Dear Customer,

With this INFINEON Technologies Information Note we would like to inform you about the following

**Errata sheet update affecting products TC29x_BB**
Errata sheet update affecting products TC29x_BB

Products affected:

<table>
<thead>
<tr>
<th>Sales Name</th>
<th>SP N°</th>
<th>OPN</th>
<th>Package</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Please refer to 1_cip16217 attached

Detailed Change Information:

Subject: TC29x_BB_Errata_Sheet_v1_6

Reason: Update of Errata Sheet from version V1.5 to V1.6

<table>
<thead>
<tr>
<th>Description:</th>
<th>Old</th>
<th>New</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- Errata Sheet V1.5
- Errata Sheet V1.6

Product Identification: Not applicable (no change of product)

Impact of Change: Not applicable (no change of product)

Attachments:

- Affected product list: 1_cip16217
- Errata Sheet V1.6: 4_cip16217

Intended start of delivery: Not applicable

If you have any questions, please do not hesitate to contact your local Sales office.
This Errata Sheet describes the deviations from the current user documentation.

<table>
<thead>
<tr>
<th>Table 1</th>
<th>Current Documentation¹)</th>
</tr>
</thead>
<tbody>
<tr>
<td>TC29x B-Step User’s Manual</td>
<td>V1.3</td>
</tr>
<tr>
<td>TC290/TC297/TC298/TC299 BB-Step Data Sheet</td>
<td>V1.1</td>
</tr>
<tr>
<td>TriCore TC1.6P &amp; TC1.6E Core Architecture, Instruction Set</td>
<td>V1.0D10, V1.0D15</td>
</tr>
<tr>
<td>OCDS User’s Manual²)</td>
<td>V2.9.1</td>
</tr>
</tbody>
</table>

¹) Newer versions replace older versions, unless specifically noted otherwise.
²) Distribution under NDA, only relevant for tool development not for application development.


Conventions used in this document

Each erratum identifier follows the pattern Module_Arch.TypeNumber:
- **Module**: subsystem, peripheral, or function affected by the erratum
- **Arch**: microcontroller architecture where the erratum was initially detected
  - **AI**: Architecture Independent
  - **TC**: TriCore
- **Type**: category of deviation
– [none]: Functional Deviation
– P: Parametric Deviation
– H: Application Hint
– D: Documentation Update

• **Number**: ascending sequential number within the three previous fields. As this sequence is used over several derivatives, including already solved deviations, gaps inside this enumeration can occur.

**Notes**

1. This Errata Sheet applies to all temperature and frequency versions and to all memory size variants, unless explicitly noted otherwise. For a derivative synopsis, see the latest Data Sheet/User’s Manual. This Errata Sheet covers several device versions. If an issue is related to a particular module, and this module is not specified for a specific device version, this issue does not apply to this device version. E.g. issues with identifier “ETH” do not apply to devices where no Ethernet MAC is specified, and issues with identifier “CIF” only apply to ADAS devices.

2. Devices marked with EES or ES are engineering samples which may not be completely tested in all functional and electrical characteristics, therefore they should be used for evaluation only. The specific test conditions for EES and ES are documented in a separate Status Sheet.

3. This device is equipped with TriCore “TC1.6P” cores. Some of the errata have workarounds which are possibly supported by the tool vendors. Some corresponding compiler switches need possibly to be set. Please see the respective documentation of your compiler. For effects of issues related to the on-chip debug system, see also the documentation of the debug tool vendor.
1 History List / Change Summary

Table 2 History List

<table>
<thead>
<tr>
<th>Version</th>
<th>Date</th>
<th>Remark</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0</td>
<td>2014-09-30</td>
<td>This device step supports Workaround 4 for problem ADC_TC.068 (Effect of VAGND Cross Coupling on Conversion Result)</td>
</tr>
<tr>
<td>1.1</td>
<td>2015-04-10</td>
<td>Updates see columns “Change” in errata sheet V1.1. Removed:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• DMA_TC.023 (Conditional Linked List or Pattern Detection: No Timestamp Support) - documented in sub-chapters “Conditional Linked List” and “Pattern Detection” of the User’s Manual</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• DSADC_TC.007 (Connection of External Modulator not supported) - connections updated (no longer described) in User’s Manual</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• GTM_AI.H001 (TOM/ATOM: PWM generation with register CM0≥1) - documented in GTM sub-chapters “TOM Channel (TOM_CH[x])” and “ATOM Signal Output Mode PWM (SOMP)” of the User’s Manual</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• MTU_TC.H001 (Program RAM access not allowed during PTAG testing) - documented in CPU sub-chapter “MBIST usage recommendations” of the User’s Manual</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• QSPI_TC.H003 (Effects of GLOBALCON. RESETS during transmission) - description of bit field RESETS updated in User’s Manual</td>
</tr>
</tbody>
</table>
## History List / Change Summary

### Table 2: History List

<table>
<thead>
<tr>
<th>Version</th>
<th>Date</th>
<th>Remark</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.2</td>
<td>2015-05-29</td>
<td>Updates see columns “Change” in errata sheet V1.2. Removed:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• PORTS_TC.H007 (Pad Configuration Selection for SMU FSP (Error Pin)) - covered by SMU_TC.H001 (Write all bit fields of SMU_PCTL with one write access)</td>
</tr>
<tr>
<td>1.3</td>
<td>2016-02-04</td>
<td>• New/updated text modules see columns “Change” in errata sheet V1.3. Removed:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• ASCLIN_TC.H002 (Auto Baud Rate Operation) - description integrated in User's Manual, section “Auto Baud Rate Detection”</td>
</tr>
<tr>
<td>1.4</td>
<td>2016-07-29</td>
<td>• New/updated text modules see columns “Change” in Table 4..6 of errata sheet V1.4. Removed:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• EBU_TC.P001 (EBU Performance Limitation) - see EBU_TC.P002 (EBU Timing Deviations) for this device step.</td>
</tr>
<tr>
<td>1.5</td>
<td>2017-03-31</td>
<td>• New/updated text modules see columns “Change” in tables 4..6 of errata sheet V1.5. Removed reference to “GTM-IP Gen1 IFX Errata Sheet” in Table 1 - all GTM errata relevant for this design step are considered in this TC29x errata sheet.</td>
</tr>
<tr>
<td>1.6</td>
<td>2018-02-23</td>
<td>• Update: new/updated text modules see columns “Change” in tables 4..6.</td>
</tr>
</tbody>
</table>
## Table 3  Errata fixed in this step

<table>
<thead>
<tr>
<th>Errata</th>
<th>Short Description</th>
<th>Change</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADC_AI.008</td>
<td>Wait-for-Read condition for register GLOBRES not detected in continuous auto-scan sequence</td>
<td>Fixed</td>
</tr>
<tr>
<td>ADC_AI.012</td>
<td>Losing Data in Sequences of Synchronous Conversions with both Standard and Alternate Reference</td>
<td>Fixed</td>
</tr>
<tr>
<td>ADC_AI.014</td>
<td>Wrong Result of Conversion in Cancel-Inject-Repeat Mode</td>
<td>Fixed</td>
</tr>
<tr>
<td>ADC_AI.H005</td>
<td>Handling the Alternate Reference</td>
<td>Fixed</td>
</tr>
<tr>
<td>ADC_AI.H006</td>
<td>Ratio of Module Clock to Converter Clock</td>
<td>Fixed</td>
</tr>
<tr>
<td>ADC_AI.H007</td>
<td>Ratio of Sample Time $t_S$ to SHS Clock $f_{SH}$</td>
<td>Fixed</td>
</tr>
<tr>
<td>ADC_TC.062</td>
<td>VADC Calibration</td>
<td>Fixed</td>
</tr>
<tr>
<td>ADC_TC.065</td>
<td>Accuracy of conversion affected by previous conversion (“Memory Effect”)</td>
<td>Fixed</td>
</tr>
<tr>
<td>ADC_TC.066</td>
<td>Offset error affected by idle time between conversions (“Idle Time Effect”)</td>
<td>Fixed</td>
</tr>
<tr>
<td>ADC_TC.067</td>
<td>Connection of Background Source to GTM Signals</td>
<td>Fixed</td>
</tr>
<tr>
<td>ADC_TC.069</td>
<td>Using the System PLL as Clock Source for the Converter Clock $f_{ADC}$</td>
<td>Fixed</td>
</tr>
<tr>
<td>ADC_TC.H010</td>
<td>Disabling Post-Calibration for upper Groups</td>
<td>Fixed</td>
</tr>
<tr>
<td>ADC_TC.H011</td>
<td>Bit DCMSB in register GLOBCFG</td>
<td>Fixed$^1$</td>
</tr>
<tr>
<td>CCU_TC.001</td>
<td>CPU Clock Frequency Generation</td>
<td>Fixed$^2$</td>
</tr>
<tr>
<td>CPU_TC.124</td>
<td>Missing ECC bits in Program Cache Tag Comparison</td>
<td>Fixed</td>
</tr>
<tr>
<td>DSADC_TC.008</td>
<td>Input Signals with Small Amplitude affected by “Dead Zone”</td>
<td>Fixed</td>
</tr>
<tr>
<td>DSADC_TC.010</td>
<td>Effect of Modulator Frequency Ratio on DC Input Signals (Idle Tones)</td>
<td>Fixed</td>
</tr>
<tr>
<td>ETH_TC.P001</td>
<td>RMII: setup time for ETHTXEN, ETHTXD[1:0]</td>
<td>Fixed</td>
</tr>
<tr>
<td>Errata</td>
<td>Short Description</td>
<td>Change</td>
</tr>
<tr>
<td>----------------</td>
<td>----------------------------------------------------------------------------------</td>
<td>------------</td>
</tr>
<tr>
<td>FFT_TC.H001</td>
<td>FFT registers only support 32-bit accesses</td>
<td>Fixed³)</td>
</tr>
<tr>
<td>GTM_AL.132</td>
<td>GTM_TOP level: AEI write to BRIDGE_MODE register can result in blocking of AEI configuration interface</td>
<td>Fixed</td>
</tr>
<tr>
<td>GTM_AL.133</td>
<td>DPLL: RAM 1 b+c initialization outside used address range</td>
<td>Fixed</td>
</tr>
<tr>
<td>GTM_AL.135</td>
<td>TIM: Bad ACB word in the case of timeout detection</td>
<td>Fixed</td>
</tr>
<tr>
<td>GTM_AL.136</td>
<td>(A)TOM: wrong output signal in case of centre aligned PWM with CM0=0 and CM0=1 on triggered channel</td>
<td>Fixed</td>
</tr>
<tr>
<td>GTM_AL.137</td>
<td>AEI related MCS Disabling</td>
<td>Fixed</td>
</tr>
<tr>
<td>GTM_AL.138</td>
<td>DPLL: Wrong PSTC/PSSC value after initialisation and restart</td>
<td>Fixed</td>
</tr>
<tr>
<td>GTM_AL.177</td>
<td>DPLL: setting of lock1= '1' status flag delayed by one event if profile has no gaps (syn_nt/ns = 0)</td>
<td>Fixed</td>
</tr>
<tr>
<td>MTU_TC.007</td>
<td>Error Overflow Indication ECCD.EOV</td>
<td>Fixed</td>
</tr>
<tr>
<td>MTU_TC.008</td>
<td>Reading register MSTATUS while a test is running</td>
<td>Fixed</td>
</tr>
<tr>
<td>MTU_TC.009</td>
<td>MBIST operation for CPU2 PSPR</td>
<td>Fixed</td>
</tr>
<tr>
<td>MTU_TC.010</td>
<td>Write Mask and Right/Left Half Select Tests not functional</td>
<td>Fixed</td>
</tr>
<tr>
<td>MTU_TC.H002</td>
<td>Writing to register MCONTROL</td>
<td>Fixed</td>
</tr>
<tr>
<td>OCDS_TC.H009</td>
<td>Power Supply Friendly Debug Monitor (PSFDM) will be Improved in Future Design Steps</td>
<td>Fixed⁴)</td>
</tr>
<tr>
<td>PMC_TC.001</td>
<td>VEXT secondary monitoring when using separate standby supply pin VEVRSB (Problem only for devices in BGA package)</td>
<td>Fixed</td>
</tr>
</tbody>
</table>

Table 3  Errata fixed in this step
Table 3  Errata fixed in this step

<table>
<thead>
<tr>
<th>Errata</th>
<th>Short Description</th>
<th>Change</th>
</tr>
</thead>
<tbody>
<tr>
<td>QSPI_TC.002</td>
<td>Missing Interrupts for Start of Frame Event when using External Slave Select Expansion</td>
<td>Fixed</td>
</tr>
<tr>
<td>QSPI_TC.003</td>
<td>QSPI Clock Configuration Restrictions</td>
<td>Fixed</td>
</tr>
<tr>
<td>QSPI_TC.004</td>
<td>User Interrupt USR not generated for BUSY phase after transition from PAUSE to RUN mode</td>
<td>Fixed</td>
</tr>
<tr>
<td>QSPI_TC.005</td>
<td>Baud rate error detection in slave mode</td>
<td>Fixed</td>
</tr>
<tr>
<td>RESET_TC.P002</td>
<td>Application Reset Boot Time</td>
<td>Fixed$^5$</td>
</tr>
<tr>
<td>RESET_TC.006</td>
<td>Mode selection from configuration pins not properly done if HWCFG[6]=0</td>
<td>Fixed</td>
</tr>
<tr>
<td>SCU_TC.028</td>
<td>GPIOs and ESR0/1 during LBIST execution not stable</td>
<td>Fixed$^6$</td>
</tr>
<tr>
<td>SCU_TC.H011</td>
<td>LBIST - Recommended Settings</td>
<td>Fixed</td>
</tr>
<tr>
<td>SMU_TC.003</td>
<td>Spikes may occur on FSP Pin SMUFSP</td>
<td>Fixed</td>
</tr>
<tr>
<td>SMU_TC.004</td>
<td>Effect of Kernel Reset on Pins selected for Emergency Stop</td>
<td>Fixed</td>
</tr>
</tbody>
</table>

1) Documented in TC29x B-step User’s Manual ≥ V1.2. DCMSB = 1 may be used with workaround 4 in ADC_TC.068
2) Formula corrected in TC29x B-step User’s Manual ≥ V1.2
3) Documented in TC29x B-step User’s Manual ≥ V1.1
4) Documented in TC29x B-step User’s Manual ≥ V1.2
5) Documented in Data Sheet, section “Reset Timings”
6) See SCU_TC.H009 (LBIST Influence on Pad Behavior)

Note: Changes to the previous errata sheet version are particularly marked in column “Change” in the following tables.
## Table 4 Functional Deviations

<table>
<thead>
<tr>
<th>Functional Deviation</th>
<th>Short Description</th>
<th>Change</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADC_AI.016</td>
<td>No Channel Interrupt in Fast Compare Mode with GLOBRES</td>
<td></td>
<td>33</td>
</tr>
<tr>
<td>ADC_TC.068</td>
<td>Effect of VAGND Cross Coupling on Conversion Result</td>
<td></td>
<td>33</td>
</tr>
<tr>
<td>ASCLIN_TC.004</td>
<td>SLSO in SPI mode still active after module disable</td>
<td></td>
<td>37</td>
</tr>
<tr>
<td>ASCLIN_TC.005</td>
<td>Unjustified collision detection error in half-duplex SPI mode</td>
<td></td>
<td>38</td>
</tr>
<tr>
<td>ASCLIN_TC.006</td>
<td>Unjustified response timeout in LIN slave mode</td>
<td></td>
<td>38</td>
</tr>
<tr>
<td>ASCLIN_TC.007</td>
<td>Break Detected in LIN Frames in Soft Suspend mode</td>
<td></td>
<td>38</td>
</tr>
<tr>
<td>ASCLIN_TC.008</td>
<td>Response timeout in LIN Mode in case of header only</td>
<td></td>
<td>39</td>
</tr>
<tr>
<td>ASCLIN_TC.009</td>
<td>RFL flag set in Buffer Mode when Receive FIFO Inlet is disabled</td>
<td></td>
<td>39</td>
</tr>
<tr>
<td>ASCLIN_TC.010</td>
<td>Flush of TXFIFO leads to frame transmission</td>
<td></td>
<td>39</td>
</tr>
<tr>
<td>BROM_TC.008</td>
<td>Sporadic Power-on Reset after Wake-up from Standby Mode</td>
<td></td>
<td>40</td>
</tr>
<tr>
<td>CCU_TC.002</td>
<td>Clock Monitors - Target Monitoring Frequency Selection</td>
<td></td>
<td>41</td>
</tr>
<tr>
<td>CIF_TC.013</td>
<td>DMA Access to Reserved/Protected Resources: FPI Error Response not correctly evaluated</td>
<td></td>
<td>41</td>
</tr>
<tr>
<td>CIF_TC.014</td>
<td>CIF Module Sub-Resets</td>
<td></td>
<td>42</td>
</tr>
<tr>
<td>CIF_TC.015</td>
<td>Security Watchdog Interrupt Control - Documentation Update</td>
<td>New</td>
<td>42</td>
</tr>
</tbody>
</table>
# Table 4 Functional Deviations (cont’d)

<table>
<thead>
<tr>
<th>Functional Deviation</th>
<th>Short Description</th>
<th>Change</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU_TC.123</td>
<td>Data Corruption possible when CPU GPR accesses made via SRI slave with CPU running</td>
<td></td>
<td>43</td>
</tr>
<tr>
<td>CPU_TC.126</td>
<td>Unexpected Address Error Alarms caused by Speculative Access to Out-of-range PMEM and DMEM Areas</td>
<td></td>
<td>44</td>
</tr>
<tr>
<td>CPU_TC.127</td>
<td>Pending Interrupt Priority Number PIPN in Register ICR</td>
<td></td>
<td>49</td>
</tr>
<tr>
<td>DAP_TC.002</td>
<td>DAP client_blockread has Performance issue in Specific Operation Modes</td>
<td></td>
<td>49</td>
</tr>
<tr>
<td>DAP_TC.003</td>
<td>DAP CRC32 definition and algorithm</td>
<td></td>
<td>50</td>
</tr>
<tr>
<td>DAP_TC.004</td>
<td>DAP client_blockwrite telegram with CRC6 and CRC32 protection options</td>
<td></td>
<td>51</td>
</tr>
<tr>
<td>DAP_TC.005</td>
<td>DAP client_read: dirty bit feature of Cerberus’ Triggered Transfer Mode</td>
<td></td>
<td>52</td>
</tr>
<tr>
<td>DAP_TC.006</td>
<td>CRC6 error in telegram following a get_CRCdown telegram prevents reset of CRC32 calculator</td>
<td>New</td>
<td>52</td>
</tr>
<tr>
<td>DAP_TC.009</td>
<td>CRC6 error in client_blockwrite telegram</td>
<td>New</td>
<td>53</td>
</tr>
<tr>
<td>DMA_TC.015</td>
<td>DMA Double Buffering: No Timestamp Support</td>
<td></td>
<td>53</td>
</tr>
<tr>
<td>DMA_TC.016</td>
<td>Byte and Half-word Write Accesses to specific Registers not supported</td>
<td></td>
<td>54</td>
</tr>
<tr>
<td>DMA_TC.017</td>
<td>Pattern Detection Double Interrupt Trigger when INTCT = 11B</td>
<td></td>
<td>54</td>
</tr>
<tr>
<td>DMA_TC.018</td>
<td>FPI timeout can cause pipelined register reads to break</td>
<td></td>
<td>55</td>
</tr>
<tr>
<td>DMA_TC.019</td>
<td>CBS Accesses with Large SPB:SRI Clock Ratios Configured</td>
<td></td>
<td>56</td>
</tr>
<tr>
<td>Functional Deviation</td>
<td>Short Description</td>
<td>Page</td>
<td></td>
</tr>
<tr>
<td>----------------------</td>
<td>------------------------------------------------------------------------------------</td>
<td>------</td>
<td></td>
</tr>
<tr>
<td>DMA_TC.020</td>
<td>DMA Conditional Linked List: Circular Buffer Enabled</td>
<td>56</td>
<td></td>
</tr>
<tr>
<td>DMA_TC.021</td>
<td>Combined Software/Hardware Controlled Mode Spurious Errors</td>
<td>56</td>
<td></td>
</tr>
<tr>
<td>DMA_TC.022</td>
<td>Conditional Linked List: Bus Error</td>
<td>57</td>
<td></td>
</tr>
<tr>
<td>DMA_TC.024</td>
<td>Suspend Request coincident with Channel Activation</td>
<td>58</td>
<td></td>
</tr>
<tr>
<td>DMA_TC.025</td>
<td>Conditional Linked List: new non-CLL mode TCS load can corrupt SDCRC RAM write</td>
<td>58</td>
<td></td>
</tr>
<tr>
<td>DMA_TC.026</td>
<td>Linked List: Failed TCS load can trigger wrap interrupt</td>
<td>58</td>
<td></td>
</tr>
<tr>
<td>DMA_TC.028</td>
<td>Transaction Request Lost interrupt behaviour</td>
<td>59</td>
<td></td>
</tr>
<tr>
<td>DMA_TC.029</td>
<td>DMA Double Buffering Overflow</td>
<td>60</td>
<td></td>
</tr>
<tr>
<td>DMA_TC.031</td>
<td>CHCSR.ICH can be incorrectly set after pattern match</td>
<td>60</td>
<td></td>
</tr>
<tr>
<td>DMA_TC.034</td>
<td>Timestamp written incorrectly to wrapped address</td>
<td>61</td>
<td></td>
</tr>
<tr>
<td>DMA_TC.035</td>
<td>Last DMA Transaction in a Linked List triggers a DMA Daisy Chain</td>
<td>61</td>
<td></td>
</tr>
<tr>
<td>DMA_TC.036</td>
<td>Linked List: SADR/DADR can be overwritten when loading a non-LL TCS</td>
<td>62</td>
<td></td>
</tr>
<tr>
<td>DMA_TC.037</td>
<td>Conditional Linked List: Bit TSR.CH not cleared for a CLL transaction upon pattern match</td>
<td>62</td>
<td></td>
</tr>
<tr>
<td>DMA_TC.038</td>
<td>Linked List: SIT interrupt when SIT bit set in newly loaded TCS</td>
<td>63</td>
<td></td>
</tr>
<tr>
<td>DMA_TC.039</td>
<td>Read Data CRC</td>
<td>63</td>
<td></td>
</tr>
</tbody>
</table>
## Table 4  Functional Deviations (cont’d)

