documentation correction

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

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

## 2.18 [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.

### 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 Functional deviations**

### **2.19 [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.

### **2.20 [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.

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 Functional deviations**

### **2.21 [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 to 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.

### **2.22 [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.

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 Functional deviations**

### **2.23 [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.

### **2.24 [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.



## 2 Functional deviations

### 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.

## 2.25 [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 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 RTW 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)



## **2 Functional deviations**

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

## **2.26 [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.

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.

## **2 Functional deviations**

### **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.

### **2.27 [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.

### **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).

### **2.28 [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.

## 2 Functional deviations

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.

## 2.29 [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 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.

## 2 Functional deviations

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 =  $000_B$
- 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 =  $001_B$  (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 =  $0_B$
- Configure remaining GMAC registers according to application requirements

### 2.30 [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.

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

#### Description

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

## **2 Functional deviations**

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”).

### **2.32 [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.

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.

### **2.33 [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/MCDSLlight 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).

### **2.34 [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/MCDSLlight 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/MCDSLlight kernel reset.

## **2 Functional deviations**

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

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

#### **Description**

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

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

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

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

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

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

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 Functional deviations**

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

#### **Description**

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

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

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

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

#### **Scope**

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

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

#### **Effects**

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

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

#### **Workaround**

None.

### **2.37 [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

## **2 Functional deviations**

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

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

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

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

### **Scope**

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

### **Effects**

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

### **Workaround**

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

#### **Workaround 1**

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

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

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

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

#### **Description**

There are two configurations where the issue occurs:

#### **Configuration A**

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

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

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

#### **Configuration B**

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

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

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

#### **Scope**

The erratum is limited to:

- 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:

## **2 Functional deviations**

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

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

### **Configuration B**

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

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

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

## **2.39 [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.

## **2.40 [MCMCAN\_AI.023] Incomplete description in section “Dedicated Tx Buffers” and “Tx Queue” of the M\_CAN documentation in the user**

## 2 Functional deviations

### **manual related to transmission from multiple buffers configured with the same Message ID**

#### **Description**

##### **Dedicated Tx Buffers**

- Wording user manual

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

- Enhancement - additional text

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

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

##### **Tx Queue**

- Wording user manual - to be deleted

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

- Replacement

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

- Wording user manual - to be deleted

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

- Replacement

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

#### **Scope**

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

#### **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.

### **2.41 [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.

## **2 Functional deviations**

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

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

### **Scope**

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

### **Effects**

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

### **Workaround**

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

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

## **2.42 [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 ( $f_{\text{MCANH}}$ )
- 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 ( $n \leq 17$ ).

## 2 Functional deviations

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

**Figure 1 TC3xx - Rx Buffer and FIFO Element ( $R_i$  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  $R_i$  of a received message into the Message RAM (after acceptance filtering is done) has not completed before  $R_{(i+1)}$  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  $R_i$  from the cache and write of data word  $R_{(i+1)}$  into the cache of the Rx handler happens

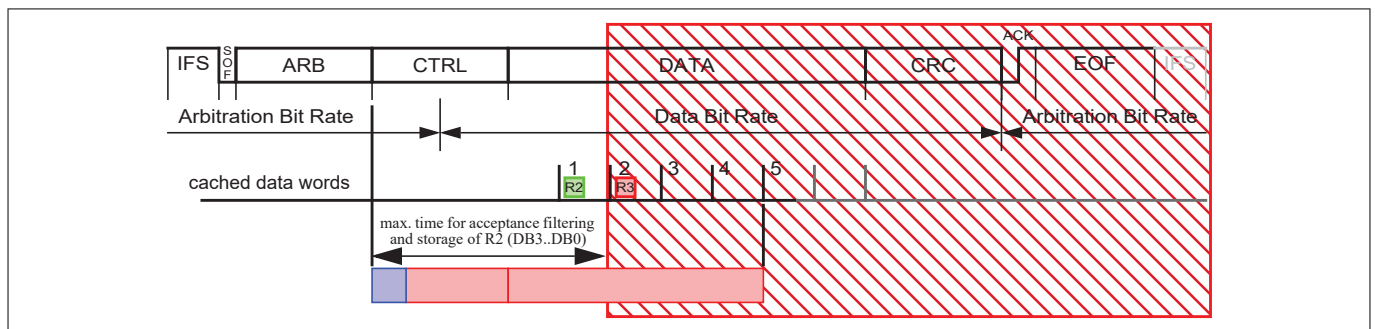
The data will be corrupted in a way, that in the Message RAM  $R_{(i+1)}$  has the same content as  $R_i$ .

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

- Rx FIFO: FIFO put index  $RXF_nS.FnPI$  is updated
- Dedicated Rx Buffer: New Data flag  $NDAT_n.NDxx$  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 2 CAN Frame with  $DLC > 4$**

## 2 Functional deviations

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

**Table 5 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.

### 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.

## 2 Functional deviations

### 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.

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

### Description

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

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

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

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

### Effects

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

**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 Functional deviations

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

#### Description

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

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

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

#### Documentation correction

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

**Table 6 Address Map as seen by Bus Masters on Bus SRI - Corrections**

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

### 2.45 [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.



## 2 Functional deviations

### 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.

## 2.46 [MTU\_TC.017] Unexpected alarms after application reset

### Description

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

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

- MC0 (CPU0\_DMEM)
- MC34 (CPU0\_DMEM1), and
- MC35 (CPU1\_DMEM1)

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

### Workaround

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

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

## 2.47 [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.*

## 2 Functional deviations

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 ( $\text{PMSWCR0.STBYRAMSEL} \neq 000_B$ ) - then just before entering standby, perform an additional dummy read to DLMU\_STBY location 0x9000 0000 or 0xB000 0000 (when using CPU0 dLMU RAM) and 0x9001 0000 or 0xB001 0000 (when using CPU1 dLMU RAM). This dummy read triggers the alarm propagation and ensures that no alarms are lost due to standby entry.

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

### Description

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

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

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

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

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

## 2.49 [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.

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

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

## 2 Functional deviations

### Correction to Data Sheet

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

### 2.50 [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 VEVR SB, 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 VEVR SB supplied SCR pins are under SCR control (implies SCR is enabled)
- If SCR is not enabled and VEXT pins are not supplied, then VEVR SB supplied pins (P33, P34) are in tristate

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

#### Actual behavior

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

#### Workaround 1

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

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

#### Workaround 2

Enable the SCR during standby mode.

#### Workaround 3

Supply VEXT pins during standby mode.

### 2.51 [PMS\_TC.005] Voltage rise at P33 and P34 up to $V_{EVR SB}$ during start-up and up to $V_{LVD RST SB}$ 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  $V_{LVD RST 5}$  and  $V_{LVD RST SB}$ , respectively.

## 2 Functional deviations

Therefore, the pad behavior at P33 and P34 pins is “pull-up”, even if pin HWCFG[6] = 0, with the following characteristics:

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

### Workaround

If an application requires to ensure the state of P33 and P34 pins within the logical “low” level, then an external pull-down must be used which can overdrive the internal pull-up.

In order to quantify the strength of such an external pull-down, parameter “Pull-up current” ( $I_{PUH}$ , CC) for the respective pin may be used as the reference. There, the values for the internal pull-up resistor (for TTL and AL) can be found via parameter  $R_{MDU}$  in table “VADC 5V” (see footnotes on parameter “Pull-up current” in the Data Sheet).

## 2.52 [PMS\_TC.006] PORST not released during cold power-on reset until $V_{DDM}$ is available

### Description

Upon a cold power-on reset, the PORST pin may be kept asserted by the PMS until the ADC Analog Supply voltage ( $V_{DDM}$ ) is above 500 mV. This might lead to an additional start-up delay dependent on when  $V_{DDM}$  is available from the external regulator relative to the  $V_{EXT}$ ,  $V_{DDP3}$ , and  $V_{DD}$  supplies. When  $V_{DDM}$  is below 500 mV, the device may not be able to carry out PBIST. As a consequence, the device remains in PORST state until PBIST result gets available and supply voltages are in operating condition. PBIST result is always available when  $V_{DDM}$  is above 500 mV. From PBIST point of view, it is important to have enough  $V_{DDM}$  voltage level, and there is no special timing requirement on when to supply  $V_{DDM}$ .

During operation, if  $V_{DDM}$  drops below the secondary monitor undervoltage threshold, an SMU alarm is generated. If  $V_{DDM}$  further drops below 500 mV, the dedicated ADC of the secondary voltage monitor stops converting and the Secondary Monitor Activity Counter (EVRMONSTAT1.ACTVCNT) freezes at the last value.

### Workaround

The ADC Analog Supply voltage ( $V_{DDM}$ ) has to be available and needs to be above 500 mV to ensure proper release of PORST during start-up and proper functioning of secondary monitors.

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

## 2.53 [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.

## 2 Functional deviations

### 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.

### 2.54 [PMS\_TC.011] VEXT supplied PU2 and PD2 pads always in tristate after standby entry - Documentation correction

#### Description

Tristate mode is enabled for VEXT supplied PU2 and PD2 pads (marked PU2 / VEXT and PD2 / VEXT in column “Buffer Type” in the Data Sheet) at the moment of and after entry to standby mode, regardless of the PMSWCR5.TRISTREQ bit setting and the HWCFG[6] pin setting (reflected in the PMSWSTAT register).

