Information note

N° 10375AERRA

Dear customer,

With this Infineon Technologies AG information note, we would like to inform you about the following

Errata sheet V1.11 affecting products TC37x_AA

Infineon Technologies AG
Postal address D-81726 München Internet www.infineon.com Headquarters Am Campeon 1-15, D-85579 Neubiberg Phone +49 (0)89 234-0

Information note

N° 10375AERRA

▸ **Products affected**	Please refer to attached affected product list "INF_10375AERRA_[customer-no].pdf"

▸ **Detailed change information**

Subject	Errata sheet V1.11 affecting products TC37x_AA
Reason	Update of Errata Sheet from version V1.10 to V1.11

Description

Old	New
▪ Errata Sheet V1.10	▪ Errata Sheet V1.11

▸ **Product identification**	Not applicable (no change of product)
▸ **Impact of change**	Assessment in Application required!
▸ **Attachments**	INF_10375AERRA_[customer-no].pdf affected product list 3_cip10375 errata sheet v1.11 (MyICP link)
▸ **Intended start of delivery**	Not applicable

If you have any questions, please do not hesitate to contact your local sales office.

ERR 10375AERRA

Errata sheet V1.11 affecting products TC37x_AA

Affected products sold to FUTURE ELECTRONICS INC. (4000624)

Sales name	SP number	OPN	Package	Customer part number
SAK-TC377TP-96F300S AA	SP001694648	TC377TP96F300SAAKXUMA1	PG-LFBGA-292-11	TC377TP96F300SAAKXUMA1
SAL-TC377TP-96F300S AA	SP001724092	TC377TP96F300SAALXUMA1	PG-LFBGA-292-11	

Errata Sheet

Rel. 1.11, 2022-11-11

Device TC37x

Marking/Step (E)ES-AA, AA

Package see Data Sheet

10375AERRA

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

Table 1 Current Documentation[1)]

AURIX™ TC3xx User’s Manual	V2.0.0	2021-02
AURIX™ TC37x Appendix to User’s Manual	V2.0.0	2021-02
TC37x AA-Step Data Sheet	V1.1	2021-03
TriCore TC1.6.2 Core Architecture Manual: - Core Architecture (Vol. 1) - Instruction Set (Vol. 2)	 V1.2.2 V1.2.2	 2020-01-15 2020-01-15
AURIX™ TC3xx Safety Manual	v2.0	2021-05-03

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
- **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 Addendum.
   This Errata Sheet covers several device variants. If an issue is related to a particular module, and this module is not specified for a specific device variant, this issue does not apply to this device variant.
   For example, issues with identifier “EMEM” do not apply to devices where no Extension Memory is specified.
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. 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	2019-07-19	First version for TC37x step AA
1.1	2019-12-02	Update: • New/updated text modules see column “Change” in tables 3..5 of errata sheet V1.1 • 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 – 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

Table 2 History List (cont'd)

Version	Date	Remark
1.1	...	... continued: • Removed: – 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 – 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
1.2	2020-03-31	Update: • New/updated text modules see column "Change" in tables 3..5 of errata sheet V1.2 • 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
1.3	2020-07-06	Update: • New/updated text modules see column "Change" in tables 3..5 of errata sheet V1.3
1.4	2020-10-23	Update: • New/updated text modules see column "Change" in tables 3..5 of errata sheet V1.4 • Text module SCR_TC.022 (Effect of application or system reset and warm PORST on MC77_ECCD and MC78_ECCD for SCR RAMs) moved from chapter "Application Hints" to chapter "Functional Problems"

Table 2 History List (cont'd)

Version	Date	Remark
1.5	2021-01-22	Update: • New/updated text modules see column "Change" in tables 3..5 of errata sheet V1.5
1.6	2021-04-22	Update: • New/updated text modules see column "Change" in tables 3..5 of errata sheet V1.6 – Text modules already published in TC3xx Errata Advance Information 2021_02: MCMCAN_AI.022, SMU_TC.H012 – Text modules already published in TC3xx Errata Advance Information 2021_03: FlexRay_TC.H004, GETH_TC.H003, HSCT_TC.H010, SCR_TC.023, SENT_TC.H007 • Removed SCU_TC.032, included description in update of SCU_TC.031 (Bits SCU_STSTAT.HWCFGx (x=1-5) could have an unexpected value in application if pins HWCFGx are left unconnected)
1.7	2021-07-23	Update: • New/updated text modules see column "Change" in tables 3..5 of errata sheet V1.7 – Text modules already published in TC3xx Errata Advance Information 2021_05: GTM_AI.362, GTM_AI.364, GTM_AI.367, GTM_AI.370/371, GTM_AI.374..376 – Text modules already published in TC3xx Errata Advance Information 2021_06: FLASH_TC.055, GTM_AI.H004 (update see this errata sheet), MCMCAN_TC.H008, MTU_TC.018, PMS_TC.015

Table 2 History List (cont'd)

Version	Date	Remark
1.8	2021-11-04	Update: • New/updated text modules see column "Change" in tables 3..5 of errata sheet V1.8 – Text modules already published in TC3xx Errata Advance Information 2021-09: FLASH_TC.056, GTM_AI.358, GTM_AI.387, MCMCAN_AI.023, RESET_TC.H006, SAFETY_TC.023, SAFETY_TC.024 • Table 2 (History List) of errata sheet V1.7: corrected "GTM.AI.*" to "GTM_AI.*" • Removed – GTM_AI.H004: replaced by GTM_AI.387 – GTM_AI.262, GTM_AI.263: only apply to TC39x step AA

Table 2 History List (cont'd)

Version	Date	Remark
1.9	2022-03-25	Update: • New/updated text modules see column "Change" in tables 3..5 of errata sheet V1.9 – Text modules already published in TC3xx Errata Advance Information 2021-12: ASCLIN_TC.012, CCU_TC.005, GTM_AI.398, GTM_AI.400, GTM_TC.025, GTM_TC.H027, I2C_TC.H009, PMS_TC.H011, SAFETY_TC.H019, SAFETY_TC.H020, SCR_TC.024, SCU_TC.033 – Text modules already published in TC3xx Errata Advance Information 2022-01: ADC_TC.H033, ASCLIN_TC.H007, GETH_AI.H003, GTM_AI.406, GTM_AI.408, OCDS_TC.H015, SMU_TC.H016 – Note: GTM_AI.263 and GTM_AI.262 re-included (also apply to TC37x), with updated workaround 2 for GTM_AI.262 • Removed SCU_TC.H016 (RSTSTAT reset values - documentation update): updated in TC3xx User's Manual V1.2.0 and following

Table 2 History List (cont'd)

Version	Date	Remark
1.10	2022-07-29	Update: • Documentation reference changed to Safety Manual v2.0 (see table 1) • New/updated text modules see column "Change" in tables 3..5 of errata sheet V1.10 – Text modules already published in TC3xx Errata Advance Information 2022-05: ADC_TC.H043, CPU_TC.H021, FLASH_TC.H021, GTM_AI.H425, MTU_TC.H016, SCU_TC.H025 – Text modules already published in TC3xx Errata Advance Information 2022-06: GTM_AI.441, GTM_AI.450, GTM_AI.451, GTM_AI.454, INT_TC.H006, LBIST_TC.H003, MCMCAN_AI.024, MCMCAN_AI.H002, PSI5_TC.005, SAFETY_TC.026, SAFETY_TC.027 • Removed GTM_AI.425: replaced by GTM_AI.H425 • Removed due to updates in Safety Manual v1.11 and following: – SAFETY_TC.004 (section 5.1 ESM[HW]:MCU:LBIST_MONITOR) – SAFETY_TC.H001 (section 3.2.1 Purpose of the SEooC) – SAFETY_TC.H003 (section 5.42 ESM[SW]:EDSADC:VAREF_PLAUSIBILITY and 5.51 ESM[SW]:EVADC:VAREF_PLAUSIBILITY) – SAFETY_TC.H004 (section 5.3 ESM[HW]:PMS:VEXT_VEVRSB_OVERVOLTAGE) – SAFETY_TC.H010 (Appendix A, sections for TC37xA and TC36xA)

Table 2 History List (cont'd)

Version	Date	Remark
... 1.10 (cont'd)	...	...: • Removed due to updates in Safety Manual v1.12 and following: – SAFETY_TC.002 (section 6.325 SM[HW]:NVM.PFLASH:FLASHCON_MONITOR), – SAFETY_TC.006 (section 6.428 SM[HW]:SMU:CCF_MONITOR) – SAFETY_TC.007 (6.346 SM[HW]:PMS:VDDM_MONITOR) – SAFETY_TC.022 (Appendix A, Alarms tables after Reset for TC37x) – SAFETY_TC.H002 (section 6.97 SM[HW]:CPU.PTAG:ERROR_DETECTION) – SAFETY_TC.H006 (section 6.349 SM[HW]:PMS:VDD_MONITOR) – SAFETY_TC.H007 (section 6.43 SM[HW]:CLOCK:PLL_LOSS_OF_LOCK_DETECTION) – SAFETY_TC.H008 (section 6.47 SM[HW]:CONVCTRL:PHASE_SYNC_ERR) – SAFETY_TC.H015 (section 6.336 SM[HW]:NVM:STARTUP_PROTECTION) – SAFETY_TC.H016 (section 5.32 ESM[SW]:CPU:SOFTERR_MONITOR)

Table 2 History List (cont'd)

Version	Date	Remark
1.11	2022-11-11	Update: • Documentation reference changed to TC3xx User's Manual V2.0.0 (see table 1) • New/updated text modules see column "Change" in tables 3..5 of errata sheet V1.11 – Text modules already published in TC3xx Errata Advance Information 2022-08: EDSADC_TC.H004, FlexRay_TC.H005, GTM_TC.028, SCR_TC.019 – Text modules already published in TC3xx Errata Advance Information 2022-09: CPU_TC.H021, INT_TC.H006, MBIST_TC.H001 • Removed: – ASCLIN_TC.012 (not applicable to design implementation in AURIX™ family) • Removed due to updates in TC3xx documentation (see also table 1): – ADC_TC.P014 (updated figure "Equivalent Circuitry for Analog Inputs" included in TC37x AA Data Sheet V1.1) – ADC_TC.H034 (updated footnote on QCONV in table "VADC 5V" in TC37x AA Data Sheet V1.1) – BROM_TC.H017 (corrected in footnote 1 to table ""ALL CHECKS PASSED" indication by CHSW" in Firmware chapter of product specific Appendix V1.3.0 (and newer versions)) – CCU_TC.004 (updated in description of register OSCCON in Clocking System chapter of TC3xx UM V1.6.0 (and newer versions))

Table 2 History List (cont'd)

Version	Date	Remark
... 1.11 (cont'd)	...	... • Removed due to updates in TC3xx documentation (see also table 1): – CPU_TC.H020 (updated in tables “Register Overview – SPR”, “Register Overview – DLMU”, sections “Scratch Pad SRAMs”, “DLMU SRAMs”, “Register access enable Protection”, “Safety Protection registers” in CPU chapter of TC3xx UM V1.5.0 (and newer versions)) – EDSADC_TC.003 (corrected in table “Settling Time Summary” in EDSADC chapter of TC3xx UM V1.6.0 (and newer versions)) – EDSADC_TC.P002 (updated footnote on table “DSADC 5V” in TC37x AA Data Sheet V1.0 (and newer versions)) – FLASH_TC.H019 (updated in section “Write Burst Once” in NVM chapter of TC3xx UM V1.4.0 (and newer versions)) – GETH_AI.018 (updated in section “Description of the Transmit Checksum Offload Engine” in GETH chapter of TC3xx UM V2.0.0) – GETH_AI.H002 (updated in section “Registers” in GETH chapter of TC3xx UM V1.4.0 (and newer versions)) – GETH_TC.001 (updated in table “Clock Lines of Ethernet MAC” in GETH chapter of product specific Appendix V1.3.0 (and newer versions)) – GETH_TC.P001 (corrected in chapter “Operating Conditions” in TC37x AA Data Sheet V1.1)

Table 2 History List (cont'd)

Version	Date	Remark
... 1.11 (cont'd)	...	... • Removed due to updates in TC3xx documentation (see also table 1): – GETH_TC.H003 (updated in table "ETH MII Signal Timing Parameters" and "ETH RMII Signal Timing Parameters" in TC37x AA Data Sheet V1.1) – GTM_TC.021 (updated in description of registers GTM_CANOUTSEL0, GTM_CANOUTSEL1 in GTM chapter of product specific Appendix V1.3.0 (and newer versions)) – GTM_TC.022 (updated in description of register ATOMi_AGC_ENDIS_STAT in GTM chapter of TC3xx UM V2.0.0) – GTM_TC.H022 (updated in description of register ATOMi_AGC_ENDIS_CTRL in GTM chapter of TC3xx UM V1.4.0 (and newer versions)) – MTU_TC.019 (updated in description of register MCi_MCONTROL in MTU chapter of TC3xx UM V1.4.0 (and newer versions)) – MTU_TC.H017 (updated in section "SRAM Error Detection & Correction (EDC/ECC)" in MTU chapter of TC3xx UM V2.0.0) – PINS_TC.P001 (updated in chapter "Pin Reliability in Overload" and table "Example Inactive Lifetime Temperature Profile" in TC37x AA Data Sheet V1.1)

Table 2 History List (cont'd)

Version	Date	Remark
... 1.11 (cont'd)	...	... • Removed due to updates in TC3xx documentation (see also table 1): – PMS_TC.012 (updated in section "Primary under-voltage monitors and Cold PORST" in PMS/PMSLE chapter of TC3xx UM V1.6.0/V1.5.0 (and newer versions)) – PMS_TC.H005 (updated in section "Standby ControlleR (SCR) Interface" in PMS/PMSLE chapter of TC3xx UM V1.6.0 (and newer versions)) – RESET_TC.P003 (updated in table "Reset" in TC37x AA Data Sheet V1.1) – SAFETY_TC.H018 (updated in Safety Manual v2.0, sections 4.3.1 (Safety Related Functions, table in Introduction), 4.3.2.6 (Digital Acquisition ASIL B/D), 4.3.2.7 (Digital Actuation ASIL B/D)) – SCR_TC.H013 (notes added after the description of register RCT_CON in SCR chapter of TC3xx UM V1.1.0 (and newer versions)) – SCR_TC.H015 (updated in table "WUF Configuration Registers Address Map" in SCR chapter of TC3xx UM V2.0.0) – SCU_TC.H022 (updated in section "Functional Description" of chapter "LBIST Support" in SCU chapter of TC3xx UM V2.0.0) – SMU_TC.H015 (updated in figure "Reference clocks for FSP timings" and in description of register FSP in SMU chapter of TC3xx UM V1.6.0 (and newer versions))

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

Table 3 Functional Deviations

Functional Deviation	Short Description	Change	Page
ADC_TC.095	Ramp trigger ignored when ramp ends		38
BROM_TC.013	CAN BSL does not send error message if no valid baudrate is detected		38
BROM_TC.014	Lockstep Comparator Alarm for CPU0 after Warm PORST, System or Application Reset if Lockstep is disabled		39
BROM_TC.016	Uncorrectable ECC error in Boot Mode Headers		39
CCU_TC.005	ASC and CAN bootstrap loaders may not work if external clock is missing		40
CPU_TC.130	Data Corruption when ST.B to local DSPR coincides with external access to same address		40
CPU_TC.131	Performance issue when MADD/MSUB instruction uses E0/D0 register as accumulator		41
CPU_TC.132	Unexpected PSW values used upon Fast Interrupt entry		42
CPU_TC.133	Test sequence for DTAG single or double bit errors		44
DAP_TC.005	DAP client_read: dirty bit feature of Cerberus' Triggered Transfer Mode		45
DAP_TC.007	Incomplete client_blockread telegram in DXCM mode when using the "read CRCup" option		45

Table 3 Functional Deviations (cont'd)

Functional Deviation	Short Description	Change	Page
DMA_TC.059	ACCEN Protection not implemented for ERRINTRr		46
DMA_TC.066	DMA Double Buffering Operations - Update Address Pointer		46
DMA_TC.067	DMA Double Buffering Software Switch Buffer Overflow		47
DMA_TC.068	DMA Double Buffering Lost DMA Request		47
FLASH_TC.053	Erase Size Limit for PFLASH		48
FLASH_TC.055	Multi-bit errors detected by PFlash are not communicated to SPB masters		49
FLASH_TC.056	Reset value for register HF_ECCC is 0x0000 0000 - Documentation correction		50
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		51
FlexRay_AI.088	A sequence of received WUS may generate redundant SIR.WUPA/B events		52
FlexRay_AI.089	Rate correction set to zero in case of SyncCalcResult=MISSING_TERM		52
FlexRay_AI.090	Flag SFS.MRCS is set erroneously although at least one valid sync frame pair is received		53
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		54
FlexRay_AI.092	Initial rate correction value of an integrating node is zero if pMicroInitialOffsetA,B = 0x00		54

Table 3 **Functional Deviations** (cont'd)

Functional Deviation	Short Description	Change	Page
FlexRay_AI.093	Acceptance of startup frames received after reception of more than gSyncNodeMax sync frames		55
FlexRay_AI.094	Sync frame overflow flag EIR.SFO may be set if slot counter is greater than 1024		56
FlexRay_AI.095	Register RCV displays wrong value		57
FlexRay_AI.096	Noise following a dynamic frame that delays idle detection may fail to stop slot		57
FlexRay_AI.097	Loop back mode operates only at 10 MBit/s		58
FlexRay_AI.099	Erroneous cycle offset during startup after abort of startup or normal operation		59
FlexRay_AI.100	First WUS following received valid WUP may be ignored		60
FlexRay_AI.101	READY command accepted in READY state		60
FlexRay_AI.102	Slot Status vPOC!SlotMode is reset immediately when entering HALT state		61
FlexRay_AI.103	Received messages not stored in Message RAM when in Loop Back Mode		61
FlexRay_AI.104	Missing startup frame in cycle 0 at coldstart after FREEZE or READY command		62
FlexRay_AI.105	RAM select signals of IBF1/IBF2 and OBF1/OBF2 in RAM test mode		63
FlexRay_AI.106	Data transfer overrun for message transfers Message RAM to Output Buffer (OBF) or from Input Buffer (IBF) to Message RAM		64

Table 3 Functional Deviations (cont'd)

Functional Deviation	Short Description	Change	Page
GETH_AI.001	Packets with Destination Address (DA) mismatch are delayed until EOP is received in threshold (cut-through) mode		67
GETH_AI.008	Application Error Along with Start-of-Packet Can Corrupt the FCS Field of the Previous Frame in the MAC Pipeline		68
GETH_AI.009	Corrupted Rx Descriptor Write Data		69
GETH_AI.010	Fatal Bus Error Interrupt Might Be Generated for Incorrect DMA Channel		70
GETH_AI.011	Receive Queue Overflow at End of Frame Along with SPRAM Read-Write Conflict Can Cause Data Loss		71
GETH_AI.012	Incorrect Flexible PPS Output Interval When Fine Time Correction Method is Used		71
GETH_AI.013	False Dribble and CRC Error Reported in RMII PHY 10Mbps Mode		73
GETH_AI.014	Receive DMA Channel Generates Spurious Receive Watchdog Timeout Interrupt		74
GETH_AI.015	MAC Receive VLAN Tag Hash Filter Always Operates in Default Mode		76
GETH_AI.016	Receive DMA Header Split Function Incorrectly Overruns the Allocated Header Buffer		77
GETH_AI.017	Carrier-Sense Signal Not Generated When False Carrier Detected in RGMII 10/100 Mbps Mode		79
GETH_TC.002	Initialization of RGMII interface		80
GTM_AI.254	TIM TDU: TDU_STOP=b101 not functional		81

Table 3 Functional Deviations (cont'd)

Functional Deviation	Short Description	Change	Page
GTM_AI.262	SPEC-DPLL: PSSC/PSTC behavior description incorrect	Update	82
GTM_AI.263	DPLL: DPLL_STATUS.LOCK1 flag (0 ->1) delayed after direction change when DPLL operating in DPLL_CTRL_0.RMO = 1		84
GTM_AI.304	MCS: Scheduling modes Single Prioritization and Multiple Prioritization are not functional		85
GTM_AI.306	DPLL: DPLL_NUTC.syn_t_old, DPLL_NUSC.syn_s_old not updated according specification		85
GTM_AI.307	IRQ: AEI_IM_ADDR is not set in GTM_IRQ_NOTIFY register if cluster 0 is disabled		86
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		87
GTM_AI.318	MCS: NARD(I) instruction terminates unexpectedly		88
GTM_AI.319	(A)TOM: Unexpected (A)TOM_CCU1TCx_IRQ in up/down counter mode		89
GTM_AI.320	ATOM: Unexpected restart of a SOMS oneshot cycle while ATOM[i]_CH[x]_CM0 is zero		89
GTM_AI.322	DPLL: PSTC, PSSC not updated correctly after fast pulse correction completed (DPLL_CTRL1.PCM1/2 = 0)		90

Table 3 Functional Deviations (cont'd)

Functional Deviation	Short Description	Change	Page
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)		91
GTM_AI.325	TIM: Bits ACB[2:1] lost on interface to ARU (always zero)		92
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		94
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	Update	94
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		103
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		104
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		105
GTM_AI.334	DPLL RAM content of single address can be corrupted after leaving debug mode		105
GTM_AI.335	TOM output signal to SPE not functional if up/down counter mode is configured		106

Table 3 Functional Deviations (cont'd)

Functional Deviation	Short Description	Change	Page
GTM_AI.336	GTM Bus Bridge: Incorrect AEI access execution in case the previous AEI access was aborted with the access timeout abort function		107
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		108
GTM_AI.340	TOM/ATOM: Generation of TRIG_CCU0/TRIG_CCU1 trigger signals skipped in initial phase of A/TOM SOMP one-shot mode		109
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		110
GTM_AI.344	DPLL: Incorrect AEI_STATUS on internal MCS2DPLL interface on valid and implemented address accesses		112
GTM_AI.345	SPE: Incorrect behaviour of direction change control via SPE_CMD.SPE_CTRL_CMD bits		113
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		114
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	Update	115

Table 3 **Functional Deviations** (cont'd)

Functional Deviation	Short Description	Change	Page
GTM_AI.348	DPLL: Correction of missing pulses delayed after start of pulse generation		116
GTM_AI.349	TOM-SPE: OSM-Pulse width triggered by SPE_NIPD for selected CMU_FXCLK not correct		117
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		118
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)		119
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		120
GTM_AI.353	SPEC-ATOM: Specification of the smallest possible PWM Period in SOMP mode wrong, when ARU_EN=1		121
GTM_AI.354	MCS: Unresolved hazard resulting from RAW (Read After Write) dependency		123
GTM_AI.357	MCS: instructions XCHB, SETB, and CLRB do not suppress register write		124
GTM_AI.358	TOM/ATOM: Synchronous update of working register for RST_CCU0=1 and UDMODE=0b01 not correct		125
GTM_AI.359	TOM: Both edges on TOM_OUT_T at unexpected times for RST_CCU0=1 and UDMODE>0		126
GTM_AI.360	SPEC-(A)TOM: PCM mode (BITREV=1) is only available for UDMODE=0		127

Table 3 Functional Deviations (cont'd)

Functional Deviation	Short Description	Change	Page
GTM_AI.361	IRQ: Missing pulse in single-pulse interrupt mode on simultaneous interrupt and clear event		128
GTM_AI.362	MCS: Using wrong WURM mask during execution of instruction WURMX or WURCX		129
GTM_AI.364	ATOM: ARU read request does not start at expected timepoint in UDMODE=1 and UDMODE=3		130
GTM_AI.367	MCS: Instructions WURMX and WURCX implement invalid extended register set for argument A		131
GTM_AI.370	TOM/ATOM: Unexpected reset of CN0 in up-down counter mode and CM0=2		132
GTM_AI.371	MCS: Instruction MWRIL applies unexpected address offset calculation		133
GTM_AI.374	SPEC-ATOM: Statement on timing of duty cycle output level change not correct for SOMP up/down-counter mode		134
GTM_AI.375	ATOM: Data from ARU are read only once in SOMC mode even though ARU blocking mode is disabled while FREEZE=1 and ENDIS=0		135
GTM_AI.376	TOM/ATOM: Interrupt trigger signals CCU0TC_IRQ and CCU1TC_IRQ are delayed by one CMU_CLK period related to the output signals		136

