AURIX™ TC4Dx errata sheet

Marking/Step: (E)ES-AB

32-bit single-chip microcontroller
10581AERRA

About this document

Scope and purpose

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

Please take note of the following information:

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

Current documentation

- AURIX™ TC4Dx user manual
- TC4Dx A-step COM Datasheet
- TriCore™ TC1.8 architecture manual volume 1
- TriCore™ TC1.8 architecture manual volume 2
- AURIX™ TC4x architecture reference
- AURIX™ TC4x debug and trace
- AURIX™ TC4Dx Safety Manual

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

Conventions used in this document

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

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

About this document

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

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

Table of contents

1 Errata overview

List of errata referenced in this document.

Table 1 Functional deviations

Issue title	Change	Page
[ADC_TC.030] Missing access errors for specific ADC registers		12
[CANXL_AI.001] Wrong TX messages sequence when using TX priority queue slot 0-7		12
[CANXL_AI.002] Wrong Tx descriptor fetch after TX-scan in specific condition		13
[CCU_TC.011] Arbitrary SRI clock security error alarm		14
[CCU_TC.012] Sporadic CCUSTAT.LCK after writing SYSPLLCON1 or PERPLLCON1		15
[CCU_TC.013] Functional problem of amplitude regulation in OSC module		16
[CCU_TC.015] MPLL accumulated jitter violation		17
[CCU_TC.016] Amplitude regulation in kHz XTAL might be unstable		17
[CCU_TC.017] Unexpected alarm when activating KDIV monitors		17
[CCU_TC.018] fTPB to feGTM relation in divider ratio table to be changed to fTBP to feGTM_half relation		19
[CCU_TC.020] Short pulse on output clock when changing KDIV		19
[CCU_TC.021] Mismatch in divider value information and register description		20
[CCU_TC.022] Update divider ratio table fFSI/fSPB		20
[CCU_TC.023] Inconsistency between SYSPLLCON1 register description and SYSPLL functional description		21
[CCU_TC.025] Allowed divider ratios between f_{SRICS} and f_{FSI} is missing		22
[CCU_TC.026] UM correction for CPB bitfield		22
[CCU_TC.028] Remove SUP mode from TC4xx UM Clock part		23
[CPU_TC.134] FPU conversion instructions assert invalid flag before rounding		23
[CPU_TC.144] Retriggering of CFI lockstep self-test fails		24
[CPU_TC.146] Potential performance issue when quad-word (128-bit) load or store instructions are followed by loop instruction (LOOP, LOOP.U)		25
[CPU_TC.147] CPUCS reset values when CSRM debug enabled		26
[CPU_TC.148] No APU protection on internal path from CPU to local scratchpad RAMs		27
[CPU_TC.149] Incorrect reference to PSW.CDC in DBGSR description		29
[DAP_TC.012] Timeout counters when MWBE = 1, waiting time greater than programmed		29
[DAP_TC.013] Update table DAP/JTAG interface pins' on-chip pull device setting		30
[DMA_TC.069] Spurious DMA lockstep alarms		30
[DMA_TC.070] DMA resource partition overflow error status is unreliable		31
[DRE_TC.001] Pending requests in registers COBL_BPR0/1 cannot be cleared		31
[DRE_TC.002] Pending requests in register CIBL_BPR cannot be cleared		32

(table continues...)

1 Errata overview

Table 1 (continued) Functional deviations

Issue title	Change	Page
[DRE_TC.003] Frame loss in multi-cast due to CRE abort sequence on the CAN side on the previous multi-cast transfer		32
[DRE_TC.005] ETH2CAN - ACF_CAN_ADDR/LE_ACF_CAN_ADDR field not being reset		33
[DRE_TC.006] Inconsistency between GETH and LETH while triggering error interrupt by the DRE		33
[DRE_TC.008] Forwarding ID (FID) wrongly assembled for LETH		34
[eGTM_AI.530] GTM_AEI: AEI_READY erroneously set		34
[eGTM_TC.005] Kernel reset corner case causes AXI hang-up		34
[eGTM_TC.006] eGTM wrapper cannot be switched off (CLC) when soft suspend is active		35
[ERAY_TC.001] Wrong reset behavior SSH-ERAY	new	35
[GETH_AI.029] CBS credit not decremented during the IPG phase of transmission		36
[GETH_AI.032] Time Aware Shaper (TAS) additional IPG in case of back-to- back packet transmission		37
[GETH_AI.033] Received packet not routed to the programmed VLAN filter fail queue		37
[GETH_AI.034] Minimum inter packet gap of the MAC transmitter mismatches the programmed non-standard value MII speed modes		38
[GETH_AI.035] Receive interrupt watchdog clock-based timer not reset on other timers timeout and receive inter triggering events		39
[GETH_AI.036] MAC incorrectly starts packet transmission before threshold number of bytes are available in the transmit queue		39
[GETH_AI.037] Receive DMA stops operation when packet flush initiation and exit from suspend mode overlap		40
[GETH_AI.038] Unintended closure of receive descriptor when intermediate descriptor contains definition error		40
[GETH_AI.039] Transmit packet not terminated when underflow occurs in MII speed modes		41
[GETH_AI.040] Rx DMA stall due to incomplete context descriptor closure	new	41
[GETH_AI.041] RX DMA stall with varying RX packet length and active Tx DMA in forwarding port	new	42
[GETH_AI.042] RX frames stall in case of normal status word only	new	42
[GETH_AI.045] Bridge is padding extra 8 bytes in forwarding packet due to delayed word acceptance in the egress port	new	43
[HSPHY_TC.005] Loss of Receive Ethernet communication during temperature change		43
[LETH_AI.005] CBS credit not decremented during the IPG phase of transmission		44
[LETH_AI.008] Time Aware Shaper (TAS) additional IPG in case of back-to- back packet transmission		45

(table continues...)

Table 1 (continued) Functional deviations

Issue title	Change	Page
[LETH_AI.010] Reading back contents of the register 10BT1S_PLCA_TIMER, the TOT and BT bitfields are swapped		46
[LETH_AI.011] 10BT1S Delayed transmission after TO starts as follower	new	46
[LETH_AI.013] Long COMMIT without TRANSMIT	new	47
[LETH_AI.014] 10BT1S Incorrect encoding of first bit after TX command	new	47
[LETH_AI.015] 10BT1S Unintended dropping of next received packet when preceding corrupted SILENCE symbol resembles any of the SSD/ESD/HB/BEACON symbols	new	48
[LETH_AI.016]PLCA cycle time between 2 BEACON deviates from expected value	new	48
[LETH_AI.017] Incorrect description of register bit-field Portj_B10T1S_PLCA_Sts.PS	new	49
[LETH_AI.018] RX frames stall in case of normal status word only	new	49
[LETH_AI.019] 10BT1S PLCA_R bit of Portj_B10T1S_PLCA_Ctrl register is not self clearing	new	51
[LETH_AI.020] 10BT1S Coordinator transmits the packet without BEACON (during unintended TO)	new	51
[LETH_AI.022] 10BT1S Unintended dropping of next received packet when first of the two SYNC symbols right shifted by one bit position resembles the SSD symbol	new	52
[LETH_AI.023] Incorrect access type is specified for bit-fields in register Portj_B10T1S_PLCA_Sts	new	52
[LETH_AI.024] Transmit timestamp is not properly transferred in descriptor if TxDMA channel is mapped to any queue except TxQ0 and bridge is enabled.	new	53
[LETH_TC.010] Missing PTP time sync concept among all LETH0 MAC ports		53
[LETH_TC.013] LETH0 PPS output for some PORT pins is inverted		55
[LETH_TC.015] FRP last instruction index is not updated correctly in receive descriptor (RDES2)		56
[LETH_TC.017] Layers of LETH0_P3 MDIO mapped in P13.6 for TC4Dx		57
[LMU_TC.003] LMU can lock up when writes immediately follow a protection modification		57
[MCDS_TC.069] COUNT field of FIFOOVRCNT wraps around and does not saturate		58
[MCDS_TC.070] FIFOPRE[11:5] consideration for PRE and PRE-POST mode decision		58
[MCDS_TC.071] Multick not working as expected		59
[MCDS_TC.072] Unexpected VM change STATE messages		59
[MCDS_TC.073] VM change STATE message after IP message instead of before		59
[MCDS_TC.075] Wrong first tline after TRIF FIFO overrun		60
[MCDS_TC.077] MCDS_TSUEMUCNT not counting		60
[MCDS_TC.078] Wrong configuration of DMA0 Trace	new	61

(table continues...)

1 Errata overview

Table 1 (continued) Functional deviations

Issue title	Change	Page
[MSC_TC.018] TFIP is used to select service request output line SR0-3 for time frame Manchester interrupt TFMI instead of TFMIP		61
[MSC_TC.019] MSC alternate SRI clock source feature is not functional		61
[MSC_TC.025] Slow channel feature is not functional		62
[MSC_TC.026] PPDE field considered when EXEN=1 and EXEN=0 as well		62
[PCIE_AI.002] L1 PM Substates Capabilities Register fields are not HWINIT		62
[PCIE_AI.003] PME retransmission time not satisfied		63
[PCIE_AI.004] DMA Configuration Register access failure		63
[PCIE_AI.005] TLP with ECRC error not handled as advisory non-fatal error	changed	64
[PCIE_AI.006] Data transfer completes without channel status update		64
[PCIE_AI.007] Updating PCIe common reference clock configuration	new	65
[PMS_TC.020] XRAM uncorrectable error alarm issue		65
[PMS_TC.029] Resets MONBISTSTAT		66
[PMS_TC.030] clk_xtal is not synchronous for one cycle after transition from STANDBY1 to RUN mode		66
[PMS_TC.032] HPBG trimming via SCR_SCU_PMSHPBG_BGTRIM		66
[PORTS_TC.010] P20.6 is in Pull up mode even if HWCFG[6]=0		67
[PORTS_TC.011] PCSRSEL=1 blocks the emergency stop signal		67
[PPU_AI.003] Single stepping might lead to problems		68
[PPU_TC.002] Trace flush is not working		68
[PPU_TC.005] External event (CTRL.REQWU) only cleared after second wait event instruction executes	changed	69
[SAFETY_TC.031] Test procedure update for SM[HW]:SRI:ERROR_HANDLING and SM[HW]:LLI:ERROR_HANDLING		69
[SAFETY_TC.032] Wrong register name in GETH:ISR_MONITOR		69
[SAFETY_TC.033] Wrong alarm name and reaction for LMU REDUNDANCY SM		70
[SAFETY_TC.034] Missing double-bit error alarm for DMI and STU access to VDSP memory	new	70
[SCR_TC.026] I2C does not support multi-master systems		71
[SCR_TC.029] I2C does not support Software reset (I2C_SRST.SRST)		71
[SCR_TC.030] Incorrect baud rate after clock stretching		72
[SCR_TC.032] [I2C] Slave does not return to IDLE state after receiving NACK in transmit mode		72
[SCR_TC.034] [I2C] Slave does not return to IDLE state after sending NACK to the data byte of a general call		73
[SCR_TC.035] [OCDS] debugging of SCR reset does not work		73

(table continues...)

1 Errata overview

Table 1 (continued) Functional deviations

Issue title	Change	Page
[SDMMC_AI.003] Nst timing violation for STOP command	new	73
[SMM_TC.006] Port issue if port pin configured to reset at the end of reset		74
[SMU_TC.016] FS timing status not updated		74
[SMU_TC.019] Missing P33.8 EMS A input in pin definition tables	new	75
[SRI_TC.004] Writes to reserved address space are overwriting DOM0_ACCEN registers with a similar offset from their base		75
[TRIF_TC.001] TME target address needs to be 32-byte aligned		75
[VMT_TC.004] Wrong register value shown in MBIST chapter	new	76
[xSPI_TC.002] Clock low to CS high (tCKLCSH) timing JESD251C specification violation	changed	77

Table 2 Parametric deviations

Issue title	Change	Page
[CCU_TC.P003] CLOCK Parameter changing (XTAL1 amplitude)		78
[DAP_TC.P003] DAP pin driver strength not sufficient		78
[MSC_TC.P003] LVDS and SS timing update		78

Table 3 Application hints

Issue title	Change	Page
[ADC_TC.H047] Unexpected global service request after GLSRCFGx reconfiguration		80
[ADC_TC.H049] CHy_STAT register triggers arbitration issue		80
[ADC_TC.H051] Single EMUX input per TMADC instance	new	80
[ASCLIN_TC.H010] Incomplete reset isolation		80
[CCU_TC.H019] Amplitude regulation in OSC module		81
[CDSP_TC.H001] Application reset impacts CDSP debug registers		81
[CDSP_TC.H002] Incomplete reset isolation		81
[CPU_TC.H025] Avoiding unbounded delays in store buffer residency	new	81
[CPU_TC.H026] Spurious lockstep error triggered on TriCore™ if DSPR is accessed by CPU or slave before DSPR-MBIST	new	92
[DRE_TC.H002] Throughput performance drop for GETH or LETH to LETH forwarding via DRE		92
[GETH_AI.H005] The MMC_Rx_Packet_SMD_Err_Cntr counter incorrectly updated in internal Loopback mode		93
[GETH_AI.H006] Packets received in the highest numbered queue not sent to the software after the packet to be duplicated to multiple DMAs is dropped		93
[GETH_TC.H007] Incomplete reset isolation		94
[HSPHY_TC.H004] Problem with clk_sram during application reset		94

(table continues...)

Table 3 (continued) Application hints

Issue title	Change	Page
[HSPHY_TC.H007] Missing register description in the user manual to configure Rx adaption control		95
[HSPHY_TC.H008] Incorrect Initialization sequence for SGMII leading to alarms		95
[HSSL_TC.H002] Incomplete reset isolation		96
[LETH_AI.H001] Combination of forwarding and reception to host on Port0 does not works as expected	new	96
[LETH_TC.H001] Incomplete reset isolation		96
[LETH_TC.H002] DMA channel locks up when time stamping is enabled and context descriptor is not available		97
[MCDS_TC.H008] Trace start from halt after reset state with two MCDS instances		97
[MCDS_TC.H009] Fan comparators do not trigger in case of multiple discontinuities		97
[MCMCAN_TC.H014] Incomplete reset isolation		98
[MCMCAN_TC.H015] CRE ERRCTRL Register Wrong Access Mode		98
[MSC_TC.H016] Incomplete reset isolation		98
[PACKAGE_TC.H009] Max. compressive force during customer housing mounting		99
[PCIE_AI.H001] In non-D0 power state, the controller does not handle the Cpl TLP as an unexpected completion	new	99
[PCIE_TC.H002] Region upper address register (RGNUA) value is evaluated as "less or equal" instead of "strictly less than"		99
[PCIE_TC.H003] Interrupt detection should clear its internal status		100
[PCIE_TC.H005] Register write error response is not evaluated		100
[PCIE_TC.H006] Incomplete reset isolation		101
[PCIE_TC.H007] Outbound traffic is not blocked in hardware when MSE=0 nor when BME = 0		101
[PMS_TC.H020] Standby RAM address missing in PMS documentation		101
[PPU_AI.H001] Unclear statement in the user manual with respect to vector memory ECC errors		102
[PSI5S_TC.H002] Incomplete reset isolation		102
[PSI5_TC.H003] Incomplete reset isolation		102
[QSPI_TC.H010] Incomplete reset isolation		103
[QSPI_TC.H012] HS mode functionality is not supported for all baud-rate configurations		103
[SAFETY_TC.H021] Remove AA step designator from package specific data		103
[SCR_TC.H017] I2C clock synchronization in slave mode (clock stretching)		103
[SCR_TC.H018] Do not use master mode stop bit (I2C_CNTR.STP=1) in slave mode		104
[SCR_TC.H021] I2C Clock stretching		104

(table continues...)

Table 3 (continued) Application hints

Issue title	Change	Page
[SCR_TC.H022] OCDS SCR reset during monitor mode		104
[SENT_TC.H008] Incomplete reset isolation		104
[xSPI_TC.H001] XIP write is not supported		105
[xSPI_TC.H002] 8bit and 16bit XIP read issue through un-cached memory		105

Table 4 Legacy issues

Issue title	Change	Page
[CPU_TC.H024] Usage of atomic instructions SWAPMSK.W and LDMST to access registers with bit-fields that can also be updated by hardware (rwh)		106
[DMA_TC.071] Daisy Chain request is lost when repeat triggers too soon		107
[eGTM_AI.516] SPE-RTL: IRQ raised on disabled inputs		107
[eGTM_AI.517] (A)TOM: Missing edge on output signal (A)TOM_OUT		108
[eGTM_AI.H803] SPEC-(A)TOM: Missing priority information for register update		109
[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		109
[FlexRay_AI.088] A sequence of received WUS may generate redundant SIR.WUPA/B events		110
[FlexRay_AI.089] Rate correction set to zero in case of SyncCalcResult=MISSING_TERM		110
[FlexRay_AI.090] Flag SFS.MRCS is set erroneously although at least one valid sync frame pair is received		111
[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		111
[FlexRay_AI.092] Initial rate correction value of an integrating node is zero if pMicroInitialOffsetA,B = 0x00		112
[FlexRay_AI.093] Acceptance of start-up frames received after reception of more than gSyncNodeMax sync frames		112
[FlexRay_AI.094] Sync frame overflow flag EIR.SFO may be set if slot counter is greater than 1024		112
[FlexRay_AI.095] Register RCV displays wrong value		113
[FlexRay_AI.096] Noise following a dynamic frame that delays idle detection may fail to stop slot		113
[FlexRay_AI.097] Loop back mode operates only at 10 MBit/s		114
[FlexRay_AI.099] Erroneous cycle offset during start-up after abort of start-up or normal operation		114
[FlexRay_AI.100] First WUS following received valid WUP may be ignored		115
[FlexRay_AI.101] READY command accepted in READY state		115

(table continues...)

1 Errata overview

Table 4 (continued) Legacy issues

Issue title	Change	Page
[FlexRay_AI.102] Slot status vPOC!SlotMode is reset immediately when entering HALT state		115
[FlexRay_AI.103] Received messages not stored in Message RAM when in Loop Back Mode		116
[FlexRay_AI.104] Missing start-up frame in cycle 0 at coldstart after FREEZE or READY command		116
[FlexRay_AI.105] RAM select signals of IBF1/IBF2 and OBF1/OBF2 in RAM test mode		117
[FlexRay_AI.106] Data transfer overrun for message transfers Message RAM to Output Buffer (OBF) or from Input Buffer (IBF) to Message RAM		117
[FlexRay_AI.H004] Only the first message can be received in External Loop Back mode		119
[FlexRay_AI.H005] Initialization of internal RAMs requires one eray_bclk cycle more		120
[FlexRay_AI.H006] Transmission in ATM/Loopback mode		120
[FlexRay_AI.H007] Reporting of coding errors via TEST1.CERA/B		120
[FlexRay_AI.H009] Return from test mode operation		120
[FlexRay_AI.H010] Driver software must launch CLEAR_RAMS command before reading from E-Ray RAMs		121
[FlexRay_AI.H011] Behavior of interrupt flags in FlexRay™ Protocol Controller (E-Ray)		121
[HSCT_TC.H010] Interface control command timing on the LVDS ports		122
[MCMCAN_AI.025] Sporadic data corruption (payload) in case acceptance filtering is not finished before reception of data R3 (DB7..DB4) is completed		123
[MSC_TC.027] De-feature of ABRA for MSC		125
[QSPI_TC.006] Baud rate error detection in slave mode (error indication in current frame)		126
[SDMMC_AI.002] Transfer complete interrupt not generated post ADMA3 transfer		126
[SDMMC_AI.H001] Packet buffer is not fully utilized for the write traffic		126
[SENT_TC.H009] Unexpected NNI error behavior		127
[SMU_TC.015] SMU alarm emulation might trigger unwanted active alarm reaction		127

2 Functional deviations

2.1 [ADC_TC.030] Missing access errors for specific ADC registers

Description

Missing access errors for the following ADC registers:

- CLKEN_CDSP
- TMADCx_SWTRCFG
- TMADCx_RESFCLR
- TMADCx_ERRFCLR
- TMADCx_BNDFCLR
- TMADCx_RESFSET
- CDSP_BNDCF
- CDSP_BNDFCL
- CDSP_GLOBHCD
- CDSP_GLOBRCD
- CDSP_GLOBRD
- CDSP_OSCCDSPA
- CDSP_OSCCDSPB
- CDSP_RESEVy
- CDSP_RESEVCLRy
- CDSP_WUERREV
- CDSP_WUERRCL
- CDSP_DSPa_BOUNDSEL

Scope

ADC registers

Effects

8-bit or 16-bit accesses to the register may result in unpredictable register content, but they do not get flagged with bus errors to the Communication Peripheral Bus (COMPB).

Workaround

Limit access to the registers listed above to 32-bit access.

2.2 [CANXL_AI.001] Wrong TX messages sequence when using TX priority queue slot 0-7

Description

Definition:

- PQ(N) defines the TX **P**riority **Q**ueue slot N
- FQ(N) defines the TX **F**IFO **Q**ueue N
- TX-Scan is the internal TX message arbitration. Two internal buffers are used to store TX messages with the highest priority

Application assumptions for the description:

- The PQ(N) is running concurrently with the FQ(N)

- The FQ(N) message is currently transmitted to the TX CAN bus and is stored in the first internal buffer
- The PQ(N) message is waiting to be sent as the next candidate for transmission (previous result of the TX-Scan). This message is stored in the second internal buffer
- The next message in the FQ(N) has a higher priority than the message in PQ(N)

The issue is caused by a missing check in the Message Handler during TX-Scan. In order to keep going with the same FQ, the TX-Scan considers not only the current, but also the next TX message. In some cases, this next message may have been already selected by the TX-Scan and stored in one of the two internal buffers. Therefore, the Message Handler keeps going with the same FQ, one message after the other. Although TX messages may have identical queue number N, it could be either a PQ(N) or a FQ(N) message. In current implementation, only the message queue number is considered. If the next message of the FQ(N) has the higher priority and the PQ(N) message is already in one of the two internal buffers, the TX-Scan selects the PQ(N), as the queue numbers are identical. There is no issue for FQ(N) and PQ(M) messages in the internal buffers, the next message in the FQ(N) will replace the PQ(M) message.

Scope

The problem occurs for the internal TX message arbitration, called TX-Scan. If a TX message from a PQ(N) (that is already selected as next candidate for transmission) has a lower priority than a new TX message in the FQ(N), then the PQ(N) message will start transmission first.

Effects

TX message ordering may not be correct according to the TX message priority.

***Note**: This issue has no safety impact since any delay of transmission caused by this issue can be detected using the existing safety mechanisms.*

Workaround

Do not use FQ and PQ with the same number. If FQ(N) is used, do not use the PQ(N). As an example: if FQ(1), FQ(2) and FQ(5) are used, do not use PQ(1), PQ(2) and PQ(5). On the other hand, PQ(8) up to PQ(31) can be used without any restrictions.

2.3 [CANXL_AI.002] Wrong Tx descriptor fetch after TX-scan in specific condition

Description

Definition:

- PQ(Z) defines the TX Priority Queue slot Z
- FQ(N) defines the TX FIFO Queue N
- TXDESCN[i] defines the current TX Descriptor at the index i in the FQ(N)
- TXDESCM[k] defines the current TX Descriptor at the index k in the FQ(M)
- TX-Scan is the internal TX message arbitration. Two internal buffers are used to store the two TX messages with the highest priority

Here are the application assumptions:

- The FQ(N) is running concurrently with either a FQ(M) or a PQ(Z) or both
- Either the TXDESCM[k] or PQ(Z) is having higher priority against the TXDESCN[i]
- The FQ(N) has only one valid TX descriptor TXDESCN[i] left to execute, the TXDESCN[i+1] is invalid at that time

- The TXDESCN[i] is selected by the TX-Scan for transmission, leading to the upload of the TXDESCN[i+1] (which is invalid) in the local memory
- Due to the CAN bus arbitration, or a CAN bus OFF or an error on physical link, the TXDESCN[i] is not transmitted

The FQ(M) with a TXDESCM[k] valid or a PQ(Z) is started. The TXDESCN[i] currently selected by the TX-Scan is preempted by the TXDESCM[k] or PQ(Z). When the TXDESCM[k] or PQ(Z) message is sent, the FQ(N) gets the priority back and sends the message defined in TXDESCN[i]. At that time, an internal status to define the next descriptor, is wrongly updated. As the TXDESCN[i+1] in local memory is invalid, the FQ(N) goes immediately on hold. When the TXDESCN[i+1] is set valid, the software can restart the FQ(N). It turns out that the TXDESCN[i+2] is fetched in place of the TXDESCN[i+1].

Scope

The problem occurs for a FQ(N), if the following conditions are met:

- TXDESCN[i] from FQ(N) is selected by the TX-Scan and provided to the CAN bus, but not transmitted due to CAN bus arbitration or a CAN bus OFF or error on physical link
- TXDESCN[i+1] is invalid at the time TXDESCN[i] has been selected for transmission. As a result, the invalid TXDESCN[i+1] is uploaded to the local memory and will not be updated without a restart
- TXDESCM[k] from FQ(M) or a PQ(Z) is having higher priority over the TXDESC[i]

Effects

We have to consider two scenarios when the FQ(N) restarts:

1. TXDESCN[i+1] and TXDESCN[i+2] are valid: When the FQ(N) restarts, the rolling counter value defined in the TXDESCN[i+2] does not match the one expected from TXDESCN[i+1]. This mismatch leads to a safety error and an interrupt (DECERR) can be triggered, if enabled
2. TXDESCN[i+1] is valid and TXDESCN[i+2] is invalid: When the FQ(N) restarts, the FQ(N) is set right away to on hold (TXDESCN[i+2] is invalid), while a valid message is defined in TXDESCN[i+1]. The message in TXDESCN[i+1] is not considered for transmission. No error is generated as there are no checks done for an invalid descriptor. Nevertheless, a functional interrupt can be generated for an invalid descriptor detection

Workaround

Whatever the strategy for managing the FQ(N) (interrupts or polling), the following workaround is valid: If the FQ(N) is on hold and the FQ(N) current address pointer is pointing to a valid TX descriptor TXDESCN[i], restart the FQ(N). If the safety mechanism is enabled, and a safety error triggers, check safety and status registers to identify a TX_DESC_REQ_ERR and verify that the TXDESCN[i+1] and TXDESCN[i+2] are valid.

Despite the occurrence of such issue, the TXFQN is still working properly.

2.4 [CCU_TC.011] Arbitrary SRI clock security error alarm

Description

The security clock monitor is mentioned in the feature list of the CLOCK chapter in the user manual. This monitor is supposed to send an alarm named "SRI Clock Security Error" to the SMU_CS when the f_{SRI} clock becomes higher than its design limit.

The defect is that this monitor can activate an alarm even when f_{SRI} never exceeds the allowed maximum frequency.

***Note**: For alarm index mapping, please refer to the device specific alarm mapping tables in the SMU chapter "Alarm mapping table for ALMCS*".*

Scope

SMU_CS

Effects

The alarm "SRI Clock Security Error" is activated in SMU_CS although f_{SRI} never left the allowed frequency range. Its occurrence is unpredictable. It can be already set at application start after any reset, or occur during application run-time.

Workaround

The generated alarm shall be ignored. That is, no SMU_CS error reaction will be configured.

2.5 [CCU_TC.012] Sporadic CCUSTAT.LCK after writing SYSPLLCON1 or PERPLLCON1

Description

A clock register lock mechanism was implemented for TC4xx. As the clocking system contains several registers that are updated by a posting mechanism, it is necessary to maintain data coherency using a "lock" while the posted write transaction is ongoing and the register write takes effect at the final destination. This bit in the CCUSTAT register indicates if the posting mechanism is active, that means ongoing and not finished yet. While it is set, write access to the respective registers will not be executed without further notification to the application.

One clock output of the PLL (f_{PLL1} or f_{PLL2} or f_{PLL3}) may not be working (no clock) or having wrong clock frequencies (wrong Kx divider setting). This issue may occur in the initialization phase or reconfiguration phase of the clock settings. The Kx-DIV will never release the CCU lock again and all clock configuration registers are locked and can not be accessed anymore.

The issue can be detected by polling the CCU lock bit at least for 100 µs. When in this time the clock is not recovering, a system reset needs to be executed to recover from the issue.

The issue can occur on any of the PLL clock outputs.

Scope

All PLL output paths

Effects

The wrong frequency (wrong K-Divider) or dead clock at one of the PLL output paths, and the lock of the clock configuration register can only be restored by a system reset.

Workaround

To avoid this potential issue, the following configuration sequence for the K-dividers must be executed always for the configuration or re-configuration of any PLL.

PLL configuration phase:

1. PLL off
2. Check for CCU lock: CLOCK_CCUSTAT.B.LCK == 0 with a timeout of at least 100 µs
3. Configure K-dividers in xPLLCON1 register
   - **a.** Set KxPREDIV = div by 2 (xPLLCON1.KxPREDIV = div by 2)
   - **b.** Check for CCU lock: CLOCK_CCUSTAT.B.LCK == 0 with a timeout of at least 100 µs (only for PerPLL)
   - **c.** Set KxDIV = 0x01 (xPLLCON1.KxDIV = 0x01)
   - **d.** Check for CCU lock: CLOCK_CCUSTAT.B.LCK == 0 with a timeout of at least 100 µs (only for PerPLL)

e. Set other PLL parameters as described in the user manual
f. Check for CCU lock: CLOCK_CCUSTAT.B.LCK == 0 with a timeout of at least 100 µs (only for PerPLL)

4. PLL on (power PLL)
5. Check for CCU lock: CLOCK_CCUSTAT.B.LCK == 0 with a timeout of at least 100 µs
6. Check if PLL is powered: xPLLSTAT.PWRSTAT == 1 with a timeout at least 100 µs
7. Wait for 1ms to avoid jitter
8. Check for PLL lock: xPLLSTAT.PLLLOCK == 1 with a timeout at least 100 µs
9. Configure KxPREDIV (if necessary)
   a. Set KxPREDIV to final value
10. Wait for 10µs and check for CCU lock: CLOCK_CCUSTAT.B.LCK == 0 with a timeout of at least 100 µs
11. Configure KxDIV
    a. Set KxDIV to final value
    b. Check for CCU lock: CLOCK_CCUSTAT.B.LCK == 0 with a timeout of at least 100 µs

Changing the KxDIV value during run time (KxPREDIV is not allowed to change during run time)

1. Check for PLL lock: xPLLSTAT.PLLLOCK == 1 with a timeout at least 100 µs
2. Check for CCU lock: xLOCK_CCUSTAT.B.LCK == 0 with a timeout at least 100 µs
3. Configure K-dividers in xPLLCON1 register
   a. KxDIV = 0x01 (xPLLCON1.KxDIV = 0x01)
4. Wait for 10µs and check for CCU lock: CLOCK_CCUSTAT.B.LCK == 0 with a timeout of at least 100 µs
5. Configure KxDIV
   a. Set KxDIV to final value
   b. Check for CCU lock: CLOCK_CCUSTAT.B.LCK == 0 with a timeout of at least 100 µs

2.6 [CCU_TC.013] Functional problem of amplitude regulation in OSC module

Description

The amplitude regulation for the internal crystal (main) oscillator is implemented as a peak detection. Once the XTAL1 pin voltage reaches the threshold the regulator is activated and will decrease the oscillator gain permanently (till the next oscillator power-down/power-up cycle). Therefore, the amplitude regulation is defeatured in TC4Dx.

Scope

The maximum peak-to-peak amplitude on pin is affected.

Effects

During the amplitude regulation operation, once the maximum amplitude threshold is reached, the event is latched and will stay active for the entire power-up cycle, even if the amplitude would drop below specified minimum limit.

Further amplitude reduction over time might be caused by temperature change.

The possibility of unwanted peak-detection events by external disturbances cannot be totally ruled out (EMC related issue).

Workaround

With the bit-field GAINSEL in the OSCCON register the amplitude can be manually configured. The default value is maximum gain (GAINSEL=11). Based on the crystal matching report, the amplitude can be reduced with two intermediate gain configurations (GAINSEL=10/01) or reduced to the minimum gain configuration (GAINSEL=00).

2.7 [CCU_TC.015] MPLL accumulated jitter violation

Description

SYSPLL and PerPLL show high accumulated RMS jitter. The distribution shows an oscillation (violating the limit: 100 ps), before settling to a lower value.