For a definition of the buffer types see also chapter “Legend” in the Data Sheet.

#### Recommendation

If the application requires the pull-up state of VEXT supplied PU2 pads (or pull-down state of PD2 pads), then it shall ensure it by means of external pull-up devices (or pull-down devices for PD2 pads) in the event of:

- Standby entry while the VEXT supply ramps down
- Standby entry with the VEXT supply available

#### Documentation correction for TC3xx User’s Manual V1.5.0 and following

In TC3xx User’s Manual V1.5.0 and following versions, the description of this behavior has been included in the PMS and PMSLE chapters. Erroneously, the term “PU1” was used instead of “PU2 and PD2”.

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 Functional deviations

### 2.55 [PMS\_TC.015] EVRC synchronization – Documentation update for register EVRSDCTRL11 (PMS) and EVRSDCTRL2 (PMSLE)

#### Description

The formulas for  $d f_{\text{MAXDEV}}$  (Maximum Deviation of the Synchronization Input Frequency) and SYNCHYST (Lock Unlock Hysteresis Window) that are documented in the description of fields SYNCMAXDEV and SYNCHYST in register EVRSDCTRL11 (chapter PMS) and EVRSDCTRL2 (chapter PMSLE) of the TC3xx User's Manual shall be corrected/updated as listed below.

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

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

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

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

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

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

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

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

### 2.56 [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 Functional deviations**

### **2.57 [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.

### **2.58 [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.

### **2.59 [QSPI\_TC.013] Slave: No Rx FIFO 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 Rx FIFO write of the currently received frame may not occur.

Also in case of a Tx FIFO underflow, the Rx FIFO 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 Rx FIFO write has occurred. This implies the need for a gap between frames if a BACON update occurs.

### **2.60 [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.



## **2 Functional deviations**

### **Workaround**

If parity is enabled, select even parity if data length is odd, and select odd parity if data length is even.

### **2.61 [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 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.

### **2.62 [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>

### **2.63 [RIF\_TC.004] External ramp feature not reliable**

#### **Description**

The external ramp feature based on input signal RAMP1 does not work reliably and should not be used. Otherwise, the final sample of a ramp may be corrupted or missing/dropped.

#### **Workaround**

Use the Frame Watchdog to identify the end of a ramp.

## **2 Functional deviations**

### **2.64 [RIF\_TC.005] RIF LVDS calibration must not be used**

#### **Description**

The AURIX™ TC3xx User's Manual offers the capability of RIF LVDS calibration for data rates above 200 MBaud. Using the RIF LVDS calibration on the RIF0 and/or RIF1 instance can result in unreliable calibration status showing either a time-out (calibration doesn't finish within the specified timeframe) or calibration NOK (not OK) status. This might lead to wrong calibration, and the calibration results must not be used for application.

#### **Impacted use cases**

All applications using data rates above 200 MBaud might be affected and therefore changes on RIF LVDS calibration usage is required.

#### **Workaround**

The RIF LVDS calibration feature has to be completely disabled under all conditions. This has no negative impact on RIF LVDS data transmission functionality for data rates  $\leq 355$  MBaud. For higher data rates ( $> 355 \dots \leq 400$  MBaud) the external clock skew must not be higher than  $\pm 450$  ps.

### **2.65 [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.

### **2.66 [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. This statement is not correct.

#### **Documentation update**

The clock alive monitor for  $f_{SPB}$  is visible to all interfaces in the SMU.

## **2 Functional deviations**

### **2.67 [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.

### **2.68 [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.

### **2.69 [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.

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.

## 2 Functional deviations

### 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.”

### 2.70 [SCR\_TC.015] Bit SCU\_PMCN1.WCAN\_DIS does not disable WCAN PCLK input

#### Description

Setting bit SCU\_PMCN1.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>).

### 2.71 [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.

### 2.72 [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

### 2.73 [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

### 2.74 [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.

### 2.75 [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).

## 2.76 [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.

## 2.77 [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

### 2.78 [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 7 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>

### 2.79 [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

### 2.80 [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 8** 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

### 2.81 [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.

## 2.82 [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).

## 2.83 [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

### 2.84 [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.

### 2.85 [SPU\_TC.019] ACCEN0 register description - Correction

#### Description

The register description of the ACCEN0.ENx field in the SPU and SPULCKSTP chapter states for default setting 0<sub>B</sub>:

- “RO , Write access will terminate without error but will not be executed”

This is incorrect since a write access in this case will result in Bus Error.

#### Correction

The correct description of the ACCEN0.ENx field in the SPU and SPULCKSTP modules for default setting 0<sub>B</sub> is:

- “RO , Write access will terminate **with** error and will not be executed”

### 2.86 [SPU\_TC.020] SPU Power Sensitivity to IDM\_RM\_IOLR.ILR setting

#### Description

When reading data from Radar Memory, if the Inner Loop repeat value is not wholly divisible by the nominal number of datasets<sup>3)</sup> per datablock<sup>4)</sup>, the SPU processing activity becomes uneven and this can lead to repetitive I<sub>DD</sub> load jumps during the execution of an SPU configuration.

For example, if IDM\_RM\_CONF.FORMAT is set to CMPLX32BIT and IDM\_RM\_IOLR.ILR is set to 0x15, then the Inner Loop consists of 22 datasets. The SPU breaks these datasets up into 5 datablocks containing 4 datasets each plus one datablock containing 2 data sets. This leads to an unbalanced pipeline in the SPU. While the ODM is processing datablock no. 5, containing 4 datasets, the FFT engine is processing datablock 6 which contains 2 datasets.

<sup>3)</sup> The term dataset is synonymous with a RAMP of samples for a given antenna, or with an FFT output set.

<sup>4)</sup> The term datablock refers to the contents of an internal SPU buffer RAM, which comprises of a number of datasets, and is the unit of data upon which one of the SPU processing stages operates at a time.

## 2 Functional deviations

The ODM and FFT nominally process data at the same rate, so the FFT completes the processing of datablock 6 in half the time that the ODM processes datablock 5 and is consequently stalled waiting for the ODM to complete. Then, once the ODM has completed dataset 5, the FFT starts processing dataset 1 of the next Outer Loop Increment.

The FFT engine is the largest current consumer within the SPU so this break in FFT processing, followed by the restart, gives rise to a  $I_{DD}$  load jump similar to that seen at the start of execution of an SPU configuration (The point at which data sheet measurements were taken). There are, additionally, two factors which need to be taken into consideration.

1. The time taken to reach the final current consumption will be shorter compared to the current ramp at the start of a measurement cycle (See Note 1 below)
2. The frequency of the load jumps will be higher compared to the frequency of the load jumps occurring at the beginning and end of a measurement cycle. This can conceivably cause interaction with the components of an external power supply (See Note 2 below) making it difficult to measure the exact magnitude of the current change occurring with the TC3x device

### Note 1

The FFT storage contains zero data at reset and is flushed with zero data at the end of a measurement cycle. This means that, at the start of measurement cycle, the FFT storage is always filled with zero data. Until non-zero data is loaded into the FFT pipeline, zero data is applied to the FFT multipliers and FFT RAM ECC encoders and decoders, giving rise to a relatively slow ramp (a clock cycle count of approximately twice the FFT length) of the current drawn by the FFT as the non-zero data propagates through the FFT pipeline and the activity at the RAM interfaces and FFT multipliers increases.

In the case of  $I_{DD}$  load jumps created by an unbalanced pipeline, the FFT storage is not initialised. This is because the stall occurs in the middle of execution of the configuration so the FFT pipeline is not flushed with zero data. The FFT therefore restarts immediately with the higher current consumption associated with real data in the FFT pipeline.

### Note 2

With random data,  $I_{DD}$  load jumps of 500 mA have been measured for a single SPU with the TC3x being powered from an external bench supply.

### Workaround

The issue described above can be avoided by ensuring that the following constraints are adhered to:

- If `IDM_RM_CONF.FORMAT` is set to `CMPLX32BIT` and `IDM_RM_IOLR.ILR + 1 > 4` then `IDM_RM_IOLR.ILR + 1` should be wholly divisible by 4
- For all other settings of `IDM_RM_CONF.FORMAT`, if `IDM_RM_IOLR.ILR + 1 > 8` then `IDM_RM_IOLR.ILR + 1` should be wholly divisible by 8

## 2.87 [SPU\_TC.023] Write to register `SPU_STAT`: use only 32-bit writes

### Description

When using data types other than “32-bit word” to write to the lower half-word (bits [15:0]) of register `SPU_STAT`, an incorrect byte lane enable signal may get selected.

### Workaround

Only use a 32-bit word write or 32-bit word read-modify-write (RMW) instruction to write to register `SPU_STAT`.

## **2 Functional deviations**

### **2.88 [SPU\_TC.024] Register CRC: assume PACTR.RST bit as 0<sub>B</sub>- Documentation Update**

#### **Description**

Section “Register CRC” in the SPU chapter of the TC3xx User’s Manual contains the following description:

#### **Register CRC**

“The register CRC uses a 32 bit ethernet polynomial which assumes that the register contents are converted to an LSB first sequential data stream. The data stream will be generated using the registers from ID\_CONF to CTRL inclusive. Unused register addresses, and the address of the REGRCRC register, itself shall be replaced by 0000 0000<sub>H</sub> in the datastream. Additionally, the CTRL.TRIG bit and CTRL.BUSY bit should be assumed to be 0<sub>B</sub> (value when read) and the value of the PACTR.COUNT bitfield should be assumed to be 0<sub>D</sub> for calculating the reference CRC.”