Table 3 Functional Deviations (cont'd)

Functional Deviation	Short Description	Change	Page
GTM_AI.387	DPLL: Wrong calculation of pulse generator frequency for DPLL_CTRL_0.AMT/S=1 and DPLL_CTRL_11.ADT/S=1 when number of pulses (DPLL_CTRL_0.MLT or DPLL_MLS1/2.MLS1/2) is too small		137
GTM_AI.398	DPLL: Incorrect DPLL_THVAL calculation leading to a false direction decision in case tbu_ts0 wraps around		137
GTM_AI.400	MCS-RTL: Division instruction may produce unexpected memory overflow and wrong results		138
GTM_AI.404	MCS-RTL: Division instruction reports unrelated ECC error		139
GTM_AI.406	(A)TOM: FREEZE mode has no effect on (A)TOM_OUT_T in up-down counter mode with RST_CCU0=1		139
GTM_AI.408	(A)TOM-RTL: Missing edge on output signal (A)TOM_OUT when CN0 is reset with force update event		141
GTM_AI.409	DPLL: Flags of registers DPLL_STA_FLAG are not set		142
GTM_AI.411	A change of the BRIDGE_MODE register might be delayed indefinitely		143
GTM_AI.419	TIM: Potentially wrong capture values		144
GTM_AI.422	DPLL: Wrong DPLL_RDT_S_ACT/DPLL_RDT_T_ACT value in case of overflow correction		146
GTM_AI.428	DPLL: Pulse correction is executed twice		147

Table 3 Functional Deviations (cont'd)

Functional Deviation	Short Description	Change	Page
GTM_AI.429	TIM: Missing glitch detection interrupt event		148
GTM_AI.430	TIM: Unexpected increment of filter counter	Update	149
GTM_AI.431	TIM: Glitch detection interrupt event of filter is not a single cycle pulse		151
GTM_AI.441	DPLL: Missing pulse correction in case of DPLL_CTRL_1.SMC = 1		152
GTM_AI.450	DPLL: Stored time stamp values do not consider filter delays		153
GTM_AI.451	DPLL: Wrong measured position stamps in RAM		153
GTM_AI.454	(A)TOM: No output if trigger generation feature is used		154
GTM_AI.456	DPLL: No action calculation	New	155
GTM_AI.458	DPLL: Missing TOR interrupt and status flag	New	155
GTM_AI.462	(A)TOM: Missing CCU0TC_IRQ interrupt signal		156
GTM_AI.463	DPLL: DPLL_PVT not cleared after direction change	New	157
GTM_AI.464	DPLL: Pulse correction executed twice when DPLL_CTRL_11.INCF1/2 is activated	New	158
GTM_AI.465	(A)TOM: Missing CCU0TC_IRQ interrupt signal for UDMODE > 0		159
GTM_AI.466	TOM: Unexpected behavior of TOM_OUT_T for UDMODE>0		160
GTM_AI.474	DPLL: DPLL_PSTC, DPLL_PSSC erroneously modified	New	161

Table 3 Functional Deviations (cont'd)

Functional Deviation	Short Description	Change	Page
GTM_AI.475	DPLL: Incorrect values of DPLL_RCDT_TX, DPLL_RCDT_SX	New	162
GTM_AI.476	MCS: Unexpected instruction execution while disabling of MCS channel	New	162
GTM_AI.477	DPLL: DPLL_DCGI interrupt not triggered	New	163
GTM_AI.478	DPLL: Incorrect calculation of DPLL_THVAL, DPLL_THVAL2	New	164
GTM_TC.018	DPLLRAM trace data can be wrong		165
GTM_TC.019	ARU can not be traced if GTM cluster 5 is disabled		166
GTM_TC.020	Debug/Normal read access control via bit field ODA.DRAC		166
GTM_TC.025	Register DPLL_IRQ_NOTIFY - Documentation update for bits SORI and DCGI		167
GTM_TC.026	Table "GTM IP Application Constraints" #1 (DPLL) - Documentation correction		168
GTM_TC.028	Incorrect MCS behavior when SSH registers are accessed while MCS is running	New	168
HSCT_TC.012	HSCT sleep mode not supported		169
HSCT_TC.013	Internal Loopback Mode not reliable		169
MCMCAN_AI.015	Edge filtering causes mis-synchronization when falling edge at Rx input pin coincides with end of integration phase		170
MCMCAN_AI.017	Retransmission in DAR mode due to lost arbitration at the first two identifier bits		171
MCMCAN_AI.018	Tx FIFO Message Sequence Inversion		173
MCMCAN_AI.019	Unexpected High Priority Message (HPM) interrupt		175

Table 3 Functional Deviations (cont'd)

Functional Deviation	Short Description	Change	Page
MCMCAN_AI.022	Message order inversion when transmitting from dedicated Tx Buffers configured with same Message ID		177
MCMCAN_AI.023	Incomplete description in section *.5.2 "Dedicated Tx Buffers" and *.5.4 "Tx Queue" of the M_CAN documentation in the User's Manual related to transmission from multiple buffers configured with the same Message ID		178
MCMCAN_AI.024	Frame transmitted despite confirmed transmit cancellation		180
miniMCDS_TC.005	TriCore wrap around write access causes redundant miniMCDS message		181
miniMCDS_TC.006	Selection of SRI trace sources		182
miniMCDS_TC.007	Selection of CPU trace sources		182
miniMCDS_TC.008	MCDS kernel reset shall not be used		183
MTU_TC.012	Security of CPU Cache Memories During Runtime is Limited		183
MTU_TC.017	Unexpected alarms after application reset		184
MTU_TC.018	Gated SRAM alarms		184
PADS_TC.011	Pull-ups activate on specific analog inputs upon PORST	Update	186
PADS_TC.013	Buffer type definition for P21.2: no ES functionality - Data Sheet documentation correction		187

Table 3 Functional Deviations (cont'd)

Functional Deviation	Short Description	Change	Page
PMS_TC.005	Voltage rise at P33 and P34 up to V_{EVRSB} during start-up and up to $V_{LVDRSTSB}$ during power-down		187
PMS_TC.006	PORST not released during Cold Power-on Reset until VDDM is available		188
PMS_TC.007	VDDP3 or VDD Overvoltage during start-up may not be detected by PBIST		188
PMS_TC.011	VEXT supplied PU2 and PD2 pads always in tristate after standby entry - Documentation correction		189
PMS_TC.013	Minimum V_{DD} supply voltage for f_{SRI} > 200 MHz on TC375TI		191
PMS_TC.014	Parasitic coupling on shared ADC pins depending on supply voltages		191
PMS_TC.015	EVRC synchronization – Documentation update for register EVRSDCTRL11 (PMS) and EVRSDCTRL2 (PMSLE)		192
PSI5_TC.005	Incorrect read pointer upon two consecutive RDFn read operations when two or more channels are configured		193
QSPI_TC.006	Baud rate error detection in slave mode (error indication in current frame)		194
QSPI_TC.009	USR Events for PT1=2 (SOF: Start of Frame)		195
QSPI_TC.010	Move Counter Mode - USR Events for PT1=4 (RBF: Receive Buffer Filled)		195
QSPI_TC.013	Slave: No RxFIFO write after transmission upon change of BACON.MSB		196
QSPI_TC.014	Slave: Incorrect parity bit upon TxFIFO underflow		196

Table 3 Functional Deviations (cont'd)

Functional Deviation	Short Description	Change	Page
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		197
QSPI_TC.017	Slave: Reset when receiving an unexpected number of bits		197
SAFETY_TC.023	MCU infrastructure Safety Related Function - Documentation Update		198
SAFETY_TC.024	Clock alive monitor for f_{SPB} - Documentation update		198
SAFETY_TC.025	Wrong alarm listed in safety mechanism SM[HW]:SRI:SRI_TRANSACTION_INTEGRITY		199
SAFETY_TC.026	Alarm for SM[HW]:IR:CFG_MONITOR - Documentation update		199
SAFETY_TC.027	Single point fault detection for lockstep CPUs - Documentation update		199
SCR_TC.015	Bit SCU_PMCON1.WCAN_DIS does not disable WCAN PCLK input		200
SCR_TC.016	DUT response to first telegram has incorrect C_START value		201
SCR_TC.018	SSC Receive FIFO not working		201
SCR_TC.019	Accessing the XRAM while SCR is in reset state	Update	202
SCR_TC.020	Stored address in mon_RETH may be wrong after a break event		202
SCR_TC.021	RTC not counting after reset if P33.10 is high		203

Table 3 **Functional Deviations** (cont'd)

Functional Deviation	Short Description	Change	Page
SCR_TC.022	Effect of application or system reset and warm PORST on MC77_ECCD and MC78_ECCD for SCR RAMs		204
SCR_TC.023	External interrupts EXINT0, EXINT1 may get locked		204
SCR_TC.024	Field ADRES in register ADCOMP_RES - Documentation correction		205
SCU_TC.031	Bits SCU_STSTAT.HWCFGx (x=1-5) could have an unexpected value in application if pins HWCFGx are left unconnected		205
SCU_TC.033	TESTMODE pin shall be held at static level during LBIST		206
SMU_TC.012	Unexpected alarms when registers FSP or RTC are written		207
SMU_TC.013	Unexpected setting of Alarm Missed Event bit xAEM in Alarm Executed Status register SMU_AEX		208

Table 4 **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		210
ADC_TC.P015	Increased RMS Noise for TC374* and TC375* devices		210
FLASH_TC.P003	Program Flash Erase Time per Multi-Sector Command		212

Table 4 Deviations from Electrical- and Timing Specification (cont'd)

AC/DC/ADC Deviation	Short Description	Change	Page
PADS_TC.P011	High Performance LVDS Pads - Documentation update to Data Sheet		212
PORST_TC.P002	V_{IH} and V_{IL} definition for PORST pad - Additional Data Sheet footnote		213

Table 5 Application Hints

Hint	Short Description	Change	Page
ADC_TC.H026	Additional Waiting Phase in Slow Standby Mode		214
ADC_TC.H032	ADC accuracy parameters - Definition		214
ADC_TC.H033	Basic Initialization Sequence for Primary and Secondary EVADC Groups		215
ADC_TC.H035	Effect of input leakage current on Broken Wire Detection		216
ADC_TC.H036	Minimum Input Buffering Time - Additional information		218
ADC_TC.H037	CPU read access latency to result FIFO buffer		218
ADC_TC.H039	DMA read access latency to result FIFO buffer		219
ADC_TC.H043	Information on supervision signal $V_{ANACOMM}$ not relevant - Documentation update		220
ASCLIN_TC.H001	Bit field FRAMECON.IDLE in LIN slave tasks		220

Table 5 Application Hints (cont'd)

Hint	Short Description	Change	Page
ASCLIN_TC.H006	Sample point position when using three samples per bit - Documentation update		220
ASCLIN_TC.H007	Handling TxFIFO and RxFIFO interrupts in single move mode – Documentation update		221
ASCLIN_TC.H008	SPI master timing – Additional information to Data Sheet characteristics		222
BROM_TC.H008	CAN BSL does not support DLC = 9 and DLC = 11		223
BROM_TC.H009	Re-Enabling Lockstep via BMHD		224
BROM_TC.H014	SSW behavior in case of wrong state or uncorrectable error in UCBs - Documentation Update		224
BROM_TC.H015	Different initial values for CPU0_PMEM SSH registers in MTU after cold PORST if SOTA/SWAP is enabled		225
CCU_TC.H012	Configuration of the Oscillator- Documentation Update		225
CLC_TC.H001	Description alignment for bits DISR, DISS, EDIS in register CLC - Documentation Update		226
CPU_TC.H019	Semaphore handling for shared memory resources		227
CPU_TC.H021	Resource update failure despite correct SW synchronization upon retried FPI write transactions by CAN and ERAY modules	Update	231
DAM_TC.H002	Triggering DAM MEMCON.RMWERR and INTERR flags		232

Table 5 Application Hints (cont'd)

Hint	Short Description	Change	Page
DSADC_TC.H010	Support for synchronous use of two or more DSADC channels		233
DTS_TC.H002	Unexpected alarms after start-up/wake-up when temperature is close to lower/upper limit	Update	234
EDSADC_TC.H001	Auxiliary filter cleared with start of integration window - Additional information		234
EDSADC_TC.H003	Behavior of EDSADC result register in case of hardware controlled integration		235
EDSADC_TC.H004	CIC3 filter properties - Documentation update	New	236
EVR_TC.H001	External input capacitor value - Additional Data Sheet footnote		236
FLASH_TC.H021	Flash Wait State configuration		236
FlexRay_AI.H004	Only the first message can be received in External Loop Back mode		237
FlexRay_AI.H005	Initialization of internal RAMs requires one eray_bclk cycle more		238
FlexRay_AI.H006	Transmission in ATM/Loopback mode		238
FlexRay_AI.H007	Reporting of coding errors via TEST1.CERA/B		238
FlexRay_AI.H009	Return from test mode operation		238
FlexRay_AI.H011	Behavior of interrupt flags in FlexRay™ Protocol Controller (E-Ray)		239
FlexRay_TC.H003	Initialization of E-Ray RAMs - Documentation Update		240
FlexRay_TC.H004	Bit WRECC in register TEST2 has no function		240

Table 5 Application Hints (cont'd)

Hint	Short Description	Change	Page
FlexRay_TC.H005	ERAY OTGB2 trigger set active even if disabled	New	241
FPI_TC.H003	Burst write access may lead to data corruption		241
GETH_AI.H001	Preparation for Software Reset		242
GETH_AI.H003	Undefined behavior when LD bit is set and buffer length B1L or B2L is zero - Additional information		243
GETH_TC.H002	Stopping and Starting Transmission - Additional information		244
GPT12_TC.H002	Bits TxUD and TxUDE in incremental interface mode - Additional information		245
GTM_AI.H425	MCS: Instructions BRDI and BWRI evaluate unused address bits		246
GTM_TC.H010	Trigger Selection for EVADC and EDSADC		247
GTM_TC.H019	Register GTM_RST - Documentation Update		247
GTM_TC.H021	Interrupt strategy mode selection in IRQ_MODE		248
GTM_TC.H027	Register ODA (OCDS Debug Access) - Documentation update		249
HSCT_TC.H009	High speed dividers five phase clock sequence ordering		249
HSCT_TC.H010	Interface control command timing on the LVDS ports		250
I2C_TC.H008	Handling of RX FIFO Overflow in Slave Mode		251
I2C_TC.H009	Connections of Serial Clock Inputs		252

Table 5 Application Hints (cont'd)

Hint	Short Description	Change	Page
INT_TC.H006	Number of SRNs supporting external interrupt/service requests – Documentation update	Update	252
LBIST_TC.H003	Update reset behavior of LBISTCTRL0 and LBISTCTRL3 register - Additional information		253
MBIST_TC.H001	Destructive MBIST requires DSPR0 initialization	New	254
MCMCAN_AI.H001	Behavior of interrupt flags in CAN Interface (MCMCAN)		254
MCMCAN_AI.H002	Busoff Recovery		255
MCMCAN_TC.H001	Behavior of undefined data bytes read from Receive Buffer	New	257
MCMCAN_TC.H006	Unintended Behavior of Receive Timeout Interrupt		257
MCMCAN_TC.H007	Delayed time triggered transmission of frames		258
MCMCAN_TC.H008	Parameter "CAN Frequency" - Documentation update to symbol in Data Sheet		259
miniMCDS_TC.H001	Program trace of CPUx (x > 0) program start not correct		259
MSC_TC.H014	Symbol T_A in specification of FCLPx clock period in Data Sheet - Additional information		260
MTU_TC.H015	ALM7[0] may be triggered after cold PORST		260
MTU_TC.H016	MCi_FAULTSTS.OPERR[2] may be triggered at power-up in case LBIST is not run		261

Table 5 Application Hints (cont'd)

Hint	Short Description	Change	Page
OCDS_TC.H014	Avoiding failure of key exchange command due to overwrite of COMDATA by firmware		261
OCDS_TC.H015	System or Application Reset while OCDS and lockstep monitoring are enabled		262
OCDS_TC.H016	Release of application reset via OJCONF may fail		263
OCDS_TC.H018	Unexpected stop of Startup Software after system/application reset		263
PACKAGE_TC.H004	Exposed pad dimensions and package outlines for QFP packages - Updates to TC37x Data Sheet		264
PADS_TC.H008	Overload coupling for LVDS RX pads – Additional information		265
PADS_TC.H009	Pull-ups active on P33/P34 pins in standby mode when SCR is disabled	New	267
PMS_TC.H003	VDDPD voltage monitoring limits		267
PMS_TC.H008	Interaction of interrupt and power management system - Additional information		268
PMS_TC.H009	Interaction of warm reset and standby mode transitions		270
PMS_TC.H011	Supply mode and topology selection - Allowed combinations of VEXT and VDDM - Documentation update		271
PSI5_TC.H001	No communication error in case of payload length mismatch		272
QSPI_TC.H008	Details of the Baud Rate and Phase Duration Control - Documentation update		272

Table 5 Application Hints (cont'd)

Hint	Short Description	Change	Page
RESET_TC.H006	Certain registers may have different reset values than documented in TC3xx User's Manual - Documentation update	Update	272
SAFETY_TC.H011	SM[HW]:GTM:TOM_TOM_MONITORING_WITH_IOM – Additional information		277
SAFETY_TC.H013	ESM[SW]:SYS:MCU_FW_CHECK - Access to MC40 FAULTSTS register – Additional information		280
SAFETY_TC.H017	Safety Mechanisms requiring initialization - Documentation update		280
SAFETY_TC.H019	SM[HW]:NVM.FSIRAM:REG_MONITOR_TEST should not be considered		284
SAFETY_TC.H020	Test of SM[HW]:VMT:REG_MONITOR is missing - Documentation update		285
SCR_TC.H009	RAM ECC Alarms in Standby Mode		285
SCR_TC.H010	HRESET command erroneously sets RRF flag		286
SCR_TC.H011	Hang-up when warm PORST is activated during Debug Monitor Mode		286
SCR_TC.H012	Reaction in case of XRAM ECC Error		286
SCR_TC.H014	Details on WDT pre-warning period		287
SCU_TC.H019	Connection on ERU input E_REQ7(5)		287
SCU_TC.H020	Digital filter on ESRx pins - Documentation update		288
SCU_TC.H021	LBIST execution affected by TCK/DAP0 state		288
SCU_TC.H023	Behavior of bit RSTSTAT.PORST after wake-up from standby mode		288
SCU_TC.H025	Field EEA in register CHIPID - Additional information		289

Table 5 Application Hints (cont'd)

Hint	Short Description	Change	Page
SENT_TC.H006	Parameter V_{ILD} on pads used as SENT inputs		290
SENT_TC.H007	Range for divider value DIV - Documentation correction		293
SMU_TC.H010	Clearing individual SMU flags: use only 32-bit writes		293
SMU_TC.H012	Handling of SMU alarms ALM7[1] and ALM7[0]		294
SMU_TC.H013	Increased Fault Detection for SMU Bus Interface (SMU_CLC Register)		295
SMU_TC.H016	SMU_stdby restriction for using P33.8 as Emergency Stop input		295
SMU_TC.H017	Handling of ALM21[7] when safety flip-flop self-test is executed		296
SRI_TC.H001	Using LDMST and SWAPMSK.W instructions on SRI mapped Peripheral Registers (range 0xF800 0000-0xFFFF FFFF)		297
SSW_TC.H001	Security hardening measure for the startup behavior		298
STM_TC.H004	Access to STM registers while STMDIV = 0		299

2 Functional Deviations

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.

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.005 ASC and CAN bootstrap loaders may not work if external clock is missing

Description

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

Effects

No code download for initial device programming is started.

Note: This problem may only occur for initial start up of unprogrammed devices. If automatic start of the external crystal oscillation is programed in UCB DFLASH, the problem will not occur.

Workaround

Trigger reset and retry if bootstrap loader does not respond.

If connection to the device is possible via a debug tool, use the tool to reconfigure OSCCON.MODE = 00_B (when using an external crystal), and then trigger reset.

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.

CPU_TC.133 Test sequence for DTAG single or double bit errors

The error injection method described in the section “13.5.2.1.4 Error injection and Alarm Triggering” in the MTU chapter of the TC3xx User’s Manual using the ECCMAP method is not sufficient to trigger alarms pertaining to the DTAG RAM of each CPU. In the case of DTAG RAM, an alternate method relying on the Read Data and Bit Flip register (RDBFL) must be used instead.

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

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

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

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

Workaround

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

DTAG SRAM test sequence

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

1. Read an DTAG SRAM location into RDBFL register
   (see section 13.3.5.1.6 “Reading a Single Memory Location”).

2. Flip some bit in RDBFL[0].
3. Writeback the content of the RDBFL into the DTAG SRAM
   (see section 13.3.5.1.7 “Writing a Single Memory Location”).
4. Read the DTAG SRAM location again.

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

Note: Absolute chapter numbers in the text above refer to MTU chapter version V7.4.12 included in the TC3xx User’s Manual V1.6.0. They may change if used in other versions of this document.

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.

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.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.

FLASH_TC.055 Multi-bit errors detected by PFlash are not communicated to SPB masters

Problem

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

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

that a bus error is returned to the reading master.

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

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

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

When any master reads PFlash through the SRI (this includes CPUs reading the PFlash located at another CPU or its local bank with disabled Fast Path) a single transfer with multi-bit error returns a bus error but a burst read is reporting this error using a forced “Transaction ID Error” (concept described in “On-Chip

System Connectivity {and Bridges}”). The bus error is always communicated back to the master. The handling of the Transaction ID Error however is master specific.

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

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

When an SPB master like HSM performs a burst read from a PFlash bank this SRI Transaction ID Error terminates at the SFI_F2S bridge. The SPB master does not receive a bus error and continues operation with wrong data. The SFI_F2S bridge signals the error to the XBar for alarm generation.

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

Workaround

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

As described above also SFI_F2S bridge informs the XBar to generate an alarm “Safety Mechanism: Built-in SRI Error Detection; Alarm: XBAR0 Bus Error Event” to the SMU. With HSM as requesting master the XBAR0 just captures the occurrence of this error but doesn’t capture address or other transaction data in its Error Capture registers.

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

FLASH_TC.056 Reset value for register HF_ECCC is 0x0000 0000 - Documentation correction

In the register description for register HF_ECCC (DF0 ECC Control Register) in the TC3xx User’s Manual, the application reset value is documented as 0xC000 0000.

However, this register is cleared by the startup software SSW, and the user software will read the reset value of 0x0000 0000.

Documentation correction

- The application reset value for register HF_ECCC is 0x0000 0000.

Note: The user must consider that field HF_ECCC.TRAPDIS is 00_B after reset, which means a bus error trap is generated if an uncorrectable ECC error occurs upon read from DF0, or read from DF1 when DF1 is configured as not HSM_exclusive.

<u>FlexRay_AI.087</u> 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.

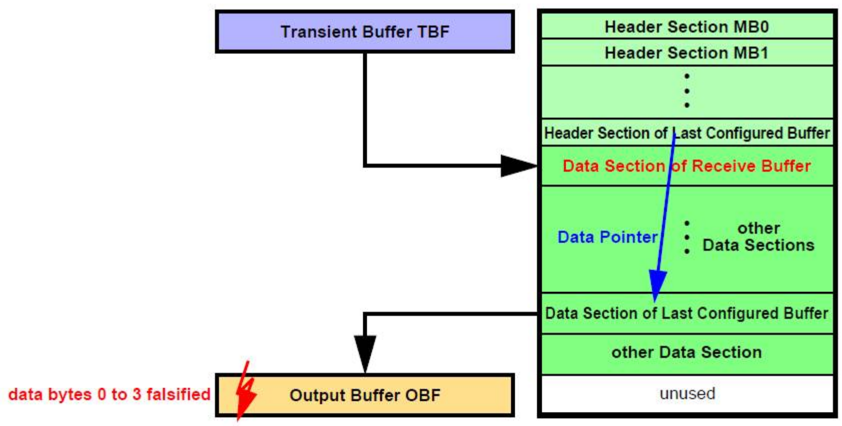

Figure 1 Message buffer transfer from Message RAM to OBF

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.

