Errata Sheet

Rel. 1.5, 2020-07-06

Device **TC38x**

Marking/Step **(E)ES-AD, AD**

Package **see Data Sheet**

No. 075/20

This Errata Sheet describes the deviations from the current user documentation.

Table 1 Current Documentation[1)]

AURIX™ TC3xx User's Manual	V1.2.0	2019-04
AURIX™ TC38x Appendix to User's Manual	V1.2.0	2019-04
TC38x AD/AE-Step Data Sheet	V1.1	2019-09
TriCore TC1.6.2 Core Architecture Manual: - Core Architecture (Vol. 1) - Instruction Set (Vol. 2)	 V1.1 V1.2.2	 2017-08-24 2020-01-15
AURIX™ TC3xx Safety Manual	V1.06	E11/19

1) Newer versions replace older versions, unless specifically noted otherwise.

Make sure you always use the corresponding documentation for this device (User's Manual, Data Sheet, Documentation Addendum (if applicable), TriCore Architecture Manual, Errata Sheet) available in category 'Documents' at **www.infineon.com/AURIX** and **www.myInfineon.com**.

Conventions used in this document

Each erratum identifier follows the pattern **Module_Arch.TypeNumber**:

- **Module**: subsystem, peripheral, or function affected by the erratum
- **Arch**: microcontroller architecture where the erratum was initially detected
  - **AI**: Architecture Independent
  - **TC**: TriCore

- **Type**: category of deviation
  - **[none]**: Functional Deviation
  - **P**: Parametric Deviation
  - **H**: Application Hint
  - **D**: Documentation Update
- **Number**: ascending sequential number within the three previous fields. As this sequence is used over several derivatives, including already solved deviations, gaps inside this enumeration can occur.

Notes

1. 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 Data Sheet/User's Manual.
   This Errata Sheet covers several device versions. If an issue is related to a particular module, and this module is not specified for a specific device version, this issue does not apply to this device version.
   E.g. issues with identifier "EBU" do not apply to devices where no EBU is specified, and issues with identifier "RIF" only apply to ADAS devices.
2. Devices marked with EES or ES are engineering samples which may not be completely tested in all functional and electrical characteristics, therefore they should be used for evaluation only.
   The specific test conditions for EES and ES are documented in a separate Status Sheet.
3. This device is equipped with TriCore "TC1.6.2" core(s). Some of the errata have workarounds which are possibly supported by the tool vendors. Some corresponding compiler switches need possibly to be set. Please see the respective documentation of your compiler.
   For effects of issues related to the on-chip debug system, see also the documentation of the debug tool vendor.

1 History List / Change Summary

Table 2 History List

Version	Date	Remark
1.0	2018-10-24	First version for TC38x step AD • Update with reference to errata sheet V1.1 for TC38x step AB: – For TC38x step AD, FLASH_TC.H016 (Implications on Power-up cycles and Standby mode) replaces FLASH_TC.049 (Limitation of number of power-on/-off cycles) of previous TC38x design steps
1.1	2019-03-29	Update: • New/updated text modules see column “Change” in tables 4..6 of errata sheet V1.1. • Removed: – ADC_TC.089 (GLOBRES ignores result positioning): documented in chapter “Data Alignment” of the EVADC chapter in TC3xx User’s Manual V1.0.0 and following – ADC_TC.H023 (Multiple software writes not propagated to Hardware Data Interface): covered by ADC_TC.091 (Write to GxRES15 not propagated to HDI) – ADC_TC.H025 (Access to EVADC registers in Hard Suspend Mode): documented in specific Note in description of OCS register in the EVADC chapter in TC3xx User’s Manual V1.0.0 and following – MTU_TC.H013 (First Write Access to SSH registers of SCR_RAMINT): issue not present in this design step

Table 2 History List (cont'd)

Version	Date	Remark
1.1	...	... continued: • Removed: – OSC_TC.H001 (Oscillator Operation at VEXT = 3.3V): documented in specific Note in description of OSCCON.GAINSEL field in the CCU chapter in TC3xx User's Manual V1.0.0 and following
1.2	2019-07-19	Update: • New/updated text modules see column "Change" in tables 4..6 of errata sheet V1.2 • Removed: – SRI_TC.H002 (CPU Accesses from PFlash: Stall cycles for local and SRI resources - Documentation update): corrected in table "CPU Accesses: Stall cycles for local and SRI resources" in the CPU chapter of TC3xx User's Manual V1.1.0 and following
1.3	2019-12-02	Update: • New/updated text modules see column "Change" in tables 4..6 of errata sheet V1.3 • Removed: – CCU_TC.H013 (Field SHBY in register OSCCON - Documentation Update): updated description of OSCCON.SHBY in TC3xx User's Manual V1.1.0 and following – CCU_TC.H014 (System Reset value of CCUCON0; Clock Ramp Example - Documentation Updates): updated CCUCON0 description in TC3xx User's Manual V1.1.0 and following

Table 2 History List (cont'd)

Version	Date	Remark
1.3	...	... continued: • Removed: – CPU_TC.H017 (MSUB.Q does not match MUL.Q+SUB.Q - Documentation Update): updated description in TriCore TC1.6.2 Core Architecture Manual V1.1, Vol. 2 Instruction Set – FLASH_TC.H017 (UCB_SWAP MARKERHx and CONFIRMATIONHx - Documentation Update): updated description in NVM chapter "UCB_SWAP_ORIG and UCB_SWAP_COPY" of TC3xx User's Manual V1.1.0 and following – FLASH_TC.H018 (Sequence for Programming/Erasing - Documentation Update): updated description in DMU chapter "Performing Flash Operations" in TC3xx User's Manual V1.2.0 and following – LVDS_TC.H001 (Driver ground potential difference - Additional information): Note included in chapter "High performance LVDS pads" in TC38x AD/AE Data Sheet V1.1 and following – SBCU_TC.H001 (Mapping of CAN2 alarm signal to register ALSTAT1): updated description of register SBCU_ALSTATx (x=1) in SBCU chapter of TC38x Appendix V1.2.0 and following – SCU_TC.H017 (LBIST Configuration B - documentation update): documentation in in SCU chapter of TC38x Appendix V1.1.0 and following describes implementation in TC38x steps AC, AD, AE and following

Table 2 History List (cont'd)

Version	Date	Remark
1.3	...	...continued: • Removed: – STM_TC.H003 (Suspend control for STMx - Documentation Update): updated OCS register description in STM chapter of TC3xx User's Manual V1.2.0 and following
1.4	2020-03-31	Update: • Table 3 (Errata fixed in this step) updated: – added BROM_TC.H018 (documented in table ""ALL CHECKS PASSED" indication by CHSW" in TC38x Appendix V1.0.0 (and following)) – added CPU_TC.H018 (only applies to TC38x steps AA, AB) – added VEXTMON_TC.P001 (updated in TC38xAD Data Sheet V1.0 and following) • New/updated text modules see column "Change" in tables 4..6 of errata sheet V1.4 • Removed: – 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
1.5	2020-07-06	Update: • New/updated text modules see column "Change" in tables 4..6 • reference to CPU_TC.H018 corrected in row for V1.4 (see above): CPU_TC.H018 only applies to TC38x steps AA, AB

Table 3 Errata fixed in this step[1)]

Errata	Short Description	Change
ADC_TC.P013	Increased RMS Noise on TC389 steps AA and AB	Fixed
BROM_TC.011	Use of ABM with PFLASHx banks (x>0)	Fixed
BROM_TC.012	Pull-ups active when LBIST is started from firmware	Fixed
BROM_TC.H007	Side effect of LBIST execution during start-up	Fixed
BROM_TC.H018	“ALL CHECKS PASSED” indication by CHSW - Documentation update	Fixed
CPU_TC.H018	Customer-ID numbering - Documentation update for earlier design steps	Fixed
FLASH_TC.050	Replace Logical Sector command not reliable	Fixed
SMU_TC.011	ALM8[20] permanently active	Fixed
VEXTMON_TC.P001	Secondary supply monitor accuracy after trimming - Limits in 3.3 V range	Fixed

1) Reference = TC38x step AB

Note: Changes to the previous errata sheet version are particularly marked in column “Change” in the following tables.

Table 4 Functional Deviations

Functional Deviation	Short Description	Change	Page
ADC_TC.083	**Changes of RPE not reflected for Start-up Calibration**		**28**
ADC_TC.084	**Wrong Hysteresis after clearing bit FCR**		**28**
ADC_TC.085	**No Service Request when Reference Value FCREF is modified**		**28**
ADC_TC.086	**Wrong Activation of Safety Pull Devices for synchronous Conversions**		**29**

Table 4 Functional Deviations (cont'd)

Functional Deviation	Short Description	Change	Page
ADC_TC.087	Ramp not re-started when writing to FCxFCRAMP0 while FCxFCM.RUNRAMP=11_B		29
ADC_TC.088	Conversions in timer mode may be delayed		29
ADC_TC.090	SRDIS does not disable service requests		30
ADC_TC.091	Write to GxRES15 not propagated to HDI		30
ADC_TC.092	Polling a result register may disturb result FIFO buffer or wait-for-read mode		31
ADC_TC.093	Wait-for-read mode does not work correctly under some circumstances		31
ADC_TC.094	Clearing DRC via register VFR is not indicated		32
ADC_TC.095	Ramp trigger ignored when ramp ends		32
ADC_TC.096	Handling of result register GxRES15		33
ADC_TC.097	Irregular daisy chaining sequence with GLOBRES in WFR mode		33
BareDie_TC.001	TC380 Dimensions – Data Sheet documentation update	New	34
BROM_TC.013	CAN BSL does not send error message if no valid baudrate is detected		35
BROM_TC.014	Lockstep Comparator Alarm for CPU0 after Warm PORST, System or Application Reset if Lockstep is disabled		35
BROM_TC.016	Uncorrectable ECC error in Boot Mode Headers		35
CCU_TC.004	Oscillator supervision – Documentation update for register OSCCON	New	36

Table 4 Functional Deviations (cont'd)

Functional Deviation	Short Description	Change	Page
CPU_TC.130	Data Corruption when ST.B to local DSPR coincides with external access to same address		38
CPU_TC.131	Performance issue when MADD/MSUB instruction uses E0/D0 register as accumulator		39
CPU_TC.132	Unexpected PSW values used upon Fast Interrupt entry		40
DAP_TC.005	DAP client_read: dirty bit feature of Cerberus' Triggered Transfer Mode		41
DAP_TC.007	Incomplete client_blockread telegram in DXCM mode when using the "read CRCup" option		42
DAP_TC.008	DAP Unidirectional Wide Mode (UWM) not working		42
DAP_TC.009	CRC6 error in client_blockwrite telegram		42
DMA_TC.059	ACCEN Protection not implemented for ERRINTRr		43
DMA_TC.066	DMA Double Buffering Operations - Update Address Pointer		43
DMA_TC.067	DMA Double Buffering Software Switch Buffer Overflow		44
DMA_TC.068	DMA Double Buffering Lost DMA Request		44
FLASH_TC.051	ALM7[31] erroneously triggered by NVM operations on PFlash		46
FLASH_TC.053	Erase Size Limit for PFLASH		47
FlexRay_AI.087	After reception of a valid sync frame followed by a valid non-sync frame in the same static slot the received sync frame may be ignored		47

Table 4 Functional Deviations (cont'd)

Functional Deviation	Short Description	Change	Page
FlexRay_AI.088	A sequence of received WUS may generate redundant SIR.WUPA/B events		48
FlexRay_AI.089	Rate correction set to zero in case of SyncCalcResult=MISSING_TERM		49
FlexRay_AI.090	Flag SFS.MRCS is set erroneously although at least one valid sync frame pair is received		49
FlexRay_AI.091	Incorrect rate and/or offset correction value if second Secondary Time Reference Point (STRP) coincides with the action point after detection of a valid frame		50
FlexRay_AI.092	Initial rate correction value of an integrating node is zero if pMicroInitialOffsetA,B = 0x00		51
FlexRay_AI.093	Acceptance of startup frames received after reception of more than gSyncNodeMax sync frames		52
FlexRay_AI.094	Sync frame overflow flag EIR.SFO may be set if slot counter is greater than 1024		52
FlexRay_AI.095	Register RCV displays wrong value		53
FlexRay_AI.096	Noise following a dynamic frame that delays idle detection may fail to stop slot		54
FlexRay_AI.097	Loop back mode operates only at 10 MBit/s		54
FlexRay_AI.099	Erroneous cycle offset during startup after abort of startup or normal operation		55
FlexRay_AI.100	First WUS following received valid WUP may be ignored		56
FlexRay_AI.101	READY command accepted in READY state		57

Table 4 Functional Deviations (cont'd)

Functional Deviation	Short Description	Change	Page
FlexRay_AI.102	Slot Status vPOC!SlotMode is reset immediately when entering HALT state		57
FlexRay_AI.103	Received messages not stored in Message RAM when in Loop Back Mode		58
FlexRay_AI.104	Missing startup frame in cycle 0 at coldstart after FREEZE or READY command		58
FlexRay_AI.105	RAM select signals of IBF1/IBF2 and OBF1/OBF2 in RAM test mode		59
FlexRay_AI.106	Data transfer overrun for message transfers Message RAM to Output Buffer (OBF) or from Input Buffer (IBF) to Message RAM		60
GETH_AI.001	Packets with Destination Address (DA) mismatch are delayed until EOP is received in threshold (cut-through) mode		63
GETH_AI.002	Incorrect Weighted Round Robin Arbitration between Tx and Rx DMA Channels to Access the common Host Bus		65
GETH_AI.003	Header-Payload Split Function Does Not Support IPv6 Packets Received With Zero TCP Payload		66
GETH_AI.005	Application Error Along With Start-of-Packet Can Corrupt the Ongoing Transmission of MAC Generated Packets		67
GETH_AI.006	Incorrect IP Header or Payload Checksum Status Given After MTL TX FIFO Flush		68
GETH_AI.007	IEEE 1588 Timestamp Interrupt Status Bits are Incorrectly Cleared on Write Access to the CSR Register with Similar Offset Address		69

Table 4 Functional Deviations (cont'd)

Functional Deviation	Short Description	Change	Page
GETH_AI.008	Application Error Along with Start-of-Packet Can Corrupt the FCS Field of the Previous Frame in the MAC Pipeline		71
GETH_AI.009	Corrupted Rx Descriptor Write Data		71
GETH_AI.010	Fatal Bus Error Interrupt Might Be Generated for Incorrect DMA Channel		72
GETH_AI.011	Receive Queue Overflow at End of Frame Along with SPRAM Read-Write Conflict Can Cause Data Loss		73
GETH_AI.012	Incorrect Flexible PPS Output Interval When Fine Time Correction Method is Used		74
GETH_AI.013	False Dribble and CRC Error Reported in RMII PHY 10Mbps Mode		75
GETH_AI.014	Receive DMA Channel Generates Spurious Receive Watchdog Timeout Interrupt		76
GETH_AI.015	MAC Receive VLAN Tag Hash Filter Always Operates in Default Mode		78
GETH_AI.016	Receive DMA Header Split Function Incorrectly Overruns the Allocated Header Buffer	New	79
GETH_AI.017	Carrier-Sense Signal Not Generated When False Carrier Detected in RGMII 10/100 Mbps Mode	New	81
GETH_AI.018	Description of the Transmit Checksum Offload Engine - Documentation update	New	82
GETH_TC.001	Reference clock for Time Stamp Update logic is f_{GETH}		83
GETH_TC.002	Initialization of RGMII interface		83
GTM_AI.254	TIM TDU: TDU_STOP=b101 not functional		84

Table 4 Functional Deviations (cont'd)

Functional Deviation	Short Description	Change	Page
GTM_AI.262	DPLL: PSSC behavior in mode change (DPLL_CTRL_0.RMO = 0 ->1)		85
GTM_AI.263	DPLL: DPLL_STATUS.LOCK1 flag (0 ->1) delayed after direction change when DPLL operating in DPLL_CTRL_0.RMO = 1		86
GTM_AI.298	TOM/ATOM: wrong output behaviour in SOMP oneshot mode when oneshot pulse is triggered by TIM_EXT_CAPTURE(x)		87
GTM_AI.299	TOM/ATOM: wrong output behaviour in SOMP oneshot mode when oneshot pulse is triggered by trig_[x-1]		88
GTM_AI.300	DPLL: Change to forward operation when DPLL_THMI is set to zero does not work correctly		88
GTM_AI.301	DPLL: Reset of DPLL_STATUS.BWD1=1 by disabling the DPLL does not cause the direction to change from backward to forward in any case		90
GTM_AI.304	MCS: Scheduling modes Single Prioritization and Multiple Prioritization are not functional		91
GTM_AI.305	TIM Signal Generation with serial shift mode TSSM: If TSSM_OUT is used in channel x and channel x+1 uses edges of FOUT_PREV, these edges show an unexpected delay which lead to a delayed operation of channel measurement or TDU functionality of channel x+1		92
GTM_AI.306	DPLL: DPLL_NUTC.syn_t_old, DPLL_NUSC.syn_s_old not updated according specification		93

Table 4 Functional Deviations (cont'd)

Functional Deviation	Short Description	Change	Page
GTM_AI.307	IRQ: AEI_IM_ADDR is not set in GTM_IRQ_NOTIFY register if cluster 0 is disabled		94
GTM_AI.308	TIM, ARU: Limitation that back-to-back TIM data transfers at full ARU clock rate cannot be transferred correctly with ARU dynamic routing feature		94
GTM_AI.309	TIM Signal Generation with serial shift mode TSSM in channel x: Generated TSSM_OUT signal used in lookup table of inputsrc module of channel x has unpredictable delay		95
GTM_AI.318	MCS: NARD(I) instruction terminates unexpectedly		96
GTM_AI.319	(A)TOM: Unexpected (A)TOM_CCU1TCx_IRQ in up/down counter mode		96
GTM_AI.320	ATOM: Unexpected restart of a SOMS oneshot cycle while ATOM[i]_CH[x]_CM0 is zero		97
GTM_AI.322	DPLL: PSTC, PSSC not updated correctly after fast pulse correction completed (DPLL_CTRL1.PCM1/2 = 0)		98
GTM_AI.323	DPLL: Registers DPLL_NUTC.SYN_T and DPLL_NUSC.SYN_S are updated by the profile (ADT_T.NT/ADT_S.NS) before the DPLL is synchronized (DPLL_STATUS.SYT/S=0)		99
GTM_AI.325	TIM: Bits ACB[2:1] lost on interface to ARU (always zero)		100

Table 4 Functional Deviations (cont'd)

Functional Deviation	Short Description	Change	Page
GTM_AI.326	TIM: ARU bit ACB[0] (signal level) incorrect in case a second ARU request occurs while the actual request is just acknowledged		102
GTM_AI.329	Interference of MCS to AEI/ADC and CPU to AEI traffic within the same cluster could result in incorrect MCS program execution		102
GTM_AI.331	GTM_TIM[i]_AUX_IN_SRC and GTM_EXT_CAP_EN_[i] register: wrong status 2 by AEI write access if cluster 0 is disabled		111
GTM_AI.332	Access to registers GTM_TIM[i]_AUX_IN_SRC and GTM_EXT_CAP_EN_[i] via legacy address space: read data always 0 for AEI read access while cluster 0 is disabled		112
GTM_AI.333	MCS bus master interface: a not word aligned address access to DPLL ram region can cause incorrect execution of MCS channel code		113
GTM_AI.334	DPLL RAM content of single address can be corrupted after leaving debug mode		113
GTM_AI.335	TOM output signal to SPE not functional if up/down counter mode is configured		114
GTM_AI.336	GTM Bus Bridge: Incorrect AEI access execution in case the previous AEI access was aborted with the access timeout abort function		115
GTM_AI.339	DPLL: Control bits DPLL_CTRL_11.PCMF1 and DPLL_CTRL_11.PCMF2 are not reset to 0 after a pulse correction is completed		116

Table 4 Functional Deviations (cont'd)

Functional Deviation	Short Description	Change	Page
GTM_AI.340	TOM/ATOM: Generation of TRIG_CCU0/TRIG_CCU1 trigger signals skipped in initial phase of A/TOM SOMP one-shot mode		117
GTM_AI.341	TOM/ATOM: False generation of TRIG_CCU1 trigger signal in SOMP one-shot mode with OSM_TRIG=1 when CM1 is set to value 1		118
GTM_AI.344	DPLL: Incorrect AEI_STATUS on internal MCS2DPLL interface on valid and implemented address accesses		120
GTM_AI.345	SPE: Incorrect behaviour of direction change control via SPE_CMD.SPE_CTRL_CMD bits		121
GTM_AI.346	ATOM SOMS mode: Shift cycle is not executed correctly in case the reload condition is deactivated with ATOM[i]_AGC_GLB_CTRL.UPEN = 0		122
GTM_AI.347	TOM/ATOM: Reset of (A)TOM[i]_CH[x]_CN0 with TIM_EXT_CAPTURE are not correctly synchronized to selected CMU_CLK/CMU_FXCLK		123
GTM_AI.348	DPLL: Correction of missing pulses delayed after start of pulse generation	Update	124
GTM_AI.349	TOM-SPE: OSM-Pulse width triggered by SPE_NIPD for selected CMU_FXCLK not correct	New	126
GTM_AI.350	TOM-SPE: Update of SPE[i]_OUT_CTRL triggered by SPE_NIPD not working for a delay value 1 in TOM[i]_CH[x]_CM1	New	126

Table 4 Functional Deviations (cont'd)

Functional Deviation	Short Description	Change	Page
GTM_AI.351	MAP: Disable of input lines by MAP_CTRL register not implemented for input signals TSPP0 TIM0_CHx(48) (x=0..2) and TSPP1 TIM0_CHx(48) (x=3..5)	New	127
GTM_AI.352	ATOM: No reload of data from ARU in SOMS and SOMP mode if TIM_EXT_CAPTURE(x) or TRIGIN(x) is selected as clock source	New	128
GTM_AI.353	SPEC-ATOM: Specification of the smallest possible PWM Period in SOMP mode wrong, when ARU_EN=1	New	130
GTM_AI.354	DMCS: Unresolved hazard resulting from RAW (Read After Write) dependency	New	131
GTM_TC.018	DPLLRAM trace data can be wrong		132
GTM_TC.019	ARU can not be traced if GTM cluster 5 is disabled		133
GTM_TC.020	Debug/Normal read access control via bit field ODA.DRAC		134
GTM_TC.021	Registers CANOUTSEL0, CANOUTSEL1 - Documentation update for fields SELx (x = 0, 1)		135
GTM_TC.296	ARU data at the GTM OTGBM interface may be doubled		138
HSCT_TC.012	HSCT sleep mode not supported		138
HSCT_TC.013	Internal Loopback Mode not reliable		138
MCMCAN_AI.015	Edge filtering causes mis-synchronization when falling edge at Rx input pin coincides with end of integration phase		139
MCMCAN_AI.017	Retransmission in DAR mode due to lost arbitration at the first two identifier bits		140

Table 4 Functional Deviations (cont'd)

Functional Deviation	Short Description	Change	Page
MCMCAN_AI.018	Tx FIFO Message Sequence Inversion		142
MCMCAN_AI.019	Unexpected High Priority Message (HPM) interrupt		144
MCMCAN_AI.020	Message transmitted with wrong arbitration and control fields		146
miniMCDS_TC.003	Trace messages get lost when ticks are enabled		148
miniMCDS_TC.004	NESTED_ISR incremented by resets		149
miniMCDS_TC.005	TriCore wrap around write access causes redundant miniMCDS message		150
miniMCDS_TC.006	Selection of SRI trace sources		150
miniMCDS_TC.007	Selection of CPU trace sources		151
miniMCDS_TC.008	MCDS kernel reset shall not be used		151
MTU_TC.012	Security of CPU Cache Memories During Runtime is Limited		152
MTU_TC.017	Unexpected alarms after application reset		153
MTU_TC.018	Gated SRAM alarms		153
MTU_TC.019	Type properties for reserved bits MCONTROL.R8, R12..R14 - Documentation update		155
PADS_TC.011	Pull-ups activate on specific analog inputs upon PORST		155
PER_PLL_TC.001	Peripheral PLL Divider Bypass must not be used		156
PMS_TC.004	EVRC DCDCSYNC output affected by Warm PORST		159

Table 4 Functional Deviations (cont'd)

Functional Deviation	Short Description	Change	Page
PMS_TC.005	Voltage rise at P33 and P34 up to 2.5V during start-up and power-down		159
PMS_TC.006	PORST not released during Cold Power-on Reset until VDDM is available		160
PMS_TC.007	VDDP3 or VDD Overvoltage during start-up may not be detected by PBIST		160
PMS_TC.011	VEXT supplied PU1 pads always in tristate after standby entry		161
PMS_TC.012	Short to Supply and Ground Detection – Documentation update	New	162
QSPI_TC.006	Baud rate error detection in slave mode (error indication in current frame)		162
QSPI_TC.009	USR Events for PT1=2 (SOF: Start of Frame)		163
QSPI_TC.010	Move Counter Mode - USR Events for PT1=4 (RBF: Receive Buffer Filled)		163
QSPI_TC.013	Slave: No RxFIFO write after transmission upon change of BACON.MSB		164
QSPI_TC.014	Slave: Incorrect parity bit upon TxFIFO underflow		164
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		165
QSPI_TC.017	Slave: Reset when receiving an unexpected number of bits		165
SCR_TC.014	SCR pins switched to reset state on warm PORST or Standby Entry and Exit event		166
SCR_TC.015	Bit SCU_PMCON1.WCAN_DIS does not disable WCAN PCLK input		167

Table 4 Functional Deviations (cont'd)

Functional Deviation	Short Description	Change	Page
SCR_TC.016	DUT response to first telegram has incorrect C_START value		167
SCR_TC.017	WCAN module not reliable in TC38x		168
SCR_TC.018	SSC Receive FIFO not working		168
SCR_TC.019	Accessing the XRAM while SCR is in reset state		169
SCR_TC.020	Stored address in mon_RETH may be wrong after a break event		169
SCU_TC.030	Connections of SCU - Documentation in TC3xx Appendix		170
SMU_TC.012	Unexpected alarms when registers FSP or RTC are written		170

Table 5 Deviations from Electrical- and Timing Specification

AC/DC/ADC Deviation	Short Description	Change	Page
ADC_TC.P009	Increased TUE for G10 when using Alternate Reference		172
ADC_TC.P012	Increased RMS Noise		172
ADC_TC.P014	Equivalent Circuitry for Analog Inputs - Additional information		173
EDSADC_TC.P002	Parameters Input Current, Gain Error - Additional information		173
FLASH_TC.P003	Program Flash Erase Time per Multi-Sector Command		174

Table 5 Deviations from Electrical- and Timing Specification (cont'd)

AC/DC/ADC Deviation	Short Description	Change	Page
GETH_TC.P001	Maximum and minimum GETH operating frequency - Documentation update		174
RESET_TC.P003	Parameter limits for t_{PI} (Ports inactive after ESR0 reset active) – Documentation update	New	175

Table 6 Application Hints

Hint	Short Description	Change	Page
ADC_TC.H026	Additional Waiting Phase in Slow Standby Mode		176
ADC_TC.H028	Inconsistent contents in GLOBRES if result writes come too close		176
ADC_TC.H029	Storing result values to a full FIFO structure		177
ADC_TC.H030	Flushing a running queue may corrupt previous conversion results		177
ADC_TC.H032	ADC accuracy parameters - Definition		178
ADC_TC.H033	Basic Initialization Sequence for Primary and Secondary EVADC Groups		178
ADC_TC.H034	Effect of reduced reference voltage on parameter QCONV - Data Sheet footnote update		179
ADC_TC.H035	Effect of input leakage current on Broken Wire Detection		180
ADC_TC.H036	Minimum Input Buffering Time - Additional information		181

Table 6 Application Hints (cont'd)

Hint	Short Description	Change	Page
ADC_TC.H037	CPU read access latency to result FIFO buffer		182
ASCLIN_TC.H001	Bit field FRAMECON.IDLE in LIN slave tasks	Update	183
BROM_TC.H008	CAN BSL does not support DLC = 9 and DLC = 11		183
BROM_TC.H009	Re-Enabling Lockstep via BMHD		183
BROM_TC.H014	SSW behavior in case of wrong state or uncorrectable error in UCBs - Documentation Update		184
BROM_TC.H015	Different initial values for CPU0_PMEM SSH registers in MTU after cold PORST if SOTA/SWAP is enabled		185
BROM_TC.H016	CHSW fails check of PMS_EVR registers after software triggered LBIST		185
BROM_TC.H017	CHSW results after LBIST execution		188
CCU_TC.H012	Configuration of the Oscillator- Documentation Update		188
CLC_TC.H001	Description alignment for bits DISR, DISS, EDIS in register CLC - Documentation Update		189
CPU_TC.H019	Semaphore handling for shared memory resources		190
CPU_TC.H020	Inconsistent register description in CPU chapter - Documentation update		193
DAM_TC.H001	RAM size of DAM instance in TC38x - Documentation correction		197
DAM_TC.H002	Triggering DAM MEMCON.RMWERR and INTERR flags	Update	198