The last sentence in this paragraph is missing the write-only bit PACTR.RST, that also has to be given a value of zero when computing the reference CRC for example from configuration sets stored in SPU configuration memory.

#### **Documentation Update**

Therefore, the documentation in section “Register CRC” in the SPU chapter shall be updated to include bit PACTR.RST as follows:

#### **Register CRC**

“The register CRC uses a 32 bit ethernet polynomial which assumes that the register contents are converted to an LSB first sequential data stream. The data stream will be generated using the registers from ID\_CONF to CTRL inclusive. Unused register addresses, and the address of the REGRCRC register itself, shall be replaced by 0000 0000<sub>H</sub> in the datastream. Additionally, the CTRL.TRIG bit, CTRL.BUSY **and PACTR.RST** bit should be assumed to be 0<sub>B</sub> (value when read) and the value of the PACTR.COUNT bit-field should be assumed to be 0<sub>D</sub> for calculating the reference CRC.”

### 3 Parametric deviations

## 3 Parametric deviations

### 3.1 [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.

### 3.2 [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

- $t_{\text{MERP}} \leq 0.52 \text{ s}$  (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:**  $t_{\text{ER\_Dev}} \leq 19.14 \text{ s}$  (instead of 18.5 s as specified in the Data Sheet)
- **TC38x:**  $t_{\text{ER\_Dev}} \leq 11.9 \text{ s}$  (instead of 11.5 s as specified in the Data Sheet)
- **TC3Ex:**  $t_{\text{ER\_Dev}} \leq 14.98 \text{ s}$  (instead of 14.5 s as specified in the Data Sheet)
- **TC37x, TC37xEXT:**  $t_{\text{ER\_Dev}} \leq 7.24 \text{ s}$  (instead of 7 s as specified in the Data Sheet)
- **TC35x, TC36x:**  $t_{\text{ER\_Dev}} \leq 5.16 \text{ s}$  (instead of 5 s as specified in the Data Sheet)
- **TC33xEXT, TC33x/TC32x:**  $t_{\text{ER\_Dev}} \leq 3.08 \text{ s}$  (instead of 3 s as specified in the Data Sheet)

The increased values should be considered for example when defining erase timeout limits.

### 3.3 [PADS\_TC.P014] Electrical characteristics for P20.2/TESTMODE

#### Description

The buffer type for P20.2/TESTMODE is defined in column “Buffer Type” of table “Port 20 Functions” in the Data Sheet as “class S/PU / VEXT”.

The electrical characteristics specified in table “Class S 5V” and “Class S 3.3V” (if present in the Data Sheet) literally only apply to class S pads on P40 (and P41 if available), as they are connected to  $V_{\text{DDM}}$ .

#### Recommendation

Tables “Class S 5V” and “Class S 3.3V” apply analogously to the electrical characteristics of P20.2/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 Parametric deviations

#### 3.4 [PORST\_TC.P002] $V_{IH}$ and $V_{IL}$ definition for PORST pad - Additional Data Sheet footnote

##### Description

The following footnote shall be added in column “Note/Test Condition” of Data Sheet table “PORST Pad” to the parameters “Input high voltage level” (symbol  $V_{IH}$ ) and “Input low voltage level” (symbol  $V_{IL}$ ):

**Note:** *The levels defined are valid within the operating conditions of  $V_{EXT} = 5\text{ V} \pm 10\%$  or  $V_{EXT} = 3.3\text{ V} \pm 10\%$ , 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):
    - $I_{BWG} = V_{AIN} * C_{AINS} * CR$
  - Broken Wire Detection against  $V_{AREF}$  (BWR):
    - $I_{BWR} = (V_{AIN} - V_{AREF}) * C_{AINS} * CR$

## 4 Application hints

where

- $V_{AIN}$ : ADC input voltage at the detection threshold (typ. 10% of full scale for BWD, 80% of full scale for BWR)
- $C_{AINS}$ : ADC input sampling capacitance, typ. 2.5 pF
- $CR$ : Conversion Rate, that is number of conversions per second per input

### Recommendation

The absolute value of the Broken Wire Detection current ( $I_{BWG}$  or  $I_{BWR}$ ) at the BWD threshold shall be at least 2x the maximum input leakage current  $I_{OZ}$  (absolute value).

### Examples

#### 1. Typical case example

Assuming that  $T_J \leq 150^\circ\text{C}$  (max) and ADC inputs are used in a configuration where  $I_{OZ} \leq 150$  nA (see Data Sheet),  $I_{BWG}$  should be  $\geq 300$  nA according to the recommendation above.

With  $C_{AINS} = 2.5$  pF and  $V_{AIN} = 0.5$  V (10% of full scale) it can be calculated from the formula for  $I_{BWG}$  above that  $CR$  should be  $\geq 240\,000$  samples per second and input.

#### 2. Worst case example

Assuming that  $T_J \leq 170^\circ\text{C}$  (max) and ADC inputs are used in a configuration where  $I_{OZ} \leq 800$  nA (see Data Sheet),  $I_{BWG}$  should be  $\geq 1600$  nA according to the recommendation above.

With  $C_{AINS} = 2.5$  pF and  $V_{AIN} = 0.5$  V (10% of full scale) it can be calculated from the formula for  $I_{BWG}$  above that  $CR$  should be  $\geq 1\,280\,000$  samples per second and input.

### Recommendations for increasing the Broken Wire Detection current

In order to increase the Broken Wire Detection current,

1. Relax the detection threshold, for example for BWG from 10% to 20% of the full scale voltage
2. Increase the conversion rate  $CR$  per input by introducing additional conversions

## 4.5 [ADC\_TC.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 (GLOBCFG.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  $t_{SUCAL}$ :

- <sup>2)</sup> In the synchronized mode ( $USC=0$ ),  $t_{SUCAL}$  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  $V_{AGND}$ ) or to charge (BWD against  $V_{AREF}$ ) the input node of the EVADC. The level ( $V_{AREF}$  or  $V_{AGND}$ ) is selected in the bit-field BWDCH as documented in the description of register GxCHCTRY in the EVADC chapter of the TC3xx user manual.

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

### **Documentation update**

The description related to broken wire detection (BWD) in the following parts of the EVADC chapter shall be updated to reflect both BWD against  $V_{AGND}$  and BWD against  $V_{AREF}$ :

- Section "Safety Features"  
Broken wire detection (BWD) preloads the converter network with a selectable level before sampling the input channel. The result will then reflect the preload value if the input signal is no more connected
- Chapter "Broken Wire Detection"  
To test the proper connection of an external analog sensor to its input pin, the converter's capacitor can be precharged to a selectable value before the regular sample phase. If the connection to the sensor is interrupted, the subsequent conversion value will rather represent the precharged value than the expected sensor result. By using a precharge voltage outside the expected result range (broken wire detection uses  $V_{AGND}$  or  $V_{AREF}$ ) a valid measurement (sensor connected) can be distinguished from a failure (sensor detached)  
Broken wire detection can be enabled for each channel separately by the bit-field 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 Tx FIFO single move mode**

As described in section “Single Move Mode” of chapter “TxFIFO interrupt generation” in the user manual, the purpose of the Single Move Mode is to keep the Tx FIFO as full as possible, refilling the Tx FIFO by writing to it as soon as there is a free element. The single move mode supports primarily a DMA operation using single move per Tx FIFO interrupt.

See also the note at the end of this section in the user manual.

"In Single Move Mode multiple software writes or block DMA moves would lead to multiple interrupts and (false) transaction lost events. Therefore they should be avoided - only single moves should be used."

To complement the above description, the following two sentences shall be added to the section before the reference to figure “Interrupt generation in the single move mode” in the user manual.

## **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.H020] Processing in case no valid BMHD found

#### Description

The section "Processing in case no valid BMHD found" in the Firmware chapter of the TC3xx user manual states in step 4 as follows:

- 4. instal the address with offset 0x0020 in logical sector S40 in PFLASH0 (I.e. 0xA000A020) as user code start address into BOOT\_ADDR

The absolute address specified in the above sentence is incorrect. It must be 0xA00A0020 instead of 0xA000A020.

#### Documentation update

The absolute address specified in step 4 in the section “Processing in case no valid BMHD found” in the Firmware chapter of the TC3xx user manual shall be corrected as follows:

- 4. Install the address with offset 0x0020 in logical sector S40 in PFLASH0 (that is 0xA00A0020) as user code start address into BOOT\_ADDR

### 4.19 [CCU6\_TC.H001] CCU6 module clock source information - Documentation Update

#### Description

In the CCU6 module chapter of the AURIX™ TC3xx User’s manual, the CCU6 module clock source information is missing.

#### Documentation update

The CCU6 module is clocked with the SPB clock, so  $f_{CC6} = f_{SPB}$ .

See also figures “Clocking System example” in the Clocking System chapter of the TC3xx User’s manual, where the CCU6 module is connected to  $f_{SPB}$ .

### 4.20 [CCU\_TC.H012] Configuration of the Oscillator- Documentation Update

#### Description

As described in chapter „Configuration of the Oscillator” in the CCU chapter of the User’s Manual, configuration of the oscillator is always required before an external crystal / ceramic resonator can be used as clock source.