Scope

SYSPLL and PerPLL

Effects

Violation of internal spec limit. No performance issue. Accumulated peak jitter limits are not affected.

Workaround

No workaround - no impact on technical performance.

2.8 [CCU_TC.016] Amplitude regulation in kHz XTAL might be unstable

Description

The amplitude regulation of the kHz OSC might be unstable at low temperatures (-40°C).

Scope

Real Time Clock (RTC) if the external crystal 32 kHz is used (RTC_CON0.RTCCLKSEL = 1).

Effects

At lower temperatures (-40°C) the amplitude regulation might be unstable.

Workaround

Please do not activate the amplitude regulation via the RTC_CON0.REGEN bit (REGEN=0). Per default, the amplitude regulation is deactivated.

2.9 [CCU_TC.017] Unexpected alarm when activating KDIV monitors

Description

The KDIV monitors of the System PLL and Peripheral PLL can raise false alarm after enabling the monitor.

Scope

All devices are affected. Depends on clock configuration of System PLL or Peripheral PLL.
The issue occurs in:

- System PLL: K2-PREDIV is 1.2 or 1.6 or K3-PREDIV is not equal to 2.0
- Peripheral PLL: K2/K4-PREDIV is 1.2 or 1.6 or K3-PREDIV is not equal to 2.0

Effects

KDIV monitor raises sporadic false alarms directly after monitor enable when the clock was started.

Workaround

If the PPU is not used, the KDIV-3 Monitor of the System PLL should be disabled.

2 Functional deviations

The following sequence should be always included for every monitor enable. It is reducing the probability of the failure. With every loop the issue probability is reduced significantly.

After start-up of the PLL:

1. Activate System PLL and Peripheral PLL KDIV Monitors
2. Wait 1µs
   a. Deactivate System PLL monitors if affected
      1. Clear alarm in STDBY SMU for System PLL
      2. Set restart_mon = 1
   b. Deactivate affected Peripheral PLL monitors if affected
      1. Step through all three monitors and only deactivate affected monitor (see details in MONSTAT register below)
      2. Clear alarm in SMU for Peripheral PLL
      3. Set restart_mon = 1
   c. If restart_mon = 1:
      1. Wait 1us
      2. Activate System PLL and Peripheral PLL KDIV Monitors
      3. Increment retry_counter
      4. step_counter = 0
      5. restart_mon = 0
   d. Increment step_counter
   e. Wait 1us
   f. Go to a. if step_counter <= 100 and retry_counter < MAX_RETRIES (Infineon recommends MAX_RETRIES = 8)
3. If retry_counter is above the maximum defined loops: Real KDIV issue assumed. Raise alarm flag of KDIV Monitor

The following settings needs to be avoided for both PLLs and under all circumstances:

- K3-PREDIV = 1.0 (higher KDIV-3 Monitor failure rate)
- It is recommended to enable the KDIV-Monitors on backup-clock

The following clock setting is recommended for maximum performance mode with a minimal issue probability:

Table 5 System PLL configuration

System PLLCON1	K3-PREDIV	K3-DIV	K2-PREDIV	K2-DIV	f_{DCO} [MHz]	f_{PLL0} [MHz]	f_{PPU} [MHz]
1101_H	1.1	2	1.0	2	1000	500	454.55

Table 6 Peripheral PLL configuration

Peripheral PLLCON 1	K2-PREDIV	K2-DIV	K3-PREDIV	K3-DIV	K4-PREDIV	K4-DIV	f_{DCO} [MHz]	f_{PLL1} [MHz]	f_{PLL2} [MHz]	f_{PLL3} [MHz]
$03A104_H$	1	5	2	2	1	4	800	160	200	200

Table 7 MONSTAT register

Field	Bits	Type	Description
KDIV1	8	rh	PER_PLL1 K-divider monitor status 0 … no alarm 1 … alarm

(table continues...)

Table 7 (continued) MONSTAT register

Field	Bits	Type	Description
KDIV2	9	rh	PER_PLL2 K-divider monitor status 0 … no alarm 1 … alarm
KDIV3	10	rh	PER_PLL3 K-divider monitor status 0 … no alarm 1 … alarm

***Note**: The mentioned bit-fields in the MONSTAT register above will be made visible in the next user manual update.*

2.10 [CCU_TC.018] fTPB to feGTM relation in divider ratio table to be changed to fTBP to feGTM_half relation

Description

The table "Divider ratio restrictions" in the CLOCK chapter is wrongly showcasing the relation between fTPB and feGTM to be >=1.
This relation of frequencies is instead targeting fTPB and feGTMhalf.

***Note**: feGTMhalf is always half the frequency of feGTM*

The correct row of the table is shown below

Table 8 Divider ratio restrictions

Clock A	Clock B	Allowed values	Recommended value for 400MHz use-case	Recommended value for 500MHz use-case
fTPB	feGTMhalf	n>=1	1	1

Scope

Divider ratio restrictions.

Effects

The wrong relation currently in the user manual is restricting the feGTM to the maximum of fTPB which is 250 MHz.

Workaround

Apply the shown clock ratio of fTPB:feGTMhalf instead of fTPB:feGTM.

2.11 [CCU_TC.020] Short pulse on output clock when changing KDIV

Description

Changing K-DIVS on the fly could cause a short pulse.

Scope

System PLL and Peripheral PLL output signals are affected.

Effects

Short pulse will be distributed to the IPs. Clear effect cannot be judged.

Workaround

Changing K-DIVIDER on the fly must be avoided at all circumstances.
Therefore, frequency throttling for start-up behavior must be avoided.
Please use SystemPLL to start-up the ramp oscillator.
Please use PeripheralPLL to start-up direct frequency step to the target frequency.

2.12 [CCU_TC.021] Mismatch in divider value information and register description

Description

CCU system clock options table shows incorrect value for FSI2DIV. The table is corrected as follows:

Table 9 CCU system clock options

CCU clock output	Clock source				
	f_{BACK} (system reset) 1)	f_{PLL0} 2)	f_{BACK} 3)	f_{RAMP} 4)	Off 5)
f_{FSI2}	÷ FSI2DIV=1	÷ SRIDIV FSI2DIV=1	÷ SRIDIV FSI2DIV=1	÷ SRIDIV FSI2DIV=1	

1) Set by system reset, CLKSELS = 01_B
2) Set by SW at anytime, CLKSELS = 00_B
3) Set by HW due to clock emergency switch, set by SW at anytime, CLKSELS = 01_B
4) Set by SW at anytime, set by HW during application reset or clock emergency switch execution, CLKSELS = 10_B
5) Set by SW through respective divider or clock selection setting

Scope

CLOCK_SYSCCUCON0

Effects

There is a mismatch between the table CCU system clock options clock output f_{FSI2}, and the documented FSI2DIV After Boot-FW Value in the register description of CLOCK_SYSCCUCON0 in chapter System CCU clock control register 0 in table Reset values of CLOCK_SYSCCUCON0.

Workaround

No workaround, only correction to the table.

2.13 [CCU_TC.022] Update divider ratio table fFSI/fSPB

Description

The divider ratio restrictions table has an incorrect entry. The table is corrected as follows:

Table 10 Divider ratio restrictions

Clock A	Clock B	Allowed values	Recommended value for 400 MHz use-case[1)]	Recommended value for 500 MHz use-case[2)]
f_{FSI}	f_{SPB}	n ≥ 1	n = 2	n = 2

1) The recommended defaults apply for a system source clock frequency of $f_{source0}$ = 400 MHz only.
2) The recommended defaults apply for a system source clock frequency of $f_{source0}$ = 500 MHz only.

Scope

CLOCK_SYSCCUCON0.

Effects

The divider ratio restrictions table has an incorrect entry.

Workaround

No workaround, only correction to the table.

2.14 [CCU_TC.023] Inconsistency between SYSPLLCON1 register description and SYSPLL functional description

Description

There is an inconsistency between CLOCK_SYSPLLCON1 register description and the Functional description chapter of the SYSPLL. K2PRE is added to the figure with the recommendation to only use the K2PRE = 1.0 pre-divider value. The description is corrected as follows:

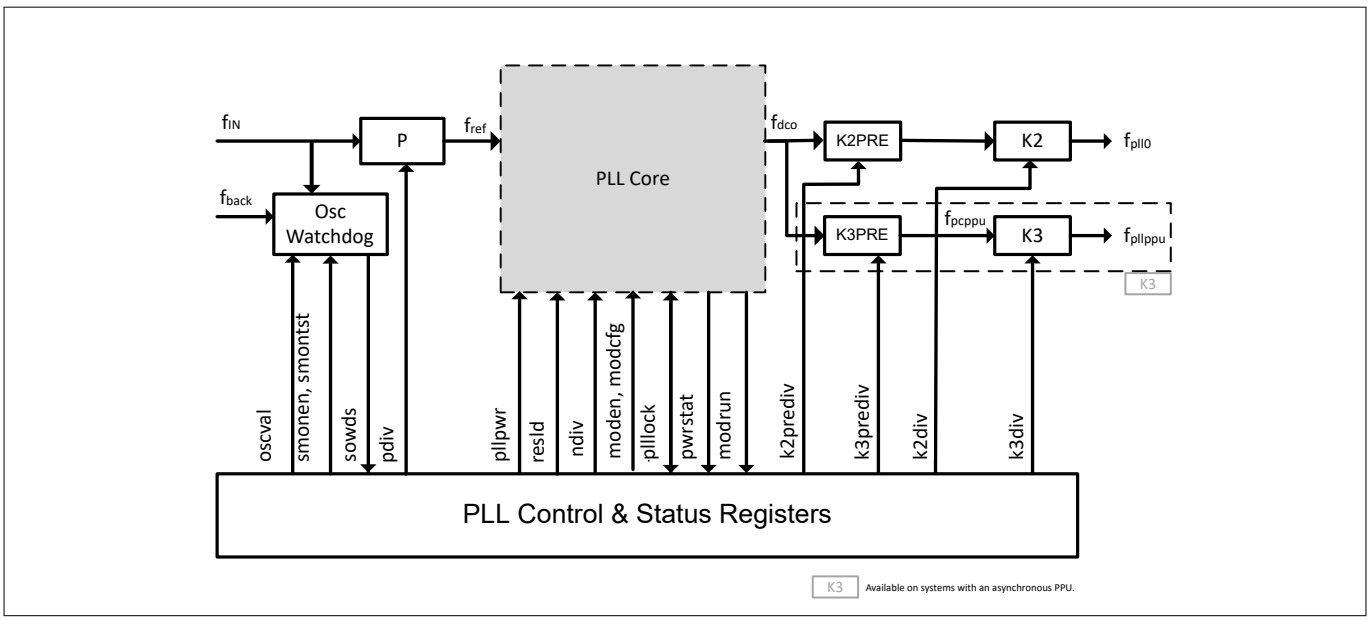


Figure 1 System PLL block diagram

Scope

CLOCK_SYSPLLCON1.

Effects

There is an inconsistency between CLOCK_SYSPLLCON1 register description and the Functional description chapter of the SYSPLL.

Workaround

No workaround, only correction to the description.

2.15 [CCU_TC.025] Allowed divider ratios between f_{SRICS} and f_{FSI} is missing

Description

The Divider ratio restrictions table is missing a line for Clock A = f_{SRICS} and Clock B = f_{FSI}. The table is corrected as follows:

Table 11 Divider ratio restrictions

Clock A	Clock B	Allowed values	Recommended value for 400 MHz use-case[1)]	Recommended value for 500 MHz use-case[2)]
f_{SRICS}	f_{FSI}	n = 1, 2, 3,4,5[3) 4)]	n = 4	n = 5

1) The recommended defaults apply for a system source clock frequency of $f_{source0}$ = 400 MHz only.
2) The recommended defaults apply for a system source clock frequency of $f_{source0}$ = 500 MHz only.
3) Note that f_{SRICS}/f_{FSI}=1 is not supported by all devices. Please refer to the device specific chapters for further restrictions.
4) Do not apply values that would lower f_{FSI} below its reset value which is 100 MHz. This is needed to guarantee a graceful shutdown operation.

Scope

CLOCK_SYSCCUCON0.

Effects

The Divider ratio restrictions table is missing a line for Clock A = f_{SRICS} and Clock B = f_{FSI}.

Workaround

No workaround, only correction to the table.

2.16 [CCU_TC.026] UM correction for CPB bitfield

Description

The description of CLOCK_SYSCCUCON0.CPBDIV is missing a note with respect to the impact of setting CPBDIV to 0. The description is corrected by adding a note that setting CLOCK_SYSCCUCON0.CPBDIV to 0 can lead to a bus error.

Scope

CLOCK_SYSCCUON0.

Effects

The description of CLOCK_SYSCCUCON0.CPBDIV is missing a note with respect to the impact of setting CPBDIV to 0.

Workaround

No workaround, only correction to the description.

2.17 [CCU_TC.028] Remove SUP mode from TC4xx UM Clock part

Description

The Crystal oscillator control register description incorrectly shows the bitfield SUPM. The bitfield SUPM is reserved and must not be used. This reference is now removed from the register description and replaced with bitfield RSVD11.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		CLKSELPHY2	CLKSELPHY0	0	RSVD26	INSEL		0		RSVD21	RSVD20	X2CAP3EN	X2CAP2EN	X2CAP1EN	X2CAP0EN
r		rw	rw	r	rw	rw		r		rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
X1CAP3EN	X1CAP2EN	X1CAP1EN	X1CAP0EN	RSVD11	MODE			HYSCTL		HYSEN	RSVD4	RSVD2		GAINSEL	
rw	rw	rw	rw	rw	rw			rw		rw	rw	rw		rw	

Scope

CLOCK_OSCCON.

Effects

The Crystal oscillator control register description incorrectly shows the bitfield SUPM.

Workaround

No workaround, the incorrect reference is removed form the register description.

2.18 [CPU_TC.134] FPU conversion instructions assert invalid flag before rounding

Description

FPU conversion instructions (from floating-point to unsigned integer format) raise the invalid flag incorrectly prior to rounding, which deviates from the IEEE 754-2019 standard.

Expected behavior

According to the standard, the invalid flag should only be raised if the rounded result is outside the range of the target.

Observed behavior

When converting from floating-point to unsigned integer (FTOU, FTOUZ, DFTOU or DFTOUZ) or unsigned long (DFTOUL or DFTOULZ), all negative numbers are forced to zero and the invalid flag is set regardless of the rounding settings.

Workaround

When trapping on invalid is configured (FPU_SYNC_TRAP_CON.FIE==1), TriCore™ raises a coprocessor synchronous trap (CSE) when an invalid condition is detected. The trap handler should check that the FI condition is due to a relevant conversion instruction by checking the value in FPU_SYN_TRAP_OPC. The trap handler should then check the rounding mode and input operand value in order to detect whether the conversion involved a negative value that would round to 0 or not.

If the rounded value would be 0, then emulation of the instruction (by placing 0 in the destination GPR, resetting FPU_SYNC_TRAP_CON.TST, and returning to the instruction after the FPU conversion operation) should occur. If the rounded result value would not be zero then the original invalid operation processing should proceed.

When trapping on invalid is not configured (FPU_SYNC_TRAP_CON.FIE==0), no traps are raised and PSW.FI is always set when an invalid condition is detected. For a conversion involving a negative value that rounds to 0, PSW.FI will be set when it should not.

In this case there is no good workaround as the detection of an invalid condition can only be done in software.

2.19 [CPU_TC.144] Retriggering of CFI lockstep self-test fails

Description

The TC1.8 CPU NVM interface lockstep incorporates a hardware-based self-test. The self-test function of the core NVM interface (triggered by setting CFILCLTEST.LCLT=1_B) provides the application software with the ability to generate a lockstep alarm.

If the application software requires to test the lockstep logic multiple times during the same power-on cycle, setting the LCLT bit-field multiple times will not result in additional tests or alarms, after the initial usage. While the bit-field (write only with automatic hardware clearing) reads back a 0_B, the internal state machine is not correctly reset after an initial test.

Scope

This problem is limited to multiple testing of the CFI lockstep alarm in between power-on reset.

Effects

No further CFI lockstep test, or lockstep alarm is possible without an explicit clear of the associated lockstep test register.

Workaround

After triggering the hardware-based self-test function, the application software must write 0 to the lockstep test register (CFILCLTEST.LCLT=0_B). This ensures that any subsequent test will occur.

2.20 [CPU_TC.146] Potential performance issue when quad-word (128-bit) load or store instructions are followed by loop instruction (LOOP, LOOP.U)

Description

A quad-word (128-bit) load (LD.DD) or store (ST.DD) instruction immediately followed by a loop instruction (LOOP, LOOP.U) can result in non-optimal performance depending on the memories addressed. One or two stall cycles are inserted between a quad-word (128-bit) load or store and the subsequent loop instruction. This is not the case when smaller size load or store operations are immediately followed by a loop instruction. Depending on the memories accessed this may or may not increase the cycles per loop iteration by up to one or two cycles. If the iteration cycles are increased then maximum bandwidth will be reduced.

In the following two examples, used to copy data between two memory locations, a[4] contains the address of the source memory location, a[5] contains the address of the destination memory, and a[2] contains the number of iterations minus 1. In Example 1, which uses quad-word (128-bit) instructions, the loop instruction stalls for one or two additional cycles per iteration compared to an equivalent copy using only double-word (64-bit) load and store operations as used in Example 2.

Example 1: Copy based on quad-word operations

```
.label128
    ld.dd   e0/e2, [a4+]16
    st.dd   [a5+]16, e0/e2
    loop    a2, .label128
```

Example 2: Copy based on double-word operations

```
.label64
    ld.d    e0, [a4+]8
    st.d    [a5+]8, e0
    loop    a2, .label64
```

If the source and destination memories are local or cached remote memories, the additional stall cycles increase the loop iteration time accordingly. However, if the destination memory is anything else (non-cached remote DSPR, remote DLMU or LMU) then some or all of the stalls before the LOOP instruction will not add to the number of cycles per loop iteration. In addition, if the source memory is not local and is non-cached, while the stalls before the LOOP instruction do add to the loop iteration count, the copy bandwidth is still much higher than the 64-bit version.

Scope

The erratum is limited to quad-word (128-bit) load or store instructions immediately followed by a loop instruction (LOOP, LOOP.U), used for targeting local, or cached memory locations for both source and destination.

Effects

Loops using quad-word (128-bit) load or store can have a longer execution time than equivalent ones using double-word (64-bit) load or stores.

Workaround

Only when targeting local or cacheable memory locations, within loops, for example implementing memcpy(), loops using double-word (64-bit) load or stores are recommended over quad-word (128-bit) ones.

Loops using quad word (128-bit) load or store should be used to target memories where the benefit of the quad-word accesses outweigh the small number of stall cycles. This is the case when accessing a non-local memory with non-cached naturally aligned accesses. This is because a single 128-bit external transfer is significantly faster than two 64-bit ones.

2.21 [CPU_TC.147] CPUCS reset values when CSRM debug enabled

Description

When CSRM debugging is allowed, the CPUCS ACCEN and CPUCS_CFICS ACCEN registers are updated by the CSRM_FW to grant Cerberus read and write access to all the instance resources. This difference under CSRM debug is not documented in the user manual.

The following table shows the correct reset values of the ACCEN registers visible to the application software when CSRM debugging is allowed:

Table 12 CPUCS and CPU_CFICS after Boot-FW reset values when CSRM debug is allowed

Register name	Reset value
CPU_CFICS_ACCENPFSFRCFG_WRB	00004000_H
CPU_CFICS_ACCENPFSFRCFG_RDA	10003000_H
CPU_CFICS_ACCENPFSFRCFG_RDB	00004000_H
CPU_CFICS_ACCENPFSFR_WRB	00004000_H
CPU_CFICS_ACCENPFSFR_RDA	10003000_H
CPU_CFICS_ACCENPFSFR_RDB	00004000_H
CPUCS_ACCENSPRCFG_WRA	10003000_H
CPUCS_ACCENSPRCFG_WRB	00004000_H
CPUCS_ACCENSPRCFG_RDA	10003000_H
CPUCS_ACCENSFRCFG_WRA	10003000_H
CPUCS_ACCENSFRCFG_WRB	00004000_H
CPUCS_ACCENSFRCFG_RDA	10003000_H
CPUCS_ACCENSFRCFG_RDB	00004000_H
CPUCS_ACCENSPR0_WRA	10003000_H
CPUCS_ACCENSPR0_WRB	00004000_H
CPUCS_ACCENSPR0_RDA	10003000_H
CPUCS_ACCENSFR_WRA	10003000_H
CPUCS_ACCENSFR_WRB	00004000_H
CPUCS_ACCENSFR_RDA	10003000_H
CPUCS_ACCENSFR_RDB	00004000_H

Scope

The erratum is limited to the CPUCS instance when CSRM debugging is enabled.

Effects

Cerberus is granted read and write access to CPUCS and CPU_CFICS resources when CSRM debugging is enabled.

Workaround

No workaround.

2.22 [CPU_TC.148] No APU protection on internal path from CPU to local scratchpad RAMs

Description

The TC1.8 CPU implements several access protection units (APU), to provide protection against illegal or unintended bus accesses to the local memory system and control registers. The protection does not cover the local accesses to scratchpad RAMs. The CPU block diagram incorrectly shows the following paths as protected, this is incorrect:

- PMI accesses to local PSPR or PCACHE
- DMI accesses to local DSPR or DCACHE

The protected paths are:

- S-SRI0 accesses to PSPR, DSPR, DLMU, SFR, CSFR, or STM
- S-SRI1 (if present) accesses to CFI, or FSFR
- PMI or PCACHE accesses to CFI
- DMI or DCACHE accesses to CFI
- DMI accesses to local DLMU

The following figure must be used instead:

2 Functional deviations

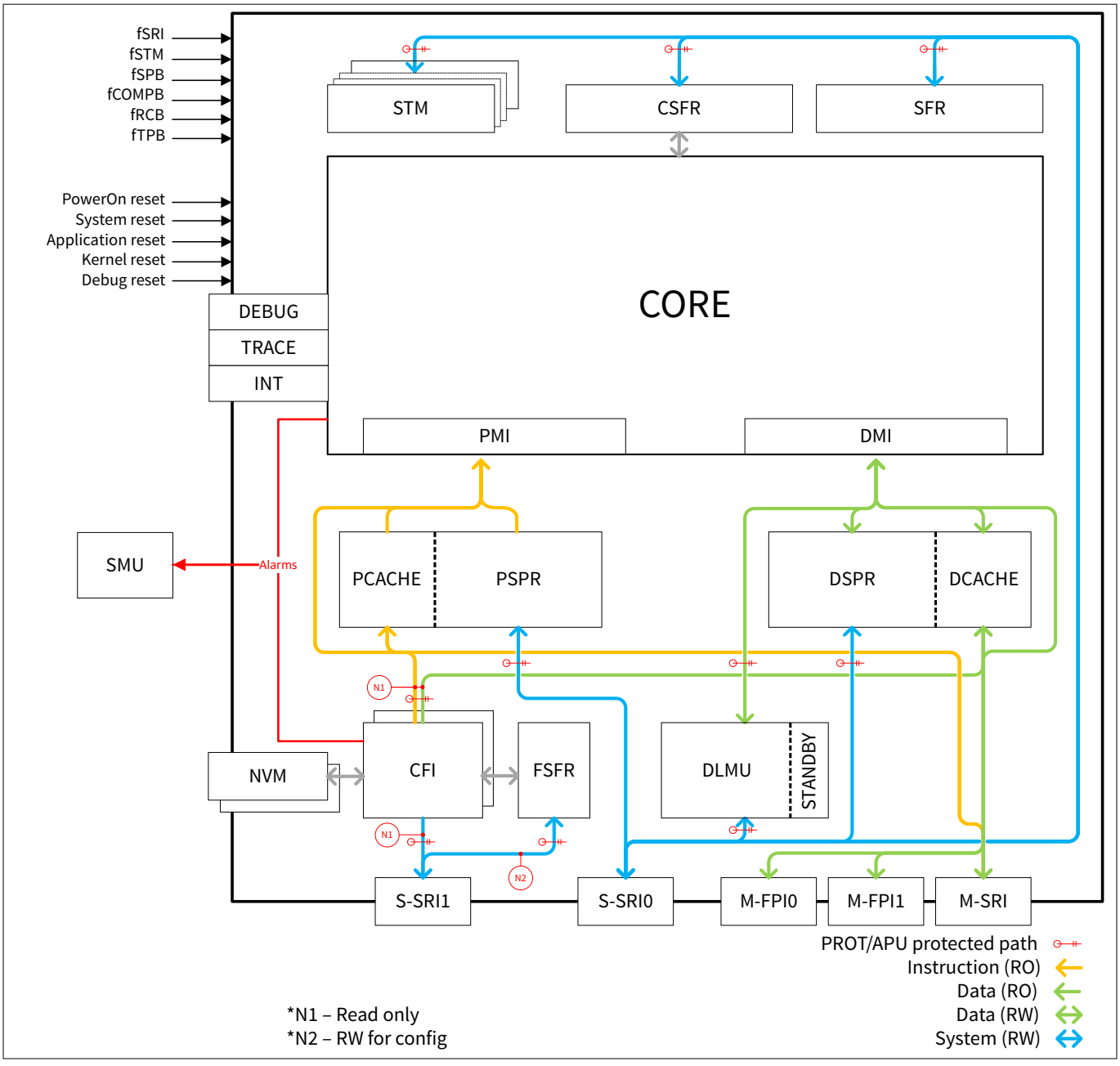


Figure 2 CPU block diagram

Scope

This erratum is limited to the CPU block diagram.

Effects

The diagram shows a protection that does not exist. The APU protection is only applicable to the slave accesses.

Workaround

Access from PMI to local PSPR can be protected using the code memory protection unit.
Accesses from DMI to local DSPR can be protected using the data memory protection unit.

2.23 [CPU_TC.149] Incorrect reference to PSW.CDC in DBGSR description

Description

The debug status register (HRA_DBGSR) provides information related to the DEBUG function of the CPU. The description of several bit-fields (DE and HALT) incorrectly refers to the PSW.CDC (Program status word core depth counter) when the intention was to refer to CDC (Core debug controller). In this register description CDC always stands for core debug controller and there is no link between the DBGSR and the PSW.

Table 13 DE and HALT bit-field definition

Field	Bits	Type	Description
DE	0	rh	**Debug enable** Determines whether the core debug controller is enabled or not. 0_B The core debug controller is disabled 1_B The core debug controller is enabled
HALT	2:1	rwh	**CPU halt request / status field** HALT can be set or cleared by software. HALT[0] is the actual Halt bit. HALT[1] is a mask bit to specify whether or not HALT[0] is to be updated on a software write. HALT[1] is always read as 0. HALT[1] must be set to 1 in order to update HALT[0] by software (R: read; W: write). 00_B R: CPU running. W: HALT[0] unchanged. 01_B R: CPU halted. W: HALT[0] unchanged. 10_B R: Not Applicable. W: reset HALT[0]. 11_B R: Not Applicable. W: If DBGSR.DE == 1 (The core debug controller is enabled), set HALT[0]. If DBGSR.DE == 0 (The core debug controller is not enabled), HALT[0] is left unchanged.

Scope

Description of the HRA_DBGSR register.

Effects

The description of several bit-fields incorrectly refers to the PSW.CDC instead of the core debug controller.

Workaround

None

2.24 [DAP_TC.012] Timeout counters when MWBE = 1, waiting time greater than programmed

Description

The timeout feature of Debug module waits longer than the programmed value in DAPISC register fields in specific conditions.

Scope

Debug functionality

Effects

During the wait for the start bit of reply telegram, customers may give up before actual number of clock cycles are expired when DAPISC.MW8E is enabled.

Workaround

This issue can be fixed by updating the tool software to wait for 15 clock cycles more than the programmed value in DAPISC.MAXWAIT8 when DAPISC.MW8E bit is set to 1.

2.25 [DAP_TC.013] Update table DAP/JTAG interface pins' on-chip pull device setting

Description

Debug and Trace section of user manual mentions about the on-chip pull device settings which are not in sync with the device datasheet.

Scope

Debug pins pull device settings.

Effects

If the user refers to debug section of the UM for pull device settings, they may end-up in incorrect pull device settings for debug pins.

Workaround

Users must refer to the device datasheet for the actual pull device settings of debug pins.

2.26 [DMA_TC.069] Spurious DMA lockstep alarms

Description

The DMA can generate spurious lockstep alarms in the presence of errored write responses.

The spurious DMA lockstep alarms can only be triggered shortly after an errored DMA write move on the SRI interconnect. DMA writes on the SRI interconnect can be errored due to systematic software faults which should be eliminated by the system integrator, or due to transient or permanent faults in the MCU.

Scope

DMA

Effects

If the DMA receives a write error response followed by a valid write response within the next three SRI clock cycles from the SRI interconnect, the DMA master and checker cores will no longer be in lockstep. The DMA reports a lockstep alarm to the SMU as a result.

Workaround

There is no workaround to prevent the spurious DMA lockstep alarm. As there should be no systematic write errors, these alarms can only occur due to a transient or permanent fault. The DMA functionality can only be recovered by applying an MCU application reset.

2.27 [DMA_TC.070] DMA resource partition overflow error status is unreliable

Description

The DMA is specified to raise an overflow error (OVFLER) error interrupt when a new DMA resource partition (RP) error is reported while a previous DMA resource partition error is still pending in the RP error status register ERRSRr.

As a result of a bug in the TC4Dx DMA, the ERRSRr.OVFLER bit does not behave as specified under the following conditions:

- When a daisy-chain transaction request lost error (DCTRLER) occurs due to a misconfigured channel which executes on DMA move engine 1, the ERRSRr.OVFLER bit is incorrectly set even when there is no error overflow
- When two or more RP error events occur on successive SPB clock cycles, the ERRSRr.OVFLER bit is not set as it should be

Scope

System DMA

Effects

The DMA RP overflow error status bit, ERRSRr.OVFLER is not reliable as an indication that multiple DMA RP error interrupts have occurred.

Workaround

Application software should ignore and always clear the DMA RP overflow error status bit, ERRSRr.OVFLER in its error handling interrupt service routine.

2.28 [DRE_TC.001] Pending requests in registers COBL_BPR0/1 cannot be cleared

Description

The pending requests for CAN output buffers in registers COBL_BPR0 and COBL_BPR1 cannot be cleared by software as they are read-only.

Scope

COBL_BPR0 and COBL_BPR1

Effects

Software will not be able to clear the pending request bits in COBL_BPR0 and COBL_BPR1 registers in order to free up the corresponding COBUF for new message. The corresponding COBUF would be processed again in the next arbitration.

Workaround

Software write of 0x3F to COBUFj_UCRH.DID or COBUFj_MCRH.DIDx causes invalid destination error and the frame is dropped. Wait until the pending request is cleared after 64 frames latest.

2.29 [DRE_TC.002] Pending requests in register CIBL_BPR cannot be cleared

Description

The pending requests for CAN input buffers in register CIBL_BPR cannot be cleared by software as they are read-only.

Scope

CIBL_BPR

Effects

Software will not be able to clear the pending request bit in CIBL_BPR register. The corresponding CIBUF would be processed again in the next arbitration.

Workaround

Software write of $3F_H$ to CIBUFj_RHEAD.ETHDID causes invalid destination error and the frame is dropped. Wait until the pending request is cleared after 20 frames latest.

2.30 [DRE_TC.003] Frame loss in multi-cast due to CRE abort sequence on the CAN side on the previous multi-cast transfer

Description

Frame(s) are lost when there are two consecutive Ethernet to CAN multi-cast requests and an SPB bus transaction error ME_ERR.SPBBE (INT_13) happens during the processing of the first multi-cast request.

Scope

Ethernet to CAN multi-cast

Effects

During an SPB bus transaction error scenario, the ongoing processing of multi-cast request 1 is aborted and the request is cleared by the DRE. But while processing the next multi-cast request 2, the DRE starts routing only from the DID index where it stopped while processing the prior multi-cast request 1. The internal index count is not reset after the error.

For example, if SPB bus transaction error occurs while the DRE is processing DID1 of multi-cast request 1, the DID0 of the next multi-cast request 2 is ignored and the DRE starts from DID1.