Table 6 Application Hints (cont'd)

Hint	Short Description	Change	Page
DTS_TC.H002	Unexpected alarms after start-up/wake-up when temperature is close to lower/upper limit		198
EDSADC_TC.H001	Auxiliary filter cleared with start of integration window - Additional information		199
EDSADC_TC.H003	Behavior of EDSADC result register in case of hardware controlled integration		199
FLASH_TC.H016	Implications on Power-up and Standby mode wake-up cycles		200
FLASH_TC.H019	Write Burst Once command – Documentation update		201
FlexRay_AI.H004	Only the first message can be received in External Loop Back mode		201
FlexRay_AI.H005	Initialization of internal RAMs requires one eray_bclk cycle more		202
FlexRay_AI.H006	Transmission in ATM/Loopback mode		202
FlexRay_AI.H007	Reporting of coding errors via TEST1.CERA/B		203
FlexRay_AI.H009	Return from test mode operation		203
FlexRay_AI.H011	Behavior of interrupt flags in FlexRay™ Protocol Controller (E-Ray)		203
FlexRay_TC.H003	Initialization of E-Ray RAMs		204
FPI_TC.H003	Burst write access may lead to data corruption		204
GETH_AI.H001	Preparation for Software Reset		205
GETH_AI.H002	Back-to-back writes to same register - Additional information		206
GTM_TC.H010	Trigger Selection for EVADC and EDSADC		207

Table 6 Application Hints (cont'd)

Hint	Short Description	Change	Page
GTM_TC.H019	Register GTM_RST - Documentation Update		207
GTM_TC.H021	Interrupt strategy mode selection in IRQ_MODE		208
GTM_TC.H022	Field ENDIS_CTRLx in register ATOMi_AGC_ENDIS_CTRL - Documentation Update		209
HSCT_TC.H009	High speed dividers five phase clock sequence ordering		210
I2C_TC.H008	Handling of RX FIFO Overflow in Slave Mode		211
LBIST_TC.H001	LBIST signature for configuration A not present in UCB_SSW in specific devices		211
MCMCAN_AI.H001	Behavior of interrupt flags in CAN Interface (MCMCAN)		213
MCMCAN_AI.H002	Busoff Recovery		213
MCMCAN_TC.H006	Unintended Behavior of Receive Timeout Interrupt		215
MCMCAN_TC.H007	Delayed time triggered transmission of frames		215
miniMCDS_TC.H001	Program trace of CPUx (x > 0) program start not correct		216
MTU_TC.H015	ALM7[0] may be triggered after cold PORST		217
MTU_TC.H016	MCi_FAULTSTS.OPERR[2] may be triggered at power-up in case LBIST is not run		217
OCDS_TC.H014	Avoiding failure of key exchange command due to overwrite of COMDATA by firmware		217

Table 6 Application Hints (cont'd)

Hint	Short Description	Change	Page
OCDS_TC.H015	System or Application Reset while OCDS and lockstep monitoring are enabled		219
OCDS_TC.H016	Release of application reset via OJCONF may fail		219
OCDS_TC.H018	Unexpected stop of Startup Software after system/application reset		220
PMS_TC.H003	VDDPD voltage monitoring limits		220
PMS_TC.H005	SCR clock in system standby mode - Documentation update		221
PORTS_TC.H012	LVDS Input Pad on P14.9/10 in BGA-292 Packages		222
PSI5_TC.H001	No communication error in case of payload length mismatch	New	223
QSPI_TC.H008	Details of the Baud Rate and Phase Duration Control - Documentation update	New	223
SAFETY_TC.H001	Features intended for development only – Documentation update to Safety Manual		223
SAFETY_TC.H002	SM[HW]:CPU.PTAG:ERROR_DETECTION – Documentation update to Safety Manual		225
SAFETY_TC.H003	ESM[SW]:EDSADC:VAREF_PLAUSIBILITY and ESM[SW]:EVADC:VAREF_PLAUSIBILITY – Additional information		226
SAFETY_TC.H004	ESM[HW]:PMS:VEXT_VEVRSB_OVERVOLTAGE – Wording update		227
SAFETY_TC.H006	SM[HW]:PMS:VDD_MONITOR – Documentation update	New	228
SAFETY_TC.H007	SM[HW]:CLOCK:PLL_LOSS_OF_LOCK_DETECTION – Documentation update	New	229

Table 6 Application Hints (cont'd)

Hint	Short Description	Change	Page
SAFETY_TC.H008	Link between ESM[SW]:CONVCTRL:ALARM_CHECK and SM[HW]:CONVCTRL:PHASE_SYNC_ERR - Additional information	New	229
SCR_TC.022	Effect of application or system reset and warm PORST on MC77_ECCD and MC78_ECCD for SCR RAMs	New	230
SCR_TC.H009	RAM ECC Alarms in Standby Mode		231
SCR_TC.H010	HRESET command erroneously sets RRF flag		231
SCR_TC.H011	Hang-up when warm PORST is activated during Debug Monitor Mode		231
SCR_TC.H012	Reaction in case of XRAM ECC Error		232
SCR_TC.H014	Details on WDT pre-warning period		233
SCU_TC.H016	RSTSTAT reset values - documentation update		233
SCU_TC.H020	Digital filter on ESRx pins - Documentation update		234
SCU_TC.H021	LBIST execution affected by TCK/DAP0 state		234
SCU_TC.H022	Effect of LBIST execution on SRAMs - Additional information	New	234
SENT_TC.H006	Parameter V_{ILD} on pads used as SENT inputs		235
SMU_TC.H010	Clearing individual SMU flags: use only 32-bit writes		238
SMU_TC.H012	Handling of SMU alarms ALM7[1] and ALM7[0]		239

Table 6 Application Hints (cont'd)

Hint	Short Description	Change	Page
SMU_TC.H013	Increased Fault Detection for SMU Bus Interface (SMU_CLC Register)		240
SMU_TC.H015	Calculation of the minimum active fault state time TFSP_FS - Additional information	New	240
SRI_TC.H001	Using LDMST and SWAPMSK.W instructions on SRI mapped Peripheral Registers (range 0xF800 0000-0xFFFF FFFF)		241
STM_TC.H004	Access to STM registers while STMDIV = 0		241

2 Functional Deviations

ADC_TC.083 Changes of RPE not reflected for Start-up Calibration

When bit GxANCFG.RPE (Reference Precharge Enable) is changed just prior to a start-up calibration, the new value is not reflected to the analog part, but the previous value is used.

Bit RPE is reflected correctly after reset (RPE = 1_B) and after each conversion.

Workaround

If RPE needs to be changed for a repeated start-up calibration, execute a dummy conversion before writing 1_B to bit SUCAL in register GLOBCFG.

ADC_TC.084 Wrong Hysteresis after clearing bit FCR

Disabling a fast compare channel (FCxFCM.ANON = 0_B) also clears the result flag FCxFCBFL.FCR.

After reenabling the channel, however, the previous hysteresis value is used, independent of FCR.

If the previous result was 1 then the lower hysteresis limit is used (instead of the upper limit as for FCR = 0_B) until the input signal falls below the lower limit.

ADC_TC.085 No Service Request when Reference Value FCREF is modified

The fast compare channel offers a mode where modifications of the reference value FCxFCM.FCREF are indicated via a service request (FCxFCM.SRG = 10_B).

Hardware modifications of FCREF in alternate mode, ramp mode or analog source mode (FCxFCM.AUE ≠ 00_B), however, are not indicated with a service request.

Workaround

Generate the service request from the source of the respective hardware signal. This is possible for alternate mode and analog source mode.

ADC_TC.086 Wrong Activation of Safety Pull Devices for synchronous Conversions

If automatic test sequences are used (GLOBTE.TFEx=1 and enabled via GxQINR2) for synchronized conversions, the respective pull devices may either not be activated or may be assigned to the preceding conversion.

Workaround

Do not use automatic test sequences together with synchronized conversions, but use either feature exclusively.

ADC_TC.087 Ramp not re-started when writing to FCxFCRAMP0 while FCxFCM.RUNRAMP=11_B

Ramp generation for fast compare channels can be started by writing a value to register FCxFCOMP0.

When mode RUNRAMP=11_B (stop ramp upon trigger) is selected in register FCxFCM, writing to FCxFCOMP0 does not re-start the ramp while the ramp counter is counting.

Note: Starting an initial ramp by writing to FCxFCOMP0 starts the ramp in both modes (RUNRAMP=11_B or 01_B).

Workaround

Use mode RUNRAMP=01_B, which is independent of the trigger.

ADC_TC.088 Conversions in timer mode may be delayed

When operating in timer mode (GxQCTRLi.TMEN=1_B), conversions are expected to start with the falling edge of the trigger signal.

Other conversion requests during the preface time are erroneously accepted and started which leads to delays of the targeted conversion.

Enabling cancel-inject-repeat mode limits the possible delay to 2 module clock cycles.

Note: This delay does not accumulate, i.e. the intended conversion rate is preserved.

ADC_TC.090 SRDIS does not disable service requests

When using source events for daisy chaining, usually only the last source event is meant to generate a service request to the system.

Setting bit SRDIS (service request disable) in register GxTRCTR of the other groups should prevent service requests from being generated.

This disable function, however, is not working.

Workaround

Set the corresponding service request node pointer to a non-existent output (e.g. GxSEVNP.SEViNP = 1000_B).

ADC_TC.091 Write to GxRES15 not propagated to HDI

The Hardware Data Interface (HDI) propagates all values written to result register GxRES15 to other modules. This is also done when writing to GxRES15 via software.

However, when multiple write operations are executed consecutively, the updated value may not be propagated to the HDI.

Workaround

If GxRES15 needs to be written via software several times, make sure that at least 50 SPB clock cycles are between two consecutive write accesses.

ADC_TC.092 Polling a result register may disturb result FIFO buffer or wait-for-read mode

Values from a result FIFO buffer are read from the lowest register of the respective structure.

When polling this output register via software (i.e. reading the register before the result is available), the FIFO mechanism may be blocked due to an internal synchronization issue, or wait-for-read mode may be blocked.

Workaround

Do not poll the output register; instead, only read the register after the corresponding service request has been generated and before a new conversion has been finished.

ADC_TC.093 Wait-for-read mode does not work correctly under some circumstances

Wait-for-read mode prevents conversions from being started while the targeted result register is occupied, i.e. its valid flag is set because it has not yet been read.

The following configurations can lead to malfunctions:

1. Wait-for-read selected for the input register of a result FIFO structure

If wait-for-read is selected for the input register of a result FIFO structure, the register will not be released once it was occupied, and no subsequent conversions will be started.

Recommendation

Do not use wait-for-read on FIFO structures.

2. Valid flag or DRC of result register cleared by setting bit GxVFR.VFy

If the valid flag or DRC of a result register is cleared by writing 1_B to the corresponding bit GxVFR.VFy, the result register may not be released and no subsequent conversions will be started.

Recommendation

Do not clear the valid flag or DRC by setting GxVFR.VFy when wait-for-read is used for the corresponding result register.

3. Difference mode selected for a result register

When difference mode (GxRCRy.DMM = 10_B) is configured for a result register, the wait-for-read feature will neither work for this result register nor for result register GxRES0 of that converter group.

Recommendation

When using difference mode (GxRCRy.DMM = 10_B), wait-for-read mode must not be used for the corresponding result register nor for result register GxRES0 of that group (i.e. configure both GxRCRy.WFR = 0_B and GxRCR0.WFR = 0_B).

ADC_TC.094 Clearing DRC via register VFR is not indicated

Writing 1_B to a bit within register GxVFR clears the corresponding valid flag VF and the data reduction counter DRC. In register GxRES(D)y, however, the cleared DRC is not indicated. Bitfield DRC is only updated after the next result value becomes available for accumulation.

The accumulation process itself works as intended.

Workaround

None.

ADC_TC.095 Ramp trigger ignored when ramp ends

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

- A trigger that occurs while the ramp is running will restart the ramp from its defined starting level.

- A trigger that occurs exactly at the time when the ramp is completed (corner case) will be ignored and lead to no action.

Workaround

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

ADC_TC.096 Handling of result register GxRES15

Due to a synchronization issue, the contents of result register GxRES15 is unreliable if a new result is provided in GxRES15, and at the same time

- result register GxRES15 is read, or
- the valid flag VF15 in register GxVFR is set to 1_B in order to clear VF15 and the Data Reduction Counter (DRC) in register GxRES15.

Workaround

Use the corresponding interrupt to read result register GxRES15, or to clear valid flag VF15 and DRC (writing 1_B to GxVFR.VF15).

ADC_TC.097 Irregular daisy chaining sequence with GLOBRES in WFR mode

Since daisy chaining covers several groups of the EVADC, the result values are likely to be stored in the global result register GLOBRES.

When GLOBRES is configured to wait-for-read (WFR) mode, the first conversion of a subsequent group will not be suspended until GLOBRES has been read. This exception is due to a timing issue.

Workaround

Add a dummy conversion to GLOBRES to the end of the sequence of a given group. The dummy result can be identified by its associated group/channel

number and will be overwritten automatically by the first conversion of the next group.

Note: In order to avoid the read of the dummy result the following timing relationship is required:
the time interval between the read of the last valid result (before the dummy conversion) in the given group and the read of the first conversion result in the next group must be greater than the dummy conversion time plus the time for the first conversion in the next group.

BareDie_TC.001 TC380 Dimensions – Data Sheet documentation update

The documentation updates below apply to the following specific sections for the TC380 bare die variant in the current version of the TC38x Data Sheet:

TC380 Carrier Tape

In table “TC380 Chip Dimensions”, the value for dimension T shall be corrected from 0.2 mm to 0.3 mm:

Table 7 TC380 Chip Dimensions

Device	X	Y	T
TC380	7.392 mm[1)]	7.665 mm	0.3 mm

1) Values have an accuracy of +/- 0.005 mm

Bare Die Variant Pin Configuration of TC38x

The following note shall be added to table “Pad List”

Note: The coordinates are in units of 0.001 µm and refer to the center of the bond area within the pad outline. The center of the bond area is offset from the center of the pad outline by 37.5 µm towards the chip center. The coordinate origin is located near the pin 1 die corner inside the die area. The coordinate origin has no distinct corresponding physical feature or marking.

BROM_TC.013 CAN BSL does not send error message if no valid baudrate is detected

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”, 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.

BROM_TC.014 Lockstep Comparator Alarm for CPU0 after Warm PORST, System or Application Reset if Lockstep is disabled

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.

BROM_TC.016 Uncorrectable ECC error in Boot Mode Headers

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 (e.g. clear them to 0).

CCU_TC.004 Oscillator supervision – Documentation update for register OSCCON

The formulas for the threshold frequencies f_{LV}, f_{HV} that are documented in the description of bits PLLLV and PLLHV in register OSCCON in the current version of the TC3xx User's Manual are not correctly representing the actual device behavior. The formulas shall be updated as listed below.

In addition, the note listed below on the range of the reference frequency f_{OSC} supervised by the oscillator watchdog shall be added to the description of field OSCVAL.

Table 8 Register OSCCON - Documentation updates[1)]

Field	Bits	Type	Description
PLLLV	1	rh	**Oscillator for PLL Valid Low Status Bit** ... By using the crystal's nominal frequency (f_{oscnom}), the lower threshold frequency f_{LV} calculates as follows: • $f_{LV} = f_{oscnom}$ * **0.75** - 0.31 MHz (typical case for back-up clock after trimming) • $f_{LV} = f_{oscnom}$ * 0.53 - 0.39 MHz (lower boundary for back-up clock before trimming) ...
PLLHV	8	rh	**Oscillator for PLL Valid High Status Bit** ... By using the crystal's nominal frequency (f_{oscnom}), the upper threshold frequency f_{HV} calculates as follows: • $f_{HV} = f_{oscnom}$ * **1.46** + 0.29 MHz (typical case for back-up clock after trimming) • $f_{HV} = f_{oscnom}$ * 1.86 + 0.21 MHz (higher boundary for back-up clock before trimming) ...
OSCVAL	20:16	rw	**OSC Frequency Value** ... The reference frequency calculates as follows: f_{OSC} = (OSCCON.OSCVAL - 1 + 16) MHz **Note**: **Valid range for f_{OSC} is from 16 MHz - 40 MHz. For any other value set outside this range, the status of flags PLLHV and PLLLV is undefined.**

1) Only the direct context of the updated text is shown here, with most significant modifications to present text in bold. For the other parts see the description of register OSCCON in the TC3xx User's Manual.

CPU_TC.130 Data Corruption when ST.B to local DSPR coincides with external access to same address

Under certain conditions, when a CPU accesses it's local DSPR using "store byte" (ST.B) instructions, coincident with stores from another bus master (remote CPU, DMA etc.) 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 that 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.

Workarounds

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, other DSPR etc.).

CPU_TC.131 Performance issue when MADD/MSUB instruction uses E0/D0 register as accumulator

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/D0 register. The accumulator input is that to which the multiplication result is added to / subtracted from in a MAC instruction.

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

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

Example:

```
maddm.h e0, e0, d3, d5ul ; MUL/MAC writing E0 as result
ld.d    e8, [a5]   ; Load instruction causing IP bubble
maddm.h e0, e0, d6, d8ul ; MAC using E0 as accumulator.
                   ; Should be delayed by 1 cycle due to
                   ; dependency to result of previous LD.D,
                   ; but is delayed for 2 cycles
```

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

CPU_TC.132 Unexpected PSW values used upon Fast Interrupt entry

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

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

- TC1.6E (in TC21x..TC27x): Only the first instruction of the ISR is affected.
- TC1.6P (in TC26x..TC29x), TC1.6.2P (in TC3xx): Up to 4 instructions at the start of the ISR may be affected. However, if the following precondition 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.

Workarounds

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 (e.g. 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 preconditions above.

DAP_TC.005 DAP client_read: dirty bit feature of Cerberus' Triggered Transfer Mode

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 (e.g. 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".

DAP_TC.007 Incomplete client_blockread telegram in DXCM mode when using the “read CRCup” option

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.

DAP_TC.008 DAP Unidirectional Wide Mode (UWM) not working

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

DAP UWM is working only for certain telegrams (e.g. sync, dapisc) but not for read or write accesses to the device.

Workaround

Use regular DAP Unidirectional Mode or any other DAP mode.

DAP_TC.009 CRC6 error in client_blockwrite telegram

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

If a CRC6 error happens in a client_blockwrite telegram, the DAP module will not execute the write and the tool will run into timeout according to the DAP protocol.

But in this case a following client_blockwrite (with start address) will be ignored by the DAP module.

Workaround

If the tool is running into a timeout after a client_blockwrite telegram it should transmit a dummy client_blockread telegram (e.g. len=0, arbitrary address) which will clean up the DAP client_blockwrite function.

DMA_TC.059 ACCEN Protection not implemented for ERRINTRr

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

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

Workaround

None.

DMA_TC.066 DMA Double Buffering Operations - Update Address Pointer

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_B),
- DMA Double Source Buffering Automatic Hardware and Software Switch
  – (DMA channel DMA_ADICRz.SHCT = 1001_B),
- DMA Double Destination Buffering Software Switch Only
  – (DMA channel DMA_ADICRz.SHCT = 1010_B),
- DMA Double Destination Buffering Automatic Hardware and Software Switch
  – (DMA channel DMA_ADICRz.SHCT = 1011_B).

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.

DMA_TC.067 DMA Double Buffering Software Switch Buffer Overflow

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.

DMA_TC.068 DMA Double Buffering Lost DMA Request

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.

FLASH_TC.051 ALM7[31] erroneously triggered by NVM operations on PFlash

Due to a timing issue, alarm ALM7[31] (Flash Stored Configuration Error) can erroneously be triggered by one of the following NVM operations:

- a sleep/wakeup request, or
- any FLASH command sequence (Write*, Erase*, ...) except “Reset to Read”, “Enter Page Mode”, “Load Page”, “Disable/Resume Protection”, “Clear Status”.

This erroneous alarm cannot be ignored as it would hide other real faults (see safety manual for safety mechanisms resulting in alarm ALM7[31]).

If system reset is configured as reaction to ALM7[31], the system may end up in an endless loop NVM operation / system reset (for example when running a SOTA update).

Note: This problem may only occur for NVM operations on PFlash. It does not occur for NVM operations on DFlash.

Workaround reducing the number of system resets:

- Alarm reaction from SMU for ALM7[31] shall be configured as interrupt.
- Interrupt service routine shall:
  - while NVM PFLASH operation is executed: clear SMU alarm ALM7[31] at first occurrence and trigger alarm reaction system reset only if ALM7[31] appears again immediately (within 5 SPB clock cycles). This means: this check should be performed between 5 SPB clock cycles after clearing the alarm, or at latest within the Diagnostic Test Interval, i.e. max. 125µs after clearing the alarm after the first occurrence;
  - while no NVM PFLASH operation is executed: trigger alarm reaction system reset.

Note: There is an increased (but still very low) risk for undetected multi-bit errors in NVM read data of other banks during this delay time.

- Execute data integrity check after system reset.

Workaround avoiding the alarm:

Set the system clock frequency (f_{SRI}) to ≤ 100 MHz while performing the NVM operations. In this case the timing issue will not occur and ALM7[31] will not be triggered erroneously.

FLASH_TC.053 Erase Size Limit for PFLASH

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°C OR junction temperature T_J > 70°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.

FlexRay_AI.087 After reception of a valid sync frame followed by a valid non-sync frame in the same static slot the received sync frame may be ignored

Description:

If in a static slot of an even cycle a valid sync frame followed by a valid non-sync frame is received, and the frame valid detection (prt_frame_decoded_on_X) of the DEC process occurs one sclk after valid frame detection of FSP process (fsp_val_syncfr_chx), the sync frame is not taken into account by the CSP process (devte_xxs_reg).

Scope:

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

Effects:

In the described case the sync frame is not considered by the CSP process. This may lead to a SyncCalcResult of MISSIMG_TERM (error flag SFS.MRCS set). As a result the POC state may switch to NORMAL_PASSIVE or HALT or the Startup procedure is aborted.

Workaround

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

FlexRay_AI.088 A sequence of received WUS may generate redundant SIR.WUPA/B events

Description:

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

Scope:

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

Effects:

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

Workaround

Ignore redundant SIR.WUPA/B events.

FlexRay_AI.089 Rate correction set to zero in case of SyncCalcResult=MISSING_TERM

Description:

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

Scope:

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

Effects:

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

Workaround

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

FlexRay_AI.090 Flag `SFS.MRCS` is set erroneously although at least one valid sync frame pair is received

Description:

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

the node receives in slot n+1 a sync frame that matches with a sync frame received in the even cycle 2c, the sync frame pair is not taken into account by CSP process. This may cause the flags SFS.MRCS and EIR.CCF to be set erroneously.

Scope:

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

Effects:

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

Workaround

Correct configuration of gSyncNodeMax.

FlexRay_AI.091 Incorrect rate and/or offset correction value if second Secondary Time Reference Point (STRP) coincides with the action point after detection of a valid frame

Description:

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

Scope:

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

Effects:

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

Workaround

Configure action point offset smaller than static frame length.

FlexRay_AI.092 Initial rate correction value of an integrating node is zero if pMicroInitialOffsetA,B = 0x00

Description:

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

Scope:

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

Effects:

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

Workaround

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

FlexRay_AI.093 Acceptance of startup frames received after reception of more than gSyncNodeMax sync frames

Description:

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

Scope:

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

Effects:

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

Workaround

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

FlexRay_AI.094 Sync frame overflow flag `EIR.SFO` may be set if slot counter is greater than 1024

Description:

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

Scope:

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

Effects:

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

Workaround

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

FlexRay_AI.095 Register RCV displays wrong value

Description:

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

Scope:

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

Effects:

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

Workaround

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

FlexRay_AI.096 Noise following a dynamic frame that delays idle detection may fail to stop slot

Description:

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

Scope:

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

Effects:

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

Workaround

None.

FlexRay_AI.097 Loop back mode operates only at 10 MBit/s

Description:

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

Scope:

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

Effects:

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

Workaround

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

FlexRay_AI.099 Erroneous cycle offset during startup after abort of start-up or normal operation

Description:

An abort of startup or normal operation by a READY command near the macotick border may lead to the effect that the state INITIALIZE_SCHEDULE is one macrotick too short during the first following integration attempt. This leads to an early cycle start in state INTEGRATION_COLDSTART_CHECK or INTEGRATION_CONSISTENCY_CHECK.

As a result the integrating node calculates a cycle offset of one macrotick at the end of the first even/odd cycle pair in the states INTEGRATION_COLDSTART_CHECK or INTEGRATION_CONSISTENCY_CHECK and tries to correct this offset.

If the node is able to correct the offset of one macrotick (pOffsetCorrectionOut >> gdMacrotick), the node enters NORMAL_ACTIVE with the first startup attempt.

If the node is not able to correct the offset error because pOffsetCorrectionOut is too small (pOffsetCorrectionOut ≤ gdMacrotick), the node enters ABORT_STARTUP and is ready to try startup again. The next (second) startup attempt is not effected by this erratum.

Scope:

The erratum is limited to applications where READY command is used to leave STARTUP, NORMAL_ACTIVE, or NORMAL_PASSIVE state.

Effects:

In the described case the integrating node tries to correct an erroneous cycle offset of one macrotick during startup.

Workaround

With a configuration of pOffsetCorrectionOut >> gdMacrotick • (1+cClockDeviationMax) the node will be able to correct the offset and therefore also be able to successfully integrate.

FlexRay_AI.100 First WUS following received valid WUP may be ignored

Description:

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

Scope:

The erratum is limited to the reception of redundant wakeup patterns.

Effects:

Delayed setting of status interrupt flags `SIR.WUPA/B` for redundant wakeup patterns.

Workaround

None.

FlexRay_AI.101 READY command accepted in READY state

Description:

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

Scope:

The erratum is limited to the READY state.

Effects:

Flag CCSV.CSI is set. Cold starting needs to be enabled by POC command ALLOW_COLDSTART (SUCC1.CMD = 1001_B).

Workaround

None.

FlexRay_AI.102 Slot Status vPOC!SlotMode is reset immediately when entering HALT state

Description:

When the protocol engine is in the states NORMAL_ACTIVE or NORMAL_PASSIVE, a HALT or FREEZE command issued by the Host resets vPOC!SlotMode immediately to SINGLE slot mode (CCSV.SLM[1:0] = 00_B). According to the FlexRay protocol specification, the slot mode should not be reset to SINGLE slot mode before the following state transition from HALT to DEFAULT_CONFIG state.

Scope:

The erratum is limited to the HALT state.

Effects:

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

Workaround

None.

FlexRay_AI.103 Received messages not stored in Message RAM when in Loop Back Mode

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

Scope:

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

Effects:

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

Workaround

Leave HALT state by hardware reset.

FlexRay_AI.104 Missing startup frame in cycle 0 at coldstart after FREEZE or READY command

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

Scope:

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

Effects:

During coldstart it may happen that no startup frame is sent in cycle 0 after entering COLDSTART_COLLISION_RESOLUTION state from COLDSTART_LISTEN state.

Severity:

Low, as the next coldstart attempt is no longer affected. Coldstart sequence is lengthened but coldstart of FlexRay system is not prohibited by this behaviour.

Workaround

Use a static slot greater or equal 8 for the startup / sync message.

FlexRay_AI.105 RAM select signals of IBF1/IBF2 and OBF1/OBF2 in RAM test mode

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

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

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

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

Scope:

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

Effects:

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

Severity:

Low, workaround available.

Workaround

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

FlexRay_AI.106 Data transfer overrun for message transfers Message RAM to Output Buffer (OBF) or from Input Buffer (IBF) to Message RAM

The problem occurs under the following conditions:

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

2) The Host triggers a transfer from / to the Last Configured Buffer in the Message RAM with a specific time relation to the start of the TBF transfer described under 1).