Depending on the supply voltage ramp-up characteristics the behavior described in the following note may be observed:

## 4 Application hints

**Note:** *If VEXT is present then the oscillator could start oscillating (crystal/resonator connected). As soon as Cold PORST of AURIX™ is released, the oscillator is set to External Input Mode and the oscillation decays. This characteristic behavior has no impact on the oscillator start-up as initiated by software.*

### 4.21 [CCU\_TC.H017] XTAL1 input signal compatibility with Infineon CTRX81xx devices

#### Description

Irrespective of the related parameters in the datasheet, the input XTAL1 can also accept an external 25 MHz clock on the XTAL1 input with characteristics that satisfy the following conditions:

- Input voltage at XTAL1:  $2.5\text{ V} \leq V_{IX} \leq 3.3\text{ V} + 5\%$
- Slew Rate at XTAL1:  $SR_{XTAL1} \geq 0.15\text{ V/ns}$
- Duty Cycle of waveform at XTAL1:  $0.4 \leq DC_{X1} \leq 0.6$

Where, slew rate is the only parameter that is extended compared to the original datasheet values.

**Note:** *In the systems incorporating one or more CTRX81xx devices, the I/O voltages of the microcontroller will be limited to a maximum of 3.3 V +5% as this is the maximum I/O voltage supported by the CTRX81xx devices.*

#### Scope

XTAL1 input signal compatibility with output clock provided by Infineon CTRX81xx devices.

#### Effects

The described parameters allow interoperability with the CTRX81xx MMICs which have altered the parameters for the clock output compared to the previous RXS81xx product generation.

### 4.22 [CCU\_TC.H018] Encoding of bit-field ADASDIV in register CCUCON5

#### Description

In the description of register CCUCON5 in the "Clocking System" chapter of the TC3xx user manual, the resulting frequency for setting  $ADASDIV = E_H$  is incorrectly documented as  $f_{ADAS} = f_{SOURCE0}/12$ .

#### Documentation update

The resulting frequency for setting  $CCUCON5.ADASDIV = E_H$  is  $f_{ADAS} = f_{SOURCE0}/14$ .

### 4.23 [CLC\_TC.H001] Description alignment for bits DISR, DISS, EDIS in register CLC - Documentation Update

#### Description

For the description of bits DISR, DISS, and EDIS (if available) in register CLC (and CLC1 for I2C), different styles are used in the current version of the TC3xx user manual.

For the following modules, the function of these bits depending on their status ( $0_B$  or  $1_B$ ) is not explicitly described:

- ASCLIN, CIF, E-RAY, FCE, GETH, GTM, HSPDM, HSSL (incl. HSCT), I2C, MCMCAN, MSC, PSI5, PSI5-S, QSPI, SDMMC, SENT, STM

For these modules, the missing parts of the bit description can be taken from the following general description:

## 4 Application hints

**Table 9** 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>	1	rh	<b>Module Disable Status Bit</b> 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

## 4.24 [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.

## 4 Application hints

### 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)
- 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 10**      **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)

## 4 Application hints

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

**Table 11** Example 1b: Access order is enforced by correct semaphore handling

CPUx	CPUy	Memory Access Sequence
st-1 (resource-update)	CMPSWAP.W (semaphore-acquire)	
DSYNC	DSYNC	
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*

## 4.25 [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 12** 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

**(table continues...)**

## 4 Application hints

**Table 12 (continued) Possible SW/HW interaction producing incorrect state**

CPUx	CPUy	Incorrect resource access sequence
...	Semaphore acquire	CPUy – Semaphore acquire
...	Resource read (load instruction)	CPUy – Resource read (incorrect state)
...	...	CPUx - Pending FPI write succeeds (Resource update completed)

### 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 (CPUx) before execution of the DSYNC instruction.

Hence, the modified sequence of CPUx for the example given above must be:

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

## 4.26 [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 Application hints

### 4.27 [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.

#### 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.

### 4.28 [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<sup>CBLS</sup> or 2<sup>CBLD</sup> 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<sup>CBLS</sup> or 2<sup>CBLD</sup> 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 [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

### 4.31 [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).

### 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.

#### 4 Application hints

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.

### 4.32 [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.*

### 4.33 [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.*

### 4.34 [FLASH\_TC.H024] PFLASH erase and program time is affected by time slicing but not clearly documented

#### Description

In the NVM chapter in the TC3xx user manual, the sub-section "Time Slice Control and Flash Parameters" describes the erase and program time increase due to time slicing. It is mentioned only for DFLASH in the user manual. However, this is applicable for PFLASH also.

#### Documentation update

The timing penalty of time slicing described in the sub-section "Time Slice Control and Flash Parameters" of the NVM chapter in the TC3xx user manual is applicable not only for DFLASH but it is also applicable for PFLASH operations when running concurrently to HSM operations.

## 4 Application hints

### 4.35 [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

#### Recommendation

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

### 4.36 [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 Application hints**

### **4.37 [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.

#### **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”.

### **4.38 [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

### **4.39 [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).*

## 4 Application hints

### 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)
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

## 4.40 [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 Application hints

### 4.41 [HSPDM\_TC.H001] Accesses to specific HSPDM registers while 160/320 MHz clocks are disabled

#### Description

Accesses to the ADCTG, ADCTGCNT and CON registers will not complete if the 160/320 MHz clocks have been disabled (bit HSPDMPERON=0<sub>B</sub> in register CCUCON2). No bus error will be generated in this case, and the CPU/DMA gets stuck issuing retries indefinitely.

**Note:** After reset, bit CCUCON2.HSPDMPERON=1<sub>B</sub>, i.e. the 160/320 MHz clocks are enabled.

#### Recommendation

Ensure the 160/320 MHz clocks are enabled (CCUCON2.HSPDMPERON=1<sub>B</sub>) before accessing the ADCTG, ADCTGCNT and CON registers.

### 4.42 [HSPDM\_TC.H002] Gap between stop/start of bit streaming

#### Description

When bit streaming is running and software writes to HSPDM\_CON in two consecutive accesses to stop and then start the bit streaming, the software-initiated start may not take effect.

#### Recommendation

Software should check for the status HSPDM\_CON.RUN = 0<sub>B</sub> before writing to HSPDM\_CON to start bit streaming.

### 4.43 [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 13](#) below.

The following statement in the IR chapter “External Interrupts” of the TC3xx User’s Manual is conflicting with the description above:

- “Eight SRNs (Int\_SCUSRC[7:0]) are reserved to handle external interrupts.”

#### Documentation update

The statement in the IR chapter shall be changed as follows:

- “**Four** SRNs (SRC\_SCUERUx (x=0-3)) are implemented to handle external interrupts.”

Table “OGU to SRC connection” in the SCU chapter shall be changed as follows:

**Table 13 OGU to SRC connection**

OGUy.ERU_IOUTy (OGU I-output signal)	SRC_SCUERUx (interrupt SRC register)
OGU0.ERU_IOUT0, OGU4.ERU_IOUT4	SRC_SCUERU0

(table continues...)

## 4 Application hints

**Table 13** (continued) OGU to SRC connection

OGUy.ERU_IOUTy (OGU I-output signal)	SRC_SCUERUx (interrupt SRC register)
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.

### 4.44 [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).

### 4.45 [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.

### 4.46 [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

## **4.47 [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  $I_{DDMBIST}$ ).

## **4.48 [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**

### **4.49 [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.

### **4.50 [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.

### **4.51 [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.

### **4.52 [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.

### **4.53 [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, NTBTTri, NTCTTri

set bit-field RELOAD to one value less than the calculated counter value.

## 4.54 [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 14** Operating Conditions - CAN Frequency: symbol update

Parameter	Symbol	Values			Unit	Note / Test Condition
		Min.	Typ.	Max.		
CAN Frequency	$f_{\text{MCAN}}$ SR	-	-	80	MHz	

## 4.55 [MTU\_TC.H015] ALM7[0] may be triggered after cold PORST

### Description

During firmware start-up after cold PORST, alarm status flag AG7.SF0 (correctable SRAM error) may erroneously be set to 1, although no error occurred. This is due to a dummy read to an uninitialized SRAM by firmware.

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

### Recommendation

As alarms for correctable errors are uncritical in general, no action is required (alarm can be ignored). The application may only react on the error overflow.

In addition, to ensure that SMU alarm ALM7[0] does not correspond to a real SRAM correctable error, the user may refer to the ESM MCU\_FW\_CHECK described in the Safety Manual.