<table>
<thead>
<tr>
<th>Functional Deviation</th>
<th>Short Description</th>
<th>Change</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>DMA_TC.040</td>
<td>DMA Linked Lists: Intermittent Clearing of Hardware Transaction Request Enable with mixed mode Transaction Control Sets</td>
<td></td>
<td>63</td>
</tr>
<tr>
<td>DMA_TC.041</td>
<td>DMA Circular Buffer Wrap Interrupt</td>
<td></td>
<td>64</td>
</tr>
<tr>
<td>DMA_TC.042</td>
<td>DMA Interrupt from Channel reported before Completion of DMA Transaction</td>
<td></td>
<td>65</td>
</tr>
<tr>
<td>DMA_TC.043</td>
<td>DMA Write Move Data Corruption for non 32-byte Aligned Cacheable Source Address</td>
<td></td>
<td>66</td>
</tr>
<tr>
<td>DMA_TC.044</td>
<td>Clock Switch after SPB Error Reported results in Spurious SRI Error</td>
<td></td>
<td>66</td>
</tr>
<tr>
<td>DMA_TC.045</td>
<td>DMA Reconfigures DMA Channels Lockup</td>
<td></td>
<td>67</td>
</tr>
<tr>
<td>DMA_TC.046</td>
<td>Shadow Operation Read Only Mode</td>
<td></td>
<td>67</td>
</tr>
<tr>
<td>DMA_TC.047</td>
<td>DMA Double Buffering Buffer Switch</td>
<td></td>
<td>68</td>
</tr>
<tr>
<td>DMA_TC.048</td>
<td>DMARAM Internal ECC Error</td>
<td></td>
<td>69</td>
</tr>
<tr>
<td>DMA_TC.049</td>
<td>Bus Error Reported During LL TCS Load</td>
<td></td>
<td>69</td>
</tr>
<tr>
<td>DMA_TC.050</td>
<td>Clearing CHCSR.FROZEN during Double Buffering</td>
<td></td>
<td>70</td>
</tr>
<tr>
<td>DMA_TC.051</td>
<td>DMARAM Alarm</td>
<td></td>
<td>70</td>
</tr>
<tr>
<td>DMA_TC.052</td>
<td>SER and DER During Linked List Operations</td>
<td></td>
<td>71</td>
</tr>
<tr>
<td>DMA_TC.053</td>
<td>TS16_ERR Type of Error Reporting Unreliable</td>
<td>New</td>
<td>71</td>
</tr>
<tr>
<td>DMA_TC.054</td>
<td>DMA Channel Halt Acknowledge Unreliable</td>
<td>New</td>
<td>71</td>
</tr>
<tr>
<td>DMA_TC.055</td>
<td>ICU to DMA Interface in Sleep Mode</td>
<td>New</td>
<td>72</td>
</tr>
<tr>
<td>DMA_TC.056</td>
<td>TSR and SUSENR Access Protection Unreliable</td>
<td>New</td>
<td>72</td>
</tr>
</tbody>
</table>
Table 4  Functional Deviations (cont’d)

<table>
<thead>
<tr>
<th>Functional Deviation</th>
<th>Short Description</th>
<th>Change</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>DMA_TC.057</td>
<td>Double Buffering Overflow Causes Other Channel Corruption</td>
<td>New</td>
<td>74</td>
</tr>
<tr>
<td>DMA_TC.058</td>
<td>Linked List Load Transaction Control Set (TCS) Integrity Error</td>
<td>New</td>
<td>75</td>
</tr>
<tr>
<td>DSADC_TC.004</td>
<td>Byte Write Access to CH8RUN, M8RUN, CH9RUN, M9RUN</td>
<td></td>
<td>76</td>
</tr>
<tr>
<td>DSADC_TC.011</td>
<td>Modulator Coupling Option no longer supported</td>
<td></td>
<td>76</td>
</tr>
<tr>
<td>DSADC_TC.012</td>
<td>Common Mode Hold Voltage Not Applied During Calibration</td>
<td></td>
<td>77</td>
</tr>
<tr>
<td>DSADC_TC.013</td>
<td>Common Mode Voltage Selection</td>
<td></td>
<td>77</td>
</tr>
<tr>
<td>DTS_TC.001</td>
<td>Temperature Sensor Formula</td>
<td></td>
<td>78</td>
</tr>
<tr>
<td>EBU_TC.024</td>
<td>Dynamic Changes of SDRAM clock are not possible</td>
<td></td>
<td>79</td>
</tr>
<tr>
<td>EBU_TC.025</td>
<td>Connection of SDRAM Devices not supported</td>
<td></td>
<td>79</td>
</tr>
<tr>
<td>ETH_AI.003</td>
<td>Overflow Status bits of Missed Frame and Buffer Overflow counters get cleared without a Read operation</td>
<td></td>
<td>80</td>
</tr>
<tr>
<td>ETH_TC.004</td>
<td>DMA Access to Reserved/Protected Resources: FPI Error Response not correctly evaluated</td>
<td></td>
<td>80</td>
</tr>
<tr>
<td>FFT_TC.001</td>
<td>FFT Access with disabled FFT Module</td>
<td></td>
<td>81</td>
</tr>
<tr>
<td>FFT_TC.002</td>
<td>FFT Kernel Reset Function</td>
<td></td>
<td>81</td>
</tr>
<tr>
<td>FFT_TC.003</td>
<td>No Error reported upon Write to FFT Registers in User Mode</td>
<td></td>
<td>82</td>
</tr>
<tr>
<td>FLASH_TC.044</td>
<td>Repetitive Erase Suspend Requests on Data Flash</td>
<td></td>
<td>82</td>
</tr>
<tr>
<td>Functional Deviation</td>
<td>Short Description</td>
<td>Change</td>
<td>Page</td>
</tr>
<tr>
<td>----------------------</td>
<td>------------------------------------------------------------------------------------</td>
<td>--------</td>
<td>------</td>
</tr>
<tr>
<td>FlexRay_AI.087</td>
<td>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</td>
<td></td>
<td>83</td>
</tr>
<tr>
<td>FlexRay_AI.088</td>
<td>A sequence of received WUS may generate redundant SIR.WUPA/B events</td>
<td></td>
<td>84</td>
</tr>
<tr>
<td>FlexRay_AI.089</td>
<td>Rate correction set to zero in case of SyncCalcResult=MISSING_TERM</td>
<td></td>
<td>85</td>
</tr>
<tr>
<td>FlexRay_AI.090</td>
<td>Flag SFS.MRCS is set erroneously although at least one valid sync frame pair is received</td>
<td></td>
<td>86</td>
</tr>
<tr>
<td>FlexRay_AI.091</td>
<td>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</td>
<td></td>
<td>86</td>
</tr>
<tr>
<td>FlexRay_AI.092</td>
<td>Initial rate correction value of an integrating node is zero if pMicroInitialOffsetA,B = 0x00</td>
<td></td>
<td>87</td>
</tr>
<tr>
<td>FlexRay_AI.093</td>
<td>Acceptance of startup frames received after reception of more than gSyncNodeMax sync frames</td>
<td></td>
<td>88</td>
</tr>
<tr>
<td>FlexRay_AI.094</td>
<td>Sync frame overflow flag EIR.SFO may be set if slot counter is greater than 1024</td>
<td></td>
<td>88</td>
</tr>
<tr>
<td>FlexRay_AI.095</td>
<td>Register RCV displays wrong value</td>
<td></td>
<td>89</td>
</tr>
<tr>
<td>FlexRay_AI.096</td>
<td>Noise following a dynamic frame that delays idle detection may fail to stop slot</td>
<td></td>
<td>90</td>
</tr>
<tr>
<td>FlexRay_AI.097</td>
<td>Loop back mode operates only at 10 MBit/s</td>
<td></td>
<td>91</td>
</tr>
<tr>
<td>FlexRay_AI.099</td>
<td>Erroneous cycle offset during startup after abort of startup or normal operation</td>
<td></td>
<td>91</td>
</tr>
<tr>
<td>Functional Deviation</td>
<td>Short Description</td>
<td>Change</td>
<td>Page</td>
</tr>
<tr>
<td>----------------------</td>
<td>------------------------------------------------------------------------------------</td>
<td>--------</td>
<td>------</td>
</tr>
<tr>
<td>FlexRay_Ai.100</td>
<td>First WUS following received valid WUP may be ignored</td>
<td></td>
<td>92</td>
</tr>
<tr>
<td>FlexRay_Ai.101</td>
<td>READY command accepted in READY state</td>
<td></td>
<td>93</td>
</tr>
<tr>
<td>FlexRay_Ai.102</td>
<td>Slot Status vPOC!SlotMode is reset immediately when entering HALT state</td>
<td></td>
<td>93</td>
</tr>
<tr>
<td>FlexRay_Ai.103</td>
<td>Received messages not stored in Message RAM when in Loop Back Mode</td>
<td></td>
<td>94</td>
</tr>
<tr>
<td>FlexRay_Ai.104</td>
<td>Missing startup frame in cycle 0 at coldstart after FREEZE or READY command</td>
<td></td>
<td>94</td>
</tr>
<tr>
<td>FlexRay_Ai.105</td>
<td>RAM select signals of IBF1/IBF2 and OBF1/OBF2 in RAM test mode</td>
<td></td>
<td>95</td>
</tr>
<tr>
<td>FlexRay_Ai.106</td>
<td>Data transfer overrun for message transfers Message RAM to Output Buffer (OBF) or from Input Buffer (IBF) to Message RAM</td>
<td></td>
<td>96</td>
</tr>
<tr>
<td>GTM_Ai.139</td>
<td>ATOM SOMC mode: forced update does not activate comparison</td>
<td></td>
<td>99</td>
</tr>
<tr>
<td>GTM_Ai.140</td>
<td>ATOM SOMC mode: a write access to ATOM_CH_CTRL sets WRF if CCU0 compare match already occurred but CCU1 compare match open</td>
<td></td>
<td>100</td>
</tr>
<tr>
<td>GTM_Ai.141</td>
<td>TIM: Incorrect data captured to GPR registers and routed via ARU when EGPRi_SEL,GPRi_SEL= 100 in TIM channel mode TIEM, TPWM, TIPM, TPIM, TGPS</td>
<td></td>
<td>101</td>
</tr>
</tbody>
</table>
Table 4  Functional Deviations (cont’d)

<table>
<thead>
<tr>
<th>Functional Deviation</th>
<th>Short Description</th>
<th>Change</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>GTM_AI.142</td>
<td>TIM: Incorrect data captured to GPR registers and routed via ARU when EGPRi_SEL,GPRi_SEL= 100 in TIM channel mode TBCM</td>
<td></td>
<td>102</td>
</tr>
<tr>
<td>GTM_AI.143</td>
<td>GTM_TOP level: AEI pipelined write to GTM_BRIDGE_MODE register directly after setting aei_reset='0' can result in blocking of AEI configuration interface</td>
<td></td>
<td>103</td>
</tr>
<tr>
<td>GTM_AI.146</td>
<td>ATOM SOMC mode: compare match does not clear WR_REQ</td>
<td></td>
<td>103</td>
</tr>
<tr>
<td>GTM_AI.150</td>
<td>TIM: Valid edge after Timeout</td>
<td></td>
<td>104</td>
</tr>
<tr>
<td>GTM_AI.152</td>
<td>DPLL: THVAL value not immediately available at inactive trigger slope</td>
<td></td>
<td>105</td>
</tr>
<tr>
<td>GTM_AI.153</td>
<td>TIM: Incorrect data captured to CNTS register when TIM channel operates in mode TPWM or TPIM and CNTS_SEL = 1 and selected CMU_CLK ≠ sys_clk</td>
<td></td>
<td>105</td>
</tr>
<tr>
<td>GTM_AI.154</td>
<td>TOM: Incorrect duty cycle in PCM mode (bit reversed mode)</td>
<td></td>
<td>106</td>
</tr>
<tr>
<td>GTM_AI.158</td>
<td>DPLL: Reset of pcm1/pcm2 bits in relation to an interrupt</td>
<td></td>
<td>107</td>
</tr>
<tr>
<td>GTM_AI.161</td>
<td>DPLL MTI/TORI-IRQ's are not activated when low_res='1' and ts0_hrt='1'; MSI/SORI-IRQ's are not activated when low_res='1' and ts0_hrs='1'</td>
<td></td>
<td>108</td>
</tr>
<tr>
<td>GTM_AI.162</td>
<td>DPLL: Input signal (active edge) which is rejected by PVT-check occurring at a gap in the profile causes that the MTI_IRQ is not activated within this gap</td>
<td></td>
<td>108</td>
</tr>
<tr>
<td>GTM_AI.163</td>
<td>TIM: timeout signaled when TDU unit is reenabled</td>
<td></td>
<td>109</td>
</tr>
<tr>
<td>Functional Deviation</td>
<td>Short Description</td>
<td>Change</td>
<td>Page</td>
</tr>
<tr>
<td>----------------------</td>
<td>------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>--------</td>
<td>------</td>
</tr>
<tr>
<td>GTM_AI.164</td>
<td>TIM: capturing of data into TIM[\textit{i}]_CH[x]_CNTS with setting CNTS_SEL=1 not functional in TPWM and TPIM mode</td>
<td></td>
<td>110</td>
</tr>
<tr>
<td>GTM_AI.166</td>
<td>DPLL: The Content of registers DPLL_apt_sync.APT_2b_ext and DPLL_aps_sync.APS_1c2_ext is added independently of the state of DPLL_apt_sync.APT_2b_status or DPLL_aps_sync.APS_1c2_status to the pointers apt_2b/aps_1c2</td>
<td></td>
<td>110</td>
</tr>
<tr>
<td>GTM_AI.167</td>
<td>ATOM SOMP mode: for RST_CCU0=1 and ARU_EN=1, if CN0 reaches CM0 an update of the register SRx is requested</td>
<td></td>
<td>111</td>
</tr>
<tr>
<td>GTM_AI.168</td>
<td>DPLL: CPU read / write accesses to RAM2 in competition to DPLL accesses to RAM2 may lead to wrong SYN_T data read by DPLL</td>
<td></td>
<td>112</td>
</tr>
<tr>
<td>GTM_AI.169</td>
<td>DPLL: no TORI/SORI interrupt in case low_res = 1 AND ts0_hrt/s = 0</td>
<td></td>
<td>114</td>
</tr>
<tr>
<td>GTM_AI.170</td>
<td>DPLL: Action calculation: requested action not always calculated immediately</td>
<td></td>
<td>114</td>
</tr>
<tr>
<td>GTM_AI.172</td>
<td>TIM: overflow bit in TIM ARU data not set; signal level bit in ARU data has opposite value</td>
<td></td>
<td>116</td>
</tr>
<tr>
<td>GTM_AI.173</td>
<td>DPLL: new PMT data not received</td>
<td></td>
<td>117</td>
</tr>
<tr>
<td>GTM_AI.174</td>
<td>DPLL: PMT result not sent to ARU</td>
<td></td>
<td>118</td>
</tr>
<tr>
<td>GTM_AI.178</td>
<td>MCS: Evaluation of CAT bit after blocking ARU instruction</td>
<td></td>
<td>119</td>
</tr>
<tr>
<td>GTM_AI.181</td>
<td>TIM: Incorrect signal level bit ECNT[0] in mode TIEM, TPWM, TIPM, TPIM, TGPS</td>
<td></td>
<td>120</td>
</tr>
<tr>
<td>Functional Deviation</td>
<td>Short Description</td>
<td>Change</td>
<td>Page</td>
</tr>
<tr>
<td>----------------------</td>
<td>------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>--------</td>
<td>------</td>
</tr>
<tr>
<td>GTM_Ai.202</td>
<td>(A)TOM: no CCU1 interrupt in case of CM1=0 or 1 and RST_CCU0=1</td>
<td></td>
<td>121</td>
</tr>
<tr>
<td>GTM_Ai.204</td>
<td>TIM: incorrect signal level on TIM_MODE change if TIM channel is disabled</td>
<td></td>
<td>121</td>
</tr>
<tr>
<td>GTM_Ai.205</td>
<td>TIM: unexpected CNTS register update in TPWM OSM mode</td>
<td></td>
<td>122</td>
</tr>
<tr>
<td>GTM_Ai.208</td>
<td>DPLL: Start of sub-increment generation and action calculation delayed by one input event if PCM1/2 bits are set and DPLL_STATUS.FTD = '0'</td>
<td></td>
<td>122</td>
</tr>
<tr>
<td>GTM_Ai.209</td>
<td>TOM/ATOM: no update of CM0/CM1/CLK_SRC via trigger signal from preceding instance if selected CMU_CLKx is not SYS_CLK</td>
<td></td>
<td>123</td>
</tr>
<tr>
<td>GTM_Ai.210</td>
<td>ATOM: data loss in SOMS one-shot mode if ARU is enabled and the period of the selected CMU_CLKx is greater than ARU-cycle-time/2</td>
<td></td>
<td>124</td>
</tr>
<tr>
<td>GTM_Ai.212</td>
<td>F2A: stream data register will not be deleted after disabling stream</td>
<td></td>
<td>125</td>
</tr>
<tr>
<td>GTM_Ai.215</td>
<td>FIFO: read pointer will be incremented in ring buffer mode on empty FIFO channel with read access from AFD_CHx_BUF_ACC</td>
<td></td>
<td>125</td>
</tr>
<tr>
<td>GTM_Ai.218</td>
<td>DPLL: PWI-IRQ permanently activated</td>
<td></td>
<td>126</td>
</tr>
<tr>
<td>GTM_Ai.219</td>
<td>DPLL: Wrong internal pointer calculation in case of backwards direction can lead to wrong PMT calculation results (PMT in PAST)</td>
<td></td>
<td>127</td>
</tr>
</tbody>
</table>
### Table 4  Functional Deviations (cont’d)