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

a) Message buffer transfer from Message RAM to OBF

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

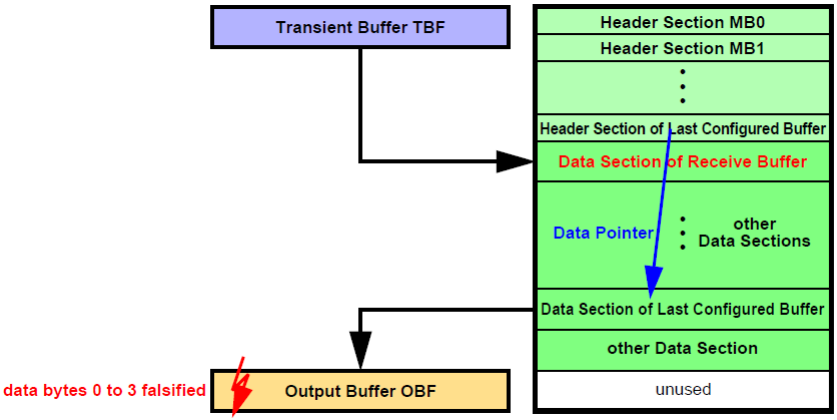

Figure 1 Message buffer transfer from Message RAM to OBF

b) Message buffer transfer from IBF to Message RAM

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

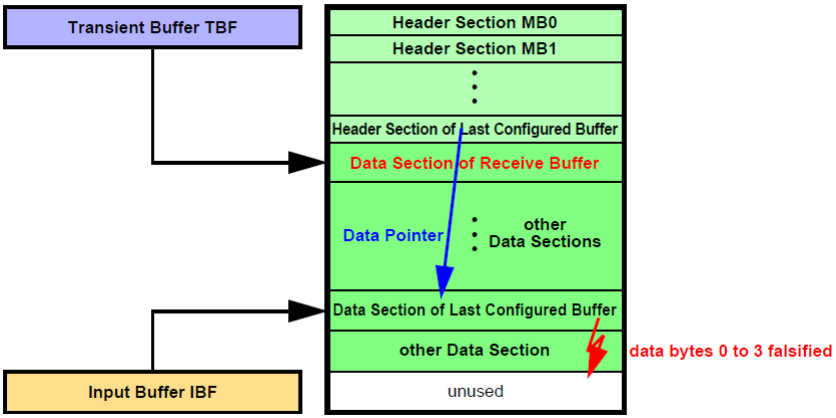

Figure 2 Message buffer transfer from IBF to Message RAM

Scope:

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

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

AND

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

Effects:

a) When a message is transferred from the Last Configured Buffer in the Message RAM to the OBF and **PLC** = 127 it may happen, that at the end of the transfer the OBF word on address 00h (payload data bytes 0 to 3) is overwritten with unexpected data (see **Figure 1**).

b) When a message is transferred from IBF to the Last Configured Buffer in the Message RAM, it may happen, that at the end of the transfer of the Data Section one additional write access overwrites the following word, which may be the first word of another message's Data Section in the Message RAM (see **Figure 2**).

Severity:

Medium, workaround available, check of configuration necessary.

Workaround

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

OR

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

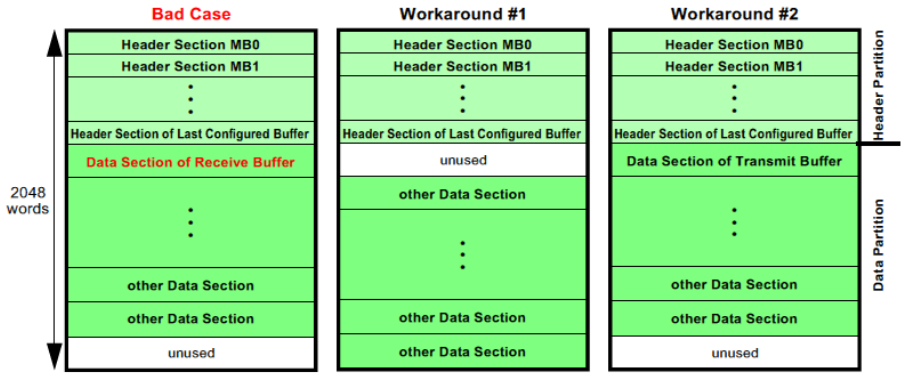

Figure 3 Message RAM Configurations

GETH_AI.001 Packets with Destination Address (DA) mismatch are delayed until EOP is received in threshold (cut-through) mode

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.

GETH_AI.002 Incorrect Weighted Round Robin Arbitration between Tx and Rx DMA Channels to Access the common Host Bus

The DWC_EQOS has independent Transmit (Tx) and Receive (Rx) DMA engines. The transaction requests from the Tx and Rx DMA engines are arbitrated to allow access to the common AHB or DMA master interface. The following two types of arbitrations are supported by programming Bit[1] of the DMA_Mode register

- Weighted Round-Robin Arbitration
- Fixed-Priority Arbitration

The Bits[14:12] control the ratio of the weights between the Tx DMA and the Rx DMA in the Weighted Round Robin scheme. Table 11-3 in the DesignWare Cores Ethernet Quality-of-Service Databook, Version 4.21a explains the expected behavior.

However, due to this defect, programmed Priority Ratio (Bit[14:12] - PR) in Weighted-Round Robin scheme is not adhered to, when Rx DMA is given higher priority over Tx DMA or vice-versa. This is due to clock-cycle gaps present between the completion of one transfer initiated by Rx (or Tx) DMA and the next request for a transfer from Rx (or Tx) DMA. In this gap, the arbiter allocates the bus to the request from Tx (or Rx) DMA. Therefore, the arbiter operates in a 1:1 (round-robin) scheme even though higher ratio is given to the requests from Rx (or Tx) DMA.

In the Rx DMA engine, the gaps are introduced due to latency involved during arbitration across multiple Rx Queues (if selected), arbitration across multiple Rx DMA Channels (if selected), and inherent latency inside the Rx DMA controller. Similarly, in the Tx DMA engine, the gaps are introduced due to latency involved during arbitration across multiple Tx DMA Channels (if selected).

The arbiter module is fixed to delay the arbitration to absorb the various latencies. This allows the possibility of successful consecutive transfers from Tx or Rx DMA engines as per the programmed Priority Ratio. Also, the delay introduced on the Rx path due to arbitration across Rx Queues must be eliminated by programming bit[3] (RXQ_FRM_ARBIT) of MTL_RxQx_Control register to 1.

Impacted Use Cases:

When all the following conditions are met in the use case:

1. “Weighted Round Robin” arbitration scheme is selected by programming Bit[1] of the DMA_Mode register to 0
2. Programming different weights in the TXPR and PR fields of DMA_Mode register.
3. Only One Queue and One DMA is enabled.
4. Both Tx and Rx DMAs are simultaneously requesting for access.

Consequence:

The expected QoS (Quality of Service) requirement between Tx and Rx DMA Channels for host bus bandwidth allocation might not get adhered to. This defect might have an impact only if the host bus bandwidth is limited and less than or a little more than the total Ethernetline rate traffic. The impact can be in terms of Buffer Underflow (in TX when it is cut-through mode) or Buffer overflows (in RX). If the host side bandwidth is much more than the Ethernet line rate traffic, then this bandwidth allocation of WRR scheme is of no consequence.

Workaround:

Operate in Fixed Priority arbitration mode (DA=1) with Rx DMA having higher priority over Tx (TXPR=0). Operate the Tx buffers in Store-and-Forward mode to avoid any buffer Underflows/Overflows.

GETH_AI.003 Header-Payload Split Function Does Not Support IPv6 Packets Received With Zero TCP Payload

The header-payload split function identifies the boundary between the TCP/IP header bytes and the payload of TCP/UDP and stores the header and payload data in separate buffers in the host memory.

However, when TCP/IPv6 packets are received with a IPv6 Payload Length that is equal to IPv6 header length (including extensions) plus TCP header size, the DWC_EQOS does not separate the boundary and forwards the complete packet to the Header buffer. This is because, the header-payload split function

is triggered only when the IPv6 payload length field is greater than the IPv6 header length with extension fields, plus TCP header size (with at the least 1 byte of TCP payload).

Impacted Use Cases:

Received TCP/IPv6 packets that have only TCP header (and no TCP payload).

Consequence:

The dummy TCP payload and the Ethernet FCS bytes are stored in the Header buffer instead of getting stored in a separate payload buffer. Therefore, if the header buffer is mapped to a cache/faster memory, unnecessary dummy payload (if present) is stored in the cache.

There is no functional impact because the TCP stack does not forward the null-payload to the upper layer.

Workaround:

None required as there is no functional impact. TCP stack does not forward null payload to the upper layer.

GETH_AI.005 Application Error Along With Start-of-Packet Can Corrupt the Ongoing Transmission of MAC Generated Packets

On the MAC Transmit interface (MTI), if an error indication (mti_err_i) is asserted along with the Start of Packet (mti_sof_i) of a new packet while the MAC is transmitting an internally generated packet (ARP, PTO, Pause), the error indication aborts the ongoing transmission prematurely. This abort corrupts the MAC generated packet being transmitted. This defect manifests because the mti_err_i is inadvertently passed to the MAC transmitter logic directly when sampled along with mti_sof_i.

The scenarios that cause mti_sof_i and mti_err_i to be asserted together in non-Core configurations are:

1. DMA Configurations: Bus error on the first beat of frame data read from the application.

2. MTL Configurations: ati_error_i asserted along with ati_sof_i on the ATI interface.

Impacted Configurations:

Bus Error / ATI Error / MTI Error received from the system along with the first beat of packet data, manifesting as mti_sof_i and mti_err_i getting asserted together on MTI interface of MAC core, when a MAC generated packet is in transmission.

Consequence:

The MAC generated packet is sent on the line as a runt frame with corrupted FCS. The aborted packet is not retransmitted and can cause

a) Failure of intended flow control in case of PAUSE/PFC packet corruption

b) Delay in ARP Handshake from ARP Offload Engine; The ARP stack recovers because it sends ARP requests periodically

c) Delay in PTP Response/SYNC packets generated by PTP Offload Engine; The PTP stack recovers because it sends request packets periodically.

The probability of occurrence of an application error on the first beat of data and coinciding with a MAC generated packet transmission is very low.

Workaround:

No software workaround possible.

GETH_AI.006 Incorrect IP Header or Payload Checksum Status Given After MTL TX FIFO Flush

When Transmit Checksum Engine is enabled, it generates the IP Header error (IHE) and Payload checksum error (PCE) bits in the status given back to application after the packet is transmitted. These fields are written into a small FIFO when a packet is transferred from MTL to MAC. These bits are read from the FIFO and combined with the Tx status received from the MAC and given back to the application (ATI or DMA descriptor).

When a MTL Tx FIFO Queue flush is initiated for a different queue than the one from which the current packet is being transmitted, a dummy Tx status with flush indication is sent from MTL for the flushed packets. This can happen out of order with respect to the MAC status of the ongoing transmitted packet, when the queues are different.

However, due to this defect, the Checksum Engine status bits are incorrectly read out from the small FIFO along with the transfer of the dummy Tx status of the flushed queue and forwarded to the application. Later, when the MAC Tx status of the ongoing packet arrives, as the FIFO has already been read out, it returns zeros for checksum status fields instead of the actual values.

Impacted Use Cases:

When multiple transmit queues with Checksum offload engines are enabled, Drop Tx Status is not enabled (DTXSTS = 0 in MTL_Operation_Mode register) and a Flush TxQueue is given to a queue (FTQ = 1 in MTL_TxQ(#i)_Operation_Mode register) other than the one from which the ongoing packet transmission is taking place.

Consequence:

Incorrect checksum engine error status might be given for the packet under transmission (and/or the next one) at the time of Flush event in another queue.

Note: The checksum error status bits are normally used for debug purpose by the software driver. Therefore, there is no impact to normal operation.

Workaround:

None. CRC offload engine signals wrong debug status.

GETH_AI.007 IEEE 1588 Timestamp Interrupt Status Bits are Incorrectly Cleared on Write Access to the CSR Register with Similar Offset Address

When RCWE bit of MAC_CSR_SW_Ctrl register is set to 1, all interrupt status bits (events) are cleared only when the specific status bits are written with 1'b1.

However, due to the defect, the Status bits[9:0] of MAC_Timestamp_Status register at address 0x0b20 are unintentionally cleared when 1'b1 is written to

the corresponding bit positions in any CSR register with address offset [7:0] = 8'h20.

The Status bits[9:0] correspond to the following events:

- Target time interrupt (TSTARGT[n] where n= 0, 1, 2, 3)
- Timestamp Seconds Register Overflow interrupt (TSSOVF)
- Target time programming error interrupt (TSTRGTERR[n] where n= 0, 1, 2, 3)

This defect is because, address bits[11:8] are not used in decoding the select signal that is used to clear the status bits, when 1'b1 is written to that bit.

Impacted Use Cases:

Software enables the write 1 to clear interrupt status bits, by setting RCWE = 1 in MAC_CSR_SW_Ctrl register.

Consequence:

When any of the Target Time Interrupts or Timestamp Seconds overflow events occur, the software might inadvertently clear the corresponding status bits (and the interrupt gets de-asserted), if it first writes to any CSR register at the shadow address (0x0_xx20 or 0x1_xx20). Consequently, the Interrupt Service Routine might not identify the source of these interrupt events, as the corresponding status bits are already cleared.

Note: The Timestamp Seconds Register Overflow event is extremely rare (once in ~137 years) and the Target Time Error interrupt can be avoided by appropriate programming. The frequency of Target Time reached interrupt events depends on the application usage.

Workaround:

When RCWE = 1 and Timestamp event interrupts are enabled, process and clear the MAC Timestamp Interrupt events first in the Interrupt Service Routine software, so that write operations to other shadow CSR registers are avoided.

GETH_AI.008 Application Error Along with Start-of-Packet Can Corrupt the FCS Field of the Previous Frame in the MAC Pipeline

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.

GETH_AI.009 Corrupted Rx Descriptor Write Data

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 RxFIFO.

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.

GETH_AI.010 Fatal Bus Error Interrupt Might Be Generated for Incorrect DMA Channel

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

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

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

Impacted Use Cases:

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

Consequence:

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

Workaround:

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

GETH_AI.011 Receive Queue Overflow at End of Frame Along with SPRAM Read-Write Conflict Can Cause Data Loss

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.

GETH_AI.012 Incorrect Flexible PPS Output Interval When Fine Time Correction Method is Used

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.

GETH_AI.013 False Dribble and CRC Error Reported in RMII PHY 10Mbps Mode

The MAC receiver clock is derived synchronously from RMII REF_CLK, the frequency is 2.5MHz in 10Mbps speed mode and 25MHz in 100Mbps 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 10Mbps speed mode, when the RMII CRS_DV is asserted two RMII REF_CLK rising edges ahead of MAC receiver clock, the MAC reports false dribble and CRC error in the Receive status. The dribble error is reported when MAC receives odd number of nibbles (4-bit words) and CRC error is additionally reported. In this case the additional nibble captured is a repetition of the last valid nibble. The MAC forwards only the data received on byte boundaries to the software and ignores the extra nibble. Therefore, there is no data loss or corruption of packet forwarded to the software. However, if error-packet drop is enabled (FEP bit in MTL_RxQ(#i)_Operation_Mode register is set to 0), MAC drops the packets, causing packet loss and impacts the performance.

Impacted Use Cases:

The RMII PHY interface is enabled for 10Mbps operation and RMII CRS_DV is asserted two RMII REF_CLK rising edges ahead of MAC receiver clock.

Consequence:

The MAC reports false dribble and CRC error in Receive status. If error-packet drop is enabled, (FEP bit in MTL_RxQ(#i)_Operation_Mode register is set to 0), MAC drops the packets causing packet loss and impacts the performance. If the error-packet drop is disabled (FEP bit in MTL_RxQ(#i)_Operation_Mode

register is set to 1), MAC forwards the packet to the software, up to the byte boundary, and there is no data loss or corruption.

Workaround:

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

GETH_AI.014 Receive DMA Channel Generates Spurious Receive Watchdog Timeout Interrupt

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).

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.

GETH_AI.015 MAC Receive VLAN Tag Hash Filter Always Operates in Default Mode

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.

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.

GETH_AI.016 Receive DMA Header Split Function Incorrectly Overruns the Allocated Header Buffer

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).

GETH_AI.017 Carrier-Sense Signal Not Generated When False Carrier Detected in RGMII 10/100 Mbps Mode

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

However, due to the defect, when false carrier is detected in RGMII 10/100Mbps 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_B).

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.

GETH_AI.018 Description of the Transmit Checksum Offload Engine - Documentation update

In GETH chapter “Description of the Transmit Checksum Offload Engine” in the TC3xx User’s Manual, the text and formula shall be replaced by the updated description below.

Update to chapter “Description of the Transmit Checksum Offload Engine”

The checksum offload engine module supports two types of checksum calculation and insertion. The checksum engine can be controlled for each packet by setting the CIC bits (TDES3 Bits[17:16]).

The checksum for TCP, UDP, or ICMP is calculated over a complete packet, and then inserted into its corresponding header field. Because of this requirement, when this function is enabled, the Tx FIFO automatically operates in the store-and-forward mode even if the DWC_ether_qos is configured for Threshold (cut-through) mode.

You must make sure that the Tx FIFO is deep enough to store a complete packet before that packet is transferred to the MAC transmitter. The reason being that when space is not available to accept the programmed burst length of data, then the MTL Tx FIFO starts reading to avoid dead-lock. In such a case, the COE fails as the start of the packet header is read out before the payload checksum can be calculated and inserted. Therefore, you must enable the checksum insertion only in the packets that are less than the number of bytes, given by the following equation:

- Packet size < TxQiSize - ((DMA_CHi_TX_Control.TxPBL + 7) * 4),
  – where TxQiSize is configured using MTL_TxQi_Operation_Mode.TQS

GETH_TC.001 Reference clock for Time Stamp Update logic is f_{GETH}

When using PTP (Precision Time Protocol), take into account the following specification delta:

- Unlike described in table “Clock Lines of Ethernet MAC” in the Appendix to the TC3xx User’s Manual, the PTP reference clock clk_ptp_ref_i (Reference Clock for the Time Stamp Update Logic) is **not** connected to f_{SRI}.
- Instead clk_ptp_ref_i is connected to f_{GETH}:
  – clk_ptp_ref_i = f_{GETH} = AHB master interface clock.

As this results in a different reference frequency, clock dividers for the PTP time stamp engine and the achievable time stamp precision need to be calculated on the basis of f_{GETH}.

GETH_TC.002 Initialization of RGMII interface

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 180° phase shift of the Transmit Clock TXCLK relative to TXD/TXCTL.

Note: For MII and RMII see Application Hint GETH_AI.H001.

Workaround

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

- Finish active transfers and make sure that transmitters and receivers are set to stopped state:
  – Check the RPSx and TPSx status bit fields in register DMA_DEBUG_STATUS0/1.

  - Check that MTL_RXQ0_DEBUG, MTL_RXQi_DEBUG, MTL_TXQ0_DEBUG and MTL_TXQi_DEBUG register content is equal to zero.
    Note: it may be required to wait 70 f_{SPB} cycles after the last reset before checking if RXQSTS in MTL_RXQ0_DEBUG and MTL_RXQi_DEBUG are zero.
  - Wait until a currently running interrupt is finished and globally disable GETH interrupts.
- Ensure GETH_GPCTL.EPR = 000_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.

GTM_AI.254 TIM TDU: TDU_STOP=b101 not functional

Stop counting of register TO_CNT on an tdu_word_event or stop counting of TO_CNT1 on a tdu_frame_event is not possible.

Scope

TIM

Effects

TO_CNT1, TO_CNT can not be stopped counting.

Workaround

No workaround available.

GTM_AI.262 DPLL: PSSC behavior in mode change (DPLL_CTRL_0.RMO = 0 ->1)

When changing from normal mode to emergency mode (DPLL_CTRL_0.RMO = 0 ->1) the RAM1b.PSSC value is not calculated like described in the specification.

The PSSC value is not updated at the following trigger slope but at the following STATE slope with the value PSSC=PSTC.

Scope

DPLL

Effects

When changing from normal mode to emergency mode (DPLL_CTRL_0.RMO = 0- >1) the RAM1b.PSSC value is not calculated like described in the specification.

The PSSC value is not updated at the following trigger slope but at the following STATE slope with the value PSSC=PSTC.

The specification showed that the sentence “For changing from normal mode to emergency mode at the following TRIGGER slope” is not helpful because one must expect when changing the RMO bit that there are problems with the next trigger slope so that implementing a modification like described in the specification seems not the right thing to do.

Specification will be modified towards observed behavior.

Workaround

If possible leave PSSC as is.

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

GTM_AI.263 DPLL: DPLL_STATUS.LOCK1 flag (0 ->1) delayed after direction change when DPLL operating in DPLL_CTRL_0.RMO = 1

The DPLL_STATUS.LOCK1 flag does not behave like requested in the specification:

When the DPLL is operating in emergency mode (DPLL_CTRL_0.RMO = 1) and a direction change happens the lock1 flag is reset to “0” as described in the specification.

The problem is, that the LOCK1 bit is set to “1” again after 4 detected gaps (missing state irq, ms -flag) and not as requested after 2 subsequent gaps.

Scope

DPLL

Effects

When the DPLL is operating in emergency mode (DPLL_CTRL_0.RMO = 1) and a direction change happens the LOCK1 flag is reset to “0” as described in the specification.

The problem is, that the LOCK1 bit is set to “1” again after 4 detected gaps (missing state irq, ms -flag) and not as requested after 2 subsequent gaps.

Workaround

If you need to use this information one could observe the missing state interrupt to check for the correct point in time when the LOCK1 flag should be set again.

If the use of the LOCK1 flag is not time critical it could be used as is with the latency described above.

GTM_AI.298 TOM/ATOM: wrong output behaviour in SOMP oneshot mode when oneshot pulse is triggered by TIM_EXT_CAPTURE(x)

If TOM/ATOM is configured in SOMP oneshot mode (OSM = 1) and the oneshot trigger is configured to TIM_EXT_CAPTURE(x) (OSM_TRIG = 1, EXT_TRIG = 1) the output behaviour is not as expected depending on the selected CMU clock.

1. If the selected CMU clock is configured to sys_clk (ATOM: CMU_CLK_[z]_CTRL = 0, TOM: CMU_FXCLK0 used) no initial oneshot period (CN0 is set to zero and then counts until CN0 >= CM0) is executed and the output is set to SL immediately and not as expected after the first initial period.
2. If the selected CMU clock is configured to CMU_CLK_[z]_CTRL > 0 (ATOM)/CMU_FXCLK[1..n] (TOM) then an initial period is executed but the output is set immediately to SL and not as expected when the second oneshot period starts.

Scope

TOM/ATOM SOMP oneshot mode

Effects

The TOM/ATOM output is set immediately to SL and not as expected with a delay of the first initial oneshot period.

Workaround

For GTM generation v3 following workaround is possible:

Use up/down counter mode (UDMODE > 0) instead of up counter mode (UDMODE = 0).

It has to be taken into account that in up/down counter mode the oneshot cycles ends if the counter CN0 counts down and value zero is reached.

A second trigger of TIM_EXT_CAPTURE(x) in the up counting phase will be ignored but a second trigger while the counter CN0 counts down will trigger the next oneshot cycle, which will be executed directly afterwards without the initial period.

GTM_AI.299 TOM/ATOM: wrong output behaviour in SOMP oneshot mode when oneshot pulse is triggered by trig_[x-1]

If TOM/ATOM is configured in SOMP oneshot mode (OSM = 1) and the oneshot trigger is configured to trigger signal from trigger chain trig_[x-1] (OSM_TRIG = 1, EXT_TRIG = 0) the output signal is set immediately to SL and not as expected after a delay of the first initial oneshot period (CN0 counts from 0 until it reaches the value of CM0). The first initial oneshot period isn't executed.

Scope

TOM/ATOM SOMP oneshot mode

Effects

The TOM/ATOM output is set immediately to SL and not as expected with a delay of the first initial oneshot period.

Workaround

For GTM generation v3 and later following workaround is possible:

Use up/down counter mode (UDMODE > 0) instead of up counter mode (UDMODE = 0).

It has to be taken into account that in up/down counter mode the oneshot cycles ends if the counter CN0 counts down and value zero is reached.

A second trigger of from trigger chain by trig_[x-1] in the up counting phase will be ignored but a second trigger while the counter CN0 counts down will trigger the next oneshot cycle, which will be executed directly afterwards without the initial period.

GTM_AI.300 DPLL: Change to forward operation when DPLL_THMI is set to zero does not work correctly

If direction control is set up via the TRIGGER input signal (DPLL_CTRL_1.IDDS=0, DPLL_CTRL_1.SMC=0) and DPLL_THMI is set to zero the direction does not change to forward (BWD1=0) when the current

direction is backward (BWD1=1). Instead, when DPLL_THMI=0, the direction set latest is hold.

Scope

DPLL

Effects

DPLL direction does not change to forward (BWD1=0) if DPLL_THMI is set to 0. The current status of the direction is hold that means in case of BWD1=0 the direction will stay in forward (BWD1=0), in case of BWD1=1 the direction stays at backward (BWD1=1).

Workaround

- DPLL_CTRL_1.IDDS=0:
  - If the DPLL is operating in forward direction (BWD1=0) the direction can be kept by setting DPLL_THMI=0.
  - If the DPLL is operating in backward direction the direction can be switched to forward by setting the DPLL_THMI value to the biggest possible value DPLL_THMI=0x00FFFF. This should set the direction back to forward.
- Use different mechanism of direction control DPLL_CTRL_1.IDDS=1:
  - In this case the direction can be controlled by setting the TIM0_IN6 input signal of the GTM when MAP_CTRL.TSEL=0.

In both cases the direction evaluation is done with the inactive edge of the TRIGGER input signal. The TRIGGER input signal must be active even in emergency mode to handle the direction changes correctly. If the TRIGGER input signal is not in a usable condition the necessary input signal sequence can be generated by a direct modification of the input signal of TIM0_CH0 with the use of TIM[0]_IN_SRC.MAKE_0/VAL_0 and TIM[0]_CH[0]_ECTRL.USE_LUT (GTM v3.1.5 additionally).

GTM_AI.301 DPLL: Reset of DPLL_STATUS.BWD1=1 by disabling the DPLL does not cause the direction to change from backward to forward in any case

The issue occurs when the DPLL is operating in normal mode (DPLL_CTRL_0.RMO=0, DPLL_CTRL_1.SMC=0) and the direction of the trigger signal is evaluated in the mode DPLL_CTRL_1.IDDS=0 (input direction is detected comparing the THMI value with the duration between active and inactive slope of TRIGGER). If in this configuration a direction change happens on the trigger signal which is not plausible, because the direction change happens due to e.g. a disturbed signal, the direction change performed by the DPLL should be removed.

The direction in which the DPLL is operating can be read out by the status register DPLL_STATUS.BWD1. To disable the DPLL by setting DPLL_CTRL_1.DEN = 1->0->1 is resetting the BWD1 bit but this does not remove the direction change in every case and the BWD1 bit could be set to the unwanted direction again. The issue occurs when the DPLL has not received an active input signal on the STATE input such that DPLL_STATUS.fsd=0 before the DPLL is disabled (den=1->0->1) and switched to emergency mode (DPLL_CTRL_1.RMO=1). The issue does not occur if the DPLL is in the status of DPLL_STATUS.fsd=1 or if the DPLL is not switched to emergency mode (DPLL_CTRL_1.RMO=0) after the DPLL has been disabled/enabled.

Scope

DPLL

Effects

DPLL internal direction remains in current direction while DPLL_STATUS.BWD1 bit is reflecting it's reset value during a toggle sequence (1->0->1) of the DPLL enable bit DPLL_CTRL_1.DEN. At the end of the toggle sequence the BWD1 bit returns to the state of the current internal direction.

Workaround

If the issue occurs under the described conditions the wrong direction could be corrected by:

1. Adding an additional input signal (active edge followed by inactive edge while not exceeding the THMI limit) to the trigger input which switches the DPLL back to forward direction.
2. Switching to the direction control mode DPLL_CTRL_1.IDDS=1 and to control the direction by setting the GTM input signal TIM0_IN6 to e.g. zero (forward direction). For combustion engine operation and MAP_CTRL.TSEL=0 the TDIR/SDIR signals can be used to control the direction with the TIM0_IN6 input signal. This TIM0_IN6 signal must be set directly on the GTM input pin by the mechanisms provided by the semiconductor supplier who integrated the GTM. This mechanism is bound to the resource of the TIM0_IN6 input channel.