#### 4 Application hints

### 4.56 [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].

### 4.57 [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.

### 4.58 [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.

#### Documentation Update

The term DMU\_HP\_PROCONTTP must be replaced by the correct register name DMU\_HF\_PROCONTTP 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 Application hints

### 4.59 [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.

### 4.60 [OCDS\_TC.H015] System or Application Reset while OCDS and lockstep monitoring are enabled

#### Description

After a System or Application Reset the Lockstep Alarm ALMx[0] gets activated if all of the following conditions are met (x = index of CPU with checker core):

1. Lockstep monitoring is enabled by BMI.LSENAX = 1<sub>B</sub> for CPUx, AND
2. Debug System is enabled (CBS\_OSTATE.OEN = 1<sub>B</sub>), AND
3. a) CPUx is halted (either in boot-halt state or stopped by debugger tool or in idle mode) when reset is triggered, OR b) CPUx Performance Counters are enabled and CPUx Clock Cycle Count register CCNT is read

#### Recommendation

To avoid the unintended ALMx[0] under the conditions described above, either:

- Keep the debug system disabled. OR
- Ensure all CPUs that have lockstep monitoring enabled are out of halted state AND CPUx Performance Counters are disabled before executing a System or Application reset. OR
- Use PORST instead of a System or Application reset

#### 4 Application hints

### 4.61 [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.

### 4.62 [OCDS\_TC.H018] Unexpected stop of Startup Software after system or application reset

#### Description

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

As documented in the TriCore™ Architecture Manual, the settings in the Debug Status Register (DBGSR) are only cleared upon a debug or power-on reset. This may lead to unexpected behavior in the following scenario:

If CPU0 is in HALT mode, and a system or application reset is triggered, the Startup Software (SSW) starts execution on CPU0, but it is stopped again (due to the settings in DBGSR) before the SSW has finished the boot procedure.

#### Recommendation

The tool should switch the device from HALT to RUN mode through the DBGSR register.

Alternatively, a power-on reset may be performed instead of a system or application reset.

### 4.63 [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  $K_{OVP}$  and  $K_{OVN}$  to the neighbor LVDS pad are relatively large (see table “Overload Parameters” in the Data Sheet).

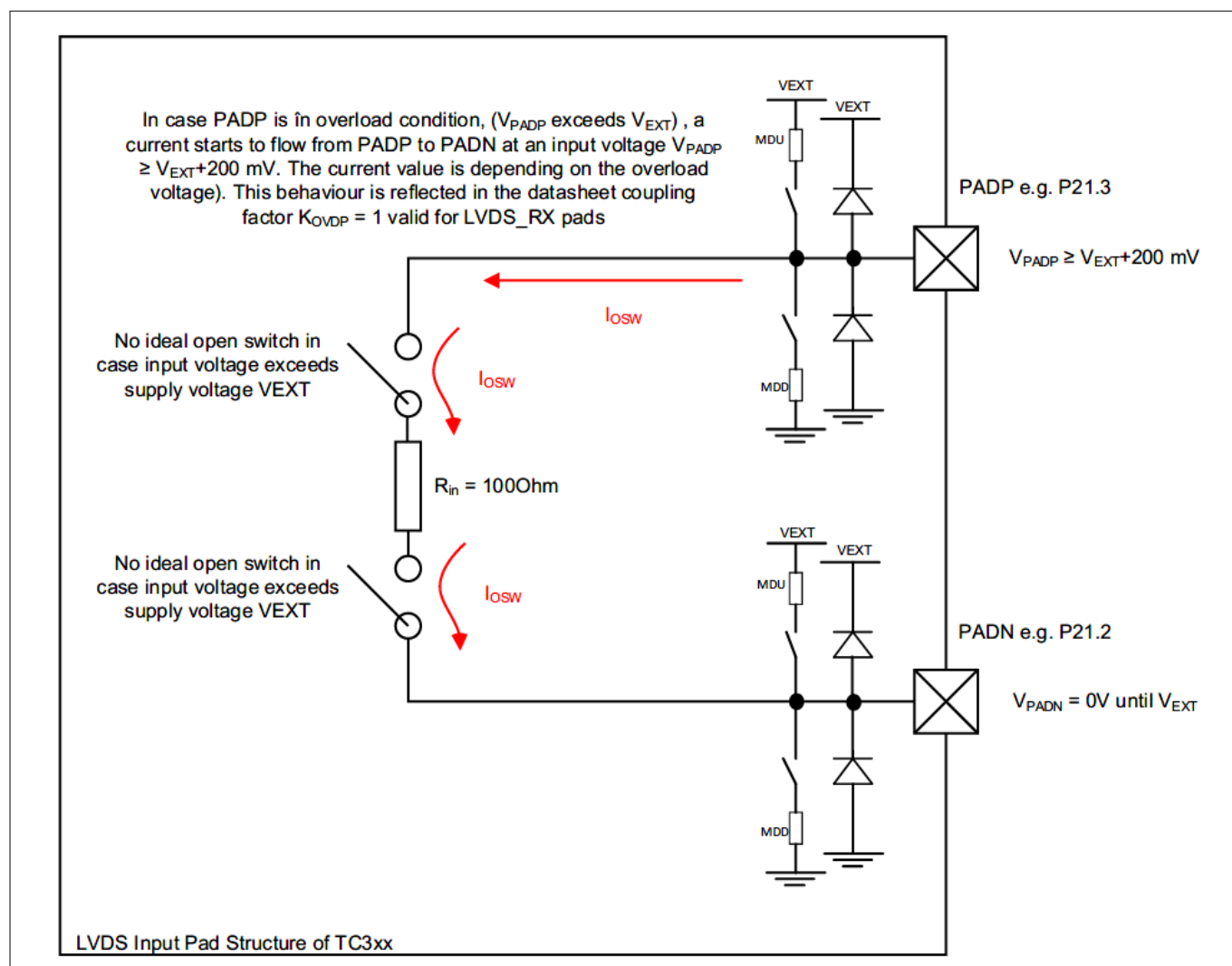
**Note:** *In this context, the term “neighbor LVDS pad” of a P-pad (N-pad) refers to the corresponding N-pad (P-pad) within the same P-/N-pad pair. If available on the package and implemented as buffer type LVDS\_RX, these are P14.9/10, P21.0/1, P21.2/3.*

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  $V_{PADP} \geq V_{EXT} + 200 \text{ mV}$ .



## 4 Application hints



**Figure 3** 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.

## 4.64 [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 - VEVRB, 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

## 4.65 [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 WDTCPUXSR.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.

## 4.66 [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.

### **4.67 [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

### **4.68 [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.

### **4.69 [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

### **4.70 [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.

### **4.71 [QSPI\_TC.H008] Details of the baud rate and phase duration control - Documentation update**

#### **Description**

To enhance readability, the last part of the second paragraph in the QSPI chapter "Details of the Baud Rate and Phase Duration Control", starting with "Variations in the baud rates of the slaves ..", shall be rephrased as shown below.

For further details see also the formulas in the chapter mentioned above and in the figures in chapter "Calculation of the Baud Rates and the Delays" in the User's Manual.

#### **Documentation update**

Variations in the baud rates of slaves of one module are supported by the ECONz.Q and the ECONz.A/B/C bit-field settings allowing for a flexible bit time variation between the channels in one module.

### **4.72 [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.

#### 4 Application hints

**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 15 TC39x, TC38x, TC37x, TC37xEXT, TC36x, TC35x, TC33xEXT and TC3Ex registers that may have different reset values than documented in TC3xx User's Manual**

Register	Initial reset value	Reset value defined in User's Manual	Remark
P20_IOCRO	0x0010 0000	0x0000 0000 (HWCFG6 = tri-state)	TESTMODE pin is PU (input pull-up), even with HWCFG6 = tri-state, as described in Data Sheet.
EMEM_TILECONFIG	0x0000 0000	0x5555 5555	This is a write-only ("w") register. For tile mode information do not read EMEM_TILECONFIG; instead, read EMEM_TILESTATE.
SBCU_DBCNTL	0x0000 7002	0x0000 7003	Bit EO is "Status of BCU Debug Support Enable" and only set after reset when OCDS is enabled. This bit is controlled by Cerberus.
CPU1_PCON1	0x0000 0001	0x0000 0000	Bit PCINV of PCON1 is set when CPU is in boot halt mode, it is cleared when CPU starts execution
CPU2_PCON1	0x0000 0001	0x0000 0000	
CPU3_PCON1	0x0000 0001	0x0000 0000	
CPU4_PCON1	0x0000 0001	0x0000 0000	
CPU5_PCON1	0x0000 0001	0x0000 0000	
HSSL0_MFLAGS	0xA000 0000	0x8000 0000	Bit TEI indicates the state of CTS (Clear To Send) signal from HSCT module. The default state of this bit is 1.
HF_OPERATION	0x0000 0X00	0x0000 0000	RES bits shall be ignored.
PMS_EVRSDCTRL0	0x3039 0001	0xF039 0001	LCK and UP bits are cleared.
PMS_EVRSDCTRL1	0x0669 0708	0x8669 0708	LCK bit is cleared.
PMS_EVRSDCTRL6	0x0023 1C94	0x8023 1C94	LCK bit is cleared.
PMS_EVRSDCTRL7	0x0000 00FE	0x8000 00FE	LCK bit is cleared.
PMS_EVRSDCTRL8	0x1121 048E	0x9121 048E	LCK bit is cleared.
PMS_EVRSDCTRL9	0x0000 0434	0x8000 0434	LCK bit is cleared.
PMS_EVRSDCTRL11	0x1207 0909	0x9207 0909	LCK bit is cleared.
PMS_EVRSDCOEFF0	0x3508 73B6	0xB508 73B6	LCK bit is cleared.
PMS_EVRSDCOEFF1	0x2294 6C46	0xA294 6C46	LCK bit is cleared.
PMS_EVRSDCOEFF6	0x0097 1802	0x8097 1802	LCK bit is cleared.
PMS_EVRSDCOEFF7	0x0000 D8F7	0x8000 D8F7	LCK bit is cleared.
PMS_EVRSDCOEFF8	0x0017 1002	0x8017 1002	LCK bit is cleared.
PMS_EVRSDCOEFF9	0x0000 A0AF	0x8000 A0AF	LCK bit is cleared.