Workaround

There is an error interrupt triggered by the DRE during an SPB bus transaction error:

- The software shall identify the erroneous multi-cast request 1 on the ME_FESRCA register and read the routing header and ME_ERR.FEDID to determine the index of the error
- The software shall then read the ME_SRCA that indicates the currently being processed request 2 and check if it is multi-cast based on routing header
- If multi-cast, it can copy the CAN frame corresponding to multi-cast request 2 and write to the Tx host buffer 1 of the left out DIDs or abort the sequences of the DIDs by writing to CRE_ABORT_SEQ register in the CRE

Additional detection of missing frame(s) is possible from the CRE:

- The CRE watchdog timer starts when there is a Tx host buffer empty. So if the CRE corresponding to DID0 does not receive a CAN frame from the DRE within the timeout period, there would be a timeout interrupt triggered

2.31 [DRE_TC.005] ETH2CAN - ACF_CAN_ADDR/LE_ACF_CAN_ADDR field not being reset

Description

EIBUFi_STATUS (i=0-5).ACF_CAN_ADDR/LE_ACF_CAN_ADDR bit-fields are reset just after EIBUFi_STATUS (i=0-5).BPR bit is cleared.

Scope

Limited to Ethernet to CAN routing path

Effects

Software cannot read the values of EIBUFi_STATUS (i=0-5).ACF_CAN_ADDR/LE_ACF_CAN_ADDR bit-fields for debugging purpose in case of CFE or RFE errors.

Workaround

There is no workaround. There is no alternative status information that can be used by the software during CFE or RFE errors.

2.32 [DRE_TC.006] Inconsistency between GETH and LETH while triggering error interrupt by the DRE

Description

DMA_CHy_Status (y=0-7) register bits TBU and FBE are both checked to be set by DRE in case of Tx descriptor error. But for LETH, these bits might not be set by the Tx DMA channel.

Scope

LETH Tx DMA channel mapped to DRE.

Effects

The DRE might not trigger an error in case of LETH Tx DMA channel error when only DMA_CHy_Status.FBE is set.

Workaround

Use the LETH error interrupt to STOP the DRE Tx descriptor handler by setting the TETHDLi_CTRL.STOP bit. Otherwise, DRE will wait indefinitely for write-back format.

***Note**: Another possible option is to use the watchdog timer feature of DRE instead of DMA_CHy_Status polling feature. DRE will then trigger a timeout interrupt.*

2.33 [DRE_TC.008] Forwarding ID (FID) wrongly assembled for LETH

Description

DRE uses Forwarding ID (FID) which is configured by user as part of Forwarding Table configuration (FTCFG) to identify the destination Ethernet port for Ethernet to Ethernet forwarding use-case. Due to defect, the DRE assembles wrong status information for FID when a received Ethernet packet is filtered by Frame parser of LETH and ELIRS bit of LETH_Portj_MTL_RXP_Control_Status register is 1. This results in incorrect forwarding decision by DRE.

Scope

The issue is applicable only when:

- LETH - xETH forwarding use case is enabled by DRE configuration and
- Frame Parser is used for filtering the corresponding received packet in LETH and
- ELIRS bit of LETH_Portj_MTL_RXP_Control_Status register is 1

Effects

Based on the Forwarding Table configuration (FTCFG), incorrect Ethernet destination might be selected or no match for the forward packet occurs in DRE.

Workaround

ELIRS bit of LETH_Portj_MTL_RXP_Control_Status register to be set to 0 and the FID of the Forwarding Table configuration (FTCFG) must be constructed only based on the EIF, DMACH and MADRM for packets received from LETH.

2.34 [eGTM_AI.530] GTM_AEI: AEI_READY erroneously set

Description

There is a combinatorial path from the AEI_WDATA[16] input port to the AEI_READY output port.

Therefore the AEI_READY could be set in the same clock cycle as AEI_SEL.

This is forbidden in Standard and Pipelined AEI protocol (protocol is GTM integration dependent).

This possibility only exists when the AEI_ADDR =x"40" (BRIDGE_MODE) and AEI_W1R0 = '1' are set; that is, only when writing '1' to the BRIDGE_MODE.BRG_RST bit-field (requesting a Bridge Reset).

Scope

GTM_AEI

Effects

In this case the combinatorial setting of AEI_READY might lead to a violation of the AEI protocol.

Workaround

Do not use AEI-Bridge Soft-Reset.

2.35 [eGTM_TC.005] Kernel reset corner case causes AXI hang-up

Description

AXI bus may be blocked when kernel reset is performed.

Scope

eGTM AXI slave

Effects

AXI interface may be blocked in case a kernel reset and a register access start at the same time within the internal slave logic.

Workaround

The eGTM kernel reset execution is limited in the following ways:

1. Kernel reset through the module group reset feature

   In this case it has to be either assured that no eGTM register access is running in parallel to the kernel reset execution, or the eGTM has to be taken out of the reset group and reset using the programmable reset (see point 2 below) after or before the group reset to the reset group was or is executed.
2. Programmable kernel reset (through RST_CTRLA + RST_CTRLB)

   When using the programmable reset, the last write to either RST_CTRLA or RST_CTRLB shall be followed by a register read to any register to avoid the issue described above.

   Alternatively, the DSYNC CPU instruction shall be performed after the last write to RST_CTRLA/B, before issuing any new read or write command within the programming sequence.

   For both, no other master shall access the eGTM in parallel.

2.36 [eGTM_TC.006] eGTM wrapper cannot be switched off (CLC) when soft suspend is active

Description

The module cannot be disabled by CLC.EN when soft-suspend is active.

Scope

Debug soft-suspend

Effects

The disable request is not executed in soft-suspend.

Workaround

Release the module from soft-suspend to disable the module.

2.37 [ERAY_TC.001] Wrong reset behavior SSH-ERAY

Description

In case when Debug reset is issued (when OCDS is disabled, then Application or System resets are triggered), then the PORST reset domain of SSH in ERAY module is reset unexpectedly.

Scope

Applicable for SSH configuration of ERAY module.

Effects

The issue leads to lose of ERAY SSH Alarm configuration contents (ALMSRCS) and certain diagnosis registers (ETRR, FAULTSTS, ERRINFO).

Workaround

The following software workaround needs to be implemented:

- Reconfigure ALMRSRCS after Debug-Reset to 0x37 and
- In case diagnosis information is required, store respective register-contents (ETRR, FAULTSTS, ERRINFO) after fault-occurrence and prior to system-restart

2.38 [GETH_AI.029] CBS credit not decremented during the IPG phase of transmission

Description

When Credit Based Shaper (CBS) is enabled for a Traffic Class (TC), the packet available in a TC is scheduled for transmission when zero or positive credit is accumulated for the TC. The credit gets incremented when a packet is available in Transmit Queue and is waiting for opportunity to get scheduled for transmission or when the current credit is negative. The credit gets decremented during the actual packet transfer. The rate at which the credit is incremented or decremented depends on the percentage of total bandwidth reserved for this TC.

As per IEEE 802.1Qav standard, the credit must also be decrement when the packet overheads are transmitted. These overheads include the preamble bytes before the start of packet, CRC/FCS bytes, and the minimum IPG (Inter packet Gap of 12bytes) after the end of packet data transfer.

However due to this defect, the MAC decrements the credit only up to last byte of packet data (last byte of Frame Check Sequence (FCS)) transfer and increments the credit during the subsequent nominal IPG period associated with that packet.

Scope

The CBS algorithm is enabled for scheduling the transmission of Audio/Video traffic.

Effects

As the credit is not decremented but incremented during the trailing IPG period of the packet, the accumulated credit value is higher/larger than expected. As a result, the next packet of that traffic class will get scheduled earlier or faster than expected and the process repeats. Consequently, in each time window for which the bandwidth reservation is measured, the actual bandwidth consumed by the TC is more than the software programmed value.

The percentage of extra bandwidth consumed by the TC depends on the packet length. This can be estimated with the following equation assuming the nominal IPG of 12 bytes.

Additional/extra bandwidth = ((# of packets x 12 Bytes) / (Total number of bytes transmitted in that window including preamble bytes of each packet)) x FractionalBW programmed for this TC. For example, 30% BW is programmed for this TC in which a stream of 100 packets of 128 bytes length is to be transferred. For this example, the additional BW consumed would be equal to = 30% x (100 x 12) / (100 x (8 + 128)) = ~2.65%. Effectively, this TC is allocated 32.65% of total BW instead of the programmed 30%.

Workaround

While programming the parameters that determine the fractional BW for this TC, calculate for a BW fraction target that is lesser the desired one so that the actual BW consumed is closer to the desired one due to the “extra/additional” BW provided by the defective algorithm. The target “lesser” percentage value used for this calculation will be an estimate based on the average length of packets transmitted.

2.39 [GETH_AI.032] Time Aware Shaper (TAS) additional IPG in case of back-to- back packet transmission

Description

When Enhancements to Scheduling Traffic (EST) feature is enabled, the Transmit Scheduler defers next packet scheduling until the current packet is completely forwarded to MAC Transmitter, so that any eligible packet received in higher priority Transmit Queue towards end of current packet transmission is scheduled over already available packet in lower priority Transmit Queue. The Transmit Scheduler design maintains a counter which triggers when packet forwarding to MAC Transmitter starts and increments on every clock cycle until the value (number of clock cycles to transmit the packet including overheads (target count), on the line) determined based on packet size (including the PREAMBLE+SFD overhead) and operating speed is reached.

However, due to the defect, extra IPG (more than programmed minimum IPG) is observed even when back-to-back packets are available for scheduling.

The trigger for counter is generated in f_{GETH} clock domain (packet data read from Transmit Queue and forwarded to the Asynchronous FIFO between f_{GETH} clock domain and MAC Transmitter clock domain) and synchronized to MAC Transmitter clock domain (clock domain in which counter in Transmit Scheduler is incremented, and data read out from Asynchronous FIFO). When the counter reaches the target count, the indication is synchronized back to f_{GETH} clock domain to enable scheduling of next eligible packet. The waiting until current packet is completely forwarded to MAC Transmitter for scheduling the next eligible packet, and the crossing of signals to and from f_{GETH} clock domain and MAC Transmitter clock domain (CDC delays), results in delayed scheduling of next eligible packet, which is reflected in the incorrect extra IPG. This can be in worst case 12 clock cycles of slowest among the two clocks (converted to bit times based on operating speed) based on the frequency and phase relationship of f_{GETH} and MAC Transmitter clocks.

Scope

The Software enables the Enhancements to Scheduling Traffic (EST) feature in MAC to schedule the Time Sensitive Networking (TSN) traffic.

Effects

The extra IPG impacts the number of packets that can be scheduled in Time Interval (TI) or slot/row of EST Gate Control List (GCL). This impacts performance and loss of data due to remote end dropping packets that are received after the expected time.

Workaround

The software must program the TI to accommodate the extra IPG based on the number of packets expected to be scheduled. For example, in 1 Gbps speed mode, extra IPG in worst case is 12 clock cycles and MAC Transmitter clock is slower, therefore, TI must be increased by at least (clock period (8 ns) * extra IPG (12)) that is, 96 ns per packet.

The above mentioned workaround is only to guarantee transmission of expected number of packets within a scheduled slot and there is no workaround to guarantee minimum IPG between transmission of back to back packets in the same time slot.

2.40 [GETH_AI.033] Received packet not routed to the programmed VLAN filter fail queue

Description

When the VLAN filter fail based queuing is enabled, by setting the VFFQE bit of Portj_MAC_RxQ_Ctrl4 register to 1, the received packets that fail the VLAN filter are routed to the Receive Queue specified in the VFFQ field.

However due to the defect, when a received packet fails the VLAN filter, it is not routed to the Receive Queue specified the VFFQ field, which is an incorrect behavior. The MAC receiver routes the packet based on other

queue routing mechanisms enabled. This is because of the premature clearing of the signal indicating VLAN filter fail status.

Scope

The issue is applicable only when VFFQE bit of Portj_MAC_RxQ_Ctrl4 register is set to 1 by the application software.

Effects

The received packet which has failed VLAN filtering is stored in wrong RxQ.

Workaround

The software must check the VLAN filter fail status bit in the Rx status that is provided along with the packet and route it to the respective processing function.

2.41 [GETH_AI.034] Minimum inter packet gap of the MAC transmitter mismatches the programmed non-standard value MII speed modes

Description

The MAC transmitter uses the value programmed in the IFP and the inter-packet-gap (IPG) fields of the Portj_MAC_Tx_Configuration register, and the EIPG field of the Portj_MAC_Extended_Configuration register to determines the minimum IPG between the transmitted packets. The IPG is one of the critical factors which influences the maximum performance or throughput that can be achieved. Therefore, the MAC transmitter attempts to transmit the packets with programmed minimum IPG unless the packets from the software are received at slower rate.

However, due to this defect, when the MAC is operated in MII speed modes, although the software provides back-to-back packets, the IPG between the packets mismatches with the programmed non-zero encoded value (non-standard IPG, that is, non-96-bit times). This is because the programmed IPG value is incorrectly interpreted in terms of 4-bit times, whereas the encoding of programmable IPG fields is in terms of 8-bit times. The minimum IPG (minIPG) in units of MAC transmitter clock cycles, is computed as follows:

- IFP = 1, minIPG = 24 + ({EIPG,IPG}*4)
- IFP = 0, minIPG = 24 - IPG

For example, with IFP=0, EIPG=0, IPG=2, the minIPG is 22, that is, 88-bit times instead of expected 80-bit times. Similarly, with IFP=1, EIPG=0, IPG=2, the minIPG is 32, that is, 128-bit times instead of the expected bit times.

Also, if an odd value is programmed, the minimum IPG is a non-integral multiple of 8-bit times, which is not compliant with the standard.

Scope

The issue is applicable only when GETH is configured for 10 or 100 Mbps MII speed modes and IPG field of Portj_MAC_Tx_Configuration register is configured to a non-zero value.

Effects

Transmission of packets with incorrect Inter Packet Gap.

Workaround

The software must program only the even numbered, and twice the required encoded value for IPG field of Portj_MAC_Tx_Configuration register. Note that this limits the available programmable values for minimum IPG.

2.42 [GETH_AI.035] Receive interrupt watchdog clock-based timer not reset on other timers timeout and receive inter triggering events

Description

When Receive Interrupt (RI) watchdog clock-based timer is enabled (RWT field of the DMA_CHj_Rx_Interrupt_Watchdog_Timer register is programmed to a non-zero value), RI status (RI field of the DMA_CHj_Status register) is set to 1 after the timer expires. If the corresponding interrupt is enabled (RIE field of the DMA_CHj_Interrupt_Enable register is programmed to 1), an interrupt is generated. If any other similar timer expires (bytes or packets-based) or RI triggering events happen (received packet with Interrupt on Completion (IOC) field of Receive Descriptor set to 1 or Id timer expiry) prior to clock-based timer expiry, the clock-based timer is reset. This is done to prevent the generation of redundant RI.

However, due to the defect, the clock-based timer is not reset even if RI is generated due to other similar timer's expiry or RI triggering events.

Scope

The issue is applicable only when Receive Interrupt Watchdog Timer is enabled by application software.

Effects

Redundant interrupts are triggered.

Workaround

The software must not enable multiple RI generation sources (bytes or packets or clock in DMA_CHj_Rx_Interrupt_Watchdog_Timer, descriptor IOC field) to be active simultaneously.

2.43 [GETH_AI.036] MAC incorrectly starts packet transmission before threshold number of bytes are available in the transmit queue

Description

When the transmit queue (TxQ) is operating in the threshold mode, the MTL transmit queue write controller (TWC) passes the pointer value to the MTL transmit queue read controller (TRC) after the threshold number of bytes specified in the TTC field the Portj_MTL_TxQ(#i)_Operation_Mode register are written to the TXQ. The passing of the pointer value is an indication that the TRC can start reading data from TXQ and provide it to MAC transmit protocol engine. From then on, the TWC continuously passes the pointer value when the subsequent words are written to the TXQ. The minimum programmable threshold value of 64 bytes ensures that the packet transmitted is at least of the minimum size specified by the protocol.

However, due to the defect, when the TTC field of the Portj_MTL_TxQ(#i)_Operation_Mode register is programmed to 0, which is encoding for 64 bytes, the TWC passes the pointer value to TRC prematurely after 32 bytes are written to TXQ, instead of 64 bytes. If subsequent words are not available in the TXQ at the required rate, the transmitted packet is unexpectedly truncated.

Scope

The issue is applicable only when TTC field of the Portj_MTL_TxQ(#i)_Operation_Mode register is programmed to 0.

Effects

The issue can potentially cause underflow of the TXQs.

Workaround

The software must program the TTC field to 2, which is encoding for a threshold of 96 bytes.

2.44 [GETH_AI.037] Receive DMA stops operation when packet flush initiation and exit from suspend mode overlap

Description

Expected behavior is, when the Receive Packet Flush functionality is enabled by programming the RPF field of the DMA_CHj_RX_Control register to 1, received packets are flushed when the receive DMA is in STOP or SUSPEND mode.

However due to the defect, if the tail pointer is updated when the receive DMA is in SUSPEND mode coinciding when the MTL provides the header status and the first packet data word to the DMA, the DMA acknowledges the MTL and drops this data as the descriptor is not yet deemed available for DMA. Since DMA has acknowledged the MTL, a flush command is not issued. As a result, when the descriptor becomes available to the receive DMA, the receive DMA does not receive the expected header status and first packet data word from MTL receive queue read controller, as these were provided earlier, acknowledged, and dropped. This causes the receive DMA to stop operation.

As a consequence of this issue, the receive DMA stops operation and the MTL receive queue overflows, resulting in data loss. To recover from this state, the user must program the SWR field of the DMA_Mode register to 1 to issue a software reset.

Scope

The issue is applicable only when RPF field of the DMA_CHj_RX_Control register is set to 1.

Effects

The receive DMA stalls and eventually packets will be lost due to RXQ overflow.

Workaround

The application software must follow the below described sequence to enable Receive Packet Flush in RxQ.

Setting the RPF bit:

1. Wait for Rx Buffer Unavailable Interrupt (RBU) of DMA_CH0_Status[7] for a corresponding DMA (do not clear it)
2. Check if RDAS field DMA_Debug_Status3 is 0 indicating that the Rx DMA is idle or DMA_CHj_Debug_Status.RDFS==3 (suspend state) for the corresponding RX DMA Channel j
3. When the above condition is true, set RPF==1

Resetting the RPF bit:

1. When tail pointer of the corresponding RX DMA channel is ready to be updated, first clear the RPF bit to 0
2. Service the RBU, clear the Rx Buffer Unavailable Interrupt
3. Reprogram the tail pointer

2.45 [GETH_AI.038] Unintended closure of receive descriptor when intermediate descriptor contains definition error

Description

When the DMA encounters a receive descriptor with definition error, it

1. Closes the descriptor with Descriptor Definition Error Indication (value of all-ones in the CTXT, FD, and LD bits of RDES3)
2. Enters the stop state and updates the DDE bit of the DMA_CHj_Status register to 1. This causes an interrupt to be generated

For an intermediate descriptor, the DMA additionally flushes the remaining part of packet from the MTL Receive Queue. The recovery mechanism is to reconfigure and restart the DMA. The DMA starts normal operation thereafter.

However due to the defect when DMA is reconfigured and restarted, it incorrectly closes the first fetched descriptor by setting the LD bit (Last Descriptor) to 1, in RDES3. This is because the last descriptor indicator signal that was set bet the Receive Descriptor Definition Error occurred is not cleared when the DMA is reconfigured and restarted.

Scope

The issue is applicable only when the application uses intermediate descriptors for received packet processing.

Effects

Incorrect closure of the first descriptor leading to application processing incorrect data.

Workaround

Software must ignore the first receive descriptor with LD = 1, that it receives immediately after the DMA is reconfigure and restarted.

2.46 [GETH_AI.039] Transmit packet not terminated when underflow occurs in MII speed modes

Description

The MAC terminates the transmission of a packet when the MTL Transmit Queue (TXQ) underflows. This avoids adverse impact on the performance as the packet is expected to be dropped by the remote node.

However, due to the defect, when underflow occurs in MII speed modes (10/ 100 Mbps), as the next data is not available to MAC Transmit Protocol Engine (TPE) when it is required for transmission of first nibble, the packet is not terminated. Instead the transmission of the previous data word (32-bit) is repeated.

When the repeat transmission of the previous data word is complete:

- If the next valid data is available, the remaining words of the packet are also transmitted. The packet corrupt due to the repeat-transmission of the same data word
- If the next valid data is unavailable, the packet is terminated with an underflow. In this case the impact limited to the repeat transmission of one data word

Scope

The issue is applicable only when operation at 10 or 100 Mbps MII speed modes.

Effects

Potential corruption of transmit packet.

Workaround

The software must operate the TXQ in store-and-forward mode, the packet sizes must not be more than (TXQ SIZE in bytes - ((PBL + 5)*8)), where PBL is the Programmable Burst Length.

2.47 [GETH_AI.040] Rx DMA stall due to incomplete context descriptor closure

Description

Under following conditions the RX DMA channel is stalled

- GETH port is configured for frame duplication to both host (Rx DMA channel) and for forwarding to the other egress port and
- The RXC channel number of the forwarding path is greater than the RXC of the RX DMA Channel and

- The ingress packet is VLAN tagged and
- Receive packet timestamp is enabled

When the Receive DMA channel is stalled no further packets are transferred to the host even when Rx Descriptors are available. This eventually results in Head of line blocking in the corresponding RXQ and hence resulting in loss of packets due to RXQ overflow.

Scope

When VLAN tagged & timestamped ingress packet is duplicated to forwarding path, followed by a duplication to Rx DMA channel, this defect can get manifested.

Effects

The Receive DMA stalls, resulting in data loss.

Workaround

In Bridge mode, the software must ensure that ingress packet is duplicated to the bridge forwarding path only once and after the duplication to all the RxDMA channels are completed. This can be done by mapping lowest RXC as the forwarding channel to egress Transmit queue among the channels to which packet is duplicated.

2.48 [GETH_AI.041] RX DMA stall with varying RX packet length and active Tx DMA in forwarding port

Description

When a forwarding traffic in Bridge ingress port is followed by a short RxDMA packet (64-byte) with timestamp enabled and there is a simultaneous TX DMA burst transfer with BLEN > 8 to the same egress port (as forwarding traffic), a sporadic Rx DMA stall is encountered in the Bridge logic.

Scope

RX DMA Stall occurs when all of the following conditions are met:

- Timestamp snapshot is enabled for all packets
- TxDMA PBL > 8
- Ingress traffic is mapped to both forwarding and RxDMA

Effects

RX stall on the RxDMA channel leading to head of line blocking in the RxQ mapped to the RxDMA. Ingress packets to other RxDMA or forwarding ports are not stalled.

Workaround

TxDMA PBL must be programmed as <= 8.

2.49 [GETH_AI.042] RX frames stall in case of normal status word only

Description

RX DMA Stall occurs when all of the following conditions are met:

- Untagged packets received in ingress MAC which are routed to RxDMA
- Receive Timestamp is disabled

Scope

The issue is applicable only when untagged Ethernet packets are received by host via RxDMA and receive timestamp is disabled.

Effects

RX stall on other ingress ports mapped to Rx DMA leading to head of line blocking in the Rx Queue if the packets from the same Rx Queue are mapped to multiple Rx DMAs.

Workaround

Enable the timestamp snapshot for all packets received by the MAC by setting the register MAC_Timestamp_Control : TSENALL bit to 1.

2.50 [GETH_AI.045] Bridge is padding extra 8 bytes in forwarding packet due to delayed word acceptance in the egress port

Description

When a small burst of forwarding traffic in Bridge ingress port is followed by burst from RxDMA packet containing SOF and there is a simultaneous TX DMA burst transfer to the same egress port (as forwarding traffic), the bridge forwarding logic pads extra 8 bytes in the forwarding packet.

Scope

Following conditions need to be met:

- Ingress traffic is mapped to both forwarding and RxDMA
- Back-to-back traffic sent to forwarding followed by RxDMA
- Simultaneous traffic from TxDMA towards the egress port

Effects

Padding of extra bytes in the forwarding packet. Egress MAC will recompute the CRC for the padded bytes and will not result in FCS error or packet drop in the receiver.

Workaround

Limit the TxDMA PBL <= 4 beats while pushing data to TXQ.

2.51 [HSPHY_TC.005] Loss of Receive Ethernet communication during temperature change

Description

The receive Ethernet SGMII or USXGMII PHY link of HSPHY can become unavailable due to corruption of received data when there is a die temperature drift (either increased or decreased) compared to the temperature at which the HSPHY module is initialized.

The corruption of receive data is due to saturation of Receive (Rx) CDR loop during the temperature drift resulting in wrong RXCLK frequency from HSPHY to GETH module. The Rx CDR loop parameter is configured by XPCSi_VR_XS_PMA_MP_16G_25G_RX_MISC_CTRL0. RX0_MISC bit-field. This bit-field as per programming sequence is wrongly configured to a low value which eventually lead to saturation of Rx CDR loop.

Scope

The issue is applicable only when HSPHY is configured for SGMII or USXGMII modes of operation for Ethernet communication.

Effects

Corruption and unavailability of receive Ethernet data

Workaround

The following changes are to be applied by software to the corresponding Ethernet speed mode programming sequence:

- SGMII 100M at 125 Mbps serial line rate
  - configure XPCSi_VR_XS_PMA_MP_16G_25G_RX_MISC_CTRL0. RX0_MISC bit-field as 177_D
- SGMII 100M or 1G mode at 1.25 Gbps serial line rate
  - configure XPCSi_VR_XS_PMA_MP_16G_25G_RX_MISC_CTRL0. RX0_MISC bit-field as 161_D
- SGMII 2.5G mode at 3.125 Gbps serial line rate
  - configure XPCSi_VR_XS_PMA_MP_16G_25G_RX_MISC_CTRL0. RX0_MISC bit-field as 96_D
- USXGMII 5G mode (Base-R)
  - configure XPCSi_VR_XS_PMA_MP_16G_25G_RX_MISC_CTRL0. RX0_MISC bit-field as 163_D

2.52 [LETH_AI.005] CBS credit not decremented during the IPG phase of transmission

Description

When Credit Based Shaper (CBS) is enabled for a Traffic Class (TC), the packet available in a TC is scheduled for transmission when zero or positive credit is accumulated for the TC. The credit gets incremented when a packet is available in Transmit Queue and is waiting for opportunity to get scheduled for transmission or when the current credit is negative. The credit gets decremented during the actual packet transfer. The rate at which the credit is incremented or decremented depends on the percentage of total bandwidth reserved for this TC.

As per IEEE 802.1Qav standard, the credit must also be decrement when the packet overheads are transmitted. These overheads include the preamble bytes before the start of packet, CRC/FCS bytes, and the minimum IPG (Inter packet Gap of 12bytes) after the end of packet data transfer.

However due to this defect, the MAC decrements the credit only up to last byte of packet data (last byte of Frame Check Sequence (FCS)) transfer and increments the credit during the subsequent nominal IPG period associated with that packet.

Scope

The CBS algorithm is enabled for scheduling the transmission of Audio/Video traffic.

Effects

As the credit is not decremented but incremented during the trailing IPG period of the packet, the accumulated credit value is higher/larger than expected. As a result, the next packet of that traffic class will get scheduled earlier or faster than expected and the process repeats. Consequently, in each time window for which the bandwidth reservation is measured, the actual bandwidth consumed by the TC is more than the software programmed value.

The percentage of extra bandwidth consumed by the TC depends on the packet length. This can be estimated with the following equation assuming the nominal IPG of 12 bytes.

Additional/extra bandwidth = ((# of packets x 12 Bytes) / (Total number of bytes transmitted in that window including preamble bytes of each packet)) x FractionalBW programmed for this TC. For example, 30% BW is

programmed for this TC in which a stream of 100 packets of 128 bytes length is to be transferred. For this example, the additional BW consumed would be equal to = 30% x (100 x 12) / (100 x (8 + 128)) = ~2.65%. Effectively, this TC is allocated 32.65% of total BW instead of the programmed 30%.

Workaround

While programming the parameters that determine the fractional BW for this TC, calculate for a BW fraction target that is lesser the desired one so that the actual BW consumed is closer to the desired one due to the “extra/additional” BW provided by the defective algorithm. The target “lesser” percentage value used for this calculation will be an estimate based on the average length of packets transmitted.

2.53 [LETH_AI.008] Time Aware Shaper (TAS) additional IPG in case of back-to- back packet transmission

Description

When Enhancements to Scheduling Traffic (EST) feature is enabled, the Transmit Scheduler defers next packet scheduling until the current packet is completely forwarded to MAC Transmitter, so that any eligible packet received in higher priority Transmit Queue towards end of current packet transmission is scheduled over already available packet in lower priority Transmit Queue. The Transmit Scheduler design maintains a counter which triggers when packet forwarding to MAC Transmitter starts and increments on every clock cycle until the value (number of clock cycles to transmit the packet including overheads (target count), on the line) determined based on packet size (including the PREAMBLE+SFD overhead) and operating speed is reached.

However, due to the defect, extra IPG (more than programmed minimum IPG) is observed even when back-to-back packets are available for scheduling.

The trigger for counter is generated in f_{LETH} clock domain (packet data read from Transmit Queue and forwarded to the Asynchronous FIFO between f_{LETH} clock domain and MAC Transmitter clock domain) and synchronized to MAC Transmitter clock domain (clock domain in which counter in Transmit Scheduler is incremented, and data read out from Asynchronous FIFO). When the counter reaches the target count, the indication is synchronized back to f_{LETH} clock domain to enable scheduling of next eligible packet. The waiting until current packet is completely forwarded to MAC Transmitter for scheduling the next eligible packet, and the crossing of signals to and from f_{LETH} clock domain and MAC Transmitter clock domain (CDC delays), results in delayed scheduling of next eligible packet, which is reflected in the incorrect extra IPG. This can be in worst case 12 clock cycles of slowest among the two clocks (converted to bit times based on operating speed) based on the frequency and phase relationship of f_{LETH} and MAC Transmitter clocks.

Scope

The Software enables the Enhancements to Scheduling Traffic (EST) feature in MAC to schedule the Time Sensitive Networking (TSN) traffic.

Effects

The extra IPG impacts the number of packets that can be scheduled in Time Interval (TI) or slot/row of EST Gate Control List (GCL). This impacts performance and loss of data due to remote end dropping packets that are received after the expected time.

Workaround

The software must program the TI to accommodate the extra IPG based on the number of packets expected to be scheduled. For example, in 1 Gbps speed mode, extra IPG in worst case is 12 clock cycles and MAC Transmitter clock is slower, therefore, TI must be increased by at least (clock period (8 ns) * extra IPG (12)) i.e., 96 ns per packet.

The above mentioned workaround is only to guarantee transmission of expected number of packets within a scheduled slot and there are no workaround to guarantee minimum IPG between transmission of back to back packets in the same time slot.

2.54 [LETH_AI.010] Reading back contents of the register 10BT1S_PLCA_TIMER, the TOT and BT bitfields are swapped

Description

The TOT and BT fields of B10T1S_PLCA_Timer indirect register are Bits [7:0] and Bits [15:8], respectively. The programmed and read values are expected to follow this format.

However, due to the defect, when the B10T1S_PLCA_Timer indirect register is read, the read values of TOT and BT fields are swapped, that is, Bits [7:0] and Bits [15:8] contain the default or values programmed earlier in BT and TOT fields, respectively. There is no issue with the programmed values in these fields, it is updated to correct registers in the hardware.

Scope

The software reads indirect registers after programming or periodically, for checking integrity, or use read-modify-write approach to program indirect registers.

Effects

As the read values of TOT and BT fields of B10T1S_PLCA_Timer indirect register is incorrect.

Workaround

Software must interpret the Bits [7:0] and Bits [15:8] as BT and TOT, respectively, when B10T1S_PLCA_Timer indirect register is read.

2.55 [LETH_AI.011] 10BT1S Delayed transmission after TO starts as follower

Description

A Follower node starts transmission immediately after valid BEACON is received and Transmit Opportunity (TO) timer is started, if a packet is available. Below is an example for TO timer set to 32 bit times (3.2us) and follower node ID = 1.