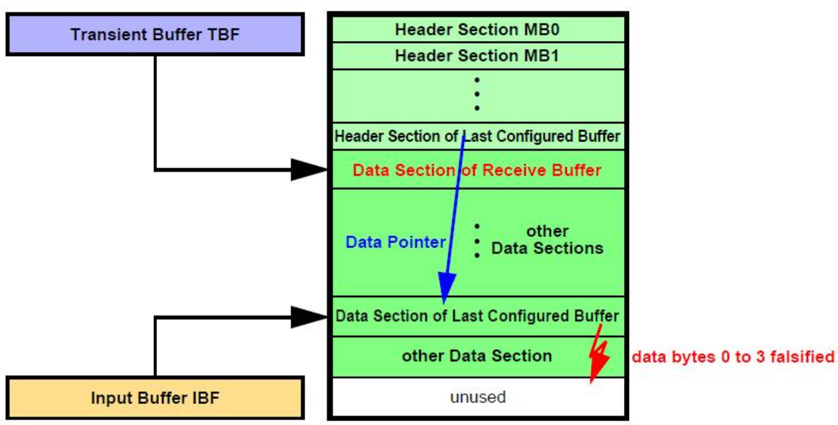

Figure 2 Message buffer transfer from IBF to Message RAM

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.

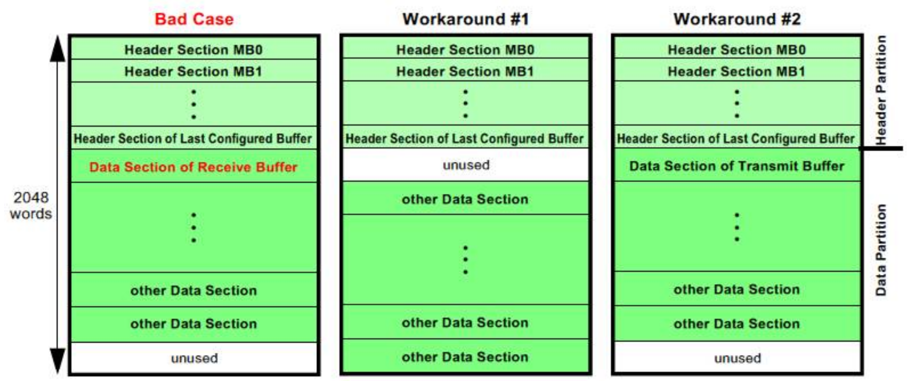

Figure 3 Message RAM Configurations

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.

<u>GETH_AI.008</u> 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_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 phase shift of the Transmit Clock TXCLK relative to TXD/TXCTL of ±180° (@1000Mbit/s), or ±36° (@100Mbit/s), or ±3.6° (@10Mbit/s).

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

Workaround

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

- Finish active transfers and make sure that transmitters and receivers are set to stopped state:
  - Check the RPSx and TPSx status bit fields in register DMA_DEBUG_STATUS0/1.
  - Check that MTL_RXQ0_DEBUG, MTL_RXQi_DEBUG, MTL_TXQ0_DEBUG and MTL_TXQi_DEBUG register content is equal to zero.
    Note: it may be required to wait 70 f_{SPB} cycles after the last reset before checking if RXQSTS in MTL_RXQ0_DEBUG and MTL_RXQi_DEBUG are zero.

  - Wait until a currently running interrupt is finished and globally disable GETH interrupts.
- Ensure GETH_GPCTL.EPR = 000_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.

<u>GTM_AI.254</u> 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 SPEC-DPLL: PSSC/PSTC behavior description incorrect

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

When changing from emergency to normal mode (DPLL_CTRL_0.RMO = 1->0) the RAM1b.PSTC value is not calculated as specified.

- In the specification it is written:

For changing from normal mode to emergency mode at the following TRIGGER slope (according to the RMO value in the shadow register) the PSSC value is calculated by PSSC = PSSM + correction value (forward direction) or PSSC = PSSM - correction value (backward direction) with the correction value = inc_cnt1 - nmb_t.

For changing from emergency mode to normal mode at the following STATE slope (according to the RMO value in the shadow register) the PSTC value is calculated by PSTC = PSTM + correction value (forward direction) or PSTC = PSTM - correction value (backward direction) with the correction value = inc_cnt1 - nmb_s. In case no further TRIGGER or STATE events the CPU has to perform the above corrections.

- Instead the following behavior should be specified:

For changing from normal mode to emergency mode at the following STATE slope (according to the RMO value in the shadow register) the PSSC value is calculated by PSSC = PSSM + correction value (forward direction) or PSSC = PSSM - correction value (backward direction) with the correction value = inc_cnt1 - nmb_s.

For changing from emergency mode to normal mode at the following TRIGGER slope (according to the RMO value in the shadow register) the PSTC value is calculated by PSTC = PSTM + correction value (forward direction) or PSTC = PSTM - correction value (backward direction) with the correction value = inc_cnt1 - nmb_t. In case no further TRIGGER or STATE events the CPU has to perform the above corrections.

Scope

DPLL

Effects

The PSSC value is not updated at the following trigger slope but at the following STATE slope with the value PSSC=PSSM +/- correction value with correction value = inc_cnt1 - nmb_s. This might lead to an unexpected behavior of the GTM IP.The PSTC value is not updated at the following STATE slope but at the following TRIGGER slope with the value PSTC=PSTM +/- correction value with correction value = inc_cnt1 - nmb_t. This might lead to an unexpected behavior of the GTM IP.

Workaround 1

If possible leave PSSC or PSTC as is.

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

Workaround 2

Alternatively, the application can disable the DPLL via DPLL_CTRL_1.DEN=0 and then re-start the DPLL in emergency or normal mode with the following sequence:

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

```
GTM_DPLL_CTRL_1.B.PIT = 1; -- only as an example
GTM_DPLL_CTRL_1.B.DMO = 0; -- only as an example
GTM_DPLL_CTRL_1.B.COA = 0; -- only as an example
GTM_DPLL_CTRL_1.B.SYSF = 1; -- only as an example
GTM_DPLL_CTRL_1.B.TSL = 1; -- only as an example
GTM_DPLL_CTRL_1.B.SSL = 3; -- only as an example
GTM_DPLL_CTRL_0.B.SEN/TEN = 1; -- only as an example
```

Then switch into emergency/normal mode and enable the DPLL again.

```
GTM_DPLL_CTRL_0.B.RMO = 1 or 0; -- only as an example
GTM_DPLL_CTRL_0.B.SEN/TEN = 1; -- only as an example
GTM_DPLL_CTRL_1.B.DEN = 1; -- only as an example
```

In this case the behavior for PSSC/PSTC is different:

For DPLL_CTRL_0.RMO = 0->1: PSSC=PSSM.

For DPLL_CTRL_0.RMO = 1->0: PSTC=PSTM.

If in this case nevertheless a different value for PSSC/PSTC is necessary the value could be written by CPU interface as already written in the specification. 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.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.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.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.

<u>GTM_AI.323</u> 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 6 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 7 Memory locations critical from CPU/DMA and MCS (N>1 wait cycles)

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

Table 7 Memory locations critical from CPU/DMA and MCS (N>1 wait cycles) (cont'd)

Register name or memory location	Type	Comment
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 6** and **Table 7** 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.

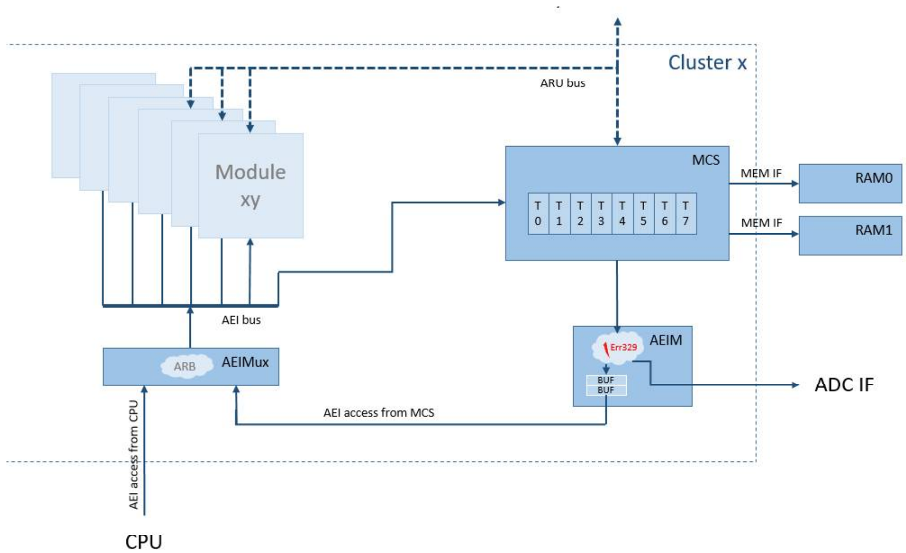

Figure 4 Cluster-internal AEI network

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.

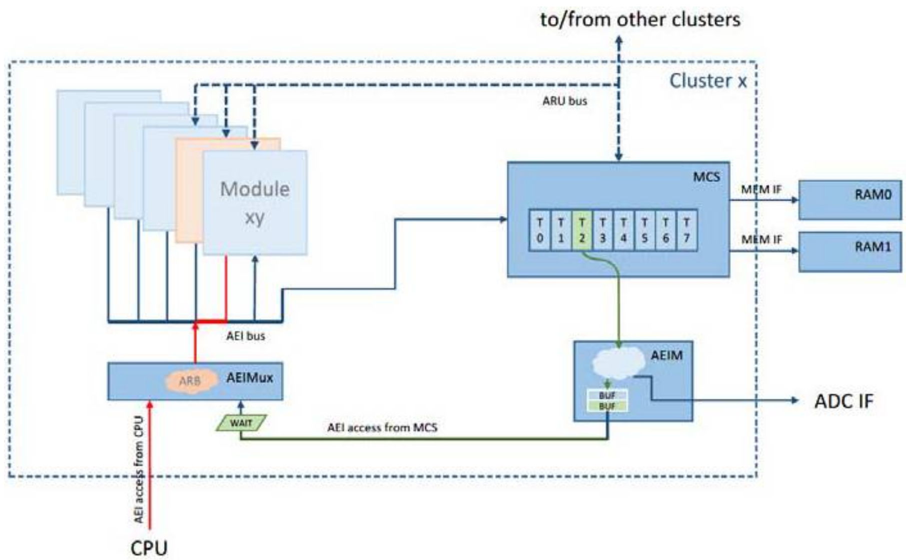

Figure 5 MCS AEI master access path

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.

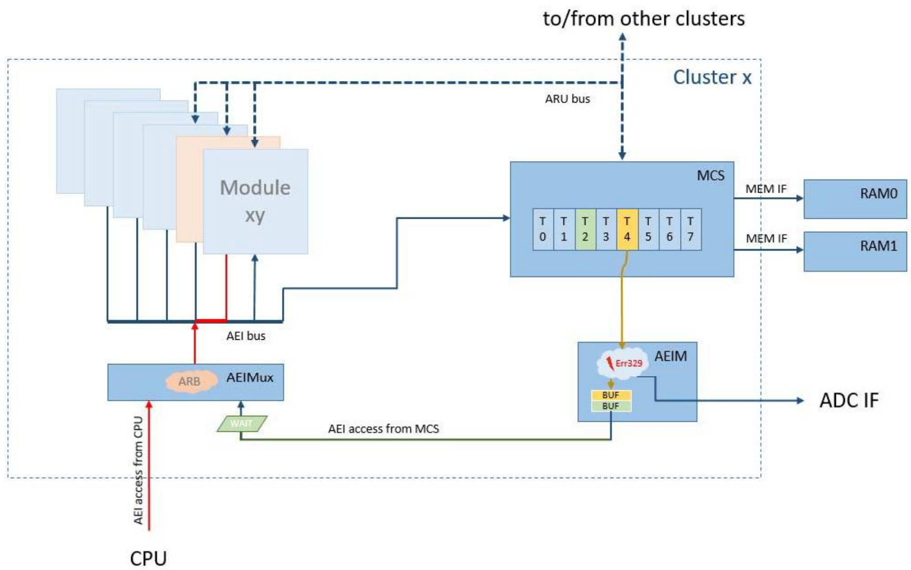

Figure 6 Critical MCS AEI master scenario

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).

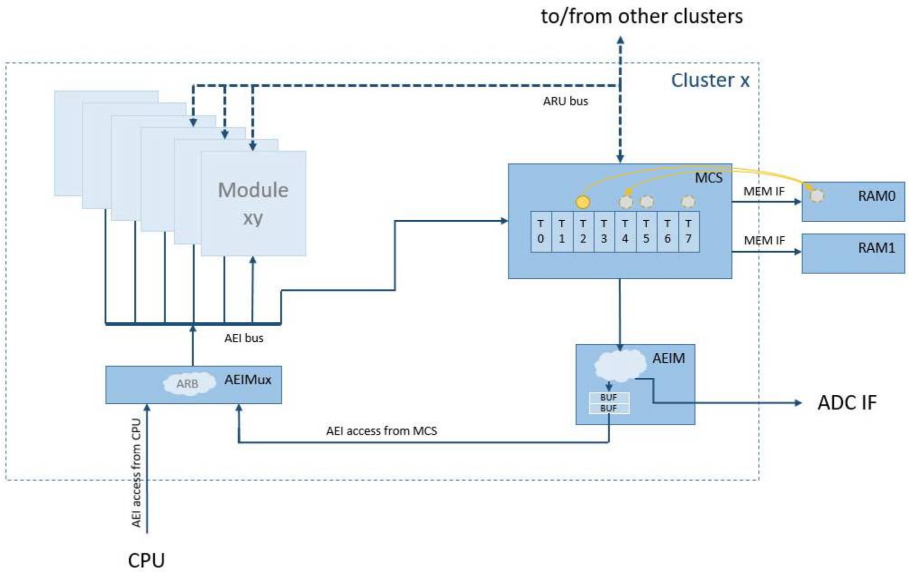

Figure 7 MCS token mechanism

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 8 **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

Do not use clock dividing for the affected (A)TOM channels, so the undivided cluster clock is used. For this configure the control registers in the CMU and CCM to generate non-dividing CMU_FXCLK and/or CMU_CLK signals. Select within the (A)TOM the non-dividing CMU_FXCLK0 (for TOM) and/or CMU_CLK0..7 (for ATOM) via the settings for CLK_SRC in the (A)TOM[i]_CH[x]_CTRL register(s).

Workaround 2

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

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.

<u>GTM_AI.352</u> 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. Alternatively use ARU dynamic routing, or reduce the value of ARU_CADDR_END to a value, which fits the PWM period. So, PWM period greater than ARU_CADDR_END + 1 + 3 ARU clock cycles.

GTM_AI.354 MCS: 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 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_AI.357 MCS: instructions XCHB, SETB, and CLRB do not suppress register write

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

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

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

Scope

MCS

Effects

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

Workaround

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

GTM_AI.358 TOM/ATOM: Synchronous update of working register for RST_CCU0=1 and UDMODE=0b01 not correct

TOM/ATOM is configured in SOMP mode with ATOM[i]_CH[x]_CTRL.MODE="10" (only for ATOM) and up-down counter mode is enabled by setting of (A)TOM[i]_CH[x]_CTRL.UDMODE=0b01. With the additional configuration of (A)TOM[i]_CH[x]_CTRL.RST_CCU0=1, the counter direction from up to down is changed with the trigger signal from a preceding channel TRIGIN[x] or with the TIM_EXT_CAPTURE signal from TIM module.

Expected behaviour:

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

Observed behaviour:

Additionally to the update of the working registers (A)TOM[i]_CH[x]_CM0 and (A)TOM[i]_CH[x]_CM1 when the channel counter (A)TOM[i]_CH[x]_CN0 reaches zero, the update is executed with the selected trigger signal TRIGIN[x] or TIM_EXT_CAPTURE(x). This is not expected in this configuration with (A)TOM[i]_CH[x]_CTRL.UDMODE=0b01.

Scope

TOM, ATOM

Effects

The synchronous update of the working register (A)TOM[i]_CH[x]_CM0 and (A)TOM[i]_CH[x]_CM1 is done unintendedly with the selected trigger signal TRIGIN[x] or TIM_EXT_CAPTURE.

Workaround

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

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

<u>GTM_AI.359</u> TOM: Both edges on TOM_OUT_T at unexpected times for RST_CCU0=1 and UDMODE>0

TOM channel is configured in up-down counter mode by setting of TOM[i]_CH[x]_CTRL.UDMODE>0 and the channel is triggered by a preceding channel or by TIM_EXT_CAPTURE with configuration of TOM[i]_CH[x]_CTRL.RST_CCU0=1.

Expected behaviour:

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

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

Observed behaviour:

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

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

Scope

TOM

Effects

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

Workaround

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

GTM_AI.360 SPEC-(A)TOM: PCM mode (BITREV=1) is only available for UDMODE=0

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

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

Scope

TOM, ATOM

Effects

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

Workaround

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

GTM_AI.361 IRQ: Missing pulse in single-pulse interrupt mode on simultaneous interrupt and clear event

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

Expected behaviour:

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

Observed behaviour:

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

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

Scope

IRQ

Effects

Missing pulse on interrupt signal IRQ_line.

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

Workaround

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

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

GTM_AI.362 MCS: Using wrong WURM mask during execution of instruction WURMX or WURCX

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

Example

Assume that the sequence

```
MOVL R6, 0x2
...
MOVL R6, 0x1
WURMX R0, STRG
```

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

Scope

MCS

Effects

WURMX or WURCX instruction is sensitive to the wrong volatile bit to be observed (e.g. interrupt or trigger bit) for one cluster clock cycle.

Workaround

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

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

GTM_AI.364 ATOM: ARU read request does not start at expected time-point in UDMODE=1 and UDMODE=3

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

Expected behaviour:

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

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

Observed behaviour:

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

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

- UDMODE=3: New ARU read request after CN0 changes the count direction from up to down.

Scope

ATOM

Effects

The effect depends on the UDMODE configuration:

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

Workaround

Workaround for UDMODE=1:

The PWM period length in up-down counter mode has to be double the length as the ARU round trip cycle (plus 3 ARU clock cycles).

Workaround for UDMODE=3:

Use AEI interface for reloading new shadow register values instead of ARU.

GTM_AI.367 MCS: Instructions WURMX and WURCX implement invalid extended register set for argument A

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

Scope

MCS

Effects

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

Workaround

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

GTM_AI.370 TOM/ATOM: Unexpected reset of CN0 in up-down counter mode and CM0=2

TOM/ATOM is configured in SOMP mode with ATOM[i]_CH[x]_CTRL.MODE=0b10 (only for ATOM) and up-down counter mode is enabled by setting of (A)TOM[i]_CH[x]_CTRL.UDMODE != 0b00.

Expected behaviour:

In this case, the counter CN0 changes its count direction from up to down either until CN0 reaches CM0-1 for RST_CCU0=0 or with the selected trigger signal TRIGIN (EXT_TRIG=0) or EXT_TRIGIN (EXT_TRIG=1) for RST_CCU0=1.

Observed behaviour:

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

- 1. In case of RST_CCU0=0:
  - The period value inside CM0 is configured to 2 and then reconfigured to a value greater than 2. After the counter CN0 starts incrementing and reaches value 1, CN0 is once reset to 0 unexpectedly, before it starts incrementing again.
- 2. In case of RST_CCU0=1 and EXT_TRIG=0:
  - The TRIGIN signal from a preceding channel is used to reset the count direction of CN0.

  - After the period value CM0 of the preceding channel is reconfigured from value 2 to a greater value, CN0 of this channel, which is triggered by the preceding channel, is once reset to 0 similar to the first scenario, which happens in the preceding channel.
- 3. In case of RST_CCU0=1 and EXT_TRIG=1:
  - The EXT_TRIGIN signal from TIM module is used to reset the count direction of CN0.
  - If the EXT_TRIGIN signal occurs while the counter CN0 is incrementing and reaches the value 1, CN0 is once reset unexpectedly. However, there is already no deterministic dependency between the EXT_TRIGIN signal and the reset of CN0.

Scope

TOM, ATOM

Effects

Unexpected reset of the counter CN0.

Workaround

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

- For scenario 1 and 2:
  - Do not use value 2 for the period, which is configured inside CM0.
- For scenario 3:
  - Do not use EXT_TRIGIN as trigger signal to change the count direction in up-down counter mode.

GTM_AI.371 MCS: Instruction MWRIL applies unexpected address offset calculation

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

Scope

MCS

Effects

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

Workaround

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

GTM_AI.374 SPEC-ATOM: Statement on timing of duty cycle output level change not correct for SOMP up/down-counter mode

The duty cycle output level is determined by ATOM[i]_CH[x]_CTRL.SL bit. The specification describes in section 15.3.3 “ATOM Signal output mode PWM (SOMP)” of the GTM chapter in the AURIX™ TC3xx User’s Manual, that “the duty cycle output level can be changed during runtime by writing the new duty cycle level into SL bit of the channels configuration register” (section 15.3.3.4). Further, it is mentioned: “the new signal level becomes active for the next trigger CCU_TRIGx (since bit SL is written)”.

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

Scope

ATOM

Effects

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

Workaround

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

GTM_AI.375 ATOM: Data from ARU are read only once in SOMC mode even though ARU blocking mode is disabled while FREEZE=1 and EN-DIS=0

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

Configuration register setting:

ATOM[i]_CH[x]_CTRL.MODE==0b01 (SOMC mode)

ATOM[i]_CH[x]_CTRL.ARU_EN==0b1 (ARU input stream enabled)

ATOM[i]_CH[x]_CTRL.ABM==0b0 (ARU blocking mode disabled)

Expected behaviour:

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

Observed behaviour:

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

Scope

ATOM

Effects

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

Workaround

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

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

GTM_AI.376 TOM/ATOM: Interrupt trigger signals CCU0TC_IRQ and CCU1TC_IRQ are delayed by one CMU_CLK period related to the output signals

Interrupt trigger signals CCU0TC_IRQ and CCU1TC_IRQ are delayed by one CMU_CLK period if the following configurations are used:

1. Both CCU0TC_IRQ and CCU1TC_IRQ are affected (ATOM: in SOMP mode) when the channel is configured in up-down counter mode ((A)TOM[i]_CH[x]_CTRL.UDMODE>0)
2. CCU1TC_IRQ only is affected (ATOM: in SOMP mode) when the channel is configured in up-counter mode ((A)TOM[i]_CH[x]_CTRL.UDMODE==0) and (A)TOM[i]_CH[x]_CTRL.SR0_TRIG is enabled

Scope

ATOM, TOM

Effects

Interrupt signals CCU0TC_IRQ and CCU1TC_IRQ are raised with a delay of one CMU_CLK period.

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

Workaround

No workaround available.

GTM_AI.387 DPLL: Wrong calculation of pulse generator frequency for DPLL_CTRL_0.AMT/S=1 and DPLL_CTRL_11.ADT/S=1 when number of pulses (DPLL_CTRL_0.MLT or DPLL_MLS1/2.MLS1/2) is too small

When the number of pulses per increment DPLL_CTRL_0.MLT is smaller than 127, or DPLL_MLS1/2.MLS1/2 is smaller than 128 and the correction of physical deviations is used (DPLL_CTRL_0.AMT/AMS=1 and DPLL_CTRL_11.ADT/ADS=1), the calculation of internal values such as DPLL_DT_T/S_ACT.DT_T/S_ACT, DPLL_RDT_T/S_ACT.RDT_T/S_ACT, and DPLL_ADD_IN_CAL1/2.ADD_IN_CAL_1/2 is wrong. The resulting frequency of the generated sub increment pulses of the DPLL is too small.

Scope

DPLL

Effects

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

Workaround

1. Do not use pulse numbers DPLL_CTRL_0.MLT < 127 and/or DPLL_MLS1/2.MLS1/2 < 128, when using correction of physical deviation (DPLL_CTRL_11.ADT/ADS=1 when DPLL_CTRL_0.AMT/AMS=1)
2. When workaround 1. cannot be applied use configuration DPLL_CTRL_11.ADT/ADS=0 when DPLL_CTRL_0.AMT/AMS=1 is used

GTM_AI.398 DPLL: Incorrect DPLL_THVAL calculation leading to a false direction decision in case tbu_ts0 wraps around

When

- a) the inactive edge of TRIGGER input signal is used for detection of the direction (DPLL_CTRL_1.IDDS=1)

and

- b) the input delay information is used to correct time stamps (DPLL_CTRL_0.IDT=1)

and

- c) in between the active input signal edge and the inactive input signal edge on TRIGGER tbu_ts0 wraps around,

then the calculation of DPLL_THVAL.THVAL is incorrect incurring a false direction decision.