**(table continues...)**



#### 4 Application hints

**Table 15** (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
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 16** TC33x/32x registers that may have different reset values than documented in TC3xx User's Manual

Register	Initial reset value	Reset value defined in User's Manual	Remark
P20_IOCRO	0x0010 0000	0x0000 0000 (HWCFG6 = tri-state)	TESTMODE pin is PU (input pull-up), even with HWCFG6 = tri-state, as described in Data Sheet.
SBCU_DBCNTL	0x0000 7002	0x0000 7003	Bit EO is "Status of BCU Debug Support Enable" and only set after reset when OCDS is enabled. This bit is controlled by Cerberus.
HF_OPERATION	0x0000 0X00	0x0000 0000	RES bits shall be ignored.
PMS_EVRSDCTRL0	0x0000 0003	0xC000 0003	LCK and UP bits are cleared.
PMS_EVRSDCTRL1	0x0012 0012	0x8012 0012	LCK bit is cleared.
PMS_EVRSDCTRL2	0x140A 0909	0x940A 0909	LCK bit is cleared.
PMS_EVRSDCTRL3	0x0000 0001	0x8000 0001	LCK bit is cleared.
PMS_EVRSDCTRL4	0x2000 2209	0xA000 2209	LCK bit is cleared.
PMS_EVRSDCTRL5	0x001B 7566	0x801B 7566	LCK bit is cleared.
PMS_EVRSDCTRL6	0x0081 0000	0x8081 0000	LCK bit is cleared.
PMS_EVRSDCTRL7	0x2061 0400	0xA061 0400	LCK bit is cleared.
PMS_EVRSDCTRL8	0x1070 0000	0x9070 0000	LCK bit is cleared.
PMS_EVRSDCTRL9	0x0000 4040	0x8000 4040	LCK bit is cleared.
PMS_EVRSDCTRL10	0x130C 0719	0x930C 0719	LCK bit is cleared.
PMS_EVRSDCOEFF0	0x0052 0083	0x8052 0083	LCK bit is cleared.
PMS_EVRSDCOEFF1	0x17B2 6996	0x97B2 6996	LCK bit is cleared.
PMS_EVRSDCOEFF2	0x4924 8BD9	0xC924 8BD9	LCK bit is cleared.
PMS_EVRSDCOEFF3	0x0780 00A3	0x8780 00A3	LCK bit is cleared.
SCU_OSCCON	0x0000 0258 for UCB_DFLASH.OSCCFG = 0; 0x0XX0 XXXX otherwise	0x0000 0X1X	SCU_OSCCFG setting is recovered from UCB_DFLASH



#### 4 Application hints

### 4.73 [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).

### 4.74 [RIF\_TC.H007] Initialization sequence for RIF

#### Description

During RIF initialization by software, the deserializer should always be initialized and enabled as the last step to avoid unintended data capture on the LVDS inputs due to possible level transitions on the FCLK signal.

### 4.75 [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)

### 4.76 [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:

#### **4 Application hints**

##### **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>).

##### **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**

#### **4 Application hints**

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.

##### **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 (RTy, where y = 0,1) via RTC.RTyE = 1<sub>B</sub>.  
Recovery Timers (RTy, where y = 0,1) are enabled after Application Reset to service the WDT timeout alarms.

##### **SM[HW]:SMU:FSP\_MONITOR**

###### **Init conditions**

## **4 Application hints**

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

## **4.77 [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.*

## **4.78 [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.

## **4.79 [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 Application hints**

### **4.80 [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)

### **4.81 [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.

### **4.82 [SCR\_TC.H012] Reaction in case of XRAM ECC Error**

#### **Description**

When the double-bit ECC reset is enabled via bit ECCRSTEN in register SCR\_RSTCON, and a RAM double-bit ECC error is detected, bit RSTST.ECCRST in register SCR\_RSTST is set, but no reset is performed.

#### **Recommendation**

The reset of the SCR module in case of a double-bit ECC error must be performed via software.

The following steps need to be done:

- Enable the double-bit ECC reset by setting bit ECCRSTEN in register SCR\_RSTCON to 1<sub>B</sub>
- Enable the RAM ECC Error for NMI generation by setting bit NMIRAMECC in register SCR\_NMICON to 1<sub>B</sub>

When a RAM double-bit ECC error is detected, an NMI to the TriCore™ is generated, and bit RSTST.ECCRST in register SCR\_RSTST is set.

The TriCore™ software first has to check the cause of the NMI wakeup by checking register SCR\_RSTST. If bit ECCRST is set, a double-bit ECC error has occurred. In this case, do the following steps:

- Fill the XRAM memory with 0
- Check whether an ECC error has occurred
- If no ECC error has occurred after filling the XRAM with 0, then:
  - Reload the contents of the XRAM
  - Perform a reset of the SCR module: Set bit SCRSTREQ in register PMSWCR4 to 1<sub>B</sub>

## 4 Application hints

### 4.83 [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).

### 4.84 [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  $\mu\text{A}$ .

### 4.85 [SCU\_TC.H020] Digital filter on ESRx pins - Documentation update

#### Description

As described in the SCU and PMS chapters of the TC3xx User's Manual, the input signals  $\overline{\text{ESR0}}$  /  $\overline{\text{ESR1}}$  can be filtered. The filter for  $\overline{\text{ESRx}}$  is enabled via bit  $\text{PMSWCR0.ESRxDFEN} = 1_B$  (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  $f_{SPB}$  should be noted:

**Note:** Pulses longer than 100 ns will always result in a trigger for  $f_{SPB} \geq 20$  MHz in RUN mode.

### 4.86 [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 Application hints

### 4.87 [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.

### 4.88 [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 17 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)

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.

### 4.89 [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.



## 4 Application hints

### 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.

### 4.90 [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 18 Field INP0 and INP1 in register EICRi (i=0-3) - Correction**

Field	Bits	Type	Description
<b>INP0</b>	14:12	rw	<b>Input Node Pointer</b> This bit-field determines the destination (output channel) for trigger event (2i) (if enabled by <b>EICRi.EIEN0</b> ). 100 <sub>B</sub> An event from input ETL 2i triggers output OGU4 (signal TR(2i) <b>4</b> ) 101 <sub>B</sub> An event from input ETL 2i triggers output OGU5 (signal TR(2i) <b>5</b> ) 110 <sub>B</sub> An event from input ETL 2i triggers output OGU6 (signal TR(2i) <b>6</b> ) 111 <sub>B</sub> An event from input ETL 2i triggers output OGU7 (signal TR(2i) <b>7</b> )
<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.

### 4.91 [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.

## 4 Application hints

### 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

## 4.92 [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>).

## 4.93 [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 19 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

(table continues...)

**4 Application hints**

**Table 19 (continued) SENT input pads and considered neighbors**

Device	Considered left neighbors		SENT input		Considered right neighbors	
			Pad	Channel		
<b>TC38x</b>	P15.0	P15.2	P15.4	11D	P15.1	P15.3
	P31.6	P31.7	P31.8	20C	P31.10	P31.9
	P31.8	P31.10	P31.9	21C	P31.12	P31.11
	P31.7	P31.8	P31.10	22C	P31.9	P31.12
	P02.13	P02.11	P02.12	23B	P02.4	P02.15
	P31.9	P31.12	P31.11	23C	P31.14	P31.13
	P31.10	P31.9	P31.12	24C	P31.11	P31.14
<b>TC3Ex</b>	P02.6	P02.7	P02.8	0C	P02.9	P02.10
	P02.4	P02.6	P02.7	1C	P02.8	P02.9
	P02.11	P02.4	P02.6	2C	P02.7	P02.8
	P15.0	P15.2	P15.4	11D	P15.1	P15.3
	P02.3	P02.11	P02.4	12B	P02.6	P02.7
	P02.1	P02.5	P02.3	13B	P02.11	P02.4
	P15.1	P15.3	P14.0	17D	P15.6	P15.7
<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

**(table continues...)**

#### 4 Application hints

**Table 19** (continued) SENT input pads and considered neighbors

Device	Considered left neighbors		SENT input		Considered right neighbors	
			Pad	Channel		
TC33x/TC32x	P02.8	P00.0	P00.1	0B	P00.2	P00.3
	P02.6	P02.6	P02.8	0C	P00.0	P00.1
	P00.0	P00.1	P00.2	1B	P00.3	P00.4
	P02.5	P02.6	P02.7	1C	P02.8	P00.0
	P02.4	P02.5	P02.6	2C	P02.7	P02.8
	P02.3	P02.4	P02.5	3C	P02.6	P02.7
	P33.4	P33.5	P33.6	4C	P33.7	P33.8

**Note:** The table above is sorted by SENT channel numbers in ascending order. The same sorting is also used in the tables below.

The following tables summarize the results of the  $V_{ILD}$  measurements of the SENT input pads potentially exceeding the  $V_{ILD}$  limits with different neighbor (2N/4N) and different edge strength/driver strength configurations.

- **VILD(DIST4N):**  $V_{ILD}$  measurements with four neighbor pads (two on the left and two on the right hand side of the SENT input) used in output mode alongside the SENT input pad on the pad frame
- **VILD(DIST2N):**  $V_{ILD}$  measurements with two neighbor pads (one on the left and one on the right hand side of the SENT input) used in output mode alongside the SENT input pad on the pad frame

**Table 20** 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)
TC39x	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
TC38x	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

(table continues...)