<table>
<thead>
<tr>
<th>Functional Deviation</th>
<th>Short Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>GTM_AI.220</td>
<td>DPLL: PVT check is deactivated in case of direction change; Behaviour implemented but not documented in specification so far</td>
<td>127</td>
</tr>
<tr>
<td>GTM_AI.221</td>
<td>DPLL: Possible inconsistency of internal pointers and parameter NUTE/NUSE when NUTE/NUSE modified in dedicated time window</td>
<td>128</td>
</tr>
<tr>
<td>GTM_AI.222</td>
<td>DPLL: TAXI-irq not deactivated for THMA=0</td>
<td>129</td>
</tr>
<tr>
<td>GTM_AI.223</td>
<td>DPLL: discontinuities in the sub increments when DPLL_NUTC/S.FST/FSS=1; set to full scale</td>
<td>129</td>
</tr>
<tr>
<td>GTM_AI.247</td>
<td>DPLL: Input event not served after DPLL_CTRL_1.DEN is activated</td>
<td>130</td>
</tr>
<tr>
<td>GTM_AI.250</td>
<td>DPLL: DPLL_STATUS.BWD1/2 not reset after DPLL_CTRL_1.DEN = 1-&gt;0-&gt;1, when DPLL_CTRL_0 has been written some time before</td>
<td>131</td>
</tr>
<tr>
<td>GTM_AI.260</td>
<td>TOM/ATOM: Async. update in SOMP mode with CM1=0 and selected CMU clock unequal sys_clk not functional</td>
<td>132</td>
</tr>
<tr>
<td>GTM_AI.270</td>
<td>(A)TOM: output signal is postponed one period for the values CM0=1 and CM1&gt;CM0 if CN0 is reset by the trigger of a preceding channel (RST_CCU0=1)</td>
<td>132</td>
</tr>
<tr>
<td>GTM_AI.271</td>
<td>DPLL: No DCGI-irq after direction change and DPLL_CTRL_0 has been written</td>
<td>133</td>
</tr>
<tr>
<td>GTM_AI.272</td>
<td>DPLL: No update of DPLL_RAM1b.PSTC after direction change and DPLL_CTRL_0 has been written</td>
<td>135</td>
</tr>
<tr>
<td>Functional Deviation</td>
<td>Short Description</td>
<td>Change</td>
</tr>
<tr>
<td>----------------------</td>
<td>------------------------------------------------------------------------------------</td>
<td>--------</td>
</tr>
<tr>
<td>GTM_AI.278</td>
<td>FIFO: Restoring of F2A (ARU to FIFO interface) read access to FIFO after GTM_HALT condition not functional</td>
<td>New</td>
</tr>
<tr>
<td>GTM_AI.292</td>
<td>DPLL: pulse correction at direction change incompletely for DPLL_CTRL_1.SMC='1'</td>
<td>New</td>
</tr>
<tr>
<td>GTM_AI.300</td>
<td>DPLL: Change to forward operation when DPLL_THMI is set to zero does not work correctly</td>
<td>New</td>
</tr>
<tr>
<td>GTM_AI.301</td>
<td>DPLL: Reset of DPLL_STATUS.BWD1=1 by disabling the DPLL does not cause the direction to change from backward to forward in any case</td>
<td>New</td>
</tr>
<tr>
<td>GTM_AI.302</td>
<td>DPLL: Pulse generation ongoing for DPLL_CTRL_1.DMO=1 (continuous mode) if DPLL_CTRL_1.sge1/2=0</td>
<td>New</td>
</tr>
<tr>
<td>GTM_AI.306</td>
<td>DPLL: DPLL_NUTC.syn_t_old, DPLL_NUSC.syn_s_old not updated according specification</td>
<td>New</td>
</tr>
<tr>
<td>GTM_AI.320</td>
<td>ATOM: Unexpected restart of a SOMS oneshot cycle while ATOM[i]_CH[x]_CM0 is zero</td>
<td>New</td>
</tr>
<tr>
<td>GTM_TC.010</td>
<td>Effects of GTM Resets</td>
<td></td>
</tr>
<tr>
<td>GTM_TC.012</td>
<td>Read Access Control by Register ODA</td>
<td></td>
</tr>
<tr>
<td>HSCT_TC.007</td>
<td>RX_FIFO overflow interrupt</td>
<td></td>
</tr>
<tr>
<td>HSCT_TC.009</td>
<td>Sleep mode not to be used</td>
<td></td>
</tr>
<tr>
<td>HSCT_TC.010</td>
<td>Master Mode Interface Test Mode not working</td>
<td></td>
</tr>
<tr>
<td>I2C_TC.001</td>
<td>I2C FIFO data buffer does not support double buffering</td>
<td></td>
</tr>
</tbody>
</table>
### Table 4  Functional Deviations (cont’d)

<table>
<thead>
<tr>
<th>Functional Deviation</th>
<th>Short Description</th>
<th>Change</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>I2C_TC.003</td>
<td>Limits on selectable INC and DEC values</td>
<td></td>
<td>146</td>
</tr>
<tr>
<td>I2C_TC.004</td>
<td>High speed mode: SCL clock ratio 1:2</td>
<td></td>
<td>147</td>
</tr>
<tr>
<td>I2C_TC.005</td>
<td>Hold Time Start violation in Multi-Master Mode</td>
<td></td>
<td>148</td>
</tr>
<tr>
<td>IOM_TC.002</td>
<td>Missed or spurious IOM events when pulse length exceeds Event Window counter range</td>
<td></td>
<td>148</td>
</tr>
<tr>
<td>IOM_TC.003</td>
<td>Unexpected Event upon Kernel Reset</td>
<td></td>
<td>149</td>
</tr>
<tr>
<td>IOM_TC.004</td>
<td>Write to IOM register space when IOM_CLC.RMC &gt; 1</td>
<td></td>
<td>149</td>
</tr>
<tr>
<td>miniMCDS_TC.002</td>
<td>Incorrect nested interrupt trace indication</td>
<td></td>
<td>149</td>
</tr>
<tr>
<td>MSC_TC.012</td>
<td>Increased Jitter for Data Frame Transmission in Repetition Mode with ABRA</td>
<td></td>
<td>150</td>
</tr>
<tr>
<td>MSC_TC.013</td>
<td>Missing Chip Select for Command Frame with Length zero</td>
<td></td>
<td>151</td>
</tr>
<tr>
<td>MSC_TC.014</td>
<td>Upstream Timeout Interrupt cannot be issued at Service Request Output SR4</td>
<td></td>
<td>152</td>
</tr>
<tr>
<td>MSC_TC.015</td>
<td>Emergency Stop not effective at Injected Bit Positions in Downstream Frame</td>
<td></td>
<td>152</td>
</tr>
<tr>
<td>MSC_TC.016</td>
<td>MSC Spikes on Data and Enable Signals</td>
<td></td>
<td>153</td>
</tr>
<tr>
<td>MTU_TC.005</td>
<td>Access to MCx_ECCD and MCx_ETRRi while MBIST disabled</td>
<td></td>
<td>155</td>
</tr>
<tr>
<td>MTU_TC.006</td>
<td>Bus error generation upon access to non-existing MBIST registers</td>
<td></td>
<td>157</td>
</tr>
<tr>
<td>MTU_TC.011</td>
<td>MBIST Bitmap not working for w0 - r1</td>
<td></td>
<td>157</td>
</tr>
<tr>
<td>MTU_TC.012</td>
<td>Security of CPU Cache Memories During Runtime is Limited</td>
<td></td>
<td>158</td>
</tr>
<tr>
<td>Functional Deviation</td>
<td>Short Description</td>
<td>Change</td>
<td>Page</td>
</tr>
<tr>
<td>----------------------</td>
<td>-------------------</td>
<td>--------</td>
<td>------</td>
</tr>
<tr>
<td><strong>MultiCAN_AI.047</strong></td>
<td>Transmit Frame Corruption after Protocol Exception (CAN FD only)</td>
<td></td>
<td>158</td>
</tr>
<tr>
<td><strong>OCDS_TC.038</strong></td>
<td>Disconnecting a debugger without device reset (“hot detach”) may require reading of OCS registers</td>
<td></td>
<td>159</td>
</tr>
<tr>
<td><strong>OCDS_TC.040</strong></td>
<td>DAP turn_off to JTAG telegram not working properly</td>
<td></td>
<td>160</td>
</tr>
<tr>
<td><strong>OCDS_TC.042</strong></td>
<td>OTGS capture registers can miss single clock cycle triggers</td>
<td></td>
<td>161</td>
</tr>
<tr>
<td><strong>OCDS_TC.043</strong></td>
<td>Read-Modify-Write Bus Transactions to Cerberus Registers</td>
<td></td>
<td>162</td>
</tr>
<tr>
<td><strong>PLL_ERAY_TC.001</strong></td>
<td>PLL_ERAY Initialization after Cold Power-up or Wake-up from Standby mode</td>
<td><strong>Update</strong></td>
<td>162</td>
</tr>
<tr>
<td><strong>PLL_TC.005</strong></td>
<td>PLL Initialization after Cold Power-up or Wake-up from Standby mode</td>
<td><strong>Update</strong></td>
<td>163</td>
</tr>
<tr>
<td><strong>PORTS_TC.002</strong></td>
<td>Behavior of P21 Port Pins upon Power-on Reset</td>
<td></td>
<td>163</td>
</tr>
<tr>
<td><strong>QSPI_TC.006</strong></td>
<td>Baud rate error detection in slave mode (error indication in current frame)</td>
<td></td>
<td>164</td>
</tr>
<tr>
<td><strong>RESET_TC.005</strong></td>
<td>Indication of Power Fail Events in SCU_RSTSTAT</td>
<td></td>
<td>165</td>
</tr>
<tr>
<td><strong>SMU_TC.005</strong></td>
<td>Unexpected/Incorrect Reset caused by SMU Alarms</td>
<td></td>
<td>165</td>
</tr>
<tr>
<td><strong>SMU_TC.006</strong></td>
<td>OCDS Trigger Bus OTGB during Application Reset</td>
<td></td>
<td>170</td>
</tr>
<tr>
<td><strong>SMU_TC.007</strong></td>
<td>Size and Position of Field ACNT in Register SMU_AFCNT</td>
<td></td>
<td>170</td>
</tr>
<tr>
<td><strong>SMU_TC.008</strong></td>
<td>Behavior of Action Counter ACNT</td>
<td></td>
<td>171</td>
</tr>
</tbody>
</table>
### Table 4  Functional Deviations (cont’d)

<table>
<thead>
<tr>
<th>Functional Deviation</th>
<th>Short Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>SMU_TC.010</td>
<td>Transfer to SMU_AD register not triggered correctly</td>
<td>172</td>
</tr>
</tbody>
</table>

### Table 5  Deviations from Electrical- and Timing Specification

<table>
<thead>
<tr>
<th>AC/DC/ADC Deviation</th>
<th>Short Description</th>
<th>Change</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADC_TC.P007</td>
<td>Additional Parameter for Data Sheet: Wakeup Time $t_{WU}$</td>
<td></td>
<td>173</td>
</tr>
<tr>
<td>ADC_TC.P011</td>
<td>Leakage current for ADC reference pins VAREF, VAGND</td>
<td>New</td>
<td>173</td>
</tr>
<tr>
<td>CHIP_TC.P001</td>
<td>TC290 Chip Dimensions</td>
<td></td>
<td>176</td>
</tr>
<tr>
<td>EBU_TC.P002</td>
<td>EBU Timing Deviations</td>
<td></td>
<td>176</td>
</tr>
<tr>
<td>FlexRay_TC.P002</td>
<td>Pad Configuration for E-Ray Parameters</td>
<td></td>
<td>179</td>
</tr>
<tr>
<td>IDD_TC.H001</td>
<td>IPC Limits used in Production Test for IDD Max Power Pattern</td>
<td></td>
<td>179</td>
</tr>
<tr>
<td>PADS_TC.P002</td>
<td>Restrictions for P00.1 .. P00.12 if $V_{DDM}$ is lower than $V_{EXT}$</td>
<td></td>
<td>180</td>
</tr>
<tr>
<td>PADS_TC.P003</td>
<td>Input Frequency $f_{IN}$ for Class S Pads</td>
<td></td>
<td>180</td>
</tr>
<tr>
<td>PADS_TC.P004</td>
<td>Input Low / High Voltage for LP Pad $V_{ILHLP}$ - Condition</td>
<td></td>
<td>181</td>
</tr>
<tr>
<td>PADS_TC.P006</td>
<td>P21.6/P21.7 Pull-up Reset Behavior</td>
<td></td>
<td>181</td>
</tr>
<tr>
<td>PADS_TC.P009</td>
<td>Bonding of VGATE1P on Bare Die Variants</td>
<td></td>
<td>181</td>
</tr>
<tr>
<td>PWR_TC.P013</td>
<td>EVR Supply Voltage $V_{EXT}$ Ramp-up</td>
<td></td>
<td>182</td>
</tr>
</tbody>
</table>
Table 6  Application Hints

<table>
<thead>
<tr>
<th>Hint</th>
<th>Short Description</th>
<th>Change</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADC_Ai.H003</td>
<td>Injected conversion may be performed with sample time of aborted conversion</td>
<td></td>
<td>184</td>
</tr>
<tr>
<td>ADC_TC.H014</td>
<td>VADC Start-up Calibration</td>
<td></td>
<td>185</td>
</tr>
<tr>
<td>ADC_TC.H015</td>
<td>Conversion Time with Broken Wire Detection</td>
<td></td>
<td>186</td>
</tr>
<tr>
<td>ADC_TC.H016</td>
<td>P02 Output Driver Setting for External Multiplexer Control</td>
<td></td>
<td>187</td>
</tr>
<tr>
<td>ADC_TC.H019</td>
<td>G6ARBCNT Connection to GTM</td>
<td></td>
<td>188</td>
</tr>
<tr>
<td>ADC_TC.H020</td>
<td>Minimum/Maximum Detection Compares 12 Bits Only</td>
<td></td>
<td>188</td>
</tr>
<tr>
<td>ADC_TC.H022</td>
<td>Sample Time Control - Formula</td>
<td></td>
<td>188</td>
</tr>
<tr>
<td>ADC_TC.H024</td>
<td>Documentation: Filter control only in registers GxRCR7/GxRCR15</td>
<td>New</td>
<td>189</td>
</tr>
<tr>
<td>ASCLIN_TC.H001</td>
<td>Bit field FRAMECON.IDLE in LIN slave mode</td>
<td></td>
<td>190</td>
</tr>
<tr>
<td>ASCLIN_TC.H003</td>
<td>Behavior of LIN Autobaud Detection Error Flag</td>
<td></td>
<td>190</td>
</tr>
<tr>
<td>ASCLIN_TC.H004</td>
<td>Changing the Transmit FIFO Inlet Width / Receive FIFO Outlet Width</td>
<td></td>
<td>190</td>
</tr>
<tr>
<td>ASCLIN_TC.H005</td>
<td>Collision detection error reported twice in LIN slave mode</td>
<td></td>
<td>192</td>
</tr>
<tr>
<td>BCU_TC.H001</td>
<td>HSM Transaction Information not captured</td>
<td></td>
<td>192</td>
</tr>
<tr>
<td>BROM_TC.H003</td>
<td>Information related to Register FLASH0_PROCOND</td>
<td></td>
<td>193</td>
</tr>
<tr>
<td>BSL_TC.H002</td>
<td>Bootstrap Loaders need initialized Memory</td>
<td></td>
<td>193</td>
</tr>
<tr>
<td>CCU6_Ai.H001</td>
<td>Update of Register MCMOUT</td>
<td></td>
<td>193</td>
</tr>
<tr>
<td>CCU6_Ai.H002</td>
<td>Description of Bit RWHE in Register ISR</td>
<td></td>
<td>194</td>
</tr>
<tr>
<td>Hint</td>
<td>Short Description</td>
<td>Change</td>
<td>Page</td>
</tr>
<tr>
<td>--------------------</td>
<td>-----------------------------------------------------------------------------------</td>
<td>--------</td>
<td>------</td>
</tr>
<tr>
<td>CCU6_AL.H003</td>
<td>Bit TRPCTR.TRPM2 in Manual Mode - Documentation Update</td>
<td></td>
<td>194</td>
</tr>
<tr>
<td>CCU_TC.H001</td>
<td>Clock Monitor Check Limit Values</td>
<td></td>
<td>195</td>
</tr>
<tr>
<td>CCU_TC.H002</td>
<td>Oscillator Gain Selection via OSCCON.GAINSEL</td>
<td></td>
<td>196</td>
</tr>
<tr>
<td>CCU_TC.H005</td>
<td>References to $f_{\text{PLL2}}$, $f_{\text{PLL2_ERAY}}$ and K3 Divider in User’s Manual</td>
<td></td>
<td>196</td>
</tr>
<tr>
<td>CCU_TC.H006</td>
<td>Clock Monitor Support - Documentation Update</td>
<td></td>
<td>197</td>
</tr>
<tr>
<td>CCU_TC.H007</td>
<td>Oscillator Watchdog Trigger Conditions for ALM3[0]</td>
<td></td>
<td>197</td>
</tr>
<tr>
<td>CPU_TC.H006</td>
<td>Store Buffering in TC1.6/P/E Processors</td>
<td></td>
<td>198</td>
</tr>
<tr>
<td>CPU_TC.H008</td>
<td>Instruction Memory Range Limitations</td>
<td></td>
<td>200</td>
</tr>
<tr>
<td>CPU_TC.H009</td>
<td>Details on CPU Clock Control</td>
<td></td>
<td>201</td>
</tr>
<tr>
<td>CPU_TC.H010</td>
<td>External Accesses to CPU Local Memory may delay CPU Execution Progress</td>
<td></td>
<td>201</td>
</tr>
<tr>
<td>CPU_TC.H012</td>
<td>Behavior of bit-wise operations on certain peripheral register bits which need to be written back with the same value</td>
<td></td>
<td>202</td>
</tr>
<tr>
<td>CPU_TC.H014</td>
<td>ACCEN* Protection for Write Access to Safety Protection Registers - Documentation Update</td>
<td>New</td>
<td>204</td>
</tr>
<tr>
<td>CPU_TC.H015</td>
<td>Register Access Modes for Safety Protection Registers - Documentation Update</td>
<td>New</td>
<td>204</td>
</tr>
<tr>
<td>DAP_TC.H002</td>
<td>DAP client_blockread in Combination with TGIP and all Parcels with CRC6</td>
<td></td>
<td>204</td>
</tr>
<tr>
<td>DAP_TC.H003</td>
<td>Not acknowledged DAP telegrams in noisy environments</td>
<td></td>
<td>205</td>
</tr>
</tbody>
</table>
Table 6  Application Hints (cont’d)

<table>
<thead>
<tr>
<th>Hint</th>
<th>Short Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>DMA_TC.H002</td>
<td>Bit CHCSRz.BUFFER can be toggled when not in Double Buffer Mode</td>
<td>205</td>
</tr>
<tr>
<td>DMA_TC.H003</td>
<td>Spurious Error Interrupt Service Requests after Transaction Lost Event in Double Buffer Mode</td>
<td>206</td>
</tr>
<tr>
<td>DMA_TC.H004</td>
<td>Transaction Request Lost upon software trigger with pattern match</td>
<td>206</td>
</tr>
<tr>
<td>DMA_TC.H005</td>
<td>Linked List Transfer leading to loading of non-Linked List TCS causes corruption</td>
<td>207</td>
</tr>
<tr>
<td>DMA_TC.H006</td>
<td>Clearing of HTRE when DMA channel is configured for Single Mode</td>
<td>207</td>
</tr>
<tr>
<td>DMA_TC.H007</td>
<td>Selecting the Priority for DMA Channels</td>
<td>208</td>
</tr>
<tr>
<td>DMA_TC.H008</td>
<td>Transaction Request State</td>
<td>209</td>
</tr>
<tr>
<td>DMA_TC.H009</td>
<td>Resetting Bits ICH and IPM in register CHCSRz</td>
<td>210</td>
</tr>
<tr>
<td>DMA_TC.H010</td>
<td>Calculation of DMA Address Checksum for DMA read moves to Cacheable Addresses</td>
<td>210</td>
</tr>
<tr>
<td>DMA_TC.H011</td>
<td>DMA_ADICRz.SHCT - Reserved Values</td>
<td>211</td>
</tr>
<tr>
<td>DMA_TC.H012</td>
<td>TCS Update in Halt State</td>
<td>211</td>
</tr>
<tr>
<td>DSADC_TC.H002</td>
<td>Influence of Temperature on DC Offset Error EDOFF (calibrated)</td>
<td>211</td>
</tr>
<tr>
<td>DSADC_TC.H003</td>
<td>FIR Filters not reset when Integration starts</td>
<td>212</td>
</tr>
<tr>
<td>DSADC_TC.H004</td>
<td>Full-scale Values produced by On-chip Modulator</td>
<td>212</td>
</tr>
<tr>
<td>DSADC_TC.H005</td>
<td>Data Strobe Setting for On-chip Modulator</td>
<td>213</td>
</tr>
<tr>
<td>DSADC_TC.H006</td>
<td>Avoiding Intermediate States</td>
<td>214</td>
</tr>
<tr>
<td>DSADC_TC.H007</td>
<td>Dithering Control</td>
<td>215</td>
</tr>
</tbody>
</table>
### Table 6  Application Hints (cont’d)