The transmission must start within 3.96us to 5.56us after the valid BEACON is received, which is computed as below:

t_{min} = (TX_EN to MDI_{min}) + ($CRS\_deasserted_{min}$) + ($MII\_propagation\_time_{min}$) + (to_timer) x Node_ID = 760ns + (to_timer) x Node_ID = 0.76us + (3.2us*1) = 3.96us

t_{max} = (TX_EN to MDI_{max}) + ($CRS\_deasserted_{max}$) + ($MII\_propagation\_time_{max}$) + (to_timer) x Node_ID = 2360 ns + (to_timer) x Node_ID = 2.36us + (3.2us*1) = 5.56us

However, due to the defect, the actual transmission starts after a delay of 6.8us, which is greater than expected by 1.24us (6.8us – 5.56us).

Scope

LETH 10BASE-T1S is configured as follower node Node ID is non-zero.

Effects

The delayed transmission in Follower node causes mismatch in alignment of TO timers of individual nodes, resulting in unexpected collisions.

Workaround

None.

2.56 [LETH_AI.013] Long COMMIT without TRANSMIT

Description

As per IEEE 802.3cg-2019 Standard, the commit timer duration i.e., the maximum time the PLCA Data state machine is allowed to stay in WAIT_MAC state is 288-bit times or 28.8us., with a tolerance of +/-0.5-bit times or +/-50ns. While the commit timer is running, the PLCA transmits COMMIT symbols to hold the line so that any packet made available in the MAC can be transmitted.

However, due to the defect, the commit timer duration is 30us when there is no packet to transmit during Transmit Opportunity leading to commit timer expiry, which is deviation of 1.2us, causing the holding of line for longer duration, delaying transmission from node with next node-ID.

Scope

LETH is configured in 10BaseT1S mode and the transmit traffic is bursty (no packet to transmit during few of the Transmit Opportunities).

Effects

The extra COMMIT symbols transmitted causes the delaying of transmission from node with next node-ID, impacting the performance.

Workaround

None.

2.57 [LETH_AI.014] 10BT1S Incorrect encoding of first bit after TX command

Description

The LETH 10BASE-T1S Digital PHY transmits RZI encoded 5-bits symbols serially on the TX pin of OA 3-pin interface. Each bit is of duration 80ns (‘0’ is transmitted as low pulse of width 20ns followed by high pulse of width 60ns, whereas ‘1’ is transmitted as alternate low and high pulses of width 20ns each). But there is a special requirement in Open Alliance TC14 PMD XCVR v1.5 specification for the first bit of first symbol of packet data after the TRANSMIT command. A ‘0’ is transmitted as high pulse of width 80ns, whereas ‘1’ is transmitted as high pulse of 40ns followed by alternate low and high pulses of width 20ns each).

However, due to the defect, 10BASE-T1S Digital PHY incorrectly transmits first bit of first symbol of packet data after TRANSMIT command is transmitted as normal ‘0’ or ‘1’ (‘0’ is transmitted as low pulse of width 20ns followed by high pulse of width 60ns, whereas ‘1’ is transmitted as alternate low and high pulses of width 20ns each) instead of complying to the special requirement.

Scope

The LETH is configured for the 10BASE-T1S.

Effects

The incorrect first symbol of packet data is detected as data corruption by the XCVR, leading to generation of collision (low pulse on ED pin). This happens for first transmission as well as the retries of the packet after back-off, therefore, none of the packets can be transmitted successfully. The transmission of COMMIT symbols without packet impacts performance.

Workaround

None.

2.58 [LETH_AI.015] 10BT1S Unintended dropping of next received packet when preceding corrupted SILENCE symbol resembles any of the SSD/ESD/HB/BEACON symbols

Description

When a non-SILENCE symbol is received interleaved between SILENCE symbols, the 10B-T1S Digital PHY discards or ignores it, and the next packet is received normally.

However, due to the defect, when any of the SSD or ESD or HB or BEACON symbol is received interleaved between preceding SILENCE symbols, the 10B-T1S Digital PHY drops the next packet. This is because the elastic buffer in PMA with programmable read threshold (default depth of 14) incorrectly stores non-SILENCE symbols up to the read threshold. These stored non-SILENCE symbols are read out along with the next packet. As this SSD/ESD/HB/BEACON symbol is received at unexpected position the packet is dropped.

Scope

The transmission or random errors on the differential line or external PMD transceiver causes corruption of the data/symbols provided to receiver.

Effects

The next packet after the occurrence of defect scenario is dropped, resulting in data loss.

Workaround

None. The defect is related to handling of error condition.

2.59 [LETH_AI.016]PLCA cycle time between 2 BEACON deviates from expected value

Description

The PLCA Cycle Time which is time interval between start of two sets of BEACONs for a bus with N follower nodes is computed as below.

$tcycle_{min}$ = (1/Rb)*(L_{BEACON}+((N+1)*L_{TO})) = 27.6 µs for 7 follower nodes, where:

Speed of operation (Rb) = 10 Mbps

Length of BEACON (L_{BEACON}) = 20 bits

Transmit Opportunity Timer (TOT field) of B10T1S_PLCA_Timer register (L_{TO}) = 32 bits

The actual PLCA Cycle Time must be $tcycle_{min}$ + RTT = 28.36 µs to 29.16 µs, where:

Round trip time is Line to CRS de-assertion delay. According to the IEEE 802.3 cg-2019 Standard, the delay from MDI to CRS de-assertion is in the range of 0.64 µs to 1.12 µs, and the delay from PLCA TXEN/TXER to MDI is in the range of 0.12 µs to 0.44 µs. Therefore, RTT is in the range of 0.76 µs and 1.56 µs.

However, the actual RTT delay is 4.43 µs which is a deviation of 2.87 µs, and the actual PLCA Cycle Time is 32 µs which is 2.84 µs greater than the expected higher limit.

Scope

LETH is configured for 10BASE-T1S mode.

Effects

The deviation in PLCA Cycle Time or Transmit Opportunity timer results in mismatch of TO timers of individual nodes, resulting in unexpected collisions.

Workaround

None.

2.60 [LETH_AI.017] Incorrect description of register bit-field Portj_B10T1S_PLCA_Sts.PS

Description

The description of the Portj_B10T1S_PLCA_Sts.PS as defined in UM as follows:

- 0: BEACONs are being received or transmitted, and the PLCA Control state diagram is in normal operation
- 1: the PLCA Control state diagram has been in the DISABLE, RESYNC, or RECOVER state for greater than the duration of the PLCA_status timer

However, as per implementation in design is as follows:

- 1: BEACONs are being received or transmitted, and the PLCA Control state diagram is in normal operation
- 0: the PLCA Control state diagram has been in the DISABLE, RESYNC, or RECOVER state for greater than the duration of the PLCA_status timer

and design implementation is as per " OPEN Alliance 10BASE-T1S PLCA Management Registers TC14 – MDIO registers for PLCA v1.3".

Scope

LETH is configured in 10BaseT1S mode and software reads the PLCA status.

Effects

Software will interpret the Portj_B10T1S_PLCA_Sts.PS incorrectly.

Workaround

Software must interpret the register bit-field as implemented in design.

2.61 [LETH_AI.018] RX frames stall in case of normal status word only

Description

The Bridge arbitrates at the ARI burst or packet boundary to route packets from multiple ingress ports to the DMA. After an ingress port wins arbitration, the Bridge blocks requests from all other ingress ports until the DMA has accepted the entire burst or entire packet along with Rx status and Rx Timestamp status from selected port.

The Bridge calculates the number of status words from the ingress port after receiving the first status word (Normal Status) of ARI Rx Status (after EOP is accepted). The Bridge identifies the end of a packet from an ingress port to a DMA channel when the DMA accepts the final status word (Rx status or Rx Timestamp status) for that channel (indicated by the ARI channel number signal).

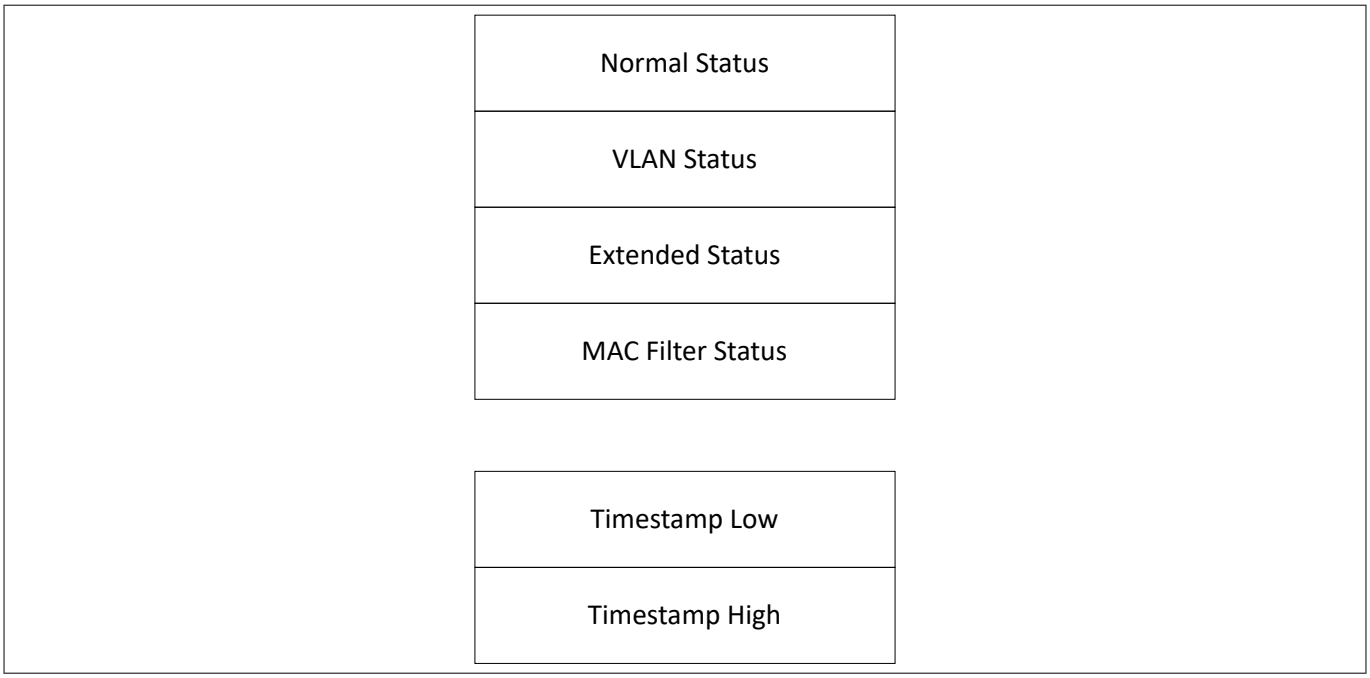


Figure 3 Format of the receive status word order

After identifying the end of a packet, all ingress ports are allowed to arbitrate and ingress port that wins the arbitration will transfer the next ARI burst to DMA.

The defect scenario occurs when the Rx packet has only Normal status (untagged packet with timestamp capturing disabled), and the Normal status is accepted by Bridge in the same clock cycle. The ARI channel number provided by the ingress port can change in the next cycle once the Normal status is accepted.

The Bridge uses registered version of Normal status to generate end-of-packet signal to the Bridge Rx Arbiter to close the ARI grant for current ARI transfer and switch to the next ARI channel. Since the ARI channel number from the MTL changed immediately after Normal status is accepted, the end-of-packet is not generated for the Rx packet and the Bridge Arbiter does not allow Rx packet transfer to other RxDMA channels.

This results in the DMA channel waiting indefinitely for the end-of-packet signal, leading to none of the mapped ingress ports able to transfer packet to that DMA (HOL blocking if packets for multiple DMAs is routed through same Rx Queue). Multiple ingress ports may be impacted as LS Bridge allows mapping of traffic from multiple ports to same DMA channel.

Scope

RX DMA Stall occurs all of the following conditions are met :

- Untagged packets with Destination Address matching Portj_MAC_Address0_High and Portj_MAC_Address0_Low are received in ingress MAC which is routed to RxDMA
- MAC Packet Filtering is used
- Timestamp snapshot is not used (TSENALL=0)

Effects

RX stall on other ingress ports mapped to Rx DMA leading to HOL blocking in the Rx Queue if the packets from the same Rx Queue are mapped to multiple Rx DMAs.

Workaround

Option-1 :

- Enable the timestamp snapshot for all packets received by the MAC by setting the register Portj_MAC_Timestamp_Control : TSENALL bit to 1
- This ensures that there will be minimum 3 status words in the Receive status

Option-2 :

- Program the MAC_Address0_High and MAC_Address0_Low registers to zero and program other MAC address registers with the expected Destination Address for the received packets
- The MAC Address Match field (this field contains the number of the MAC Address register which matches the Destination address of the received packet) of MAC Filter status will always have nonzero value
- This ensures that there will be minimum 2 status words after EOP

2.62 [LETH_AI.019] 10BT1S PLCA_R bit of Portj_B10T1S_PLCA_Ctrl register is not self clearing

Description

As per OPEN Alliance 10BASE-T1S PLCA Management Registers v1.3 specification, the access attributes of RST (PLCA Reset) bit of PLCA control register #0 is Read/Write/Self-Clear as described below:

RST: When this bit is set to a logical 1, the PLCA RS functions are reset. This bit is self-clearing and shall read as 1 while PLCA reset is in progress, otherwise it shall read as 0.

This control bit is implemented in design as PLCA_R bit of Portj_B10T1S_PLCA_Ctrl register.

However, due to the defect, the PLCA_R bit is incorrectly not self-clearing.

Scope

Software uses the PLCA reset function to reset the PLCA block.

Effects

As the PLCA_R bit is not self-clearing, the PLCA remains in reset state until the bit is written to 0_B by the software, requiring additional write access to the Portj_B10T1S_PLCA_Ctrl register to bring PLCA block out of reset.

Workaround

The software must explicitly write the PLCA_R bit to 0_B after a delay equal to 3 clock cycles each of f_{LETH} and MAC transmitter clock, to bring the PLCA block out of reset.

2.63 [LETH_AI.020] 10BT1S Coordinator transmits the packet without BEACON (during unintended TO)

Description

The Coordinator Node always transmits packet after the preceding BEACON. However, due to the defect, Coordinator Node transmits packet without BEACON. When Coordinator Node receives its own reflected BEACON without packet data, PLCA active is de-asserted and PLCA status counter is triggered. During this time if a packet from a Follower Node is received such that reception continues beyond the PLCA Status timer expiry resulting in the PLCA Status to fail. This switches the design to operate in normal CSMA/CD mode, the Coordinator Node transmits packet without BEACON after the packet from Follower Node is received. During this time if the Current ID reaches the local Node ID 0 (Coordinator Node), the CTRL FSM transitions to SEND_BEACON state (where PLCA active is set high). This suspends the current packet transmission from Coordinator Node and transmits a single BEACON symbol as the DATA FSM transitions from DATA_NORMAL to DATA_IDLE state. After this the packet from Coordinator Node is resumed as a new packet.

Scope

The software enables 10BASE-T1S Digital PHY as Coordinator Node.

Effects

The Nodes on the network go out of sync as valid BEACON symbol is not received from the Coordinator Node resulting in continuous TO timer restart/expiry and collisions.

Workaround

None.

2.64 [LETH_AI.022] 10BT1S Unintended dropping of next received packet when first of the two SYNC symbols right shifted by one bit position resembles the SSD symbol

Description

A packet is received by 10B-T1S Digital PHY when it starts with two each SYNC (at least one SYNC symbol required) and SSD symbols. This implies that a packet can be received even when first of the two SYNC symbols mismatch.

However, due to the defect, when a packet is received with the first of the two SYNC symbols when right shifted by one bit position resembles SSD symbol, the packet is unexpectedly dropped.

PMA has a symbol aligner block which looks for special symbols like SYNC (0x18)/SSD (0x04)/HB (0x0d)/BEACON (0x08) and then transmits valid data to PCS block. There is a shift register for serial to parallel conversion. The data in shift register is validated after receiving 5 bits. If the PMA does not find a special symbol in the first data, it continues to compare the data in shift register with special symbol pattern irrespective of the validity (i.e. non-5-bit boundaries). If it finds one later, it is considered as start. Due to this, for example, if first SYNC symbol is corrupted to 0x9, which when shifted right by one bit position is detected as SSD symbol resulting in providing subsequent data to PCS (second SYNC is considered as data due to incorrect SSD detection). The same can happen when special symbol is not detected before say data 0x10, 0x11, 0x12 or 0x13.

According to the IEEE 802.3-cg-2019 Standard, PCS block expects another SSD symbol immediately after it receives the corrupted SYNC symbol which matches with SSD symbol. Since there is the proper SYNC symbol following the corrupted symbol and not the SSD symbol, PCS block drops the received packet

Scope

The transmission or random errors on the Ethernet Line or external PMD transceiver causes corruption of the data/symbols provided to host controller.

Effects

The next packet after the occurrence of defect scenario is dropped, resulting in data loss.

Workaround

None. The defect is related to handling of error condition.

2.65 [LETH_AI.023] Incorrect access type is specified for bit-fields in register Portj_B10T1S_PLCA_Sts

Description

The bit-fields BCNBFTO, UNEXPB and RXINTO in register Portj_B10T1S_PLCA_Sts described in User Manual as read-only bit-fields.

However, as per design and "OPEN Alliance 10BASE-T1S PLCA Management Registers TC14 – MDIO registers for PLCA v1.3" these register bit-fields should be read-only (RO) and write 1_B to clear (W1C).

Scope

LETH is configured as a10BaseT1S. They are some errors on the line related to bit-fields mentioned above.

Effects

Due to wrong description software wrongly interprets the bit-field.

Workaround

Software must consider the bit-fields as defined in Open Alliance PLCA register specification and implemented in design.

2.66 [LETH_AI.024] Transmit timestamp is not properly transferred in descriptor if TxDMA channel is mapped to any queue except TxQ0 and bridge is enabled.

Description

When the ‘ati_txstatus_avail’ signal is asserted for a specific DMA channel, the DMA initiates the descriptor write-back process for that channel. At this stage, the DMA updates the ‘ati_txsqnum’ value to select the channel number with available transmit status, allowing it to read the corresponding Tx Status. The Bridge then transfers this ‘ati_txsqnum’ value from the DMA to the mapped egress port, based on the mapping between DMA channels and Tx Queues using ‘ati_txstatus_ack’ as qualifier. By default, the DMA outputs ‘ati_txsqnum’ as 0, the Bridge will select the ‘ati_txstatus’ from the TXQ of Ethernet Port which is mapped to TxDMA0 towards the DMA. The DMA copies the timestamp information pertaining to TxDMA0 into the descriptor for all Tx DMA channels which is then written back to TDES0, TDES1 and TDES2 incorrectly. DMA issues ‘ati_txstatus_ack’ along with ‘ati_txsqnum’ for TDES3, which the Bridge will select the ‘ati_txstatus’ from the correct Ethernet Port mapped to ‘ati_txsqnum’ and the Tx DMA descriptor TDES3 write-back for all DMA channels is not corrupted.

Scope

All of the below conditions are met:

- Multi-port configurations with TxDMA channels not mapped to TXQ0
- TTSE bit is set in the TDES2 Normal Descriptor Read

Effects

Transmit Timestamp status is not correctly written in TDES0, TDES1 during Transmit Normal Descriptor write back.

Workaround

None.

2.67 [LETH_TC.010] Missing PTP time sync concept among all LETH0 MAC ports

Description

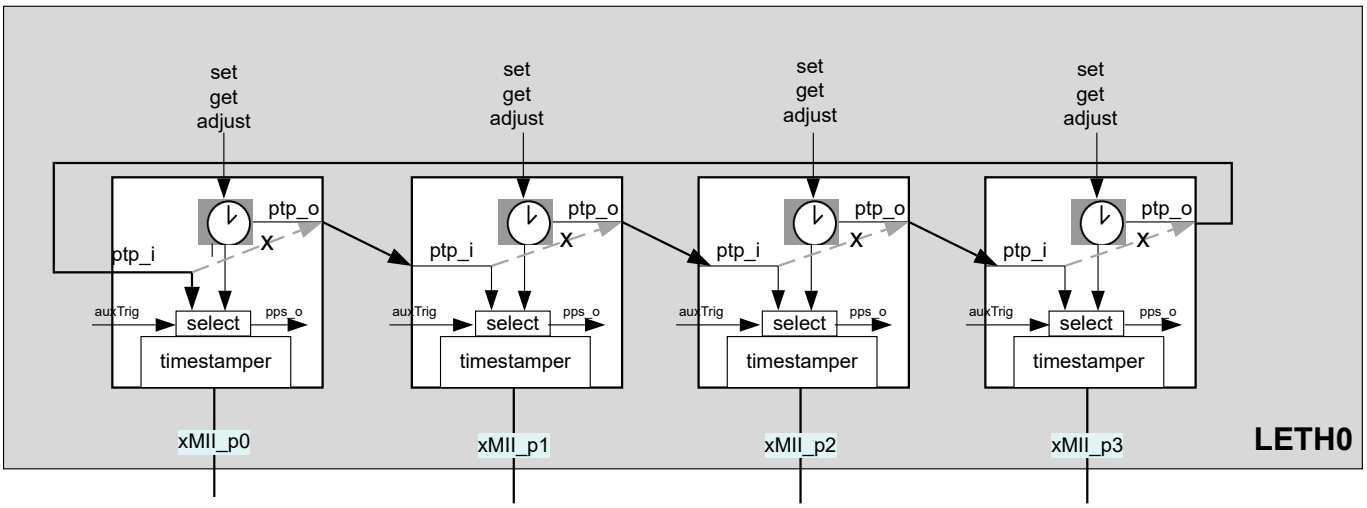
PTP (IEEE1588) transparent clocks and gPTP (IEEE802.1AS) bridges require a common time base for timestamping on ingress (RX) and egress (TX) ports. The value of RX and TX timestamps are used by the PTP/gPTP protocol to update the correctionField (CF). The CF is updated by the PTP/gPTP protocol by adding the residence time of a SYNC message to the existing value of the CF. The residence time of a SYNC message is

calculated by building the difference between the SYNC message TX timestamp and the SYNC message RX timestamp. To achieve this, both RX and TX timestamps have to be taken from the same time base.

For packet timestamping, each LETH0 MAC port can select between an internal local time base and an external time base with the Portj_MAC_Timestamp_Control.ESTI register bit.

A daisy-chaining of internal local time base output of LETH0 MAC port (j) and external time base input to LETH0 MAC port (j+1) provides a mechanism for time base distribution between LETH0 MAC ports.

However, due to limitation in LETH0 MAC ports if timebase for timestamping is selected via external time base input by setting Portj_MAC_Timestamp_Control.ESTI register bit, LETH0 MAC port cannot output the 64-bit PTP time.



Scope

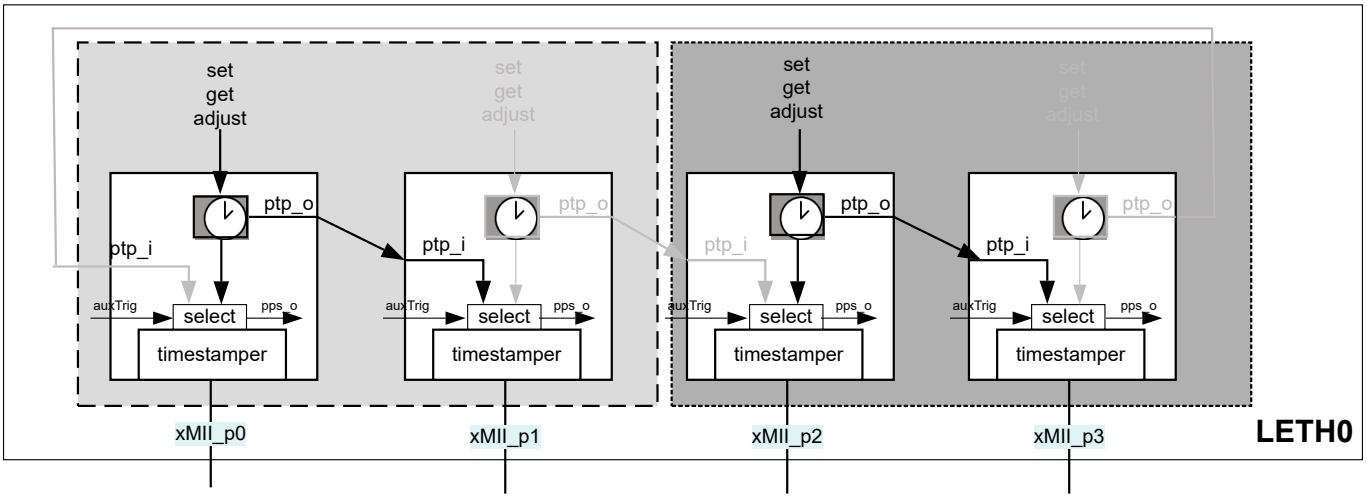
Zonal/domain controllers with PTP transparent clock/gPTP, bridge function among multiple ETH ports

Effects

The missing connection between external time base input to internal local time base output (of the same port) leads to the limitation that only pairwise daisy-chaining of time base forwarding can be configured:

LETH0 MAC port 0 internal time base output can be forwarded to external time base input of LETH0 MAC port 1

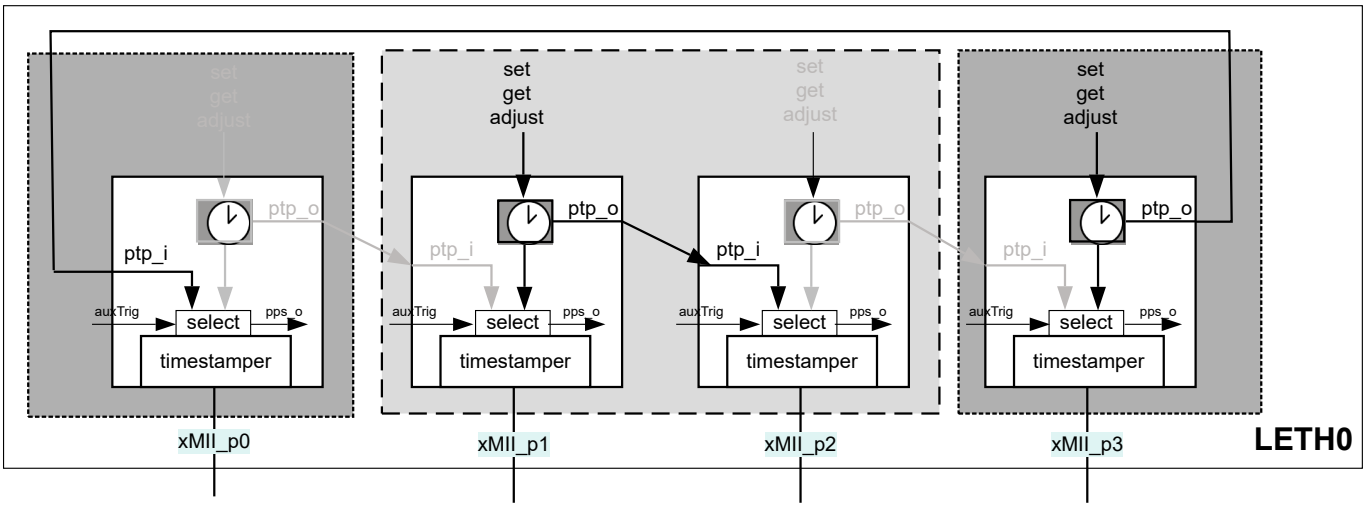
LETH0 MAC port 2 internal time base output can be forwarded to external time base input of LETH0 MAC port 3



Or

LETH0 MAC port 3 internal time base output can be forwarded to external time base input of LETH0 MAC port 0

LETH0 MAC port 1 internal time base output can be forwarded to external time base input of LETH0 MAC port 2



Workaround

No workaround possible.

2.68 [LETH_TC.013] LETH0 PPS output for some PORT pins is inverted

Description

Each MAC port in LETH0 module provides the functionality to output the Pulse-Per-Second (PPS) signal at a port pin which can be configured in period and duty cycle as fixed or flexible PPS. Due to a defect, the PPS output signal of some pins is inverted, see table below:

Table 14

LETH0 MAC port PPS	PORT_pin
LETH0_P0_PPS	02_13
LETH0_P1_PPS	32_06
	35_05
LETH0_P3_PPS	02_14
	03_13

Scope

The PPS output signals on pins as mentioned in the table above.

Effects

The PPS output for above mentioned PORT pins are inverted.

Workaround

Option 1: Use the alternate PPS signals for LETH0 MAC ports as shown in table below:

Table 15

LETH0 MAC port PPS	PORT_pin
LETH0_P0_PPS	14_02

(table continues...)

Table 15 (continued)

LETH0 MAC port PPS	PORT_pin
LETH0_P1_PPS	21_00
LETH0_P2_PPS	02_11
	03_06

Option 2: As a second option, an external inverter IC could be used. Propagation delay of this device must be taken into account for timing and calculation of period and duty cycle values.

2.69 [LETH_TC.015] FRP last instruction index is not updated correctly in receive descriptor (RDES2)

Description

When the Flexible Receive Parser (FRP) (FRPE bit of Portj_MTL_Operation_Mode register is 1_B) and Last Instruction in RX Status (ELIRS bit of Portj_MTL_RXP_Control_Status register is 1_B) are enabled, and RXPS field (bits [13:11]) of RDES2 of receive descriptor) is not equal to 5 (L2 register filter fail and receive parser processing bypassed), the RDES2[26:19] or MADRM field contains the last instruction index of the FRP, otherwise the matched MAC address register number. The RDES2[31:27] and RDES2[18:16] provides the L2/L3 Registers filters status irrespective of FRP enable or disable. However, due to the defect, FRP last instruction index status is provided in RDES2[23:16] instead of expected RDES2[26:19]. The RDES2[18:16] field is overwritten and RDES2[31:27] field is updated with all-zeros, impacting the status of L2 registers filters.

In devices with LETH bridge, when the flexible receive parser is used in multi-port bridge configuration, the MADRM bit field of RDES2 is not correctly updated with the last instruction index of the Receive Parser and DMA_CHy_RXP_Accept_Cnt does not increment although the parser has been able to detect the frame.

Scope

Software enables FRP (FRPE bit of Portj_MTL_Operation_Mode register is 1_B) and Last Instruction in RX Status (ELIRS bit of Portj_MTL_RXP_Control_Status register is 1_B).

Effects

The misplaced or incorrect FRP last instruction index status field and overwriting of L2 registers filters status fields impacts the software functions dependent on it.

Workaround

Use cases with bridge not enabled:

- Software must interpret RDES2[23:16] as Rx Parser last instruction index status, if Flexible Receive Parser (FRP) (FRPE bit of Portj_MTL_Operation_Mode register is 1_B) and Last Instruction in RX Status (ELIRS bit of Portj_MTL_RXP_Control_Status register is 1_B) are enabled, and RXPS field (bits [13:11]) of RDES2 of Receive Descriptor) is not equal to 5 (L2 Register Filter Fail and Receive Parser processing bypassed), the L2 registers filters status is not provided even if corresponding filters are also enabled. Otherwise, interpret RDES2[31:16] as L2 registers filters status

Use cases with bridge enabled:

- No workaround is possible

2.70 [LETH_TC.017] Layers of LETH0_P3 MDIO mapped in P13.6 for TC4Dx

Description

In datasheet, 2 groups of LETH0_P3_MDIO and LETH0_P3_ED signals are listed for P13.6: group-E and group-D. Only the signals of group-D are functional and can be used: LETH0_P3_MDIOD and LETH0_P3_EDD.

Scope

LETH0_P3 (MDIO and ED signals).

Effects

LETH0_P3_MDIOE and LETH0_P3_EDE not available on P13.6 and therefore cannot be used.

Workaround

P13.6 can be used for LETH0_P3_MDIO and LETH0_P3_ED, but it is necessary to configure and use group D: LETH0_P3_MDIOD and LETH0_P3_EDD.

2.71 [LMU_TC.003] LMU can lock up when writes immediately follow a protection modification

Description

An instance of the LMU can lock up under the following circumstances. A CPU is updating either the PROTSE or PROTRGN register, and another $master_y$ [CPU or DMA] is concurrently performing data writes to the same instance's SRAM. If the other $master_y$ write occurs just after the PROTSE/PROTRGN update then the LMU instance will lockup.

The specific conditions for the issue to occur are:

In the situation that a $master_y$ [CPU or DMA] is performing data accesses to LMU_z SRAMs with 8/16/32-bit writes and in parallel, another CPU_x is updating LMU_z's PROTSE/PROTRGN register. The data write occurrence of $master_y$ has to directly follow the PROTSE/PROTRGN register write by CPU_x, in fact there is only a single clock cycle window in which this issue can occur.