GTM_AI.304 MCS: Scheduling modes Single Prioritization and Multiple Prioritization are not functional

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

Scope

MCS

Effects

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

Workaround

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

GTM_AI.305 TIM Signal Generation with serial shift mode TSSM: If TSSM_OUT is used in channel x and channel x+1 uses edges of FOUT_PREV, these edges show an unexpected delay which lead to a delayed operation of channel measurement or TDU functionality of channel x+1

If channel x operates in TSSM mode and signal generation is active with (CNTS[21:20]!=b00) and channel x+1 uses the signal edges from FOUT_PREV(x+1)

- a) for channel measurements with
  TIM[i]_CH[x+1]_ECTRL.USE_PREV_CH_IN=1
  or
- b) inside the timeout detection unit with
  TIM[i]_CH[x+1]_ECTRL.USE_PREV_TDU_IN=1

The actions in TDU or channel measurement triggered by edges on FOUT_PREV(x+1) will occur with unexpected delay. (Delay correlates to shift clock in channel x).

Scope

TIM TSSM mode

Effects

Channel measurements or TDU functionality in channel x+1 triggered with unexpected delay.

Workaround

Using the LUT in the filter unit in channel x+1 ensures that the signal edge information of FOUT_PREV(x+1) is reconstructed properly on F_OUT[x+1].

Use the following settings:

TIM[i]_CH[x+1]_ECTRL.USE_LUT=b10;

TIM[i]_CH[x+1]_TDUC.TO_CNT2=0xF0;

TIM[i]_CH[x+1]_ECTRL.USE_PREV_CH_IN=0;

TIM[i]_CH[x+1]_ECTRL.USE_PREV_TDU_IN=0.

GTM_AI.306 DPLL: DPLL_NUTC.syn_t_old, DPLL_NUSC.syn_s_old not updated according specification

The DPLL specification defines for DPLL_NUTC.WSYN=1 that an update of register DPLL_NUTC allows writing of the bits DPLL_NUTC.syn_t while DPLL_NUTC.syn_t_old inherits the previous value of DPLL_NUTC_syn_t.

Differing from the specified behavior the actual hardware does not update the value of DPLL_NUTC.syn_t_old with the previous value of DPLL_NUTC.syn_t but instead updates DPLL_NUTC.syn_t_old according to the corresponding bits of the write operation executed by the CPU.

The DPLL specification defines for DPLL_NUTC.WSYN=1 that an update of register DPLL_NUSC allows writing of the bits DPLL_NUSC.syn_s while DPLL_NUSC.syn_s_old inherits the previous value of DPLL_NUSC_syn_s.

Differing from the specified behavior the actual hardware does not update the value of DPLL_NUSC.syn_s_old with the previous value of DPLL_NUSC.syn_s but instead updates DPLL_NUSC.syn_s_old according to the corresponding bits of the write operation executed by the CPU.

Scope

DPLL

Effects

The registers bits DPLL_NUTC.syn_t_old are not updated with the previous value of DPLL_NUTC_syn_t but by the bits of the input data word.

The registers bits DPLL_NUSC.syn_s_old are not updated with the previous value of DPLL_NUSC_syn_s but by the bits of the input data word.

Workaround

If the update of syn_t/s_old shall be done like described in the specification the register DPLL_NU(T/S)C.syn_t/s must be read first, then the DPLL_NU(T/S)C.syn_(t/s) can be used to modify the bits which are written to DPLL_NU(T/S)C.syn_(t/s)_old.

As the current behavior of DPLL_NUT/SC.syn_s/t_old is in use by and can be advantageous for certain applications, there is no intend to change the current

hardware behavior at this point in time. Instead a specification update to align the specification with the current hardware behavior is planned for future GTM generations.

GTM_AI.307 IRQ: AEI_IM_ADDR is not set in GTM_IRQ_NOTIFY register if cluster 0 is disabled

Bit 2 - AEI_IM_ADDR - in register GTM_IRQ_NOTIFY is not set and the related error interrupt (if enabled) does not occur if cluster 0 is disabled and

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

In case BRIDGE_MODE.MSK_WR_RSP = 0x0 (not recommended) an FPI bus error is issued for all write accesses.

Scope

IRQ

Effects

See description above.

Workaround

- a) Do not disable cluster 0.
- b) If cluster 0 is disabled: after each WRITE access check register GTM_AEI_STA_XPT; if the read value is >0, it signs an access error.
- c) If cluster 0 is disabled: do not read from cluster 0 register or any other disabled cluster register.

GTM_AI.308 TIM, ARU: Limitation that back-to-back TIM data transfers at full ARU clock rate cannot be transferred correctly with ARU dynamic routing feature

If TIM input signals with signal changes faster or equal than ARU clock rate are processed with the TIM and the results are routed via ARU in dynamic routing

mode, it is likely that there is a data loss and only each second data can be transferred.

Scope

ARU Routing, DEBUG signal interface

Effects

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

Workaround

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

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

GTM_AI.309 TIM Signal Generation with serial shift mode TSSM in channel x: Generated TSSM_OUT signal used in lookup table of inputsrc module of channel x has unpredictable delay

If channel x operates in TSSM mode and signal generation is active with (CNTS[21:20]!=b00) and channel x uses the signal TSSM_OUT(x) in the lookup table with USE_LUT(x)=0b11.

Results of lookuptable function will behave unexpected due to delayed input of TSSM_OUT(x). (Delay correlates to shift clock in channel x)

Scope

TIM TSSM mode

Effects

Lookup table in TIM channel inputsrc module shows unexpected results.

Workaround

Use lookup table of inputsrc module channel x+1. The TSSM_OUT signal of channel x which is routed via FOUT_NEXT(x) to channel x+1 can be used with USE_LUT(x+1)=0b10.

GTM_AI.318 MCS: NARD(I) instruction terminates unexpectedly

If the bit field CFG_CLK_RATE of register CCM[i]_HW_CONF is set and an MCS runs on a cluster i while bit field CLSc_CLK_DIV of register GTM_CLS_CLK_CFG is set to 1, the execution of a NARD or NARDI instruction might signalize an unsuccessful data transfer (bit field SAT of internal MCS register STA is cleared) although the data source has data available for sending. The unexpected behavior depends on the current state of the internal ARU counter.

Scope

MCS

Effects

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

Workaround

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

GTM_AI.319 (A)TOM: Unexpected (A)TOM_CCU1TCx_IRQ in up/down counter mode

If the up-down counter mode is activated (bit field UDMODE of register (A)TOM[i]_CH[x]_CTRL is set to a value greater than zero) and the interrupt (A)TOM_CCU1TCx_IRQ is enabled (bit field CCU1TC_IRQ_EN of register (A)TOM[i]_CH[x]_IRQ_EN is set), the interrupt signal (A)TOM_CCU1TCx_IRQ will be set unexpectedly directly after the interrupt (A)TOM_CCU0TCx_IRQ

was set and indicates that the counter (A)TOM[i]_CH[x]_CN0 reaches (A)TOM[i]_CH[x]_CM0.

Scope

TOM/ATOM

Effects

Interrupt signal (A)TOM_CCU1TCx_IRQ is set unexpectedly.

Workaround

If the interrupt (A)TOM_CCU1TCx_IRQ is needed, it can be disabled by the first occurrence of itself and enabled again with the interrupt (A)TOM_CCU0TCx_IRQ. With the following occurrence of the interrupt (A)TOM_CCU1TCx_IRQ it will be disabled again and so on.

GTM_AI.320 ATOM: Unexpected restart of a SOMS oneshot cycle while ATOM[i]_CH[x]_CM0 is zero

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

Scope

ATOM

Effects

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

Workaround

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

GTM_AI.322 DPLL: PSTC, PSSC not updated correctly after fast pulse correction completed (DPLL_CTRL1.PCM1/2 = 0)

When additional pulses are requested using DPLL_CTRL_11.PCMF1/2=1 AND PCMF1/2_INCCNT_B=0 the PSTC/PSSC parameters as well as NMB_T/S_TAR are not updated correctly, because either the amount of additional pulses (MPVAL1/2) are not incremented or NMB_T/S_TAR is set to a wrong value.

Scope

DPLL

Effects

After the pulse correction is performed the fields NMB_T/S_TAR are set to wrong values such that after a new input event the parameters PSTC/PSSC are not updated correctly.

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

The pulse generation itself (register DPLL_INC_CNT1/2 and the status of the angle clocks TBU_TS1/2) is correct and not affected by this issue.

Workaround

Implement the workaround in the TISI interrupt, i.e. start the workaround at the arrival of inactive edge. This ensures that swon_t/swon_s is stable and the incorrect nmb_t_tar/nmb_s_tar has already been generated. This is to ensure the following:

- Start the workaround after the incorrect nmb_t_tar/nmb_s_tar has been generated and swon_t/swon_s is not toggling anymore
- Workaround should be finished before the arrival of next active edge.

Workaround steps are as follows:

1. Check swon_t/swon_s,
   - If swon_t/swon_s = 1, save & use nmb_t_tar/nmb_s_tar for further corrections
   - Else save & use nmb_t_tar_old/nmb_s_tar_old for further corrections.
2. Set PCM1=1 to trigger the fast pulse correction with PCMF1 already set to 1
3. Wait for PCM1 to reset to 0
4. Overwrite the nmb_t_tar/nmb_s_tar or nmb_t_tar_old/nmb_s_tar_old with the correct value based on swon_t/swon_s, similarly based on the choice in step 1.

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

GTM_AI.323 DPLL: Registers DPLL_NUTC.SYN_T and DPLL_NUSC.SYN_S are updated by the profile (ADT_T.NT/ADT_S.NS) before the DPLL is synchronized (DPLL_STATUS.SYT/S=0)

The registers DPLL_NUTC.SYN_T and DPLL_NUSC.SYN_S as well as the corresponding *_OLD registers are updated unexpectedly by the profile (ADT_T.NT/ADT_S.NS) before the DPLL is synchronized (DPLL_STATUS.SYT/S=0).

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

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

Scope

DPLL

Effects

When the DPLL is enabled and before the DPLL is synchronized (by writing to the relevant pointers (DPLL_APT_2c/DPLL_APS_1C3) the

DPLL_NUTC.SYN_T/DPLL_NUSC.SYN_S registers are unexpectedly updated by the profile.

Because the SYN_T_OLD and SYN_S_OLD registers are updated by SYN_T, SYN_S they are affected as well.

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

Workaround

When DPLL_NUTC.SYN_T/_OLD, DPLL_NUSC_SYN_S/_OLD values are needed outside the DPLL it must be checked that the DPLL is already synchronized (DPLL_STATUS.SYT/SYS). When the relevant DPLL channel (TRIGGER/STATE) is not synchronized yet the SYN_T/S values should be taken into account as “1”.

GTM_AI.325 TIM: Bits ACB[2:1] lost on interface to ARU (always zero)

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

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

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

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

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

Scope

TIM, ARU transfers

Effects

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

Workaround 1

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

Workaround 2

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

Workaround 3

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

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

Workaround 4

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

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

Workaround 5

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

GTM_AI.326 TIM: ARU bit ACB[0] (signal level) incorrect in case a second ARU request occurs while the actual request is just acknowledged

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

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

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

Scope

TIM, ARU transfers

Effects

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

Workaround 1

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

Workaround 2

Use the signal level information embedded in the ARU data words (selectable by ECNT/TIM_INP_VAL).

This data will show the correct signal level.

GTM_AI.329 Interference of MCS to AEI/ADC and CPU to AEI traffic within the same cluster could result in incorrect MCS program execution

Scope

Usage of MCS AEI master port (AEI and ADC communication from MCS); MCS channel code execution; Dynamic usage of GTM_MCS_AEM_DIS.

Effects

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

Description

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

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

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

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

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

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

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

Table 9 Memory locations critical from MCS (N=1 wait cycle)

Register name or memory location
GTM_RST
GTM_CLS_CLK_CFG
BRC_RST
TIM[i]_RST
TOM[i]_TGC0_GLB_CTRL
TOM[i]_TGC1_GLB_CTRL
DPLL_CTRL_1
ATOM[i]AGC_GLB_CTRL

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

Table 10 Memory locations critical from CPU/DMA and MCS (N>1 wait cycles)

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

Technical Background

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

Depending on the addressed target of an AEI bus access, the access will complete either within one bus cycle (N=0 wait cycle), within 2 bus cycles (N=1 wait cycle) or multiple bus cycles (N>1 wait cycles). The vast majority of AEI accessible resources will respond with N=0 wait cycles. For a list of resources with N=1 or N>1 see **Table 9** and **Table 10** above.

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

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

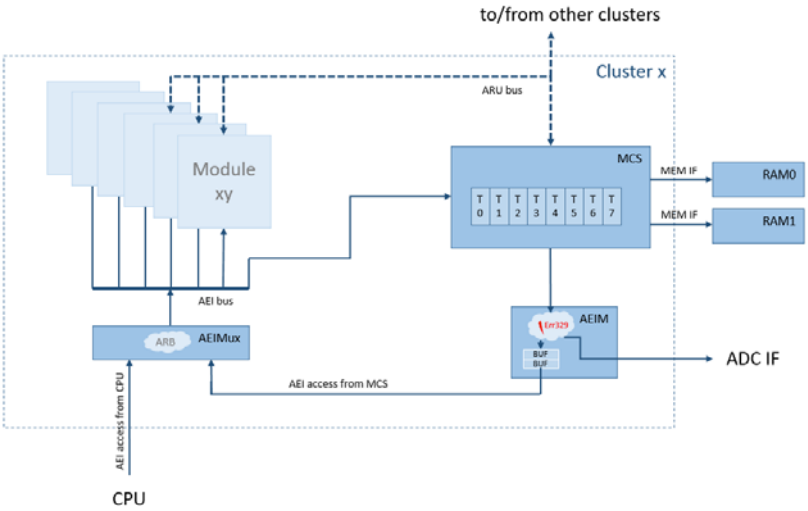

Figure 4 Cluster-internal AEI network

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

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

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

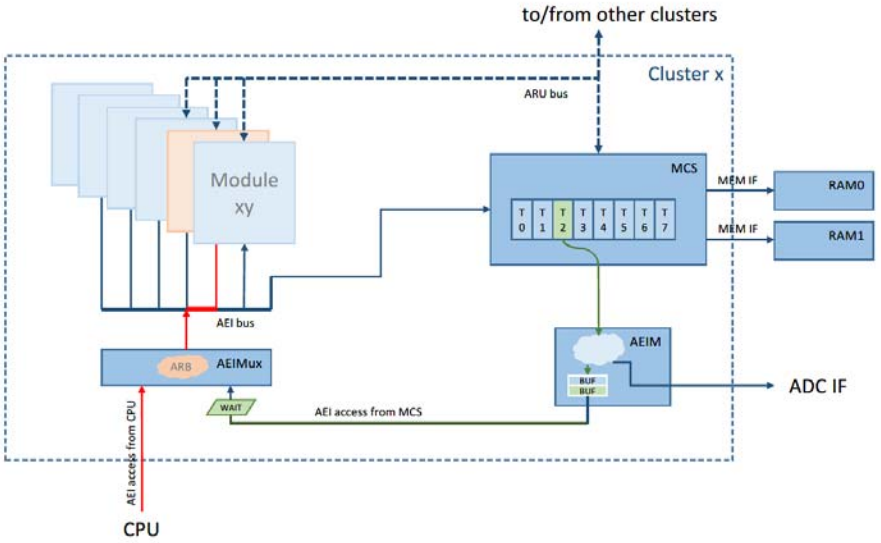

Figure 5 MCS AEI master access path

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

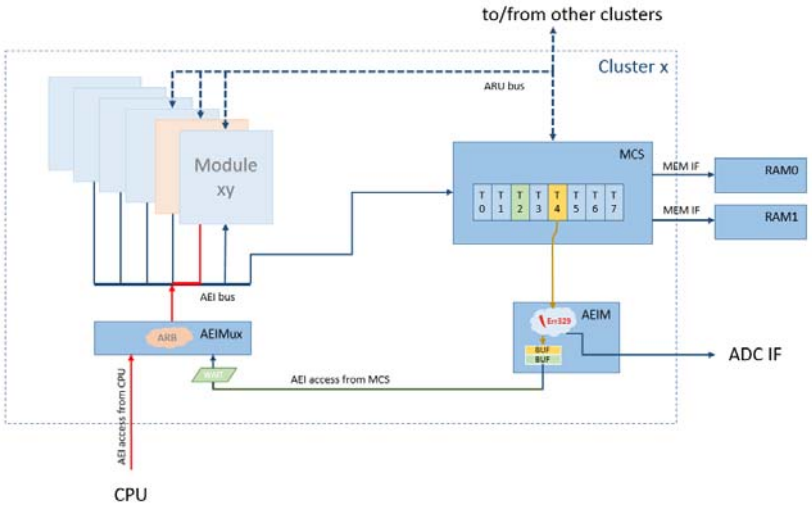

Figure 6 Critical MCS AEI master scenario

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

Workaround

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

Option 1:

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

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

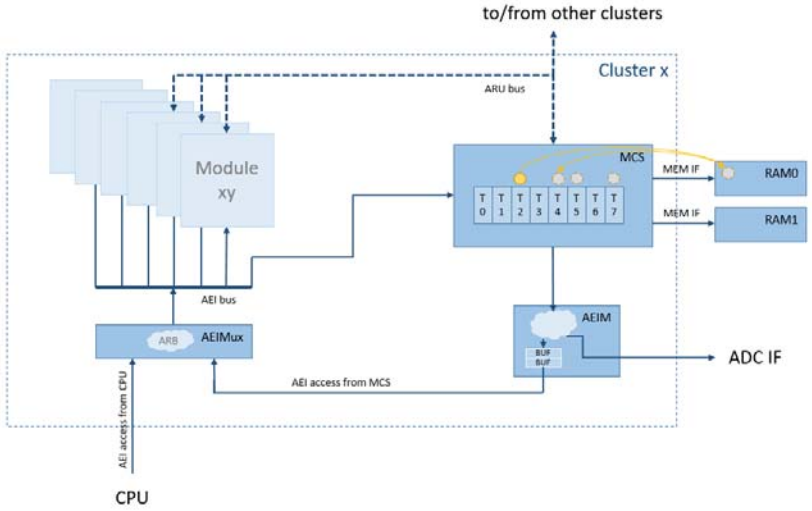

Figure 7 MCS token mechanism

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

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

Option 2:

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

Option 3:

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

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

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

Table 11 Summary of problem GTM_AI.329

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

GTM_AI.331 GTM_TIM[i]_AUX_IN_SRC and GTM_EXT_CAP_EN_[i] register: wrong status 2 by AEI write access if cluster 0 is disabled

AEI write access to the register GTM_TIM[i]_AUX_IN_SRC and GTM_EXT_CAP_EN_[i] via the legacy address space responds with status 2 even though the write operation is correctly executed and the register contains the correct new value.

Scope

Register access via legacy address.

Effects

If status 2 is responded an interrupt bit in GTM_IRQ_NOTIFY is set and the write address will be caught in register GTM_AEI_STA_XPT.

Any further AEI access with responded status >0 will not be caught in GTM_AEI_STA_XPT until GTM_AEI_STA_XPT is reset by AEI read to GTM_AEI_STA_XPT.

Workaround

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

GTM_AI.332 Access to registers GTM_TIM[i]_AUX_IN_SRC and GTM_EXT_CAP_EN_[i] via legacy address space: read data always 0 for AEI read access while cluster 0 is disabled

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

- Impacted registers are:
  - GTM_TIM[i]_AUX_IN_SRC
  - GTM_EXT_CAP_EN_[i]

Scope

Register access via legacy address.

Effects

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

Workaround

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

GTM_AI.333 MCS bus master interface: a not word aligned address access to DPLL ram region can cause incorrect execution of MCS channel code

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

Scope

MCS bus interface; MCS program execution.

Effects

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

Workaround

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

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

GTM_AI.334 DPLL RAM content of single address can be corrupted after leaving debug mode

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

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

Scope

GTM Debug

Effects

Data in RAM might be corrupted

Workaround

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

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

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

GTM_AI.335 TOM output signal to SPE not functional if up/down counter mode is configured

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

Scope

TOM - SPE interface

Effects

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

Workaround

No workaround available.

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

GTM_AI.336 GTM Bus Bridge: Incorrect AEI access execution in case the previous AEI access was aborted with the access timeout abort function

In case the GTM internal AEI access timeout abort function is in use (GTM_CTRL.TO_VAL != 0 and GTM_CTRL.TO_MODE=1), a following AEI access can be corrupted:

a) A write access might not be executed (register/ memory not written to the specified value)

b) A read access can return random data (read value does not reflect the content of the addressed register / memory).

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

Scope

CPU interface accesses

Effects

Read access returns random data.

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

Workaround

Do not use the AEI access abort mode, use the observe mode instead (Set GTM_CTRL.TO_MODE=0).

Enable additionally the timeout observe IRQ by setting GTM_IRQ_EN.AEI_TO_XPT_IRQ=1 to invoke higher level recovery mechanisms for GTM re-initialization.

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

GTM_AI.339 DPLL: Control bits DPLL_CTRL_11.PCMF1 and DPLL_CTRL_11.PCMF2 are not reset to 0 after a pulse correction is completed

In DPLL specification it is written in the description of field PCMF1 in register DPLL_CTRL_11: “When taken the MPVAL1 value to RPCUx and INC_CNT1 the PCM1 bit is reset immediately and after that also the PCMF1 bit.”

The implemented behavior of the DPLL is that the PCMF1 bit is not reset after the PCM1 bit is reset to 0. In mode DPLL_CTRL_1.SMC=1, the same is true for the signal DPLL_CTRL_11.PCMF2.

Scope

DPLL

Effects

After a pulse correction is executed by writing to DPLL_CTRL_1.PCM1=1 and this signal is reset to 0 again, the signal DPLL_CTRL_11.PCMF1 is not reset back to 0.

After a pulse correction is executed by writing to DPLL_CTRL_1.PCM2=1 and this signal is reset to 0 again, the signal DPLL_CTRL_11.PCMF2 is not reset back to 0.

Workaround

Before a following pulse correction is executed this signal must be set to 0 again if needed. When a sequence of pulse corrections with the same configuration of DPLL_CTRL_11.PCMF1 or DPLL_CTRL_11.PCMF2 is executed no modification of DPLL_CTRL_11.PCMF1 or DPLL_CTRL_11.PCMF2 is necessary.

When reset of DPLL_CTRL_11.PCMF1 or DPLL_CTRL_11.PCMF2 is needed this can be done by writing to register DPLL_CTRL_11.PCMF1/2.

GTM_AI.340 TOM/ATOM: Generation of TRIG_CCU0/TRIG_CCU1 trigger signals skipped in initial phase of A/TOM SOMP one-shot mode

Configuration in use:

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

Expected behavior:

The generation of one-shot pulses in A/TOM can be initiated by a write to CN0. In this case the pulse generation comprises of an initial phase where the signal level at A/TOM output is inactive followed by a pulse. The duration of the initial phase can be controlled by the written value of CN0, where the duration is defined by CM0-CN0. After the counter CN0 reaches the value of CM0-1, the pulse starts with its active edge, CN0 is reset, and starts counting again. When CN0 reaches CM1-1, the inactive edge of the pulse occurs. Due to the fact, that the capture compare units CCU0 and CCU1 compare also in the initial phase of the pulse generation, the trigger conditions for these comparators apply also in this initial phase. Thus, the TRIG_CCU0 and TRIG_CCU1 signals also occur in the initial phase of the one-shot pulse. When these trigger signals are enabled in the A/TOM[i]_CH[x]_IRQ_EN, an interrupt signal is generated by A/TOM on the CCU0TC and CCU1TC trigger conditions and the corresponding A/TOM[i]_CH[x]_IRQ_NOTIFY bits are set.

Observed behavior:

For certain start values of CN0 and dependent on the history of pulse generation, the trigger signals TRIG_CCU0 and TRIG_CCU1 are skipped. As a consequence, this can led to missing interrupts CCU0TC and CCU1TC on behalf of their missing trigger signals TRIG_CCU0 and TRIG_CCU1.

For the first pulse generation after enabling the channel, all trigger signals TRIG_CCU0 and TRIG_CCU1 appear as expected and described in the section expected behavior. If the channel stays enabled and a new value CN0 is written to trigger a subsequent one-shot pulse, the TRIG_CCU0/TRIG_CCU1

triggers in the initial phases of subsequent one-shot pulses are skipped under the following conditions:

- For TRIG_CCU0 trigger: if the one-shot pulse is started by writing a value to CN0 greater or equal to CM0-1.
- For TRIG_CCU1 trigger: if the one-shot pulse is started by writing a value to CN0 greater or equal to CM1-1.

Scope

TOM/ATOM

Effects

Missing TRIG_CCU0 and TRIG_CCU1 trigger signals in initial phase of subsequent pulses in A/TOM one-shot mode, when one shot-mode is started with writing to CN0 values greater equal CM0-1 or CM1-1.

Workaround 1

Disabling, resetting (channel reset), re-enabling and initializing of the channel between each one-shot pulse will ensure the correct behavior of CCU0TC and CCU1TC interrupt source.

Workaround 2

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

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

GTM_AI.341 TOM/ATOM: False generation of TRIG_CCU1 trigger signal in SOMP one-shot mode with OSM_TRIG=1 when CM1 is set to value 1

Configuration in use:

- A/TOM[i]_CH[x]_CTRL.OSM=1
- A/TOM[i]_CH[x]_CTRL.OSM_TRIG=1

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

Expected behavior:

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

After the counter CN0 reaches the value of CM0-1, the pulse starts with its active edge, CN0 is reset, and starts counting again. When CN0 reaches CM1-1, the inactive edge of the pulse occurs. Due to the fact, that the capture compare units CCU0 and CCU1 compare also in the initial phase of the pulse generation, the trigger conditions for these comparators apply also in this initial phase. Thus, the TRIG_CCU0 and TRIG_CCU1 signals also occur in the initial phase of the one-shot pulse. When these trigger signals are enabled in the A/TOM[i]_CH[x]_IRQ_EN, an interrupt signal is generated by A/TOM on the CCU0TC and CCU1TC trigger conditions and the corresponding A/TOM[i]_CH[x]_IRQ_NOTIFY bits are set.

Observed behavior:

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

- The first observed behavior is that the capture compare unit doesn't generate the TRIG_CCU1 trigger signal in the initial phase of the one-shot cycle.
- The second observed behavior is that at the end of the operation phase of the one-shot cycle, where CN0 reaches CM0-1 a second time, the capture compare unit generates a TRIG_CCU1 trigger signal which is not expected at this point in time.

Scope

TOM/ATOM

Effects

Missing TRIG_CCU1 trigger signal in initial phase of the one-shot cycle and unexpected TRIG_CCU1 trigger signal at the end of the operation phase of the one-shot cycle.

Workaround

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

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

Example for TOM: Instead of using CMU_FXCLK(1), which has the divider value 2**4, use CMU_FXCLK(0), which has the divider value 2**0. In this case, CM1 has to be configured with value 2**4 minus 2**0 which is equal to 2**4=16.

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

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

GTM_AI.344 DPLL: Incorrect AEI_STATUS on internal MCS2DPLL interface on valid and implemented address accesses

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

Scope

DPLL, MCS0

Effects

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

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

Workaround

The register bit field MCS0_CTRL_STAT.HLT_AEIM_ERR must be set to “0b0” to prevent the MCS0 channels from halt.

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

GTM_AI.345 SPE: Incorrect behaviour of direction change control via SPE_CMD.SPE_CTRL_CMD bits

A direction change (“00” <-> “01”) via SPE_CTRL_CMD disturbs the increment/decrement of the pat_ptr resulting in incorrect output patterns not corresponding to the input pattern position. Changing the direction bit in SPE_CTRL_CMD can also generate invalid IRQs.

Scope

SPE, TOM

Effects

Modifying the direction bit (“00” <-> “01”) in SPE_CTRL_CMD does not provide the correct output pattern to the BLDC motor. Due to a wrong pat_ptr position incorrect output patterns will be sent to the motor, which are not correlated to the sensor position.

In addition the SPE logic can generate unpredictable IRQs (perr_irq, dchg_irq, bis_irq).

Workaround

Do not use SPE_CTRL_CMD.

Instead reprogram the SPE_OUT_PAT register to change the direction.

GTM_AI.346 ATOM SOMS mode: Shift cycle is not executed correctly in case the reload condition is deactivated with ATOM[i]_AGC_GLB_CTRL.UPEN = 0

ATOM is configured to SOMS continuous mode by setting the following configuration bitfields:

- ATOM[i]_CH[x]_CTRL.MODE=11
- ATOM[i]_CH[x]_CTRL.OSM=0
- ATOM[i]_CH[x]_CTRL.ARU_EN=0
- ATOM[i]_AGC_GLB_CTRL.UPEN[x]=0b00

Expected behaviour:

After the counter CN0 reaches CM0, no reload cycle is executed due to the configuration of UPEN=0b00.