Scope

DPLL

Effects

Wrong value of DPLL_THVAL and false direction decision.

Workaround

Don't use DPLL_CTRL_0.IDT=1 when evaluating direction with DPLL_CTRL_1.IDDS=1.

GTM_AI.400 MCS-RTL: Division instruction may produce unexpected memory overflow and wrong results

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

Scope

MCS

Effects

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

Workaround

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

GTM_AI.404 MCS-RTL: Division instruction reports unrelated ECC error

Description

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

Scope

MCS

Effects

Unexpected signaling of an ECC error.

Workaround

None.

GTM_AI.406 (A)TOM: FREEZE mode has no effect on (A)TOM_OUT_T in up-down counter mode with RST_CCU0=1

Description

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

Configuration for TOM:

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

TOM[i]_CH[x]_CTRL.RST_CCU0=1

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

TOM[i]_TGC[g]_ENDIS_STAT.ENDIS_STAT=0

Configuration for ATOM:

ATOM[i]_CH[x]_CTRL.MODE=0b10 (SOMP mode)

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

ATOM[i]_CH[x]_CTRL.RST_CCU0=1

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

ATOM[i]_AGC_ENDIS_STAT.ENDIS_STAT=0

Expected behavior:

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

Observed behavior:

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

Scope

TOM, ATOM

Effects

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

Workaround

No workaround available.

GTM_AI.408 (A)TOM-RTL: Missing edge on output signal (A)TOM_OUT when CN0 is reset with force update event

Description

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

Configuration for TOM:

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

TOM[i]_TGC[g]_FUPD_CTRL.FUPD_CTRL[k]=1

TOM[i]_TGC[g]_FUPD_CTRL.RSTCN0_CH[k]=1

Configuration for ATOM:

ATOM[i]_CH[x]_CTRL.MODE=0b10 (SOMP mode)

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

ATOM[i]_AGC_FUPD_CTRL.FUPD_CTRL[k]=1

ATOM[i]_AGC_FUPD_CTRL.RSTCN0_CH[k]=1

Expected behavior:

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

Observed behavior:

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

A second observation is in case the SL value (ATOM[i]_CH[x]_CTRL_SOMP.SL, TOM[i]_CH[x]_CTRL.SL) changes synchronously together with the force update event, an edge on (A)TOM_OUT to the inverted SL value (ATOM[i]_CH[x]_CTRL_SOMP.SL, TOM[i]_CH[x]_CTRL.SL) when (A)TOM[i]_CH[x]_CN0.CN0 reaches (A)TOM[i]_CH[x]_CM1.CM1 does not happen.

Scope

TOM, ATOM

Effects

Missing edge and false output signal level on (A)TOM_OUT

Workaround

No workaround available.

GTM_AI.409 DPLL: Flags of registers DPLL_STA_FLAG are not set

Description

The flags of register DPLL_STA_FLAG (STA_FLAG_T, STA_FLAG_S, INC_CNT1_FLAG, INC_CNT2_FLAG) are not set when one of these flags is cleared by a write operation of MCS register DSTAX (STA_FLAG_T, STA_FLAG_S, INC_CNT1_FLAG, INC_CNT2_FLAG) within the same cluster0 clock cycle.

Scope

DPLL, MCS

Effects

The event represented by the flags will be missed and not set in the registers. This affects the DPLL_STA_FLAG register as well as the MCS register DSTAX with the bit fields: STA_FLAG_T, STA_FLAG_S, INC_CNT1_FLAG, INC_CNT2_FLAG.

Workaround

If DPLL_CTRL_0.RMO = 0, only use DPLL_STA_FLAG.STA_FLAG_T or DPLL_STA_FLAG.INC_CNT1_FLAG.

If DPLL_CTRL_0.RMO = 1 and DPLL_CTRL_SMC = 0, only use DPLL_STA_FLAG.STA_FLAG_S or DPLL_STA_FLAG.INC_CNT1_FLAG.

If DPLL_CTRL_0.RMO = 1 and DPLL_CTRL_SMC = 1, only use one of DPLL_STA_FLAG.STA_FLAG_S, DPLL_STA_FLAG.INC_CNT2_FLAG, DPLL_STA_FLAG.STA_FLAG_T, or DPLL_STA_FLAG.INC_CNT1_FLAG.

GTM_AI.411 A change of the BRIDGE_MODE register might be delayed indefinitely

Description

After a write access to the BRIDGE_MODE register, the bit fields BRG_MODE and BYPASS_SYNC will not be updated until the transaction buffer is empty. In split mode the bridge allows new transactions to be added to the buffer, even when an update of these bits is pending.

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

Note: bit field BYPASS_SYNC is not specified for TC2xx.

Scope

GTM_AEI

Effects

Frequently polling the BRIDGE_MODE register ends in a deadlock.

Workaround 1

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

Workaround 2

Use standard mode (which is entered by setting AEI_PIPE and AEI_SPLIT at zero while asserting AEI_SEL) to write and read back the affected bits.

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

<u>GTM_AI.419</u> TIM: Potentially wrong capture values

Description

Configuration

The TIM channel is configured in TIEM, TIPM, TGPS or TSSM mode by setting of TIM[i]_CH[x]_CTRL.TIM_MODE={010_B, 011_B, 101_B, 110_B}. The TIM channel is disabled (TIM[i]_CH[x]_CTRL.TIM_EN = 0) and later enabled again (TIM[i]_CH[x]_CTRL.TIM_EN = 1).

Expected behavior for TIEM/TIPM/TGPS mode

The registers TIM[i]_CH[x]_CNT, TIM[i]_CH[x]_ECNT.ECNT[15:1], TIM[i]_CH[x]_GPR0 and TIM[i]_CH[x]_GPR1 are set to their reset values. In case of an input signal edge or an input capture event or an active selected CMU clock (TGPS mode) at the same time as the channel is enabled, this event has to be taken into account and the TIM[i]_CH[x]_CNT register must be updated/incremented based on its reset value. Due to this a capture event can happen depending on the configured TIM mode and the register values.

Expected behavior for TSSM mode

The registers TIM[i]_CH[x]_CNT, TIM[i]_CH[x]_ECNT.ECNT[15:1], TIM[i]_CH[x]_GPR0 and TIM[i]_CH[x]_GPR1 are set to their initial values. The initial value for TIM[i]_CH[x]_CNT register depends on TIM[i]_CH[x]_CTRL.ISL and TIM[i]_CH[x]_CNTS.CNTS(22). If TIM[i]_CH[x]_CNTS.CNTS(22) is set to 0 and TIM[i]_CH[x]_CTRL.ISL is set to 0 the initial value of TIM[i]_CH[x]_CNT is 0x000000. An input signal event simultaneously to the channel enable is not taken into account.

Observed behavior for TIEM/TIPM/TGPS mode

If no input signal event or input capture event or active selected CMU clock (TGPS mode) occurs, the registers TIM[i]_CH[x]_CNT, TIM[i]_CH[x]_ECNT.ECNT[15:1], TIM[i]_CH[x]_GPR0 and TIM[i]_CH[x]_GPR1 are set to their reset values as expected.

If an input signal event or an input capture event or an active selected CMU clock (TGPS mode) occurs at same time as the channel gets enabled, the TIM[i]_CH[x]_CNT register continues to count (or update) based on the previous (old) value. As a result, a capture could be performed too early and/or with the wrong values.

The TIM[i]_CH[x]_ECNT.ECNT[15:1] register is set to its reset value as expected.

Observed behavior for TSSM mode

The register TIM[i]_CH[x]_CNT is not set to its initial value of 0x000000 on channel enabling when TIM[i]_CH[x]_CNTS.CNTS(22) is set to 0 and TIM[i]_CH[x]_CTRL.ISL is set to 0.

Note: The TIM channel modes TPWM, TPIM and TBCM (TIM[i]_CH[x]_CTRL.TIM_MODE = {000_B, 001_B, 100_B}) are not affected.

Scope

TIM

Effects

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

Workaround 1

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

Workaround 2

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

1. Configure TIM[i]_CH[x]_CNTS = 0
2. Enable the TIM channel with the following configuration inside the TIM[i]_CH[x]_CTRL register:
   - TIM_EN = 1
   - TIM_MODE = 101_B (TGPS)
   - ISL = 1
   - OSM = 1
   - ARU_EN = 0
   - select a fast CMU_CLK_RES, for example CLK_SEL = 000_B
3. Wait until an edge on the selected CMU_CLK_RES occurs. This can be observed on the NEWVAL IRQ notify register. This event sets the TIM[i]_CH[x]_CNT register to its reset value.
4. Disable TIM channel (TIM[i]_CH[x]_CTRL.TIM_EN = 0)
5. Configure the former TIM channel configuration in TIM[i]_CH[x]_CTRL register and enable the TIM channel again.

GTM_AI.422 DPLL: Wrong DPLL_RDT_S_ACT/DPLL_RDT_T_ACT value in case of overflow correction

Description

The wrong overflow correction occurs for DPLL_RDT_S_ACT when the DPLL is in normal mode (DPLL_CTRL_0.RMO=0, DPLL_CTRL_1.SMC=0), or for DPLL_RDT_T_ACT when the DPLL is in emergency mode (DPLL_CTRL_0.RMO=1, DPLL_CTRL_1.SMC=0).

Instead of 0xFFFFFF the value 0x000000 is written in both cases. A problem in the calculation of pulse frequency (settling behavior) or PMT values may occur when the mode in DPLL_CTRL_0.RMO is switched (normal mode <-> emergency mode). If the overflow value was not yet overwritten (due to engine revolution happening before mode's switch) the wrong value might come into use for the described calculations.

Scope

DPLL

Effects

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

Workaround

Modification of DPLL_RDT_T_ACT (emergency mode) or DPLL_RDT_S_ACT (normal mode) after detection of overflow condition is not possible but does not cause any negative effects on pulse generation or PMT calculation at all.

The values stored to DPLL_RDT_T of RAM2 or DPLL_RDT_S of RAM1bc need to be corrected by following workaround sequence:

1. Check if relevant overflow on either DPLL_RDT_T_ACT or DPLL_RDT_S_ACT occurred.
   This can be done by observation of DPLL_STATUS.CRO when interrupt DPLL_IRQ_NOTIFY.EI occurred.
2. Check which of the interrupts DPLL_IRQ_NOTIFY.TASI/SASI has occurred next and based on that DPLL_RDT_T or DPLL_RDT_S has to be corrected.
3. For DPLL_CTRL_0.RMO = 0 and DPLL_CTRL_1.SMC = 0, DPLL_RDT_S[DPLL_APS.APS -1] has to be written with 0xFFFFFF.
   For DPLL_CTRL_0.RMO = 1 and DPLL_CTRL_1.SMC = 0, DPLL_RDT_T[DPLL_APT.APT -1] has to be written with 0xFFFFFF.

<u>GTM_AI.428</u> DPLL: Pulse correction is executed twice

Description

If DPLL_CTRL_1.PCM1/2 is set during DPLL_CTRL_1.DEN = 0 or DPLL_CTRL_1.DEN changes from 1 to 0, these values are immediately

transferred to the respective shadow registers
DPLL_CTRL_1_TRIGGER_SHADOW and
DPLL_CTRL_1_STATE_SHADOW.
Since DPLL_CTRL_1.DEN = 0, the DPLL_CTRL_1.PCM1/2 bits are not cleared when transferred to the shadow registers.

When DPLL_CTRL_1.DEN = 1 and the next relevant input signal on either STATE or TRIGGER arrives, the pulse corrections are executed due to the state of the named shadow registers.

As DPLL_CTRL_1.PCM1/2 were not cleared this is leading to an additional pulse correction for the next following input signal (TRIGGER / STATE).

Scope

DPLL

Effects

One additional pulse correction with DPLL_MPVAL1/2 pulses with impact on the angle base CCM[0]_TBU_TS1 or CCM[0]_TBU_TS2.

Workaround

Avoid setting DPLL_CTRL_1.PCM1/2 when DPLL_CTRL_1.DEN = 0.

<u>GTM_AI.429</u> TIM: Missing glitch detection interrupt event

Description

Configuration

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

Expected behavior

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

Observed behavior

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

Scope

TIM

Effects

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

Workaround

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

GTM_AI.430 TIM: Unexpected increment of filter counter

Description

Configuration

TIM filter is configured in immediate edge propagation mode by setting
TIM[i]_CH[x]_CTRL.FLT_MODE_RE = 0 and/or
TIM[i]_CH[x]_CTRL.FLT_MODE_FE = 0.

The filter is enabled by setting TIM[i]_CH[x]_CTRL.FLT_EN = 1.

The filter counter threshold is set to zero by setting
TIM[i]_CH[x]_FLT_RE.FTL_RE = 0 and/or
TIM[i]_CH[x]_FLT_FE.FTL_FE = 0.

Expected behavior

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

Observed behavior

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

Scope

TIM

Effects

If an input edge occurs during the acceptance time, the following output signal change will happen one or more selected CMU clock cycles earlier than expected. This depends on the initial configuration and the reconfiguration of the filter mode and the filter counter threshold. If the filter mode for both edges is configured to immediate edge propagation and both filter counter thresholds are set to zero, the counter falsely can count up to a higher value than one without resetting. If one or both filter modes and/or thresholds are reconfigured during the application, the higher count of the filter counter can lead to a difference of more than one CMU clock cycle between the expected and actual output signal change at the next occurring input edge. If only one filter counter threshold is set to zero, the difference of the expected and actual output signal change is one CMU clock cycle.

Workaround

If acceptable, use a threshold greater than zero. Otherwise there is no workaround available.

However, there is a possibility of minimizing the absolute error, deriving from this bug. If possible, a faster CMU clock can be selected. This leads to a shorter absolute time difference between the expected and actual output signal change. Additionally when applying this, the filter counter thresholds need to be assimilated proportionally in order to make the filter work as before.

GTM_AI.431 TIM: Glitch detection interrupt event of filter is not a single cycle pulse

Description

Configuration

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

Expected behavior

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

Observed behavior

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

Scope

TIM

Effects

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

Workaround

No workaround in hardware.

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

GTM_AI.441 DPLL: Missing pulse correction in case of DPLL_CTRL_1.SMC = 1

Description

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

Scope

DPLL

Effects

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

Workaround 1

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

Workaround 2

Use DPLL_CTRL_11.PCMF1 = 1 for pulse corrections. In this case no negative values for DPLL_MPVAL1 can be used

GTM_AI.450 DPLL: Stored time stamp values do not consider filter delays

Description

For the case where the filter delay values should be considered (DPLL_CTRL_0.IDT/IDS = 1) the data values of the time stamp fields in RAM1c (DPLL_TSF_S) and RAM2 (DPLL_TSF_T) actually do not take them into account for the input signals TRIGGER/STATE.

Scope

DPLL

Effects

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

Workaround

The entry of DPLL_TSF_T[p]/_S[p] can be read, corrected (by DPLL_FTV_T/_S), and written back.

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

GTM_AI.451 DPLL: Wrong measured position stamps in RAM

Description

For the synchronous motor control (DPLL_CTRL_1.SMC = 1) in normal mode (DPLL_CTRL_0.RMO = 0) wrong values are stored in RAM1b for DPLL_PSSM and DPLL_PSSM_OLD. The entries are not derived from CCM[0]_TBU_TS2 at the point of time when the active input signal arrives but they are derived erroneously from CCM[0]_TBU_TS1 instead.

Scope

DPLL

Effects

Wrong values for DPLL_PSSM and DPLL_PSSM_OLD stored in memory. Controlling of angle clock cannot rely on these values.

Workaround

Configure relevant TIM channels which are used to define the STATE input signal such that CCM[0]_TBU_TS2 is captured in each one of the TIM[0]_CH[x]_GPR1.GPR1 registers.

After a STATE input signal has arrived wait until the point in time when the DPLL should have calculated the DPLL_PSSM or DPLL_PSSM_OLD value. This is fulfilled when the content of the bit field DPLL_STA.STA_S has passed the value 0x28.

Then write DPLL_PSSM or DPLL_PSSM_OLD with the value of the TIM[0]_CH[x]_GPR1.GPR1 register of the corresponding TIM channel causing the captured input signal edge of STATE input. Which of the DPLL_PSSM or DPLL_PSSM_OLD values has to be written might be figured out by using the content of the bit field DPLL_OSW.SWON_S.

GTM_AI.454 (A)TOM: No output if trigger generation feature is used

Description

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

Scope

TOM, ATOM

Effects

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

Workaround

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

GTM_AI.456 DPLL: No action calculation

Description

If DPLL_CTRL_1.SMC=1 and DPLL_CTRL_0.RMO=0 no action calculation is done in STATE processing unit for action channels NOAC/2 to NOAC-1 (NOAC: number of action channels).

Note: Starting with GTM V4.1. NOAC=32, while in previous versions NOAC may be set to either 32 or 24.*

Scope

DPLL

Effects

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

Workaround

None.

GTM_AI.458 DPLL: Missing TOR interrupt and status flag

Description

If DPLL_CTRL_0.RMO=1 and DPLL_CTRL_1.SMC=0, the TOR interrupt (DPLL_IRQ_NOTIFY.TORI) is not triggered and the status flag (DPLL_STATUS.TOR) is not set on encountering an out of range TRIGGER.

Scope

DPLL

Effects

The TOR interrupt (DPLL_IRQ_NOTIFY.TORI) is not triggered and the status flag (DPLL_STATUS.TOR) is not set with the described configuration.

Workaround

No workaround available in hardware.

Nevertheless the application can detect the trigger out of range interrupt by observing TBU_TS0:

If the current TRIGGER time stamp
(DPLL_TS_T.TRIGGER_TS / DPLL_TS_T_OLD.TRIGGER_TS)
+ DPLL_DT_T.DT_T_ACT * DPLL_TLR.TLR > TBU_TS0 and no active TRIGGER input was encountered, the CPU/MCS can force a TOR interrupt by writing a one to DPLL_IRQ_FORCINT.TRG_TORI.

GTM_AI.462 (A)TOM: Missing CCU0TC_IRQ interrupt signal

Description

Configuration

The channel is configured in SOMP (ATOM) up-counter mode with up/down counter mode disabled ((A)TOM[i]_CH[x]_CTRL.UDMODE=0) or not existing and triggering by a preceding channel with configuration of (A)TOM[i]_CH[x]_CTRL.RST_CCU0=1.

Expected behavior

When the counter (A)TOM[i]_CH[x]_CN0.CN0 reaches the value of (A)TOM[i]_CH[x]_CM0.CM0, the interrupt signal CCU0TC_IRQ must be triggered.

Observed behavior

In the first period after (A)TOM[i]_CH[x]_CM0.CM0 is changed to the value 0 or 1, no CCU0TC_IRQ interrupt signal is triggered.

Note: When the second period starts after (A)TOM[i]_CH[x]_CM0.CM0 is changed to the value 0 or 1 and stays at that value, then the CCU0TC_IRQ interrupt signal generation works correctly.

Scope

TOM, ATOM

Effects

Interrupt signal CCU0TC_IRQ is not triggered.

Workaround

No workaround available.

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

GTM_AI.463 DPLL: DPLL_PVT not cleared after direction change

Description

For settings of DPLL_CTRL_1.SMC=1 or alternatively DPLL_CTRL_1.SMC=0 and DPLL_CTRL_1.IDDS=1 the direction change on TRIGGER channel is done via DPLL input port “TDIR” (generated via the control path SPE or TIM to MAP to DPLL).

If there is a direction change the RAM parameter DPLL_PVT is not cleared as specified.

Scope

DPLL

Effects

DPLL_PVT is not cleared and the PVT check is not suppressed under the described conditions. The PVT violation interrupt (DPLL_PWI_IRQ) could be unexpectedly triggered.

Workaround

Reset DPLL_PVT by CPU or MCS0 write operation, when direction change is detected via DPLL_IRQ_NOTIFY.DCGI interrupt.

GTM_AI.464 DPLL: Pulse correction executed twice when DPLL_CTRL_11.INCF1/2 is activated

Description

DPLL_MPVAL1/2.MPVAL1/2 is incremented (or decremented) twice in subsequent input signal events (only in the current increment and the following) when DPLL_CTRL_11.INCF1/2 is switching to 1 and a pulse correction is triggered via DPLL_CTRL_1.PCM1/2=1.

Scope

DPLL

Effects

Pulse correction is executed twice. The angle clocks CCM[0]_TBU_TS1/2 are misaligned. DPLL_PSTM, DPLL_PSSM, DPLL_PSTM_OLD, and DPLL_PSSM_OLD will have wrong values.

Workaround

Do not change DPLL_CTRL_11.INCF1/2 to 1 while pulse correction is ongoing.

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

GTM_AI.465 (A)TOM: Missing CCU0TC_IRQ interrupt signal for UDMODE > 0

Description

Configuration

The channel is configured in SOMP (ATOM) up/down counter mode with (A)TOM[i]_CH[x]_CTRL.UDMODE>0 and will be triggered by a preceding channel with configuration of (A)TOM[i]_CH[x]_CTRL.RST_CCU0=1.

Expected behavior

When the counter (A)TOM[i]_CH[x]_CN0.CN0 reaches in the up-counting phase the value of (A)TOM[i]_CH[x]_CM0.CM0, the interrupt signal CCU0TC_IRQ must be triggered.

Observed behavior

In the first period after (A)TOM[i]_CH[x]_CM0.CM0 is changed to the value 0, the CCU0TC_IRQ interrupt signal is triggered but not in the following periods with unchanged value of (A)TOM[i]_CH[x]_CM0.CM0=0.

A second observation is that the CCU0TC_IRQ interrupt signal is not triggered in the first period after the value of (A)TOM[i]_CH[x]_CM0.CM0 is changed from 0 to 1.

Note: in this case, the CCU0TC_IRQ interrupt is triggered in the following periods with unchanged value of 1 for (A)TOM[i]_CH[x]_CM0.CM0.

Scope

TOM, ATOM

Effects

Interrupt signal CCU0TC_IRQ is not triggered.

Workaround

No workaround available.

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

GTM_AI.466 TOM: Unexpected behavior of TOM_OUT_T for UDMODE>0

Description

Configuration

The channel is configured in up-down counter mode with TOM[i]_CH[x]_CTRL.UDMODE>0 and will be triggered by a preceding channel with configuration of TOM[i]_CH[x]_CTRL.RST_CCU0=1.

Expected behavior

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

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

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

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

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

Scope

TOM

Effects

Output signal TOM_OUT_T behaves not as expected.

Workaround

No workaround available.

GTM_AI.474 DPLL: DPLL_PSTC, DPLL_PSSC erroneously modified

Description

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

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

Scope

DPLL

Effects

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

Workaround

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

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

GTM_AI.475 DPLL: Incorrect values of DPLL_RCDT_TX, DPLL_RCDT_SX

Description

If during the reciprocal value calculation an overflow happens then the parameters DPLL_RCDT_TX.RCDT_TX and DPLL_RCDT_SX.RCDT_SX are set erroneously to 0x000000. The specified value is 0xFFFFFF.

Scope

DPLL

Effects

Wrong value is stored in either DPLL_RCDT_TX (normal mode) or DPLL_RCDT_SX (emergency mode) after a detection of a reciprocal overflow condition. The derived parameters DPLL_RCDT_TX_NOM.RCDT_TX_NOM and DPLL_RCDT_SX_NOM.RCDT_SX_NOM are diverging accordingly. This is leading to a different calculation of the pulse generator frequencies (DPLL_ADD_IN_CAL1.ADD_IN_CAL1 or DPLL_ADD_IN_CAL2.ADD_IN_CAL2 in dependence of the configured DPLL mode), which might lead to a different settling behavior of the generated angle clocks in such cases.

The diverging settling behavior is not necessarily malicious.

Workaround

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

GTM_AI.476 MCS: Unexpected instruction execution while disabling of MCS channel

Description

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

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

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

Scope

MCS

Effects

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

Workaround

Provide a disabling feature by MCS program, for example:

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

GTM_AI.477 DPLL: DPLL_DCGI interrupt not triggered

Description

When synchronous motor control mode is active (DPLL_CTRL_1.SMC=1):

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

Scope

DPLL

Effects

The interrupt DPLL_DCGI does not occur.