For clarification, in the case of PROTRGN update, the lock up occurs regardless of whether the PROT region update continues to allow or prohibits the data writes from $master_y$.

Scope

This issue could arise for any instance of an LMU within the device.

Effects

The LMU instance being targeted with the specific parallel activity listed above will become locked up and will stall any master attempting a subsequent transaction (read or write) to that instance until the next reset.

Workaround

To prevent the error condition arising, there are two possible workarounds:

1. Co-ordinate data accesses between the two masters so that data writes do not occur in the susceptible cycle window. Synchronization between the two masters is required so that $master_x$ can be assured that when it executes the PROTSE/PROTRGN update , that $master_y$ has stopped issuing writes and there are no outstanding writes to the LMU instance. $Master_y$ data writes should only resume once $master_x$ has confirmed that the PROTSE/PROTRGN update has completed
2. Ensure all data writes around the register update are not less than 64-bits. In the situation that $master_y$ is not synchronized with the register update this means all data writes by $master_y$ must be 64-bit or

larger. This would realistically be achieved by accessing the LMU SRAM as cacheable by a CPU, or DMA channel data widths of only 64-bit or larger

2.72 [MCDS_TC.069] COUNT field of FIFOOVRCNT wraps around and does not saturate

Description

The counter related to the COUNT field of the FIFOOVRCNT register should increment until its maximum value and stop there. The problem is that the counter wraps around and does not saturate at the maximum value.

Scope

Trace and debug functionality.

Effects

Due to wrap around of the counter, the exact number of ERR messages cannot be determined from the COUNT field. ERR messages would be COUNT + (N*256), where N starts from 0 and increments with each wrap around. Number of ERR messages might be equal to COUNT value if less than 256 ERR messages occur. Or ERR messages would be equal to COUNT value + (N*256) if 256 or more ERR messages occur.

Workaround

In order to evaluate if a wrap around has happened (N>=1), read the COUNT field of the FIFOOVRCNT register more frequently.

2.73 [MCDS_TC.070] FIFOPRE[11:5] consideration for PRE and PRE-POST mode decision

Description

MCDS may work in PRE mode instead of PRE-POST mode under the certain configuration of FIFOTOP, FIFOBOTTOM, FIFOPRE.

MCDS should compare the bits[31:5] of top-bottom to FIFOPRE[31:5], and if they are equal then it should go to PRE mode, else (top-bottom is not equal to FIFOPRE) it should go to PRE-POST mode.

Only part of the FIFOPRE bits (31:12) are used to decide between PRE and PRE-POST mode. The FIFOPRE[11:5] bits are ignored (assumed all 1) when equating it with top-bottom. So, if bits 31:12 of top-bottom are equal to FIFOPRE[31:12] (regardless of FIFOPRE[11:5]), RTL goes into PRE mode.

Scope

Trace functionality

Effects

Design may operate in PRE mode instead of PRE-POST mode.

Workaround

Configure the FIFOTOP, FIFOBOTTOM, FIFOPRE registers appropriately so that RTL works in PRE-POST mode, taking into account that the FIFOPRE[11:5] bits are ignored when calculating the TOP-BOTTOM fraction as an integer multiple of 4 KBytes which is the size of a single paragraph.

***Note**: The physical trace buffer memory (TBUF) size is up to 32 KBytes and organized in paragraphs.*

2.74 [MCDS_TC.071] Multick not working as expected

Description

The multick message might be corrupted by continuous multicks, after a paragraph boundary or after a kernel reset.

Scope

TRACE function.

Effects

The usage of multick messages can result in wrong tick values or even decoding errors in special cases.

Workaround

All effects can be avoided by a periodic watch point message, which prevents a second multick message in a row. For the first trace session after power-on reset, the period can be 30000 ticks, for the following trace sessions only 208. Please note that only EDs support more than one trace session after power-on reset.

2.75 [MCDS_TC.072] Unexpected VM change STATE messages

Description

Unexpected STATE messages for VM changes can be encountered in the TRACE data.

Scope

Trace function.

Effects

STATE messages for the transition to a trace disabled VM may only occur, if the previous VM was trace enabled. The current implementation will also create STATE messages for transitions from a disabled VM to another disabled VM under certain conditions.

Workaround

The trace decoder should filter out these STATE messages.

2.76 [MCDS_TC.073] VM change STATE message after IP message instead of before

Description

STATE message appears after the IP message in trace data in the same MCDS clock cycle (no tick in-between).

Scope

Trace functionality.

Effects

This behavior will not cause any VM information leaks, but the reverse message order would be much more convenient for trace decoder implementation.

Workaround

Trace decoder can reverse the order and assign the message to the right VM. This is possible only when ticks are enabled.

2.77 [MCDS_TC.075] Wrong first tline after TRIF FIFO overrun

Description

If the TRIF FIFO is concurrently written and read (TRIF ONCHIP_RAM, STREAM_SGBT/ETH/DAP modes) and an overrun occurs, the first tline of the following MTSC can be wrong. Instead of the first tline of the first paragraph after the overrun, it may still contain the following tline of the location, where the overrun occurred.

Scope

TRACE functionality during TBUF overflow.

Effects

More trace data than necessary will be lost.

Workaround

Trace based measurement is not affected, since overruns are prevented by trace filtering and back pressure.

Debug trace can use this workaround:

- A tool can safely detect an overrun with the qline entry
- In this situation at least one to two paragraphs (4-8 Kbyte) of trace data are already lost
- A safe decoding restart is possible with the next paragraph start without overrun
- This means that additional 4 Kbyte of trace are lost for decoding

Since the effect will never occur, when the FIFO is read after the MCDS trace output is stopped, an overrun entry in the qline of the first MTSC can be ignored for ONCHIP_TFIF mode. The trace decoder can safely detect this situation in the qline of the first MTSC:

- MTSC counter (CNT) is 0
- The first two tlines of this MTSC are empty (TLQ = 0)
- The third tline will contain the overrun (fourth for second MCDS instance)

2.78 [MCDS_TC.077] MCDS_TSUEMUCNT not counting

Description

The MCDS.TSUEMUCNT register does not increment, and only bit 0 is observed to toggle when it should be counting. It will only ever return the value 0 or 1 when read, and toggles between the two values when trace is enabled.

Scope

MCDS.TSUEMUCNT register.

Effects

The MCDS.TSUEMUCNT register is unusable as a counter to indicate how many MCDS clock cycles have been consumed while tracing. The MCDS.TSUREFCNT is unaffected, but this traces the reference clock, which can be slower than the MCDS clock.

Workaround

The user can use the STM to count clocks while tracing, if it is not being used for something else by the software being debugged, or use MCDS.TSUREFCNT if lower resolution can be accepted.

2.79 [MCDS_TC.078] Wrong configuration of DMA0 Trace

Description

The FPI BCU for TC4Dx has been misconfigured for DMA0 and DMA1, which results in merged trace output.

Scope

The FPI BCU has been configured so that the DMA0 and DMA1 priorities are the same, and both have priority 1. DMA0 should have priority 0.

Effects

The effect of this is DMA traces if both DMAs are being traced at the same time will be confused as they will be merged, although you can identify what DMA is doing what via the observed address.

Workaround

The trace tool needs to apply additional processing to extract the needed DMA trace streams by address.

2.80 [MSC_TC.018] TFIP is used to select service request output line SR0-3 for time frame Manchester interrupt TFMI instead of TFMIP

Description

Both DTFI and TFMI interrupt lines are selected using TFIP. The same output line SRn(n=0-3) is selected for both of these interrupts.

Scope

MSC interrupt generation.

Effects

The selection of different interrupt output lines for DTFI and TFMIP interrupts is not possible.

Workaround

Upon an interrupt, the application software shall check the ISR register to determine the correct source of the interrupt, that is whether DTFI or TFMI interrupt has occurred.

2.81 [MSC_TC.019] MSC alternate SRI clock source feature is not functional

Description

MSC alternate clock source feature is not functional.

Scope

MSC alternate clock source feature

Effects

MSC alternate clock source feature is not usable.

Workaround

No workaround is available, do not use MSC alternate clock source feature.

2.82 [MSC_TC.025] Slow channel feature is not functional

Description

Slow channel feature is not available for TC4Dx.

Scope

Slow channel

Effects

Slow channel feature not available for TC4Dx.

Workaround

No workaround. Do not use MSC in slow channel mode.

2.83 [MSC_TC.026] PPDE field considered when EXEN=1 and EXEN=0 as well

Description

t_{FCL} cycles and t_{PTF} cycles are incorrectly calculated in case DSTE.PPDE is used. PPDE is always considered for calculating Ncycles irrespective of Standard mode (DSCE.EXEN=0) or Extended mode (DSCE.EXEN=1).

Scope

Limited to Standard mode when DSCE.EXEN = 0.

Effects

DSTE.PPDE is taken into consideration to compute the Ncycles even if DSCE.EXEN = 0.

Workaround

User has to set the DSTE.PPDE=0 when using the Standard mode DSCE.EXEN=0.

2.84 [PCIE_AI.002] L1 PM Substates Capabilities Register fields are not HWINIT

Description

Some L1 PM Substates Extended Capabilities register fields defined as HWINIT in the PCI Express Base Specification can be modified with DBI access while operating as a Downstream Port.

Scope

The following register fields in the L1 PM Substates Extended Capability can be modified by DBI:

- PCI-PM L1.1 Supported (L1_1_PCIaPM_SUPPORT)
- PCI-PM L1.2 Supported (L1_2_PCIPM_SUPPORT)
- ASPM L1.2 Supported (L1_2_ASPM_SUPPORT)
- ASPM L1.1 Supported (L1_1_ASPM_SUPPORT)
- L1 PM Substates ECN Supported (L1_PMSUB_SUPPORT)
- Port Common_Mode_Restore_Time (COMM_MODE_SUPPORT)
- Port T_POWER_ON Scale (PWR_ON_SCALE_SUPPORT)
- Port T_POWER_ON Value (PWR_ON_VALUE_SUPPORT)

Effects

The PCI Express Base Specification defines the HWINIT registers. Any DBI access can modify the fields mentioned above.

Workaround

Software should not attempt to write to the mentioned fields in the L1 PM Substates Extended Capability register.

2.85 [PCIE_AI.003] PME retransmission time not satisfied

Description

The PCIe specification defines that the PM_PME messages must be retransmitted every 100ms +50%/-5%.The PCIe controller does not meet this requirement when the link is in the L1 sub-state.

Scope

PM_PME messages

Effects

Delayed response time of approximately up to 10 times (depending on the f_{OSC}) than required by the PCIe specification.

Workaround

No workaround available.

2.86 [PCIE_AI.004] DMA Configuration Register access failure

Description

If non-zero Processing Hint (PH) field values are received in requests from a remote partner, destined for TRGT0, the LBC block passes those bits through as the two LSBs of the address to CDM.

If the PH[1] bit is set when accessing any DMA configuration register, the register access fails without notification.

Scope

The remote partner has TLP Processing Hint (TPH) feature enabled, and sends a request to DMA configuration registers with PH[1] bit set.

Effects

The defect has the following effect:

- Incorrect DMA configuration register access. If PH[1] bit is set, no register access happens
- The request is acknowledged as usual
- Read access returns 0; Write access has no effect

Workaround

Ensure driver software of the remote device does not generate TLPs with TPH/PH set when accessing DMA configuration registers.

2.87 [PCIE_AI.005] TLP with ECRC error not handled as advisory non-fatal error

Description

A non-posted TLP with ECRC error is not handled as advisory non-fatal.

Scope

When the ECRC_ERR_SEVERITY (ECRC Error Severity) field in the UNCORR_ERR_SEV_OFF (Uncorrectable Error Severity) register is set to non-fatal.

Effects

Controller sends ERR_NONFATAL instead of ERR_CORR when the severity is set to non-fatal, that is to send advisory non-fatal error.

Workaround

None.

2.88 [PCIE_AI.006] Data transfer completes without channel status update

Description

When we enable at least two DMA-read channels or two DMA-write channels concurrently in linked list mode, if at least one of two concurrent operating channels finishes while the DMA is processing a descriptor for the other channel, within a few cycles semaphoring can get into an erroneous state. When this happens, the status FSM stalls before it can generate interrupts or update channel status to STOPPED.

Scope

When at least two DMA-read channels or two DMA-write channels are enabled in concurrent linked list mode.

Effects

The defect has the following effect:

- Data is fully and correctly transferred (the transfer size and pointers are correctly updated)
- DMA detects end of transfer event but does not update channel status register (DMA_CH_CONTROL1_OFF_[WR|RD]CH_[0:7].cs) from RUNNING to STOPPED
- If the last valid data transfer descriptor had an interrupt setup, it is not generated
- After this event all channels in the affected DMA [Write | Read] Engine are stalled

Workaround

Use only one channel of each type, DMA-read or DMA-write when using the linked list mode.

2.89 [PCIE_AI.007] Updating PCIe common reference clock configuration

Description

UPCS override settings given in the TC4Dx PCIe functional description section in Table "UPCS common refclk override settings require a correction as follows:

Table 16 UPCS common refclk override settings corrections

Register	Value
UPCS_OVRD1	$7802009D_H$
UPCS_OVRD2	$000115F5_H$
UPCS_OVRD15	00000203_H

Scope

These overrides are required for PHY operation in PCIe common refclk mode.

Effects

These overrides need to be applied for the PHY to function as specified in PCIe common refclk mode.

Workaround

UPCS overrides are to applied as specified in the UM with the modifications listed here.

2.90 [PMS_TC.020] XRAM uncorrectable error alarm issue

Description

The SMU alarm ALMSF10[22] XRAM uncorrectable error, which originates form the SCR, could be missed due to a synchronization error. This is however, not considered a critical problem because:

- SCR and this alarm are not relevant for safety (SCR is a QM module)
- The same alarm is available anyways through the standard VMT interface ALMSF4[11] VMT Uncorrectable Error, which is not affected by the synchronization issue

Scope

SMU_STDBY

Effects

Missed redundant alarm.

Workaround

If needed, use the ALMSF4[11] VMT Uncorrectable Error instead of ALMSF10[22] XRAM uncorrectable error.

2.91 [PMS_TC.029] Resets MONBISTSTAT

Description

The MONBISTSTAT register has cold PORST as synchronous reset not warm PORST.

Scope

MONBISTSTAT register

Effects

MONBISTSTAT register is not getting reset after warm PORST.

Workaround

No workaround needed as MONBISTSTAT flags can be cleared anytime by software with the MONBISTCTRL.TSTCLR flag which is the normal use case.

2.92 [PMS_TC.030] clk_xtal is not synchronous for one cycle after transition from STANDBY1 to RUN mode

Description

During the STANDBY1 to RUN mode transition, RTC clock is not synchronous to the 100 MHz backup clock for one clock cycle of the backup clock.

Scope

clk_xtal

Effects

If SCR tries to access the RTC trim registers at the same time the STANDBY1 to RUN mode transition takes place then it SCR cannot access the RTC registers.

Workaround

If the STANDBY1_70K to RUN mode transition is initiated then SCR needs to wait two clock cycles of XTAL clock before accessing the RTC registers. The application software can use the timer to wait for minimum 2 XTAL clock.

2.93 [PMS_TC.032] HPBG trimming via SCR_SCU_PMSHPBG_BGTRIM

Description

While transition from RUN or STANDBY1 mode to STANDBY0, and then coming back, the content of the HPBGTRIM is lost. Additionally, write access to SCR register SCR_SCU_PMSHPBG_BGTRIM should update the HPBG trimming value in PMS which may not be successful.

Scope

HPBG

Effects

While transition from RUN or STANDBY1 mode to STANDBY0 and then coming back, the HPBG temperature dependent trimming is lost.

Workaround

Before entering to STANDBY0 mode the HPBGTRIM register value must be stored to SBRAM, and SBRAM retention must be enabled.

After STANDBY0 to RUN mode transition the TriCore™ can retrieve the stored HPBGTRIM register value. Depending upon the DTS temperature reading it can update the HPBG trimming to generate the temperature compensated HPBG reference.

After the STANDBY0 to STANDBY1 transition the following steps must be executed by SCR to have a temperature compensated HPBG reference:

1. If the STANDBY_70k clock is selected, then must be switched to 100 MHz clock. Optionally the higher clock division factor can be used to reduce the dynamic power consumption
2. If the DTS is disabled, then it must be enabled
3. If ADCOMP is enabled, then ADCOMP state must be stored
4. Disable the ADCOMP
5. Read the DTS output
6. Select the STANDBY_70k clock
7. Depending on the DTS reading, the corresponding TRIM (ranges from TRIM0 to TRIM7) to be retrieved from the SBRAM
8. Write the calculated final trimming value to the SCR_SCU_PMSHPBG.BGTRIM register
9. Set back the clock as it were before executing the step 1
10. If ADCOMP was previously enabled, then enable the ADCOMP

2.94 [PORTS_TC.010] P20.6 is in Pull up mode even if HWCFG[6]=0

Description

HWCFG[6] offers the possibility of GPIO state selection during reset. Either tristate or pull-up can be selected (depending on HWCFG[6]). For the P20.6 GPIO, only the pull-up configuration is possible (independent of the HWCF[6] state).

Scope

Reset state configuration of ports via HWCF[6] selection.

Effects

The state of the P20.6 GPIO during reset is configured as pull-up.

Workaround

Shortly after reset state can be changed by writing into P20_PADCFGp_DRVCFG (p=6) register.

2.95 [PORTS_TC.011] PCSRSEL=1 blocks the emergency stop signal

Description

When RGMII or xSPI is active on Port 16 (P16_PCSRSEL.PCSRx = 1_B), the port emergency stop request from SMU does not halt the ongoing RGMII or xSPI transactions in P16.

Scope

The issue is valid only for Port 16 which is configured for RGMII or xSPI.

Effects

The port emergency stop feature does not block the RGMII or xSPI transaction in P16.

Workaround

Upon a safety relevant fault reaction from SMU, the software must write P16_PCSRSEL.PCSRx bit-field to 0_B to stop the ongoing RGMII or xSPI transactions in Port 16.

2.96 [PPU_AI.003] Single stepping might lead to problems

Description

When using the debugger to step through instructions, the debugger may use one of the two techniques depending on the scenario – one is to place a break-point after the current source statement and run to this, the other is to carry out an auxiliary register write to the IS bit of the DEBUG register (single instruction stepping). The single instruction stepping is not working as expected.

Scope

Single instruction stepping debug operations of the PPU.

Effects

If single instruction stepping is used, it will not work reliably and the IS bit will not be deasserted after the step, when a vector instruction is being stepped or when a scalar instruction is stepped that immediately follows a vector instruction.

Workaround

Avoid single stepping vector instructions. Instead, place a break-point after a vector instruction sequence and run to this break-point.

2.97 [PPU_TC.002] Trace flush is not working

Description

The PPU internally has a buffer for the trace data gathered. In order to get the full trace out to the debugger, a flush can be requested via the PPU trace register ARCT_FLUSH_CTRL. To trigger the action, the flush command is set by the software or the tool.

The trace flush mechanism is not fully functional. In normal operation, the trace flush command is reset by the hardware once the command is successfully executed.

After the execution of the trace flush command, hardware does not reset this command and as a result, future trace flush requests have no effect and the trace buffer will not be emptied.

Scope

Trace

Effects

Trace flush requests have no effect and the trace buffer will not be emptied until after a PPU reset.

Workaround

In order to receive the complete trace perform the following steps:

1. Trigger flush command

2. Check that the program trace correlation message is received
3. Execute a kernel reset of PPU to be able to trace another sequence

2.98 [PPU_TC.005] External event (CTRL.REQWU) only cleared after second wait event instruction executes

Description

When the PPU is woken up by an external event via setting the CTRL.REQWU bit, the event does not get cleared. Therefore the next executed wait event (WEVT) might not behave as expected as PPU might not enter the sleep state due to the still pending request run.

Scope

Clearing of external event signal.

Effects

External event signal by setting the CTRL.REQWU bit to 1 only cleared after second WEVT instruction executes.

Workaround

To consistently clear the wait event, it is recommended to use two back-to-back WEVT instructions to ensure that the event is cleared as expected.

2.99 [SAFETY_TC.031] Test procedure update for SM[HW]:SRI:ERROR_HANDLING and SM[HW]:LLI:ERROR_HANDLING

Description

The mandatory test procedure of SM[HW]:LLI:ERROR_HANDLING and SM[HW]:SRI:ERROR_HANDLING states: "...by reading from a register...". This statement poses a limitation to the implementation of the procedure as it only mentions a read access.

Scope

Test procedure of SM[HW]:LLI:ERROR_HANDLING and SM[HW]:SRI:ERROR_HANDLING in the safety manual.

Effects

Implementation of the test procedure is limited to read accesses only.

Workaround

Performing a write access to test the safety mechanism is also a valid method. Users implementing these procedures shall consider the following wording instead of the original one: "The application software shall test this safety mechanism at least once per Multiple-Point Fault Detection Time Interval (MPFDTI) by reading from or writing to a register location where no register is located within the valid register space."

2.100 [SAFETY_TC.032] Wrong register name in GETH:ISR_MONITOR

Description

The ESM[SW]:GETH:ISR_MONITOR mentions in the paragraph "Unused interrupts" a register named Porti_MTL_PSF_Intr_Enable. This register however cannot be found in the Registers chapter of the GETH in the user manual.

Scope

Description of ESM[SW]:GETH:ISR_MONITOR in the safety manual.

Effects

User might try to disable unused interrupts on a register that cannot be found in the user manual.

Workaround

Users implementing ESM[SW]:GETH:ISR_MONITOR shall consider this register name instead: Portj_MTL_SGF_Intr_Enable.

2.101 [SAFETY_TC.033] Wrong alarm name and reaction for LMU REDUNDANCY SM

Description

The section "Reporting" of SM[HW]:LMU:CONTROL_REDUNDANCY wrongly states "the CPU generates a CPU TriCore™ lockstep mismatch alarm." In addition the section "Recommended error reaction" shows a note regarding default SMU reaction that is not relevant for this safety mechanism.

Scope

This erratum is valid for the AURIX™ TC4Dx Safety Manual V1.0, SM[HW]:LMU:CONTROL_REDUNDANCY.

Effects

Wrong alarm name and an out of scope note are shown.

Workaround

The reader shall ignore the wrong statement in the "Reporting" section and consider the following instead: "the LMU generates an LMU control redundancy mismatch alarm." In addition, the note in the section "Recommended error reaction" shall be ignored.

2.102 [SAFETY_TC.034] Missing double-bit error alarm for DMI and STU access to VDSP memory

Description

Data within Vector Memory (VMEM) within the DSP unit of ARC processors is protected by a Single Error Correction Double Error detection (SECDED) ECC logic . This is mentioned in the safety manual, under the topic Safety Mechanisms for the Parallel Processing Unit (PPU).

As stated in the safety mechanism **SM[HW]:PPU:VDSP_ERROR_CORRECTION**, the Single Bit Errors (SBE) and Double Bit Errors (DBE) detected in the data from VMEM within the DSP unit of ARC processor are expected to trigger an alarm to the SMU through the PPU alarm signals as follows :

- PPU_HS_SBE alarm : For SBE errors detected in VMEM
- PPU_HSSI_ERR alarm for DBE errors detected in VMEM

It was detected that DBE errors detected in VMEM during accesses from STU or VMEM-DMI do not trigger PPU_HSSI_ERR alarm.

Scope

DBE notifications from PPU during reads from VMEM to the PPU safety monitor.

Effects

DBE errors are uncorrectable. Any undetected DBE in the data results in a violation of the safety goal(s) of PPU.

Workaround

In addition to PPU_HSSI_ERR alarm, any DBE in PPU memories are also detected and notified by the VMT uncorrectable error alarm. The VMT alarms are not affected by the issue with PPU_HSSI_ERR alarm. Any DBE detected by VMT in the PPU VMEM triggers an uncorrectable error alarm from VMT. The system integrator shall use this alarm from VMT to detect DBEs in PPU VMEM and trigger a safety reaction from the SMU. Please refer to the VMT chapter of the User Manual for detailed description.

***Note**: PPU_HSSI_ERR also reports errors from other safety mechanisms in PPU, in addition to DBE in VMEM. Reporting of these errors continue to be triggered on PPU_HSSI_ERR and is not affected by this issue. Please refer to the safety manual for more details.*

2.103 [SCR_TC.026] I2C does not support multi-master systems

Description

The I2C module within the Stand-by Controller (SCR) subsystem offers a multi-master mode feature. The multi-master mode feature cannot be used as it is not fully functional. All other I2C features are usable.

Scope

Multi-master mode of the I2C module within the Stand-by Controller (SCR) subsystem.

Effects

Multi-master mode of the I2C module within the Stand-by Controller (SCR) subsystem is not fully functional, and therefore cannot be used.

Workaround

No workaround available.

2.104 [SCR_TC.029] I2C does not support Software reset (I2C_SRST.SRST)

Description

When the software reset feature of SCR I2C module is used in transmit or receive mode, a change in the bus state has to be expected (SCL and SDA are pulled to low). This happens when the module is operated either in the master or slave mode.

Scope

Software reset feature of I2C module within SCR subsystem.

Effects

I2C bus lines SCL and SDA are pulled to low.

Workaround

Do not use software reset feature inside SCR I2C module.

2.105 [SCR_TC.030] Incorrect baud rate after clock stretching

Description

If I2C module of SCR operates as master and the clock streching feature is requested by a slave the first clock period after the streching request appears shorter than the clock of the requested baudrate.

Scope

I2C module of SCR operating in master and clock streching feature is requested by a slave.

Effects

First clock pulse after clock streching request is shorten.

Workaround

Lower the baudrate in case of a slower slave to ensure proper communication.

2.106 [SCR_TC.032] [I2C] Slave does not return to IDLE state after receiving NACK in transmit mode

Description

Following sentence can be found in the user manual:

"Once the last byte to be transmitted has been loaded into the I2C_DATA register, the AAK bit should be cleared when IFLG is cleared. After the last byte has been transmitted, IFLG will be set and the I2C_STAT register will contain C8H. The I2C will then return to idle state (status code F8H). The AAK bit must be set to 1 before slave mode can be entered again."

However, this does not reflect the actual behavior of the silicon. In reality, when the software sets AAK=0 for the last byte to be transmitted, and the master does not acknowledge this byte, the status is C0H instead of C8H. After clearing IFLG, the I2C does return to idle state, and when the software sets AAK=0 for the last byte and the master does acknowledge it, the status is C8H and the I2C returns to idle state after clearing IFLG.

Furthermore the following sentence can be found in the user manual:

"If no acknowledge is received after transmitting a byte, IFLG will be set and the I2C_STAT register will contain C0H. The I2C will then return to idle state."

However, the behavior of the silicon is slightly different. When the software sets AAK=1, and the master does not acknowledge the data byte (which is the specified behavior of a master for the last data byte before it sends the STOP condition) the status is C0H but the I2C does NOT return to idle state. Instead, it waits for the STOP condition and flags another interrupt with status A0H.

Scope

I2C inside SCR subsystem.

Effects

User manual not reflecting the actual behavior of the silicon.

Workaround

For the first issue described above, the two cases have to be considered in the I2C software.

For the second issue described above, either do not set AAK=1, or consider the interrupt in software, and set module to idle state after interrupt.

2.107 [SCR_TC.034] [I2C] Slave does not return to IDLE state after sending NACK to the data byte of a general call

Description

In slave mode, in the case of a general call with multiple slaves, the slave device does not return to idle state when the IFLG is cleared to 0 after receiving status code 98_H (I2CSCR did transmit a not acknowledge bit to the received data byte) and the data byte was acknowledged by another slave. Instead it continues receiving the complete I2C frame until the STOP condition.

Scope

I2C communication

Effects

The I2CSCR continues receiving the I2C frame and will trigger an interrupt for every received byte and the STOP condition.

Workaround

The affected slave devices can be set back to IDLE state by clearing IFLG after every interrupt, via the user software. Alternatively, the user software can be designed to prevent setting AAK = 0 during a general call frame.

2.108 [SCR_TC.035] [OCDS] debugging of SCR reset does not work

Description

Standby-Controller (SCR) can be individually reset via various means (Host, Debugger, SCR internal event → WDT, ECC, SW) w/o reseting the debug-system. For debugging, it provides a monitor-program based solution, where the monitor-program communicates with the external tool. In case during a debug-session (tool attached, user software executed) the controller experiences a reset, a reconnect of tool and monitor-program is targeted. However, the latter is not correctly executed, and consequently the debug-session cannot be continued.

Scope

SCR-Resets excluding debug-system.

Effects

Monitor-program wrongly executed, no reconnect possible. XRAM content potentially unreliable.

Workaround

Reconfigure debug-system and continue debug-session. If required, restore XRAM-content via tool.

2.109 [SDMMC_AI.003] Nst timing violation for STOP command

Description

According to the JEDEC specification, Nst timing should be two cycles (one data cycle and end bit cycle) after the end bit of stop transmission command (CMD12). This timing is violated. The host controller stops the data traffic one cycle earlier than the protocol-mandated value of two cycles, due to which the end bit is missed.

Scope

This issue is seen when CMD12 is issued by the controller to stop a data transfer.

Effects

Non-Compliance with JEDEC specification for Nst timing.

Workaround

The user should use multiple block writes with a pre-defined block count rather than open-ended multiple block writes to avoid this issue. If an open-ended multiple block write is done, the Abort command (CMD12) must be sent after the traffic is completed, when NORMAL_INT_STAT_R.XFER_COMPLETE is asserted.

2.110 [SMM_TC.006] Port issue if port pin configured to reset at the end of reset

Description

Pads will stay in permanent reset state after system or application reset if the configuration RSTTRIGCTRLA.PINRST=1 for the reset shutdown sequence is used.

In case RSTTRIGCTRLA.PINRST is set after a system or application reset is triggered, the pins will not be available.

Scope

Pins and pads

Effects

Pads are in permanent reset state

Workaround

The issue can be resolved by triggering a warm power-on reset or higher, but effectively the configuration RSTTRIGCTRLA.PINRST=1 shall not be used.

2.111 [SMU_TC.016] FS timing status not updated

Description

If an alarm triggers the FSP protocol and a second (or more alarms) occur within 6 clock cycles of the first alarm, it could happen that the Fault State Timing Status x bit-field in SMU_SAFEx_STS.FxSTS is not set once the minimum timing duration of the FSP Fault State TFSP_FS is reached. Additionally, it could happen that the TFSP_FS timing is not re-started. Please note that, the first alarm does trigger the FSP protocol, only the status update is missed in the aforementioned corner case.

Scope

Fault State Protocol (FSP Protocol)

Effects

Missed status indication from SMU_SAFEx_STS.FxSTS; TFSP_FS timing not restarted by alarms arriving within the specified corner case timing.

Workaround

A possible workaround is to start a timer (external to SMU) when an FSP is triggered.

2.112 [SMU_TC.019] Missing P33.8 EMS A input in pin definition tables

Description

The datasheet is missing an entry related to Emergency Stop (EMS) Input A for all P33.8 in chapter 3, for all package types. In addition, the EMS input specified for P21.2 is missing the B designator.

Scope

Missing emergency stop signal at P33.8 as well as the corresponding designator at P33.8 abd P21.2.

Effects

Functionality present in design but missed in documentation.

Workaround

Consider that the function "Emergency Stop External Request A" is mapped to all P33.8 input pins under chapter 3. In addition, consider that the function "Emergency Stop External Request" mapped to P21.2 corresponds to "EMS External Request B".

2.113 [SRI_TC.004] Writes to reserved address space are overwriting DOM0_ACCEN registers with a similar offset from their base

Description

Writes to the reserved address region immediately above the OLDA address region may not result in error responses. The intended functionality should return errors on writes to reserved regions. Instead, writes to this reserved region have the potential to modify the registers of SRI0.SFR range. The unintended update occurs provided the access to the reserved address region aliases exactly on to an SFR and if a write is already permitted to that register either by the PROT or ACCEN depending on the protection mechanism applicable to that SFR. If a write is not permitted to the said SFR (respectively either by PROT to ACCEN, or ACCEN to other SFRs), an error is signaled and the update is prevented, which is the intended reserved region behavior.

Scope

SRI0 SFR registers, primarily the SRI0 APU and the registers protected by it as a secondary effect. SRI domains other than SRI0 and reserved regions other than the one immediately above the OLDA behave as documented.