Instead of a reload cycle a shift cycle has to be executed to ensure an continuous shifting.

Observed behaviour:

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

Scope

ATOM

Effects

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

Workaround

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

GTM_AI.347 TOM/ATOM: Reset of (A)TOM[i]_CH[x]_CN0 with TIM_EXT_CAPTURE are not correctly synchronized to selected CMU_CLK/CMU_FXCLK

To reset the counter (A)TOM[i]_CH[x]_CN0 (SOMP mode in ATOM), the input signal TIM_EXT_CAPTURE can be used by configuration of (A)TOM[i]_CH[x]_CTRL.EXT_TRIG=1 and (A)TOM[i]_CH[x]_CTRL.RST_CCU0=1.

The reset of the counter (A)TOM[i]_CH[x]_CN0 should happen synchronously to the internal selected CMU clock CMU_CLK/CMU_FXCLK. Therefore a synchronisation stage is implemented to synchronize the input signal TIM_EXT_CAPTURE to the internal selected CMU clock CMU_CLK/CMU_FXCLK.

It can be observed, that the reset of the counter is done immediately with the occurrence of the input signal TIM_EXT_CAPTURE and not as expected synchronously to the selected CMU clock enable CMU_CLK/CMU_FXCLK.

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

Scope

ATOM, TOM

Effects

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

Workaround 1

Select a CMU clock enable signal CMU_CLK/CMU_FXCLK by appropriate setting of (A)TOM[i]_CH[x]_CTRL.CLK_SRC which is setup inside the CMU module in that way, that each system clock is enabled. In other words this means that the selected clock enable signal CMU_CLK/CMU_FXCLK should be always active high.

Note: No frequency divider should be used for CMU_CLKz (only CMU_CLK_z_CTRL.B.CNT = 0) and CMU_FXCLKx (only CMU_FXCLK0).

Workaround 2

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

GTM_AI.348 DPLL: Correction of missing pulses delayed after start of pulse generation

The described erratum occurs in the DPLL configuration DPLL_CTRL_1.DMO=0 (Automatic end mode) and DPLL_CTRL_1.COA=0 (Fast pulse correction). When after the start of pulse generation (DPLL_CTRL_1.SGE1/2=0-->1) not all pulses scheduled could be generated, repeating the pulses at fast speed is not executed at the second TRIGGER/STATE input event.

Scope

DPLL

Effects

When the pulse generation has been started by setting DPLL_CTRL_1.SGE1/2 and not all scheduled pulses could be generated there is no fast pulse correction after the second active input signal. Beyond that the DPLL internal pulse counter DPLL_ICNT1/2 is incremented correctly so that no pulse is getting lost. After the third input event the pulse correction is working as specified.

Workaround 1

DPLL must be in direct load mode (DPLL_CTRL_1.DLM1/2 =1). Set DPLL_ADD_IN_LD1/2.ADD_IN_LD1/2=0 for the first two increments after the DPLL pulse generation has been started by DPLL_CTRL_1.SGE1/2=1 (all GTM Versions)

Workaround 2

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

Workaround 3

Use pulse correction mechanism triggered by DPLL_CTRL_1.PCM1/2:

- Set DPLL_MPVAL1/2.MPVAL1/2 to the desired number of pulses which has to be sent out fast.
- Set DPLL_CTRL_11.PCMF1/2=1 AND DPLL_CTRL_11.PCMF1/2_INCCNT_B=1.
- Trigger the fast pulses by setting DPLL_CTRL_1.PCM1/2=1.

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

GTM_AI.349 TOM-SPE: OSM-Pulse width triggered by SPE_NIPD for selected CMU_FXCLK not correct

The SPE_NIPD signal is used to reset TOM_CH_CN0 and to generate a one-shot pulse. When the CMU_FXCLK of the corresponding TOM_CH is set to a value unequal to 0, there are two effects observed:

1. the first pulse triggered by SPE_NIPD is generated with the CMU_FXCLK(0), while any subsequent pulses are generated with the configured CMU_FXCLK;
2. the pulses generated with the correct CMU_FXCLK show no determinism. Some pulses end with CCU_TRIG1, some with CCU_TRIG0.

Scope

TOM, SPE

Effects

The OSM-Pulse width triggered by SPE_NIPD are not correct.

Workaround

Use SYS_CLK by selecting CMU_FXCLK(0) instead of a value unequal to zero for CMU_FXCLK.

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

GTM_AI.350 TOM-SPE: Update of SPE[i]_OUT_CTRL triggered by SPE_NIPD not working for a delay value 1 in TOM[i]_CH[x]_CM1

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

Expected behaviour:

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

Observed behaviour:

For CM1=1, there is no change of SPE_OUT_CTRL at all, independent of CMU_FXCLK.

Scope

TOM, SPE

Effects

The update of SPE_OUT_CTRL register is not executed.

Workaround

Use SYS_CLK by selecting CMU_FXCLK(0) instead of a value unequal to zero for CMU_FXCLK.

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

GTM_AI.351 MAP: Disable of input lines by MAP_CTRL register not implemented for input signals TSPP0 TIM0_CHx(48) (x=0..2) and TSPP1 TIM0_CHx(48) (x=3..5)

The Control bits TSPP0_I0V, TSPP0_I1V, TSPP0_I2V, TSPP1_I0V, TSPP1_I1V, TSPP1_I2V of register MAP_CTRL are not operating as specified. The specified gating functions of the input signals TIM0_CH0(48), TIM0_CH1(48), TIM0_CH2(48) of TSPP0 submodule and the input signals

TIM0_CH3(48), TIM0_CH4(48), TIM0_CH5(48) of TSPP1 submodule are not implemented, hence the input signals cannot be disabled.

Scope

MAP

Effects

The specified disable function of the input signals TIM0_CH0(48), TIM0_CH1(48), TIM0_CH2(48) of TSPP0 submodule and the input signals TIM0_CH3(48), TIM0_CH4(48), TIM0_CH5(48) of TSPP1 submodule are not implemented, hence the input signals cannot be disabled.

Workaround

The combined TRIGGER or STATE output signals to the DPLL module can be disabled by using the control signals DPLL_CTRL_0.TEN(TRIGGER, TSPP0) and DPLL_CTRL_0.SEN (STATE, TSPP1).

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

GTM_AI.352 ATOM: No reload of data from ARU in SOMS and SOMP mode if TIM_EXT_CAPTURE(x) or TRIGIN(x) is selected as clock source

ATOM configuration:

- SOMP or SOMS mode (ATOM[i]_CH[x]_CTRL.MODE=0b10/0b11)
- ARU input stream enabled (ATOM[i]_CH[x]_CTRL.ARU_EN=1)
- TRIGIN(x) or TIM_EXT_CAPTURE(x) as selected clock source (ATOM[i]_CH[x]_CTRL.CLK_SRC=0b1101/0b1110)

Expected behaviour in SOMS mode:

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

Observed behaviour in SOMS mode:

ATOM channel stops after data is shifted out which was stored in shift register ATOM[i]_CH[x]_CM1.CM1 by the CPU. Data which was transferred via ARU stays in shadow register ATOM[i]_CH[x]_SR1.SR1 and will not be reloaded into the shift register; instead the channel stops.

Expected behaviour in SOMP continuous mode:

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

Observed behaviour in SOMP continuous mode:

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

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

Scope

ATOM

Effects

The reloading and update of new values for the shadow registers from ARU doesn't happen. The channel stops.

Workaround

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

If TRIGIN(x) is to be used as clock source, the output signal of the ATOM channel, which delivers the trigger signal TRIGIN(x), can be routed to TIM input

as AUX_IN signal. Now the TIM_EXT_CAPTURE(x) signal from this TIM module can be used with the same workaround as described before for TIM_EXT_CAPTURE(x) clock source. An additional clock delay of 3 cluster clocks would need to be considered for the generation of the TRIGIN(x) source.

GTM_AI.353 SPEC-ATOM: Specification of the smallest possible PWM Period in SOMP mode wrong, when ARU_EN=1

Configuration in use:

- ATOM[i]_CH[x]_CTRL.MODE=0b10 (SOMP),
- ATOM[i]_CH[x]_CTRL.ARU_EN=1,
- ATOM[i]_AGC_GLB_CTRL.UPEN_CTRLx=1

Functionality:

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

Reloading of the ATOM[i]_CH[x]_CM0.CM0 and ATOM[i]_CH[x]_CM1.CM1 registers with the values from ATOM[i]_CH[x]_SR0.SR0 and ATOM[i]_CH[x]_SR1.SR1 takes place, when the old PWM period expires (ATOM[i]_CH[x]_CN0.CN0 reaches ATOM[i]_CH[x]_CM0.CM0 in up counter mode or ATOM[i]_CH[x]_CN0.CN0 reaches 0 in up/down counter mode).

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

Problem description:

The GTM-IP specification defines as minimal possible PWM period, where the PWM characteristic can be reloaded in a predictable manner so that new data is always available in time at the ATOM channel, to be the ARU round trip time

of the specific microcontroller device. This is not correct, because the data needs two additional ARU clock cycles to flow through the ARU from a source to the ATOM channel plus one clock cycle for loading the value from the shadow registers ATOM[i]_CH[x]_SR0.SR0 and ATOM[i]_CH[x]_SR1.SR1 to the registers ATOM[i]_CH[x]_CM0.CM0 and ATOM[i]_CH[x]_CM1.CM1.

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

Scope

SPEC-ATOM

Effects

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

Workaround

The PWM period has to be larger than ARU round trip time + 3 ARU clock cycles. This can be reached by either choosing a smaller device, or by using ARU dynamic routing, or by reducing the value of ARU_CADDR_END to a value, which fits the PWM period. So, PWM period greater than ARU_CADDR_END + 1 + 3 ARU clock cycles.

GTM_AI.354 DMCS: Unresolved hazard resulting from RAW (Read After Write) dependency

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

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

Example:

Assume that following sequence

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

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

Scope

MCS

Effects

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

Workaround

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

For example:

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

- MOV GMI0, R0
- NOP
- NOP
- MOV R1, GMI0

GTM_TC.018 DPLLRAM trace data can be wrong

Note: This problem only has an effect during debugging.

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

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

Workaround

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

GTM_TC.019 ARU can not be traced if GTM cluster 5 is disabled

Note: This problem only has an effect during debugging.

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

Workaround

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

GTM_TC.020 Debug/Normal read access control via bit field ODA.DRAC

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

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

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

- for all masters when ODA.DRAC = 10_B or 11_B,
- for the Cerberus (OCDS) FPI master when ODA.DRAC = 00_B

Problem Description

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

- for all masters when ODA.DRAC = 10 or 11_B,
- for the CPU2 FPI master when ODA.DRAC = 00_B

Workaround

The problem described above has 2 aspects:

1. For CPU2 Access to GTM

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

2. For Cerberus (OCDS) Access to GTM

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

GTM_TC.021 Registers CANOUTSEL0, CANOUTSEL1 - Documentation update for fields SELx (x = 0, 1)

The description in the current version of the product specific TC3xx Appendix regarding fields SELx (x = 0, 1) in registers CANOUTSEL0 and CANOUTSEL1 is partly incorrect and will be updated as shown in the following tables.

Note: The changes only affect settings $8_H..F_H$ in fields SEL0 and SEL1 of registers CANOUTSEL0 and CANOUTSEL1, respectively.

Table 12 GTM_CANOUTSEL0 - CAN0/CAN1 Output Select Register - Documentation update for fields SELx (x = 0, 1)

Field	Description	
SELx (x = 0)	**Output Selection for GTM to CAN connection x** This bit field defines which TOM/ATOM channel output is used as CAN0/CAN1 node trigger x.	
	$0_H..7_H$	- No change: see description in TC3xx Appendix-
	8_H	**CDTM0_DTM1_2, TOM0_6**, Dead-time output of TOM0, channel 6
	9_H	**CDTM0_DTM1_3, TOM0_7**, Dead-time output of TOM0, channel 7
	A_H	**TOM0_13**, Output of TOM0, channel 13
	B_H	**TOM0_14**, Output of TOM0, channel 14
	C_H	**CDTM0_DTM5_0, ATOM0_4**, Dead-time output of ATOM0, channel 4
	D_H	**CDTM0_DTM5_1, ATOM0_5**, Dead-time output of ATOM0, channel 5
	E_H	**CDTM0_DTM5_2, ATOM0_6**, Dead-time output of ATOM0, channel 6
	F_H	**CDTM0_DTM5_3, ATOM0_7**, Dead-time output of ATOM0, channel 7
SELx (x = 1)	**Output Selection for GTM to CAN connection x** This bit field defines which TOM/ATOM channel output is used as CAN0/CAN1 node trigger x.	

Table 12 GTM_CANOUTSEL0 - CAN0/CAN1 Output Select Register - Documentation update for fields SELx (x = 0, 1) (cont'd)

Field	Description	
	$0_H..7_H$	- No change: see description in TC3xx Appendix-
	8_H	**CDTM1_DTM1_2, TOM1_6**, Dead-time output of TOM1, channel 6
	9_H	**CDTM1_DTM1_3, TOM1_7**, Dead-time output of TOM1, channel 7
	A_H	**TOM1_13**, Output of TOM1, channel 13
	B_H	**TOM1_14**, Output of TOM1, channel 14
	C_H	**CDTM1_DTM5_0, ATOM1_4**, Dead-time output of ATOM1, channel 4
	D_H	**CDTM1_DTM5_1, ATOM1_5**, Dead-time output of ATOM1, channel 5
	E_H	**CDTM1_DTM5_2, ATOM1_6**, Dead-time output of ATOM1, channel 6
	F_H	**CDTM1_DTM5_3, ATOM1_7**, Dead-time output of ATOM1, channel 7

Table 13 GTM_CANOUTSEL1 - CAN2 Output Select Register - Documentation update for fields SELx (x = 0, 1)

Field	Description	
SELx (x = 0)	**Output Selection for GTM to CAN connection x** This bit field defines which TOM/ATOM channel output is used as CAN2 node trigger x.	
	$0_H..7_H$	- No change: see description in TC3xx Appendix-
	8_H	**CDTM0_DTM1_2, TOM0_6**, Dead-time output of TOM0, channel 6
	9_H	**CDTM0_DTM1_3, TOM0_7**, Dead-time output of TOM0, channel 7
	A_H	**TOM0_13**, Output of TOM0, channel 13
	B_H	**TOM0_14**, Output of TOM0, channel 14

Table 13 GTM_CANOUTSEL1 - CAN2 Output Select Register - Documentation update for fields SELx (x = 0, 1) (cont'd)

Field	Description	
	C_H	**CDTM0_DTM5_0, ATOM0_4**, Dead-time output of ATOM0, channel 4
	D_H	**CDTM0_DTM5_1, ATOM0_5**, Dead-time output of ATOM0, channel 5
	E_H	**CDTM0_DTM5_2, ATOM0_6**, Dead-time output of ATOM0, channel 6
	F_H	**CDTM0_DTM5_3, ATOM0_7**, Dead-time output of ATOM0, channel 7
SELx (x = 1)	**Output Selection for GTM to CAN connection x** This bit field defines which TOM/ATOM channel output is used as CAN0/CAN1 node trigger x.	
	$0_H..7_H$	- No change: see description in TC3xx Appendix-
	8_H	**CDTM1_DTM1_2, TOM1_6**, Dead-time output of TOM1, channel 6
	9_H	**CDTM1_DTM1_3, TOM1_7**, Dead-time output of TOM1, channel 7
	A_H	**TOM1_13**, Output of TOM1, channel 13
	B_H	**TOM1_14**, Output of TOM1, channel 14
	C_H	**CDTM1_DTM5_0, ATOM1_4**, Dead-time output of ATOM1, channel 4
	D_H	**CDTM1_DTM5_1, ATOM1_5**, Dead-time output of ATOM1, channel 5
	E_H	**CDTM1_DTM5_2, ATOM1_6**, Dead-time output of ATOM1, channel 6
	F_H	**CDTM1_DTM5_3, ATOM1_7**, Dead-time output of ATOM1, channel 7

GTM_TC.296 ARU data at the GTM OTGBM interface may be doubled

Note: This problem only has an effect during debugging.

If GTM cluster 0 is configured to use the clock without divider (bit field GTM_CLS_CLK_CFG.CLS0_CLK_DIV = 01_B), then ARU_DBG_DATA0_L and ARU_DBG_DATA1_L may appear twice on the OTGBM0/1 busses.

Note: If GTM cluster 0 is configured to use the clock with divider (bit field CLS0_CLK_DIV = 10_B), the ARU data correctly will appear only once.

Workaround

Configure GTM cluster 0 to use the clock with divider (CLS0_CLK_DIV = 10_B) to properly trace the ARU data.

HSCT_TC.012 HSCT sleep mode not supported

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

Do not set bit SLEEPCTRL.SLPEN = 1_B.

HSCT_TC.013 Internal Loopback Mode not reliable

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

Therefore, do not use the internal loopback mode (i.e. do not set bit TESTCTRL. LLOPTXRX = 1_B).

Workaround

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

MCMCAN_AI.015 Edge filtering causes mis-synchronization when falling edge at Rx input pin coincides with end of integration phase

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.

MCMCAN_AI.017 Retransmission in DAR mode due to lost arbitration at the first two identifier bits

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.

MCMCAN_AI.018 Tx FIFO Message Sequence Inversion

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

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

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

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

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

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

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

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

Scope:

The erratum describes the case when the MCMCAN uses both, dedicated Tx Buffers and a Tx FIFO (CAN_TXBCi.TFQM = ’0’) and the messages in the Tx

FIFO do not have the highest internal CAN priority. The described sequence inversion may also happen between two non-Tx FIFO messages (Tx Queue or dedicated Tx Buffers) that have the same CAN identifier and that should be transmitted in the order of their buffer numbers (not the intended use).

Effects:

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

Workaround

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

First Workaround:

Use two dedicated Tx Buffers, e.g. 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

Second Workaround:

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.

Third Workaround:

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.

MCMCAN_AI.019 Unexpected High Priority Message (HPM) interrupt

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:

- 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.

MCMCAN_AI.020 Message transmitted with wrong arbitration and control fields

Under the following conditions a message with wrong ID, format, and DLC is transmitted:

- MCMCAN is in state “Receiver” (**PSRi.ACT =** “10”), no pending transmission
- A new transmission is requested before the 3rd bit of Intermission is reached
- The CAN bus is sampled dominant at the third bit of Intermission which is treated as SoF (see ISO11898-1:2015 Section 10.4.2.2)

Under the conditions listed above it may happen that:

- The shift register is not loaded with ID, format, and DLC of the requested message
- The MCMCAN will start arbitration with wrong ID, format, and DLC on the next bit
- In case the ID wins arbitration, a CAN message with valid CRC is transmitted
- In case this message is acknowledged, the ID stored in the Tx Event FIFO is the ID of the requested Tx message and not the ID of the message transmitted on the CAN bus, no error is detected by the transmitting MCMCAN

Scope

The erratum is limited to the case when MCMCAN is in state “Receiver” (**PSRi.ACT =** “10”) with no pending transmission (no TXBRP bit set) and a new transmission is requested before the 3rd bit of Intermission is reached and this 3rd bit of intermission is seen dominant.

When a transmission is requested by the CPU by writing to **TXBARi**, the Tx Message Handler performs an internal arbitration and loads the pending transmit message with the highest priority into its output buffer and then sets the transmission request for the CAN Protocol Controller. The problem occurs only when the transmission request for the CAN Protocol Controller is activated in the critical time window between the sample points of the 2nd and 3rd bit of Intermission and if that 3rd bit of intermission is seen dominant.

This dominant level at the 3rd bit of Intermission may result from an external disturbance or may be transmitted by another node with a significantly faster clock.

Effects

In the described case it may happen that the shift register is not loaded with arbitration and control field of the message to be transmitted. The frame is transmitted with wrong ID, format, and DLC but with the data field of the requested message. The message is transmitted in correct CAN (FD) frame format with a valid CRC.

If the message loses arbitration or is disturbed by an error, it is retransmitted with correct arbitration and control fields

Workaround 1

Request a new transmission only if another transmission is already pending or when the MCMCAN is not in state “Receiver” (when **PSRi.ACT** ≠ “10”).

To avoid activating the transmission request in the critical time window between the sample points of the 2nd and 3rd bit of Intermission, the application software can evaluate the Rx Interrupt flags **IRi.DRX**, **IRi.RF0N**, **IRi.RF1N** which are set at the last bit of EoF when a received and accepted message gets valid.

The last bit of EoF is followed by three bits of Intermission. Therefore the critical time window has safely terminated three bit times after the Rx interrupt. Now a transmission may be requested by writing to **TXBARi**.

After the interrupt, the application has to take care that the transmission request for the CAN Protocol Controller is activated before the critical window of the following reception is reached.

Workaround 2

A checksum covering arbitration and control fields can be added to the data field of the message to be transmitted, to detect frames transmitted with wrong arbitration and control fields.

Workaround 3

If a transmission is to be requested while no other transmission request is already pending and the CAN bus is not idle, set the **CCCRi.INIT** bit (which stops the CAN protocol controller), set the transmission request and finally clear the **CCCRi.INIT** bit. The message currently being received when **CCCRi.INIT** is set will be lost, but no errors (or error frames) will be generated and the CAN protocol controller will re-integrate into the CAN communication immediately at the 11 recessive bits of the next End-of-Frame & Intermission (or Idle) and will receive (or transmit) the following message.

Workaround 4

It is also possible to keep the number of pending transmissions always at >0 by frequently requesting a message, then the condition “no pending transmission” is never met. The frequently requested message may be given a low priority, losing arbitration to all other messages.

miniMCDS_TC.003 Trace messages get lost when ticks are enabled

The miniMCDS contains several trace units. Depending on the use case and configuration the trace units create messages in parallel.

Messages will be combined if they are generated in the same clock cycle. If enabled timing information is added to the trace through the insertion of <tick>/<multick> timestamp messages.

Trace messages get lost when the combined length of these messages is 116 bits and a 12 bit <multick> message is inserted.

Workaround

The following combinations of traces together with <tick>/<multick> timestamp messages are possible:

- Any variant of Program trace only (function, flow, instruction)
- Program flow or instruction trace plus trace of status information and watchpoints
- Program flow or instruction trace plus address part of data trace

For all other combinations of traces do not enable the generation of <tick>/<multick> messages.

miniMCDS_TC.004 NESTED_ISR incremented by resets

Note: This problem is only relevant for development tools.[1)]

TriCore does not provide any information about the interrupt nesting level which can be traced. Hence there is a nested interrupt counter per Processor Observation Block POB implemented inside of miniMCDS. This counter increments with every begin of an exception and decrements with every return from an exception.

System and Application Reset trigger a shutdown trap (executed by the firmware). The trap has an exception but no return from exception. This causes the NESTED_ISR bit field in the corresponding TCxDCSTS register to be incremented. If the user configures the miniMCDS to generate DCU messages then the trace message content is also incorrect.

1) This problem corresponds to problem MCDS_TC.062 for devices with MCDS.

Workaround

A tool that is notified about the System or Application Reset can correct the nested interrupt counter value in a post-processing step of the DCU message decoding.

miniMCDS_TC.005 TriCore wrap around write access causes redundant miniMCDS message

Note: This problem is only relevant for development tools. This problem corresponds to problem MCDS_TC.052 for devices with MCDS/MCDSLight.

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, miniMCDS 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.

miniMCDS 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.

miniMCDS_TC.006 Selection of SRI trace sources

Note: This problem is only relevant for development tools. It corresponds to problem MCDS_TC.065 for devices with MCDS/MCDSlight.

The miniMCDS provides the capability to trace accesses to the SRI Slaves CPUx_MEMSlave, LMU0, OLDA.

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 ongoing. This can lead to permanently corrupted MCDS trace messages.

Workaround

Switch to the trace source only in a time frame when no transactions are ongoing.

miniMCDS_TC.007 Selection of CPU trace sources

Note: This problem is only relevant for development tools. It corresponds to problem MCDS_TC.066 for devices with MCDS/MCDSlight.

The miniMCDS 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).

miniMCDS_TC.008 MCDS kernel reset shall not be used

Note: This problem is only relevant for development tools. It corresponds to problem MCDS_TC.067 for devices with MCDS.

If a miniMCDS 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 miniMCDS kernel reset.

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

MTU_TC.012 Security of CPU Cache Memories During Runtime is Limited

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

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

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

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

Workaround

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

MTU_TC.017 Unexpected alarms after application reset

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.

MTU_TC.018 Gated SRAM alarms

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 (e.g. MCx_ECCD register).

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

Additionally, only the alarm propagation is gated in this corner case, i.e. the error status is still correctly stored in the MCx_ECCD, MCx_FAULTSTS registers.

Workaround

For GTM & SCR SRAMs

Read the MCx_ECCD register periodically, depending on application safety considerations, for example within each diagnostic test interval.

- Corresponding SSH instances:
  - GTM: MC53..MC60;
  - SCR: MC77, MC78.

For DMA & CPU SRAMs (except DLMUx_STBY)

No workaround is recommended, because here the issue affects only the address error generation on a write access. In this case, the next read access (when the data would be used) will trigger the error.

For DLMU_STBY

The issue occurs in a corner case just before entering standby mode. Therefore, if standby mode is used and Standby RAM is enabled (PMSWCR0.STBYRAMSEL $\neq 000_B$) - then just before entering standby, perform an additional dummy read to DLMU_STBY location 9000 0000_H (when

using CPU0 dLMU RAM) and 9001 0000_H (when using CPU1 dLMU RAM). This dummy read triggers the alarm propagation and ensures that no alarms are lost due to standby entry.

MTU_TC.019 Type properties for reserved bits MCONTROL.R8, R12..R14 - Documentation update

In the description of register MCi_MCONTROL in the MTU chapter of the current version of the TC3xx User's Manual, the type properties of the reserved bits R8, R12, R13, R14 are indicated as read-only ("r").

- Actually, these bits are writable, i.e. the type of the reserved bits MCi_MCONTROL.R8, R12, R13, R14 is read/write ("**rw**").

Note:

As documented in the register description for MCi_MCONTROL,

- Bit R14 shall always be written with 1,
- Bits R8, R12, R13 shall always be written with 0.

PADS_TC.011 Pull-ups activate on specific analog inputs upon PORST

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

- analog inputs overlaid with general purpose inputs (class S pads) on all pins of P40 and P41[1)],
- analog inputs (class D pads) of channels with multiplexer diagnostics[2)],

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.

1) Availability depends on TC3xy device version, see the product specific Data Sheet.
2) These channels are explicitly marked with (MD) in table "Analog Input Connections for Product TC3yx" in the EVADC chapter of the product specific appendix (file TC3yX_um_appx_V1.*.*.pdf) to the AURIX™ TC3XX User's Manual.

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).

PER_PLL_TC.001 Peripheral PLL Divider Bypass must not be used

Activation of the /1.6 Divider Bypass function of the Peripheral PLL (register setting PERPLLCON0.DIVBY = 1_B) can result in device malfunction. This is due to a design weakness.

Therefore, **only** the Divider Bypass configuration **PERPLLCON0.DIVBY = 0_B must be used**.

Note: This problem is independent of the input clock frequency, and it is independent whether f_{PLL2} is used or not. It only depends on PERPLLCON0.DIVBY = 1_B.
The frequency values in the text below shall only be considered as examples.

Use Cases in chapter "Clocking System" of TC3xx User's Manual

The AURIX™ TC3xx User's Manual describes two recommended configuration sequences in chapter "Use Cases" of the "Clocking System" chapter:

- Setting **AAA** is defined as an example for a 20 MHz crystal / clock input.
  - This setting uses PERPLLCON0.DIVBY = 0_B, therefore no change is required when using this setting.
- Setting **BBB** is defined as an example for a 25 MHz crystal / clock input.
  - This setting uses **PERPLLCON0.DIVBY = 1_B**, therefore **changes** are **required** when using this setting.

Workaround

The configuration of the Peripheral PLL has to be selected such that the Divider Bypass function is disabled, i.e. PERPLLCON0.DIVBY = 0_B.

In the recommended configuration setting BBB in the User's Manual, in Step 2 the following modification is required:

- PERPLLCON0 = $00013E00_H$ (instead of $00013E01_H$)

Consequence

With this modification, assuming a 25 MHz input clock, f_{PLL2} changes to 250 MHz (from 200 MHz), while f_{PLL1} remains unchanged (160 MHz).