<table>
<thead>
<tr>
<th>Hint</th>
<th>Short Description</th>
<th>Change</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>DSADC_TC.H008</td>
<td>DSADC Gain Calibration Procedure</td>
<td></td>
<td>215</td>
</tr>
<tr>
<td>DTS_TC.H001</td>
<td>Update of Bit DTSSTAT.BUSY</td>
<td></td>
<td>216</td>
</tr>
<tr>
<td>EBU_TC.H013</td>
<td>EBU_USERCON Upper Half-word - Documentation Update</td>
<td></td>
<td>216</td>
</tr>
<tr>
<td>EBU_TC.H014</td>
<td>Reset of EBU Registers</td>
<td></td>
<td>217</td>
</tr>
<tr>
<td>EBU_TC.H002</td>
<td>EMEM will raise ECC errors when not properly initialized</td>
<td></td>
<td>217</td>
</tr>
<tr>
<td>EBU_TC.H003</td>
<td>Handling of Register TILECONFIGXM</td>
<td></td>
<td>218</td>
</tr>
<tr>
<td>ENDINIT_TC.H001</td>
<td>Endinit Protection for Registers KRST0, KRST1, KRSTCLR</td>
<td></td>
<td>219</td>
</tr>
<tr>
<td>ETH_AI.H001</td>
<td>Sequence for Switching between MII and RMII Mode</td>
<td></td>
<td>219</td>
</tr>
<tr>
<td>ETH_TC.H001</td>
<td>ETHMDIO on P21.1 not to be used for productive systems</td>
<td></td>
<td>220</td>
</tr>
<tr>
<td>ETH_TC.H002</td>
<td>Minimum operation frequency for Ethernet MAC</td>
<td></td>
<td>220</td>
</tr>
<tr>
<td>ETH_TC.H003</td>
<td>Interrupt Generation by Wake-up or Magic Packet Frames</td>
<td></td>
<td>220</td>
</tr>
<tr>
<td>FLASH_TC.H007</td>
<td>Advice for using Suspend and Resume</td>
<td></td>
<td>221</td>
</tr>
<tr>
<td>FLASH_TC.H008</td>
<td>Understanding Flash Retention/Endurance Figures in the Data Sheet</td>
<td></td>
<td>222</td>
</tr>
<tr>
<td>FLASH_TC.H009</td>
<td>PFlash Operations Concurrent to DF1 Operations</td>
<td></td>
<td>223</td>
</tr>
<tr>
<td>FlexRay_AI.H004</td>
<td>Only the first message can be received in External Loop Back mode</td>
<td></td>
<td>224</td>
</tr>
<tr>
<td>FlexRay_AI.H005</td>
<td>Initialization of internal RAMs requires one eray_bclk cycle more</td>
<td></td>
<td>225</td>
</tr>
<tr>
<td>FlexRay_AI.H006</td>
<td>Transmission in ATM/Loopback mode</td>
<td></td>
<td>225</td>
</tr>
</tbody>
</table>
## Table 6 Application Hints (cont’d)

<table>
<thead>
<tr>
<th>Hint</th>
<th>Short Description</th>
<th>Change</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>FlexRay_Ai.H007</strong></td>
<td>Reporting of coding errors via TEST1.CERA/B</td>
<td>225</td>
<td></td>
</tr>
<tr>
<td><strong>FlexRay_Ai.H009</strong></td>
<td>Return from test mode operation</td>
<td>225</td>
<td></td>
</tr>
<tr>
<td><strong>FlexRay_TC.H002</strong></td>
<td>Initialization of E-Ray RAMs</td>
<td>226</td>
<td></td>
</tr>
<tr>
<td><strong>FPI_TC.H002</strong></td>
<td>Write Access to Register ACCEN1</td>
<td>228</td>
<td></td>
</tr>
<tr>
<td><strong>GPT12_TC.H001</strong></td>
<td>Timer T5 Run Bit T5R - Documentation Correction</td>
<td>228</td>
<td></td>
</tr>
<tr>
<td><strong>GTM_TC.H003</strong></td>
<td>Typo: GTM Chapters “Multi Channel Sequencer (MCS)” and “Memory Configuration (MCFG)” sometimes use “MSC” instead of “MCS”</td>
<td>229</td>
<td></td>
</tr>
<tr>
<td><strong>GTM_TC.H004</strong></td>
<td>Correction to Bit Fields GTM_TIMi_IN_SRC.VAL_x</td>
<td>229</td>
<td></td>
</tr>
<tr>
<td><strong>GTM_TC.H005</strong></td>
<td>External Capture in TIM Pulse Integration Mode (TPIM)</td>
<td>230</td>
<td></td>
</tr>
<tr>
<td><strong>GTM_TC.H006</strong></td>
<td>AEI to FIFO Data Interface (AFD1) Registers</td>
<td>231</td>
<td></td>
</tr>
<tr>
<td><strong>GTM_TC.H007</strong></td>
<td>GTM to CAN Timer Triggers</td>
<td>231</td>
<td></td>
</tr>
<tr>
<td><strong>GTM_TC.H008</strong></td>
<td>Correction to Figure “SPE to TOM Connections”</td>
<td>232</td>
<td></td>
</tr>
<tr>
<td><strong>GTM_TC.H011</strong></td>
<td>First CM0 updates in case of SR0=1 and (A)TOM used as Triggered Channel</td>
<td>232</td>
<td></td>
</tr>
<tr>
<td><strong>GTM_TC.H014</strong></td>
<td>Synchronous Bridge Mode Restrictions</td>
<td>232</td>
<td></td>
</tr>
<tr>
<td><strong>GTM_TC.H015</strong></td>
<td>Register TIMi_CHx_CTRL - Correction to Register Image</td>
<td>233</td>
<td></td>
</tr>
<tr>
<td><strong>GTM_TC.H016</strong></td>
<td>Evaluating DSADC Signals SAULx/SBLLx</td>
<td>234</td>
<td></td>
</tr>
<tr>
<td><strong>GTM_TC.H017</strong></td>
<td>Bit DXINCON.24 (DSS10) - Documentation Correction</td>
<td>New, 234</td>
<td></td>
</tr>
<tr>
<td><strong>GTM_TC.H018</strong></td>
<td>TIM4 Mapping for CH7SEL = 1010B - Correction</td>
<td>New, 235</td>
<td></td>
</tr>
</tbody>
</table>
### Table 6  Application Hints (cont’d)

<table>
<thead>
<tr>
<th>Hint</th>
<th>Short Description</th>
<th>Change</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>HSCT_TC.H003</td>
<td>Functionality of bit TX_PWDPD</td>
<td></td>
<td>235</td>
</tr>
<tr>
<td>HSCT_TC.H005</td>
<td>Access to reserved address 0xF009 0060 when $f_{SPB} = f_{SRI}$</td>
<td></td>
<td>235</td>
</tr>
<tr>
<td>HSCT_TC.H007</td>
<td>HSSL Integrated Phase Noise</td>
<td></td>
<td>236</td>
</tr>
<tr>
<td>HSCT_TC.H008</td>
<td>Details on PLL Lock-in Time</td>
<td></td>
<td>238</td>
</tr>
<tr>
<td>I2C_TC.H001</td>
<td>I2C Module Behavior in OCDS Suspend Mode</td>
<td></td>
<td>238</td>
</tr>
<tr>
<td>I2C_TC.H002</td>
<td>Initialization of INC/DEC values in Slave mode</td>
<td></td>
<td>239</td>
</tr>
<tr>
<td>I2C_TC.H003</td>
<td>DMA Channel Configuration</td>
<td></td>
<td>240</td>
</tr>
<tr>
<td>I2C_TC.H004</td>
<td>Transfers of more than 32 Bytes</td>
<td></td>
<td>241</td>
</tr>
<tr>
<td>I2C_TC.H005</td>
<td>FIFO Data is lost during Transaction RX-&gt;TX</td>
<td></td>
<td>241</td>
</tr>
<tr>
<td>I2C_TC.H008</td>
<td>Handling of RX FIFO Overflow in Slave Mode</td>
<td>New</td>
<td>242</td>
</tr>
<tr>
<td>IOM_TC.H001</td>
<td>How to clear the IOM_LAMEWm register</td>
<td></td>
<td>242</td>
</tr>
<tr>
<td>IOM_TC.H002</td>
<td>IOM Clock Control</td>
<td></td>
<td>243</td>
</tr>
<tr>
<td>IOM_TC.H003</td>
<td>Configuration of LAMCFG.IVW and LAMEWS.THR</td>
<td></td>
<td>244</td>
</tr>
<tr>
<td>IOM_TC.H004</td>
<td>Behavior of LAMEWm.CNT when LAMEWSn.THR is 0</td>
<td></td>
<td>246</td>
</tr>
<tr>
<td>IOM_TC.H006</td>
<td>ACCEN* Protection for Write Access to IOM Registers</td>
<td></td>
<td>246</td>
</tr>
<tr>
<td>IOM_TC.H007</td>
<td>Write Access to FPCESR</td>
<td></td>
<td>246</td>
</tr>
<tr>
<td>LMU_TC.H002</td>
<td>On-the-fly BBB:SRI clock ratio switching</td>
<td></td>
<td>247</td>
</tr>
<tr>
<td>LMU_TC.H003</td>
<td>Function of Bit MEMCON.PMIC (Protection Bit for Memory Integrity Control Bit)</td>
<td></td>
<td>248</td>
</tr>
<tr>
<td>LMU_TC.H004</td>
<td>FFT Accelerator Interface</td>
<td></td>
<td>248</td>
</tr>
</tbody>
</table>
## Table 6  Application Hints (cont’d)

<table>
<thead>
<tr>
<th>Hint</th>
<th>Short Description</th>
<th>Change</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>MSC_TC.H010</td>
<td>Configuration of SCU.EMSR for the EMGSTOPMSC Signal</td>
<td></td>
<td>250</td>
</tr>
<tr>
<td>MSC_TC.H011</td>
<td>Effect of kernel reset on MSC0_FCLP when selected in Event Trigger Logic</td>
<td></td>
<td>250</td>
</tr>
<tr>
<td>MSC_TC.H012</td>
<td>Handling the overflow interrupt of the ABRA block</td>
<td></td>
<td>251</td>
</tr>
<tr>
<td>MSC_TC.H013</td>
<td>Empty Data Frames not supported with ABRA</td>
<td></td>
<td>251</td>
</tr>
<tr>
<td>MTU_TC.H003</td>
<td>AURIX™ Memory Tests using the MTU</td>
<td></td>
<td>252</td>
</tr>
<tr>
<td>MTU_TC.H004</td>
<td>Handling the Error Tracking Registers ETRR</td>
<td></td>
<td>252</td>
</tr>
<tr>
<td>MTU_TC.H005</td>
<td>Handling SRAM Alarms</td>
<td></td>
<td>253</td>
</tr>
<tr>
<td>MTU_TC.H006</td>
<td>Alarm Propagation to SMU via Error Flags in MCx_ECCD</td>
<td></td>
<td>255</td>
</tr>
<tr>
<td>MTU_TC.H007</td>
<td>Reset Values of Bit ECCS.TRE</td>
<td></td>
<td>255</td>
</tr>
<tr>
<td>MTU_TC.H008</td>
<td>Memory Controllers for DSPR</td>
<td></td>
<td>256</td>
</tr>
<tr>
<td>MTU_TC.H009</td>
<td>Reset Value for Register ECCD</td>
<td></td>
<td>257</td>
</tr>
<tr>
<td>MTU_TC.H010</td>
<td>Register MCONTROL - Bit Field Res4</td>
<td></td>
<td>258</td>
</tr>
<tr>
<td>MTU_TC.H011</td>
<td>Access Protection for Memory Control Registers</td>
<td></td>
<td>258</td>
</tr>
<tr>
<td>MTU_TC.H012</td>
<td>Kernel Reset triggers Reset of MBIST Registers</td>
<td></td>
<td>New</td>
</tr>
<tr>
<td>MTU_TC.H014</td>
<td>Access to SRAM while MTU operations are underway</td>
<td></td>
<td>New</td>
</tr>
<tr>
<td>MultiCAN_Ai.H005</td>
<td>TxD Pulse upon short disable request</td>
<td></td>
<td>260</td>
</tr>
<tr>
<td>MultiCAN_Ai.H006</td>
<td>Time stamp influenced by resynchronization</td>
<td></td>
<td>260</td>
</tr>
<tr>
<td>MultiCAN_Ai.H007</td>
<td>Alert Interrupt Behavior in case of Bus-Off</td>
<td></td>
<td>261</td>
</tr>
<tr>
<td>Hint</td>
<td>Short Description</td>
<td>Change</td>
<td>Page</td>
</tr>
<tr>
<td>--------------------------</td>
<td>-------------------------------------------------------------</td>
<td>--------</td>
<td>------</td>
</tr>
<tr>
<td>MultiCAN_TC.H003</td>
<td>Message may be discarded before transmission in STT mode</td>
<td></td>
<td>261</td>
</tr>
<tr>
<td>MultiCAN_TC.H004</td>
<td>Double remote request</td>
<td></td>
<td>262</td>
</tr>
<tr>
<td>MultiCAN_TC.H007</td>
<td>Oscillating CAN Bus may Disable the CAN Interface</td>
<td></td>
<td>263</td>
</tr>
<tr>
<td>OCDS_TC.H011</td>
<td>Application Reset during CIF Transactions while OCDS is enabled</td>
<td></td>
<td>263</td>
</tr>
<tr>
<td>OCDS_TC.H012</td>
<td>Minimum Hold Time for Inputs OCDS_TGlx</td>
<td></td>
<td>263</td>
</tr>
<tr>
<td>PADS_TC.H001</td>
<td>Hysteresis Inactive Function</td>
<td></td>
<td>264</td>
</tr>
<tr>
<td>PADS_TC.H002</td>
<td>Write Access to Register PMSWCR0 when HWCFG[6] = 0</td>
<td></td>
<td>264</td>
</tr>
<tr>
<td>PADS_TC.H003</td>
<td>Terminology Mapping: Emulation/Production Device</td>
<td></td>
<td>265</td>
</tr>
<tr>
<td>PLL_ERAY_TC.H002</td>
<td>Correction in Figure “PLL_ERAY Block Diagram”</td>
<td></td>
<td>267</td>
</tr>
<tr>
<td>PMC_TC.H001</td>
<td>Check for permanent Overvoltage during Power-up</td>
<td></td>
<td>267</td>
</tr>
<tr>
<td>PMC_TC.H002</td>
<td>Description of Register PMSWSTAT</td>
<td></td>
<td>267</td>
</tr>
<tr>
<td>PORTS_TC.H006</td>
<td>Using P33.8 while SMU is disabled</td>
<td></td>
<td>268</td>
</tr>
<tr>
<td>PORTS_TC.H008</td>
<td>Emergency Stop for LVDS TX Pads in LVDS Mode</td>
<td></td>
<td>269</td>
</tr>
<tr>
<td>PORTS_TC.H009</td>
<td>Port Input/Output Control Register listed for P01.12</td>
<td></td>
<td>269</td>
</tr>
<tr>
<td>QSPI_TC.H005</td>
<td>Stopping Transmission in Continuous Mode</td>
<td></td>
<td>269</td>
</tr>
<tr>
<td>QSPI_TC.H006</td>
<td>Corrections to Figures “QSPI - Frequency Domains” and “Phase Duration Control, Overview”</td>
<td></td>
<td>270</td>
</tr>
</tbody>
</table>
## Table 6 Application Hints (cont’d)

<table>
<thead>
<tr>
<th>Hint</th>
<th>Short Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>QSPI_TC.H007</td>
<td>RXFIFO Overflow Bit Behavior in Slave Mode</td>
<td>271</td>
</tr>
<tr>
<td>RESET_TC.H002</td>
<td>Unexpected SMU Reset Indication in SCU_RSTSTAT</td>
<td>271</td>
</tr>
<tr>
<td>RESET_TC.H003</td>
<td>Usage of the Prolongation Feature for ESR0 as Reset Indicator Output</td>
<td>272</td>
</tr>
<tr>
<td>RESET_TC.H004</td>
<td>Effect of Power-on and System Reset on DSPR</td>
<td>273</td>
</tr>
<tr>
<td>SCU_TC.H009</td>
<td>LBIST Influence on Pad Behavior</td>
<td>273</td>
</tr>
<tr>
<td>SCU_TC.H013</td>
<td>Correction to Register References in Chapter “Watchdog Timers”</td>
<td>274</td>
</tr>
<tr>
<td>SCU_TC.H014</td>
<td>Reset Value of Bit Field IOCR.PC1 - Control for Pin ESR1</td>
<td>274</td>
</tr>
<tr>
<td>SENT_TC.H002</td>
<td>SENT Nibble Tolerance</td>
<td>275</td>
</tr>
<tr>
<td>SENT_TC.H003</td>
<td>First Write Access to Registers FDR and TPD after ENDINIT Status Change</td>
<td>276</td>
</tr>
<tr>
<td>SENT_TC.H004</td>
<td>Short Serial Message - Figure Correction</td>
<td>New</td>
</tr>
<tr>
<td>SENT_TC.H005</td>
<td>Interface Connections of the SENT Module - Documentation Correction</td>
<td>New</td>
</tr>
<tr>
<td>SMU_TC.H001</td>
<td>Write all bit fields of SMU_PCTL with one write access</td>
<td>279</td>
</tr>
<tr>
<td>SMU_TC.H003</td>
<td>Alarm Table Corrections</td>
<td>279</td>
</tr>
<tr>
<td>SMU_TC.H004</td>
<td>Alarm Mapping related to ALM3[9] in ALM3 Group</td>
<td>286</td>
</tr>
<tr>
<td>SMU_TC.H005</td>
<td>Correction to Figure “SMU Register Map”</td>
<td>287</td>
</tr>
<tr>
<td>SMU_TC.H006</td>
<td>Description of Bit EFRST in Register SMU_AGC</td>
<td>287</td>
</tr>
<tr>
<td>SMU_TC.H007</td>
<td>SPB Bus Control Unit (SBCU) Alarm Signalling to SMU</td>
<td>288</td>
</tr>
<tr>
<td>Hint</td>
<td>Short Description</td>
<td>Page</td>
</tr>
<tr>
<td>--------------</td>
<td>-----------------------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>SMU_TC.H010</td>
<td>Clearing individual SMU flags: use only 32-bit writes</td>
<td>288</td>
</tr>
<tr>
<td>SRI_TC.H001</td>
<td>Using LDMST and SWAPMSK.W instructions on SRI mapped Peripheral Registers</td>
<td>289</td>
</tr>
<tr>
<td></td>
<td>(range 0xF800 0000-0xFFFF FFFF)</td>
<td></td>
</tr>
<tr>
<td>STM_TC.H001</td>
<td>Effect of kernel reset on interrupt outputs</td>
<td>290</td>
</tr>
<tr>
<td></td>
<td>STMIR0/1</td>
<td></td>
</tr>
<tr>
<td>STM_TC.H002</td>
<td>Access Protection for STM Control Registers</td>
<td>290</td>
</tr>
</tbody>
</table>
2 Functional Deviations

ADC_AI.016 No Channel Interrupt in Fast Compare Mode with GLOBRES

In fast compare mode, the compare value is taken from bitfield RESULT of the selected result register and the result of the comparison is stored in the respective bit FCR.

A channel event can be generated when the input becomes higher or lower than the compare value.

In case the global result register GLOBRES is selected, the comparison is executed correctly, the target bit is stored correctly, source events and result events are generated, but a channel event is not generated.

Workaround

If channel events are required, choose a local result register GxRESy for the operation of the fast compare channel.

ADC_TC.068 Effect of VAGND Cross Coupling on Conversion Result

Due the implementation of the clock dividers as fractional dividers, a statistical phase shift of one \( f_{VADC} \) clock can occur between the operation of different converter groups. If the last \( f_{VADC} \) clock of the sample phase of a converter group Gx coincides with the first \( f_{VADC} \) clock of a conversion step of (one or more) other converter groups Gy, the Total Unadjusted Error (TUE) of the conversion result of Gx is increased due to cross coupling via VAGND.

For TC27x and TC29x, the TUE is increased up to \( \pm 80 \text{ LSB}_{12} \)

Workarounds - Introduction

Workaround 1..3 may be used with any device step.
Workaround 4 can only be used with TC29x ≥ step BB. This is the preferred workaround for this device, since for Workaround 1 in particular increased error limits would have to be accepted for high temperature / high pin-count variants.