Workaround

When a direction change is detected by DPLL_IRQ_NOTIFY.DCGI the register DPLL_STATUS.BWD1 can be checked after the next relevant input signal edge on TRIGGER. If a second direction change is detected with the very next relevant input signal, the DPLL_DCGI can be set by writing DPLL_IRQ_FORCINT.DCGI =1. The next relevant input signal edge is the next input signal edge for DPLL_CTRL_1.SMC=1 (in contrast to the next inactive input signal edge when DPLL_CTRL_1.SMC=0).

GTM_AI.478 DPLL: Incorrect calculation of DPLL_THVAL, DPLL_THVAL2

Description

In case of LOW_RES=1, DPLL_CTRL_1.SMC=0, DPLL_CTRL_0.IDT=1, and DPLL_CTRL_0.TS0_HRT=0 the values of DPLL_THVAL, DPLL_THVAL2 are calculated incorrectly because the filter values are not divided by 8 as specified.

Scope

DPLL

Effects

Under the described conditions the values of DPLL_THVAL, DPLL_THVAL2 are incorrect. The divergence is small, but in theory this can still lead to a wrong direction decision as the THVAL is used for the evaluation of the direction change.

Example:

In case of a 45/90 µs sensor input signal for this failure to happen means to have a difference of the filter values between active and inactive input signal edge on TRIGGER larger than 450 clock cycles in case of a 20 MHz TBU_TS0 clock configuration.

Workaround

If a negative effect on the direction decision is not expected no workaround is necessary. If a negative effect cannot be excluded the use of the filter values can be switched off by setting DPLL_CTRL_0.IDT=0.

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.025 Register DPLL_IRQ_NOTIFY - Documentation update for bits SORI and DCGI

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

Documentation update

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

Table 9 Register DPLL_IRQ_NOTIFY - Documentation update for bits SORI and DCGI

Field	Bits	Type	Description
SORI	26	rw	**STATE out of range** ~~The interrupt occurs at line number 0.~~
DCGI	27	rw	**Direction change interrupt** The interrupt occurs at line number 0.

The rest of the description for register DPLL_IRQ_NOTIFY remains unchanged.

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

GTM_TC.026 Table "GTM IP Application Constraints" #1 (DPLL) - Documentation correction

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

Documentation correction

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

GTM_TC.028 Incorrect MCS behavior when SSH registers are accessed while MCS is running

Description

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

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

Scope

Access to GTM MCS SSH registers

Effects

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

Workaround

Use only setting GTMDIV = 1.

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.022 Message order inversion when transmitting from dedicated Tx Buffers configured with same Message ID

Configuration

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

Expected behaviour

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

Observed behaviour

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

Scope

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

Effects

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

Workaround

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

MCMCAN_AI.023 Incomplete description in section *.5.2 "Dedicated Tx Buffers" and *.5.4 "Tx Queue" of the M_CAN documentation in the User's Manual related to transmission from multiple buffers configured with the same Message ID

*Note: The absolute chapter number * depends on the version of the User's Manual.*

Section *.5.2 Dedicated Tx Buffers

Wording User's Manual

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

Enhancement - additional text

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

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

Section *.5.4 Tx Queue

Wording User's Manual - to be deleted

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

Replacement

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

Wording User's Manual - to be deleted

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

Replacement

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

Scope

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

Effects

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

Workaround

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

MCMCAN_AI.024 Frame transmitted despite confirmed transmit cancellation

Description

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

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

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

Scope

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

Effects

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

Workaround

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

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

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 on-going. This can lead to permanently corrupted MCDS trace messages.

Workaround

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

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 ≠ 000_B) - then just before entering standby, perform an additional dummy read to DLMU_STBY location 0x9000 0000 or 0xB000 0000 (when using CPU0 dLMU RAM) and 0x9001 0000 or 0xB001 0000 (when using CPU1 dLMU RAM). This dummy read triggers the alarm propagation and ensures that no alarms are lost due to standby entry.

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.

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).

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 to the AURIX™ TC3XX User’s Manual.

PADS_TC.013 Buffer type definition for P21.2: no ES functionality - Data Sheet documentation correction

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

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

Correction to Data Sheet

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

PMS_TC.005 Voltage rise at P33 and P34 up to V_{EVRSB} during start-up and up to $V_{LVDRSTSB}$ during power-down

The HWCFG pins (located in the V_{EXT} domain) information is evaluated when basic supply and clock infrastructure components are available as the supplies V_{EVRSB} and V_{EXT} ramp up. Tristate control information based on HWCFG[6] latched with V_{EXT} supply ramp can’t be used within the V_{EVRSB} supply domain until both supplies (V_{EXT} and V_{EVRSB}) have reached the minimum threshold value of $V_{LVDRST5}$ and $V_{LVDRSTSB}$, respectively.

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

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

Workaround

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

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

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 PU2 and PD2 pads always in tristate after standby entry - Documentation correction

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

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

Recommendation

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

- Standby entry while the VEXT supply ramps down,

- Standby entry with the VEXT supply available.

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

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

In the following sections and sentences in chapter PMS (*=11) and PMSLE (*=12), the term "PU1" shall be replaced by "**PU2 and PD2**":

- Section *.2.1.1 Supply Mode Selection:

  - „Regardless of the HWCFG[6] setting, the VEXT-buffered PU1 pads (see the PU1 buffer type in the data sheet) are set into tristate ..“
    shall be replaced by
  - “Regardless of the HWCFG[6] setting, the VEXT-buffered **PU2 and PD2** pads (see the **PU2 and PD2** buffer type in the data sheet) are set into tristate ..”.
- Section *.2.3.4.8 Entering Standby Mode (only VEVRSB domain supplied):
  - “Regardless of the PMSWCR5.TRISTREQ setting, the VEXT-buffered PU1 pads (see the PU1 buffer type in the data sheet) are set into tristate ..”
    shall be replaced by
  - “Regardless of the PMSWCR5.TRISTREQ setting, the VEXT-buffered **PU2 and PD2** pads (see the **PU2 and PD2** buffer type in the data sheet) are set into tristate ..”.
- Section *.2.3.4.9 Entering Standby Mode (both VEVRSB and VEXT domain supplied):
  - “Regardless of the PMSWCR5.TRISTREQ setting, the VEXT-buffered PU1 pads (see the PU1 buffer type in the data sheet) are set into tristate ..”
    shall be replaced by
  - “Regardless of the PMSWCR5.TRISTREQ setting, the VEXT-buffered **PU2 and PD2** pads (see the **PU2 and PD2** buffer type in the data sheet) are set into tristate ..”.
- Section *.2.3.4.10 State during Standby Mode:
  - “Regardless of the PMSWCR5.TRISTREQ setting, the VEXT-buffered PU1 pads (see the PU1 buffer type in the data sheet) are set into tristate ..”
    shall be replaced by
  - “Regardless of the PMSWCR5.TRISTREQ setting, the VEXT-buffered **PU2 and PD2** pads (see the **PU2 and PD2** buffer type in the data sheet) are set into tristate ..”.

See also the corresponding entries in the revision history for PMS chapter V2.2.31 and PMSLE chapter V1.0.4 at the end of each chapter.

PMS_TC.013 Minimum V_{DD} supply voltage for f_{SRI} > 200 MHz on TC375TI

Note: This limitation only applies to variant TC375TI of the TC37x device family, and only if used with f_{SRI} > 200 MHz.
All other TC37x devices may be used as specified in the Data Sheet.

Due to improvements in the RMS noise behavior (see section 2 for TC375TI in ADC_TC.P015), if the TC375TI uses f_{SRI} > 200 MHz, then the specified minimum V_{DD} supply voltage has to be limited to

- V_{DD} (Min.) ≥ 1.25 V - 8% = 1.15 V (instead of 1.25 V - 10% = 1.125 V)

Recommendation

For reliable operation with an application software, the undervoltage reset level of the primary voltage monitoring has to be configured accordingly (EVRRSTCON.RSTCTRIM=58_H).

It is recommended to configure the above mentioned RSTCTRIM value for the V_{DD} undervoltage reset level before the PLL configuration is started.

PMS_TC.014 Parasitic coupling on shared ADC pins depending on supply voltages

Bulk diodes exist from the V_{EXT} supply rail to the V_{DDM} supply rail through respective shared analog pins of EVADC Group 9 (P00.1 - P0.12).

If V_{EXT} > V_{DDM} and any of the shared pin voltages (V_{INPIN}) is higher than V_{DDM} by a diode voltage (V_{Diode} ~ 0.6V), i.e.

- V_{INPIN} > (V_{DDM} + V_{Diode}) OR V_{INPIN} pulled up to V_{EXT} by internal/external pull-ups > (V_{DDM} + V_{Diode})

then during start-up and operation, sink currents will flow from the pin to the V_{DDM} supply. The currents shall be limited by an internal/external pull-up resistor in order to stay within the overload conditions.

Behavior during start-up:

Only during the start-up phase, when the V_{DDM} supply voltage is less than the V_{DDPPA} (~1.3V) subthreshold limit, then the shared analog pins within an ADC multiplexer group of EVADC group G9 are internally connected together. The internal connection is high ohmic in nature (current < 100 µA). Consequently an external pull-up on one pin may be visible on the other pins in the same EVADC multiplexer group until the V_{DDM} supply is above the V_{DDPPA} limit and LVD reset limits on V_{EXT} and V_{EVRSB} have been reached.

Workaround

To avoid any current flow from V_{INPIN}/V_{EXT} to V_{DDM} and to prevent parasitic coupling on shared ADC pins:

- It needs to be ensured that the shared pin voltages (V_{INPIN}) are within the (V_{DDM} + V_{Diode}) supply range. Alternatively, V_{DDM} and V_{EXT} may be supplied together from the same supply source if the pull-ups on the pins are to the V_{EXT} rail.
- When both V_{EXT} and V_{EVRSB} are kept supplied during Standby mode, V_{DDM} should also be kept supplied if shared analog pins are pulled high.

Note: Related to this text module, in TC3xx User's Manual versions after V1.6, the row for V_{DDM} in table "5 V Nominal Supply: Voltage variations at independent supply rails during system modes" will be updated accordingly, and a diagram "Parasitic Diode Connectivity between supply rails" will be added.

<u>PMS_TC.015</u> EVRC synchronization – Documentation update for register EVRSDCTRL11 (PMS) and EVRSDCTRL2 (PMSLE)

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

SYNCMAXDEV in TC3xx User's Manual V2.0 (and earlier versions):

- d f_{MAXDEV} =
  100 MHz *(2*SYNCMAXDEV) / (SDFREQ^2+SYNCMAXDEV^2)

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

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

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

SYNCHYST in TC3xx User's Manual V2.0 (and earlier versions):

- SYNCHYST =
  round [d f_{HYST} * (SDFREQ ± SYNCMAXDEV)^2] / [d f_{HYST} * (SDFREQ ± SYNCMAXDEV) + 100 MHz]

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

- SYNCHYST =
  round [d f_{HYST} * (SDFREQ ± SYNCMAXDEV)^2 / (100 MHz ± d f_{HYST} * (SDFREQ ± SYNCMAXDEV))]
- First hysteresis band:
  - d f_{HYST} = 100 MHz / (SDFREQ + SYNCMAXDEV - SYNCHYST) - 100 MHz / (SDFREQ + SYNCMAXDEV)
- Second hysteresis band:
  - d f_{HYST} = 100 MHz / (SDFREQ - SYNCMAXDEV) - 100 MHz / (SDFREQ - SYNCMAXDEV + SYNCHYST)

PSI5_TC.005 Incorrect read pointer upon two consecutive RDFn read operations when two or more channels are configured

Description

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

Effects

Incorrect read pointer.

Scope

FIFO usage

Workaround

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

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.

SAFETY_TC.023 MCU infrastructure Safety Related Function - Documentation Update

Note: This issue applies to AURIX™ TC3xx Safety Manual version v2.0.

Section 4.3.1 (Introduction) of chapter “Safety Related Functions” in the AURIX™ TC3xx Safety Manual v2.0 mentions in the last bullet point below the table that Safety Related Functions 10, 11 and 12 shall always be correctly implemented in order to reach the ASIL level of the listed Safety Related Functions.

The listed absolute numbers 10, 11, 12 are not correct in this context.

Documentation Update

The MCU infrastructure Safety Related functions **12, 13 and 14** are assumed to be always correctly implemented.

SAFETY_TC.024 Clock alive monitor for f_{SPB} - Documentation update

The AURIX™ TC3xx Safety Manual states in section 6.37 SM[HW]:CLOCK:ALIVE_MONITOR that the clock alive monitor for f_{SPB} is only visible to HSM.

This statement is not correct.

Documentation update

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

SAFETY_TC.025 Wrong alarm listed in safety mechanism SM[HW]:SRI:SRI_TRANSACTION_INTEGRITY

In SM[HW]:SRI:SRI_TRANSACTION_INTEGRITY (section 6.477 in AURIX™ TC3xx Safety Manual v1.04 and higher versions), ALM11[12] "(CONVERTER) Phase Synchronizer Error" is listed as fault identification interface of the SM[HW]:SRI:SRI_TRANSACTION_INTEGRITY safety mechanism.

This statement is not correct.

Documentation update

ALM11[12] must not be considered as fault identification interface of the SM[HW]:SRI:SRI_TRANSACTION_INTEGRITY safety mechanism.

SAFETY_TC.026 Alarm for SM[HW]:IR:CFG_MONITOR - Documentation update

In SM[HW]:IR:CFG_MONITOR (section 6.268 in AURIX™ TC3xx Safety Manual v1.04 and higher versions) the alarm "ALM8[22] - EDC Configuration & Data Path Error" is listed as fault identification interface of SM[HW]:IR:CFG_MONITOR.

This statement is not correct.

Documentation update

Alarm "ALM10[22] - IR ACCEN Error Event" will be generated if a fault is detected by SM[HW]:IR:CFG_MONITOR.

SAFETY_TC.027 Single point fault detection for lockstep CPUs - Documentation update

The following safety mechanisms listed in chapter 6 (Safety Mechanisms) of the AURIX™ TC3xx Safety Manual

- SM[HW]:CPU:CRC
- SM[HW]:CPU:TPS
- SM[HW]:CPU:TPS_EXCEPTION_TIME_MONITOR

- SM[HW]:CPU:CODE_MPU
- SM[HW]:CPU:DATA_MPU
- SM[HW]:CPU:UM0
- SM[HW]:CPU:UM1
- SM[HW]:CPU:SV
- SM[HW]:CPU:STI

mention “Single Point Fault Detection = Yes” in column “lockstep CPU” of the table included in the description of the respective safety mechanism.

This statement is not correct.

The lockstep CPU is protected by SM[HW]:CPU:TRICORE_LOCKSTEP in any case and all the other SMs listed above are only used for freedom from interference.

Documentation update

For the safety mechanisms listed above, the corresponding entry in column “lockstep CPU” shall be corrected to

- “Single Point Fault Detection = N.A.”

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.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) by the following steps:
   a) set SCR_RSTCON.ECCRSTEN = 0_B disabling double bit ECC reset before an uncorrectable error is happening in XRAM. Bit ECCRSTEN may be set to 0_B either by explicitly writing to ECCRSTEN, or by triggering a local SCR reset which also clears the ECCRSTEN bit.
   Alternatively, if ECC reset generation is not disabled and an XRAM ECC error happens later, a periodic reset is triggered to MTU till the reset trigger is serviced. The permanent reset to MTU can be resolved by shortly enabling the SCR and disabling it again to service the pending ECC error triggered reset.
   b) set PMSWCR0.SCRWKEN = 0_B – wake-up via SCR disabled;
   c) set 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.

SCR_TC.021 RTC not counting after reset if P33.10 is high

The Real-Time Clock (RTC) in the SCR module may not reliably start counting if a high level was present on P33.10 (SCR_P01.2) during LVD reset. If enabled, the RTC will only start counting after the first high-to-low transition on P33.10 (SCR P01.2).

Note: Applications using an external (32 / 32.768 kHz) oscillator on P33.10 as clock source for the RTC are not affected.

Workaround 1

Ensure a low level on P33.10 (SCR_P01.2) during LVD reset, for example via a pull-down.

Workaround 2

Generate a high-to-low transition on P33.10 (SCR_P01.2) after LVD reset (by software or external hardware).

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.

As consequence the corresponding alarms ALM6[19], ALM6[20] and ALM6[21] in AG6 are not cleared by an application and system reset (if ECCD is not cleared by SW before triggering the reset).

Furthermore, flags in MC77_ECCD are not cleared upon warm PORST.

Workaround

Clear flags in register MC77_ECCD and MC78_ECCD via software by writing ‘0’ to the respective bits.

SCR_TC.023 External interrupts EXINT0, EXINT1 may get locked

As described in chapter “Interrupt System” of the SCR chapter in the TC3xx User’s Manual, if the external interrupt is positive (negative) edge triggered, the external source must hold the request pin low (high) for at least one CCLK cycle, and then hold it high (low) for at least one CCLK cycle to ensure that the transition is recognized.

However, for external interrupts EXINT0 and EXINT1, respectively, if the time between two triggering edges is shorter than 2 CCLK cycles, no further interrupt request is triggered after the first triggering edge. Further EXINT0 or EXINT1 interrupts are locked until the next application reset.

Note: This problem only occurs if interrupt generation on both rising and falling edge is selected, i.e. for EXINT0 if EXICON0.EXINT0 = 10_B, and for EXINT1 if EXICON0.EXINT1 = 10_B, respectively.

Workaround:

If using interrupt generation on both edges, ensure that the time between two triggering edges for EXINT0, EXINT1 is > 2 CCLK cycles. To include some margin for clock jitter and external signal slope asymmetries etc., the external source should hold the request pin low (high) e.g. for 2.1/f_{CCLK} to ensure that the transitions are correctly recognized.

Otherwise, only use external interrupts EXINT2..15.

SCR_TC.024 Field ADRES in register ADCOMP_RES - Documentation correction

In chapter "ADC Comparator Unit (ADCOMP)" of the SCR chapter in the TC3xx User's Manual, erroneously the term "ADCRES" is used instead of "ADRES" in the text and in figure "ADC Comparator Overview".

Documentation correction

The term "ADCRES" shall be replaced by "ADRES" within the text of chapter "ADC Comparator Unit (ADCOMP)", and in figure "ADC Comparator Overview".

In addition, the text in column "Description" for field ADRES in the ADCOMP Result Register ADCOMP_RES shall be corrected as follows:

Table 10 Field ADRES in register ADCOMP_RES - correction

Field	Bits	Type	Description
ADRES	7:0	rh	**ADC Conversion Result**
			This register shows the current converted ADC result. Software should ensure that the result is read before starting the next conversion. • For ADRES > 1: – VIN = [LSB * (ADRES-1)]; LSB = 23.077 mV. Full Range: 5861.54 mV. • For ADRES ≤ 1: software shall assume VIN as 0 V.

SCU_TC.031 Bits SCU_STSTAT.HWCFGx (x=1-5) could have an unexpected value in application if pins HWCFGx are left unconnected

An unexpected value for the HWCFGx pin state (x=1-5) may be latched in register field SCU_STSTAT.HWCFGx after application reset if the corresponding HWCFGx pin is not externally connected to a pull-up or a pull-down and the default reset state of port pins is set to tristate (pin P14.4/HWCFG[6] is pulled to GND).

EVRC start-up function after cold reset is not affected (HWCFG2).

EVR33 start-up function after cold reset is not affected (HWCFG1).

Only the intended function of HWCFG[3-5] pin configuration options in the corresponding reset cases is affected when BMI.PINDIS=0_B and DMU_HF_PROCONTP.BML=00_B (application boot defined by HWCFG[3-5] pins).

Workaround

Do not leave pins HWCFGx (x=1-5) unconnected if the default reset state of port pins is set to tristate (HWCFG[6] pulled to GND).

Note: This is not a general option for devices in QFP-80 and QFP-100 packages where P14.2/HWCFG2 is internally left unconnected.

If HWCFG2 is left unconnected, alternatively the application shall not rely on bit SCU_STSTAT.HWCFG[2] and may check for the correct state in the registers PMSWSTAT.HWCFGEVR or EVRSTAT.EVRC.

SCU_TC.033 $\overline{\text{TESTMODE}}$ pin shall be held at static level during LBIST

The MISR signatures documented in the product specific TC3xy Appendix to the TC3xx User’s Manual are only valid if the $\overline{\text{TESTMODE}}$ pin (P20.2) is always kept at a static **high** level during LBIST execution. This is the recommended LBIST configuration.

For a stable MISR signature, the level on this pin must not change during LBIST execution.

Workaround

For application environments where pin $\overline{\text{TESTMODE}}$ is not held high, but a static **low** level is applied to $\overline{\text{TESTMODE}}$, a different MISR signature will be received in the LBISTCTRL3.SIGNATURE field, depending on bit LBISTCTRL1.BODY.

Table 11 Contents of LBISTCTRL3 if $\overline{\text{TESTMODE}}$ is low during LBIST

Device	Design step	LIBISTCTRL3	
		BODY = 0	BODY = 1
TC39x	BA..BC	0x07EC4205	0xFEA614D6
	BD	0x935E836A	0x6A14D5B9
TC38x	AA..AD	0x1E1CD10C	0xA7587528
	AE	0x3AD1859B	0x839521BF
TC37xEXT	AA, AB	0x01E51F27	0xEDB9BD01
TC37x	AA	0x62DD6AB1	0xA373D89F
TC36x	AA	0xD833B421	0xF53338B3
TC35x	AB	0xBA38B3CB	0x1CACD3CE
TC33xEXT	AA	0xD1298927	0x1A824479
TC33x/TC32x	AA	0xC12B66CB	0x3F84CD94
TC3Ex	AA	0x4D0F493B	0xE7764C81

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).

SMU_TC.013 Unexpected setting of Alarm Missed Event bit xAEM in Alarm Executed Status register SMU_AEX

Note: This problem only applies to alarms of Alarm Type: Level (see tables “Alarm Mapping related to ALM group” in the product specific Appendix to the TC3xx User’s Manual).*

While servicing an alarm with alarm type Level, request status bit xSTS in the SMU_AEX register is set. However, the corresponding alarm missed event bit xAEM is also set, 1 cycle after the xSTS bit is set for the same alarm event (x can be any of IRQ0..2, RST0..5, NMI, EMS).

Workaround

While clearing the xSTS bit the corresponding xAEM bit should also be cleared for the alarm event.

If the xAEM bit is not cleared while clearing xSTS, only the alarm missed event xAEM functionality will not be available for later alarm events, and it does not impact any alarm action generation and xSTS bit functionality.

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.P015 Increased RMS Noise for TC374* and TC375* devices

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

1. For TC374* and TC375* except TC375TI

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

- For analog channel AN0 and AN1:
  - EN_{RMS} (Max.) ≤ 2.7 LSB_{12} in the temperature range T_A > -40°C.
- For analog channels AN2 to AN8:
  - EN_{RMS} (Max.) ≤ 2 LSB_{12} in the temperature range -40°C < T_A ≤ 0°C.
  - EN_{RMS} (Max.) ≤ 1.7 LSB_{12} for T_A > 0°C
- For all other analog channels ANx (x>8):
  - EN_{RMS} (Max.) ≤ 1.2 LSB_{12} in the temperature range T_A > -40°C.

2. For TC375TI

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

2.1 EVRC switching frequency 1.8 MHz (EVRSDCTRL0.SDFREQ=37_H)

- For analog channel AN0:
  - EN_{RMS} (Max.) ≤ 1.6 LSB_{12} in the temperature range -40°C < T_A ≤ 25°C.
  - EN_{RMS} (Max.) ≤ 1.4 LSB_{12} in the temperature range T_A > 25°C.
- For analog channel AN1:
  - EN_{RMS} (Max.) ≤ 1.4 LSB_{12} in the temperature range -40°C < T_A ≤ 25°C.
  - EN_{RMS} (Max.) ≤ 1.2 LSB_{12} in the temperature range T_A > 25°C.
- For analog channels AN2 and AN3:
  - EN_{RMS} (Max.) ≤ 1.2 LSB_{12} in the temperature range -40°C < T_A ≤ 25°C.
  - EN_{RMS} (Max.) ≤ 1 LSB_{12} in the temperature range T_A > 25°C.
- For all other analog channels ANx (x>3):
  - EN_{RMS} (Max.) ≤ 1 LSB_{12} in the temperature range T_A > -40°C.