Connected to the f_{PLL2} output are the following modules: MSC, ASCLIN (input $f_{ASCLINF}$), QSPI, and I2C (see also table “CCU Clock Options” in chapter “Clocking System”).

The 250 MHz instead of the original 200 MHz would violate the maximum allowed frequency of 200 MHz for MSC, ASCLIN, and QSPI; therefore additional considerations are required for these modules.

Workaround - additional considerations for MSC

The MSC has the option to use either f_{PLL1} or f_{PLL2} as clock, whatever represents the more suitable clock frequency.

Therefore, select f_{PLL1} as clock source for MSC: based on f_{PLL1} = 160 MHz still 40 MHz and 80 MHz are possible for f_{MSC} addressing the application requirements.

Workaround - additional considerations for QSPI

As the QSPI is a synchronous communication interface the clock frequency for the QSPI defines the maximum achievable baudrate.

Changing the configuration for K3DIV from 2 to 3 would result in a frequency of f_{PLL2} = 166.667 MHz instead the targeted 200 MHz; this would reduce the maximum baudrate from 50 MBaud to 41.667 MBaud (based on the minimum allowed divider n = 4). If this baudrate still fulfills the application requirements no further impact needs to be analyzed.

The QSPI also has the option to use either f_{PLL1} or f_{PLL2} as clock, whatever represents the more suitable clock frequency.

Therefore, when selecting f_{PLL1} as clock source for QSPI, based on f_{PLL1} = 160 MHz still 40 MBaud application requirements are fulfilled.

Workaround - additional considerations for ASCLIN

As the ASCLIN is an (a)synchronous communication interface, the clock frequency for the ASCLIN defines the maximum achievable baudrate.

Changing the configuration for K3DIV from 2 to 3 result in a frequency of f_{PLL2} = 166.667 MHz instead the targeted 200 MHz; this would reduce the maximum baudrate from 50 MBaud to 41.667 MBaud (based on the minimum allowed divider n = 4). If this baudrate still fulfills the application requirements no further impact needs to be analyzed.

The ASCLIN also has the option to use either f_{PLL1} ($f_{ASCLINS}$) or f_{PLL2} ($f_{ASCLINF}$) as clock, whatever represents the more suitable clock frequency. When selecting f_{PLL1} as clock source for ASCLIN, based on f_{PLL1} = 160 MHz still 40 MBaud application requirements are fulfilled.

The ASCLIN also provides a fractional divider that can be used for baudrate generation.

Workaround - additional considerations for I2C

As the I2C is a synchronous communication interface that does not require high frequencies for its operation, and the I2C module provides a fractional divider, there should be no issue to derive the intended clock frequency from the reduced value of f_{PLL2}. Only the I2C specific clock control register requires an update.

Workaround - additional considerations if ERAY is not used

In the recommended Peripheral PLL configuration f_{PLL1} is configured with 160 MHz in order to finally provide the ERAY module with a 80 MHz module clock required by the ERAY protocol for the tick timing.

If the ERAY function is not required for the application there is also no blocking point to configure f_{PLL1} = 200 MHz.

Examples

The following table shows some examples for resulting clock frequencies based on the assumption f_{REF} = 25 MHz, P = 1, N = 32, f_{DCO} = 800 MHz:

Table 14 Configuration Options for Peripheral PLL with input clock = 25 MHz

K2	K3	DIVBY	f_{PLL1} [MHz]	f_{PLL2} [MHz]	Comments
5	2	**1**	160	200	Setting BBB (see User's Manual) - **DIVBY=1 must not be used!**
5	2	0	160	**250**	Setting BBB with DIVBY=0: **f_{PLL2} too high**
5	3	0	160	166.67	
5	4	0	160	125	
4	3	0	200	166.67	
4	4	0	200	125	

PMS_TC.004 EVRC DCDCSYNC output affected by Warm PORST

In this design step, the DCDCSYNC synchronisation output from the Step-Down Regulator (EVRC) to the external regulator is reset on a warm PORST and has to be consequently reconfigured when using the EVRC.

Note: In future design steps, the DCDCSYNC synchronisation output and function is reset only on a cold PORST or Low Voltage Detector (LVD) reset and not on a warm PORST.

PMS_TC.005 Voltage rise at P33 and P34 up to 2.5V during start-up and power-down

Tristate control information based on HWCFG[6] latched with V_{EXT} supply ramp can't be used within the V_{EVRSB} supply domain if the V_{EVRSB} input supply voltage has not reached its low voltage reset limit $V_{LVDRSTSB}$.

Therefore, the pad behavior at P33 and P34 pins is "pull-up" up to 2.5V ($V_{LVDRSTSB}$) during the ramp-up phase and below 2.5V for the ramp-down phase of the V_{EVRSB} supply even if HWCFG[6] =0.

Workaround

If an application requires to ensure the state of P33 and P34 pins within the logical “low” level for the supply voltage below 2.5V ($V_{LVDRSTSB}$), then an external pull-down may be used.

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).

PMS_TC.006 PORST not released during Cold Power-on Reset until VDDM is available

Upon a cold power-on reset, the PORST pin is kept asserted by the PMS until the ADC Analog Supply voltage (VDDM) is above 500 mV. This might lead to an additional start-up delay dependent on when VDDM is available from the external regulator relative to the VEXT, VDDP3 and VDD supplies.

During operation, if VDDM drops below the secondary monitor undervoltage threshold, an SMU alarm is generated. If VDDM 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 (VDDM) 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.

PMS_TC.007 VDDP3 or VDD Overvoltage during start-up may not be detected by PBIST

In AURIX™ TC3xx devices, Power Built in Self Test (PBIST) is introduced to ensure that the supply voltages do not exceed absolute maximum limits during the start-up phase.

However, for a VDDP3 or VDD overvoltage event during start-up beyond operational upper limits, the PBIST is not able to detect this overvoltage event.

Workaround

Check the VDDP3 overvoltage condition in registers EVRSTAT (flag OV33) and EVRMONSTAT1 (field ADC33V) in software additionally during the start-up phase before enabling the corresponding SMU alarm.

Check the VDD overvoltage condition in registers EVRSTAT (flag OVC) and EVRMONSTAT1 (field ADCCV) in software additionally during the start-up phase before enabling the corresponding SMU alarm.

PMS_TC.011 VEXT supplied PU1 pads always in tristate after standby entry

Tristate mode is enabled for VEXT supplied PU1 pads (marked PU1 / 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).

Note: This affects only the VEXT supplied PU1 pads.
The VEVRSB supplied PU1 pads are still controlled by the settings of HWCFG[6] and PMSWCR5.TRISTREQ at the moment of and after entry to standby mode.

Recommendation

If the application requires the pull-up state of VEXT supplied PU1 pads, then it shall ensure it by means of external pull-up devices in the event of:

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

PMS_TC.012 Short to Supply and Ground Detection – Documentation update

In the first sentence of the paragraph above Figure “Short to Supply and Ground Detection” in the PMSLE and PMS chapters of the TC3xx User’s Manual, starting with “A short detection scheme may be activated for EVR33..”, the reference to the register bits to control this detection is incorrect.

Correction

The correct sentence should read as follows:

- A short detection scheme may be activated for EVR33 via **EVR33CON**. SHLVEN / SHHVEN bits.

Note: For details on EVR33CON see the register description in the PMSLE chapter in TC3xx User’s Manual V1.3.0 and following.

Note: In V1.5.0 of the TC3xx User’s Manual, the PMSLE chapter contains the correct sentence. Update in the PMS chapter will follow in the next revision.

QSPI_TC.006 Baud rate error detection in slave mode (error indication in current frame)

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_B).

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 e.g. 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.

QSPI_TC.009 USR Events for PT1=2 (SOF: Start of Frame)

In master mode, when the interrupt on USR event is associated with Start of Frame (i.e. USREN=1_B, PT1=2 in register GLOBALCON1, BACON.UINT=1_B), 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_B, PT1=2 in GLOBALCON1, BACON.UINT=1_B), 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.

QSPI_TC.010 Move Counter Mode - USR Events for PT1=4 (RBF: Receive Buffer Filled)

When a master operates in Move Counter Mode (MCCON.MCEN=1_B), and the interrupt on USR event is associated with Receive Buffer Filled (i.e. USREN=1_B, 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 e.g. BACON.UINT=1_B and GLOBALCON.PT1=4 before start of frame, and use GLOBALCON1.USREN to selectively disable/enable the user interrupt during frame transmission.

QSPI_TC.013 Slave: No RxFIFO write after transmission upon change of BACON.MSB

While a slave transmission is in progress, and if the BACON.MSB configuration is changed for the subsequent frame, then the RxFIFO write of the currently received frame may not occur.

Also in case of a TxFIFO underflow, the RxFIFO write of the currently received frame may not occur.

Workaround

As a general recommendation, in slave mode the configuration should be done before any transmission starts.

In particular to avoid the problem described above, the re-configuration of the BACON has to be done after the RxFIFO write has occurred. This implies the need for a gap between frames if a BACON update occurs.

QSPI_TC.014 Slave: Incorrect parity bit upon TxFIFO underflow

When a slave TxFIFO underflow occurs, the slave transmits only “ones” in response to a request of the master.

If parity is enabled, also the parity bit transmitted by the slave is always set to “1”. This may be incorrect, depending on data length and parity type.

Workaround

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

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

When a master operates in move counter mode (MCCON.MCEN = 1_B) 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_B or GLOBALCON1.PT2 = 101_B) shall only be used to trigger the CPU/DMA to fill the TXFIFO with the next move counter transaction data.

QSPI_TC.017 Slave: Reset when receiving an unexpected number of bits

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_B in register GLOBALCON1) to be triggered after the PT2 event “SLSI deselection” (PT2 = 101_B).

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_B.

SCR_TC.014 SCR pins switched to reset state on warm PORST or Standby Entry and Exit event

The internal modules of the SCR and their external interfaces should not be affected by a warm PORST (if PMSWCR4.PORSTREQ = 0) or on a Standby Entry or Exit event.

However, in this design step, the associated SCR pins (P33.[0:7], P34.1, P33.[9:15]) are temporarily switched to their reset state (pull-up or tristate, depending on PMSWSTAT.TRIST) upon a warm PORST (with PMSWCR4.PORSTREQ = 0) or on a Standby Entry or Exit event.

Note: This problem does not affect signals routed from P33.4..7 to the ADC Comparator Unit (ADCOMP) of the SCR if the pull devices have been disabled in the corresponding P00_IOCRk registers in the SCR. In this case, the ADCOMP functionality is not affected by these events.

After the event, the SCR pins return to their previous configuration. Internal SCR modules continue to run unaffected by a warm PORST (if PORSTREQ = 0) or a Standby Entry or Exit event.

The SCR pins remain in their reset state for the duration of the event as follows:

- During Standby Entry, SCR pins are set to reset state for a few clock cycles (<200 ns) during the transition from Run to Standby mode[1)]
- During Standby Exit, SCR pins are set to reset state after wake-up as long as PORST is asserted by the external regulator[2)]
- During a warm PORST for 80 µs .. 180 µs, or for the time the PORST pin is externally kept asserted, whichever is maximum.

1) see “Pin behavior” upon standby entry (phase between reference points 2) and 3)) in Figure “Standby entry on VEXT ramp-down and wake-up on VEXT ramp-up” in the Target Specification/User’s Manual.

2) see “Pin behavior” upon standby exit (phase between reference point 5) and active phase) in Figure “Standby entry on VEXT ramp-down and wake-up on VEXT ramp-up” in the Target Specification/User’s Manual.

As a consequence, interaction with external components connected to the SCR pins may be disturbed.

Workaround

External pull devices may be added on SCR pins to ensure safe state on these pins during the phase when they are in their reset state.

Occurrence of a Standby Entry/Exit event may be communicated to the SCR before the event to allow software running on the SCR to establish a defined scenario.

Occurrence of a warm PORST may be communicated to the SCR after the event to ensure continuity of pin functions.

SCR_TC.015 Bit SCU_PMCON1.WCAN_DIS does not disable WCAN PCLK input

Setting bit SCU_PMCON1.WCAN_DIS to 1_B 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_B).

SCR_TC.016 DUT response to first telegram has incorrect C_START value

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 (e.g. 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
poll for TRF==1 on the first telegram of the monitor processed commands, and 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.

SCR_TC.017 WCAN module not reliable in TC38x

The WCAN module does not work reliably and shall not be used in TC38x.

Workaround

Bus activity has to be monitored with the MCMCAN module. In case of bus sleep and starting activity, a wake-up can be achieved with external interrupts on the corresponding receive pins.

To avoid unexpected interrupts from the WCAN module,

- the WCAN module must not be enabled (WCAN_CFG.WCAN_EN = 0_B), and
- WCAN interrupts must be kept disabled (clear all interrupt mask bits in register WCAN_INTMRSLT to 0_B).

SCR_TC.018 SSC Receive FIFO not working

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.

SCR_TC.019 Accessing the XRAM while SCR is in reset state

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): i.e. set
   - PMSWCR0.SCRWKEN = 0_B – wake-up via SCR disabled;
   - PMSWCR4.SCREN = 0_B – SCR disabled.

SCR_TC.020 Stored address in mon_RETH may be wrong after a break event

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_H of an instruction, the stored address in mon_RETH is wrong if mon_RETL contains 00_H (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_H 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.

SCU_TC.030 Connections of SCU - Documentation in TC3xx Appendix

In table “Connections of SCU” in the SCU chapter of the product specific TC3xx Appendix V1.1.0 and V1.3.0, the lines for the interface signals SCU:E_PDOUT(0) to SCU:E_PDOUT(3) are broken.

Recommendation

Refer to table “Connections of SCU” in V1.2.0 of the product specific TC3xx Appendix for a correct description of the SCU connections.

SMU_TC.012 Unexpected alarms when registers FSP or RTC are written

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_B) and FSP[0] is toggling with a defined T_{SMU_FFS} period,
- or in Dual Rail protocol (FSP.MODE = 01_B) 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_B) (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

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_B). 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_B);
  - 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_B):
  - 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).

3 Deviations from Electrical- and Timing Specification

ADC_TC.P009 Increased TUE for G10 when using Alternate Reference

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

- up to ±12 LSB_{12} for operation with converter reference clock f_{ADCI} = 32 MHz.
- up to ±25 LSB_{12} for operation with converter reference clock f_{ADCI} = 40 MHz.
- up to ±46 LSB_{12} for operation with converter reference clock f_{ADCI} = 53.33 MHz.

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

Recommendation

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

ADC_TC.P012 Increased RMS Noise

For the TC38x, on the analog channels listed below, the specified RMS noise (EN_{RMS}) increases, independent of the noise reduction mode:

- EN_{RMS} (Max.) ≤ 1.2 LSB in the temperature range -40°C ≤ T_A ≤ +25°C.
- EN_{RMS} (Max.) ≤ 1.1 LSB in the temperature range +25°C < T_A ≤ +100°C.
- EN_{RMS} (Max.) ≤ 1.0 LSB for T_A > +100°C (this is the specified standard value).

Note: This increased RMS noise EN_{RMS} (Max.) only applies to packaged device variants and only affects channels G0CH0…G0CH7 and G1CH0...G1CH2.

ADC_TC.P014 Equivalent Circuitry for Analog Inputs - Additional information

Figure “Equivalent Circuitry for Analog Inputs” will be modified in future revisions of the Data Sheet, including the term $C_{Parasit} \leq 30$ pF, as shown in the following figure:

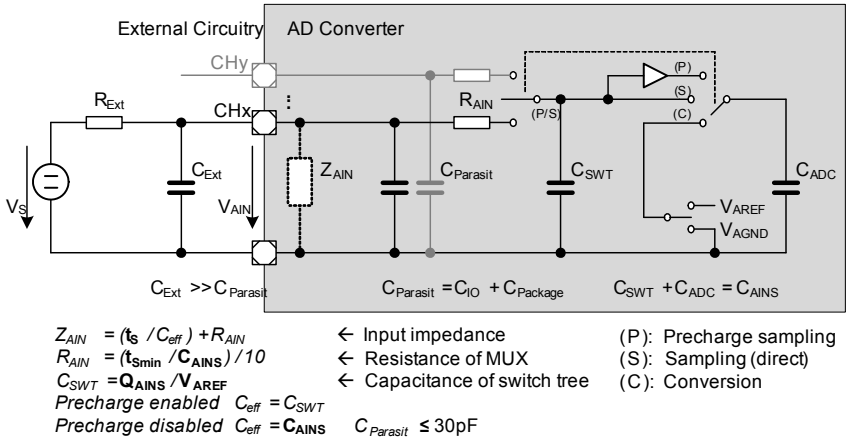

Figure 8 Equivalent Circuitry for Analog Inputs

EDSADC_TC.P002 Parameters Input Current, Gain Error - Additional information

In the latest revisions of the Data Sheet, the following information has been added to table “DSADC 5V”:

- Minimum limit for parameter Input Current (I_{RMS})
- Additional footnote on parameter “Gain Error” (ED_{GAIN}), describing the gain mismatch error between the different EDSADC channels

This footnote assumes that the considered EDSADC channels have the same calibration strategy. The footnote needs therefore to be extended with the condition ".. if they have the same calibration strategy" (see text in **bold** below).

Footnote on parameter "Gain Error" (ED_{GAIN}) - Extension

Gain mismatch error between the different EDSADC channels is within ±0.5% **if they have the same calibration strategy**.

FLASH_TC.P003 Program Flash Erase Time per Multi-Sector Command

The maximum value for parameter "Program Flash Erase Time per Multi-Sector Command" can be

- $t_{MERP} \leq 0.52$ s (instead of 0.5 s as specified in the Data Sheet).

Consequently, the maximum value for parameter "Complete Device Flash Erase Time PFlash and DFlash" can also increase by 0.04 s/Mbyte, resulting in

- $t_{ER\_Dev} \leq 11.9$ s (instead of 11.5 s as specified in the Data Sheet).

The increased values should be considered e.g. when defining erase timeout limits.

GETH_TC.P001 Maximum and minimum GETH operating frequency - Documentation update

The maximum and minimum value for the GETH frequency (f_{GETH}) will be changed from 200/150 MHz to 150/100 MHz, respectively, in the next revision of the Data Sheet (chapter "Operating Conditions") as shown in **Table 15** below.

Table 15 Operating Conditions for f_{GETH} - Documentation update

Parameter	Symbol	Values			Unit
		Min.	Typ.	Max.	
GETH frequency	f_{GETH} CC	100	-	150	MHz

Impacted Use Cases:

There is no impact on ‘optimum performance’ use cases with a 2:1 ratio of f_{SRI}:f_{GETH}, for example f_{SRI} = 300 MHz, f_{GETH} = 150 MHz.

RESET_TC.P003 Parameter limits for t_{PI} (Ports inactive after ESR0 reset active) – Documentation update

In table “Reset Timing” of the current version of the Data Sheet, parameter “Ports inactive after ESR0 reset active” (symbol t_{PI}) specifies a maximum value of $8/f_{SPB}$ (or $8/f_{BACKT}$, respectively) and/or unit (ns) which is incorrect.

The actual limits are as follows:

Table 16 Ports inactive after ESR0 reset active – Update

Parameter	Symbol	Values			Unit
		Min.	Typ.	Max.	
Ports inactive after ESR0 reset active	t_{PI} CC	$8000/f_{BACKT}$		$18000/f_{BACKT}$	s

Details

t_{PI} defines the pad reset delay on System Reset and Application Reset scenarios triggered by external ESR0 requests. These reset cases trigger execution of the shutdown sequence in the SCU within the t_{PI} time window followed by pad reset generation. The maximum time limit is defined by the timeout counter TOUTCNT which generates reset regardless of the execution state of the shutdown sequence.

4 Application Hints

ADC_TC.H026 Additional Waiting Phase in Slow Standby Mode

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 Target Specification/User’s Manual).

While idle precharge is disabled (GxANCFG.IPE = 0_B), an additional waiting phase of 1.6 µs (@f_{ADC} = 160 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.

ADC_TC.H028 Inconsistent contents in GLOBRES if result writes come too close

If result values are stored to the global result register GLOBRES shortly after each other, the bitfields in GLOBRES may be inconsistent. This is caused by internal synchronization.

Recommendation

If multiple groups deliver results to register GLOBRES, make sure that at least 5 f_{ADC} clock cycles are in between two write accesses.

ADC_TC.H029 Storing result values to a full FIFO structure

FIFO structures can be built from result registers to store several result values before they are retrieved by the system. Reading a result value makes the remaining result values move down in the FIFO structure.

The behavior depends on the status of the uppermost FIFO register (TOF) at the time a new result value is being stored:

- **Case A**: TOF register is available:
  - The new result value is stored in the TOF register (and moves down if space is available).
- **Case B**: TOF register is occupied:
  - The new result value overrides the value in the TOF register.
- **Case C**: TOF register is being copied to the next lower level:
  - The new result value is discarded and the TOF register remains available.

Recommendation

Make sure that the TOF register of the FIFO structure is available at the time a new result value is to be stored, i.e. retrieve result values from the FIFO structure early enough.

ADC_TC.H030 Flushing a running queue may corrupt previous conversion results

Request queues automatically start a sequence of conversions. A running conversion sequence can be stopped by the application. However, flushing a queue when a new conversion is being started can lead to corruption of previous result values.

Recommendation

Follow the instructions to stop/abort a running queue given in section “Queued Source Operation” of the EVADC chapter in the User’s Manual.

ADC_TC.H032 ADC accuracy parameters - Definition

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}) >= 4.5 V. If the reference voltage range is below 4.5 V by a factor of k (e.g. 3.3 V), the accuracy parameters increase by a factor of 1.1/k (e.g. 1.1 × 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}).

ADC_TC.H033 Basic Initialization Sequence for Primary and Secondary EVADC Groups

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 present version of the EVADC chapter of the User’s Manual, step “WAIT” (third step below) has been shifted **before** the begin of the start-up calibration.*

```
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 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 can be executed in the meantime)
       ;=> This starts continuous conversion of the channel
    *)time tSUCAL or flag GxARBCFG.CAL=0
```

ADC_TC.H034 Effect of reduced reference voltage on parameter QCONV - Data Sheet footnote update

The following footnote on parameter QCONV (Reference input charge consumption per conversion (from VAREF)) in table “VADC 5V” in the current version of the Data Sheet

- “For reduced reference voltages the consumed charge is reduced by factor k”

shall be changed to

- “For reduced reference voltages VAREF < 3.375V, the consumed charge QCONV is reduced by the factor of k_2 = VAREF [V] / 3.375. For reduced reference voltages 4.5V < VAREF ≤ 3.375V, QCONV is not reduced.”

ADC_TC.H035 Effect of input leakage current on Broken Wire Detection

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):
  – $I_{BWR} = (V_{AIN} – V_{AREF}) * C_{AINS} * CR$

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, i.e. 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 $T_J \leq 150°C$ (max) and ADC inputs are used in a configuration where $I_{OZ} \leq 150$ nA (see Data Sheet), I_{BWG} should be ≥ 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 ≥ 240 000 samples per second and input.

2. Worst Case Example

Assuming that $T_J \leq 170°C$ (max) and ADC inputs are used in a configuration where $I_{OZ} \leq 800$ nA (see Data Sheet), I_{BWG} should be ≥ 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 ≥ 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, e.g. for BWG from 10% to 20% of the full scale voltage.
2. Increase the conversion rate CR per input by introducing additional conversions.

ADC_TC.H036 Minimum Input Buffering Time - Additional information

As described in section “Buffer for the Analog Input” in the EVADC chapter “Analog Signal Buffering”, the analog input buffer boosts the selected analog

input signal for a certain time, when enabled. The time during which the input buffer is active can be adapted to the configured sample time by bitfields AIPS/AIPE in register GxICLASSi (i=0-1;x=0-11) / GLOBICLASSi (i=0-1), or by bitfield AIPF in register FCxFCCTRL (x=0-7)[1], respectively. The input precharge time can be configured to 8, 16, or 32 clocks of f_{ADC}.

After the programmed buffer time the sampling is continued directly from the selected input. The effective overall sampling time must cover the specified minimum sampling time t_S (see Data Sheet), i.e. the programmed sample time (in bitfields STC*) must cover both phases, buffered sampling (configured in bitfields AIP*) and direct sampling.

Note: Sampling times with input buffer enabled specified in the Data Sheet consider a buffered sample time of 200 ns, that means for f_{ADC} = 160 MHz the input precharge time (in bitfields AIP) has to be configured to 32 clocks of f_{ADC}. For input precharge times lower than 200 ns, the charge consumption from the analog input is increased accordingly.*

ADC_TC.H037 CPU read access latency to result FIFO buffer

Using the result FIFO buffer, data consistency for a sequence of conversion results can be guaranteed. This means, the results of the conversion sequence can be read by the CPU at a deterministic point in time. The data transfer from the result FIFO buffer to the CPU is usually done with a consecutive procedure of single read commands.

The read access latency between a CPU and the result FIFO buffer is defined by 6 system peripheral bus clock cycles (f_{SPB}) and 6 ADC clock cycles (f_{ADC}). The architecturally determined access time from a CPU via the system peripheral bus to the result FIFO is given by 5 system peripheral bus cycles (f_{SPB}).

Recommendation

Therefore, a waiting time between consecutive reads from the result FIFO buffer of 1 f_{SPB} cycle + 6 f_{ADC} cycles must be considered to ensure conversion

1) in TC39x step AA: GxFCCTRL (x=12-19)

results are correctly read from the FIFO. Otherwise, if this waiting time is not met for consecutive reads, the FIFO may get stuck.

The preferred way is to read the FIFO after the corresponding service request has been generated.

ASCLIN_TC.H001 Bit field FRAMECON.IDLE in LIN slave tasks

For LIN performing slave tasks, bit field FRAMECON.IDLE has to be set to 000_B (default after reset), i.e. no pause will be inserted between transmission of bytes.

If FRAMECON.IDLE > 000_B, 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).

BROM_TC.H008 CAN BSL does not support DLC = 9 and DLC = 11

The CAN Bootstrap loader (BSL) only supports messages where the number of data bytes is a multiple of 8.

Therefore, Data Length Code settings DLC = 11 (number of data bytes = 20) and DLC = 9 (number of data bytes = 12) are not allowed (see also chapter “CAN BSL flow” of chapter “AURIX™ TC3xx Platform Firmware”).

Recommendation

When using the CAN Bootstrap loader, only use settings where DLC ≠ 9 or DLC ≠ 11.

BROM_TC.H009 Re-Enabling Lockstep via BMHD

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 (e.g. via a different BMHD) is not reliably possible if warm PORST, System or Application reset is executed.

Recommendation

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

BROM_TC.H014 SSW behavior in case of wrong state or uncorrectable error in UCBs - Documentation Update

The boot sequence terminates and the device is put into error state (endless loop) in the following cases:

- **Wrong state** - i.e. different from CONFIRMED or UNLOCKED (in case an UCB has ORIGINAL and COPY: wrong state of the both) – for the following UCBs:
  - UCB_BMHDx, UCB_SWAP, UCB_SSW, UCB_USER, UCB_PFLASH, UCB_DFLASH, UCB_DBG, UCB_HSM, UCB_HSMCOTP0...1, UCB_HSMCFG, UCB_ECPRIO, UCB_OTP0...7, UCB_REDSEC, UCB_TEST, UCB_RETEST.
- **Uncorrectable ECC error** within the used locations when state valid (CONFIRMED or UNLOCKED) – for the following UCBs:
  - UCB_SSW, UCB_PFLASH, UCB_DFLASH, UCB_DBG, UCB_HSM, UCB_HSMCOTP0...1, UCB_ECPRIO, UCB_OTP0...7, UCB_REDSEC, UCB_RETEST.
- For UCB_SWAP ORIGINAL/COPY – according to the descriptions in User's 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's Manual[1)].

1) For TC39x A-step: chapter "Program Memory Unit (PMU)" of the Target Specification.

BROM_TC.H015 Different initial values for CPU0_PMEM SSH registers in MTU after cold PORST if SOTA/SWAP is enabled

If SOTA/SWAP functionality is enabled via the SOTA Mode Enable (UCB_OTP.PROCONTP.SWAPEN), and a cold PORST is performed, registers ECCD, ETRRx and ERRINFOx in MC2 (associated with CPU0_PMEM) may contain “false-positive signatures”, indicating correctable or uncorrectable ECC errors. However, these are no real errors but result from firmware side effects (prefetches to - at that time - uninitialized memory).