**Workaround 1**

Synchronize the trigger events of different converter groups as follows:

- Operate the arbiters and the analog parts of the VADC at the same clock frequency, i.e. select the divider factors DIVA and DIVD in register GLOBCFG such that \( f_{ADC_D} = f_{ADC_I} \) for all converter groups:
  - For \( f_{VADC} = f_{SPB} = 100 \text{ MHz} \), reduce \( f_{ADC_D} \) to 25 MHz (with \( \text{DIVD} = 3 \)), and increase \( f_{ADC_I} \) to 25 MHz (with \( \text{DIVA} = 3 \)). Depending on supply/reference voltage and junction temperature \( T_J \), this will result in the increased error limits shown in **Table 7** (for TC297 devices in BGA-292 package) and **Table 8** (for TC298/9 devices in BGA-416/516 package and TC290 bare die variants).
  - Otherwise, to achieve \( f_{ADC_D} = f_{ADC_I} = 20 \text{ MHz} \) with the error limits specified in the Data Sheet, \( f_{VADC} = f_{SPB} \) must be reduced to 80 MHz.
- Enlarge the length of an arbitration round to a minimum of 16 arbitration slots (i.e. bit field GxARBCFG.ARBRND ≥ 2 for any x).
- Select the conversion time (including sample time) of the longest conversion of any group Gx to be shorter than two arbitration rounds. This ensures that all converters are idle when the arbiters have determined the next conversion request.
- Synchronize the digital and the analog clock by switching off/on the Module Disable Request bit, i.e. set CLC.DISR = 1B and then CLC.DISR = 0B.
- Initiate the start-up calibration by setting bit GLOBCFG.SUCAL = 1B (mandatory after switching off/on VADC clocks via CLC.DISR).

---

**Table 7**  **TC297 in BGA-292 Package: Increased VADC Error Limits for**

<table>
<thead>
<tr>
<th>Condition</th>
<th>VAREF/VDDM</th>
<th>Total Unadjusted Error TUE</th>
<th>DNL Error EA(_{DNL})</th>
<th>INL Error EA(_{INL})</th>
<th>RMS Noise EN(_{RMS})</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>5V ± 10% ≤ 150°C</td>
<td>5V ± 10%</td>
<td>4.4</td>
<td>3.5</td>
<td>3.5</td>
<td>0.95</td>
<td>LSB</td>
</tr>
<tr>
<td>&gt; 150°C</td>
<td></td>
<td>5.7</td>
<td>3.5</td>
<td>3.5</td>
<td>0.95</td>
<td></td>
</tr>
</tbody>
</table>
Note: For error types (offset, gain, ..) not listed in the table above see corresponding values specified in the Data Sheet for $f_{\text{ADCI}} = 20 \text{ MHz}$.

Note: For available combinations of package type and temperature range see the corresponding AURIX™ TC2x Variants Data Sheet Addendum.

Workaround 2

Ensure that conversions never overlap for any two converter groups $G_x$ and $G_y$. 

---

### Table 7

TC297 in BGA-292 Package: Increased VADC Error Limits for $f_{\text{ADCI}} = 25 \text{ MHz}$, 12-Bit Resolution

<table>
<thead>
<tr>
<th>Total Unadjusted Error TUE</th>
<th>DNL Error $E_{\text{DNL}}$</th>
<th>INL Error $E_{\text{INL}}$</th>
<th>RMS Noise $E_{\text{RMS}}$</th>
<th>Unit</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>12</td>
<td>12</td>
<td>1.5</td>
<td>LSB</td>
<td>3.3 V ± 10 %</td>
</tr>
<tr>
<td>14</td>
<td>14</td>
<td>16</td>
<td>1.5</td>
<td></td>
<td>125°C &lt; $T_J$</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>&gt; 150°C</td>
</tr>
</tbody>
</table>

This workaround is not recommended for $T_J > 150^\circ \text{C}$ and $V_{\text{AREF/VDDM}} = 3.3 \text{ V} ± 10 \%$

### Table 8

TC298/9 in BGA-416/516 Package and TC290 bare die: Increased VADC Error Limits for $f_{\text{ADCI}} = 25 \text{ MHz}$, 12-Bit Resolution

<table>
<thead>
<tr>
<th>Total Unadjusted Error TUE</th>
<th>DNL Error $E_{\text{DNL}}$</th>
<th>INL Error $E_{\text{INL}}$</th>
<th>RMS Noise $E_{\text{RMS}}$</th>
<th>Unit</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.4</td>
<td>6.0</td>
<td>5.4</td>
<td>0.95</td>
<td>LSB</td>
<td>5V ± 10 %</td>
</tr>
<tr>
<td>8.4</td>
<td>7.9</td>
<td>7.1</td>
<td>0.95</td>
<td></td>
<td>&gt; 150°C</td>
</tr>
<tr>
<td>10</td>
<td>12</td>
<td>12</td>
<td>1.5</td>
<td>LSB</td>
<td>3.3 V ± 10 %</td>
</tr>
<tr>
<td>14</td>
<td>14</td>
<td>16</td>
<td>1.5</td>
<td></td>
<td>125°C &lt; $T_J$</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>&gt; 150°C</td>
</tr>
</tbody>
</table>

This workaround is not recommended for $T_J > 150^\circ \text{C}$ and $V_{\text{AREF/VDDM}} = 3.3 \text{ V} ± 10 \%$
This may be achieved under software control, or by exclusively using the VADC background request source.

For this workaround, no restrictions apply on clock and arbitration round settings.

**Workaround 3**

Use the converters within a synchronization group in master/slave configuration, such that they are synchronized for parallel sampling, triggered by one common master. In this case, the cross coupling effect will not occur as long as only one synchronization group is performing conversions.

For devices that support more than one synchronization group, operate the synchronization groups in an interleaving manner.

For this workaround, no restrictions apply on clock and arbitration round settings.

**Workaround 4**

To avoid the cross coupling effect, this device step (see “Workarounds - Introduction” above) supports selection of signal CCU6061_TRIG1 to synchronize the start of the converter groups to a raster of $5/f_{SPB}$ (i.e. 50 ns @$f_{SPB} = 100$ MHz). The resulting jitter (delay from trigger to start of conversion) is thus limited to max. $5/f_{SPB}$.

For this workaround, either CCU60_T13 or CCU61_T13 is configured (reserved) to provide the synchronization signal. The selection is performed via bit field TRIG1SEL in register CCU60_MOSEL:
- TRIG1SEL = 000B: signal CCU60_COUT63 from CCU60_T13 is selected
- TRIG1SEL = 001B: signal CCU61_COUT63 from CCU61_T13 is selected

The synchronization signal is enabled inside the VADC module by setting bit GLOBCFG.DCMSB = 1B. The default function of this bit (DCMSB = 0B: one clock cycle for MSB conversion step) is hardwired and thus stays unaffected.

The following examples describe the initialization of CCU60 or CCU61, respectively, to provide a 20 MHz synchronization signal @$f_{SPB} = 100$ MHz:

**Example for CCU60 initialization**

```c
CCU60_CLC = 0x0;       // enable CCU60 kernel
```
CCU60_T13PR = 0x4; // 4+1 clock periods with ..
CCU60_CC63SR = 0x1; // duty cycle 40 ns low / 10 ns high
CCU60_PSLR |= 0x0080; // passive state level of COUT63 = 1
CCU60_MODCTR |= 0x8000; // ECT13O = 1 enables T13 output
    // (CC63ST -> COUT63)
CCU60_TCTR4 |= 0x4200; // set bit T13STR and T13RS ..
    // to enable shadow transfer and start T13
CCU60_MOSEL &= 0x1C7; // CCU6061_TRIG1 is CCU60_COUT63

Example for CCU61 initialization

CCU61_CLC = 0x0; // enable CCU61 kernel

Note: In case an application only uses kernel CCU61, ensure that kernel CCU60 is also clocked until register CCU60_MOSEL is configured.

CCU60_CLC = 0x0; // ensure CCU60 kernel is clocked until CCU60_MOSEL is configured
CCU61_T13PR = 0x4; // 4+1 clock periods with ..
CCU61_CC63SR = 0x1; // duty cycle 40 ns low / 10 ns high
CCU61_PSLR |= 0x0080; // passive state level of COUT63 = 1
CCU61_MODCTR |= 0x8000; // ECT13O = 1 enables T13 output
    // (CC63ST -> COUT63)
CCU61_TCTR4 |= 0x4200; // set bit T13STR and T13RS ..
    // to enable shadow transfer and start T13
CCU60_MOSEL |= 0x8; // CCU6061_TRIG1 is CCU60_COUT63

**ASCLIN_TC.004 SLSO in SPI mode still active after module disable**

It is expected that in SPI mode, after module disable, the Slave Select Output signal SLSO should be in idle state according to configuration of Slave Polarity in Synchronous mode (IOCR.SPOL).

However, in this design step, when the module is disabled, the Slave Select Output signal SLSO is always 0 (low) independent of IOCR.SPOL, i.e., it is still active even when IOCR.SPOL = 1B.
Workaround

Before disabling the ASCLIN module, set SLSO to the desired level in the corresponding Port control registers.

**ASCLIN TC.005 Unjustified collision detection error in half-duplex SPI mode**

In Half Duplex SPI mode, when collision detection is enabled and the number of stop bits in SPI frame is configured as any value from 1 to 7 in FRAMECON.STOP, a Collision Error (FLAGS.CE) is triggered during the trailing phase (i.e., during stop bits), although RX and TX signal are identical.

Workaround

In half-duplex SPI mode, set FRAMECON.STOP = 0 if trailing phase is irrelevant, or ignore/disable collision error if FRAMECON.STOP > 0.

**ASCLIN TC.006 Unjustified response timeout in LIN slave mode**

When ASCLIN is configured as LIN slave and Response timeout is configured as DATCON.RM = 1_B, Response timeout is triggered even when an incomplete LIN Header frame is received. The timeout counter runs further after Header timeout detection without reset and triggers Response Timeout when it reaches the Response Timeout Threshold value defined by DATCON.RESPONSE.

Workaround

Ignore the Response Timeout which comes directly after a Header Timeout has occurred and before the next break is detected.

**ASCLIN TC.007 Break Detected in LIN Frames in Soft Suspend mode**

When ASCLIN has entered Soft Suspend mode (OCS.SUS = 0x2), it still detects a Break Field in LIN frames and triggers an interrupt if enabled (FLAGSENABLE.BDE = 1_B).
Workaround

Ignore a detected break event when the module has been soft-suspended (e.g. set FLAGSENABLE.BDE = 0B when using soft suspend mode).

**ASCLIN TC.008  Response timeout in LIN Mode in case of header only**

In LIN (Master/Slave) mode, when Header Only (DATCON.HO = 1B) is configured, Response timeout could occur even though no Response frame is expected.

Workaround

To avoid the unwanted interrupt, disable the interrupt on Response Timeout by FLAGSENABLE.RTE = 0B whenever Header Only (DATCON.HO = 1B) is configured.

**ASCLIN TC.009  RFL flag set in Buffer Mode when Receive FIFO Inlet is disabled**

When RXFIFO is configured in Buffer Mode (RXFIFOCON.BUF = 1B) and Receive FIFO Inlet is disabled (RXFIFOCON.ENI = 0B), the receive FIFO level flag is set (FLAGS.RFL = 1B) even though RXFIFO is not filled with new incoming data.

Workaround

To avoid the unwanted Receive FIFO Level interrupt, disable it by setting FLAGSENABLE.RFLE = 0B whenever Receive FIFO Inlet is disabled (RXFIFOCON.ENI = 0B),

**ASCLIN TC.010  Flush of TXFIFO leads to frame transmission**

When the TXFIFO is flushed (TXFIFOCON.FLUSH = 1B), it triggers transmission of a frame in the following corner case:

- Starting condition:
- TXFIFO is not empty and TXFIFOCON.ENO = 0 B

- Triggering condition:
  - Write to TXFIFOCON with both TXFIFOCON.FLUSH = 1 B and TXFIFOCON.ENO = 1 B

**Workaround**

Do not flush TXFIFO and change bit TXFIFOCON.ENO from 0 B to 1 B in one single write to TXFIFOCON if TXFIFO is not empty.

**BROM TC.008  Sporadic Power-on Reset after Wake-up from Standby Mode**

On a wake-up from Standby mode, the Standby RAM redundancy installation procedure is executed. In case there is a sporadic Power-on reset in a time window between 600 µs - 1 ms after Standby mode wake-up, it can happen that the application data stored in specific Standby RAM cells are overwritten.

*Note: This effect can occur only on devices where non-zero data are stored in CPU0 DSPR at locations D000 2000 H to D000 203F H by the Startup Software (SSW) after cold power-on (see section “Preparation before to enter Stand-by mode” in the BootROM chapter of the User’s Manual). Only CPU0 DSPR Standby RAM is affected, EMEM in ADAS or ED devices is not affected.*

**Workarounds**

1. Calculate CRC over critical Standby RAM data and store result before Standby mode entry. On a consequent wake-up, CRC of the critical data shall be carried out. The CRC is a general recommended measure for improved robustness of Standby RAM handling.
   
   Or / and

2. Keep a copy of the critical data at a second location in Standby RAM. On wake-up, compare data from both locations to ascertain their integrity.
CCU_TC.002 Clock Monitors - Target Monitoring Frequency Selection

The following two configuration options for the Target Monitoring Frequency Selection in bit fields xxxSEL of register CCUCON3 and CCUCON4 must not be used:

- xxxSEL = 11_B: 7.5 MHz is selected as target monitoring frequency
- xxxSEL = 10_B: 6.6 MHz is selected as target monitoring frequency

Otherwise, in rare cases, unexpected SMU alarms may occur sporadically or continuously for the respective clock monitors.

Workaround

To avoid this effect, use only the following configuration options for the Target Monitoring Frequency Selection in bit fields xxxSEL of register CCUCON3 and CCUCON4:

- xxxSEL = 01_B: 6 MHz is selected as target monitoring frequency
- xxxSEL = 00_B: 5 MHz is selected as target monitoring frequency

CIF_TC.013 DMA Access to Reserved/Protected Resources: FPI Error Response not correctly evaluated

The CIF module includes a configurable DMA function to support CIF data transfers from/to EMEM. If the CIF DMA is accessing reserved system address ranges or protected resources (e.g., protected via system MPU/ACCEN register), the CIF DMA transactions via the Back Bone Bus (BBB) will be finished on the on-chip bus with an Error Acknowledge.

Depending on the target address, the first transaction with an Error Acknowledge will be captured by the BCU_BBB (BBB Bus Control Unit). An interrupt can be generated (SRC_EMEM in the Interrupt Router IR) if the related SRN is enabled.

However, the CIF DMA will not be stopped by an Error Acknowledge. It will ignore the Error Acknowledge. (i.e. bits BUS_ERROR in register CIFMI_RIS/CIFMI_MIS are not set).

In this situation the CIF data transferred by the DMA to an invalid internal address will be lost, and CIF will go on with invalid read data.
CIF_TC.014  CIF Module Sub-Resets

*Note:* This problem only affects the ADAS variants and corresponding emulation devices of TC29x and TC26x.

The CIF sub-module resets that can be triggered by writing to the IRCL register may cause the following potential problem:

**Description**

There is a risk that flip-flops in other not-reset sub-modules that receive data from the reset sub-module may be set to a random value.

**Workaround**

Do not use the reset bits of the CIF_IRCL register.
Instead, use the module reset provided by CIFBBB_KRST0 and KRST1 registers.

CIF_TC.015  Security Watchdog Interrupt Control - Documentation Update

*Note:* This erratum might affect the SFR C Header Definitions. In such cases, SFR usage in the software shall be analyzed within the applications for their correct handling.

The actual implementation (function mapping) for bits [3:0] in the following Security Watchdog Interrupt status/control registers CIFWD_*

- CIFWD_IMSC Watchdog Interrupt Mask Register
- CIFWD_RIS Watchdog Raw Interrupt Status Register
- CIFWD_MIS Watchdog Masked Interrupt Status Register
- CIFWD_ICR Watchdog Interrupt Clear Register
- CIFWD_ISR Watchdog Interrupt Set Register

differs from the documentation of the function of bits [3:0] as follows:
Errata Sheet

Functional Deviations

Table 9 Security Watchdog Interrupt Status/Control Registers - Documentation Update

<table>
<thead>
<tr>
<th>Event</th>
<th>Actual Implementation</th>
<th>Documentation (incorrect)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Horizontal Start End</td>
<td>*_WD_HSE_TO</td>
<td>3</td>
</tr>
<tr>
<td>Horizontal End Start</td>
<td>*_WD_HES_TO</td>
<td>2</td>
</tr>
<tr>
<td>Vertical Start End End</td>
<td>*_WD_VSE_TO</td>
<td>1</td>
</tr>
<tr>
<td>Vertical End Start End</td>
<td>*_WD_VES_TO</td>
<td>0</td>
</tr>
</tbody>
</table>

Note: * = IMSC for register CIFWD_IMSC
* = RIS for register CIFWD_RIS
* = MIS for register CIFWD_MIS
* = ICR for register CIFWD_ICR
* = ISR for register CIFWD_ISR

CPU_TC.123 Data Corruption possible when CPU GPR accesses made via SRI slave with CPU running

Data corruption may occur when another master accesses a TriCore CPU’s General Purpose Registers (GPRs) via its SRI slave port whilst the CPU is running (i.e. not Idle, Halted or Suspended). The TriCore GPRs are A0-A15 and D0-D15. The scenarios in which data corruption may occur are different for the TC1.6P and TC1.6E processors as described below.

TC1.6P - Data corruption may occur when one of the CPU GPRs is written via the SRI slave port whilst the CPU is running. Both AGPR and DGPR writes may be affected.

TC1.6E - Data corruption may occur when one of the CPU Address GPRs (A0-A15) is read via the SRI slave port whilst the CPU is running. However, data
corruption can only occur when the slave AGPR read interacts with the execution of a specific form of store instruction. The store instructions affected by this issue are ST.A and ST.DA, where the address register to be stored is modified by the addressing mode of the store instruction. For example:

\[ \text{ST.A \ [+A0], A0} \]

However, such store instructions are architecturally undefined and should not be being used. In the case of this errata all data written to memory by this store instruction may be corrupted.

**Workaround**

Writes to a CPU’s GPRs via its SRI slave port must never be performed whilst the CPU is running. If it is necessary for an external master to write to a CPU’s GPR then that CPU must first be placed in Idle, Halt or Suspend mode.

If it is necessary for an external master to read a TC1.6E CPU’s AGPR whilst that CPU is running then store instructions of the form above (where any source register is modified by the addressing mode of the store instruction) are not allowed.

---

**CPU TC.126 Unexpected Address Error Alarms caused by Speculative Access to Out-of-range PMEM and DMEM Areas**

**Overview**

An interaction between the TC1.6P TriCore processor speculation system and the SRAM address error system can lead to unexpected and unintended address error alarms being generated. These unexpected address error alarms are generated by speculative TriCore accesses into out-of-range memory areas of non-power of two local PMEM (PSPR + PCache) or DMEM (DSPR + DCache) memories.

In TC29x, the following CPUs and memories are affected:

- PMEM side issue: CPU0 PMEM
- DMEM side issue: CPU1 and CPU2 DMEM.
Figure 1  Example for Non-Power of Two PMEM

Note: The issue only affects non-power of two PMEM and DMEM memories of a TC1.6P CPU as described above. Other CPU cores and memory configurations are not affected.

Note: At no time does TriCore speculation lead to incorrect code being executed. The only erroneous effect seen is the generation of spurious address error alarms.

Effect due to the Issue

Due to the issue, the corresponding PSPR and DSPR address error alarms are generated. The ETRR error address buffer in the corresponding MBIST will contain addresses which are out of range of the implemented SRAM size.

MPU related preconditions for the Issue

Both PMEM and DMEM side issues will not occur during normal functional execution (i.e. without PSE traps) if the MPU of the corresponding CPU is completely disabled (CPUx_SYSCON.PROTEN = 0B).

If MPU is enabled, both the PMEM and DMEM issues will still not occur during normal functional execution if all used Protection Register Sets (PRS) in the MPU have un-restricted execution permissions for the entire executable
address space (segments 5 to $E_{H}$). The settings of PRS sets not used at all within the application have no effect.

In addition, the DMEM side issue can occur only when an MPX or PSE trap is generated (This does not apply to PMEM side issue, which can be caused during normal code execution).

**TC29x: Scenarios leading to Speculative Access to Out-of-range PMEM Areas**

Address error alarms ALM0[8] (and - should the situation arise - associated address buffer overflow alarms ALM0[9]) can be generated by speculative fetch accesses from CPU0 to the following out-of-range PMEM locations:

- Segment C: $C^{**} C000_{H} \ldots C^{**} FFFF_{H}$
- Segment 7: $701^{*} C000_{H} \ldots 701^{*} FFFF_{H}$

The PMEM side issue depends on specific code patterns looking similar to an out of range fetch instruction, if accessed from an unaligned address. This issue may be potentially triggered during any return instruction (ret / rfe etc.), if additionally an external access to the DMEM occurs during the return instruction.