2.2 EVRC switching frequency 0.8 MHz (EVRSDCTRL0.SDFREQ=$7D_H$)

- For analog channel AN0:
  - EN_{RMS} (Max.) ≤ 1.4 LSB_{12} in the temperature range -40°C < T_A ≤ 25°C.
  - EN_{RMS} (Max.) ≤ 1.2 LSB_{12} in the temperature range T_A > 25°C.
- For analog channel AN1:
  - EN_{RMS} (Max.) ≤ 1.2 LSB_{12} in the temperature range -40°C < T_A ≤ 25°C.
  - EN_{RMS} (Max.) ≤ 1 LSB_{12} in the temperature range T_A > 25°C.
- For all other analog channels ANx (x>1):

– EN_{RMS} (Max.) ≤ 1 LSB_{12} in the temperature range -40°C < T_A ≤ 25°C.
– EN_{RMS} (Max.) ≤ 0.8 LSB_{12} in the temperature range T_A > 25°C.

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} ≤ 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} ≤ 7.24 s (instead of 7 s as specified in the Data Sheet).

The increased values should be considered e.g. when defining erase timeout limits.

PADS_TC.P011 High Performance LVDS Pads - Documentation update to Data Sheet

In the current version of the Data Sheet, the note below table “LVDS - IEEE standard LVDS general purpose link (GPL)” referring to the termination resistor R_T shall be extended as follows:

Documentation Update

Note: R_T (or RT) in table ‘LVDS - IEEE standard LVDS general purpose Link (GPL)’ is a termination resistor of the receiver according to figure 3-5 in IEEE Std 1596.3-1996 and is represented in Figure 3-1[1)] either by R_{in} or by R_T=100 Ohm but not both. If R_T (or RT) is mentioned in column Note / Test Condition always the internal resistor R_{in} in Figure 3-1 is the selected one.

1) see corresponding TC3xx Data Sheet, figure “LVDS pad Input model”

PORST_TC.P002 V_{IH} and V_{IL} definition for PORST pad - Additional Data Sheet footnote

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

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

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.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 EVADC chapter of TC3xx User's Manual V1.2.0 and earlier versions, step "WAIT" (third step below) has been shifted **before** the begin of the start-up calibration.*

```
EVADC_GxANCFG = 0x00300000
  ;Analog clock frequency is 160 MHz / 4 = 40 MHz (example)
  ;CALSTC = 00
EVADC_GxARBCFG = 0x00000003 ;Enable analog block
WAIT   ;Pause for extended wakeup time (≥ 5 µs)
        ;(other operations not related to EVADC can be
executed in the meantime)
EVADC_GLOBCFG = 0x80000000  ;Begin start-up calibration
EVADC_GxARBPR = 0x01000000  ;Enable arbitration slot 0
EVADC_GxQMR0 = 0x00000001   ;Enable request source 0
EVADC_GxICLASS0 = 0x00000002
  ;Select 4 clocks for sampling time: 4 / 40 MHz = 100 ns
  ;The default setting stores results in GxRES0,
  ;service requests are issued on GxSR0
EVADC_GxRCR0 = 0x80000000
  ;Enable result service requests, if required
EVADC_GxQINR0 = 0x00000020
  ;Request channel 0 in auto-repeat mode
```

```
WAIT   ;Wait for start-up calibration to complete *)
        ;(other operations not related to EVADC can be
executed in the meantime)
       ;=> This starts continuous conversion of the channel
    *)time tSUCAL or flag GxARBCFG.CAL=0
```

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} = 0.5V (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} = 0.5V (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}).

1) in TC39x step AA: GxFCCTRL (x=12-19)

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 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.

ADC_TC.H039 DMA read access latency to result FIFO buffer

Using the result FIFO buffer, data consistency for a sequence of conversion results can be guaranteed. Initiated by a service request, the results of the conversion sequence can be read by the DMA, as soon as the last conversion result is available in the result FIFO buffer. This approach enables data integrity for application cases where the EVADC trigger scheme is asynchronous to the related SW task. To update the output of the result FIFO buffer, the latency is defined by 6 system peripheral bus clock cycles (f_{SPB}) and 6 ADC clock cycles (f_{ADC}).

To ensure a deterministic and interrupt-free transfer of the complete content of the FIFO buffer, single DMA transfer with the related number of data moves is the most efficient approach. For this purpose, the DMA source address is assigned to the output stage of the FIFO buffer and the destination address in the corresponding memory (DLMU, EMEM, …) has to increment or decrement linearly. In this mode, the initial data move needs 6 f_{SPB} cycles and 13 system resource interface clock cycles (f_{SRI}), and every subsequent data move needs 3 f_{SPB} clock cycles and 11 f_{SRI} clock cycles.

That means, this DMA configuration cannot be used to read the content of the FIFO buffer. Starting from the second data move, the corresponding latency ($3 \times f_{SPB}$ + $11 \times f_{SRI}$) is shorter than the time ($6 \times f_{SPB}$ + $6 \times f_{ADC}$) required to update the output stage of the FIFO buffer. As this waiting time is not met for consecutive reads, this leads to an inconsistent representation in the corresponding memory region (DLMU, EMEM, …) because the FIFO may get stuck.

Recommendation

To use the result FIFO buffer together with the DMA, a Linked List DMA configuration could be used. Using this kind of configuration, it is ensured that the DMA latency ($6 \times f_{SPB} + 13 \times f_{SRI}$) is longer than the time to update the result FIFO buffer ($6 \times f_{SPB} + 6 \times f_{ADC}$).

Note: See also ADC_TC.H037 for CPU read access.

ADC_TC.H043 Information on supervision signal $V_{ANACOMM}$ not relevant - Documentation update

The functionality of supervision signal $V_{ANACOMM}$ listed in table "Supervision Signals" of the EVADC chapter in the TC3xx User's Manual V2.0 (and earlier versions) can only be used during the Infineon production test. It cannot be used in a customer application.

Documentation update

Disregard the first line about $V_{ANACOMM}$ in table "Supervision Signals" in the EVADC chapter of the TC3xx User's Manual.

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).

ASCLIN_TC.H006 Sample point position when using three samples per bit - Documentation update

As documented in the description of field BITCON.SAMPLEPOINT, "... if three sample points at position 7, 8, 9 are required, this bit field would contain 9".

In general, if three samples per bit are selected (BITCON.SM = 1_B), field BITCON.SAMPLEPOINT defines the position of the last sample point.

Documentation update

The text related to three sample points in figure “ASCLIN Bit Structure” in the ASCLIN chapter of the User’s Manual should be updated as follows:

- 16x Oversampling, 3 sample points, relevant sample position 7, 8, 9 (BITCON.OVERSAMPLING = 16, BITCON.SM = 1, BITCON.SAMPLEPOINT = 9)
  – instead of “16x Oversampling, 3 sample points, relevant sample position 8”
- 8x Oversampling, 3 sample points, relevant sample position 3, 4, 5 (BITCON.OVERSAMPLING = 8, BITCON.SM = 1, BITCON.SAMPLEPOINT = 5)
  – instead of “8x Oversampling, 3 sample points, relevant sample position 4”

ASCLIN_TC.H007 Handling TxFIFO and RxFIFO interrupts in single move mode – Documentation update

Present description for TxFIFO single move mode

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

See also the note at the end of this section in the User’s Manual:

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

To complement the above description, the following two sentences shall be added to the section before the reference to figure “Interrupt generation in the single move mode” in the User’s Manual:

Documentation update for TxFIFO single move mode

If TxFIFO can handle new data, it generates an interrupt but expects just one data of the defined frame width. The DMA or the user should not write multiple data at once to avoid unexpected behavior.

Present description for RxFIFO single move mode

As described in section “Single Move Mode” of chapter “RxFIFO interrupt generation” in the User’s Manual, the purpose of the Single Move Mode is to keep the RxFIFO as empty as possible, by fetching the received elements one by one as soon as possible. The single move mode supports primarily a DMA operation using single move per RxFIFO interrupt.

See also the note at the end of this section in the User’s Manual:

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

To complement the above description, the following two sentences shall be added to the section before the reference to figure “RXFIFO - Interrupt Triggering in the Single Move Mode” in the User’s Manual:

Documentation update for RxFIFO single move mode

If RxFIFO can handle new data, it generates an interrupt but fetches just one data of the defined frame width. The DMA or the user should not read multiple data at once to avoid unexpected behavior.

ASCLIN_TC.H008 SPI master timing – Additional information to Data Sheet characteristics

The following note shall be added to chapter “ASCLIN SPI Master Timing” in the Data Sheet:

Note: The specified timings describe the pad capabilities for the respective driver strength configuration. For the maximum achievable baud rate in a *given application, the MRST input timings need to be considered in particular.*

Background information

Chapter “ASCLIN SPI Master Timing” in the Data Sheet contains separate tables for different output driver configurations. As can be seen from these tables, the master output timings directly depend on the selected driver strength. The corresponding parameters are marked as controller characteristics with symbol “CC”.

The setup and hold timings for input data received from the slave are marked as system requirements with symbol “SR”. They must be provided by the system in which the device is designed in.

In a given application, the maximum rate at which data can be received from a slave on the master receive input MRST may be limited by the required setup time t_{52} (MRST setup to ASCLKO latching edge). As data is shifted by the slave on one edge of ASCLKO and latched by the master on the opposite edge, one phase of ASCLKO must always be greater than the minimum required MRST setup time (assuming the sampling point is in the middle). This means the ASCLKO period t_{50} must be > 2 x t_{52}.

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)].

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.

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.

1) For TC39x A-step: chapter “Program Memory Unit (PMU)” of the Target Specification.

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 12 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

Table 12 General description of bits DISR, DISS, EDIS in register CLC

Field	Bits	Type	Description
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: Sleep Mode cannot be entered 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 13 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)

Table 13 Example 1a: Out of order memory access due to different access paths to semaphore and shared resource (cont'd)

CPUx	CPUy	Memory Access Sequence
		ld-2 (resource-read) "stale data"
		st-1 (resource-update)

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

Table 14 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.H021 Resource update failure despite correct SW synchronization upon retried FPI write transactions by CAN and ERAY modules

Note: Module names in the text follow the TC3xx syntax conventions.
Correlation of module names:
*- **TC3xx**: MCMCAN*
*- **TC2xx**: MultiCAN+*

Scope

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

Description

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

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

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

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

Table 15 Possible SW/HW interaction producing incorrect state

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

Recommended sequence (workaround)

A read operation to the shared resource must be performed by the first CPU (CPUx) before execution of the DSYNC instruction.

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

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

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

DSADC_TC.H010 Support for synchronous use of two or more DSADC channels

Note: This Application Hint refers to the DSADC module in AURIX™ TC2xx devices and to the EDSADC module in AURIX™ TC3xx devices.

The Global Run Control register GLOBRC controls the general operation of the available channels of the EDSADC module. For every EDSADC channel, register GLOBRC supports an individual bit for the related modulator (GLOBRC.MxRUN) and the related digital filter chain (GLOBRC.CHxRUN), where x depends on the number of implemented channels in the respective AURIX™ microcontroller device.

For applications where two or more EDSADC channels have to provide synchronous results, all related channels shall be enabled synchronously using a single write access to register GLOBRC. This approach guarantees synchronization between EDSADC channels under all loading conditions of the system peripheral bus (SPB).

Recommendation

To handle the EDSADC channel specific modulator settling time, the following sequence is proposed:

- Enable all modulators of the application specific synchronization group by a single write access to the corresponding MxRUN bits in the upper half-word of the Global Run Control Register:
  - GLOBRC = XXXX 0000_H, where $XXXX_H$ depends on the number of implemented modulators;
- Wait for modulator settling time t_{MSET} (see Data Sheet);
- Enable all modulators and corresponding digital filter chains of the application specific synchronization group by a single write access to the corresponding MxRUN and CHxRUN bits in the Global Run Control Register:
  - GLOBRC = XXXX $XXXX_H$, where $XXXX_H$ depends on the number of implemented modulators/demodulator channels.

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 (ALM9[1], ALM21[8]) or upper limit overflow (ALM9[0], ALM21[9]). Also, the corresponding flag DTSLIM.LLU or DTSLIM.UOF is set.

Recommendation

The application software shall clear the respective flag in DTSLIM and **afterwards** clear related SMU alarms. In case alarms are retriggered, application SW shall consider these as real alarms, and trigger a reaction within the FTTI (Fault Tolerant Time Interval) of the respective application.

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).

EDSADC_TC.H004 CIC3 filter properties - Documentation update

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

Table “CIC Filter Properties” shall be ignored.

EVR_TC.H001 External input capacitor value - Additional Data Sheet footnote

Description

The following footnote shall be added to parameter “External input capacitor value” (symbol C_{IN}) in table “EVRC SMPS External components” in the TC3yx Data Sheets:

Note: From EVRC view there is no defined hard limit for the maximum value of the input capacity, the specified upper limit is determined by the measurement setup. From EVRC view, the typical value of C_{IN} can be up to ~4x higher compared to the value listed in the Data Sheet.

FLASH_TC.H021 Flash Wait State configuration

Configuring flash wait states in your application is critical for correct operation.

Refer to these parts of the documentation of the respective TC3*x design step for guidance on avoiding data read errors over the lifetime of the device:

- Data Sheet, chapter “Flash Target Parameters”:

   – minimum access times t_{PF} / t_{PFECC} for PFLASH,
   – and t_{DF} / t_{DFECC} for DFLASH
- AURIX™ TC3xx User's Manual, NVM chapter "Configuring Flash Read Access Cycles" (6.5.2.1.2 in AURIXTC3XX_um_part1_v2.0.pdf)
- Application Note AP32381 (AURIX™ 2nd Generation startup and initialisation)

When **increasing** the SRI and FSI clock frequencies: first set registers HF_PWAIT and HF_DWAIT to the correct values, and then change the clock configuration.

When **decreasing** the SRI and FSI clock frequencies: first change the clock configuration, and then set registers HF_PWAIT and HF_DWAIT to the correct values.

Note: Applications that omit configuration of HF_PWAIT and HF_DWAIT may work in the development phase, but encounter data read errors in the field.

<u>FlexRay_AI.H004</u> 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 - Documentation Update

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.

Documentation Update

Note: In order to ensure proper FlexRay communication, RAM test mode must be explicitly disabled via TEST1.TMC = 00b at the end of the initialization sequence.

Therefore, Step16 in section "CLEAR_RAMS Command" of the TC3xx User's Manual must be updated from

- 16. Switch off Test Mode: TEST1.WRTEN = 0b

to

- 16. Switch off Test Mode: **TEST1.TMC = 00b and** TEST1.WRTEN = 0b

FlexRay_TC.H004 Bit WRECC in register TEST2 has no function

In the AURIX™ implementation of the E-Ray module, bit WRECC in register TEST2 has no function.

Recommendation

The value read from WRECC should not be evaluated by software, the value written (0_B or 1_B) to it is irrelevant.

For new software projects, keep bit WRECC at its reset value (0_B) for easier migration to future AURIX™ generations.

FlexRay_TC.H005 ERAY OTGB2 trigger set active even if disabled

Description

The trigger set TS32_SCSC of the ERAY IP-module is associated with OTGB2. An internal “valid” signal should be asserted only in case the trigger set is selected via OTSS.OTGB2.

Expected behavior

The OTGB2 trigger set valid signal should be gated by the bit field OTSS.OTGB2.

Actual behavior

The ERAY IP does not gate the valid signal with the OTSS.OTGB2 state, but only the data are gated. Meaning the OTGB2 trigger valid signal is only dependent on the slot counter and transfer buffer state changes, irrespective of the OTSS.OTGB2 value.

Recommendation

Ignore all OTGB2 ERAY triggers when data reported is only 0s.

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.H003 Undefined behavior when LD bit is set and buffer length B1L or B2L is zero - Additional information

Description

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

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

Additional information

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

Recommendation

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

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

GETH_TC.H002 Stopping and Starting Transmission - Additional information

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

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

Stopping and Starting Transmission

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

Notes

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

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

Steps

1. Disable the Transmit DMA (if applicable) by clearing Bit 0 (ST) of DMA_CH0 Register.
2. Wait for any previous frame transmissions to complete. You can check this by reading the appropriate bits of MTL_TxQ0_Debug Register (TRCSTS is not 01 and TXQSTS=0).
3. Disable the MAC transmitter and MAC receiver by clearing Bit 0 (RE) and Bit 1 (TE) of the MAC_Configuration Register.

a) The receiver will be stopped after the register got written (verified by reading back the register) in approximately 3*f_{GETH} + 6*RXCLK cycles if no future packet is in receive state (see description of bit RE).

4. Disable the Receive DMA (if applicable), after making sure that the data in the Rx FIFO is transferred to the system memory (by reading the appropriate bits of MTL_RxQ0_Debug Register, PRXQ=0 and RXQSTS=00).
   a) In case received data of a queue is not intended to be processed anymore set bit DMA_CHi_RX_CONTROL.RPF of each DMA channel moving out packets from this queue to flush all packets inside the queue after stopping the receive process.
5. Make sure that both Tx Queue and Rx Queue are empty (TXQSTS is 0 in MTL_TxQ0_Debug Register and RXQSTS is 0 in MTL_RxQ0_Debug Register).
   a) In case a late packet arrived, either forward to the system memory by re-enabling the RX DMAs (see step 6) or flush data out of the queue (see step 4.a)
6. To restart the operation, first start the DMAs, and then enable the MAC Transmitter and Receiver.

GPT12_TC.H002 Bits TxUD and TxUDE in incremental interface mode - Additional information

Description

The present description of the incremental interface mode for timers T2, T3, T4 in the User's Manual, including figures and tables, implicitly refers to the following configuration of bits TxUD and TxUDE (x = 2, 3, 4):

- TxUD = 0_B
- TxUDE = 1_B

This is the recommended and validated setting for these bits in incremental interface mode.

Additional information

When bit TxUD = 1_B, the count direction of timer Tx is inverted compared to the setting with TxUD = 0_B in incremental interface mode.

The setting of bit TxUDE is irrelevant in incremental interface mode, the behavior of Tx for TxUDE = 0_B and TxUDE = 1_B is identical. The figures related to incremental interface mode shall be interpreted as if TxUDE is permanently tied to 1_B.

GTM_AI.H425 MCS: Instructions BRDI and BWRI evaluate unused address bits

Description

Bus master instructions with indirect addressing (BRDI and BWRI) use the bits 2 to 15 of register B for defining their target address. However, if the bit slice B[1:0] is unequal to 0, the current implementation of the GTM-IP behaves as follows:

bit field CCM[i]_AEIM_STA.AEIM_XPT_STA is updated with value 1 and CCM[i]_AEIM_STA.AEIM_XPT_ADDR is updated with the address of the selected register B. Further, if bit field MCS[i]_CTRL_STAT.HLT_AEIM_ERR is set, then the associated MCS channel stops and indicates an unexpected bus master error.

Scope

MCS

Effects

The word alignment for the address of indirect bus master instructions is checked unexpectedly leading to a false error indication.

BRD and BRDI accesses of any MCS channel in the affected MCS instance that are following the non-wordaligned access might receive no data or wrong data. The received data for a BRD or BRDI instruction might also be stored in another register of the same MCS channel. Further, the MCS program execution of any channel might be stopped.

Workaround

The use of the lower address bits is forbidden for indirect addressing. Hence this error needs to be avoided by the application.

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.

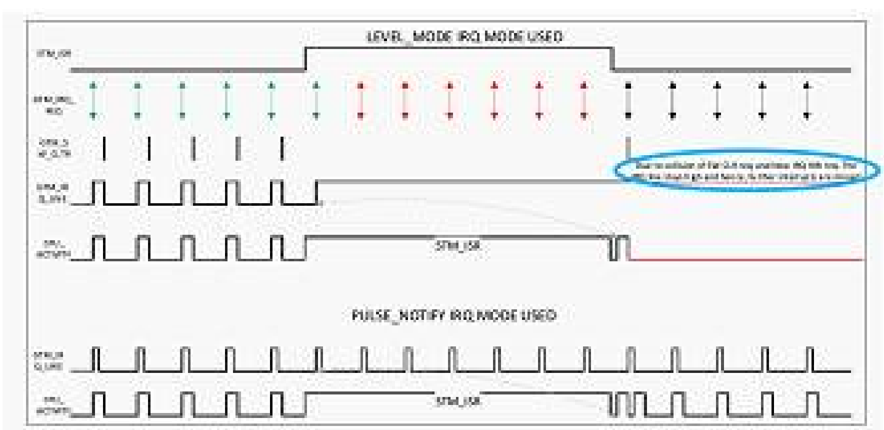

Figure 8 Interrupt Level vs. Pulse-Notify mode - Corner Case Example

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.H027 Register ODA (OCDS Debug Access) - Documentation update

In the GTM chapter of the TC3xx User's Manual, the following note is located below the description of register ODA in section "OCDS Debug Access Register":

- 6. TIM[i]_CH[x]_GPR0/1: Reading these register will reset the ECNT counter.

Documentation update

Note 6. (see above) does not apply and shall be deleted.

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.

HSCT_TC.H010 Interface control command timing on the LVDS ports

As described in section "Interface Control" of the HSCT chapter in the User's Manual, a HSCT master device is sending interface control commands to a slave device by setting the command in register IFCTRL.IFCVS and triggering IFCTRL.SIFCV.

Once triggered, the interface command is scheduled for take over into the transmission FIFO for sending. Only when the interface command has been taken over into the FIFO, sending of the next interface command must be triggered by software. Therefore, software must monitor the takeover by a transition of IRQ.IFCFS from 0 to 1.

As flag IRQ.IFCFS only indicates

- takeover of the interface command into the FIFO
- readiness for the next interface command to be triggered

the user might falsely assume that also the actual sending on the LVDS TX port has already occurred once the IRQ.IFCFS flag is set, which is not true.

Instead the timing shown in **Figure 9** applies.

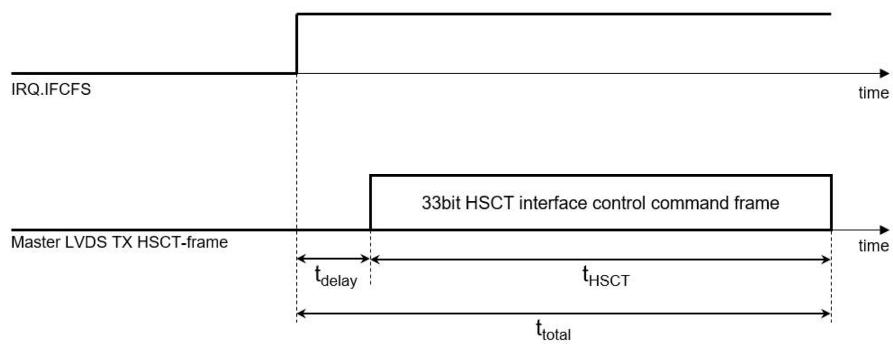

Figure 9 Timing of LVDS TX interface command sending in relation to IRQ.IFCFS

Recommendation

Before changing interface configurations, software must guarantee not having transfers active on the interface. Therefore the time of t_{total} has to be taken into account:

- $t_{total} = t_{delay} + t_{HSCT}$

While t_{HSCT} of the HSCT control command frame is determined and can be calculated from the actual baud rate, there is an additional time t_{delay} to be taken into account with

- $t_{delay} \geq 10$ µs

This value is valid for $f_{SRI} \geq 100$ MHz, $f_{SPB} \geq 100$ MHz and baud rate ≥ 5 MBaud.

<u>I2C_TC.H008</u> 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).

I2C_TC.H009 Connections of Serial Clock Inputs

In tables "Connections of I2C0" and "Connections of I2C1" (for TC39x, TC38x TC3Ex only) in the respective product specific appendix to the TC3xx User's Manual, only the serial clock output connections are shown. The serial clock input connections (located on the same port pins) are not shown.

Recommendation

See the corresponding TC3xy Data Sheet for the available I2C serial clock input connections (I2C0:SCLA/SCLB/SCLC, I2C1:SCLA/SCLB) on the respective port pins.