**4 Application hints**

**Table 20 (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)
<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
<b>TC36x</b>	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
<b>TC33xEXT</b>	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

**(table continues...)**

#### 4 Application hints

**Table 20 (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)
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 21 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
TC38x	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
TC3Ex	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

(table continues...)

**4 Application hints**

**Table 21 (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>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
<b>TC33xEXT</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:SENT3C	3C	P02.5	OK	OK
<b>TC33x/TC32x</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
	SENT:SENT4C	4C	P33.6	OK	OK



#### 4 Application hints

**Table 22 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 23 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
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 24 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 Application hints

### 4.94 [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]

### 4.95 [SMU\_TC.H010] Clearing individual SMU flags: use only 32-bit writes

#### Description

The SMU registers shall only be written via 32-bit word accesses (i.e. ST.W instruction), as mentioned in table “Registers Overview” of the SMU chapter in the User’s Manual.

If any other instruction such as LDMST or SWAPMSK.W is used to modify only a few bits in the 32-bit register, then this may have the effect of modifying/clearing unintended bits.

#### Recommendation (Examples in C Language)

- **Example 1:** To clear status flag SF2 in register AG0, use:
  - SMU\_AG0.U = 0x0000 0004;
- **Example 2:** To clear status flags EF2 in register RMEF and RMSTS, use:
  - SMU\_RMEF.U = 0xFFFF FFFB;
  - SMU\_RMSTS.U = 0xFFFF FFFB;

Here the <REGISTER>.U implies writing to the register as an unsigned integer, which normally results in a compiler translation into an ST.W instruction.

#### Safety Considerations

As long as software uses only 32-bit writes to the SMU registers, there is no risk of malfunction.

In case the software does not use 32-bit writes (and for example uses bit-wise operations such as LDMST instructions instead) – then potentially unintended flags may be written and modified in the SMU registers. Depending on the application, this may potentially have an impact on safety and/or diagnostics.

**Note:** *The SMU reaction itself (for example alarm action triggering) is not affected even if the software unintentionally clears additional bits by not using a 32-bit write as recommended.*

### 4.96 [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).

## **4 Application hints**

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

### **4.97 [SMU\_TC.H013] Increased Fault Detection for SMU Bus Interface (SMU\_CLC Register)**

#### **Description**

Transient faults can possibly affect the SMU\_CLC register and lead to disabling the SMU\_core. This unintended switching off of SMU\_core cannot be detected if the FSP protocol is not used at all or used in FSP bi-stable mode.

#### **Recommendation**

In order to increase the capability of the microcontroller to detect such faults it is recommended to:

- Option 1:  
Use FSP Dynamic dual-rail or Time-switching protocol only, don't use FSP bi-stable protocol
- Option 2:  
In case FSP protocol is not used at all or Recommendation Option 1 is not possible, the [Application SW] shall read periodically, once per FTTI, the SMU\_CLC register to react on unintended disabled SMU

### **4.98 [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 Application hints

### 4.99 [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.

### 4.100 [SMU\_TC.H019] TC33xEXT: Incorrect ALM1[0] and ALM1[2] descriptions

#### Description

The TC33xEXT specific alarm mapping table in the TC33xEXT appendix to the TC3xx user manual states that CPU1 has a lockstep comparator error associated to ALM1[0] and a PFLASH read path lockstep error associated to ALM1[2]. This is incorrect, and purely a documentation error. CPU1 does not have an associated PFLASH. There is no lockstep, no PFLASH read path lockstep, therefore there are no associated alarms, and no possibility to test using the SCU\_LCLTEST register.

#### Scope

This problem is limited to the documentation.

#### Documentation update

The alarm table should be as follows:

**Table 25 Alarm mapping related to ALM1 group**

Alarm Index	Module	Safety Mechanism & Alarm Indication
ALM1[0]	Reserved	Reserved
ALM1[2]	Reserved	Reserved

### 4.101 [SPU\_TC.H011] In Place FFT with RIF data as input

#### Description

The In Place FFT mode of the SPU (BEx\_ODP\_CONF.IPF=1) can be used with RIF data as input as well as with the primary use case of EMEM data.

#### **4 Application hints**

When the In Place FFT mode is configured with ID\_RM\_CONF.PM=0 and ID\_RM\_CONF.TRNSPS=0, it is expected that the value for ID\_RM\_BLO.BLO is constrained to be the sample size of the data as configured in ID\_RM\_CONF.FORMAT. This causes the output data to be written to consecutive locations in EMEM. This constraint on the BLO field is explicit when reading data from EMEM. Please refer to section “Principles of Operation (Radar Memory)” in the SPU chapter of the TC3xx User’s Manual where this is stated in the second of the assumptions.

This assumption is hard coded into the implementation of the address generator which means that, if the constraint is violated and the value of ID\_RM\_BLO.BLO is not set to the sample size, it is possible for data to be written to EMEM locations which are not consecutive. The patterns can be determined using the following information

- If ID\_RM\_CONF.FORMAT is set to a data type with a size of 16 bits then the write address increment (in bytes) will be ID\_RM\_BLO.BLO x 16
- If ID\_RM\_CONF.FORMAT is set to a data type with a size of 32 bits then the write address increment (in bytes) will be ID\_RM\_BLO.BLO x 8
- If ID\_RM\_CONF.FORMAT is set to a data type with a size of 64 bits then the write address increment (in bytes) will be ID\_RM\_BLO.BLO x 4

#### **Recommendation**

If using values for ID\_RM\_BLO.BLO other than the sample size encoded in ID\_RM\_CONF.FORMAT, these values must be constrained so that

1. the write address increments by at least one 256-bit EMEM word
2. the write address increments by a whole number of 256-bit EMEM words

### **4.102 [SPU\_TC.H013] CFAR and Local Max function when spectrum extension is disabled - Additional information**

#### **Description**

Both the CFAR and Local Max function use the concept of a window on both the leading and lagging sides of the “cell under test” (CUT) which are used for comparison with the CUT either directly (in the case of the Local Max) or indirectly using a value derived from the window cells (in the case of the CFAR).

For example, for a CUT at index 7 of a results set and a window size of 2, the leading window would comprise of cells 5, 6 and the lagging window would comprise of cells 8, 9.

The behavior of interest occurs when the CUT is close to, or at, one of the limits of the results set. In these boundary conditions some, or all, of either the leading or lagging window will contain invalid data. The spectrum extension detects these conditions and ensures that cells of the windows always contain useful data. This behavior is described in the User Manual.

What is not currently described is the behavior when the spectrum extension function is disabled.

#### **Additional information**

When the spectrum extension function is disabled, both the CFAR and Local Max functions will detect the conditions where the CUT is close to, or at, the results boundaries and exclude the window cells that do not contain valid data from comparison with the CUT.

### **4.103 [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.*

## 4.104 [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.

**4.105 [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	2020-01-17	First version for TC33xEXT step AA
1.1	2020-03-31	Update: <ul style="list-style-type: none"> <li>New/updated text modules see column “Change” in tables 3..5 of errata sheet V1.1</li> <li>Removed: <ul style="list-style-type: none"> <li>CPU_TC.H016 (List of OS and I/O Privileged Instructions - Documentation Update): updated description in TriCore™ TC1.6.2 Core Architecture Manual V1.2.1, Vol. 2 Instruction Set, table 14</li> </ul> </li> </ul>
1.2	2020-07-06	Update: <ul style="list-style-type: none"> <li>New/updated text modules see column “Change” in tables 3..5 of errata sheet V1.2</li> </ul>
1.3	2020-10-23	Update: <ul style="list-style-type: none"> <li>New/updated text modules see column “Change” in tables 3..5 of errata sheet V1.3</li> <li>Text module SCR_TC.022 (Effect of application or system reset and warm PORST on MC77_ECCD and MC78_ECCD for SCR RAMs) moved from chapter “Application Hints” to chapter “Functional Problems”</li> </ul>
1.4	2021-01-22	Update: <ul style="list-style-type: none"> <li>New/updated text modules see column “Change” in tables 3..5 of errata sheet V1.4</li> </ul>
1.5	2021-04-22	Update: <ul style="list-style-type: none"> <li>New/updated text modules see column “Change” in tables 3..5 of errata sheet V1.5 <ul style="list-style-type: none"> <li>Text modules already published in TC3xx Errata Advance Information 2021_02: MCMCAN_AI.022, SMU_TC.H012</li> <li>Text modules already published in TC3xx Errata Advance Information 2021_03: EMEM_TC.H007, GETH_TC.H003, SCR_TC.023, SENT_TC.H007, SPU_TC.023</li> </ul> </li> <li>Removed SCU_TC.032, included description in update of SCU_TC.031 (Bits SCU_STSTAT.HWCFGx (x=1-5) could have an unexpected value in application if pins HWCFGx are left unconnected)</li> </ul>
1.6	2021-07-23	Update: <ul style="list-style-type: none"> <li>New/updated text modules see column “Change” in tables 3..5 of errata sheet V1.6 <ul style="list-style-type: none"> <li>Text modules already published in TC3xx Errata Advance Information 2021_06: FLASH_TC.055, MCMCAN_TC.H008, MTU_TC.018, PMS_TC.015</li> </ul> </li> </ul>
1.7	2021-11-04	Update: <ul style="list-style-type: none"> <li>New/updated text modules see column “Change” in tables 3..5 of errata sheet V1.7 <ul style="list-style-type: none"> <li>Text modules already published in TC3xx Errata Advance Information 2021-09: FLASH_TC.056, MCMCAN_AI.023, MEMMAP_TC.001, SAFETY_TC.023, SAFETY_TC.024</li> </ul> </li> </ul>