**TC29x: Scenarios leading to Speculative Access to Out-of-range DMEM Areas**

Address error alarms (and - should the situation arise - associated address buffer overflow alarms) can be generated by speculative accesses from CPU1 and CPU2 to the following out-of-range DMEM locations:

- Segment D: $D^{**3} E000_{H} \ldots D^{**3} FFFF_{H}$ (CPU1 and CPU2),
- Segment 5: $5003 E000_{H} \ldots 5003 FFFF_{H}$ (CPU2),
- Segment 6: $6003 E000_{H} \ldots 6003 FFFF_{H}$ (CPU1).

The unexpected alarms generated are:

- for CPU2: ALM6[12] (address error alarm) and ALM6[13] (address buffer overflow alarm)
- for CPU1: ALM1[12] (address error alarm) and ALM1[13] (address buffer overflow alarm)

The DMEM side issue can occur only when a fetch trap (MPX or PSE trap) is being executed.
If these traps are being used in the application to only denote an error condition, it just needs to be taken care that an additional alarm may potentially occur when the traps are executed.

**Workaround for PMEM Side**

In case the MPU related preconditions for the Issue are relevant, then in order to avoid the PMEM side alarms, the workaround is to disable address error notification of the affected PMEMs to the SMU completely in the MTU itself. Address buffer overflow error notification / alarm is not disabled.

In the TC29x, the following error notifications shall be disabled within the MBIST:
- CPU0 PMEM address error notification via bit ECCD.AENE = 0 in MC16

*Note: Due to this, neither alarms nor CPU traps are generated for the above mentioned address errors.*

The concept of handling correctable, un-correctable and address buffer overflow errors / alarms is not affected and needs not be changed. Correctable error counting threshold is still effective, as address buffer overflow alarm is not disabled.

**Workaround for DMEM Side**

The DMEM side issue can only be triggered when an MPX or PSE trap is executed. Therefore, if these traps denote only an error condition, then during the traps, an additional DMEM alarm needs to be considered and taken care of.

In case these traps are used for functional purposes and triggered on purpose - for example by jumping to an “entry point” without execution permission in order to trigger an MPX trap - then the following workaround can be applied:
- The “entry point” has to be placed in a cacheable 32-byte aligned location, with the PCACHE enabled.
- The “entry point” needs to begin with a sequence on 4 NOP16 instructions.
- There shall be no external access to PSPR during the jump to the “entry point”.
- In case it cannot be ensured that external accesses to PSPR shall not occur, then the sequence shall be preceded by a DSYNC, ISYNC sequence with interrupts disabled.
Example Code Sequence

(Assuming PCACHE is enabled)
__asm(“disable”); //needed only if external PSPR accesses
__asm(“dsync”); //needed only if external PSPR accesses
__asm(“isync”); //needed only if external PSPR accesses
__asm(“ja entry_point”);

Where entry_point is a cacheable 32-byte aligned address, beginning with 4 NOP16 instructions.

In case the preconditions for the issue are relevant, and the above workaround cannot be applied, then in order to avoid the DMEM issue, the workaround is to disable address error notification of the affected DMEMs to the SMU completely in the MTU itself. Address buffer overflow error notification / alarm is not disabled.

In the TC29x, the following error notifications shall be disabled within the MBIST:
- CPU1 DMEM address error notification via bit ECCD.AENE = 0B in MC6 (DSPR) and MC20 (DSPR2)
- CPU2 DMEM address error notification via bit ECCD.AENE = 0B in MC0 (DSPR) and MC21 (DSPR2)

Note: Due to this, neither alarms nor CPU traps are generated for the above mentioned address errors.

The concept of handling correctable, un-correctable and address buffer overflow errors / alarms is not affected and needs not be changed. Correctable error counting threshold is still effective, as address buffer overflow alarm is not disabled.

Safety / FMEDA Considerations

The impact of the following needs to be considered / calculated for the above mentioned CPU PMEMs and DMEMs:
- Increase of residual failure rate due to disabling of address error monitor
• Coverage of certain address faults with ECC: certain address errors additionally produce ECC-incorrect data, which can be detected by the ECC.

**CPU_TC.127** Pending Interrupt Priority Number PIPN in Register ICR

In the TriCore Architecture Manual, it is described for the Pending Interrupt Priority Number ICR.PIPN that it is reset to 0x0 in case there is no request pending.

However, the AURIX™ hardware implementation behaves differently, as the value of PIPN is not changed after the interrupt is serviced in case there is no further request pending.

**DAP_TC.002** DAP client_blockread has Performance issue in Specific Operation Modes

For achieving the highest block read bandwidth, the following word is already read chip internally while a word is transmitted on DAP. This read ahead is under certain conditions disabled in the case that the “All parcels with CRC6” bit is set in the telegram. In this case the distance between the reply parcels becomes significantly longer, due to the missing read ahead. This effect occurs also in Wide Mode.

The data values in the parcels are always correct, it is just a performance issue.

**Workaround**

Don’t use the “All parcels with CRC6” option, use “Read CRCup” instead. This mode is anyway better in terms of performance for larger blocks (no CRC6 overhead for each parcel) and data protection (32 bit CRC). For a few words, the impact of this performance issue might be tolerable. For the first word a read ahead is not possible anyway.
**DAP_TC.003 DAP CRC32 definition and algorithm**

The DAP CRC32 algorithm is different from the IEEE 802.3 Ethernet CRC.

**Workaround**

Use the following (VHDL) algorithm for each incoming data bit. The CRC32 value is initialized with all ones.

In Wide Mode the function is called for both DAP data bits in each DAP0 clock cycle.

```vhdl
subtype crc32_t is std_ulogic_vector(31 downto 0);

function calc_crc32_f(crc_now : crc32_t;
                       bit_new : std_ulogic)
     return crc32_t is
     begin
       variable crc : crc32_t;
       begin
         crc(31 downto 1) := crc_now(30 downto 0);
         crc(0) := bit_new xor crc_now(31);
         crc(1) := bit_new xor crc_now(0) xor crc_now(31);
         crc(2) := bit_new xor crc_now(1) xor crc_now(31);
         crc(4) := bit_new xor crc_now(3) xor crc_now(31);
         crc(5) := bit_new xor crc_now(4) xor crc_now(31);
         crc(7) := bit_new xor crc_now(6) xor crc_now(31);
         crc(8) := bit_new xor crc_now(7) xor crc_now(31);
         crc(10) := bit_new xor crc_now(9) xor crc_now(31);
         crc(11) := bit_new xor crc_now(10) xor crc_now(31);
         crc(12) := bit_new xor crc_now(11) xor crc_now(31);
         crc(16) := bit_new xor crc_now(15) xor crc_now(31);
         crc(22) := bit_new xor crc_now(21) xor crc_now(31);
         crc(23) := bit_new xor crc_now(22) xor crc_now(31);
         crc(26) := bit_new xor crc_now(25) xor crc_now(31);
       return crc;
     end calc_crc32_f;
```
DAP_TC.004 DAP client_blockwrite telegram with CRC6 and CRC32 protection options

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

When issuing a DAP client_blockwrite telegram from the tool to the device several CRC protection options are available, namely CRC6 and CRC32.

Expected Behavior

- For CRC6 the expected behavior is:
  1. A CRC6 will be appended to the reply of only the last parcel of the telegram.
  2. An optional CRC6 can be appended to the device’s “single startbit response” by setting DAPISC.RC6.
- For CRC32 the expected behavior is:
  3. The telegram can optionally send the CRCdown value as the last parcel.

Actual Implementation

- For the actual implementation the CRC6 slightly differs as follows:
  1. The CRC6 of the last parcel will be erroneous if DAPISC.RC6 is set or if the CRCdown option is enabled.
  2. If DAPISC.RC6 = 1_B, an unintentional CRC6 will be appended to the device response of parcels which are not the last parcel.
- For the actual implementation the CRC32 option slightly differs as follows:
  3. If also the CRC6 option is set, the CRCdown option will not return the correct CRCdown value.

Workaround for (3)

Workaround for (3) is not to use the CRCdown feature of the client_blockwrite telegram, but to use the dedicated get_CRCdown telegram.
DAP_TC.005  DAP client_read: dirty bit feature of Cerberus’ Triggered Transfer Mode

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

The DAP telegram client_read reads a certain number of bits from an IOclient (e.g. Cerberus). The parameter k can be selected to be zero, which is supposed to activate reading of 32 bits plus dirty bit.

However, in the current implementation, the dirty bit feature does not work correctly.

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

DAP_TC.006  CRC6 error in telegram following a get_CRCdown telegram prevents reset of CRC32 calculator

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

If a CRC6 error occurs in the telegram following a get_CRCdown telegram the AURIX™ internal CRC32 calculator does not get reset, as is the expected behavior for get_CRCdown.

This effect can lead to unexpected CRC32 values for the next get_CRCdown telegram. This corresponds to the perception of the tool that there has been a CRC32 error, even if the data was transmitted correctly.

Workaround 1

Accept extra traffic for a required retransmission: In this case the tool could see a CRC32 error which is not based on a wrong transmission, but on the missing reset of the AURIX™ internal CRC32 calculator. This would trigger the retransmission of correctly sent data.
Workaround 2

Check for no-reply after a get_CRCdown telegram: If the tool does not receive an answer for the telegram following a get_CRCdown, it needs to resend the get_CRCdown telegram and ignore the data.

**DAP_TC.009** CRC6 error in client_blockwrite telegram

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

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

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

**Workaround**

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

**DMA_TC.015** DMA Double Buffering: No Timestamp Support

When a DMA channel is configured for DMA Double Buffering, and flow control (or appendage of time stamp) is selected, i.e. DMA_ADICRz.STAMP = 1B, the Move Engine may lock up.

**Workaround**

When a DMA channel is configured for DMA Double Buffering then flow control (or appendage of time stamp) should not be selected, i.e. bit DMA_ADICRz.STAMP must be = 0B.
DMA_TC.016  Byte and Half-word Write Accesses to specific Registers not supported

*Note:* This erratum might affect the SFR C Header Definitions. In such cases, SFR usage in the software shall be analyzed within the applications for their correct handling.

Byte and half-word write accesses via the SPB (System Peripheral Bus) to the Regfile and Request Control logic are not supported.

This affects the following registers:
- DMA_OTSS (OCDS Trigger Set Select)
- DMA_ERRINTR (Error Interrupt)
- DMA_PRR0 (Pattern Read Register 0)
- DMA_PRR1 (Pattern Read Register 1)
- DMA_MODEy (Hardware Resource Mode)
- DMA_HRRz (Hardware Resource Partition)
- DMA_SUSENRz (Channel Suspend Enable)
- DMA_TSRz (Transaction State)

**Workaround**

Make sure only 32-bit word data is written to the registers listed above by selecting the appropriate data types.

DMA_TC.017  Pattern Detection Double Interrupt Trigger when INTCT = 11B

A DMA channel z is configured for pattern detection by programming the DMA_CHCFGRz.PATSEL to reference a data value set in one of the pattern read registers DMA_PRR0 or DMA_PRR1. If DMA_ADICRz.INTCT = 11B then DMA channel z will generate a channel interrupt trigger and set CHSRz.ICH each time TCOUNT is decremented.

If a pattern match is detected then a channel interrupt trigger will be correctly generated but a second channel interrupt trigger will be generated when TCOUNT decrements. The second interrupt trigger is a bug and should not occur.
If the DMA channel z interrupt trigger is directed via the Interrupt Router to generate a DMA hardware request to another DMA channel then the second interrupt trigger may result in a Transaction Request Lost event.

**Workaround**

Workaround is to ignore the generation of the Transaction Request Lost event:

- Either disable the generation of error interrupt service requests by setting ADICRz.ETRL = 0B
- Or if the error interrupt service request is enabled, check all error status bits. If only the TRL bit for DMA channel x (pattern detection channel) is set then clear TRL and continue normal DMA operation.

**DMA_TC.018  FPI timeout can cause pipelined register reads to break**

Due to a problem in the FPI slave interface (SIF) to the System Peripheral Bus (SPB) in the DMA module, a register access which is pipelined behind an access which is timed-out may terminate early and return the wrong data to the bus.

The scenario for this problem to occur is as follows:

1. An FPI read transaction is performed which takes a long time in the data phase. Pipelined behind this is a register access to DMA or Cerberus.
2. The first transaction is timed out, and in the same cycle the register access is taken by the SIF.

**Workaround**

Timeout indicates a severe problem, meaning that something took unexpectedly long. In the event of an FPI timeout on the SPB, an error routine should be run to determine the error, and perform a system reset.
DMA TC.019 CBS Accesses with Large SPB:SRI Clock Ratios Configured

When operating in debug mode and a large SPB:SRI clock ratio is configured then Cerberus accesses to the SRI address space may be unreliable and result in the Cerberus hanging.

Workaround
Limit the SPB:SRI clock ratio to 1:1, 2:1, 3:1 or 4:1, and do not perform Cerberus accesses to the SRI address space while switching the SPB:SRI clock ratio.

DMA TC.020 DMA Conditional Linked List: Circular Buffer Enabled

When a DMA channel is configured for Conditional Linked List (i.e. ADICRz.SHCT = 1111B) and circular buffer operation (i.e. ADICRz.SCBE = 1B OR ADICRx.DCBE = 1B) then if the source and destination addresses are not set to wrap boundaries then the behaviour will not be as intended, e.g. the wrap bits CHCSRz.WRPS and CHCESRz.WRPD may be spuriously set.

Workaround
If a DMA channel is configured for Conditional Linked List and circular buffers are enabled then the user must set the source and destination addresses to wrap boundaries.

DMA TC.021 Combined Software/Hardware Controlled Mode Spurious Errors

A DMA channel is configured for combined software/hardware controlled mode. If the Move Engine is servicing a DMA channel software request and a DMA channel hardware trigger is received then a Transaction Request Lost event is set. When the Move Engine completes the current DMA access the TSRz.CH bit is not cleared. The DMA channel will continue to request channel arbitration.
as the CH bit is set. If the DMA channel wins arbitration then the Move Engine will continue to service the DMA channel.

In summary, 2 DMA requests (software and hardware) have resulted in 2 X DMA transfers and 1 X Transaction Request Lost (i.e. 3 X DMA actions for 2 X DMA triggers) i.e. a spurious error is generated.

**Workaround**

If a DMA channel is configured for combined software/hardware mode then increased attention must be paid to de-conflict the triggering of DMA channels from the servicing of DMA requests. The workaround will remove the source of spurious errors.

---

**DMA_TC.022  Conditional Linked List: Bus Error**

When a DMA channel is configured for Conditional Linked List (i.e. ADICRz.SHCT = 1111B) then if a bus error is reported then:

- If there is a pattern match then the number of DMA moves subsequently executed may not be as intended.
- If there is an error during the loading of a new Transaction Control Set then the DMA channel does not clear the TSRz.CH bit and begins the next DMA transaction with an erroneous Transaction Control Set.

**Workaround**

If a DMA channel is configured for Conditional Linked List then the user must enable the error interrupt service request. On receiving notification of an error interrupt service request the user must read the Move Engine Error Status Registers to confirm that no bus errors were reported:

- If DMA_ERRSRx.DER = 0B and DMA_ERRSRx.SER = 0B then no bus errors reported.
- If a bus error is reported then check the last error channel DMA_ERRSRx.LEC.
- If DMA_ERRSRx.DLLER = 1B then there was an error during the loading of a new Transaction Control Set.
**DMA TC.024  Suspend Request coincident with Channel Activation**

If DMA channel z is suspend enabled (SUSENRz.SUSEN = 1\(_B\)) and the DMA receives a suspend request then if during the same clock cycle the DMA channel becomes active in a Move Engine, the following effects will occur:

- SUSACRz.SUSAC is set for a cycle and then cleared
- A DMA transfer is performed for DMA channel z
- SUSACRz.SUSAC is set again on completion of the DMA transfer and the DMA channel is finally suspended.

**Workaround**

When polling SUSACRz.SUSAC in software, additionally check whether DMA channel z is active in a Move Engine x by reading bit field MExSR.CH.

**DMA TC.025  Conditional Linked List: new non-CLL mode TCS load can corrupt SDCRC RAM write**

When a Conditional Linked List (CLL) transaction is running and gets a CLL pattern match, this will stop the running transaction and cause a transaction control set (TCS) load.

In case the new TCS load is set up so that it is not in CLL mode, then the SDCRC value of the new TCS may get corrupted.

**Workaround**

Avoid selection of non-CLL mode in the TCS loaded after a CLL pattern match.

**DMA TC.026  Linked List: Failed TCS load can trigger wrap interrupt**

When a Transaction Control Set (TCS) linked list load is performed, and an error is received during the load process, this terminates the load. A DMA linked list error is indicated by the error status flag ERRSRx.DLLER.

If the DADR address left in the register matches the destination wrap boundary, this results in the issuing of a destination wrap interrupt in case the destination wrap interrupt enable is set. Hence a failed TCS load has triggered an interrupt.
Note: This only happens for destination interrupts. Logic is already in place to exclude source interrupts.

Workaround

An error interrupt for the DMA linked list error is triggered by the status flag ERRSRx.DLLER if enabled by EERx.ELER. Therefore the destination wrap buffer interrupt can be ignored in this case.

**DMA TC.028 Transaction Request Lost interrupt behaviour**

The Transaction Request Lost error interrupt can be viewed as a debug feature, supporting the prevention of missed DMA channel hardware requests.

If a Transaction Request Lost event occurs for channel z (indicated by TSRz.TRL = 1) during the arbitration of DMA channels at the end of a DMA Transaction/Transfer (dependent on RROAT) then the generation of the corresponding transaction request lost interrupt service request depends on the SRI:SPB clock ratio and whether the next active channel n is the same (n=z) or a different one (n≠z):

- If the next active channel n is the same as the current one (n=z):
  - For a 1:1 SRI:SPB clock ratio, two TRL error interrupts for the same cause are generated (for the current transaction, and again with the next channel load).
    The second TRL interrupt is signaled by an interrupt overflow (via bit SRC.IOV=1B) and should be ignored. The error interrupt routine should clear SRC.IOV by writing a 1B to SRC.IOVCLR.
  - For other clock ratios (i.e. SRI clock speed > SPB clock speed), one TRL error interrupt is generated after the next DMA channel z transaction control set load.

- If the next active channel n is different to the current one (n≠z):
  - For a 1:1 SRI:SPB clock ratio, one TRL error interrupt is seen for the current DMA channel z transaction
  - For other clock ratios (i.e. SRI clock speed > SPB clock speed), the generation of the TRL error interrupt is delayed until the next time DMA channel z is active.
The TSRz.TRL generated error interrupt will only be triggered when ADICRz.ETRL = 1.

**DMA_TC.029  DMA Double Buffering Overflow**

For all four DMA Double Buffering modes:
- DMA Double Source Buffering Software Switch Only,
- DMA Double Source Buffering Automatic Hardware and Software Switch,
- DMA Double Destination Buffering Software Switch Only,
- DMA Double Destination Buffering Automatic Hardware and Software Switch,

if the application allows the buffers to overflow then the DMA may lock up.

**Workaround**

If a DMA channel is configured for Double Buffering then the DMA channel Enable Transaction Request Lost bit (DMA_ADICRz.ETRL) must be set to 1B.

**DMA_TC.031  CHCSR.ICH can be incorrectly set after pattern match**

If a pattern match is seen during a transaction, the transaction is halted for the current active channel. The move engine zeroes its internal move counter, and holds the transfer count status MEx_CHCSR_TCOUNT at the last value. However, the MEx_CHCSR.ICH bit will still be set indicating a TCOUNT decrement.

**Workaround**

As there is a pattern match, a DMA channel pattern match interrupt service request will be generated. The pattern match interrupt routine can service the interrupt and clear the status bits including ICH.
DMA TC.034  Timestamp written incorrectly to wrapped address

The specified behaviour is that a timestamp is appended to the end of a DMA transaction. In the destination circular buffer use case it must be appended to the end of data and must not overwrite destination data. A design bug results in the timestamp overwriting destination circular buffer data.

In destination circular buffer mode the timestamp is written to the next aligned address after the last written sample. If the last sample is at the top of the circular buffer, the timestamp will be written above it. In other words, the timestamp address does not wrap back.

*Note:* Timestamp works as specified when using only source circular buffer.