Effects

Writes to SRI0.SFR aliases update the SRI0.SFR if write permissions to those registers are already enabled. This includes ACCEN when permitted by PROT, or to any other register in SRI0.SFR, when permitted by the ACCEN.

Workaround

There is no mechanism to prevent the condition. It is highly likely that the writes to reserved address regions would have been systematically pruned away during software development itself. However, in the event there remains a software risk, unintentional updates can be prevented by locking the PROT when an update to ACCEN is not required. Further, all other registers which are protected by ACCEN can be protected by disabling the access for all TAG-ID's in the ACCEN itself.

2.114 [TRIF_TC.001] TME target address needs to be 32-byte aligned

Description

Trace data in target memory needs to be 32-byte aligned.

Scope

Trace

Effects

64 bit alignment could result in data scrambling.

Workaround

Use 32 byte address alignment for the Trace targets.

2.115 [VMT_TC.004] Wrong register value shown in MBIST chapter

Description

In the VMT documentation, a wrong value is provided to re-enable SSH alarm generation in MCi_ALMSRCS. The provided value does not re-enable alarm generation for operational errors (ALMSRCS.OPENE == 0_B).

The wrong value is provided in three notes in the following sections:

- "MBIST cycle test phases"
  - Kx: Init
    - Step 2
      - Wrong:
        - Note: Only after this step, functional access to memories under test is available. Alarm generation shall be re-enabled through ALMSRCS == 27_H
      - Correct:
        - Note: Only after this step, functional access to memories under test is available. Alarm generation shall be re-enabled through ALMSRCS == 37_H
  - K3: Extract
    - Wrong:
      - Note: Alarm generation shall be re-enabled through ALMSRCS == 27_H
    - Correct:
      - Note: Alarm generation shall be re-enabled through ALMSRCS == 37_H
- "NDT for DLMU"
  - Wrong:
    - Note: Alarm generation shall be re-enabled through ALMSRCS == 27_H
  - Correct:
    - Note: Alarm generation shall be re-enabled through ALMSRCS == 37_H

Scope

VMT chapter of the User Manual, section "MBIST algorithms"

Effects

With ALMSRCS == 27_H (=> ALMSRCS.OPENE == 0_B), alarm generation for operational errors is not re-enabled.

Workaround

Use the correct value to re-enable alarm generation with ALMSRCS == 37_H (=> ALMSRCS.OPENE == 1_B).

2.116 [xSPI_TC.002] Clock low to CS high (tCKLCSH) timing JESD251C specification violation

Description

It is recommended to configure Tx DLL to 90° degree for center alignment of SPI Clock with Data. But by doing so, the JESD251C specification of CK Low to CS High gets violated.

Below are the actual minimum values for Clock low to CS high for the respective speed grades:

Table 17 Minimum values for Clock low to CS high (tCKLCSH CC)

Speed grade	symbol	Minimum value in ns
XSPI400	tCKLCSH CC	2.4
XSPI333	tCKLCSH CC	3.1
XSPI266	tCKLCSH CC	4.2
XSPI200	tCKLCSH CC	6.1

Clock LOW to CS HIGH timing parameter requires CS to be activated for certain amount of time after clock deactivation to make sure the external device has enough time to latch the input data in case of write or in case of read the external device provides the last data bit/byte to the host before CS goes high. As soon as CS is inactivated, the SPI devices ignores all the operation and bus becomes tri-stated. Violating this parameter affects last data frame getting latched in to external device during write or getting read by host during read.

Scope

xSPI DDR mode feature when used at full speed of external device.

Effects

Violating this parameter affects last data getting latched in to external device during write or getting read by host during read.

Workaround

1. Use external devices with relaxed CK Low to CS High specification
2. For external devices where CK Low to CS High specification is violated:
   - **a.** Use xSPI at next available lower speed-grade to work with external devices of a higher speed-grade class. For example, TC4Dx operated at xSPI333 can meet Clock Low to CS High specification of xSPI400 external device
   - **b.** For Protocols which uses clock generated by external device through data strobe pin (DS) for read, the full baud rate can be used for reading, using that external clock (DS) with the Rx DLL configured at 90° center alignment. For write operation in that case, half of the baud rate shall be used. Switching between read and write baud rate can be controlled using BAUDR.SCKDIV register field

3 Parametric deviations

3.1 [CCU_TC.P003] CLOCK Parameter changing (XTAL1 amplitude)

Description

The parameter “Input amplitude (peak to peak) at XTAL1" in the OSC_XTAL characteristics table should have the minimum value of 2.5 V for the External Clock Mode instead of -0.5 V.

Scope

The minimum value of the Input amplitude (peak to peak) at XTAL1 parameter.

Effects

Violation of the Input amplitude (peak to peak) at XTAL1 parameter.

Workaround

Use the 2.5 V minimum value for the Input amplitude (peak to peak) at XTAL1 parameter in the External Clock Mode.

3.2 [DAP_TC.P003] DAP pin driver strength not sufficient

Description

Data Valid Time specified in the data sheet for DAP pins cannot be achieved with the implemented PAD drivers under all conditions. Specified DAP frequency of 160 MHz cannot be reached depending on the tool hardware.

Scope

Debug DAP Interface

Effects

Specified DAP frequency of 160 MHz cannot be reached depending on the tool hardware and operating conditions.

Workaround

The board and tool hardware design shall minimize the capacitance. The tool shall determine the best sampling point for each response individually with the start bit. This automatically compensates temperature changes. Another option is using the sync telegram periodically. If this is not possible the specified valid time can be always achieved by reducing the frequency to 120 MHz.

3.3 [MSC_TC.P003] LVDS and SS timing update

Description

Timing parameter t_{45} of the MSC interface need to be relaxed for the following pad types:

- Fast pad, driver strong edge sharp, target: 3.5 ns, current achieved value: 5 ns
- LVDS pad, driver strong edge sharp, ABRA block bypassed, target: 5 ns, current achieved value: 10 ns
- LVDS pad, driver strong edge sharp, ABRA block used, target: 4 ns, current achieved value: 10 ns
- LVDS pad, driver strong edge medium, target: 10 ns, current achieved value: 17 ns

3 Parametric deviations

Scope

t_{45} timing of MSC interface

Effects

Violation of t_{45} parameter

Workaround

Low transmission speed.

4 Application hints

4.1 [ADC_TC.H047] Unexpected global service request after GLSRCFGx reconfiguration

Description

When reconfiguring the event enables for global service requests (GLSRCFGx.EVEN) in a running application, an additional service request can get generated.

Recommendation

Clear all event enables (TC4Dx: GLSRCFGx.EVEN[3:0]) before the intended configuration update.

4.2 [ADC_TC.H049] CHy_STAT register triggers arbitration issue

Description

If delay counter, triggered by HWTR1 expires at the same clock cycle as trigger pulse coming from HWTR0, then both hardware triggers are ignored and the ch_stat of the respective channel will be updated with the error information of HWTR1. Effect is observed after software disable of the trigger delay counter. This can happen not only in the minimum sampling time window as described, but also in the hold or conversion time window.

Recommendation

The corner cases caused by using the overlapping triggers must be considered in the application.

4.3 [ADC_TC.H051] Single EMUX input per TMADC instance

Description

A single channel per TMADC instance is qualified to be used as EMUX input. Bit 15 of TMADCx_CHy_CFG.EMUXEN shown as '1' in picture "TMADC EMUX control" in the user manual suggests, that multiple channels in one TMADC instance are qualified to be enabled at the same time. This is not the case.

Recommendation

For each TMADC instance, only one channel shall set its EMUXEN bit to 1.

4.4 [ASCLIN_TC.H010] Incomplete reset isolation

Description

Reset isolation is not completely implemented for the kernel or module reset if the module is in suspend or disable mode. As a result, reset domain crossing issues could be caused.

Recommendation

Software must ensure to exit the suspend or disable mode before requesting the kernel or module reset.

4.5 [CCU_TC.H019] Amplitude regulation in OSC module

Description

The MHz OSC design is slightly updated between TC4Dx AA and TC4Dx AB. The following two changes are implemented in AB step:

1. The gain step size between the different gain settings are linearized, equidistant and monotonic now
2. The gain settings value 01 and 10 are swapped, keeping the gain settings 00 and 11

With these changes, the adaption of the gain settings due to the used crystal is easier and more comprehensible.

No change for default gain setting 11.

Recommendation

Please see the description above.

4.6 [CDSP_TC.H001] Application reset impacts CDSP debug registers

Description

Application reset may impact CDSP debug register states.

Recommendation

Debug reset needs to be performed with each application reset.

4.7 [CDSP_TC.H002] Incomplete reset isolation

Description

Reset isolation is not completely implemented for module reset within CDSP. Module reset causes unintended write to data and instruction closely coupled memories (DCCM, ICCM) start addresses.

Recommendation

Reload DCCM and ICCM data after module reset or use software reset instead.

4.8 [CPU_TC.H025] Avoiding unbounded delays in store buffer residency

Description

In a multiprocessor system, sharing variables between different cores is the basis for implementing synchronization for programs running concurrently on multiple processors.

In a multiprocessor system, the following mechanism may be used to pass information between different cores:

- CPU_A
  - Writes to a shared variable V
- CPU_B
  - Waits for a change in the value of the shared variable V

Usually, a wait in CPU_B may be implemented as a polling loop.

System performance may be adversely impacted if the visibility by CPU_B of the write to the shared variable is delayed. In an extreme case, an unbounded delay may obstruct system progress.

Scenario 1: Store operation with possibly unbounded delay scenario 1

Consider a scenario where:

- CPU_A
  - Writes to a shared variable V and then
  - Performs a task that involves repeated loads from the same memory module
- CPU_B
  - Waits for a change in the value of the shared variable V

When CPU_A executes a store instruction targeting V, the information is recorded in a Store Buffer (see Store Buffers paragraph in the CPU chapter of AURIX™ TC2xx, TC3xx and TC4xx User Manuals). CPU_A will then attempt to complete the store information by starting the write access to the target memory where V is allocated.

This write access is subject to arbitration with other concurrent accesses to the same target memory location. If other accesses such as loads from the same CPU_A, or accesses from other masters like additional CPUs, DMA, etc., have higher priority, the write access may be delayed. This delays CPU_B's visibility of the update to V. In an extreme case, if the competing accesses are repeated indefinitely, the visibility will be delayed indefinitely, potentially leading to a significant disruption in CPU_B's operation.

To prevent such delay, a DSYNC[1)] instruction should be inserted in the CPU_A code following the write operation to V, and preceding the execution of repeated loads by CPU_A. This ensures timely visibility of the update to V by other CPUs.

Example code: Store operation with possibly unbounded delay

The following code snippet illustrates a scenario where a store operation may be delayed due to access arbitration with other loads. In this example, CPU_A executes a store instruction (ST.W) to variable V, followed by a generic instruction (INSTR_X) and a loop (L1) of load instructions (LD.W) that access variables R, S, and T.

These load instructions share access arbitration with the store operation to variable V, potentially causing a delay in the completion of the store operation. The loop L1 terminates when a specific value of variable R is read. The variable R is expected to be updated by another CPU (or DMA) in the system.

```
CPU_A_code:
    ST.W [V], Dv
    INSTR_X
L1:
    LD.W Dr, [R] 2)
    LD.W Ds, [S] 2)
    LD.W Dt, [T] 2)
    JNZ  Dr, L1
```

The following table illustrates the execution pipeline of CPU_A, highlighting the store operation to variable V and the subsequent loop of load instructions that access variables R, S, and T. The table also shows the contents of the store buffer, access arbitration, and memory access.

Table 18 Example execution: Store operation with unbounded delay

CPU_A: Start of execution pipeline	CPU_A: End of execution pipeline[1)]	CPU_ A: Store Buffer content	Access arbitration	Memory access
ST.W, store to variable V	-	-	-	-

(table continues...)

1 For TC1.8, the LSYNC instruction can be used as an alternative to DSYNC to ensure correctness. LSYNC is sufficient to prevent the delay to V's visibility and has a lower execution cost.

2 Assumption: access to the variables R, S, and T share access arbitration with the access to the variable V

Table 18 (continued) Example execution: Store operation with unbounded delay

CPU_A: Start of execution pipeline	CPU_A: End of execution pipeline[1)]	CPU_ A: Store Buffer content	Access arbitration	Memory access
INSTR_X	-	-	-	-
LD.W, load from variable R	ST.W [V]	Store to V	Store to V Load from R (higher priority)	-
LD.W, load from variable S	INSTR_X	Store to V	Store to V Load from S (higher priority)	Load from R
LD.W, load from variable T JNZ	LD.W [R]	Store to V	Store to V Load from T (higher priority)	Load from S
LD.W, load from variable R	LD.W [S]	Store to V	Store to V Load from R (higher priority)	Load from T
LD.W, load from variable S	LD.W [T] Conditional jump	Store to V	Store to V Load from S (higher priority)	Load from R
LD.W, load from variable T JNZ	LD.W [R]	Store to V	Store to V Load from T (higher priority)	Load from S
…	…	Store to V	Store to V Load from ... (higher priority)	Load from T
…	…	…	…	…

1) The number of cycles in the CPU execution pipeline is implementation dependent, see the core User Manual for details.

Key points:

- The store operation to variable V is delayed due to access arbitration with the load instructions
- The load instructions from variables R, S, and T have higher priority and win arbitration, causing the store operation to be delayed
- The store buffer content remains unchanged until the store operation is completed
- The memory access column shows the actual memory access that occurs, which is delayed due to access arbitration

It is noteworthy that the "Store to V" operation is not reflected in the Memory access column. This indicates that the store operation to variable V has not been completed, and the new value has not been written to memory.

While the iterations of loop L1 continue, the value in memory of V will not be updated by CPU_A. Consequently, if another CPU (CPU_B) were to access variable V, it would read the previous value of variable V. If CPU_B awaits a change in V's value, it may experience an unbounded delay due to the delayed completion of the store operation.

Scenario 1: Resolution with DSYNC instruction added

In this example, the DSYNC instruction is added after the store operation to variable V. The DSYNC instruction ensures the visibility of the update to variable V as the store buffer is emptied before commencing the load from variable R.

Example code: with DSYNC instruction added

The following code snippet illustrates the use of the DSYNC instruction to ensure that a store operation is visible before proceeding with subsequent memory instructions.

```
CPU_A code:
    ST.W [V], Dv
    DSYNC
    INSTR_X
L1:
    LD.W Dr, [R]
    LD.W Ds, [S]
    LD.W Dt, [T]
    JNZ  Dr, L1
```

The following table illustrates the execution pipeline of CPU_A, highlighting the use of the DSYNC instruction to ensure the visibility of the store operation to variable V, before the subsequent loop of load instructions that access variables R, S, and T.

Table 19 Example execution: with DSYNC instruction added

CPU_A: Start of execution pipeline	CPU_A: End of execution pipeline[1)]	CPU_ A: Store Buffer content	Access arbitration	Memory access
ST.W, store to variable V	-	-	-	-
DSYNC	-	-	-	-
(DSYNC) – waits for store buffer empty	-	-	-	-
(DSYNC) – waits for store buffer empty	ST.W [V]	Store to V	Store to V	-
INSTR_X	-	-	-	Store to V
LD.W load from variable R	DSYNC	-	Load from R	-
LD.W load from variable S	INSTR_X	-	Load from S	Load from R
LD.W load from variable T JNZ	LD.W [R]	-	Load from T	Load from S
…	…	-	Load from ...	Load from T
…	…	…	…	…

1) The number of cycles in the CPU execution pipeline is implementation dependent, see the core User Manual for details.

Key points:

- The store operation to variable V is followed by the DSYNC instruction, which ensures that the store operation is no longer in the store buffer
- The DSYNC instruction waits for the store buffer to be drained
- The load loop execution is delayed until the store operation to variable V is commenced, ensuring that CPU_B can observe the updated value of variable V in a timely manner

It is noteworthy that the "Store to V" operation is now reflected in the Memory access column. This indicates that the store operation to variable V has been commenced, and will be written to memory. Consequently, this ensures that CPU_B can observe the updated value of the variable V in a timely manner.

Scenario 2: Store operation with possibly unbounded delay

Consider a scenario where:

- CPU_A
  - Writes to a shared variable V and then
  - Performs a task that involves repeated loads from the same memory module
- CPU_B
  - Polls the shared variable V in a loop using a Read-Modify-Write instruction (for example CMPSWAP.W), waiting for a change in V value

Example code: Store operation with possibly unbounded delay

The following code snippets illustrate the interaction between CPU_A and CPU_B, highlighting the potential issues that can arise when accessing shared variables. The loop L1 terminates when a specific value of variable R is read. The variable R is expected to be updated by CPU_B. The loop L2 terminates when a specific value of variable V; expected to be updated by CPU_A, is observed.

```
CPU_A code:
    ST.W [V],Dv
    INSTR_X
L1:
    LD.W Dr, [R] 3)
    INSTR_Y 4)
    INSTR_Z 4)
    JNZ  Dr, L1

CPU_B code:
L2:
    [...]
    CMPSWAP.W [V], Ex
    JNE       Dx, Dz, L2
```

In this example, CPU_A executes a store instruction to variable V, followed by a loop (L1) containing a load instruction from variable R. The load instruction from variable R shares access arbitration with the accesses to variable V from both CPU_A and CPU_B. In the same loop (L1), CPU_A also executes two other instructions, INSTR_Y and INSTR_Z, which are not store instructions.

Meanwhile, CPU_B executes a loop (L2) containing a compare-and-swap atomic instruction (CMPSWAP) on variable V and registers Ex = [Dx, Dx+1], which checks if the content of V is equal to the content of register Dx+1 and if they are equal then swaps the contents of V with the register Dx. Register Dx is unconditionally updated with the contents of V. At the tail of the loop, if the value of register Dx and the value in register Dz are not equal CPU_B jumps back to label L2. If CPU_B is waiting for the store from CPU_A to update V to the value in Dz, it may experience an unbounded delay due to the delayed completion of the store operation.

3 Assumption: access to the variable R shares access arbitration with the access to the variable V
4 Assumption: neither INSTR_Y nor INSTR_Z is a store instruction

4 Application hints

The following table illustrates the execution pipeline of CPU_A, highlighting the store operation to variable V and the subsequent loop of the load instruction that accesses variable R. The table also shows the contents of the store buffer, CPU_B access to variable V, access arbitration, and memory access.

Table 20 Example execution: Store operation with unbounded delay

CPU_A, start of execution pipeline	CPU_A, end of execution pipeline [1]	CPU A, Store Buffer contents	CPU_B, access request	Access arbitration	Memory access
ST.W, store to variable V	-	-	-	-	-
INSTR_X	-	-	-	-	-
LD.W load from variable R	ST.W [V]	Store to V	-	Store to V Load from R (higher priority)	-
INSTR_Y	INSTR_X	Store to V	RMW, read from V (CMPSWAP.W)	Store to V RMW, read from V (higher priority)	CPU_A, Load from R
INSTR_Z	LD.W [R]	Store to V	-	Store to V Locked by RMW [2]	CPU_B, RMW read from V (previous value of V)
JNZ	INSTR_Y	Store to V	RMW, write to V (CMPSWAP.W)	Store to V Locked by RMW [2] RMW, write to V	-
LD.W load from variable R	INSTR_Z	Store to V	-	Store to V Load from R (higher priority)	CPU_B, RMW, write to V
INSTR_Y	Conditional jump	Store to V	RMW, read from V (CMPSWAP.W)	Store to V RMW, read from V (higher priority)	CPU_A, Load from R
INSTR_Z	LD.W [R]	Store to V	-	Store to V Locked by RMW [2]	CPU_B, RMW read from V (previous value of V)
JNZ	INSTR_Y	Store to V	RMW, write to V (CMPSWAP.W)	Store to V Locked by RMW [2] RMW, write to V	-
…	…	…	…	…	…

1) The number of cycles in the CPU execution pipeline is implementation dependent, see the core User Manual for details.
2) Read-Modify-Write (RMW) operation holds exclusive access to the memory for multiple cycles.

Key Points:

- In this example, CPU_B may wait forever for variable V to be updated by CPU_A because CPU_A's store to variable V may be continuously blocked by the combination of the RMW memory access from CPU_B and the load access from CPU_A itself
- This may lead to an unbounded delay in the progress of CPU_B which could lead to system unavailability

Scenario 2: Resolution with DSYNC instruction added

In this example, the DSYNC[5)] instruction is added after the store operation to variable V. The DSYNC instruction ensure the visibility of the update to the variable V as the store buffer is emptied before commencing the load from variable R.

The CPU_B code remains unchanged.

Example code: with DSYNC instruction added

```
CPU_A_code:
    ST.W [V], Dv
    DSYNC
    INSTR_X
L1:
    LD.W Dr, [R]6)
    INSTR_Y
    INSTR_Z
    JNZ Dr,  L1

CPU_B_code:
    [no change]
```

The following table shows the start and end of the execution pipeline for each instruction of CPU_A, as well as the contents of the store buffer, access requests from CPU_B to variable V, access arbitration, and memory access, highlighting the use of the DSYNC instruction to ensure that a store operation is completed before proceeding with subsequent instructions.

Table 21 Example execution: with DSYNC instruction added

CPU_A, start of execution pipeline	CPU_A, end of execution pipeline [1)]	CPU A, Store Buffer contents	CPU_B, access request	Access arbitration	Memory access
ST.W, store to variable V	-	-	-	-	-
DSYNC	-	-	RMW, read from V (CMPSWAP.W)	RMW, read from V	-
(DSYNC) – waits for store buffer empty	ST.W [V]	Store to V	-	Store to V Locked by RMW [2)]	CPU_B, RMW read from V (previous value of V)

(table continues...)

5 For TC1.8, the LSYNC instruction can be used as an alternative to DSYNC to ensure correctness. LSYNC is sufficient to prevent the delay to V's visibility and has a lower execution cost.

6 Assumption: access to the variable R share access arbitration with the access to the variable V

Table 21 (continued) Example execution: with DSYNC instruction added

CPU_A, start of execution pipeline	CPU_A, end of execution pipeline [1]	CPU A, Store Buffer contents	CPU_B, access request	Access arbitration	Memory access
(DSYNC) – waits for store buffer empty	-	Store to V	RMW, write to V (CMPSWAP.W)	Store to V Locked by RMW [2] RMW, write to V	-
(DSYNC) – waits for store buffer empty	-	Store to V	-	Store to V	CPU_B, RMW, write to V
INSTR_X	-	-	RMW, read from V (CMPSWAP.W)	RMW, read from V	CPU_A, Store to V
LD.W load from variable R	DSYNC	-	-	Load from R Locked by RMW [2]	CPU_B, RMW read from V (updated value of V)
(LD.W) – waits for access	INSTR_X	-	-	Load from R Locked by RMW [2] RMW, write to V	-
(LD.W) – waits for access	-	-	-	Load from R	CPU_B, RMW, write to V
INSTR_Y	-	-	-	-	CPU_A, Load from R
INSTR_Z	LD.W [R]	-	-	-	-
JNZ	INSTR_Y	-	-	-	-

1) The number of cycles in the CPU execution pipeline is implementation dependent, see the core User Manual for details.
2) Read-Modify-Write operation holds exclusive access to the memory for multiple cycles.

In this example, the visibility of the store to the variable V is enforced by using a DSYNC instruction before CPU_A proceeds with the load loop instruction

Key Points:

- The store operation to variable V is followed by the DSYNC instruction, which ensures that the store operation is no longer in the store buffer
- The DSYNC instruction waits for the store buffer to be drained
- The load loop execution is delayed until the store operation to variable V is commenced, ensuring that CPU_B can observe the updated value of variable V in a timely manner

Conditions for unbounded delay for store completion

The completion of a store may have an unbounded delay if all of the following conditions are met.

Condition set:

- The store is followed by sequence of load instructions such as a load or loads executed in a loop **AND**
- There is neither a DSYNC instruction nor an atomic memory operation executed after the store operation **AND**
- The load operation competes in internal CPU arbitration against the store operation **AND**
- There are no store operations executed between those loads (*)

***Note**: This applies not only to the store (ST*) instruction but also to any instruction that involves a store operation. In particular, as described in the TriCore™ Architecture Manual (Context Switching for Function Calls), a CALL instruction stores registers to a CSA, therefore it qualifies as a "store operation" with respect to these conditions.*

The following code examples demonstrate the conditions for delayed store completion. These examples are designed to aid in determining whether a particular code sequence is vulnerable to potential excessive delay in store completion. Examples are given of both code sequences with potential delay (requiring a DSYNC addition, or other modification for timely completion), and of code sequences where no excessive delay occurs (no code modification is required).

Example 1: code with potential delay of store completion

The following code example illustrates a scenario where the completion of a store operation may be delayed due to the conditions outlined in the main section.

Code example:

```
    ST.W [V], Dv
    MOVH Ax, …
    LEA  Ax, Ax, …
L1:
    LD.W Dr, [R]
    LD.W Ds, [S]
    LD.W Dt, [T]
    JNZ  Dr, L1
```

The location of the variables in this case is:

- V, R, S, and T are in the CPU's own DSPR **OR**
- V, R, S, and T are in the CPU's own DLMU **OR**
- V, R, S, and T are outside the CPU's own DSPR/DLMU

In this example, a store operation is executed to variable V, followed by a sequence of load instructions that access variables R, S, and T. The load instructions are executed in a loop, and the load operations compete in internal CPU arbitration against the store operation.

All the conditions listed in the condition set are true in this example. Due to these conditions, the completion of the store operation to variable V may be delayed.

Example 1 recommended solution: Using DSYNC instruction

The following code example illustrates the recommended solution to ensure timely completion of store operations, particularly in scenarios where the conditions outlined in the condition set are true.

Code example:

```
    ST.W [V], Dv
    DSYNC
    MOVH Ax, …
    (alternative DSYNC location)
    LEA Ax, Ax, …
    (alternative DSYNC location)
L1:
    LD.W Dr, [R]
    LD.W Ds, [S]
    LD.W Dt, [T]
    JNZ  Dr, L1
```

The DSYNC instruction, inserted after the store operation, ensures the store buffer is empty of all the preceding stores. The store to V is completed in a timely manner. The DSYNC instruction can be inserted at alternative locations, such as after the MOVH or LEA instructions, to ensure that the store buffer is empty before proceeding with subsequent instructions.

Example 2: code with no unbounded delay

The following code example illustrates a scenario where the completion of a store operation is not affected by the conditions outlined in the condition set; to be clear there is no unbounded delay.

Code example:

```
    ST.W [V], Dv
    MOVH Ax, …
    LEA Ax, Ax, …
L1:
    LD.W Dr, [R]
    LD.W Ds, [S]
    LD.W Dt, [T]
    JNZ  Dr, L1
```

The location of the variables in this case is:

- R, S, and T are in the CPU's own DSPR, and
- V is in another CPU's DSPR

In this example, a store operation is executed to variable V, followed by a sequence of load instructions that access variables R, S, and T. However, the loads from R, S, and T do not compete in arbitration against the store to V, as R, S, and T are located in different DSPRs. As a result, the completion of the store operation to variable V is not affected by the loads from R, S, and T. The store operation is completed in a timely manner.

Example 3: code with no unbounded delay

The following code example illustrates a scenario where a store operation is executed to variable V, followed by a sequence of load instructions that access variables R, S, and T. However, the sequence of repeated operations also contains a store instruction to variable U.

Code example:

```
    ST.W [V], Dv
    MOVH Ax, …
    LEA Ax, Ax, …
L1:
    LD.W Dr, [R]
    LD.W Ds, [S]
    LD.W Dt, [T]
    ST.B [U], Du
    JNZ  Dr,  L1
```

In this case example the sequence of repeated operations contains a store instruction. The store buffer quickly fills up when stores are executed in a loop. Once the store buffer is full, the CPU waits for at least one store to exit the store buffer before executing the next store instruction. Therefore, in this case there is no unbounded delay in the completion of stores.

Example 4: code with no unbounded delay

The following code example illustrates a scenario where a store operation is executed to variable V, followed by a sequence of load instructions that access variables R, S, and T. However, the sequence of repeated operations also contains a function call.

Code example:

```
    ST.W [V], Dv
    MOVH Ax, …
    LEA Ax, Ax, …
L1:
    LD.W Dr, [R]
    LD.W Ds, [S]
    LD.W Dt, [T]
    CALL foo
    JNZ  Dr, L1
```

In this example the sequence of repeated operations contains a function call (which requires a CSA save operation). This will cause the store buffer to fill up and as in the previous example, the store operation to V will exit the store buffer in a timely manner.

Example 5: code with unbounded delay in store completion

The following code example illustrates another scenario where the completion of a store operation may be delayed due to the conditions outlined in the condition set.

Code example:

```
    ST.W [V], Dv
    MOVH Ax, …
    LEA  Ax, Ax, …
L1:
    LD.W Dr, [R]
    ADD  Dx, 1
    JNZ  Dr, L1
```

The location of variables in this case is:

- V and R located in CPU's own DSPR **OR**
- V and R located in CPU's own DLMU

***Note**:* *It is assumed that R could be updated by an event (originating from an external source to the CPU_A itself).*

In this example, a store operation is executed to variable V, followed by a sequence of load instructions that access variable R. The sequence of repeated operations is executed in a loop, and the the load operations compete in internal CPU arbitration against the store operation. All the conditions listed in 1.4 are true, consequently the completion of the store to V may be delayed. In this case an unbounded delay may occur if there are concurrent accesses from other CPUs (or DMAs) accessing the same memory.

In general, as the exact pattern of behavior of other CPUs, DMA, etc. may be difficult to ensure, it is recommended to add a DSYNC instruction to such code sequences. This will ensure that the store operation exits the store buffer before proceeding with subsequent instructions, eliminating the delay to visibility of the updated value.

Recommendation

In summary, the Store Buffer mechanism is an important feature of TriCore™ and its existence is mostly transparent to uni-processor code. However, when multiprocessor synchronization is to be performed, it requires careful consideration and management to prevent unexpected behavior. As described above, by using the DSYNC instruction to flush the Store Buffer and ensure the completion of store operations, developers can guarantee that shared resources are updated correctly and in a timely manner, ensuring the correct behavior of the system.

4.9 [CPU_TC.H026] Spurious lockstep error triggered on TriCore™ if DSPR is accessed by CPU or slave before DSPR-MBIST

Description

Executing an MBIST operation on the DSPR can later cause the CPU to trigger a spurious lockstep alarm. This alarm is spurious in that it is not indicative of a real fault. The spurious alarm does not happen during the MBIST operation, but rather when the DSPR is later accessed over the bus (via slave i/f).

For the spurious lockstep alarm not to happen, the two following points must be adhered to:

1. No access to DSPR by CPU (pipeline or slave i/f) between the latest system reset and the MBIST operation on DSPR.
2. The last sequence of the MBIST must read all zeros (data and checker bits) from the DSPR.

In any case, note that issuing an application reset after MBIST clears the issue.

Note that this problem only affects the DSPR, so an application carrying out an MBIST operation on the PSPR or DLMU will not see spurious alarms in the same circumstances. It also does not affect non-lockstep CPUs. Also note that the volatile memory test (VMT) is not affected by this because it complies with this APP-HINT.

Recommendation

When doing an MBIST operation on DSPR, it is recommended to not access DSPR beforehand and also to read zeros out of the last MBIST operation. If the DSPR has to be accessed beforehand, then CPU must be issued an application reset after the MBIST in order to avoid this alarm. Where possible, we would recommend doing those kind of operations on the PSPR/DLMU rather than the DSPR in order to avoid these considerations.

4.10 [DRE_TC.H002] Throughput performance drop for GETH or LETH to LETH forwarding via DRE

Description

Frame loss at Ingress MAC port occurs in the following conditions:

- Ingress MAC port (GETH or another instance of LETH) forwards frame to LETH MAC port via DRE and
- The ingress MAC port receives sustained burst of traffic matching the ideal maximum line throughput of the egress LETH MAC port

The frame loss is due to data path between DRE and LETH not able to sustain the bandwidth of the ingress burst traffic. For each forward request on a given EIBUF of DRE, the request is marked as completed only after the corresponding Ethernet frame is successfully transferred at the LETH MAC ports and acknowledged by LETH DMA channel write-back. Only after this acknowledge the next Ingress MAC frame is triggered for transfer from Ingress MAC DMA channel to the DRE EIBUF. This causes the receiving sustained burst traffic to queue within RxQ of Ingress MAC port and frames will be dropped when the RxQ buffer is full.