Recommendation

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

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

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

BROM_TC.H016 CHSW fails check of PMS_EVR registers after software triggered LBIST

*Note: This problem only affects earlier design steps and engineering samples (marking EES or ES) of the devices listed in **Table 17** below. It is fixed for qualified devices (QS) and design steps listed in **Table 18**.*

Problem Description

The Checker Software (CHSW) executes the check of some PMS registers for the default values after cold start as well as after LBIST execution (the device start-up in such a case is handled by CHSW as upon a cold power-on), whereas two related checks are executed:

a) indicated by SCU_STMEMx.EVRT1_* bits – this check covers the registers:

- EVRMONCTRL,
- EVRMONFILT

b) indicated by SCU_STMEMx.EVRT2_* bits – this check covers the registers:

- EVROVMON,
- EVROVMON2,
- EVRUVMON,
- EVRUVMON2,
- EVROSCCTRL,
- EVRRSTCON,
- EVRTRIM

The above check(s) may not pass after a software triggered LBIST, if the user's software modifies the following of the above mentioned registers before the LBIST:

- EVRMONCTRL,
- EVRMONFILT
  – resulting in bit SCU_STMEM4.EVRT1_CF = 1_B,

and/or

- EVROVMON,
- EVROVMON2,
- EVRUVMON,
- EVRUVMON2
  – resulting in bit SCU_STMEM4.EVRT2_CF = 1_B,

Recommendation

User's software shall return the PMS registers EVRMONCTRL, EVRMONFILT, EVROVMON, EVROVMON2, EVRUVMON, EVRUVMON2 to their default values prior to the software LBIST trigger.

Affected devices are identified as shown in **Table 17**:

Note: For these devices, location 0xAF40 0804 in UCB04 (UCB_SSW) contains 0x0000 0017.

Table 17 Devices with CHSW failing check of PMS_EVR registers after software triggered LBIST

Device	Design Step	Lot Numbers
TC39x	BA, BB	all
	BC	EES, ES up to 19844025yz[1)]
TC38x	AA, AB, AC	all
	AD	EES, ES up to 19842011xyz[1)]
TC35x	AA	all
	AB	devices with marking EES
TC37xEXT	AA	EES, ES up to 19919012xyz[1)]

1) For specific Lot Number of received samples see respective Status Sheet

The problem is fixed for the following devices and design steps as shown in **Table 18**:

Note: For these devices, location 0xAF40 0804 in UCB04 (UCB_SSW) contains 0x0000 001F.

Table 18 Devices with CHSW not failing check of PMS_EVR registers after software triggered LBIST

Device	Design Step	Lot Numbers
TC39x	BC	all QS, ES > 19844025yz[1)]
	BD and following	all
TC38x	AD	all QS, ES > 19842011xyz[1)]
	AE and following	all
TC35x	AB and following	devices without marking EES
TC37xEXT	AA	all QS, ES > 19919012xyz[1)]
	AB and following	all

1) For specific Lot Number of received samples see respective Status Sheet

BROM_TC.H017 CHSW results after LBIST execution

In AURIX™ TC3xx devices, LBIST execution terminates – independent whether successfully finished or interrupted by power-drop or external PORST - with a warm reset.

The Startup Software executed afterwards follows the flow as after cold power-on with the purpose to perform full device initialization.

If Checker Software (CHSW) is activated by the user configuration, the checks will be executed as required after cold power-on and the results are indicated in registers SCU_STMEM3...6 accordingly. Consequently, a successful device start-up will be indicated with the SCU_STMEM3...6 values shown in row “Cold power-on” of table “ALL CHECKS PASSED indication by CHSW for TC3xx” in the Firmware chapter of the TC3xx Appendix for the corresponding TC3xx device.

Recommendation

If using Checker Software, check bits SCU_STMEM3...6.[7:4] after start-up to determine which type of reset has been processed by device firmware. Then verify the SCU_STMEM3...6 contents against the values defined for the respective reset type in table “ALL CHECKS PASSED..” of the TC3xx Appendix for the corresponding TC3xx device.

CCU_TC.H012 Configuration of the Oscillator- Documentation Update

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

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

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

CLC_TC.H001 Description alignment for bits DISR, DISS, EDIS in register CLC - Documentation Update

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's Manual.

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

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

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

Table 19 General description of bits DISR, DISS, EDIS in register CLC

Field	Bits	Type	Description
DISR	0	rw	**Module Disable Request Bit** Used for enable/disable control of the module 0_B No disable requested 1_B Disable requested
DISS	1	rh	**Module Disable Status Bit** Bit indicates the current status of the module 0_B Module is enabled 1_B Module is disabled
EDIS	3	rw	**Sleep Mode Enable Control** Used for module Sleep Mode control 0_B Sleep Mode request is regarded. Module is enabled to go into Sleep Mode on a request. 1_B Sleep Mode request is disregarded. Module is enabled to go into Sleep Mode on a request.

Note:

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's Manual*

CPU_TC.H019 Semaphore handling for shared memory resources

Overview

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

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

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

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

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

Operational Details

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

- Local resource to the CPU
  - Local DSPR
  - Local DLMU (AURIX™ TC3xx)
- 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 $0_H..D_H$). 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 CPUx and the resource is local to CPUy. CPUx already owns the semaphore at the start of the described sequence. CPUy has not acquired the semaphore prior to accessing the resource.

Table 20 Example 1a: Out of order memory access due to different access paths to semaphore and shared resource

CPUx	CPUy	Memory Access Sequence
st-1 (resource-update)	ld-1 (semaphore-check)	
st-2 (semaphore-release)	ld-2 (resource-read)	
		st-2 (semaphore-release)
		ld-1 (semaphore-check)
		ld-2 (resource-read) “stale data”
		st-1 (resource-update)

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

Table 21 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

CPU_TC.H020 Inconsistent register description in CPU chapter - Documentation update

1. Overview

The current version of the CPU chapter in the TC3xx family User's Manual uses incorrect names for some of the Bus MPU registers. In multiple places the register names are combined with an incorrect index variable 'x'. Variable 'x' normally refers to the CPU instance. For these registers the intention was to refer to a particular range number.

The following description provides a summary of all the incorrect references as well as their actual intended values.

Note: Absolute chapter numbers refer to CPU chapter version V1.1.19 included in the TC3xx User's Manual V1.4.0. They may change if used in other versions of this document.

2. Chapter "Summary of SFR Reset Values and Access modes"(5.3.4.22.2)

Both of the tables named

- **Register Overview – SPR** (ascending Offset Address)
- **Register Overview – DLMU** (ascending Offset Address)

incorrectly use the letter 'x' as an index variable where '**i**' was intended in the Short Name, Long Name and Offset Address columns.

Corrected description of Register Overview tables

Corrected parts of these tables ('**i**' intended in 3 places per row) see below:

Table 22 Corrections to table "Register Overview – SPR"

Short Name	Long Name	Offset Address
SPR_SPROT_ RGNACCENA**i**_R	CPUx Safety Protection Region SPR Read Access Enable Register A**i**	$0E088_H$+**i***10_H
SPR_SPROT_ RGNACCENB**i**_R	CPUx Safety Protection Region SPR Read Access Enable Register B**i**	$0E08C_H$+**i***10_H

Table 23 Corrections to table "Register Overview – DLMU"

Short Name	Long Name	Offset Address
DLMU_SPROT_ RGNLA**i**	CPUx Safety Protection DLMU Region Lower Address Register **i**	$0E200_H$+**i***10_H
DLMU_SPROT_ RGNUA**i**	CPUx Safety Protection DLMU Region Upper Address Register **i**	$0E204_H$+**i***10_H
DLMU_SPROT_ RGNACCENA**i**_W	CPUx Safety Protection Region DLMU Write Access Enable Register A**i**	$0E208_H$+**i***10_H
DLMU_SPROT_ RGNACCENB**i**_W	CPUx Safety Protection Region DLMU Write Access Enable Register B**i**	$0E20C_H$+**i***10_H

Table 23 Corrections to table "Register Overview – DLMU" (cont'd)

Short Name	Long Name	Offset Address
DLMU_SPROT_ RGNACCENA**i**_R	CPUx Safety Protection Region DLMU Read Access Enable Register A**i**	$0E288_H$+**i***10_H
DLMU_SPROT_ RGNACCENB**i**_R	CPUx Safety Protection Region DLMU Read Access Enable Register B**i**	$0E28C_H$+**i***10_H

3. Section "Scratch Pad SRAMs" in chapter "Bus MPU" (5.4.6.1)

This section uses incorrect index variable 'x' or no index variable in references to the SPR_SPROT_* registers.

Corrected description of section "Scratch Pad SRAMs"

Each scratchpad region is defined using the registers, SPR_SPROT_RGNLAi (i=0-7) to define the lower address of the region, SPR_SPROT_RGNUAi (i=0-7) to define the upper address of the region.

Each region may be enabled for writes on a per bus master basis using the register SPR_SPROT_RGNACCENAi_W (Masters 31-0) and SPR_SPROT_RGNACCENBi_W (Masters 63-32).

Each region may be enabled for reads on a per bus master basis using the register SPR_SPROT_RGNACCENAi_R (Masters 31-0) and SPR_SPROT_RGNACCENBi_R (Masters 63-32).

A write access to the PSPR/DSPR memory is seen as valid if the master tag of the access is enabled in the SPR_SPROT_RGNACCENi_W register and the address of the access satisfies the following relationship:-

SPR_SPROT_RGNLAi <= address < SPR_SPROT_RGNUAi

A read access to the PSPR/DSPR is seen as valid if the master tag of the access is enabled in the SPR_SPROT_RGNACCENi_R register and the address of the access satisfies the following relationship:-

SPR_SPROT_RGNLAi <= address < SPR_SPROT_RGNUAi

If any of these conditions are not satisfied, the access is seen as invalid.

Accesses from all masters to the local DSPR (excluding data access from the local CPU) are checked by the SPR safety mechanism.

Accesses from all masters to the local PSPR (excluding fetch access from the local CPU) are checked by the SPR safety mechanism

4. Section “DLMU SRAMs” in chapter “Bus MPU” (5.4.6.1)

This section uses incorrect index variable ‘x’ or no index variable in references to the DLMU_SPROT_* and SPR_SPROT_RGNACCEN* registers.

Corrected description of section “DLMU SRAMs”

Each DLMU region is defined using the registers, DLMU_SPROT_RGNLAi (i=0-7) to define the lower address of the region, DLMU_SPROT_RGNUAi (i=0-7) to define the upper address of the region.

Each region may be enabled for writes on a per bus master basis using the register DLMU_SPROT_RGNACCENAi_W (Masters 31-0) and DLMU_SPROT_RGNACCENBi_W (Masters 63-32).

Each region may be enabled for reads on a per bus master basis using the register DLMU_SPROT_RGNACCENAi_R (Masters 31-0) and DLMU_SPROT_RGNACCENBi_R (Masters 63-32).

A write access to the local DLMU memory is seen as valid if the master tag of the access is enabled in the DLMU_SPROT_RGNACCENi_W register and the address of the access satisfies the following relationship:-

DLMU_SPROT_RGNLAi <= address < DLMU_SPROT_RGNUAi

A read access to the local DLMU memory is seen as valid if the master tag of the access is enabled in the DLMU_SPROT_RGNACCENi_R register and the address of the access satisfies the following relationship:-

DLMU_SPROT_RGNLAi <= address < DLMU_SPROT_RGNUAiIf any of these conditions are not satisfied, the access is seen as invalid.

Accesses from all masters to the local DLMU (including those from the local CPU) are checked by the DLMU safety protection mechanism.

5. Chapter “Register access enable Protection” (5.4.6.2)

This chapter uses incorrect index variable ‘x’ in reference to the SFR_SPROT_ACCEN*_W registers.

Corrected paragraphs of chapter “Register access enable Protection”

The CPUs implement the standard memory protection scheme for peripheral registers using the SFR_SPROT_ACCENA_W (Masters 31-0) and SFR_SPROT_ACCENB_W (Masters 63-32) register. This allows all CPU CSFR and SFR registers to be protected from write access by untrusted masters.

The SFR_SPROT_ACCENA_W and SFR_SPROT_ACCENB_W registers define which masters may write the SFR and CSFR registers via bus access through the SRI slave interface.

The SFR_SPROT_ACCENA_W and SFR_SPROT_ACCENB_W registers are protected by the safety_endinit signal.

6. Chapter “Safety Protection registers” (5.4.7.2)

This chapter describes all the registers in detail. The register names and descriptions of the registers listed in **Table 22** and **Table 23** all use incorrect index ‘x’ when ‘i’ was intended

- in the register name,
- in the offset calculation,
- in the register description.

DAM_TC.H001 RAM size of DAM instance in TC38x - Documentation correction

Section “TC38x Specific IP Configuration” in the DAM chapter of the TC38x specific appendix to the AURIX™ TC3XX User’s Manual states:

- “RAM size for the TC38x is 32 KB per instance”.

This is incorrect/misleading since the TC38x only has a single DAM instance with 64 KB.

Correction

The correct description for the TC38x specific configuration is:

- “RAM size for the TC38x DAM instance is **64 KB**”.

DAM_TC.H002 Triggering DAM MEMCON.RMWERR and INTERR flags

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

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

Note: For devices with EMEM see also Application Hint EMEM_TC.H006

DTS_TC.H002 Unexpected alarms after start-up/wake-up when temperature is close to lower/upper limit

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 or upper limit overflow, respectively. Also, the corresponding flag DTSLIM.LLU or DTSLIM.UOF is set.

Recommendation

The application software shall clear the respective flag in DTSLIM and **afterwards** clear related SMU alarms. In case alarms are retriggered, application SW shall consider these as real alarms, and trigger a reaction within the FTTI (Fault Tolerant Time Interval) of the respective application.

EDSADC_TC.H001 Auxiliary filter cleared with start of integration window - Additional information

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

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

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

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

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

EDSADC_TC.H003 Behavior of EDSADC result register in case of hardware controlled integration

The hardware triggered integration (DICFGx.ITRMODE = 01_B, DICFGx.ITRMODE = 10_B) can be used to define a timeframe where a specific number of samples coming from the time equidistant data stream of the upstreamed filter chain is accumulated. The values which are accumulated within this timeframe will be stored in the EDSADC result register and can be read by DMA or CPU. As soon the integration procedure is finished (ISTATx.INTEN = 0_B), the source for the result register changes from the integrator output to the upstreamed filter chain. When no result FIFO is used,

results in the results register will be continuously overwritten by the subsequent incoming results. This means, the last result of the hardware signal controlled integration is available in the EDSADC result register only for the timeframe which is defined by the data rate period of the upstreamed filter chain.

Recommendation

To avoid the described overwriting procedure, the EDSADC result FIFO can be used. Using the result FIFO, the accumulation result is accessible by CPU or DMA up to the time where another hardware signal controlled integration is initiated. For this purpose, the result FIFO has to use one of the following configurations:

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

These configurations have the effect that service requests are only generated during the integration window. In case of disabled integrator stage (ISTATx.INTEN = 0_B), results generated by the upstreamed filter chain will not trigger service requests. However, results from the upstreamed filter chain will be stored in higher stages of the result FIFO. When the FIFO stages are fully loaded, all other results from the upstreamed filter chain are discarded and FIFO write error is indicated (RFCx.WREC=1B).

FLASH_TC.H016 Implications on Power-up and Standby mode wake-up cycles

During power-up and standby mode wake-up cycles (from V_{DDP3} off), the PFlash may get disturbed depending on the maximum value of V_{DDP3} and the duration of the PORST assertion during this time. Therefore, the following limitations must be considered under the following conditions:

- While PORST is asserted,
  - V_{DDP3} must never exceed 3.5 V ($V_{DDP3\_nominal}$ +6%)
  - V_{DDP3} must never exceed 3.4 V ($V_{DDP3\_nominal}$ +3%) for more than 0.5 ms during the first phase after power-up, and then must remain below 3.4 V for another x ms before PORST is released.

- Under these conditions, the total amount n of power-up and standby mode wake-up cycles (from V_{DDP3} off) is limited according to the following formula: **n ≤ 600k * 2 / x**
  - Example: for x = 2 [ms]: n ≤ 600k cycles; for x = 16 [ms]: n ≤ 75k cycles.

Put Flash into sleep mode before entering standby mode (with V_{DDP3} on). If this cannot be achieved, limit the cumulated standby time to ≤ 10 minutes with V_{DDP3} ≤ 3.4 V (VDDP3_nominal +3%).

Note: In case of special application request, please contact your local IFX representative.

FLASH_TC.H019 Write Burst Once command – Documentation update

For the Write Burst Once command, the current version of the TC3xx User's Manual states in section "Command Sequence Definitions":

"This function starts the programming process for an aligned group of pages as the normal "Write Burst" does. But before programming it checks if the pages are erased. If the page is not erased (allowing correctable errors) the command fails with PVER and EVER."

*Note: The actual implementation of the Write Burst Once command is similar to the Write Page Once command, therefore it will be updated in the next version of the TC3xx User's Manual as follows (changes marked in **bold** below:*

Documentation Update - Write Burst Once

"This function starts the programming process for an aligned group of pages as the normal "Write Burst" does. But before programming it checks if the pages are erased. If the **pages are** not erased (allowing correctable errors) the command fails with **EVER**."

FlexRay_AI.H004 Only the first message can be received in External Loop Back mode

If the loop back (TXD to RXD) will be performed via external physical transceiver, there will be a large delay between TXD and RXD.

A delay of two sample clock periods can be tolerated from TXD to RXD due to a majority voting filter operation on the sampled RXD.

Only the first message can be received, due to this delay.

To avoid that only the first message can be received, a start condition of another message (idle and sampling '0' -> low pulse) must be performed.

The following procedure can be applied at one or both channels:

- wait for no activity (TEST1.AOx=0 -> bus idle)
- set Test Multiplexer Control to I/O Test Mode (TEST1.TMC=2), simultaneously TXDx=TXENx=0
- wait for activity (TEST1.AOx=1 -> bus not idle)
- set Test Multiplexer Control back to Normal signal path (TEST1.TMC=0)
- wait for no activity (TEST1.AOx=0 -> bus idle)

Now the next transmission can be requested.

FlexRay_AI.H005 Initialization of internal RAMs requires one eray_bclk cycle more

The initialization of the E-Ray internal RAMs as started after hardware reset or by CHI command CLEAR_RAMS (SUCC1.CMD[3:0] = 1100_B) takes 2049 eray_bclk cycles instead of 2048 eray_bclk cycles as described in the E-Ray Specification.

Signalling of the end of the RAM initialization sequence by transition of MHDS.CRAM from 1_B to 0_B is correct.

FlexRay_AI.H006 Transmission in ATM/Loopback mode

When operating the E-Ray in ATM/Loopback mode there should be only one transmission active at the same time. Requesting two or more transmissions in parallel is not allowed.

To avoid problems, a new transmission request should only be issued when the previously requested transmission has finished. This can be done by checking registers TXRQ1/2/3/4 for pending transmission requests.

FlexRay_AI.H007 Reporting of coding errors via TEST1.CERA/B

When the protocol engine receives a frame that contains a frame CRC error as well as an FES decoding error, it will report the FES decoding error instead of the CRC error, which should have precedence according to the non-clocked SDL description.

This behaviour does not violate the FlexRay protocol conformance. It has to be considered only when TEST1.CERA/B is evaluated by a bus analysis tool.

FlexRay_AI.H009 Return from test mode operation

The E-Ray FlexRay IP-module offers several test mode options

- Asynchronous Transmit Mode
- Loop Back Mode
- RAM Test Mode
- I/O Test Mode

To return from test mode operation to regular FlexRay operation we strongly recommend to apply a hardware reset via input eray_reset to reset all E-Ray internal state machines to their initial state.

Note: The E-Ray test modes are mainly intended to support device testing or FlexRay bus analyzing. Switching between test modes and regular operation is not recommended.

FlexRay_AI.H011 Behavior of interrupt flags in FlexRay™ Protocol Controller (E-Ray)

In the corner case described below, the actual behavior of the interrupt flags of the FlexRay™ Protocol Controller (E-RAY) differs from the expected behavior.

Note: This behaviour only applies to E-RAY interrupts INT0 and INT1. All other E-RAY interrupts are not affected.

Expected Behavior

When clearing an interrupt flag by software, the resulting value of the flag is expected to be zero.

A hardware event that occurs afterwards then leads to a zero to one transition of the flag, which in turn leads to an interrupt service request.

Actual Behavior in Corner Case

When the interrupt flag is being cleared by software in the same clock cycle as a new hardware event sets the flag again, then the hardware event wins and the flag remains set without being cleared.

As interrupt requests are generated only upon zero to one transitions of the flag, no interrupt request will be generated for this flag until the flag is successfully cleared by software later on.

Workaround

After clearing the flag, the software shall read the flag and repeat clearing until the flag reads zero.

FlexRay_TC.H003 Initialization of E-Ray RAMs

After Power-On reset, the E-Ray RAMs hold arbitrary values which causes ECC errors (MHDS) when a read operation is performed on an E-Ray RAM location. Hence the E-Ray RAMs should be initialized always after a Power-On reset.

Recommendation

The E-Ray RAMs initialization can be performed using the CLEAR_RAMS command of the E-Ray module. A safe initialization sequence of the E-Ray RAM blocks using the CLEAR_RAMS command is described in section “CLEAR_RAMS Command” of chapter “FlexRay™ Protocol Controller (E-Ray)” in the AURIX™ TC3xx Target Specification/User’s Manual.

FPI_TC.H003 Burst write access may lead to data corruption

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.

GETH_AI.H001 Preparation for Software Reset

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_B.

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_B.

GETH_AI.H002 Back-to-back writes to same register - Additional information

After a write operation to a register in the GETH register address space, a certain minimum delay is required before the next write to the same location.

Otherwise, the value written by the second write will not result in an update of the register in the destination clock domain, although the value read back from this location appears to be correct.

The current version of the GETH chapter in the TC3xx User's Manual contains the following statement (see 3rd bullet point in GETH sub-chapter "Registers"):

- ".. Thus, the delay between two writes to the same register location should be at least 4 cycles of the destination clock .."

This information is insufficient. Instead, follow the modified recommendation below.

Recommendation

After a write operation, there should not be any further writes to the same location until the first write is updated. Thus, the delay between two writes to the same register location should be at least 6 cycles of the destination clock

(Reference Clock for the Time Stamp Update (f_{GETH}), PHY receive clock or PHY transmit clock). The delay shall be calculated with the slowest of these clocks.

GTM_TC.H010 Trigger Selection for EVADC and EDSADC

If the GTM output selection in the SELz bit fields for ADC triggers (registers ADCTRIGxOUTy, DSADCOUTSELxy) is changed during SW runtime, multi-bit changes may lead to unintended ADC triggering.

Recommendation

Before changing the trigger source in the GTM output selection fields SELz, ensure that the ADCs at the trigger destination will not react on intermediate state changes of the trigger signals.

GTM_TC.H019 Register GTM_RST - Documentation Update

In the current documentation, bit 0 in register GTM_RST is described as

- **Type**: r
- **Description**: Reserved - Read as zero, should be written as zero.

Documentation Update

Actually, bit 0 in register GTM_RST is implemented as follows:

- **Type**: **rw**
- **Description**: Reserved - Read as zero, **shall** be written as zero.

Note: This Application Hint relates to problem GTM-IP-316 reported by the GTM IP supplier. On this AURIX™ TC3xx device step, the reported problem has no effect, independent of the value written to bit GTM_RST.0. However, GTM_RST.0 shall always be written with 0_B as documented in the register description to ensure compatibility with future versions.

GTM_TC.H021 Interrupt strategy mode selection in IRQ_MODE

The default setting for field IRQ_MODE in register IRQ_MODE is Interrupt Level mode (00_B).

Figure “GTM interrupts” in chapter “GTM Implementation” of the TC3xx User’s Manual shows how the interrupt signal (GTM_IRQ_XXX) triggers an interrupt towards the Interrupt Router (IR), depending on IRQ_MODE.

As described in the text below this figure, while using Level mode, if more internal “interrupt” events are generated (i.e. two TOM channels generating a CCU0 interrupt), just one interrupt signal is sent to the IR, and no more interrupts are triggered until the SW clears the GTM_IRQ_XXX line towards the IR.

Hence, in Level Mode, in some scenarios where another interrupt request is generated by GTM while the ISR handle also requests a SW clear, then, as the interrupt event is dominant over the clear event (for simultaneous interrupt and clear events), GTM_IRQ_XXX is not cleared and remains high. As a consequence, the IR observes no transition on GTM_IRQ_XX. Thus, any forthcoming interrupt events in this scenario are lost as there is no chance to release the CPU IRQ when a collision happens as shown in Figure below.

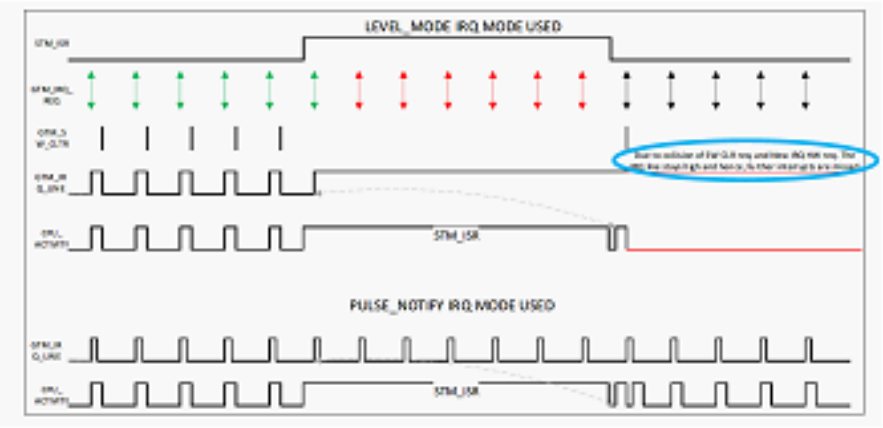

Figure 9 Interrupt Level vs. Pulse-Notify mode - Corner Case Example

If Pulse-Notify mode is selected, every internal trigger will be forwarded to the IR, irrespective of the time of occurrence and the clear event, as the pulse-notify leads to set and reset of GTM_IRQ_XXX as compared to only setting of the GTM_IRQ_XXX line in Level mode.

Recommendation

Therefore, it is recommended to use the Pulse-Notify mode to ensure that none of the interrupts might be lost by the IR even in corner timing cases.

As described above, this scenario in Level mode is only a corner case due to the timing of the SW and ISR handle. If using a different IRQ_MODE setting, evaluate your system performance for sufficient timing margin.

GTM_TC.H022 Field ENDIS_CTRLx in register ATOMi_AGC_ENDIS_CTRL - Documentation Update

The description for settings ENDIS_CTRLx = 10_B and ENDIS_CTRLx = 11_B in register ATOMi_AGC_ENDIS_CTRL in the current version of the TC3xx User's Manual is incorrect.

In will be updated in future revisions of the TC3xx User's Manual as shown in **Figure 10** below.

Field	Bits	Type	Description
ENDIS_CTRLx (x=0-7)	2*x+1:2*x	rw	**ATOM channel x enable/disable update value** If FREEZE=0: If an ATOM channel is disabled, the counter **CN0** is stopped and the output register of SOU unit is set to the inverse value of control bit SL. On an enable event, the counter **CN0** starts counting from its current value. If FREEZE=1: If an ATOM channel is disabled, the counter CN0 is stopped (SOMP, SOMS mode) and each comparison is stopped (SOMC, SOMB mode). On an enable event, the counter CN0 starts counting from its current value or a comparison is restarted. Write of following double bit values is possible: *Note: If the output is disabled (OUTEN[x]=0), the ATOM channel x output ATOM_OUT[x] is the inverted value of bit SL.* 00_B Don't care, bits 1:0 of register ENDIS_STAT will not be changed on an update trigger 01_B Disable channel on an update trigger 10_B Enable channel on an update trigger 11_B Don't change bits 1:0 of this register

Figure 10 Updated description for settings ENDIS_CTRLx = 10B and ENDIS_CTRLx = 11B in register ATOMi_AGC_ENDIS_CTRL

HSCT_TC.H009 High speed dividers five phase clock sequence ordering

For correct operation of the phase correlator and avoiding degradation of BER during operation, the five phase clock generated by high speed dividers must remain in correct sequence.

To prevent the sequence of the five phase clock from being disordered, certain requirements have to be fulfilled when powering on/off of the HSCT physical layer.

Recommendation

Powering on:

Make sure the Peripheral PLL clock is already locked and a correct clock is provided before the HSCT physical layer is powered on by setting bit CONFIGPHY.PON to 1_B.