**Workaround**

If a DMA channel is configured

- for destination circular buffering (ADICRz.DCBE = 1\textsubscript{B}) AND
- increment of DMA destination address (ADICRz.INCD = 1\textsubscript{B}) AND
- the appendage of a DMA timestamp (ADICRz.STAMP = 1\textsubscript{B}),

the DMA timestamp shall be appended only if the following DMA channel parameters are configured:

- ADICRz.DMF = 001\textsubscript{B} (Address offset is 2 x CHCFGRz.CHDW)
- AND CHCFGRz.CHDW = 010\textsubscript{B} (32-bit data width for moves, SDTW).

In all other destination circular buffer (ADICRz.DCBE = 1\textsubscript{B}) use cases, the appendage of DMA timestamp shall be disabled (ADICRz.STAMP = 0\textsubscript{B}).

DMA TC.035  Last DMA Transaction in a Linked List triggers a DMA Daisy Chain

DMA Channels can be daisy chained by setting the bit CHCFGRz.PRSEL = 1\textsubscript{B}. When a higher priority DMA channel z completes a DMA transaction then it will initiate a DMA transaction on the next lower priority DMA channel z-1 by setting the access pending bit TSRz-1.CH.

However, if the current transaction was the last one in a linked list, and PRSEL is set to daisy chain, TSRz-1.CH of the next lower channel z-1 is set just after the TCS (transaction control set) load, that is, before the last transaction of the
linked list has even started. Therefore the last TCS is not executed by the Linked List.

**Workaround**

Do not use Daisy Chain with Linked Lists (i.e. if ADICRz.SHCT[3:2] = 11\text{B} then CHCFGRz.PRSEL = 0\text{B}).

If the use case needs to trigger a further TCS in the next lower DMA channel then the trigger should be routed via the Interrupt Router.

**DMA TC.036  Linked List: SADR/DADR can be overwritten when loading a non-LL TCS**

If a Linked List (LL) loads in a non-LL Transaction Control Set (TCS) which has a shadow mode selected (ADICRz.SHCT = 0001\text{B} or 0010\text{B} or 0100\text{B} or 0101\text{B}), during the write-back it can overwrite the contents of SADR/DADR in the newly loaded TCS before the DMA transaction has been run.

**Workaround**

Do not use shadow address modes with DMA Conditional Linked List.

*Note: The Application Note AP32245 “DMA Linked List” will highlight that shadow address modes are not required.*

**DMA TC.037  Conditional Linked List: Bit TSR.CH not cleared for a CLL transaction upon pattern match**

When a Conditional Linked List (CLL) pattern match is found, the transaction ends. TSR.CH should be cleared, and set later during write-back of the Transaction Control Set (TCS) if the newly loaded TCS is auto-starting (i.e. CHCSRz.SCH = 1\text{B}).

Due to an internal problem TSR.CH is not cleared in this case.

**Workaround**

There is no workaround.
The assessment is that a DMA CLL transaction that does not get a match will transition to the next DMA transaction. The CH bit will be cleared.

**DMA TC.038  Linked List: SIT interrupt when SIT bit set in newly loaded TCS**

The Set Interrupt Trigger (SIT) bit is a means of generating a DMA channel interrupt service request via software. It is a debug feature that allows to trigger the Interrupt Router, without configuring the DMA channel and executing a DMA transaction.

When a new Transaction Control Set (TCS) is loaded in linked list mode, and the SIT bit in the new TCS being loaded is set in the value written to register CHCSRz, a channel interrupt trigger will be activated.

Therefore, the SIT bit should always be set to 0B when using linked lists.

*Note: The latest versions of the documentation are/will be updated to reflect this.*

**DMA TC.039  Read Data CRC**

The Read Data CRC (RDCRC) calculates an IEEE 802.3 ethernet CRC32 checksum as DMA moves read data through the DMA. The DMA implementation of the algorithm does not zero extend the read data for SDTB (8-bit) and SDTH (16-bit) accesses resulting in the calculation of a wrong checksum value.

The RDCRC must only be used with STDW (32-bit), SDTD (64-bit), BTR2 (128-bit) and BTR4 (256-bit) access sizes. It must be noted that SDTD, BTR2 and BTR4 are only supported for SRI-source to SRI-destination transactions.

**DMA TC.040  DMA Linked Lists: Intermittent Clearing of Hardware Transaction Request Enable with mixed mode Transaction Control Sets**

When a DMA channel is configured for linked list operation, if a Transaction Control Set (TCS) is configured for Continuous Mode...
(DMA_CHCFGGrz.CHMODE = 1B) and the next TCS is configured for Single Mode (DMA_CHCFGGrz.CHMODE = 0B) then DMA_TSRz.HTRE may be intermittently cleared disabling the servicing of DMA hardware requests.

**Workaround**

If a DMA channel is configured for linked list operation then all application DMA transactions must be configured for Continuous Mode (DMA_CHCFGGrz.CHMODE = 1B). If there is a need for the application to clear the Hardware Transaction Request Enable (DMA_TSRz.HTRE = 0B) then two additional dummy DMA transactions should be serviced by the DMA in the linked list:

- **Dummy Transaction 1:**
  the TCS is configured as a linked list TCS (DMA_ADICRz.SHCT = 0xC, 0xD or 0xE) in Single Mode (DMA_CHCFGGrz.CHMODE = 0) and auto start (DMA_CHCSRz.SCH = 1B). The TCS should configure a single DMA move to read a word from memory in order to write DMA_TSRz.DCH = 1B and disable subsequent DMA hardware requests.

- **Dummy Transaction 2:**
  the TCS is configured for normal shadow control mode (DMA_ADICRz.SHCT = 0000B) and Single Mode. A dummy DMA move is performed.

**DMA_TC.041 DMA Circular Buffer Wrap Interrupt**

If a DMA channel is configured for source circular buffer operation (ADICRz.SCBE = 1B), the DMA shall correctly calculate the DMA source addresses. When the DMA source address wraps, the DMA is unreliable in updating the wrap source buffer status (CHCSRz.WRPS). If the wrap source buffer interrupt is enabled (ADICRz.WRPSE = 1B), the DMA is unreliable in triggering a source wrap buffer interrupt.

If a DMA channel is configured for destination circular buffer operation (ADICRz.DCBE = 1B), the DMA shall correctly calculate the DMA destination addresses. When the DMA destination address wraps, the DMA is unreliable in updating the wrap destination buffer status (CHCSRz.WRPD). If the wrap
destination buffer interrupt is enabled (ADICRz.WRPDE = 1_B), the DMA is unreliable in triggering a destination wrap buffer interrupt.

**Workaround**
The source wrap buffer interrupt shall be disabled (ADICRz.WRPSE =0_B).
The destination wrap buffer interrupt shall be disabled (ADICRz.WRPDE =0_B).

If a DMA channel is configured for circular buffer operation (ADICRz.SCBE = 1_B or ADICRz.DCBE = 1_B), the DMA channel shall be configured as follows:

- The size of the DMA transaction shall equal the size of the circular buffer.
- If a source circular buffer is configured (ADICRz.SCBE = 1_B), the initial DMA source address shall be the start address of the source circular buffer.
- If a destination circular buffer is configured (ADICRz.DCBE = 1_B), the initial DMA destination address shall be the start address of the destination circular buffer.
- The DMA channel interrupt control shall be configured to trigger an interrupt on completion of the DMA transaction (DMA_ADICRz.INTCT = 10_B and DMA_ADICRz.IRDV = 0000_B).

If a DMA channel is configured for both source circular buffer operation (ADICRz.SCBE = 1_B) AND destination circular buffer operation (ADICRz.DCBE = 1_B), the size of the source circular buffer shall equal the size of the destination circular buffer.

**DMA_TC.042  DMA Interrupt from Channel reported before Completion of DMA Transaction**

The Interrupt from Channel (ICH) status bit should be set on completion of a DMA transaction. If the DMA channel is configured to append a DMA Timestamp then validation have discovered that the ICH bit is set before the DMA timestamp has been written.

**Workaround 1**
On receipt of a DMA channel interrupt service request software shall poll the Move Engine (ME) Status Register(s) to confirm the DMA channel is no longer active.
1. Check active DMA channel in ME SR.
2. Check Write Status in ME SR.

If these fields in both ME are no longer the DMA channel that triggered the DMA channel interrupt service request then the DMA transaction has completed.

Workaround 2
To avoid polling the Move Engine status, the user may use a DMA linked list to execute the following DMA transactions:

- DMA transaction 1:
  - move operation (DMA timestamp shall not be selected)
- DMA transaction 2:
  - single 32-bit DMA move to copy DMA timestamp from DMA TIME register to next 32-bit aligned destination after DMA transaction 1.

**DMA_TC.043 DMA Write Move Data Corruption for non 32-byte Aligned Cacheable Source Address**

If the DMA channel TCS selects a 256-bit channel data width and a non 32-byte aligned source address then the beat order of the DMA write move will be different for DMA read moves to cacheable (segments 8 and 9) and non-cacheable (segments A and B) source addresses. The effect is data corruption for accesses to cacheable addresses.

Workarounds
1. Use 32-byte aligned source addresses for DMA read move to cacheable addresses (segments 8 and 9).
2. Use non-cacheable source addresses (segments A and B).

**DMA_TC.044 Clock Switch after SPB Error Reported results in Spurious SRI Error**

If an SPB error is reported, and then immediately the SRI:SPB clock ratio is changed, then if the next DMA read move is to an SRI source address a spurious error may be reported.
Workaround

1. The system shall not change the SRI:SPB clock ratio while the DMA is active.
2. The DMA error handler should monitor the reporting of SPB and SRI errors after a clock switch.

**DMA_TC.045  DMA Reconfigures DMA Channels Lockup**

If two or more DMA channels are used to re-configure other DMA channels (i.e. perform a DMA write move to DMA address space) the DMA may lock up if the re-configuration DMA channels are assigned to different DMA hardware resource partitions.

The effect of the DMA lock up is to lock up other SPB master interfaces which attempt a write access to DMA address space.

**Workaround**

All DMA channels used to re-configure other DMA channels shall be assigned to the same hardware resource partition in their corresponding DMA Channel Hardware Resource Registers HRRz.

**DMA_TC.046  Shadow Operation Read Only Mode**

If a DMA channel is configured for Source Address Buffering Read Only (ADICR.SHCT = 0001B) or Destination Address Buffering Read Only (ADICR.SHCT = 0010B), the DMA is unreliable when performing a shadow address update. In these modes, the SADR/DADR registers may get directly updated (instead of SHADR) in the middle of a transaction, potentially resulting in a DMA data transfer corruption.

**Workaround**

The DMA channel configuration for Read Only Modes (SHCT = 0001B or SHCT = 0010B) must not be used.
Instead, to update the SADR/DADR in the middle of a transaction, use the corresponding Direct Write Mode for Source Address Buffering (ADICR.SHCT = 0101\textsubscript{B}) or Destination Address Buffering (ADICR.SHCT = 0110\textsubscript{B}), and write the new address to the SHADR register.

**DMA_TC.047 DMA Double Buffering Buffer Switch**

If a DMA channel is configured for one of the DMA double buffering operations:

- DMA Double Source Buffering Software Switch Only  
  - (DMA channel ADICRz.SHCT = 1000\textsubscript{B}),
- DMA Double Source Buffering Automatic Hardware and Software Switch  
  - (DMA channel ADICRz.SHCT = 1001\textsubscript{B}),
- DMA Double Destination Buffering Software Switch Only  
  - (DMA channel ADICRz.SHCT = 1010\textsubscript{B}),
- DMA Double Destination Buffering Automatic Hardware and Software Switch  
  - (DMA channel ADICRz.SHCT = 1011\textsubscript{B}),

AND software executes a Software Buffer Switch operation (DMA channel CHCSR.SWB = 1\textsubscript{B}), the DMA will not perform the buffer switch reliably.

**Workaround**

The following DMA double buffering operations must not be used:

- DMA Double Source Buffering Software Switch Only  
  - (DMA channel ADICRz.SHCT = 1000\textsubscript{B}),
- DMA Double Destination Buffering Software Switch Only  
  - (DMA channel ADICRz.SHCT = 1010\textsubscript{B}).

A DMA channel must be configured for DMA Double Buffering operations using one of the following configurations:

- DMA Double Source Buffering Automatic Hardware and Software Switch  
  - (DMA channel ADICRz.SHCT = 1001\textsubscript{B}),
- DMA Double Destination Buffering Automatic Hardware and Software Switch  
  - (DMA channel ADICRz.SHCT = 1011\textsubscript{B}),
AND software shall not assert a Software Buffer Switch. Only the DMA hardware shall perform buffer switching.

**DMA_TC.048 DMARAM Internal ECC Error**

If the DMA detects an integrity error when loading a TCS from DMARAM,

- The DMA shall
  - set DMA_MEMCON.INTERR,
  - trigger an alarm to the SMU,
  - record the DMA channel number in DMA_ERRSRx.LEC,
  - set the error status bit DMA_ERRSRx.RAMER.
- The DMA shall **not** execute the DMA transaction.

Erroneously,

- The DMA will not record the DMA channel number in DMA_ERRSRx.
- The DMA will execute the DMA transaction.

**Workaround**

None.

**DMA_TC.049 Bus Error Reported During LL TCS Load**

If a DMA channel is configured for Linked List (LL) operation AND a bus error is reported during the load of a new Transaction Control Set (TCS), the DMA shall set the DMA_ERRSRx.DLLER status bit (Move Engine x DMA Linked List Error).

Erroneously, the DMA additionally sets the DMA_ERRSRx.SER status bit (Move Engine x Source Error).

**Workaround**

None.
DMA_TC.050 Clearing CHCSR.FROZEN during Double Buffering

If a DMA channel is configured for one of the following Double Buffering operations:
• 1001\textsubscript{B} Double Source Buffering Automatic Hardware and Software Switch
• 1011\textsubscript{B} Double Destination Buffering Automatic Hardware and Software Switch

AND the active buffer fills/empties before software has cleared the DMA channel CHCSRz.FROZEN bit, the DMA shall overflow/underflow the active buffer.

Erroneously, the DMA will not trigger a Transaction Request Lost (TRL) error.

Workaround
Software shall clear DMA channel CHCSRz.FROZEN before the active buffer overflows/underflows.

DMA_TC.051 DMARAM Alarm

A DMARAM alarm is reported for the following error conditions:
• Internal ECC error: if the DMARAM signals an ECC error, the DMA shall set MEMCON.INTERR and trigger a DMARAM alarm.
• SPB read access: if the DMARAM signals an ECC error, the DMA shall set MEMCON.DATAERR and trigger a DMARAM alarm.
• SPB write access: if the DMARAM signals an ECC error during the read phase of an internal Read Modify Write, the DMA shall set MEMCON.RMWERR and trigger a DMARAM alarm.

Erroneously, the DMA additionally sets the following bits:
• SPB read access: if the DMARAM signals an ECC error, the DMA sets MEMCON.INTERR.
• SPB write access: if the DMARAM signals an ECC error, the DMA sets MEMCON.INTERR.

Workaround
None.
DMA_TC.052  SER and DER During Linked List Operations

If a DMA channel is configured for Linked List operation and a source error (ERRSRx.SER) or destination error (ERRSRx.DER) is reported, the DMA shall complete the current DMA transaction. In the case of the Conditional Linked List pattern matches will be disabled. The new transaction control set must not be loaded and the linked list stops to allow debug of the current DMA transaction. Erroneously, the DMA does not reliably stop the linked list operation (when it should) on completion of the current DMA transaction.

Workaround
None.

DMA_TC.053  TS16_ERR Type of Error Reporting Unreliable

During debugging, the error trigger set (TS16_ERR) may be used to identify the type of DMA error and the number of the DMA channel. After TS16_ERR reports an error the error type bits (ME0SE, ME0DE, ME1SE and ME1DE) are not cleared. If TS16_ERR reports a subsequent error, the type of error reporting is unreliable.

Workaround
After TS16_ERR reports an error, the error type bits must be cleared.

DMA_TC.054  DMA Channel Halt Acknowledge Unreliable

Software may halt a DMA channel by writing to the halt request bit (TSRz.HLTREQ = 1B). When a DMA channel enters the halt state, the DMA reports DMA channel halt acknowledge (TSRz.HLTACK = 1B).

The reporting of DMA channel halt acknowledge is unreliable when software sets the TSRz.HLTREQ bit just as channel z is about to be scheduled to a move engine. In this case, the DMA may report a DMA channel is halted when the DMA channel is active in a move engine.
Workaround
If the DMA reports a DMA channel is halted, the software should check the DMA channel is not active in a move engine by monitoring the active channel in the move engine status register(s).

**DMA TC.055 ICU to DMA Interface in Sleep Mode**

The Interrupt Router triggers DMA hardware requests via the ICU interface. If the DMA is in sleep mode, the DMA will not acknowledge DMA hardware requests. The effect is to lock up the ICU to DMA interface.

Workaround
The application must disable the triggering of DMA hardware requests before placing the DMA in sleep mode.

**DMA TC.056 TSR and SUSENR Access Protection Unreliable**

The DMA access protection is part of a system wide access protection scheme to restrict write accesses to DMA registers to individual on-chip bus masters. If the application software configures DMA freedom from interference measures (i.e. when any on-chip bus master write to the DMA is prohibited by a DMA access enable setting), then on-chip bus master writes to the DMA channel TSR and SUSENR registers are unreliable and may result in the following effects:

1. **Safety Related Effects**
   - 1.1. An illegal write access to a DMA channel TSR register will succeed with no indication.

The safety related effects (in point 1.1) relate to the DMA channel reset, halt and hardware request control functions in the TSR register. The most severe safety effect is that a DMA operation may be lost.
Workaround (for 1.1):
If the application software implements temporal monitoring of DMA transactions (e.g. using DMA timestamp) to detect lost DMA operations, the application software will detect the effect of the illegal access to DMA channel TSR register.

2. Non Safety Related Effects
• 2.1. An illegal write access to a DMA channel SUSENR register may succeed with no indication.
  – Impact of 2.1: The SUSENR register is a debug only register. No impact is foreseen during a normal application.
• 2.2. A legal write access to a DMA channel TSR register may fail with an indication - this means unexpected bus errors may be triggered when accessing TSR registers.
• 2.3. A legal write access to a DMA channel SUSENR register may fail with an indication - this means unexpected bus errors may be triggered when accessing SUSENR registers.
  – Impact of 2.2 & 2.3: Unexpected SPB bus errors and hence CPU traps and SPB error alarms may occur during application run.

Workaround (for 2.2 & 2.3):
If the system implements DMA freedom from interference measures, then the Impact of 2.2 & 2.3 will occur, and cause unexpected SPB bus errors and hence CPU traps and SPB error alarms when writing to TSR and SUSENR registers.

In order to work around this problem, the application software shall implement all of the following steps:
• W1: Before an intended write access to a DMA channel TSR or SUSENR register, perform an additional preceding write access to a DMA channel Transaction Control Set (TCS) register of the same DMA channel.
  – TCS registers include the DMA channel RDCRC, SDCRC, SADR, DADR, SHADR, ADICR, CHCSR and CHCFGR registers.
• W2: Ensure that this additional preceding write access to a DMA channel TCS register has no real effect. Recommendation: Simply read and write back the RDCRCR register.
• W3: Perform the write access to the DMA channel TSR register.
Ensure that no other on-chip bus master can access any DMA register of a different resource partition between steps W2 and W3 in the workaround above.

**Example Code Snippet:**
To update TSR register of DMA channel 25 with value:
1. Uint32 temp = DMA_RDCR25.U;
2. DMA_RDCR25.U = temp;
3. DMA_TSR25.U = value;

**DMA_TC.057 Double Buffering Overflow Causes Other Channel Corruption**

If DMA channel z is configured for one of the DMA double buffering operations:
- DMA Double Source Buffering Software Switch Only
  – (DMA channel ADICRz.SHCT = 1000B),
- DMA Double Source Buffering Automatic Hardware and Software Switch
  – (DMA channel ADICRz.SHCT = 1001B),
- DMA Double Destination Buffering Software Switch Only
  – (DMA channel ADICRz.SHCT = 1010B),
- DMA Double Destination Buffering Automatic Hardware and Software Switch
  – (DMA channel ADICRz.SHCT = 1011B),

AND DMA channel z overflows\(^1\), then as expected DMA channel z reports a Transaction Request Lost (TRL) event (TSRz.TRL = 1B) and clears the DMA request (TSRz.CH = 0B). However, erroneously the DMA channel z TRL event may cause the setting of the TRL flag and the clearing of a DMA request in one or more other DMA channels (note: dependent on the scheduling of DMA channels around this event).