The duration of sustained burst traffic without frame loss depends on various factors like parallel forward request between multiple Pairs of Ethernet MAC port in DRE, SRI transactions from other participants, operating clock frequency, configuration of Ethernet DMA channels to perform burst transactions.

Therefore, the following use-case described must be taken only as a reference but not as an absolute quantification of performance loss:

- 64-byte frames at 100 Mbps of a GETH ingress port is forwarded via DRE to a 100 Mbps LETH egress port
- Maximum operation clock frequency for fGETH (250 MHz), fDRE (500 MHz) and fLETH (250 MHz)
- No other parallel traffic in SRI and no other activity of GETH, LETH & DRE

- (Ingress) GETH configured for
  - RxQ size: 8 KByte
  - Rx DMA channel Burst length: 16
- (Egress) LETH configured for
  - TxQ size: 8 KByte
  - Tx DMA channel Burst lengt: 16

For the above use-case, there is no loss of frame until the 268th receive packet and the net Ingress bandwidth without frame loss is approximately 81 Mbps.

Recommendation

The system integrator is expected to analyze the traffic nature of GETH or LETH to LETH forwarding use-case of DRE and the following recommendations are to be considered for enhancing the capability to sustain longer burst traffic:

- Configure Ingress and Egress Ethernet DMA channel for maximum burst transactions on SRI
- Increased FIFO size allocation for RxQs of Ingress MAC port
- Operate at maximum clock frequency of fGETH, fLETH and fDRE

4.11 [GETH_AI.H005] The MMC_Rx_Packet_SMD_Err_Cntr counter incorrectly updated in internal Loopback mode

Description

The MMC_Rx_Packet_SMD_Err_Cntr counter is updated when SMD error is detected for the received preemption packets. However, due to the defect, when the MAC is operated in internal Loop back mode, although the MAC receiver correctly identifies the end of the packet, the MAC preemption receiver incorrectly identifies the additional words read as SMD error and therefore updates the MMC_Rx_Packet_SMD_Err_Cntr counter incorrectly.

Recommendation

Software must ignore the updates to the MMC_Rx_Packet_SMD_Err_Cntr counter during internal Loopback mode.

4.12 [GETH_AI.H006] Packets received in the highest numbered queue not sent to the software after the packet to be duplicated to multiple DMAs is dropped

Description

When a received packet routed to the highest numbered queue is identified for duplication to multiple DMA packet is forwarded only to the enabled DMAs from the identified set. If any of the DMAs from the identified set is disabled, the packet is considered as dropped for that DMA. The subsequent packets are forwarded as usual to the software through specified DMAs.

However, due to the defect when the packet to be duplicated to multiple DMAs is dropped for any disabled DMAs, the subsequent packets from the queue are not sent to the software. This is because the header status for first packet, after the duplicated packet, is not sent to the DMA. The DMA uses the information in the header status to request correct amount of data (Programmable Burst Request or PBL) from the MTL Receive Queue. Since the required information is not available, DMA places the request for an incorrect amount of data and the MTL does not respond.

Recommendation

Software must ensure that all the programmable fields that specify the DMA numbers to which packets are duplicated, does not include any disabled DMA channels.

4.13 [GETH_TC.H007] Incomplete reset isolation

Description

Reset isolation is not correctly implemented for the interrupts, alarms, and PPS signals going from GETH to other blocks (IR, SMU, GTM/eGTM) that could cause potential reset domain crossing issues.

Recommendation

Software must ensure that before applying a kernel reset, module group reset or software reset (SWR bit of the DMA_Mode register):

1. All TX and RX transactions must be finished
2. MAC transmitter and receiver must be disabled
3. PPS generation must be disabled or stopped, disabling alarms and interrupts. If it is ensured that there is no on-going transaction then it is not needed to disable interrupts and alarms

***Note**:* *If GETH is part of a module reset group, the described workaround needs to be applied before the module group reset is asserted. This is possible by using the module group reset trigger not directly as a reset trigger in SMM but rather as a trigger for an ISR which prepares GETH as described here and requesting the module group reset by software in SMM after successful preparation of GETH.*

4.14 [HSPHY_TC.H004] Problem with clk_sram during application reset

Description

There is a remote possibility that a false HSPHY data uncorrectable error alarm could be generated by the HSPHY SRAM control logic after an application reset or system reset and before the re-enabling of the HSPHY under the following conditions:

1. A PHY is transmitting TRACE data and an application reset or system reset has been asserted and the HSPHY not yet been re-enabled (CLC.DISR = 1)
2. A PHY is being used for PCIE or Ethernet communication and an application reset or system reset has been asserted and the HSPHY not yet been re-enabled (CLC.DISR = 1)

In gate level simulation (GLS) unknown states ('X' states) have been seen to propagate through the SRAM support logic (SSH) as alarms to the Safety Management Unit (SMU). It cannot therefore be ruled out that these are detected as alarm events.

Recommendation

Reset all PHYs involved in mission mode data transfer (PCIE and Ethernet), immediately prior to an application reset or system reset by set the corresponding CTRL1.RSTx to 1, where x = PHY Index.

Please note that if it is necessary to capture TRACE data up to and after an application reset or system reset, this solution cannot be used for the TRACE related PHY. It is then recommended to disable the SRAM alarm registration from the TRACE associated PHY going to the Safety Management Unit (SMU). TRACE transmission can only be executed in dedicated emulation devices and not in mission mode so masking of these TRACE alarms can be regarded as safety irrelevant.

4.15 [HSPHY_TC.H007] Missing register description in the user manual to configure Rx adaption control

Description

The following register and its corresponding configuration is not described in the UM.
Register Short Name: XPCSi_VR_XS_PMA_MP_12G_AFE_DFE_EN_CTRL (i=0-1)
Register Short Description: VR XS or PMA Multi-protocol 12G PHY AFE-DFE Enable Register
Offset address:0860174_H+i*800000_H
Register Width: 32 bits
Reset Value: 00000011_H
Bit-fields:

- Bit-field: AFE_EN_0
  - Bit-field Offset: 0
  - Bit-field width : 1
  - Bit-field Access: rw
  - Description : This bit can be set to enable Rx adaption circuitry
- Bit-field: DFE_EN_0
  - Bit-field Offset: 4
  - Bit-field width : 1
  - Bit-field Access: rw
  - Description : This bit can be set to enable Rx adaption and decision feedback equalization (DFE) circuitry
- Other bit-fields are Reserved - Read as 0_B and shall be written with 0_B

Recommendation

For SGMII 100M at 125 Mbps serial line rate, SGMII 100M or 1G mode at 1.25 Gbps serial line rate and SGMII 2.5G mode at 3.125 Gbps serial line rate, write value of 0_B to AFE_EN_0 and DFE_EN_0 bit-fields of XPCSi_VR_XS_PMA_MP_12G_AFE_DFE_EN_CTRL register as part of the PHY register configuration in Programming sequence description.
For USXGMII 5G mode (Base-R), default reset values of 1_B for AFE_EN_0 and DFE_EN_0 bit-fields apply.

4.16 [HSPHY_TC.H008] Incorrect Initialization sequence for SGMII leading to alarms

Description

The HSPHY chapter of TC4Dx User Manual V1.0 describes for the programming sequence of Ethernet SGMII 100M, 1G and 2.5G mode that RX_RST bit of VR_XS_PMA_MP_12G_16G_25G_RX_GENCTRL1 register is to be asserted before the PCS speed selection is configured. This reset on some corner cases leads to uncorrectable safety alarm triggered by HSPHY.

Recommendation

In the programming sequence of HSPHY initialization for Ethernet SGMII 100M, 1G and 2.5G mode, do not execute the step which asserts and de-asserts the RX_RST_0 bit of VR_XS_PMA_MP_12G_16G_25G_RX_GENCTRL1 register.

4.17 [HSSL_TC.H002] Incomplete reset isolation

Description

Reset isolation is not correctly implemented. As a result, reset domain crossing issues could be caused and the HSSL module will not be in the expected state after a kernel reset or module group reset. Potential effects could be:

- Unexpected HSSL module function
- Unexpected register values

Recommendation

Software must ensure that the HSSL module clock is disabled (CLC.DISR=1 and CLC.DISS=1) before issuing kernel reset or module group reset.

4.18 [LETH_AI.H001] Combination of forwarding and reception to host on Port0 does not works as expected

Description

LETH bridge provides functionality to map the transmit queues from different MAC ports to the host DMA path or to forwarding path to other MAC ports that is configured via registers PORTj_RXC_MAP and PORTj_TXQ_MAP and PORTj_CTRL_REG enables the required RXCs and transmit queues.

There can be unintended behavior of packet transfer towards host or to forwarding path if above mentioned registers are not programmed in correct order.

Recommendation

For the proper operation of receive packet transfer to host and to the forward path, PORTj_CTRL_REG register must be programmed after the PORTj_RXC_MAP and PORTj_TXQ_MAP registers.

4.19 [LETH_TC.H001] Incomplete reset isolation

Description

Reset isolation is not correctly implemented for the interrupts, alarms, and PPS signals going from LETH to other blocks (IR, SMU, or eGTM) that could cause potential reset domain crossing issues.

Recommendation

Software must ensure that before applying a kernel reset, module group reset or software reset (SWR bit of the DMA_Mode register):

1. All TX and RX transactions must be finished
2. MAC transmitter and receiver must be disabled
3. PPS generation must be disabled or stopped, disabling alarms and interrupts. If it is ensured that there is no on-going transaction then it is not needed to disable interrupts and alarms

***Note**: If LETH is part of a module reset group, the described workaround needs to be applied before the module group reset is asserted. This is possible by using the module group reset trigger not directly as a reset trigger in SMM but rather as a trigger for an ISR which prepares LETH as described here and requesting the module group reset by software in SMM after successful preparation of LETH.*

4.20 [LETH_TC.H002] DMA channel locks up when time stamping is enabled and context descriptor is not available

Description

The Receive DMA writes the packet data to buffers pointed by the Normal descriptor and updates the receive packet status in the last Normal descriptor for that packet. If timestamp is captured for the packet, TSA bit in RDES1 of last Normal descriptor is set to 1 indicating that the captured timestamp status is available in the next descriptor.

The captured timestamp is updated in the next descriptor which is called Context descriptor. The Receive Queue can have packets intended to multiple Receive DMA. Therefore, when any Receive DMA is stopped or suspended due to unavailability of descriptors, the corresponding packets can be flushed to prevent head of line (HOL) blocking in the Receive Queue. The complete or remaining partial packet, along with status is flushed, only when the RPF bit in the DMA_CHy_Rx_Control register is programmed to 1_B.

The Receive Queue Read Controller provides the receive packet status and timestamp status, if available, along with the last burst of packet data, that is, no separate request is placed by Receive DMA. The timestamp status is provided irrespective of whether there is descriptor available for updating it as Context descriptor.

However, when the received packet data is forwarded to application, but descriptor is not available to update the captured timestamp status in Context descriptor, the timestamp status launched by MTL Receive Queue Read Controller on DMA-MTL interface is neither accepted by the Receive DMA nor flushed. This is because when the Receive DMA enters suspend state, it does not issue flush when only the timestamp status is remaining to be accepted. This results in the DMA-MTL interface parked with the Receive DMA until the descriptors are made available, preventing transfer of packets to any other active Receive DMA.

Recommendation

The software must create one additional descriptor when Receive Buffer Unavailable interrupt is generated with AIS and RBU status bits in DMA_CHy_Status register is set to 1_B, and final descriptor updated contains TSA bit in RDES1 Normal descriptor and LD bit in RDES3 Normal descriptor are set to 1_B.

4.21 [MCDS_TC.H008] Trace start from halt after reset state with two MCDS instances

Description

For two MCDS instances, the trace_done trigger is cross coupled for a synchronized trace recording. This ensures a common trace_done irrespective of which instance detected the trigger. This approach works fine for triggers which are getting active after the trace start. However, there is an issue with an OCDS break_in trigger artifact when the trace is started, and concurrently OCDS suspend is released. This concurrency is needed for reproducible time counter values and it is achieved by writing the TRIF_CTRL register bits TRCSTART and TRCSUSP concurrently. Due to the delay of the cross triggering, an artificial pulse is immediately activating trace_done at the other MCDS instance.

Recommendation

Do not cross couple the break_in trigger to the other MCDS instance. Just use break_in at both MCDS instances locally for trace_done generation.

4.22 [MCDS_TC.H009] Fan comparators do not trigger in case of multiple discontinuities

Description

The MCDS Instruction pointer comparators do not always trigger in case of discontinuities in consecutive CPU clock cycles.

Recommendation

This behavior results from the clock differences between CPU and MCDS trace module. Discontinuities in consecutive CPU clock cycles under certain conditions might happen. One condition is the phase relationship of the two clocks. The same IP trigger can hit or fail depending on which of the two CPU clocks within the single MCDS clock it occurs.

The debug trigger logic inside the CPU runs with full frequency and can be used to confirm such situation.

Consider the following options:

- Move the trigger to a nearby sequential instruction address
- Use the CPU debug logic to trigger (effective trigger trace is delayed)
- Use the CPU debug logic to check for missed triggers if the MCDS IP trigger never hits

4.23 [MCMCAN_TC.H014] Incomplete reset isolation

Description

Reset isolation is not correctly implemented. As a result, reset domain crossing issues could be caused.

Recommendation

Software must ensure that before issuing a kernel reset or a module group reset:

- MCMCAN module is not in OCDS suspend mode or in sleep mode and also no suspend or sleep is being requested during the reset
- The Ni_STARTADR and Ni_ENDADR range registers are initialized to zero
- CRE is disabled (Ni_CRE_CONFIG.EN = 0) for all CAN nodes, and then 5 SPB clock cycles shall be waited before issuing the reset to ensure that there is no DRE trigger

***Note**: In case of a module group reset, if the group includes DRE, then CRE does not need to be disabled.*

4.24 [MCMCAN_TC.H015] CRE ERRCTRL Register Wrong Access Mode

Description

In MCMCAN we have two kinds of access protection units (APU):

- APU-P4: Generic APU protecting the generic module registers
- APU-PNi: Node specific APU protecting the node specific registers

The APU-P4 was wrongly assigned to the Ni_ERRCTRL which is a node specific register.

Software must be aware of this inconvenience and account for it when performing register accesses.

Recommendation

In case the software uses two separate partitions for register initialization, then in the common partition where all generic module registers are initialized, the Ni_ERRCTRL register must also be initialized for the active nodes (i.e., those with an enabled clock via MCR register). To avoid RHBUF empty interrupts, the watchdog timeout monitoring event group 2 won´t be enabled and therefore the Ni_ERRCTRL.WDG2 will remain unset.

4.25 [MSC_TC.H016] Incomplete reset isolation

Description

Reset isolation is not correctly implemented for the suspend and disable acknowledge signals. As a result, reset domain crossing issues could be caused.

Recommendation

Software must ensure that the MSC module clock is disabled (CLC.DISR=1 and CLC.DISS=1) before issuing a kernel reset or a module group reset.

4.26 [PACKAGE_TC.H009] Max. compressive force during customer housing mounting

Description

The maximum compressive force during customer housing mounting for F2(H)BGA-292 is 65N and for F2(H)BGA-436 is 100N.

PCB and its housing should be designed in a way to prevent bending or flexing of the PCB when applying a compressive top-side force on the BGA package.

Recommendation

Please see the description.

4.27 [PCIE_AI.H001] In non-D0 power state, the controller does not handle the Cpl TLP as an unexpected completion

Description

In non-D0 power states, the PCIe controller logs the Completion TLPs as unexpected completions in the header log and passes the completions to the application as good TLPs. This is an optional check implemented by the controller. The issue will occur if the controller enters a a non-D0 power state while CPLs are outstanding. TLP completions are handled as if the controller is in the D0 state and sent to the application as good TLPs.

Recommendation

No workaround is available. The condition can be detected if unexpected CPL logging (unexp_cpl_err) is enabled via RADM filtering.

4.28 [PCIE_TC.H002] Region upper address register (RGNUA) value is evaluated as "less or equal" instead of "strictly less than"

Description

The APU upper address S2A_ACCENy_RGNUA (y=0-7) is evaluated as "less than or equal to" instead of "strictly less than". This leads to the upper addresses of ATU and APU not being exactly aligned.

Recommendation

There are the two possible scenarios that can arise with ATU and APU alignment in correlation with the S2A_ACCENy_RGNUA (y=0-7) address evaluation. Customers are requested to choose the preferred workaround as per their application requirements.

Table 22 Options for best possible alignment of APU upper address to ATU granularity

Workaround	Options for best possible alignment of APU upper address to ATU granularity of 4kB		ATU region upper bound address	Effect
1	When a register of the following array S2A_ACCENy_RGNUA (y=0-7) is configured as 0000 1000_H [1]	Corresponding offset address of the corresponding register in array S2A_ACCENy_RGNUA (y=0-7) is translated to: (4095+1) [2]	Bit fields [11:0] of register IATU_LIMIT_ADDR_OFF_OUTBOUND_i (i=0-7) are hardwired to 1111 1111 1111_B this is translated to a minimum granularity of 4096	The APU allows accesses to address offset 4096 for which no ATU translation of this region is available. Those accesses pass through this ATU region untranslated and if not matched by another ATU region are carried forward untranslated onto the wire.
2	When a register of the following array S2A_ACCENy_RGNUA (y=0-7) is configured as 000 $0FC0_H$ [1]	Corresponding offset address of the corresponding register in array S2A_ACCENy_RGNUA (y=0-7) is translated to: (4095+1)-64 [2]	Bit fields [11:0] of register IATU_LIMIT_ADDR_OFF_OUTBOUND_i (i=0-7) are hardwired to 1111 1111 1111_B this is translated to a minimum granularity of 4096	APU will not allow accesses to upper 63 bytes of the ATU region.

1) The lower 6 address bits of S2A_ACCENy_RGNUA (y=0-7) are reserved to zero
2) Where '+1' factor is added due to the "less than or equal to" evaluation of the S2A_ACCENy_RGNUA (y=0-7)

4.29 [PCIE_TC.H003] Interrupt detection should clear its internal status

Description

When more than one status bit represent an interrupt, upon this interrupt being serviced, each of the status bits are not cleared automatically.

Recommendation

Some of the interrupts may have up to 2 status bits. In this case, Software should reset all relevant register status bits for the corresponding interrupt after it is being serviced. Otherwise, a new PCIE interrupt could be pending and would not be reflected in the interrupt status bits.

4.30 [PCIE_TC.H005] Register write error response is not evaluated

Description

Error response for DBI writes are not being propagated. On a reset event like hot reset, PERST or a link reset, a link interrupt is triggered to notify the application. In such an event, not all PCIE registers are accessible until

the reset sequence is completed. Accessing these registers for a write may lead to errors which may go unnoticed.

Recommendation

In order to avoid this, user must poll the following registers in the described order. Any reset event (hot reset, PERST or others) which would reset the PCIE controller, will trigger a link interrupt.

The registers will be accessible again after the two steps below:

1. Wait until LNK_DVCTYP.PWD becomes 0 (recommended timeout 40 us)
2. Wait until PM_STAT0.LTSSMSTAT has a value greater than 0 (recommended timeout 400 us)

4.31 [PCIE_TC.H006] Incomplete reset isolation

Description

Reset isolation is not correctly implemented for internal enable signals. As a result, reset domain crossing issues could be caused.

Recommendation

Software must ensure that the PCIE module clock is disabled (CLC.DISR=1 and CLC.DISS=1) before issuing module group reset. The following steps are recommended:

1. Disable clock (CLC.DISR=1)
2. Check/Poll status of clock for being disabled (CLC.DISS=1)
3. Issue reset
4. Enable clock (CLC.DISR=0)
5. Check/Poll status of clock for being enabled (CLC.DISS=0)
6. Check/Poll status of reset for being executed

4.32 [PCIE_TC.H007] Outbound traffic is not blocked in hardware when MSE=0 nor when BME = 0

Description

When BME (register STATUS_COMMAND_REG, bit-field PCI_TYPE0_BUS_MASTER_EN) is set to 0_B in the EP (endpoint) mode, and when MSE (register TYPE1_STATUS_COMMAND_REG, bit-field MSE) is set to 0_B in the RC (root complex) mode, the outbound traffic will not be blocked in hardware.

Recommendation

Application software should do one of the following:

- Make sure the BME field is set to 1_B before initiating outbound traffic in EP mode and that the MSE field is set to 1_B before initiating outbound traffic in RC mode
- Do not initiate outbound traffic

4.33 [PMS_TC.H020] Standby RAM address missing in PMS documentation

Description

Standby RAM start and end addresses are missing.

Recommendation

The Standby RAM blocks can be enabled by setting the PMS_STANDBY_CON0.SBRAMSEL bit-field. For both CPU0 and CPU1 64 Kbyte, 2x64 Kbyte or both Standby RAM blocks can be kept supplied during Standby mode.

If CPU0 64 Kbyte Standby RAM block kept supplied during Standby mode the address range is from 90000000_H to $9000FFFF_H$.

If CPU0 2x64 Kbyte Standby RAM block kept supplied during Standby mode the address range is from 90000000_H to $9001FFFF_H$.

If CPU1 64 Kbyte Standby RAM block kept supplied during Standby mode the address range is from 90080000_H to $9008FFFF_H$.

If CPU1 2x 64 Kbyte Standby RAM block kept supplied during Standby mode the address range is from 90080000_H to $9009FFFF_H$.

4.34 [PPU_AI.H001] Unclear statement in the user manual with respect to vector memory ECC errors

Description

In the user manual, the statement regarding the exceptions in the "Vector memory" section of the PPU chapter is improved. It becomes clearer based on which conditions "Machine check - uncorrectable ECC or parity Error in Vector Memory" exception is triggered.

Recommendation

The following sentence is added in the "Vector memory" section of the PPU chapter:

Exceptions

In case an uncorrectable ECC error is detected during a load/store operation from/to vector memory, a "Machine check - uncorrectable ECC or parity Error in Vector Memory" exception is triggered. The error is reported in the HSSI_ECC_ERROR_VISION0 register and PPU_HSSI_ERR alarm is also triggered.

4.35 [PSI5S_TC.H002] Incomplete reset isolation

Description

Reset isolation is not correctly implemented for the suspend and disable acknowledge signals. As a result, reset domain crossing issues could be caused.

Recommendation

Software must ensure that the PSI5S module clock is disabled (CLC.DISR=1 and CLC.DISS=1) before issuing kernel reset or module group reset. Suspend or sleep must not be requested during the reset.

4.36 [PSI5_TC.H003] Incomplete reset isolation

Description

Reset isolation is not correctly implemented for the suspend and disable acknowledge signals. As a result, reset domain crossing issues could be caused.

Recommendation

Software must ensure that the PSI5 module clock is disabled (CLC.DISR=1 and CLC.DISS=1) before issuing module group reset. Suspend or sleep must not be requested during the reset.

4.37 [QSPI_TC.H010] Incomplete reset isolation

Description

Reset isolation is not correctly implemented for the suspend and disable acknowledge signals. As a result, reset domain crossing issues could be caused.

Recommendation

The following register bit-fields need to be set prior to reset:

- GLOBALCON.EN = 0
- GLOBALCON.CLKSEL = 0
- CLC.DISR = 0
- CLC.DISS = 0
- Suspend or sleep must not be requested during the reset

4.38 [QSPI_TC.H012] HS mode functionality is not supported for all baud-rate configurations

Description

High speed mode is not functional for all configurations.

Recommendation

High speed mode cannot be used to shift the sample point to the next clock cycle. Therefore, it is recommended not to use high speed mode during operation.

4.39 [SAFETY_TC.H021] Remove AA step designator from package specific data

Description

The AURIX™ TC4Dx Safety Manual V1.0 shows the AA-step designator at the beginning of chapter 5.3.1, in the title of table 94 and in the title of table 95. This information however is not only exclusive for the AA-step device but also valid for all AURIX™ TC4Dx device steps.

Recommendation

Ignore the AA-step designator in the locations mentioned above and consider the data is also valid for other AURIX™ TC4Dx steps.

4.40 [SCR_TC.H017] I2C clock synchronization in slave mode (clock stretching)

Description

When the I2C is in slave mode, it will hold the SCL line low until I2C_CNTR.IFLG has been cleared with exception of receiving a STOP condition.

The I2C_CNTR.IFLG is set under following conditions:

- After the address has been transferred (if acknowledged)
- After each byte has been transferred
- After a repeated START condition

Recommendation

Consider the description above for functionality scope.

4.41 [SCR_TC.H018] Do not use master mode stop bit (I2C_CNTR.STP=1) in slave mode

Description

While the SCR I2C is driving the SDA line to low and the I2C_CNTR.STP is set, the SCR I2C module returns to idle but keeps the SDA line low. This can occur, for example during the transmission of a data- or an acknowledge-bit.

Recommendation

Do not use the I2C_CNTR.STP bit while the module is configured as slave.

4.42 [SCR_TC.H021] I2C Clock stretching

Description

When the I2CSCR is in slave mode, the slave device normally will hold the SCL line low, until I2C_CNTR.IFLG has been cleared. The only exception where clock stretching will not be applied is after receiving a STOP condition.

Recommendation

Sending a STOP condition closely followed by START condition will result similar like a repeated start, and by this applying clock stretching.

4.43 [SCR_TC.H022] OCDS SCR reset during monitor mode

Description

Standby-Controller (SCR) can be individually reset via various means (Host, Debugger, SCR-internal event). For debugging, it provides a monitor-program based solution, where the monitor-program communicates with the external tool. In case during execution of the monitor-program (for example, when probing values via tool) the controller experiences a reset, a reconnect of tool and monitor-program is targeted. However, the latter is not correctly entered, and consequently the user-application gets started, breakpoints are ignored.

Recommendation

In order to restore the debug-state, the tool shall store Debug-System-Config (DBG_*), then send HRESET. After entering monitor program (halt-after-reset), respective configuration can be restored, halt released and debug-session can be continued.

4.44 [SENT_TC.H008] Incomplete reset isolation

Description

Reset isolation is not correctly implemented for the suspend and disable acknowledge signals. As a result, reset domain crossing issues could be caused.

Recommendation

Software must ensure that before issuing a kernel reset or a module group reset:

- SENT module is not in OCDS suspend mode or in sleep mode
- No suspend or sleep is being requested during the reset

4.45 [xSPI_TC.H001] XIP write is not supported

Description

Execute-In-Place (XIP) writes are not supported.

Recommendation

Use DMA transfers or non XIP writes via SRI slave.

4.46 [xSPI_TC.H002] 8bit and 16bit XIP read issue through un-cached memory

Description

It is expected that the XIP read using 8-bit and 16-bit pointer through un-cached memory address would provide proper data.

However, for XIP read of 8-bit and 16-bit through un-cached memory address with DFS = 8-bit and 16-bit respectively, does not read proper data. For XIP read of 32-bit, it is working as expected.

Recommendation

Perform XIP read of 8-bit and 16-bit through un-cached memory with DFS = 32-bit and XIP_DFS_HC needs to be enabled (fixed DFS feature). In addition, the data must be programmed with DFS = 32-bit.

5 Legacy issues carried over from previous product generations

5.1 Purpose of this chapter

The following topics have been identified and evaluated in the previous product generations.

Customers who are using previous AURIX™ generations will already be familiar with these issues and may re-use the problem assessment from previous AURIX™ generations.

Customers who are new to AURIX™ should evaluate the following topics thoroughly.

5.2 [CPU_TC.H024] Usage of atomic instructions SWAPMSK.W and LDMST to access registers with bit-fields that can also be updated by hardware (rwh)

Description

Atomic instructions like SWAPMSK.W and LDMST in the AURIX™ microcontroller provide atomicity and bit-wise operations to the targeted memory locations or peripheral registers. They are also referred to as Read-Modify-Write (RMW) instructions. The bit-manipulation functionality allows software to update individual bits in a register without affecting other, selecting the bits to be read and written through a mask in the instruction.

***Note**: Please refer to the TriCore™ Architecture Manual for further information about these instructions and their formats.*

***Note**: Additionally, please refer to the dedicated note in the User Manual indicating the usage of instructions SWAPMSK and LDMST for specific registers containing the "rwh" field, indicating that this field can be accessed and modified by software and hardware.*

As a general remark, consider that when implementing time-critical or safety-critical systems, operations involving registers that can be updated by both hardware and software, poses a risk of conflicts between hardware and software updates. In particular when using atomic instructions like SWAPMSK.W or LDMST to access registers with "*rwh*" fields. If the bit-field is accessed simultaneously by the hardware and the software, the hardware update events may be lost, and the bit-field updated by the hardware may be overwritten by the software write operation performed via the atomic instruction.

Recommendation

LDMST or SWAPMSK.W instructions should be used only with bit mask enabled for registers containing *rwh* bits.

It is essential to consider that an atomic instruction can take multiple cycles to complete, during which any hardware event can occur and be lost. For example, a SWAPMSK.W operation, used for reading (READ) a "*rwh*" flag from a register (for example, GTM IRQ_NOTIFY) and then for writing (WRITE) to clear flags of the register, takes around 6 clock cycles. It is possible that when the read operation is performed, no flag is set by hardware. However, within the 6-clock-cycle time interval, a hardware event can set a flag after the READ but before the WRITE of the atomic operation. At that point, the event flag was not captured by the READ operation but could be cleared by the WRITE operation. This is especially true if the bit-field is used or both status reporting and for clearing itself (for example, read value 1 indicates the hardware event was raised, while writing 1 to the same bit-field clears it). To generalize, if software and hardware update bit(s) in the same cycle, the software wins and the hardware event is lost.

5.3 [DMA_TC.071] Daisy Chain request is lost when repeat triggers too soon

Description

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

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

Scope

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

Effects

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

Workaround

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

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

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

5.4 [eGTM_AI.516] SPE-RTL: IRQ raised on disabled inputs

Description

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

Scope

SPE

Effects

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

Workaround

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

5.5 [eGTM_AI.517] (A)TOM: Missing edge on output signal (A)TOM_OUT

Description

If an (A)TOM channel is configured to be triggered by a previous channel by setting of (A)TOM[i]_CH[x]_CTRL.RST_CCU0=1 (SOMP mode in ATOM) and there is a pipeline/synchronization register within the trigger chain between the triggering channel and the triggered channel, the edge to the inverse SL at the output signal (A)TOM_OUT is not generated for (A)TOM[i]_CH[x]_CM1.CM1<2 and (A)TOM[i]_CH[x]_CM0.CM0>(A)TOM[i]_CH[x]_CM1.CM1. The problem only occurs if the selected clock resolution for the triggered channel has a divider factor of more than 1 related to the cluster clock CLS[i]_CLK.

Since GTM Generation 3, the problem does not occur if the pipeline/synchronization register is internally of (A)TOM module and the clock divider for the cluster clock CLS[i]_CLK is configured with a clock divider of 2 by setting of GTM_CLS_CLK_CFG.CLS[j]_CLK_DIV = 10_B.

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

Scope

ATOM, TOM

Effects

Missing edge on output signal (A)TOM_OUT.

Workaround 1

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

Workaround 2a

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

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

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

Workaround 2b

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

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

Workaround 3

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

5.6 [eGTM_AI.H803] SPEC-(A)TOM: Missing priority information for register update

Description

The following information is missing in the specification and has to be placed inside the TGC Sub-unit/AGC Sub-unit chapter:

- Inside ATOM chapter: The trigger condition has always priority over the bus write access to the ATOM[i]_AGC_OUTEN_STAT and ATOM[i]_AGC_ENDIS_STAT registers, even if ATOM[i]_AGC_OUTEN_CTRL.OUTEN_CTRL[k] / ATOM[i]_AGC_ENDIS_CTRL.ENDIS_CTRL[k] is set to 00_B. This means that the bus write access to ATOM[i]_AGC_OUTEN_STAT and ATOM[i]_AGC_ENDIS_STAT register is ignored in the clock cycle when the trigger condition is active
- Inside TOM chapter: The trigger condition has always priority over the bus write access to the TOM[i]_TGC[g]_OUTEN_STAT and TOM[i]_TGC[g]_ENDIS_STAT registers, even if TOM[i]_TGC[g]_OUTEN_CTRL.OUTEN_CTRL[k] / TOM[i]_TGC[g]_ENDIS_CTRL.ENDIS_CTRL[k] is set to 00_B. This means that the bus write access to TOM[i]_TGC[g]_OUTEN_STAT and TOM[i]_TGC[g]_ENDIS_STAT register is ignored in the clock cycle when the trigger condition is active

***Note**: The trigger override does not happen if the trigger is a HOST_TRIG, as this is initiated by a bus write itself and cannot happen at the same time as another bus write to the register.*

***Note**: This AppHint is published as GTM-IP-523 by Bosch.*

Scope

TOM, ATOM

Effects

In (A)TOM the bus write access to the "OUTEN_STAT" and "ENDIS_STAT" registers is overridden by a trigger update and the desired values are not written into the register.