**Revision history**

Document version	Date of release	Description of changes
1.8	2022-03-25	<p>Update:</p> <ul style="list-style-type: none"> <li>New/updated text modules see column “Change” in tables 3..5 of errata sheet V1.8 <ul style="list-style-type: none"> <li>Text modules already published in TC3xx Errata Advance Information 2021-12: ASCLIN_TC.012, CCU_TC.005, 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, OCDS_TC.H015, SMU_TC.H016</li> </ul> </li> <li>Removed SCU_TC.H016 (RSTSTAT reset values - documentation update): updated in TC3xx User’s Manual V1.2.0 and following</li> </ul>
1.9	2022-07-29	<p>Update:</p> <ul style="list-style-type: none"> <li>Documentation reference changed to Safety Manual v2.0 (see table 1)</li> <li>New/updated text modules see column “Change” in tables 3..5 of errata sheet V1.9 <ul style="list-style-type: none"> <li>Text modules already published in TC3xx Errata Advance Information 2022-05: ADC_TC.H043, CPU_TC.H021, FLASH_TC.H021, MTU_TC.H016, SCU_TC.H025</li> <li>Text modules already published in TC3xx Errata Advance Information 2022-06: INT_TC.H006, LBIST_TC.H003, MCMCAN_AI.024, MCMCAN_AI.H002, SAFETY_TC.026, SAFETY_TC.027</li> </ul> </li> <li>Removed due to updates in Safety Manual v1.11 and following: <ul style="list-style-type: none"> <li>SAFETY_TC.004 (section 5.1 ESM[HW]:MCU:LBIST_MONITOR)</li> <li>SAFETY_TC.005 (section 5.116 SMC[SW]:SPU:BYPASS_CRC)</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> <li>SAFETY_TC.H012 (Appendix A for 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
1.10	2022-11-11	<p>Update:</p> <ul style="list-style-type: none"> <li>Documentation reference changed to TC3xx User's Manual V2.0.0 and TC3xx ED Target Specification V3.2.0 (see table 1)</li> <li>New/updated text modules see column "Change" in tables 3..5 of errata sheet V1.10 <ul style="list-style-type: none"> <li>Text modules already published in TC3xx Errata Advance Information 2022-08: 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, SPU_TC.H013</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 TC33xEXT AA Data Sheet V0.7 (and newer versions))</li> <li>ADC_TC.H034 (updated footnote on QCONV in table "VADC 5V" in TC33xEXT AA Data Sheet V0.7 (and newer versions))</li> <li>BROM_TC.H017 (corrected in footnote 1 to table "ALL CHECKS PASSED" indication by CHSW" in Firmware chapter of product specific Appendix V1.3.0 (and newer versions))</li> <li>CCU_TC.004 (updated in description of register OSCCON in Clocking System chapter of TC3xx UM V1.6.0 (and newer versions))</li> <li>CPU_TC.H020 (updated in tables "Register Overview – SPR", "Register Overview – DLMU", sections "Scratch Pad SRAMs", "DLMU SRAMs", "Register access enable Protection", "Safety Protection registers" in CPU chapter of TC3xx UM V1.5.0 (and newer versions))</li> <li>FLASH_TC.H019 (updated in section "Write Burst Once" in NVM chapter of TC3xx UM V1.4.0 (and newer versions))</li> <li>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 TC33xEXT AA Data Sheet V1.1)</li> <li>GETH_TC.H003 (updated in table "ETH MII Signal Timing Parameters" and "ETH RMII Signal Timing Parameters" in TC33xEXT AA Data Sheet V1.1)</li> <li>HSPDM_TC.H004 (figure "ADC Trigger Concept" updated in HSPDM chapter in TC3xx UM V1.3.0 (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 TC33xEXT AA Data Sheet V1.1)</li> </ul> </li> </ul>

**Revision history**

Document version	Date of release	Description of changes
		<ul style="list-style-type: none"> <li>- PADS_TC.P012 (updated footnote 5) on table “Absolute Maximum Ratings” in TC33xEXT AA 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.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 TC33xEXT AA Data Sheet V1.0 (and newer versions))</li> <li>- SAFETY_TC.H018 (updated in Safety Manual v2.0, sections 4.3.1 (Safety Related Functions, table in Introduction), 4.3.2.6 (Digital Acquisition ASIL B/D), 4.3.2.7 (Digital Actuation ASIL B/D))</li> <li>- SCR_TC.H013 (notes added after the description of register RCT_CON in SCR chapter of TC3xx UM V1.1.0 (and newer versions))</li> <li>- SCR_TC.H015 (updated in table “WUF Configuration Registers Address Map” in SCR chapter of TC3xx UM V2.0.0)</li> <li>- SCU_TC.H022 (updated in section “Functional Description” of chapter “LBIST Support” in SCU chapter of TC3xx UM V2.0.0)</li> <li>- SDMMC_TC.H002 (updated figure “SDIO Card Interrupt Sequence” included in SDMMC chapter of TC3xx UM V1.6.0 (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> <li>- SPU_TC.021 (updated in SPU Software Based Self Test (SBST) User Manual V1.0.2.1.0 (2021-03-09))</li> <li>- SPU_TC.022 (updated in section “Interrupts” in SPU chapter of TC3xx UM V2.0.0)</li> <li>- SPU_TC.H007 (documented in section “RAM Initialization” in SPU chapter in TC3xx UM V1.0.0 (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>• PADS_TC.P014</li> <li>• ADC_TC.H044, CCU6_TC.H001, RESET_TC.H007, SCU_TC.H026, SCU_TC.H027 (Errata already published in TC3xx Errata Advance Information 2022-12)</li> <li>• CCU_TC.P001 (Errata already published in TC3xx Errata Advance Information 2023-01)</li> </ul> <p>Changed:</p> <ul style="list-style-type: none"> <li>• PMS_TC.006 (Errata already published in TC3xx Errata Advance Information 2022-12)</li> <li>• SCR_TC.016, SCU_TC.H026 (Errata already published in TC3xx Errata Advance Information 2023-01)</li> </ul> <p>Removed:</p> <ul style="list-style-type: none"> <li>• ADC_TC.H036 - Updated in section "Buffer for the Analog Input" in EVADC chapter of TC3xx user manual V1.5.0 (and newer versions)</li> <li>• ADC_TC.H037 - Information included in section "Result FIFO buffer timing" with TC3xx user manual V2.0.0</li> <li>• ADC_TC.H039 - Information included in section "Result FIFO buffer timing" with TC3xx user manual V2.0.0</li> </ul> <p>When an erratum is used by different families or devices, the erratum is now identical in all errata sheets. Differences between the different families or devices are clearly highlighted in the erratum:</p> <ul style="list-style-type: none"> <li>• CPU_TC.131, FLASH_TC.P003, 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>• SCR_TC.H010 - Changed ordered list from 'a, b, c' to '1, 2, 3'</li> </ul>

## Revision history

Document version	Date of release	Description of changes
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, 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, CCU_TC.H018, CPU_TC.H022, GETH_AI.H004 (Errata already published in TC3xx Errata Advance Information 2023-06)</li> </ul> <p>Removed:</p> <ul style="list-style-type: none"> <li>BROM_TC.H008 - Updated in section "CAN BSL flow" of chapter "AURIX™ TC3xx Platform Firmware" in TC3xx UM V1.1.0 (and newer versions)</li> <li>PADS_TC.H009 - Replaced by PADS_TC.016</li> </ul>
2.2	2023-12-11	<p>For new and changed errata see also column "Change" in tables 2, 3, and 4.</p> <p>New:</p> <ul style="list-style-type: none"> <li>CCU_TC.H017, CPU_TC.H023, PMS_TC.H018, SMU_TC.H019 (Errata already published in TC3xx Errata Advance Information 2023-09)</li> <li>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>SCU_TC.H028</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>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>

## 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-ckx1669709184625**

## Important notice

The information contained in this application note is given as a hint for the implementation of the product only and shall in no event be regarded as a description or warranty of a certain functionality, condition or quality of the product. Before implementation of the product, the recipient of this application note must verify any function and other technical information given herein in the real application. Infineon Technologies hereby disclaims any and all warranties and liabilities of any kind (including without limitation warranties of non-infringement of intellectual property rights of any third party) with respect to any and all information given in this application note.

The data contained in this document is exclusively intended for technically trained staff. It is the responsibility of customer's technical departments to evaluate the suitability of the product for the intended application and the completeness of the product information given in this document with respect to such application.

## Warnings

Due to technical requirements products may contain dangerous substances. For information on the types in question please contact your nearest Infineon Technologies office.

Except as otherwise explicitly approved by Infineon Technologies in a written document signed by authorized representatives of Infineon Technologies, Infineon Technologies' products may not be used in any applications where a failure of the product or any consequences of the use thereof can reasonably be expected to result in personal injury.