Powering off:

Note that, according to section “Disable Request (Power-Off)” in the HSCT chapter of the TC3xx User’s Manual, high speed dividers need to be disabled (INIT.RXHD = 000_B and INIT.TXHD = 000_B) before the physical layer is powered off by setting CONFIGPHY.PON to 0_B.

I2C_TC.H008 Handling of RX FIFO Overflow in Slave Mode

If the I2C kernel has detected a RX FIFO overflow in slave mode, a RX_OFL_srq request in generated, the incoming character is discarded, and the kernel puts a not-acknowledge on the bus and changes to listening state.

However, it does not generate an EORXP_ind signal, so that the remaining characters in the FIFO can not be moved out by means of data transfer requests.

Recommendation

Upon an RX FIFO overflow in slave mode, received data may be invalid. However, they may be read from the FIFO e.g. for analysis if required.

In order to flush the FIFO and correctly resume communication

- set bit RUNCTRL.RUN = 0_B (switch to configuration mode),
- set bit RUNCTRL.RUN = 1_B (participate in I2C communication).

LBIST_TC.H001 LBIST signature for configuration A not present in UCB_SSW in specific devices

For specific Engineering Samples (EES/ES), the LBIST signature for configuration A is not present in UCB_SSW (see chapter “UCB_SSW” in the NVM chapter of the AURIX™ User’s Manual, and chapter “LBIST considerations for TC3xx” in TC3xx Appendix).

Affected devices are identified as shown in **Table 24**:

Table 24 Devices with missing LBIST signature for configuration A in UCB_SSW

Device	Design Step	Lot Numbers
TC39x	BA	all
	BB	all
	BC	up to 19835004xyz[1)]
TC38x	AA	all
	AB	all
	AC	all
	AD	up to 19840018xyz[1)]
TC35x	AA	all

1) For specific Lot Number of received samples see respective Status Sheet

The following devices and design steps include the LBIST signature for configuration A in UCB_SSW:

Table 25 Devices including LBIST signature for configuration A in UCB_SSW

Device	Design Step	Lot Numbers
TC39x	BC	> 19835004xyz[1)]
TC38x	AD	> 19840018xyz[1)]
TC35x	AB	all

1) For specific Lot Number of received samples see respective Status Sheet

Recommendation

For the devices and design steps listed in **Table 24** above, if required, store the LBIST signature for configuration A in the internal Flash memory.

MCMCAN_AI.H001 Behavior of interrupt flags in CAN Interface (MCMCAN)

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.

MCMCAN_AI.H002 Busoff Recovery

Note: The following text is copied from Application Note M_CAN_AN004 V1.1 by Robert Bosch GmbH and describes the busoff recovery handling in the MCMCAN module used in AURIX™ TC3xx devices.

The M_CAN enters Busoff state according to CAN protocol conditions. The Busoff state is reported by setting PSR.BO. Additionally, the M_CAN sets CCCR.INIT to stop all CAN operation.

To restart CAN operation, the application software needs to clear CCCR.INIT.

After CCCR.INIT is cleared, the M_CAN's CAN state machine waits for the completion of the Busoff 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 MCMAN chapter of the TC3xx User's Manual the description of Busoff Recovery states that "Once CCCR.INIT has been cleared by the CPU, the device will then wait for 129 occurrences of Bus Idle (129 * 11 consecutive recessive bits) before resuming normal operation. At the end of the Busoff Recovery sequence, the Error Management Counters will be reset".[1)]

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 ECR.REC. Additionally, each occurrence of the Bus Idle condition is flagged by PSR.LEC = 5 = Bit0Error, which triggers an interrupt (IR.PEA) when IR.PEAE is enabled.

While the Busoff Recovery proceeds, the CAN activity is reported as "Synchronizing", PSR.ACT = 0 and PSR.BO remains set. The time from resetting CCCR.INIT to the clearing of PSR.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 Busoff Recovery proceeds.

The M_CAN does not start transmissions while the Busoff Recovery proceeds. When a transmission is requested while the Busoff Recovery proceeds, it will be started after the Recovery has completed and CAN activity entered Idle state, PSR.ACT = 1.

When the Busoff Recovery has completed, PSR.BO, ECR.TEC, and ECR.REC are cleared, and one CAN bit time later PSR.ACT is set to Idle.

After PSR.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 (PSR.ACT = 2) when it samples a dominant bit during Idle state or it will become transmitter (PSR.ACT = 3) when it detects a pending transmission request during Idle state.

1) See Note in description of field LEC and footnote 1) in description of bit BO in Protocol Status Register i (PSRi).

MCMCAN_TC.H006 Unintended Behavior of Receive Timeout Interrupt

On following conditions:

1. Receive timeout feature is enabled (i.e. NTRTR.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 IE.RF0NE, IE.RF1NE or IE.DRXE are 0),

then an unintended receive timeout interrupt (if enabled, i.e. NTRTR.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 IE.RF0NE, IE.RF1NE, or IE.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 IE.RF0NE =1.

MCMCAN_TC.H007 Delayed time triggered transmission of frames

The value written in the bit-field RELOAD of register NTATTRi(i=0-3), NTBTTRi(i=0-3), NTCTTRi(i=0-3) 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(i=0-3) register.

Once a value is written to bit-field RELOAD with bit STRT=1 the timer starts counting. This timer counts one value more than the written value in bit-field RELOAD, then it triggers the transmission of a message object.

Effect

The message object transmission is delayed by one counter cycle with respect to the desired count time written in bit-field RELOAD.

Recommendation

In order to transmit a message object at a specific time, when using one of these registers:

- NTATTRi(i=0-3), NTBTTRi(i=0-3), NTCTTRi(i=0-3),

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

miniMCDS_TC.H001 Program trace of CPUx (x > 0) program start not correct

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 miniMCDS, 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.

MTU_TC.H015 ALM7[0] may be triggered after cold PORST

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.

MTU_TC.H016 MCi_FAULTSTS.OPERR[2] may be triggered at power-up in case LBIST is not run

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.

OCDS_TC.H014 Avoiding failure of key exchange command due to over-write of COMDATA by firmware

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.

OCDS_TC.H015 System or Application Reset while OCDS and lockstep monitoring are enabled

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_B for CPUx, AND
2. Debug System is enabled (CBS_OSTATE.OEN = 1_B), AND
3. CPUx is halted (either in boot-halt state or stopped by debugger tool or in idle mode) when reset is triggered OR CPUx Performance Counters are enabled.

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.

OCDS_TC.H016 Release of application reset via OJCONF may fail

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.

OCDS_TC.H018 Unexpected stop of Startup Software after system/application reset

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 via the DBGSR register.

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

PMS_TC.H003 VDDPD voltage monitoring limits

The EVR pre-regulator (EVRPR) generates the internal VDDPD voltage. Its upper and lower threshold limits are monitored by the VDDPD secondary monitor, while the minimum VLVDRSTC voltage (LVD reset level) is monitored by the VDDPD detector with built-in reference.

The secondary voltage monitor’s upper and lower voltage thresholds for the VDDPD 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 VDDPD internal voltage regulator.

Note: The PREOVVAL and PREUVVAL values of EB_H and $C7_H$, respectively, mentioned in column "Note/Test Conditions" for VDDMON in the Data Sheet are only examples used to characterize the VDDMON accuracy under the specified conditions and shall not be used for the configuration of the EVROVMON2.PREOVVAL and EVRUVMON2.PREUVVAL fields in an application.

Recommendation

- The over-voltage alarm threshold setting in EVROVMON2.PREOVVAL needs not to be modified. The register reset value 0xFE = 1.460 V is appropriate (as well as the next lower value 0xFD = 1.454 V).
- For the under-voltage alarm threshold setting in EVRUVMON2.PREUVVAL:
  - The register reset value 0xBC = 1.079 V (typical) may be kept. It matches the LVD reset level (VLVDRSTC) 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 future versions of the User's Manual, the part for the VDDPD voltage monitor in figure "Voltage Monitoring - VEVRSB, VDDM & VDDPD" in the PMS and PMSLE chapter will be 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.

PMS_TC.H005 SCR clock in system standby mode - Documentation update

The following statement in the fourth paragraph of PMS and PMSLE chapters "Standby ControlleR (SCR) Interface" is incorrect:

"The 70 kHz stand-by clock source is the default SCR clock active in Standby Mode. The SCR clock source may be switched to the internal 20 MHz (derived

from the 100 MHz back-up clock) clock source via SCRCLKSEL register bit thus enabling higher performance on the SCR subsystem."

It shall be replaced by the following statement:

Correction

The 20 MHz stand-by clock source is the default SCR clock active in System Standby Mode enabling higher performance of the SCR subsystem. The SCR clock source may be switched to the internal low-power 70 kHz clock by the SCR subsystem via bit CMCON.OSCPD if bit PMSWCR4.SCRCLKSEL is set to 1_B.

PORTS_TC.H012 LVDS Input Pad on P14.9/10 in BGA-292 Packages

After startup the not-bonded pad P14.11 (in BGA-292 packages) is automatically disabled by the configuration of the initial value of register P14_PDISC.

As (unwanted) side effect the LVDS pad on P14.9/10 is also disabled.

Note: The CMOS functionality on P14.9 and P14.10 is not affected.

Recommendation

When using P14.9 and P14.10 as CMOS pads no further action is necessary.

When using P14.9/P14.10 as LVDS input bit P14_PDISC.PDIS11 has to be changed to 0_B.

In order to avoid disturbance from the not bonded pad P14.11 the pull-up of this pad should be activated by ensuring that P14_IOCR8.PC11 contains 00010_B (which is by default already the case in applications that operate with HWCFG6 = 1_B).

PSI5_TC.H001 No communication error in case of payload length mismatch

When the payload of a frame is higher than the set payload size PDLxy for channel x and slot y, then neither the CRC error nor any other error flag is reliably set in all cases.

When less data is received than the set payload size PDLxy, there are error flags (NBI) that can handle this scenario.

Recommendation

The payload data received should match the configured payload size PDLxy for channel x and slot y (register/field RCRAx.PDLy).

QSPI_TC.H008 Details of the Baud Rate and Phase Duration Control - Documentation update

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 bitfield settings allowing for a flexible bit time variation between the channels in one module.

SAFETY_TC.H001 Features intended for development only – Documentation update to Safety Manual

Certain features of the AURIX™ TC3xx microcontrollers are not considered in SEooC (Safety Element out of Context) development scope as they are

intended to be used only during the system development phase. Consequently, in the productive stage of an Electrical Control Unit (ECU), these features shall not be used by the system integrator. They will be categorized in future revisions of the Safety Manual as “Development-Only” features.

The table below shows an extensive list of these features.

Table 26 AURIX™ TC3xx Features intended for Development-only

Functional Block	Functions	Function Name	Remarks
GTM	TRIGGER_Data_Generation	DTRACE	Interface to trigger tracing of internal GTM signals. Regarded as not being part of any application after development phase is finalized
MCMCAN	Receive	DEBUG_RX	Interface used for debugging purpose to receive DAP frame over CAN frame. Development-only feature as debugging is not part of the safety relevant application assumption
MCMCAN	Transmit	DEBUG_TX	Interface used for debugging purpose to transmit DAP frame over CAN frame. Development-only feature as debugging is not part of the safety relevant application assumption
SCU	Watchdog	WDTDebug Suspend	This function shall only be used during Debug-operation in order to avoid unintended alarms/resets
TRACE	Other interfaces	TRACE	This feature should be used only during development to document and analyze the run-time behavior of a software

Table 26 AURIX™ TC3xx Features intended for Development-only

Functional Block	Functions	Function Name	Remarks
DEBUG	all	DEBUG	Debug feature (OCDS, MCDS) shall only be activated and used during system development phase and shall be deactivated before release for production
CPU	OVERLAY	OVERLAY	Overlay is typically used for evaluation of SW calibration parameters. Once the latter is finalized, the feature shall be deactivated
CPU	Data Acquisition	OLDA	Online Data Acquisition feature shall only be used during SW evaluation phase
Firmware	Data Storage in EMEM	EMEM-Prolog	This feature is aimed to be used for calibration purposes. It shall be deactivated together with final storage of SW in PFLASH
CAN/LIN	Code Loading	BSL	This feature is only used for initial device programming. It is not needed anymore once the application software is stable

SAFETY_TC.H002 SM[HW]:CPU.PTAG:ERROR_DETECTION – Documentation update to Safety Manual

Section 6.97 “SM[HW]:CPU.PTAG:ERROR_DETECTION” in chapter “Safety Mechanisms” of the current version of the AURIX™ TC3xx Safety Manual contains an incorrect part marked **bold** in the sentence copied below:

Incorrect sentence

“The SRAM implements a DED ECC logic. This mechanism detects SBE and DBE during read operations. In case an DBE is detected or **ECE is disabled** an SBE is detected a critical uncorrectable error alarm is forwarded to the SMU.”

Corrected sentence

“The SRAM implements a DED ECC logic. This mechanism detects SBE and DBE during read operations. In case a DBE is detected or a SBE is detected a critical uncorrectable error alarm is forwarded to the SMU.”

Summary

The CPU.PTAG memory is protected by an error detection code ECC capable to detect both single and double bit errors. Single bit errors (SBE) will lead to an uncorrectable error alarm regardless of the ECE configuration.

CPU PTAG does not offer SBE error correction capabilities. Indeed, both SBEs and DBEs will trigger an uncorrectable critical error alarm.

The correct hardware implementation of the ECC of the CPU PTAG memory is described in section 4.2.3.3.2 “Monitoring Concept” of the Safety Manual. See the note copied below:

“NOTE: CPU.PTAG do not offer SBE error correction capabilities. SBE and DBE are detected and an uncorrectable error is generated as described in SM[HW]:CPU.PTAG:ERROR_DETECTION.”

Note: Absolute section numbers in the text above apply to V1.06 of the AURIX™ TC3xx Safety Manual.

SAFETY_TC.H003 ESM[SW]:EDSADC:VAREF_PLAUSIBILITY and ESM[SW]:EVADC:VAREF_PLAUSIBILITY – Additional information

The safety mechanisms ESM[SW]:EDSADC:VAREF_PLAUSIBILITY and ESM[SW]:EVADC:VAREF_PLAUSIBILITY require the system integrator to implement a check of the reference voltages (V_{AREF}, V_{AGND}) of the EDSADC and EVADC of the AURIX™ TC3xx, respectively, either by an external monitor or by internally converting a known signal and comparing the result with the expected value.

Typically, when executing these safety mechanisms, two conversions take place and V_{AREF} comparison is done. Since only a low discharging is applied, an additional resistance of several tens of kOhms which could be caused by an external failure effect does not play that major role. This leads to the test being considered as passed.

However, when the application SW is switching to normal operating mode (e.g. 200 conversions every ms), the discharge of the VAREF input is significantly higher and a systematical failure will happen on all channels due to the increased additional resistance.

Recommendation:

The V_{AREF} plausibility check shall be performed under representative application conditions.

SAFETY_TC.H004 ESM[HW]:PMS:VEXT_VEVRSB_OVERVOLTAGE – Wording update

In the last paragraph of field “Description” in section 5.3 ESM[HW]:PMS:VEXT_VEVRSB_OVERVOLTAGE of the current version of the AURIX™ TC3xx Safety Manual, the word “could” will be replaced by the word “will” (marked in **bold** in the text below):

Current Description

“It is assumed that the system detects and disables the external voltage supplies VEXT and VEVRSB of the MCU in case of an over-voltage condition with the continuous monitoring of the external voltage supplies of the MCU.

The Over-voltage can be divided into 2 regions:

- Up to absolute maximum rating: Mechanisms are in a separate domain. Exceeding the operating conditions for longer than the specified time may lead to an increase of the micro-controller failure rate.

- Above absolute maximum rating: this is the reliability issue. In case internal circuitry is damaged, LBIST and HWBIST will detect it during start-up.

- If microcontroller is permanently damaged due to the exposure to the voltage level exceeding the specified maximum supply voltage, self-test (LBIST, HW BIST) could detect this fault during start-up.”

Modified Description

It is assumed that the system detects and disables the external voltage supplies VEXT and VEVRSB of the MCU in case of an over-voltage condition with the continuous monitoring of the external voltage supplies of the MCU.

The Over-voltage can be divided into 2 regions:

- Up to absolute maximum rating: Mechanisms are in a separate domain. Exceeding the operating conditions for longer than the specified time may lead to an increase of the micro-controller failure rate.

- Above absolute maximum rating: this is the reliability issue. In case internal circuitry is damaged, LBIST and HWBIST will detect it during start-up.

- If microcontroller is permanently damaged due to the exposure to the voltage level exceeding the specified maximum supply voltage, self-test (LBIST, HW BIST) **will** detect this fault during start-up.

Note: Absolute section numbers in the text above apply to V1.06 of the AURIX™ TC3xx Safety Manual.

SAFETY_TC.H006 SM[HW]:PMS:VDD_MONITOR – Documentation update

The following sentence including an absolute value of the core supply voltage (V_{DD}) in section 6.349 “SM[HW]:PMS:VDD_MONITOR” of chapter “Safety Mechanisms” in the current version of the AURIX™ TC3xx Safety Manual

- Detects whether VDD (1.3V supply generated by EVR or from external) is within expected range.

shall be changed as listed below:

Updated sentence

- Detects whether the VDD supply voltage is within the expected range.

The typical value for V_{DD} is 1.25 V. See chapter “Electrical Specification” in the corresponding TC3xx device Data Sheet for absolute values and limits.

Note: Absolute section numbers in the text above apply to V1.06 of the AURIX™ TC3xx Safety Manual.

SAFETY_TC.H007 SM[HW]:CLOCK:PLL_LOSS_OF_LOCK_DETECTION – Documentation update

The term “VCO” in the following sentence included in section 6.43 “SM[HW]:CLOCK:PLL_LOSS_OF_LOCK_DETECTION” of chapter “Safety Mechanisms” in the current version of the AURIX™ TC3xx Safety Manual

- The PLL has a lock detection that supervises the VCO part of the PLL in order to differentiate between stable and unstable VCO circuit behavior.

shall be replaced by “**DCO**” as listed below:

Updated sentence

- The PLL has a lock detection that supervises the **DCO** part of the PLL in order to differentiate between stable and unstable **DCO** circuit behavior.

As described in chapter “Clocking System” of the TC3xx User’s Manual, the PLL implementation in the TC3xx devices has a DCO, it does not have a VCO.

Note: Absolute section numbers in the text above apply to V1.06 of the AURIX™ TC3xx Safety Manual.

SAFETY_TC.H008 Link between ESM[SW]:CONVCTRL:ALARM_CHECK and SM[HW]:CONVCTRL:PHASE_SYNC_ERR - Additional information

ESM[SW]:CONVCTRL:ALARM_CHECK is the SW measure to trigger the HW mechanism SM[HW]:CONVCTRL:PHASE_SYNC_ERR (see section “Safety Measures” in the CONVCTRL chapter of the TC3xx User’s Manual).

As of today, there is no link between these two mechanisms in the Safety Manual. To provide this information to system integrators, ESM[SW]:CONVCTRL:ALARM_CHECK (section 5.13 in the current version of the AURIX™ TC3xx Safety Manual) shall be added to the “**Tests**” field of

SM[HW]:CONVCTRL:PHASE_SYNC_ERR (section 6.47 in Safety Manual), as shown below:

6.47 SM[HW]:CONVCTRL:PHASE_SYNC_ERR - Update to field “Tests”

Description

The Safety Mechanism supervises the operation of the Phase Synchronizer by monitoring the parity bit of its prescaler (PHSCFG.PHSDIV) and the counter value.

...

Tests

ESM[SW]:CONVCTRL:ALARM_CHECK

...

Note: Absolute section numbers in the text above apply to V1.06 of the AURIX™ TC3xx Safety Manual.

SCR_TC.022 Effect of application or system reset and warm PORST on MC77_ECCD and MC78_ECCD for SCR RAMs

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.

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.

SCR_TC.H009 RAM ECC Alarms in Standby Mode

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).

SCR_TC.H010 HRESET command erroneously sets RRF flag

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 (a-c) after an HRESET, the issues with the HRESET command can be solved:

- Execute HRESET
  a) Execute HSTATE to remove reset bit from shift register.
  b) Perform JTAG tool reset to remove flag RRF (receive register flag).
  c) Execute HCOMRST to remove flag TRF (transmit register flag).

SCR_TC.H011 Hang-up when warm PORST is activated during Debug Monitor Mode

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.

SCR_TC.H012 Reaction in case of XRAM ECC Error

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_B.
- Enable the RAM ECC Error for NMI generation by setting bit NMIRAMECC in register SCR_NMICON to 1_B.

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_B.

SCR_TC.H014 Details on WDT pre-warning period

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 overflown, and it is not possible to stop this reaction (reset).

SCU_TC.H016 RSTSTAT reset values - documentation update

Table “Reset Values of RSTSTAT” in the SCU chapter of the current version of the User’s Manual is missing the scenario for “LVD Reset”. In addition, the reset value for “Cold PowerOn Reset” needs to be modified as shown in the following table:

Table 27 Reset Values of RSTSTAT - Update

Reset Type	Reset Value	Note
Cold PowerOn Reset	$0XX1\ 0000_H$	
LVD Reset	$1001\ 0000_H$	

Details:

For more detailed information about the reset triggers please refer to table “Voltage Monitoring” in the PMS chapter. Following information can be found there:

- Cold PowerOn Reset:
  - As the table shows, bit PORST is always set with the corresponding reset source (SWD, EVR33 or EVRC). Therefore the “Cold PowerOn Reset” value is $0XX1\ 0000_H$.
- LVD Reset:
  - Bit STBYR is only set when the corresponding voltage drops below the LVD voltage limit. Bit PORST is set on LVD reset as well. Therefore the “LVD Reset” value is $1001\ 0000_H$.

– Bits SWD, EVR33 and EVRC are not set in this case, because after LVD reset the system has an initial ramp-up which will not set these bits.

SCU_TC.H020 Digital filter on ESRx pins - Documentation update

As described in the SCU and PMS chapters of the TC3xx User's Manual, the input signals $\overline{\text{ESR0}}$ / $\overline{\text{ESR1}}$ can be filtered. The filter for $\overline{\text{ESRx}}$ is enabled via bit PMSWCR0.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} ≥ 20 MHz in RUN mode.

SCU_TC.H021 LBIST execution affected by TCK/DAP0 state

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.

SCU_TC.H022 Effect of LBIST execution on SRAMs - Additional information

In sub-chapter "LBIST Support" in the SCU chapter of the TC3xx User's Manual, the section starting with:

"A successfully finished LBIST procedure is indicated by the LBISTCTRL0.LBISTDONE bit.."

shall be extended in future revisions of the SCU chapter as shown below:

Additional information

A successfully finished LBIST procedure is indicated by the LBISTCTRL0.LBISTDONE bit. Value of LBISTCTRL0.LBISTDONE bit is not affected by the System or Application reset (it preserves its value). In case of warm or cold power-on reset, it resets LBISTDONE bit to 0, and soon after, if LBIST is configured to start, it will get its new result value.

Note: SRAM redundancy registers are part of the scan chain and hence corrupted by LBIST. Therefore, SRAMs contents are not reliable after LBIST and shall be initialized after LBIST, prior to usage.
DLMU SRAM with standby capability can be used instead to store information such as the LBIST execution count.

SENT_TC.H006 Parameter V_{ILD} on pads used as SENT inputs

Some port pins may have restrictions when used as SENT inputs, depending on the number of active neighbor pins (on the pad frame) and their output driver setting.

In the implementation of the SENT module and product integration within Infineon Technologies products there are never negative values for V_{ILD}, so V_{ILDmin} is 0 mV. Considering the same tolerance as the SENT standard V_{ILDmax} is 100 mV.

Note: All SENT port pins not listed in the tables below have no restrictions on their application usage as SENT inputs.

Table 28 SENT input pads and considered neighbors for TC38x

Considered left neighbors		SENT input		Considered right neighbors	
		Pad	Channel		
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

Table 28 SENT input pads and considered neighbors for TC38x (cont'd)

Considered left neighbors		SENT input		Considered right neighbors	
		Pad	Channel		
P31.9	P31.12	P31.11	23C	P31.14	P31.13
P31.10	P31.9	P31.12	24C	P31.11	P31.14

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 29 Effect of Driver Settings Fss, Sms, Sm on SENT inputs for TC38x

SENT Channel			Neighbors: Fast pads configured as Fss, others Sms/Sm	
Name	**Number**	**Pin**	**VILD(DIST4N)**	**VILD(DIST2N)**
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 30 Effect of Driver Settings Fsm, Sms, Sm on SENT inputs for TC38x

SENT Channel			Neighbors: Fast pads configured as Fsm, others Sms/Sm	
Name	**Number**	**Pin**	**VILD(DIST4N)**	**VILD(DIST2N)**
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

Table 31 Effect of Driver Settings Fm, Sms, Sm on SENT inputs for TC38x

SENT Channel			Neighbors: Fast pads configured as Fm, others Sms/Sm	
Name	**Number**	**Pin**	**VILD(DIST4N)**	**VILD(DIST2N)**
SENT:SENT11D	11D	P15.4	OK	OK
SENT:SENT20C	20C	P31.8	OK	OK
SENT:SENT21C	21C	P31.9	OK	OK
SENT:SENT22C	22C	P31.10	OK	OK
SENT:SENT23B	23B	P02.12	OK	OK
SENT:SENT23C	23C	P31.11	OK	OK
SENT:SENT24C	24C	P31.12	OK	OK

Table 32 Abbreviations used for pad configuration

Symbol	Pad type	Driver Strength / Edge Mode
Fss	Fast	strong driver, sharp edge
Fsm	Fast	strong driver, medium edge
Fm	Fast	medium driver
Sms	Slow	medium driver, sharp edge
Sm	Slow	medium driver

Recommendation

From the tables above, following is the conclusion based on the measured V_{ILD} values for each pad in different configurations:

Table 33 Conclusion for SENT application usage

Symbol	Conclusion for SENT application usage
OK	V_{ILD} is below the standard threshold (100mV) and hence pin can be used in the mentioned configuration.
x	V_{ILD} 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): • Configure the neighboring pads have to weaker edge mode / driver strength (Fsm or Fm instead of Fss), • Use SENT input with 2N neighbors instead of 4N.

SMU_TC.H010 Clearing individual SMU flags: use only 32-bit writes

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 (e.g. alarm action triggering) is not affected even if the software unintentionally clears additional bits by not using a 32-bit write as recommended.

SMU_TC.H012 Handling of SMU alarms ALM7[1] and ALM7[0]

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 Target Specification/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 v1.0).

Also, in order to clear the SMU alarms ALM7[1] and ALM7[0], it is necessary to clear the alarms within this MC40.

Recommendation

Therefore, the register addresses listed below have to be written as follows:

0xF00638F0 = (16-bit write) 0x0

0xF0063810 = (16-bit write) 0x0

SMU_TC.H013 Increased Fault Detection for SMU Bus Interface (SMU_CLC Register)

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.

SMU_TC.H015 Calculation of the minimum active fault state time TFSP_FS - Additional information

In Figure "Reference clocks for FSP timings" in the SMU chapter of the TC3xx User's Manual, the "**&**" symbol in the formula for the minimum active fault state time TFSP_FS designates "field concatenation":

TFSP _FS =
TSMU _FS *(SMU_FSP.TFSP_HIGH[] **&** SMU_FSP.TFSP_LOW [] + 1)

Note: Field TFSP_LOW is hardcoded to 0x3FFF in register SMU_FSP. So if SMU_FSP.TFSP_HIGH is 0x1, then SMU_FSP.TFSP_HIGH[] & SMU_FSP.TFSP_LOW[] = 0x7FFF.

SRI_TC.H001 Using LDMST and SWAPMSK.W instructions on SRI mapped Peripheral Registers (range 0xF800 0000-0xFFFF FFFF)

The LDMST and SWAPMSK.W instructions in the AURIX™ microcontrollers are intended to provide atomicity as well as bit-wise operations to a targeted memory location or peripheral register. They are also referred to as Read-Modify-Write (RMW) instructions.

The bit-manipulation functionality is intended to provide software a mechanism to write to individual bits in a register, without affecting other bits. The bits to be written can be selected via 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 (i.e. address range 0xF800 0000 to 0xFFFF FFFF, including (if available) 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 – i.e the SRI is locked between the READ and the WRITE transaction.

STM_TC.H004 Access to STM registers while STMDIV = 0

If accesses to STM kernel registers are performed while field STMDIV = 0_H in CCU Clock Control register CCUCON0 (i.e. clock f_{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_H.

Recommendation

Do not access any STM kernel register while CCUCON0.STMDIV = 0_H.