INT_TC.H006 Number of SRNs supporting external interrupt/service requests – Documentation update

SCU chapter "Output Gating Unit" of the TC3xx User's Manual (V1.6 and newer versions) contains the following statement:

- "Interrupt/service requests can be generated only by the ERU OGU[0-3] via its outputs ERU_IOUT[0-3]."

Actually, all ERU OGUs [0-7] can generate interrupts/service requests by outputs ERU_IOUT[0-7]. As only 4 interrupt nodes are available for ERU interrupts/service requests, multiple sources share interrupt nodes as shown in **Table 16** below.

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

- "Eight SRNs (Int_SCUSRC[7:0]) are reserved to handle external interrupts."

Documentation update

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

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

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

Table 16 OGU to SRC connection

OGUy.ERU_IOUTy (OGU I-output signal)	SRC_SCUERUx (interrupt SRC register)
OGU0.ERU_IOUT0, OGU4.ERU_IOUT4	SRC_SCUERU0
OGU1.ERU_IOUT1, OGU5.ERU_IOUT5	SRC_SCUERU1
OGU2.ERU_IOUT2, OGU6.ERU_IOUT6	SRC_SCUERU2
OGU3.ERU_IOUT3, OGU7.ERU_IOUT7	SRC_SCUERU3

In the SCU chapter, all references regarding the relation between ERU_IOUT*, ERU_INT* and SRC_SCUERU* shall be interpreted according to the table above:

The ERU_IOUTi and ERU_IOUTi+4 outputs are signaled through the ERU_INTi signals to the service request control registers SRC_SCUERUi in the interrupt router module (IR); i=0-3.

LBIST_TC.H003 Update reset behavior of LBISTCTRL0 and LBISTCTRL3 register - Additional information

Even though the LBISTCTRL3.[31:0] and LBISTCTRL0.28 register bits are cleared by a power-on reset they will automatically recover their values from stored contents of the central LBIST controller in the TCU (Test Control Unit) afterwards.

So on first software access the user will never see the initial reset values, but the updated LBIST done status and MISR result from the TCU LBIST controller.

The stored LBIST done status and MISR result in the central TCU LBIST controller will be cleared only through an externally applied warm power-on reset, during any cold power-on reset (triggered from EVR voltage monitors), or through the LBISTCTRL0.1 register bit (LBISTRES).

MBIST_TC.H001 Destructive MBIST requires DSPR0 initialization

Description

When performing a destructive MBIST, the DSPR0 content might be corrupted (meaning it contains ECC uncorrectable errors). For this reason, after the execution of that test, the user has to completely initialize DSPR0 with ECC correct data.

However, if a power interruption happens during the execution of a destructive MBIST and DSPR0 is configured to be not initialized during startup (see “RAM Configuration” register HF_PROCONRAM in TC3xx User´s Manual), the device may hang and it will not be able to execute user code; ESR0 pin remains permanently low. From this state, it can only be unblocked by a power-off/-on cycle.

Recommendation

Ensure when performing a destructive MBIST of DSPR0 that this memory is under all possible conditions initialized before being used by software. This can be achieved by applying the following measures:

- Take care that all error conditions cause either a cold or warm power-on reset. Configure DSPR0 initialization by SSW after cold and warm power-on-reset (see “RAM Configuration” register HF_PROCONRAM in TC3xx User´s Manual).
- After finishing the MBIST the software can perform the RAM initialization by itself or can trigger a cold or warm power-on reset to let SSW perform the needed initialization.

MCMCAN_AI.H001 Behavior of interrupt flags in CAN Interface (MCM-CAN)

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™ devices.
Register names in the text follow the TC3xx syntax conventions.
Correlation of register names:
*- **TC3xx**: CCCRi, PSRi, ECRi, IRi, IEi*
*- **TC4xx**: Ni_CCCR, Ni_PSR, Ni_ECR, Ni_IR, Ni_IE*

The M_CAN enters Busoff state according to CAN protocol conditions. The Busoff state is reported by setting PSRi.BO. Additionally, the M_CAN sets CCCRi.INIT to stop all CAN operation.

To restart CAN operation, the application software needs to clear CCCRi.INIT.

After CCCRi.INIT is cleared, the M_CAN's CAN state machine waits for the completion of the 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 MCMCAN chapter of the 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 ECRi.REC. Additionally, each occurrence of the Bus Idle condition is flagged by PSRi.LEC = 5 = Bit0Error, which triggers an interrupt (IRi.PEA) when IEi.PEAE is enabled.

While the Busoff Recovery proceeds, the CAN activity is reported as "Synchronizing", PSRi.ACT = 0 and PSRi.BO remains set. The time from resetting CCCRi.INIT to the clearing of PSRi.BO will be (in the absence of dominant bits on the CAN bus) 1420 (11 * 129 + 1) CAN bit times plus synchronization delay between clock domains.

The M_CAN does not receive messages while the 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, PSRi.ACT = 1.

When the Busoff Recovery has completed, PSRi.BO, ECRi.TEC, and ECRi.REC are cleared, and one CAN bit time later PSRi.ACT is set to Idle.

After PSRi.ACT reaches Idle, it will remain in Idle for at least one CAN bit time. The M_CAN's CAN state machine will become receiver (PSRi.ACT = 2) when it samples a dominant bit during Idle state or it will become transmitter (PSRi.ACT = 3) when it detects a pending transmission request during Idle state.

1) See Note in description of field LEC and footnote 1) in description of bit BO in Protocol Status Register i (PSRi).

MCMCAN_TC.H001 Behavior of undefined data bytes read from Receive Buffer

During CAN reception, the corresponding Receive Buffer in MCMCAN RAM is written only with number of bytes specified in DLC of the received frame, while the remaining bytes of the receive buffer retain the old/undefined values. Unlike MultiCAN+ in AURIX™ TC2xx devices, the additional bytes of the receive buffer are not overwritten with zeros.

Recommendation

When reading from the Receive Buffer of the MCMCAN RAM, the application software should ensure that only the number of bytes as specified in DLC of the received CAN frame are considered as valid data. The remaining bytes read should be ignored.

MCMCAN_TC.H006 Unintended Behavior of Receive Timeout Interrupt

Note: Register names in the text follow the TC3xx syntax conventions.
Correlation of register names:
*- **TC3xx**: NTRTRi (i=0-3), IEi (i=0-3)*
*- **TC4xx**: Ni_TIMER_RXTOUT (i=0-3), Ni_IE (i=0-3)*

On following conditions:

1. Receive timeout feature is enabled (i.e. NTRTRi.RELOAD != 0), **and**
2. Received CAN frames are stored in RxFIFO 0/1 or Dedicated Rx Buffers, **and**
3. Respective New CAN frame received interrupts are disabled (i.e. bits IEi.RF0NE, IEi.RF1NE or IEi.DRXE are 0),

then an unintended receive timeout interrupt (if enabled, i.e. NTRTRi.TEIE = 1) is triggered, although a valid CAN frame is newly received and stored in the respective RxFIFO 0/1 or Dedicated Rx Buffers.

Recommendation

Enable the corresponding receive interrupt via bits IEi.RF0NE, IEi.RF1NE, or IEi.DRXE, depending on the usage of RxFIFO0/1 or dedicated Rx Buffers for proper function of the receive timeout interrupt.

Example

If RxFIFO 0 is used, set IEi.RF0NE =1.

MCMCAN_TC.H007 Delayed time triggered transmission of frames

Note: Register names in the text follow the TC3xx syntax conventions.
Correlation of register names:
*- **TC3xx**: NTATTRi (i=0-3), NTBTTRi (i=0-3), NTCTTRi (i=0-3), NTCCRi (i=0-3)*
*- **TC4xx**: Ni_TIMER_TXTRIG0 (i=0-3), Ni_TIMER_TXTRIG1 (i=0-3), Ni_TIMER_TXTRIG2 (i=0-3), Ni_TIMER_CCR (i=0-3)*

The value written in the bit field RELOAD of register NTATTRi, NTBTTRi, NTCTTRi represents the reload counter value for the timer used for triggered transmission of message objects (Classical CAN or CAN FD frames).

The timer source and the prescaler value is defined in the NTCCRi register.

Once a value is written to bit field RELOAD with bit STRT = 1 the timer starts counting. This timer counts one value more than the written value in bit field RELOAD, then it triggers the transmission of a message object.

Effect

The message object transmission is delayed by one counter cycle with respect to the desired count time written in bit field RELOAD.

Recommendation

In order to transmit a message object at a specific time, when using one of these registers:

- NTATTRi, NTBTTRi, NTCTTRi,

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

MCMCAN_TC.H008 Parameter "CAN Frequency" - Documentation update to symbol in Data Sheet

As described in chapter "Clocking System" of the AURIX™ TC3xx User's Manual,

- f_{MCANH} defines the frequency for the internal clocking of the MCMCAN module,
- f_{MCAN} defines the basic frequency for the MCMCAN module used for the baud rate generation.

Documentation Update

For consistency with the description in the TC3xx User's Manual, the symbol for parameter "CAN frequency" in table "Operating Conditions" in the Data Sheet shall be changed from "f_{CAN}" to "$\mathbf{f_{MCAN}}$" as shown below:

Table 17 Operating Conditions - CAN Frequency: symbol update

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

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.

MSC_TC.H014 Symbol T_A in specification of FCLPx clock period in Data Sheet - Additional information

The tables in chapter "MSC Timing" in the Data Sheet use symbol "T_A" in the specification of parameter "FCLPx clock period".

In this context, $T_A = 1/f_A$, where f_A is the input clock frequency of the MSC ABRA block (see also the MSC chapter in the User's Manual).

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.

Note/Documentation correction

In the corresponding text at the beginning of section "SSH Safety Faults Status Register" in the MTU chapter of the TC3xx User's Manual (V1.2.0 and later versions), the term MCi_FAULTSTS.MISCERR[2] shall be replaced by MCi_FAULTSTS.**OPERR**[2].

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.

<u>OCDS_TC.H015</u> 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. a) CPUx is halted (either in boot-halt state or stopped by debugger tool or in idle mode) when reset is triggered, OR
   b) CPUx Performance Counters are enabled and CPUx Clock Cycle Count register CCNT is read.

Recommendation

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

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

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.

PACKAGE_TC.H004 Exposed pad dimensions and package outlines for QFP packages - Updates to TC37x Data Sheet

In the scope of the harmonization of the package drawings, the drawings for the LQFP-176 package of the TC37x have been updated. No change of form, fit or function is implied.

The dimensions for the exposed diepad are included in the figure.

Furthermore, for the exposed die pad, the maximum boundary of the structural corner protrusions to be considered during system design and integration has been added.

This information shall substitute the corresponding information in the TC37x AA-step Data Sheet V1.1.

Package Outlines LQFP-176 for TC37x

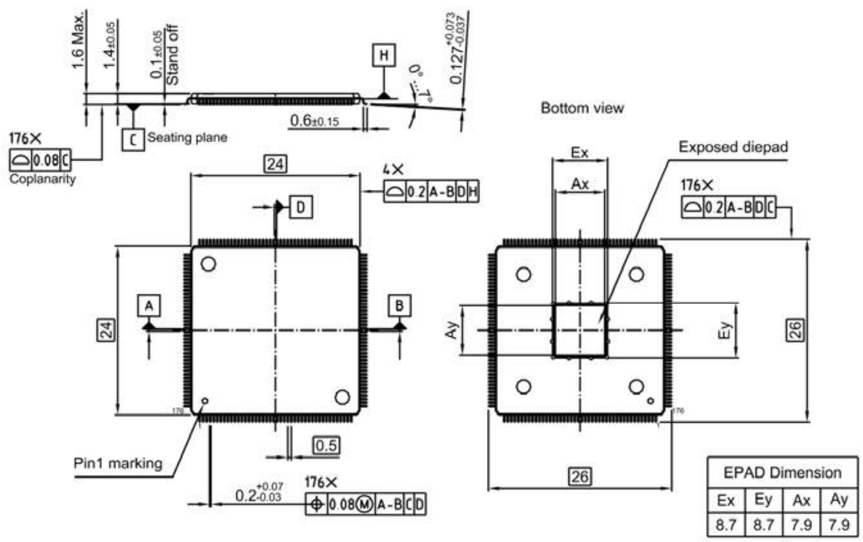

Figure 10 Package Outlines LQFP-176 for TC37x

Note: For the exposed diepad of the LQFP-176 package of the TC37x, structural corner protrusions have to be considered for purposes of system design and integration with a maximum boundary of 9.4 mm.

You can find all of our packages, sorts of packing and others on our Infineon Internet Page http:/www.infineon.com/products

PADS_TC.H008 Overload coupling for LVDS RX pads – Additional information

Due to the internal circuit structure, for pads with buffer type “LVDS_RX” (see column “buffer type” in port function tables in the Data Sheet), the specified overload coupling factors K_{OVDP} and K_{OVDN} to the neighbor LVDS pad are relatively large (see table “Overload Parameters” in the Data Sheet).

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

The following figure illustrates the effect for a positive overload condition (V_{PADP} > V_{EXT}) on the P-pad (PADP) of a pin pair where both the P-pad and the corresponding N-pad (NPAD, with 0V ≤ V_{NPAD} ≤ V_{EXT}) are used as digital CMOS inputs.

The overload current I_{OSW} through the non-ideal open switch significantly increases when V_{PADP} ≥ V_{EXT} + 200 mV.

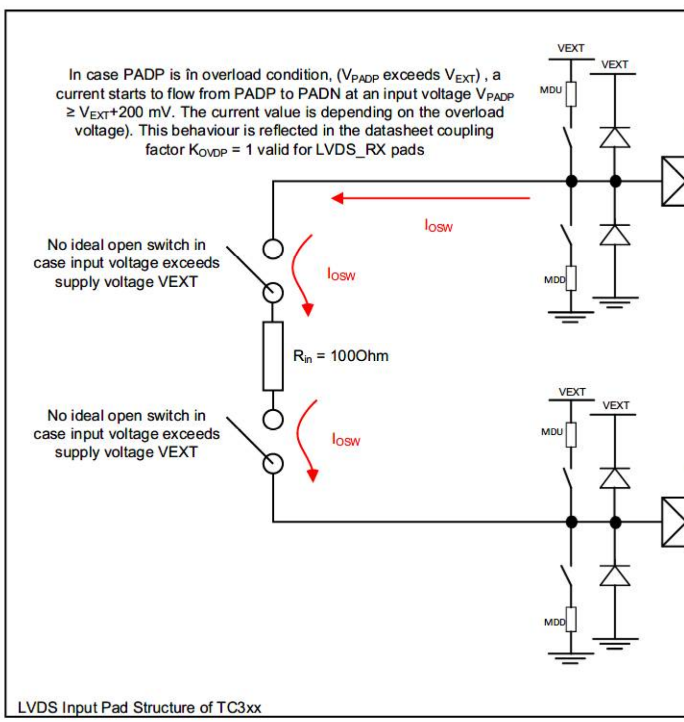

Figure 11 Positive overload on P-pad of a LVDS RX pin pair (both used as inputs)

Note: In general, the effect will also occur if the N-pad is used as output, or for a negative overload condition on either the P-pad or the N-pad.

PADS_TC.H009 Pull-ups active on P33/P34 pins in standby mode when SCR is disabled

Expected behavior

When the SCR is enabled (default after cold PORST), and the device is configured to have pull-ups enabled on all pins during runtime and tristate when in standby mode (setting HWCFG6=1, PMSWCR5.B.TRISTREQ=1), then P33/P34 pins are in tristate as configured and expected in standby mode.

Unexpected behavior

However, when the SCR is disabled, pull-ups are active on P33/P34 pins in standby mode in the configuration described above.

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.H008 Interaction of interrupt and power management system - Additional information

The description of steps to enter Idle, Sleep and Standby Mode in chapter "Power Management Overview" of the PMS and PMSLE chapters in the current TC3xx User's Manual is not comprehensive in explaining the dependency on pending interrupts as well as received interrupts. Hence, more explanation is provided here.

For a CPU to enter Idle Mode, it must have no interrupts pending. If it is in Idle Mode it will stay in Idle Mode until one of the specified wake-up events occurs – one of these is to have a pending interrupt.

Any SRN targeting a specific CPU (i.e. TOS set to that CPU), which is enabled, i.e. has SRE set, and has received a trigger event, i.e. has SRR set (whether by a received trigger from a peripheral or a master using the SETR control bit in the SRN) is a pending interrupt. Thus, even if a peripheral is shut down by having its clocks gated off, if it has presented a trigger event to the IR, and the SRE bit for that SRN is set, there will be a pending interrupt to the specified CPU.

It is not necessary for the priority of the pending interrupt to allow it to be taken, nor is it necessary for the CPU to have interrupt servicing enabled. It is possible and valid for Idle Mode to be entered with interrupts disabled, and to only re-enable interrupt acceptance subsequent to resuming execution. Equally, the CPU's priority may well dictate that the interrupt cannot be serviced immediately on re-enabling interrupts.

There may be some interrupts in a system that a CPU will be required to service and must exit Idle Mode (or Sleep Mode) or prevent entry to Idle Mode (or Sleep or Standby Mode) on their arrival. If one of these interrupts is raised prior to, or just as Idle Mode, Sleep Mode or Standby Mode is requested then that mode will not be entered.

The description for the REQSLP field states

- "In Idle Mode or Sleep Mode, these bits are cleared in response to an interrupt for the CPU, or when bit 15 of the corresponding CPU Watchdog Timer register (bit WDTCPUxSR.TIM[15]) changes from 0 to 1."

For clarity, this also means, if a write to PMCSRx.REQSLP occurs while the IR has a pending interrupt for CPUx the write data will be ignored and the REQSLP value will remain as 00_B "Run Mode".

For the system to enter Sleep or Standby Mode by writing to PMCSRx.REQSLP (as opposed through an external low voltage condition), all CPUs must be in Idle Mode. Typically, first other CPUs will be brought into Idle Mode and then the master CPU will be the last to enter to Idle Mode as a transitional state of the request for the system mode Sleep or Standby. Consequently any pending interrupts for any CPU will prevent the entry into Sleep or Standby Mode.

Recommendation

To ensure the transition to a power save mode, for a CPU intended to enter Idle Mode or for a system entering Sleep or Standby mode, all interrupts that are not intended to cause Run Mode to be re-entered or retained, should either have the SRE bit cleared in the respective SRN or be guaranteed to have the SRR bit clear.

If modifying the SRE bit of an SRN, to ensure the new state is reflected in IR arbitration information conveyed to the PMS and CPUs, sufficient time for an arbitration must have elapsed. Hence, a subset of the synchronisation described in subsection “Changing the SRN configuration” of the IR chapter in the TC3xx User’s Manual is required.

After the last SRN (for CPUx) has been updated

- Read back the last SRN
- Read the LWSRx register

Clearing the SRR bit or disabling the source of the trigger can also be used if there are no timing hazards; i.e. no risk of a trigger being raised just before reconfiguring the peripheral (to not raise triggers), or no risk of an SRN that has had SRR cleared being set again while other SRNs are accessed. If the timing behaviour of these interrupt sources allows them to be disabled at source or in the SRN these are also valid methods. So long as the SRE bit and SRR bit are not both set, there will not be a pending interrupt. If the SRR bits are cleared, after the last SRN is modified there also needs to be a synchronisation step for the IR outputs to reflect the update before the PMCSRx is written.

Once there are no pending interrupts, request the power saving mode by writing to the respective PMCSRx.

Note: There will still be several system clock cycles till the power saving mode is enabled by the PMS during which the CPU will continue to execute instructions.

To ensure a deterministic boundary for execution to end after the power saving mode request, the write to PMCSRx should be followed by a DSYNC and a WAIT instruction.

PMS_TC.H009 Interaction of warm reset and standby mode transitions

Chapter “Power Management” in the PMS and PMSLE chapters of the TC3xx User’s Manual in general describes how the standby mode transitions are performed from the AURIX™ system point of view (see also figure “Power down modes and transitions”).

This application hint addresses the specific use cases

- when a standby request by VEXT ramp down is issued during warm reset, or
- when a warm reset is triggered when a standby mode transition is ongoing.

The PMS and PMSLE modules have a separate state machine operating independently from the rest of the AURIX™ system. The PMS and PMSLE modules and states are not affected by warm resets (e.g. application reset). Table "Effect of Reset on Device Functions" in the SCU chapter of the TC3xx User's Manual shows how the AURIX™ modules are affected by different reset types. The PMS and PMSLE modules behave in the same way as the EVR module listed in this table.

Therefore standby mode entry is achieved even in the reset state of the AURIX™ system modes.

PMS_TC.H011 Supply mode and topology selection - Allowed combinations of VEXT and VDDM - Documentation update

Tables "Allowed Combinations of Nominal External Supply Voltages between Voltage Rails" in the PMS and PMSLE chapters of the TC3xx User's Manual define the allowed combinations of the external supply voltages for the target applications and verified use cases of the TC3xx family.

These tables do not include the combination VEXT = 5V and VDDM = 3.3V.

Documentation update

For consistency, in tables "Supply Mode and Topology selection" in the same chapters, for the configurations with "VEXT & VEVRSB = **5.0V** external supply" in column "Supply Pin Voltage/Level", the term "VDDM = VAREFx = 5V **or 3.3V**" shall be replaced by

- VDDM = VAREFx = 5V

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.

RESET_TC.H006 Certain registers may have different reset values than documented in TC3xx User's Manual - Documentation update

The following registers may show different reset values compared to those documented in the TC3xx User's Manual or TC3yx appendix. During device start-up, the initial hardware reset values of certain registers may be updated. Consequently, user software may read different values. Please refer to the table below for further details.

Note: The TC3xx User's Manual chapters and/or register bit-field descriptions may contain information in addition to reset values/tables.

Note: The registers listed in the following table apply to TC39x, TC38x, TC37x, TC37xEXT, TC36x, TC35x, TC33xEXT and TC3Ex. For TC33x/32x see separate table below.
*Presence of CPU*_PCON1 registers depends on number of available CPUs.*

Table 18 TC39x, TC38x, TC37x, TC37xEXT, TC36x, TC35x, TC33xEXT and TC3Ex registers that may have different reset values than documented in TC3xx User's Manual

Register	Initial reset value	Reset value defined in User's Manual	Remark
P20_IOCR0	0x0010 0000	0x0000 0000 (HWCFG6 = tri-state)	TESTMODE pin is PU (input pull-up), even with HWCFG6 = tri-state, as described in Data Sheet.
EMEM_TILECONFIG	0x0000 0000	0x5555 5555	This is a write-only ("w") register. For tile mode information do not read EMEM_TILECONFIG; instead, read EMEM_TILESTATE.
SBCU_DBCNTL	0x0000 7002	0x0000 7003	Bit EO is "Status of BCU Debug Support Enable" and only set after reset when OCDS is enabled. This bit is controlled by Cerberus.

Table 18 TC39x, TC38x, TC37x, TC37xEXT, TC36x, TC35x, TC33xEXT and TC3Ex registers that may have different reset values than documented in TC3xx User's Manual (cont'd)

Register	Initial reset value	Reset value defined in User's Manual	Remark
CPU1_PCON1	0x0000 0001	0x0000 0000	Bit PCINV of PCON1 is set when CPU is in boot halt mode, it is cleared when CPU starts execution
CPU2_PCON1	0x0000 0001	0x0000 0000	
CPU3_PCON1	0x0000 0001	0x0000 0000	
CPU4_PCON1	0x0000 0001	0x0000 0000	
CPU5_PCON1	0x0000 0001	0x0000 0000	
HSSL0_MFLAGS	0xA000 0000	0x8000 0000	Bit TEI indicates the state of CTS (Clear To Send) signal from HSCT module. The default state of this bit is 1.
HF_OPERATION	0x0000 0X00	0x0000 0000	RES bits shall be ignored.
PMS_EVRSDCTRL0	0x3039 0001	0xF039 0001	LCK and UP bits are cleared.
PMS_EVRSDCTRL1	0x0669 0708	0x8669 0708	LCK bit is cleared.
PMS_EVRSDCTRL6	0x0023 1C94	0x8023 1C94	LCK bit is cleared.
PMS_EVRSDCTRL7	0x0000 00FE	0x8000 00FE	LCK bit is cleared.
PMS_EVRSDCTRL8	0x1121 048E	0x9121 048E	LCK bit is cleared.
PMS_EVRSDCTRL9	0x0000 0434	0x8000 0434	LCK bit is cleared.
PMS_EVRSDCTRL11	0x1207 0909	0x9207 0909	LCK bit is cleared.