Recommendation

To nevertheless ensure that the desired value is actually stored in the target register, consider one of the following hints:

1. Write first the channel k within "ENDIS_CTRL" ("OUTEN_CTRL") and write the desired channel k in "ENDIS_STAT" ("OUTEN_STAT") afterward. Additionally, set all unused channels within "ENDIS_CTRL" ("OUTEN_CTRL") to 11_B. This way, either the asynchronous write or the synchronous write becomes effective
2. If recommendation 1 is not the case, then read back the value of the "OUTEN_STAT" and "ENDIS_STAT" registers to ensure the written value is actually present in the register

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

Description

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

Scope

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

Effects

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

Workaround

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

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

Description

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

Scope

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

Effects

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

Workaround

Ignore redundant SIR.WUPA/B events.

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

Description

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

Scope

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

Effects

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

Workaround

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

5.10 [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.

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

Description

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

Scope

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

Effects

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

Workaround

Configure action point offset smaller than static frame length.

5.12 [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 start-up of the node by only one micro tick.

5.13 [FlexRay_AI.093] Acceptance of start-up frames received after reception of more than gSyncNodeMax sync frames

Description

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

Scope

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

Effects

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

Workaround

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

5.14 [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.

5.15 [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.

5.16 [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

5.17 [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).

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

Description

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

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

If the node is able to correct the offset of one macro tick (pOffsetCorrectionOut >> gdMacrotick), the node enters NORMAL_ACTIVE with the first start-up attempt.

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

Scope

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

Effects

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

Workaround

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

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

Description

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

Scope

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

Effects

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

Workaround

None

5.20 [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

5.21 [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

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

Description

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

Scope

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

Effects

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

Workaround

Leave HALT state by hardware reset.

5.23 [FlexRay_AI.104] Missing start-up frame in cycle 0 at coldstart after FREEZE or READY command

Description

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

Scope

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

Effects

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

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

Workaround

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

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

Description

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

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

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

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

Scope

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

Effects

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

Workaround

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

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

Description

The problem occurs under the following conditions:

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

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

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

a) Message buffer transfer from Message RAM to OBF

5 Legacy issues carried over from previous product generations

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.

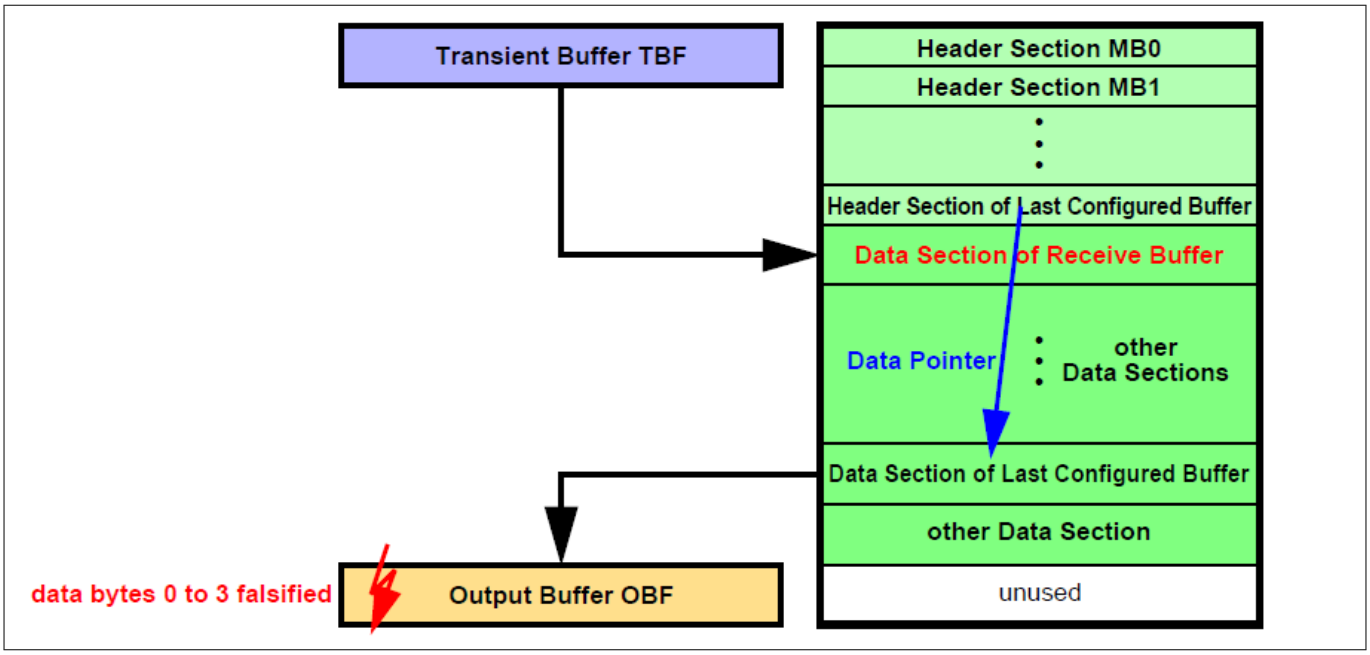


Figure 4 Message buffer transfer from Message RAM to OBF

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.

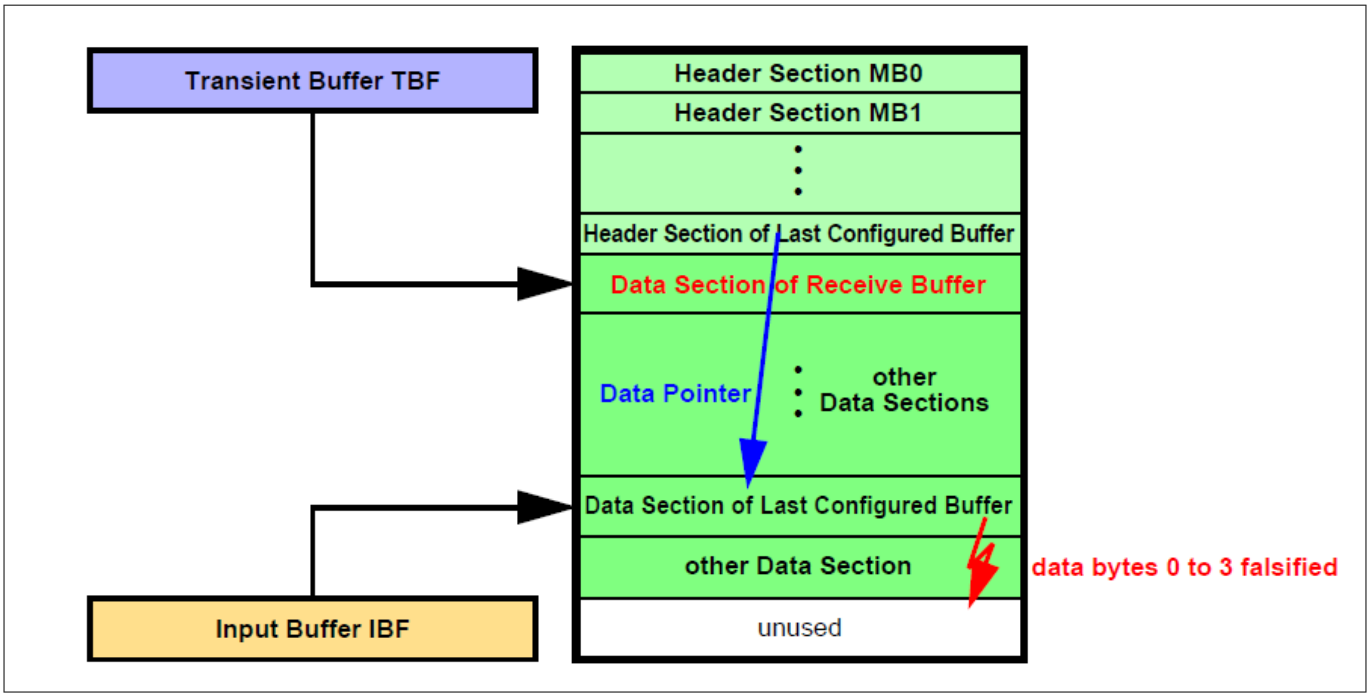


Figure 5 Message buffer transfer from IBF to Message RAM

Scope

The erratum is limited to the case when (see Figure 6 “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 4)

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 5)

Workaround 1

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

Workaround 2

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

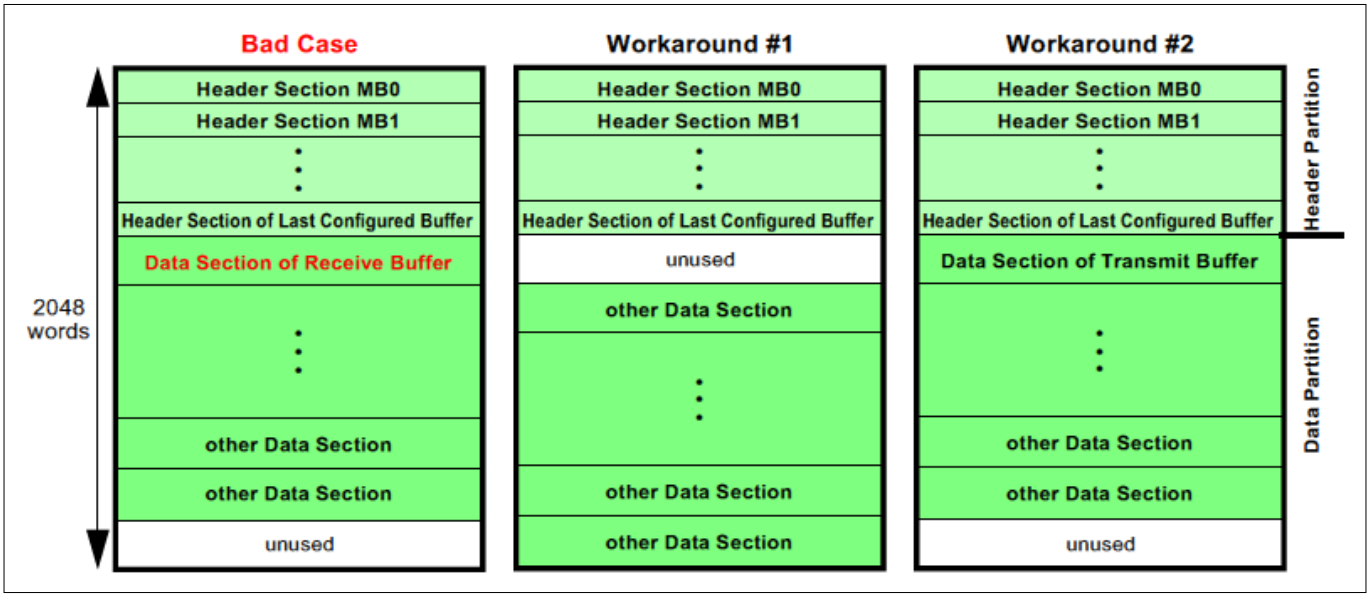


Figure 6 Message RAM configurations

5.26 [FlexRay_AI.H004] Only the first message can be received in External Loop Back mode

Description

If the loop back (TXD to RXD) will be performed via external physical transceiver, there will be a large delay between TXD and RXD.

A delay of two sample clock periods can be tolerated from TXD to RXD due to a majority voting filter operation on the sampled RXD.

Only the first message can be received, due to this delay.

To avoid that only the first message can be received, a start condition of another message (idle and sampling '0' -> low pulse) must be performed.

The following procedure can be applied at one or both channels:

- wait for no activity (TEST1.AOx = 0 -> bus idle)

- set Test Multiplexer Control to I/O Test Mode (TEST1.TMC = 2), simultaneously TXDx = TXENx = 0
- wait for activity (TEST1.AOx = 1 -> bus not idle)
- set Test Multiplexer Control back to Normal signal path (TEST1.TMC = 0)
- wait for no activity (TEST1.AOx = 0 -> bus idle)

Now the next transmission can be requested.

5.27 [FlexRay_AI.H005] Initialization of internal RAMs requires one eray_bclk cycle more

Description

The initialization of the E-Ray internal RAMs as started after hardware reset or by CHI command CLEAR_RAMS (SUCC1.CMD[3:0] = 1100_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.

5.28 [FlexRay_AI.H006] Transmission in ATM/Loopback mode

Description

When operating the E-Ray in ATM/Loopback mode there should be only one transmission active at the same time. Requesting two or more transmissions in parallel is not allowed.

To avoid problems, a new transmission request should only be issued when the previously requested transmission has finished. This can be done by checking registers TXRQ1/2/3/4 for pending transmission requests.

5.29 [FlexRay_AI.H007] Reporting of coding errors via TEST1.CERA/B

Description

When the protocol engine receives a frame that contains a frame CRC error as well as an FES decoding error, it will report the FES decoding error instead of the CRC error, which should have precedence according to the non-clocked SDL description.

This behavior does not violate the FlexRay protocol conformance. It has to be considered only when TEST1.CERA/B is evaluated by a bus analysis tool.

5.30 [FlexRay_AI.H009] Return from test mode operation

Description

The E-Ray FlexRay IP-module offers several test mode options

- Asynchronous Transmit Mode
- Loop Back Mode
- RAM Test Mode
- I/O Test Mode

To return from test mode operation to regular FlexRay operation we strongly recommend to apply a hardware reset via input eray_reset to reset all E-Ray internal state machines to their initial state.

***Note**: The E-Ray test modes are mainly intended to support device testing or FlexRay bus analyzing. Switching between test modes and regular operation is not recommended.*

5.31 [FlexRay_AI.H010] Driver software must launch CLEAR_RAMS command before reading from E-Ray RAMs

Description

After a Power-on-Reset, the RAMs used by the E-Ray module must be written once. Reading from RAM locations before at least writing once to them may cause a Parity Error Trap (AUDO* microcontroller family) or an ECC Error Trap.

Recommendation

The recommended solution is to trigger a CLEAR_RAMS command (via register SUCC1). CLEAR_RAMS fills a defined value into all memory locations. A safe initialization sequence of the E-Ray RAM blocks using the CLEAR_RAMS command is described in section “CLEAR_RAMS Command” of the E-Ray chapter in the corresponding AURIX™ user manual.

An alternate solution is to write explicitly by software to all RAM locations, which are intended to be read later, for example by writing the complete configuration and writing into all allocated message buffers, including receive buffers. The latter activity may be required if buffers are configured to store frames sent in the dynamic segment. The sent frames may be smaller than the configured buffer size. If the software reads the amount of configured data (not the amount of received data), it may read from non-activated RAM locations.

For AURIX™ devices, as a further option, the MBIST auto-initialization algorithm may be used. See section “Filling a Memory with Defined Contents” in the corresponding AURIX™ user manual.

5.32 [FlexRay_AI.H011] Behavior of interrupt flags in FlexRay™ Protocol Controller (E-Ray)

Description

In the corner case described below, the actual behavior of the interrupt flags of the FlexRay™ Protocol Controller (E-Ray) differs from the expected behavior.

***Note**: This behaviour only applies to E-Ray interrupts INT0 and INT1. All other E-Ray interrupts are not affected.*

Expected behavior

When clearing an interrupt flag by software, the resulting value of the flag is expected to be zero.

A hardware event that occurs afterwards then leads to a zero to one transition of the flag, which in turn leads to an interrupt service request.

Actual behavior in corner case

When the interrupt flag is being cleared by software in the same clock cycle as a new hardware event sets the flag again, then the hardware event wins and the flag remains set without being cleared.

As interrupt requests are generated only upon zero to one transitions of the flag, no interrupt request will be generated for this flag until the flag is successfully cleared by software later on.

Workaround

After clearing the flag, the software shall read the flag and repeat clearing until the flag reads zero.

5.33 [HSCT_TC.H010] Interface control command timing on the LVDS ports

Description

As described in section “Interface Control” of the HSCT chapter in the user manual, a HSCT master device is sending interface control commands to a slave device by setting the command in register IFCTRL.IFCVS and triggering IFCTRL.SIFCV.

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

As flag IRQ.IFCFS only indicates

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

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

Instead the timing shown in the following figure applies.

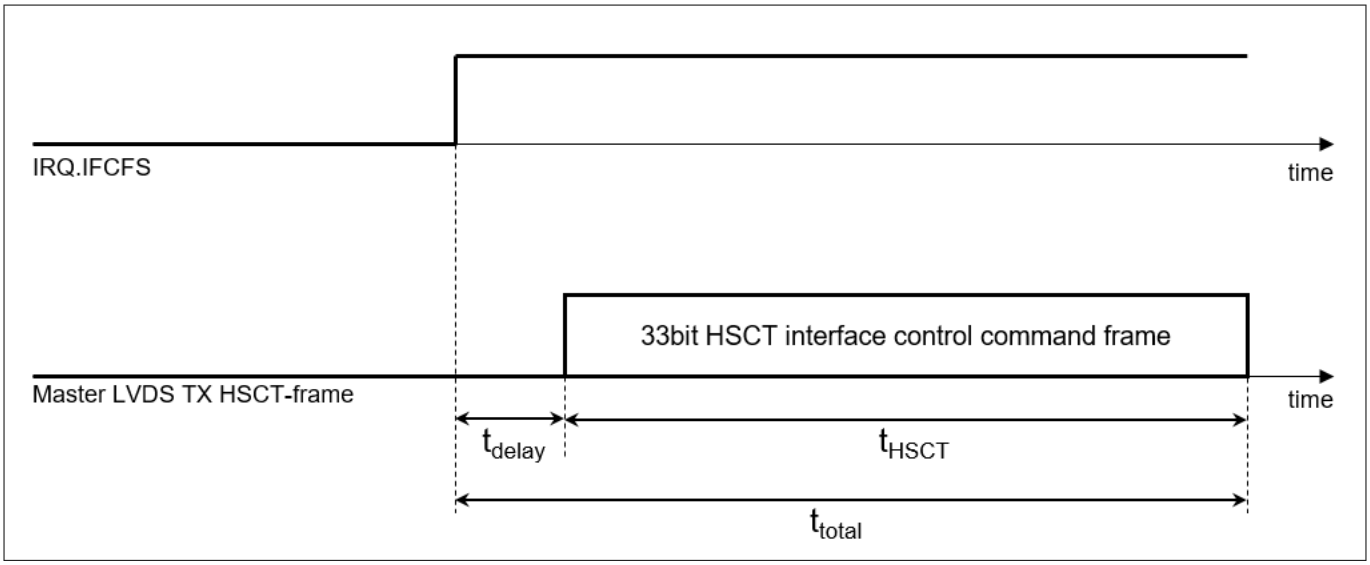


Figure 7 Timing of LVDS TX interface command sending in relation to IRQ.IFCFS

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, the additional time t_{delay} has to be taken into account with:

- $t_{delay} \geq 10\ \mu s$

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

5.34 [MCMCAN_AI.025] Sporadic data corruption (payload) in case acceptance filtering is not finished before reception of data R3 (DB7..DB4) is completed

Description

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

- ***TC3xx**: NDATni, IRi, RXFnSi*
- ***TC4xx**: Ni_NDATn, Ni_IR, Ni_RXFnSi*

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

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

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

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

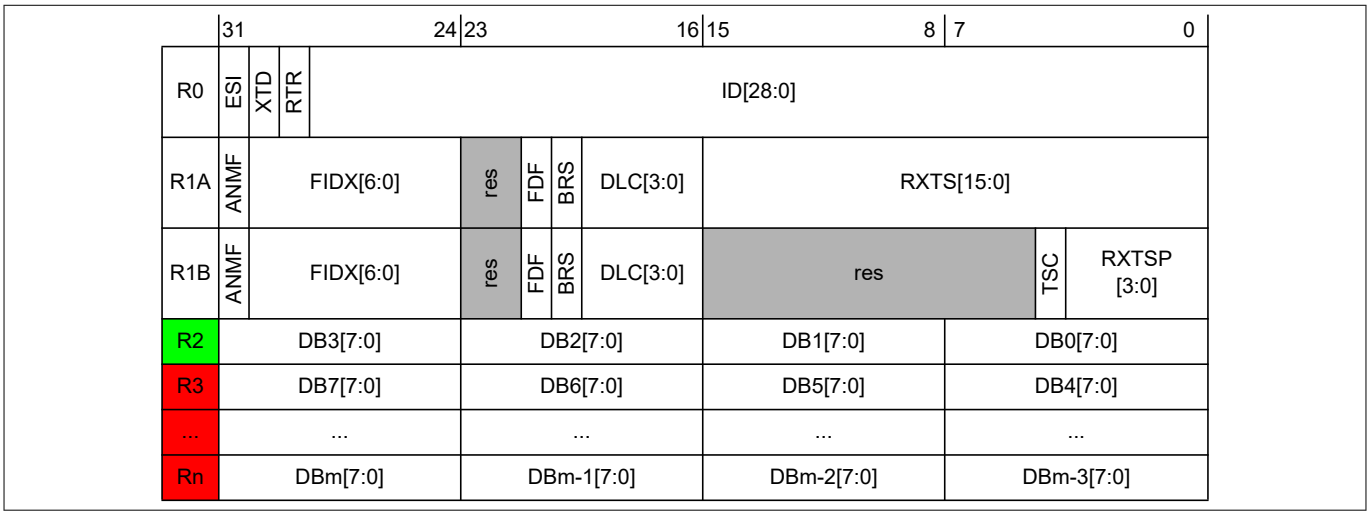


Figure 8 TC4xx - Rx Buffer and FIFO Element (R_i holds Data Byte DB(x+3)..DBx (with x=4*(i-2)))

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

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

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

Despite the corrupted data, the M_CAN signals the storage of a valid frame in the Message RAM:

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

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

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

5 Legacy issues carried over from previous product generations

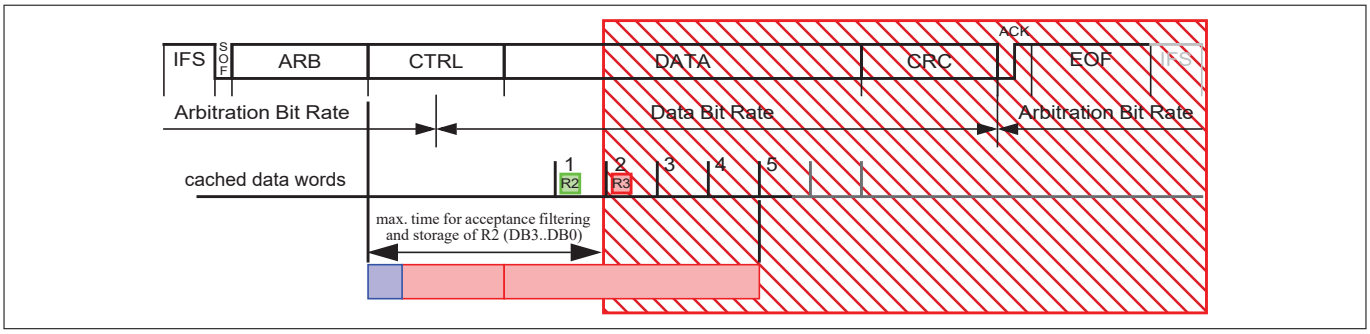


Figure 9 CAN Frame with DLC > 4

TC4xx: Minimum host clock frequency for CAN FD when DLC = 5

Table 23 TC4xx: Minimum host clock frequency for CAN FD when DLC = 5

***Note**: All calculated values for TC4xx involve an enabled CAN Routing Engine (CRE).*

		Arbitration bit rate = 0.5 Mbit/s			Arbitration bit rate = 1 Mbit/s		
No. of configured Active FE[1) 2)] 11-bit IDs/ 29-bit IDs	**Number of Active CAN nodes**	**Data bit rate = 1 Mbit/s**	**Data bit rate = 2 Mbit/s**	**Data bit rate = 4 Mbit/s**	**Data bit rate = 2 Mbit/s**	**Data bit rate = 4 Mbit/s**	**Data bit rate = 5 Mbit/s**
32/16	2	10	19	32	20	37	44
	3	13	24	41	26	47	57
	4	16	29	45	31	58	69
64/32	2	19	35	61	38	70	84
	3	25	45	78	49	90	108
	4	30	55	96	60	110	132
96/48	2	28	52	90	56	103	124
	3	36	67	116	72	133	160
	4	44	81	141	88	162	195
128/64	2	37	68	119	74	136	164
	3	48	88	153	95	175	211[3)]
	4	58	107	187	116	214[3)]	-[4)]

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

Scope

The erratum is limited to the case when the host clock frequency used in the actual device is below the limit shown in section "TC4xx: Minimum host clock frequency for CAN FD when DLC = 5".

Effects

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

Workaround

Check whether the minimum host clock frequency, that is shown in section "TC4xx: Minimum host clock frequency for CAN FD when DLC = 5", is below the host clock frequency used in the actual device.

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

If no, use one of the following two workarounds.

Workaround 1

Try different configurations by changing the following parameters until ensuring that the actual synchronous clock f_{MCANH} frequency is above the minimum host clock frequency shown in section "TC4xx: Minimum host clock frequency for CAN FD when DLC = 5".

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

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

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

Workaround 2

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

5.35 [MSC_TC.027] De-feature of ABRA for MSC

Description

Various issues have been identified when using the ABRA block for MSC.

Scope

When enabling ABRA block.

Effects

Failing ABRA test cases.

Workaround

To mitigate these issues, it is recommended that users do not use the ABRA block within the MSC module.

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

Description

According to the specification, a baud rate error is detected if the incoming shift clock supplied by the master has less than half or more than double the expected baud rate (determined by bit-field GLOBALCON.TQ).

However, in this design step, a baud rate error is detected not only if the incoming shift clock has less than half the expected baud rate (as specified), but also already when the incoming shift clock is somewhat (i.e. less than double) higher than the expected baud rate.

In this case, the baud rate error is indicated in the current frame.

Workaround

It is recommended not to rely on the baud rate error detection feature, and not to use the corresponding automatic reset enable feature (i.e. keep GLOBALCON.AREN = 0_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 for example by parity, CRC, etc., while clocking problems of an AURIX™ master are detected by mechanisms implemented in the master.

Protection against the effects of high frequency glitches is provided by the spike detection feature in slave mode.

5.37 [SDMMC_AI.002] Transfer complete interrupt not generated post ADMA3 transfer

Description

Transfer complete interrupt is not generated after completion of an ADMA3 transfer.

Scope

When using ADMA3 transfers.

Effects

System deadlock requiring reset.

Workaround

This issue can be avoided by not using ADMA3 data transfers and reverting to SDMA or ADMA2 data transfers.

5.38 [SDMMC_AI.H001] Packet buffer is not fully utilized for the write traffic

Description

For write transfers, if the (Block size * Block count) exceeds the packet buffer size (1024 bytes), the SDMMC DMA engine will not use the last block of the packet buffer. The engine pauses traffic at the N-1 block and waits for a transfer to complete before resuming, resulting in unused space in the packet buffer. In write transfers involving two blocks of 512 bytes each, the host will require an additional approximately 128 clock cycles to fill the packet buffer before the card transfer can begin. As a result, the transfer of the second 512-byte block may take longer because it cannot be filled in parallel with the first block transfer, leading to inefficiencies in the write process.

Recommendation

Please refer to the description above for the functionality scope.

5.39 [SENT_TC.H009] Unexpected NNI error behavior

Description

The NNI interrupt is triggered when the actual number of transmitted nibbles exceeds the expected count predefined in RCRx.FRL. Specifically, when IEP = 0 and no pause pulse is used, NNI interrupt performs as expected. However, when IEP = 1 and a pause pulse is used, the interrupt is not triggered if the number of transmitted nibbles surpasses the expected value by one nibble. In this case, the NNI interrupt is only triggered when the number of nibbles transmitted surpasses the expected value by two or more nibbles.

Recommendation

Due to this issue, SENT messages could be missed. This can be detected by implementing timeout or message rate checking mechanisms.

5.40 [SMU_TC.015] SMU alarm emulation might trigger unwanted active alarm reaction

Description

While the SMU is in START state, a level alarm is raised by the hardware and stays active. Since the SMU is in START state, the corresponding configured reaction, for example RESET_REQ, is not triggered. In this context, another alarm for which there is no SMU reaction configured, is triggered by the software using the alarm emulation function of SMU. The expected behavior is that SMU does not react to the software triggered alarms, however, the alarm reaction of previously set alarm, for example RESET_REQ, would unexpectedly be triggered.

Scope

Alarm emulation

Effects

Unintended alarm reaction. The actual reaction will depend on the configuration.

Workaround

While the SMU is in START state, all alarms must be cleared in SMU and at the source safety mechanism before using the alarm emulation function.

Revision history

Document version	Date of release	Description of changes
		History redacted. First public release

Revision history

Document version	Date of release	Description of changes

Table 24 Errata fixed in this step (Reference = TC4Dx step AA)

Errata	Short description	Change
CCU_TC.014	XTAL3 RTC duty cycle violates lower limit in Victim/Aggressor measurement	Fixed
DRE_TC.007	LETH reads TX descriptor while DRE updates the OWN bit	Fixed
FLASH_TC.059	Remove limitation on BMHDs and dual UCBs in MDS	Fixed
FLASH_TC.H025	BMHDs must be symmetric wrt to PWD	Fixed
GETH_AI.028	FCS not recalculated when Layer 2 header of the forward traffi via GETH bridge is modified at Egress port	Fixed
GETH_AI.030	Head-Of-Line (HOL) blocking in GETH bridge receive MAC interface	Fixed
GETH_TC.007	Incorrect output of PPS pulse on PORTS	Fixed
LETH_AI.002	10BASE-T1S - RX frames - stall	Fixed
LETH_AI.004	FCS not recalculated when Layer 2 header of the Forward traffi via LETH bridge is modified at Egress port	Fixed
LETH_AI.006	10BaseT1S ED pulse decode only >30 ns by EQOS	Fixed
LETH_AI.007	LETH bridge hang due to ATI abort in forwarding path	Fixed
LETH_AI.009	10BASE-T1S Transmission aborted with error excessive collisions	Fixed
LETH_TC.001	Some pins with MDC signals in MII/RMII mode not functional	Fixed
LETH_TC.003	Incorrect output of PPS pulse on PORTS	Fixed
PCIE_TC.007	Module group reset issue when AXI WR	Fixed
PMS_TC.031	Clock XTAL stops during STANDBY1_70k	Fixed
PMS_TC.P001	STANDBY1 - i_EVRSB baseline current higher than observed in simulation	Fixed
PMS_TC.P002	STANDBY0 - i_EVRSB baseline current higher than observed in simulation	Fixed
SAFETY_TC.030	Usage of ESM softare broken wire detection in FMEDA	Fixed

(table continues...)

Table 24 (continued) Errata fixed in this step (Reference = TC4Dx step AA)

Errata	Short description	Change
SMM_TC.004	First pulse on the ESR0 pin is ignored	Fixed
SMM_TC.H003	ESR0 released immediately before 300usec when PORST released	Fixed

Trademarks

All referenced product or service names and trademarks are the property of their respective owners.

Edition 2025-06-18

Published by

Infineon Technologies AG

81726 Munich, Germany

Do you have a question about any aspect of this document?

Email: erratum@infineon.com

Document reference
IFX-nto1723186984314

Important notice

The information given in this document shall in no event be regarded as a guarantee of conditions or characteristics ("Beschaffenheitsgarantie").

With respect to any examples, hints or any typical values stated herein and/or any information regarding the application of the product, Infineon Technologies hereby disclaims any and all warranties and liabilities of any kind, including without limitation warranties of non-infringement of intellectual property rights of any third party.

In addition, any information given in this document is subject to customer's compliance with its obligations stated in this document and any applicable legal requirements, norms and standards concerning customer's products and any use of the product of Infineon Technologies in customer's applications.

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

Warnings

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

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