Table 18 TC39x, TC38x, TC37x, TC37xEXT, TC36x, TC35x, TC33xEXT and TC3Ex registers that may have different reset values than documented in TC3xx User's Manual (cont'd)

Register	Initial reset value	Reset value defined in User's Manual	Remark
PMS_EVRSD COEFF0	0x3508 73B6	0xB508 73B6	LCK bit is cleared.
PMS_EVRSD COEFF1	0x2294 6C46	0xA294 6C46	LCK bit is cleared.
PMS_EVRSD COEFF6	0x0097 1802	0x8097 1802	LCK bit is cleared.
PMS_EVRSD COEFF7	0x0000 D8F7	0x8000 D8F7	LCK bit is cleared.
PMS_EVRSD COEFF8	0x0017 1002	0x8017 1002	LCK bit is cleared.
PMS_EVRSD COEFF9	0x0000 A0AF	0x8000 A0AF	LCK bit is cleared.
SCU_OSCCO N	0x0000 0258 for UCB_DFLASH. OSCCFG = 0; 0x0XX0 XXXX otherwise	0x0000 0X1X	SCU_OSCCFG setting is recovered from UCB_DFLASH

Note: The registers listed in the following table apply to TC33x/TC32x.

Table 19 TC33x/32x registers that may have different reset values than documented in TC3xx User's Manual

Register	Initial reset value	Reset value defined in User's Manual	Remark
P20_IOCR0	0x0010 0000	0x0000 0000 (HWCFG6 = tri-state)	TESTMODE pin is PU (input pull-up), even with HWCFG6 = tri-state, as described in Data Sheet.
SBCU_DBCNTL	0x0000 7002	0x0000 7003	Bit EO is "Status of BCU Debug Support Enable" and only set after reset when OCDS is enabled. This bit is controlled by Cerberus.
HF_OPERATION	0x0000 0X00	0x0000 0000	RES bits shall be ignored.
PMS_EVRSDCTRL0	0x0000 0003	0xC000 0003	LCK and UP bits are cleared.
PMS_EVRSDCTRL1	0x0012 0012	0x8012 0012	LCK bit is cleared.
PMS_EVRSDCTRL2	0x140A 0909	0x940A 0909	LCK bit is cleared.
PMS_EVRSDCTRL3	0x0000 0001	0x8000 0001	LCK bit is cleared.
PMS_EVRSDCTRL4	0x2000 2209	0xA000 2209	LCK bit is cleared.
PMS_EVRSDCTRL5	0x001B 7566	0x801B 7566	LCK bit is cleared.
PMS_EVRSDCTRL6	0x0081 0000	0x8081 0000	LCK bit is cleared.
PMS_EVRSDCTRL7	0x2061 0400	0xA061 0400	LCK bit is cleared.

Table 19 TC33x/32x registers that may have different reset values than documented in TC3xx User's Manual (cont'd)

Register	Initial reset value	Reset value defined in User's Manual	Remark
PMS_EVRSD CTRL8	0x1070 0000	0x9070 0000	LCK bit is cleared.
PMS_EVRSD CTRL9	0x0000 4040	0x8000 4040	LCK bit is cleared.
PMS_EVRSD CTRL10	0x130C 0719	0x930C 0719	LCK bit is cleared.
PMS_EVRSD COEFF0	0x0052 0083	0x8052 0083	LCK bit is cleared.
PMS_EVRSD COEFF1	0x17B2 6996	0x97B2 6996	LCK bit is cleared.
PMS_EVRSD COEFF2	0x4924 8BD9	0xC924 8BD9	LCK bit is cleared.
PMS_EVRSD COEFF3	0x0780 00A3	0x8780 00A3	LCK bit is cleared.
SCU_OSCCON	0x0000 0258 for UCB_DFLASH. OSCCFG = 0; 0x0XX0 XXXX otherwise	0x0000 0X1X	SCU_OSCCFG setting is recovered from UCB_DFLASH

SAFETY_TC.H011 SM[HW]:GTM:TOM_TOM_MONITORING_WITH_IOM – Additional information

Safety mechanism SM[HW]:GTM:TOM_TOM_MONITORING_WITH_IOM in the AURIX™ TC3xx Safety Manual describes how the usage of redundant ATOM/TOM channels in combination with the IOM enables the detection of faults on TX PWM generated by the GTM.

However, the description of this safety mechanisms omits to explicitly mention that the redundant ATOM/TOM channel shall be selected from different modules.

Indeed, if the mission and monitor ATOM/TOM channels are selected from the same module then SM[HW]:GTM:TOM_TOM_MONITORING_WITH_IOM would not be able to detect faults of the GTM logic (within the module) shared by both channels.

Also, FMEDA assumes that the redundant ATOM/TOM channels are selected from different modules.

Recommendation

To implement SM[HW]:GTM:TOM_TOM_MONITORING_WITH_IOM select the redundant ATOM/TOM channel from different ATOM/TOM modules.

Alternatives

If it is not possible to follow the recommendation above (e.g. because of lack of resources), it is recommended to implement alternative safety mechanisms instead (see also text module SAFETY_TC.001 for TC33x/TC32x):

Option 1

- Implement SM[HW]:GTM:TOM_TOM_MONITORING_WITH_IOM with ATOM channel and TOM channel (as redundancy) from different modules, respectively.

<u>Considerations handling of FMEDA for systems using ATOM/TOM channels (Option 1) from the same module</u>:

For systems where two ATOM/TOM channels from the same module are used in a redundant manner, the changes indicated in the following have to be made in the FMEDA to reflect the limitations in terms of coverage of SM[HW]:GTM:TOM_TOM_MONITORING_WITH_IOM:

- ESM[SW]:GTM:TOM_TIM_MONITORING and
- SM[HW]:GTM:TOM_TOM_MONITORING_WITH_IOM

has to be removed from columns “Safety Mechanism to Prevent Violation of Safety Goal” and “Safety Mechanism to Prevent Faults from Being Latent” in the

FMEDA. The changes are to be implemented for the architectural elements present in **Table 20** and **Table 21** for the following reasons.:

- ESM[SW]:GTM:TOM_TIM_MONITORING is not applicable because it is not relevant in case of redundant ATOM/TOM channels with IOM use-case
- SM[HW]:GTM:TOM_TOM_MONITORING_WITH_IOM will not be able to detect faults from the shared logic. Below architectural element present in **Table 20** and **Table 21** are the shared logic between two ATOM or TOM channels used from the same module
- ESM[SW]:IR:ISR_MONITOR has to be kept as it is for the interrupt related failure modes that still can be detected by this ESM.

Table 20 FMEDA configuration changes for redundant ATOM channels from same ATOM module

Name	Sub-Part	Element Class	Function Name	Classification	Safety Mechanism to Prevent..	
					..Violation of Safety Goal	..Faults from Being Latent
GTM ATOM (mission)	MC_GTM _TC3xx. ATOMx	m	PWM_ GEN	Single Point Fault	-	-

Table 21 FMEDA configuration changes for redundant TOM channels from same TOM module

Name	Sub-Part	Element Class	Function Name	Classification	Safety Mechanism to Prevent..	
					..Violation of Safety Goal	..Faults from Being Latent
GTM TOM (mission)	MC_GTM _TC3xx. TOMx	m	PWM_ GEN	Single Point Fault	-	-

Option 2

- Use ESM[SW]:GTM:TOM_TIM_MONITORING instead of SM[HW]:GTM:TOM_TOM_MONITORING_WITH_IOM

SAFETY_TC.H013 ESM[SW]:SYS:MCU_FW_CHECK - Access to MC40 FAULTSTS register – Additional information

The FSI RAM is used to configure the PFLASH. For security related reason, the access to this RAM is restricted. Therefore, in order to avoid accesses to this RAM through its SSH, the MBIST Controller 40 (MC40) is not disclosed in the AURIX™ TC3xx Target Specification/User's Manual.

However, according to Appendix A of the Safety Manual, for SSH(40) register MC40_FAULTSTS must be compared to an expected value by ESM[SW]:SYS:MCU_FW_CHECK after reset.

Recommendation

When implementing ESM[SW]:SYS:MCU_FW_CHECK, the register address listed below has to be used to access the FAULTSTS register of MBIST Controller 40:

- MC40_FAULTSTS (0xF006 38F0)

SAFETY_TC.H017 Safety Mechanisms requiring initialization - Documentation update

In chapter "Safety Mechanisms" of the AURIX™ TC3xx Safety Manual, safety mechanisms that need to be initialized by Application SW have a link in the "Init Conditions" field to a Safety Mechanism Configuration (SMC). This SMC provides a description of what has to be implemented to activate the respective safety mechanism (SM).

This is not valid for all safety mechanisms. Some of them have no SMC as "Init Conditions", although they need to be activated.

Documentation update

Following is the list of safety mechanisms that have to be activated with the respective “Init Conditions”, in addition to the SMs that are already listed with a link to a SMC in field “Init Conditions” in the Safety Manual:

SM[HW]:CLOCK:ALIVE_MONITOR

Init conditions

The application SW shall enable the clock alive monitoring by setting the corresponding bit in CCUCON3 register after PLLs have been set up and are running.

SM[HW]:CPU:TPS

Init conditions

The application SW shall enable the temporal protection system (configure CPU_SYSCON.TPROTEN = 1_B).

SM[HW]:CPU:TPS_EXCEPTION_TIME_MONITOR

Init conditions

The application SW shall enable the temporal protection system (configure CPU_SYSCON.TPROTEN = 1_B).

SM[HW]:CPU:CODE_MPU

Init conditions

The application SW shall configure the Code MPU according to TriCore™ TC1.6.2 Core Architecture Manual Volume 1 V1.1 - Chapter 10 “Memory Protection System”.

SM[HW]:CPU:DATA_MPU

Init conditions

The application SW shall configure the Data MPU according to TriCore™ TC1.6.2 Core Architecture Manual Volume 1 V1.1 - Chapter 10 “Memory Protection System”.

SM[HW]:CPU:UM0

Init conditions

The application SW shall configure the CPU access privilege level control to User-0 Mode (CPU_PSW.IO = 00_B).

SM[HW]:CPU:UM1

Init conditions

The application SW shall configure the CPU access privilege level control to User-1 Mode (CPU_PSW.IO = 01_B).

SM[HW]:CPU:SV

Init conditions

The application SW shall configure the CPU access privilege level control to Supervisor Mode (CPU_PSW.IO = 10_B). (Default configuration).

SM[HW]:CPU:STI

Init conditions

The application SW shall configure the safe task identifier (CPU_PSW.S = 1_B).

SM[HW]:DMA:TIMESTAMP

Init conditions

The application SW shall enable the appendage of a DMA timestamp (configure DMA_ADICRc.STAMP = 1_B).

SM[HW]:EMEM:CONTROL_REDUNDANCY

Init conditions

The application SW shall enable the EMEM module SRI control redundancy logic (EMEM_SCTRL.LSEN = 10_B).

SM[HW]:EMEM:READ_WRITE_MUX

Init conditions

The application SW shall configure the mode of the EMEM tiles via EMEM_TILECONFIG and enable access to the EMEM tiles via EMEM_TILECC and EMEM_TILECT.

SM[HW]:LMU:CONTROL_REDUNDANCY

Init conditions

The application SW shall enable the LMU control redundancy logic (LMU_SCTRL.LSEN = 10_B).

SM[HW]:NVM.PFLASH:ERROR_CORRECTION

Init conditions

The application SW shall enable the ECC error correction (CPUx_FLASHCON2.ECCCORDIS = 10_B).

Enabled after reset.

SM[HW]:NVM.PFLASH:ERROR_MANAGEMENT

Init conditions

The application SW shall enable address buffer recording (CPUx_FLASHCON2.RECDIS = 10_B).

Enabled after reset.

SM[HW]:NVM.PFLASH:FLASHCON_MONITOR

Init conditions

The application SW shall initialize CPUx_FLASHCON2.

SM[HW]:SPU:REDUNDANCY_SCC

Init conditions

SM[HW]:SPU:REDUNDANCY_SCC is enabled when either of SM[HW]:SPU:PARTIAL_REDUNDANCY or SM[HW]:SPU:REDUNDANCY are enabled.

SM[HW]:SCU:EMERGENCY_STOP

Init conditions

By default after reset, the Synchronous mode is selected; in this mode, the application SW shall enable (via EMSR.ENON = 1_B) the setting of the Emergency stop flag (EMSR.EMSF) on an inactive-to-active level transition of the port input.

Alternatively, the application SW can:

- Select the Asynchronous mode (EMSR.MODE = 1); in this mode the occurrence of an active level at the port input immediately activates the emergency stop signal.
- Configure Alarm(s) in SMU to trigger an Emergency Stop

SM[HW]:SMU:RT

Init conditions

The application SW shall enable the Recovery Timers (RTy, where y = 0,1) via RTC.RTyE = 1_B.

Recovery Timers (RTy, where y = 0,1) are enabled after Application Reset to service the WDT timeout alarms.

SM[HW]:SMU:FSP_MONITOR

Init conditions

FSP Monitor is enabled after Power-on Reset. The application SW shall ensure that the FSP is in the Fault Free State (SMU_ReleaseFSP()) before entering the RUN state with the SMU_Start() command.

SM[HW]:PMS:VDDM_MONITOR

Init conditions

SMC[SW]:PMS:Vx_MONITOR_CFG

SAFETY_TC.H019 SM[HW]:NVM.FSIRAM:REG_MONITOR_TEST should not be considered

The SM[HW]:NVM.FSIRAM:REG_MONITOR_TEST is defined in the AURIX TC3xx Safety Manual as a self-test mechanism for the FSI.RAM SFFs.

Recommendation

The system integrator has to consider the FSI.RAM SFFs as not testable, as in fact there are no means to trigger this test.

Note: SM[HW]:NVM.FSIRAM:REG_MONITOR_TEST is not part of the FMEDA and will not impact the FIT rate.

SAFETY_TC.H020 Test of SM[HW]:VMT:REG_MONITOR is missing - Documentation update

The "Tests" field of SM[HW]:VMT:REG_MONITOR (section 6.500 in AURIX™ TC3xx Safety Manual v1.04 and higher versions) is empty. Users may think that the safety flip-flops mechanism is not testable, which is not true.

Documentation update

SM[HW]:VMT:REG_MONITOR is testable through SMU by ESM[SW]:SMU:REG_MONITOR_TEST.

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.H019 Connection on ERU input E_REQ7(5)

Table "Connections of SCU" in the TC37x and TC37xEXT specific Appendix to the AURIX™ TC3xx User's Manual includes the following connection

- SCU:E_REQ7(5) from GTM:SCU_TRIG(3)

As TOM3 is not present in the GTM implementation in TC37x and TC37xEXT, this connection is not functional in the TC37x and TC37xEXT and therefore should not be used.

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 $f_{SPB} \geq 20$ 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.H023 Behavior of bit RSTSTAT.PORST after wake-up from standby mode

After cold-power on (power up from no power supply), bit RSTSTAT.PORST is always set independent of PORST pad level (pulled high or low by user).

After wake-up from standby, bit RSTSTAT.PORST indicates if the PORST pad was asserted after the wake-up trigger.

Recommendation

If the user expects that bit RSTSTAT.PORST is always set after wake-up from standby, the PORST pad should be kept low externally until all supplies are in operating condition.

SCU_TC.H025 Field EEA in register CHIPID - Additional information

In the SCU chapter of the TC3xx User's Manual, field EEA in register CHIPID is described as follows:

Table 22 Field EEA in register CHIPID

Field	Bits	Type	Description
EEA	16	rh	**Emulation or ADAS Extension Available** Indicates if the emulation or ADAS extension hardware is available or not. 0_B EEC is not available (SAK-TC3xxxU or SAK-TC3xxxP) 1_B EEC is available (SAK-TC3xxxE or SAK-TC3xxxF)

The product names/feature packages (SAK-TC3xxxU, ...) mentioned in this description shall only be understood as examples; they may not exist for each TC3yx device variant.

Recommendation

For a summary of the functionality available in each TC3yx device variant see the TC3yx Data Sheet Addendum.

As can be seen from the Chip ID values defined in this document, bit EEA = 1_B in TC39x, TC37xEXT, TC35x and TC33xEXT devices, and bit EEA = 0_B in TC3Ex, TC38x, TC37x, TC36x and TC33x/TC32x devices.

For a summary of the functionality of TC3xx emulation devices and their Chip ID values see the TC3xx_ED Data Sheet.

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 23 SENT input pads and considered neighbors for TC37x and TC37xEXT

Considered left neighbors		SENT input		Considered right neighbors	
		Pad	Channel		
P02.6	P02.7	P02.8	0C	P02.9	P02.10
P00.0	P00.1	P00.2	1B	P00.3	P00.4
P02.5	P02.6	P02.7	1C	P02.8	P02.9
P02.4	P02.5	P02.6	2C	P02.7	P02.8
P02.3	P02.4	P02.5	3C	P02.6	P02.7
P15.0	P15.1	P15.2	10D	P15.3	P15.4
P15.2	P15.3	P15.4	11D	P15.5	P15.6
P02.2	P02.3	P02.4	12B	P02.5	P02.6
P02.1	P02.2	P02.3	13B	P02.4	P02.5
P02.0	P02.1	P02.2	14B	P02.3	P02.4

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 24 Effect of Driver Settings Fss, Sms, Sm on SENT inputs for TC37x and TC37xEXT

SENT Channel			Neighbors: Fast pads configured as Fss, others Sms/Sm	
Name	**Number**	**Pin**	**VILD(DIST4N)**	**VILD(DIST2N)**
SENT:SENT0C	0C	P02.8	x	OK
SENT:SENT1B	1B	P00.2	x (TC37xEXT) OK (TC37x)	OK
SENT:SENT1C	1C	P02.7	x	OK
SENT:SENT2C	2C	P02.6	x	x (TC37xEXT) OK (TC37x)
SENT:SENT3C	3C	P02.5	x	OK
SENT:SENT10D	10D	P15.2	x (TC37xEXT) OK (TC37x)	OK
SENT:SENT11D	11D	P15.4	x	OK
SENT:SENT12B	12B	P02.4	x	OK
SENT:SENT13B	13B	P02.3	x	x (TC37xEXT) OK (TC37x)
SENT:SENT14B	14B	P02.2	x	OK

Table 25 Effect of Driver Settings Fsm, Fm, Sms, Sm on SENT inputs for TC37x and TC37xEXT

SENT Channel			Neighbors: Fast pads configured as Fsm or Fm, others Sms/Sm	
Name	**Number**	**Pin**	**VILD(DIST4N)**	**VILD(DIST2N)**
SENT:SENT0C	0C	P02.8	OK	OK
SENT:SENT1B	1B	P00.2	OK	OK
SENT:SENT1C	1C	P02.7	OK	OK
SENT:SENT2C	2C	P02.6	OK	OK
SENT:SENT3C	3C	P02.5	OK	OK
SENT:SENT10D	10D	P15.2	OK	OK
SENT:SENT11D	11D	P15.4	OK	OK
SENT:SENT12B	12B	P02.4	OK	OK
SENT:SENT13B	13B	P02.3	OK	OK
SENT:SENT14B	14B	P02.2	OK	OK

Table 26 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 27 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.

SENT_TC.H007 Range for divider value DIV - Documentation correction

In section “Baud Rate Generation” and in the description of register CFDRx in the SENT chapter of the TC3xx User’s Manual, the range for the divider value DIV is documented as

- DIV = [2200, 49100]

The upper limit of this range is incorrect.

Documentation correction

The correct range that can be used for the divider value DIV is

- DIV = [2200, 52428]

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 following register addresses have to be written to clear the FSI RAM Fault Status and ECC Detection Register:

MCi_FAULTSTS (i=40, 0xF00638F0) = (16-bit write) 0x0

MCi_ECCD (i=40, 0xF0063810) = (16-bit write) 0x0

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.H016 SMU_stdby restriction for using P33.8 as Emergency Stop input

Description

SMU_core can be configured to report the presence of faults on port pin P33.8 using the Fault Signaling Protocol. However, SMU_stdby in combination with SMU_core can also be configured for handling common cause faults in a diverse way.

On the detection of faults, the SMU_STDBY sets P33.8 in high impedance state (fault state). In order to recognize the high impedance state of the P33.8 as the fault state, an external pull-down is needed if bi-stable FSP is used (see for example section “Interface to the Pads (ErrorPin)” in the SMU chapter of the User’s Manual). When the SMU_stdby sets P33.8 in high impedance state (fault state), the SMU_STS.FSP[0] bitfield as well as P33_IN.P8 bitfield do not reflect the actual logic level of the pin P33.8.

While SMU_stdby sets P33.8 to high impedance the port triggered emergency stop function on this port will be disabled and no emergency stop event will be generated as consequence.

Recommendation

Therefore it is recommended to use port pin P21.2 (PORT B) as Emergency Stop input when SMU_stdby is configured to report fault on P33.8.

SMU_TC.H017 Handling of ALM21[7] when safety flip-flop self-test is executed

Description

After execution of the Safety flip-flop self-test when the PMS/PMSLE module is enabled (SMU_RMCTL[5] = 1), the alarm ALM21[7] might not be reported. This is due to the fact that the clock of the SMU_stdby (where the ALM21[7] belongs to) is slower than the alarm pulse, therefore the alarm might not be reported.

This happens only during the Safety flip-flop self-test, because in case of a real error during run mode, the alarm signal will persist longer and it will be caught also by a slower clock.

Furthermore, flags used to signal the end of the test (SMU_RMSTS[5]) or a failure in the test (SMU_RMEF[5]) are working properly.

A more detailed view shows that the error signal coming out from the PMS/PMSLE Safety flip-flop is connected to both SMU_core and SMU_stdby for diverse processing during run mode. During the Safety flip-flop self-test of the PMS/PMSLE module, it is expected that the error signal should trigger an alarm in the SMU_core only, which means that the ALM21[7] is superfluous.

Recommendation

When the Safety flip-flop self-test is executed and the PMS/PMSLE module is enabled (SMU_RMCTL[5] = 1), ignore alarm ALM21[7] and clear it when the test is completed.

<u>SRI_TC.H001</u> 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.

SSW_TC.H001 **Security hardening measure for the startup behavior**

In order to increase the robustness of the debug protection mechanism against malicious attacks, it is now strongly suggested to always apply another layer of protection in combination with it.

Recommendation

On top of the debug protection mechanism, enabled via UCB_DBG through the HF_PRONCONDBG.DBGIFLCK bit using a 256-bit password, user shall set the global PFLASH or DFLASH read protection.

Both protections can be enabled individually or together. It is not mandatory to set both protections at the same time.

In most cases PFLASH will be the preferred option since standard drivers for DFLASH (e.g. for EEPROM emulation) do not support DFLASH protection.

In order to enable the global PFLASH read protection, HF_PROCONPF.RPRO has to be set to 1 inside the UCB_PFLASH_ORIG/COPY.

In order to enable the global DFLASH read protection, HF_PROCONDF.RPRO has to be set to 1 inside the UCB_DFLASH_ORIG/COPY.

Be aware that the global read protection will apply also a write protection over the entire PFLASH or DFLASH memory respectively.

The enabled read protection is always effective for the startup hardening. For the Flash read access by CPUs it has only an effect in case the device is not booting from internal Flash.

In case a software update is needed, the write protection, inherited as side effect from the global read protection, can be temporarily disabled executing the "Disable Protection" command sequence.

The PFLASH write protection is also contained in the same UCB_PFLASH_ORIG/COPY, so this leads to have only one password (different from the Debug password) to disable write and read protection mechanisms at the same time.

If the user removes the global PFLASH read protection this will remove also the PFLASH write protection at the same time.

Same for the DFLASH write protection, which is included in the UCB_DFLASH_ORIG/COPY. Another single password is used to disable write and read protection over Data Flash 0 at the same time. Data Flash 1 and HSM PFLASH sectors are protected with another security mechanism via “exclusive protection”.

The disabled protection is valid until the next reset or executing the “Resume Protection” command sequence.

For further details please refer to AP32399 “TC3xx debug protection (with HSM)” or to chapter “Non Volatile Memory (NVM) Subsystem“ in the AURIX™ TC3xx User’s Manual.

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.