---

\(^1\) Note: DMA channel z overflow condition is when CHCSRz.FROZEN = 1B AND CHCSRz.TCOUNT = 0D (second buffer is now full) and another DMA request arrives before the application software has changed this state.
If a DMA channel is configured for Double Buffering, the application software must ensure that the DMA channel does not overflow for correct operation of all DMA channels.

If the application software fails to prevent a double buffer overflow and the DMA error handler is notified of a TRL event in this situation, the application software must apply a DMA channel reset to all used DMA channels. On completion the application software must re-configure all used DMA channels.

**Safety Related Effects**

The effect of this problem is that one DMA channel may interfere with another DMA channel resulting in a lost DMA operation. The interference is independent of resource partition assignment.

**Workaround**

None.

**DMA_TC.058 Linked List Load Transaction Control Set (TCS) Integrity Error**

If DMA channel \( z \) is configured for one of the following linked list operations:

- DMA Linked List
  - (DMA channel ADICRz.SHCT = 1100\(_B\))
- Accumulated Linked List
  - (DMA channel ADICRz.SHCT = 1101\(_B\))
- Safe Linked List
  - (DMA channel ADICRz.SHCT = 1110\(_B\))
- Conditional Linked List
  - (DMA channel ADICRz.SHCT = 1111\(_B\))

Then on completion of a DMA transaction a new TCS is loaded into DMA channel \( z \) from the on-chip bus.

The DMA ignores data integrity errors in the new TCS:

- The DMA does not trigger an alarm to the SMU.
- The DMA does not store any DMA error status.
- The DMA may execute a corrupted DMA transaction.
Detection of most corrupted DMA transactions is provided by the DMA safety mechanisms as follows:

- Use of the DMA address checksum to detect address generation faults.
- Use of the DMA timestamp\(^1\) to detect temporal faults.

**Workaround**

None.

**DSADC_TC.004  Byte Write Access to CH8RUN, M8RUN, CH9RUN, M9RUN**

The following bits in register GLOBRC cannot be written via 8-bit (byte) write access:

- CH8RUN (GLOBRC.8), CH9RUN (GLOBRC.9)
- M8RUN (GLOBRC.24), M9RUN (GLOBRC.25)

However, these bits are written correctly via 16-bit (half-word) or 32-bit (word) write access. No restrictions apply to read access.

**Workaround**

Use only 16-bit (half-word) or 32-bit (word) write access to bits CH8RUN, M8RUN, CH9RUN, M9RUN in register GLOBRC.

**DSADC_TC.011  Modulator Coupling Option no longer supported**

The modulator coupling feature (two adjacent 3rd-order modulators can optionally be combined to operate as a 4th-order modulator and a 2nd-order modulator) is no longer supported in this device step.

Therefore, only the default setting DICFGx.DSRC = 0000\(^b\) must be used.

*Note: The DSADC parameter specification of the Data Sheet is achieved with the internal modulators operating in the default 3rd-order configuration*

---

\(^{1}\) Conditional Linked List does not support the appendage of timestamps (ADICRz.STAMP = 0\(^b\)).
(DICFGx.DSRC = 0000B).
The User’s Manual will be adapted accordingly.

**DSADC_TC.012** Common Mode Hold Voltage Not Applied During Calibration

The common mode hold voltage $V_{CMH}$ can be applied to a pin while this pin is not connected to the standard common mode voltage $V_{CM}$. This is the case while the input pin is not selected by the analog input multiplexer (MODCFGx.INMUX, if available for the respective channel) or while the modulator is switched off.

During calibration the modulator is connected to internal signal sources (MODCFGx.INCFG*), i.e. not to the pin.

In this case neither $V_{CM}$ nor $V_{CMH}$ are connected to the pin.

The voltage provided to passive sensors may, therefore, decrease for a short while (i.e. during the calibration time).

**Workaround**

None.

**DSADC_TC.013** Common Mode Voltage Selection

The common mode voltage $V_{CM}$ depends on the configuration of bit field MODCFGx.CMVS and on the supply voltage range selected by bit GLOBCFG.LOSUP.

The divider factors and resulting values of $V_{CM}$ for $V_{AREF} = 3.3$ V listed in the description of CMVS in register MODCFGx in the DSADC chapter of the User’s Manual are incorrect. The corrected description is shown in the following Table 10.
**Table 10**  Corrected Description of Bit Field CMVS in Register MODCFGx

<table>
<thead>
<tr>
<th>Field</th>
<th>Bits</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>Defines the common mode voltage $V_{CM}$ for the input buffers of a twin-modulator.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>$V_{CM}$ depends on CMVS and on the selected supply voltage range (GLOBCFG.LOSUP).</td>
</tr>
<tr>
<td>00B</td>
<td>$V_{CM} = V_{AREF} / 3.0</td>
<td>2.0 (for LOSUP = 0</td>
<td>1)</td>
</tr>
<tr>
<td></td>
<td>$V_{CM} = 1.67 V$ for $V_{AREF} = 5.0 \text{ V}</td>
<td>3.3 \text{ V}$</td>
<td></td>
</tr>
<tr>
<td>01B</td>
<td>$V_{CM} = V_{AREF} / 2.27</td>
<td>1.5 (for LOSUP = 0</td>
<td>1)</td>
</tr>
<tr>
<td></td>
<td>$V_{CM} = 2.21 V$ for $V_{AREF} = 5.0 \text{ V}</td>
<td>3.3 \text{ V}$</td>
<td></td>
</tr>
<tr>
<td>10B</td>
<td>$V_{CM} = V_{AREF} / 2.0</td>
<td>1.32 (for LOSUP = 0</td>
<td>1)</td>
</tr>
<tr>
<td></td>
<td>$V_{CM} = 2.5 V$ for $V_{AREF} = 5.0 \text{ V}</td>
<td>3.3 \text{ V}$</td>
<td></td>
</tr>
<tr>
<td>11B</td>
<td>Reserved</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

This means that $V_{CM} = V_{AREF} / 2.0$ is selected by CMVS = 10B when LOSUP = 0B, and by CMVS = 00B when LOSUP = 1B.

*Note: For most applications $V_{CM} = V_{AREF} / 2.0$ will be the optimum (see Section “Common Mode Voltage” in User’s Manual, chapter DSADC)*

---

**DTS TC.001**  Temperature Sensor Formula

The formula documented in older Data Sheet versions may result in an increased temperature error when calculating the junction temperature $T_j$ of the device from a DTS temperature measurement.
To properly calculate the temperature measured by the DTS in [°C] from the RESULT bit field of register SCU_DTSSTAT, it is recommended to use the following formulas depending on the contents of bit field SCU_DTSCON[30:29]:

- While bit field SCU_DTSCON[30:29] = 00_B: \( T_j = \frac{\text{RESULT} - 607D}{2.13} \)
- While bit field SCU_DTSCON[30:29] = 01_B: \( T_j = \frac{\text{RESULT} - 646D}{2.11} \)

Bit field SCU_DTSCON[30:29] can only deliver one of the two values (00_B, 01_B) listed above (constant for a given device).

Make sure the application software does not modify the values installed during device start-up in register SCU_DTSCON.

Note: The description in the Data Sheet will be updated appropriately.

**EBU_TC.024 Dynamic Changes of SDRAM clock are not possible**

The SDRAM clock SDCLKO is required by SDRAM memories and the frequency of this output is controlled by the BUSRAP.EXTCLOCK field of the region which has BUSRCON.AGEN set to 8_D. BUSWAP.EXTCLOCK has no effect for SDRAM.

In this implementation of the EBU, the SDRAM clock (SDCLKO) is output on the same pin as the Burst Flash Clock (BFCLKO). The additional logic required to share the clock pin contains a limitation which prevents dynamic changes of the SDRAM clock frequency.

Attempts for clock frequency changes requested by writing to BUSRAP.EXTCLOCK will have no effect. The SDRAM clock ratio set at boot time will therefore be permanent.

This also applies to shutdowns of the clock requested by SDRAM power saving modes, i.e. the SDRAM clock will always be present at the SDCLKO pin, independent of the setting of bit SDRMCON.SDCMSEL.

**EBU_TC.025 Connection of SDRAM Devices not supported**

SDRAM devices (corresponding setting BUSRCONx/BUSWCONx.AGEN = 8_D) cannot be supported by the EBU as timing of the interface signals cannot be guaranteed.
Workaround
The use of e.g. PSRAM as an alternative is suggested.

ETH AI.003  Overflow Status bits of Missed Frame and Buffer Overflow counters get cleared without a Read operation

The DMA maintains two counters to track the number of frames missed because of the following:
- Rx Descriptor not being available
- Rx FIFO overflow during reception

The Missed Frame and Buffer Overflow Counter register indicates the current value of the missed frames and FIFO overflow frame counters. This register also has the Overflow status bits (Bit 16 and Bit 28) which indicate whether the rollover occurred for respective counter. These bits are set when respective counter rolls over. These bits should remain high until this register is read.

However, erroneously, when the counter rollover occurs second time after the status bit is set, the respective status bit is reset to zero.

Effects
The application may incorrectly detect that the rollover did not occur since the last read operation.

Workaround
The application should read the Missed Frame and Buffer Overflow Counter register periodically (or after the Overflow or Rollover status bits are set) such that the counter rollover does not occur twice between read operations.

ETH TC.004  DMA Access to Reserved/Protected Resources: FPI Error Response not correctly evaluated

The ETH module includes a configurable DMA function to support the ETH Rx/Tx data transfers from/to system memory resources. The ETH DMA function accesses the system memory resources via the on-chip bus system (SPB/SRI).
If the ETH DMA is accessing reserved system address ranges or protected resources (e.g. protected via system MPU/ACCEN register), the ETH DMA transactions via the on-chip bus system will be finished on the on-chip bus with an Error Acknowledge.

Depending on the target address, the first transaction with an Error Acknowledge will be captured by the BCU_FPI (SPB Bus Control Unit) and/or by the XBAR_SRI. An interrupt can be generated by BCU_FPI / XBAR_SRI if the related SRN is enabled, and an Alarm is signalled to the SMU.

However, the ETH DMA will not be stopped by an Error Acknowledge. It will ignore the Error Acknowledge (i.e. bits FBI and EB in register ETH_STATUS are not set).

In this situation the ETH RX data transferred by the DMA to an invalid internal address will be lost, ETH will go on with invalid TX data.

**FFT_TC.001  FFT Access with disabled FFT Module**

*Note: This problem only applies to Emulation (ED) and ADAS devices.*

Contrary to the specification, read and write access to registers and RAM in the FFT module is possible while the module is disabled (bit FFT_CLC.DISS = 1B, default after reset).

**FFT_TC.002  FFT Kernel Reset Function**

*Note: This problem only applies to Emulation (ED) and ADAS devices.*

The kernel reset function (via bits RST in registers FFT_KRST0/1) does not properly reset all the FFT engine registers.

**Workaround**

Instead of performing a kernel reset on the FFT module, reset the entire device.
FFT_TC.003  No Error reported upon Write to FFT Registers in User Mode

Note: This problem only applies to Emulation (ED) and ADAS devices.

FFT registers FFT_CLC, FFT_CSR, FFT_BASEADDRESS, FFT_ODA, FFT_OCS, FFT_KRSTCLR, FFT_KRST1, FFT_KRST0 can only be written to in supervisor mode.

However a write to these registers in user mode does not report any error (on the bus or via an SMU alarm).

FLASH_TC.044  Repetitive Erase Suspend Requests on Data Flash

Note: This problem only affects devices with microcode version ≤ v2.2, identified by SCU_CHIPID.[23:17] = SCU_CHIPID.UCODE ≤ 0100010B. Devices with microcode version ≥ v2.3 are not affected. These devices are identified by SCU_CHIPID.[23:17] = SCU_CHIPID.UCODE ≥ 0100011B

If a suspend request of a Data Flash erase operation hits a specific time window tc of ~ 250..1000 µs (depending on Data Flash size) within a window of ~ 250 ms, other not addressed Flash cells in the same Data Flash bank can get gradually erased in case the critical window is repeatedly hit > 100 times cumulated over lifetime. For typical use cases only the first 24 µs of this window are relevant in which the UCB is selected.

The gradual erase may finally result in data corruption, and in case the UCB as part of DF0 is affected, it will prevent the device from booting.

Sources of Erase Suspend Request

Typical sources to trigger an erase suspend request are:

- TriCore: Erase suspend command issued from TriCore.
  Most common use case: EEPROM emulation (e.g. erase suspend for DF0 write or read requests).
- Reset: all resets are automatically triggering a suspend of running erase processes such that the Flash is put into a non-critical state.
• HSM (for TC29x and TC27x devices with HSM): HSM can automatically trigger erase suspend requests at DF0 when it is writing or erasing within its own block DF1 while DF0 is being erased.
  – Note: HSM reading from its own DF1 is not an issue.

Safety Aspects of this Problem

• Errors within DF0 are detected by the ECC mechanisms or by integrity checks on application level, e.g. CRC.
• Boot relevant entries of the UCB are additionally protected in order to detect special cases like full erase states, too. In case the errors within the boot relevant entries of the UCB are not correctable by the ECC the device will not boot.

Note: The problem effect was detected during a “suspend stress sweep” test of the erase sequence that systematically hit the critical time window. The probability of failure for a given application therefore depends on whether and how often the critical time window is hit by the sources generating an erase suspend request.

Note: PFLASH erase suspend requests theoretically could also trigger this problem. However, usually PFLASH is much less frequently reprogrammed in comparison to DFLASH, and it is only suspended to keep e.g. communication to an external programmer device alive. The result of an erase/program will be verified by an external programmer, and any problem would be found during the verify (end-of-line or in garage).

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

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

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

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

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

**FlexRay.AI.088  A sequence of received WUS may generate redundant SIR.WUPA/B events**

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

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

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

Ignore redundant SIR.WUPA/B events.

---

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

**Description:**

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

**Scope:**

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

**Effects:**

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

**Workaround**

It is recommended to set gMaxWithoutClockCorrectionPassive to 1. If missing sync frames cause the node to enter NORMAL_PASSIVE state, use higher level application software to leave this state and to initiate a re-integration into the cluster. HALT state can also be used instead of NORMAL_PASSIVE state by setting pAllowHaltDueToClock to true.
FlexRay_AI.090  Flag SFS.MRCS is set erroneously although at least one valid sync frame pair is received

Description:
If in an odd cycle 2c+1 after reception of a sync frame in slot n the total number of different sync frames per double cycle has exceeded gSyncNodeMax and the node receives in slot n+1 a sync frame that matches with a sync frame received in the even cycle 2c, the sync frame pair is not taken into account by CSP process. This may cause the flags SFS.MRCS and EIR.CCF to be set erroneously.

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

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

Workaround
Correct configuration of gSyncNodeMax.

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

Description:
If a valid sync frame is received before the action point and additionally noise or a second frame leads to a STRP coinciding with the action point, an incorrect deviation value of zero is used for further calculations of rate and/or offset correction values.
Scope:
The erratum is limited to configurations with an action point offset greater than static frame length.

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

Workaround
Configure action point offset smaller than static frame length.

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

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

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

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

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

**FlexRay_A1.093 Acceptance of startup frames received after reception of more than gSyncNodeMax sync frames**

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

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

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

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

**FlexRay_A1.094 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 \(c_{\text{StaticSlotIDMax}} + g_{\text{SyncNodeMax}} = 1023 + g_{\text{SyncNodeMax}}\), 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 \(g_{\text{SyncNodeMax}}\) and the number of static slots plus the number of dynamic slots is greater or equal than \(1023 + g_{\text{SyncNodeMax}}\).

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 \(g_{\text{SyncNodeMax}}\) to number of transmitted and received sync frames plus one or avoid configurations where the total of static and dynamic slots is greater than \(c_{\text{StaticSlotIDMax}}\).

---

**FlexRay_A1.095  Register RCV displays wrong value**

Description:
If the calculated rate correction value is in the range of \([-p_{\text{ClusterDriftDamping}} .. +p_{\text{ClusterDriftDamping}}\] \(, \) \(vRateCorrection\) 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 \([-p_{\text{ClusterDriftDamping}} .. +p_{\text{ClusterDriftDamping}}\] instead of zero.

Scope:
The erratum is limited to the case where the calculated rate correction value is in the range of \([-p_{\text{ClusterDriftDamping}} .. +p_{\text{ClusterDriftDamping}}\].

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

**Workaround**

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

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

**Description:**

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

**Scope:**

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

**Effects:**

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

**Workaround**

None.
**FlexRay_AI.097** Loop back mode operates only at 10 MBit/s

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

**Scope:**
The erratum is limited to test cases where loop back is used with the baud rate prescaler ($\text{PRTC1.BRP}[1:0]$) 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 ($\text{PRTC1.BRP}[1:0] = 00_B$).

**FlexRay_AI.099** Erroneous cycle offset during startup after abort of startup or normal operation

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

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

If the node is able to correct the offset of one macrotick ($\text{pOffsetCorrectionOut} \gg \text{gdMacrotick}$), the node enters NORMAL_ACTIVE with the first startup attempt.

If the node is not able to correct the offset error because $\text{pOffsetCorrectionOut} \leq \text{gdMacrotick}$, the node enters...
ABORT_STARTUP and is ready to try startup again. The next (second) startup attempt is not effected by this erratum.

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

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

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

**FlexRay.AI.100 First WUS following received valid WUP may be ignored**

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

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

Effects:
Delayed setting of status interrupt flags SIR.WUPA/B for redundant wakeup patterns.
Workaround
None.

**FlexRay Al.101  READY command accepted in READY state**

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

Scope:
The erratum is limited to the READY state.

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

Workaround
None.

**FlexRay Al.102  Slot Status vPOC!SlotMode is reset immediately when entering HALT state**

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

Scope:
The erratum is limited to the HALT state.
Effects:
The slot status vPOC!SlotMode is reset to SINGLE when entering HALT state.

Workaround
None.

**FlexRay AI.103** Received messages not stored in Message RAM when in Loop Back Mode

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

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

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

Workaround
Leave HALT state by hardware reset.

**FlexRay AI.104** Missing startup frame in cycle 0 at coldstart after FREEZE or READY command

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

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

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

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

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

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

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

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

If the parity logic of the erroneously selected IBF1 resp. OBF2 detects a parity error, bit MHDS.PIBF resp. MHDS.POBF in the E-Ray Message Handler Status register is set although the addressed IBF2 resp. OBF1 had not error. The logic for setting MHDS.PIBF / MHDS.POBF does not distinguish between set conditions from IBF1 or IBF2 resp. OBF1 or OBF2.
Due to the IBF / OBF swap mechanism as described in section 5.11.2 in the E-Ray Specification, the inverted behaviour with respect to IBF1,2 and OBF1,2 can be observed depending on the IBF / OBF access history.

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

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

Severity:
Low, workaround available.

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

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

The problem occurs under the following conditions:
1) A received message is transferred from the Transient Buffer RAM (TBF) to the message buffer that has its data pointer pointing to the first word of the Message RAM’s Data Partition located directly after the last header word of the Header Partition of the Last Configured Buffer as defined by MRC.LCB.
2) The Host triggers a transfer from / to the Last Configured Buffer in the Message RAM with a specific time relation to the start of the TBF transfer described under 1).

Under these conditions the following transfers triggered by the Host may be affected:
a) Message buffer transfer from Message RAM to OBF
When the message buffer has its payload configured to maximum length \((PLC = 127)\), the OBF word on address 00h (payload data bytes 0 to 3) is overwritten with unexpected data at the end of the transfer.

![Diagram of message buffer transfer from Message RAM to OBF]

**Figure 2**  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 3  Message buffer transfer from IBF to Message RAM

Scope:
The erratum is limited to the case when (see Figure 4 “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 2).
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 3).
Severity:
Medium, workaround available, check of configuration necessary.

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

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

Figure 4 Message RAM Configurations

GTM_A1.139 ATOM SOMC mode: forced update does not activate comparison

Under following configuration:
• ATOM SOMC mode,
• ARU_EN=1,
• no comparison active (bit DV in register ATOM[i]_CH[x]_STAT = 0)

If in this case a late update is tried by first setting WR_REQ (see register ATOM[i]_CH[x]_CTRL), then updating SRx register and maybe ACB control bits in register ATOM[i]_CH[x]_CTRL and finally updating the CMx register via
a forced update, the register CMx are updated correctly but no new comparison is activated.
The ACBO bits are erroneously not cleared.
The ARU read request is canceled because of WR_REQ=1.

**Scope**
ATOM SOMC mode.

**Effects**
In the described case, the ARU read request is canceled but no new comparison with new CMx register values is activated.
The system may stick in waiting for late update event to happen.
The ACBO bits are erroneously not cleared.

**Workaround**
After the forced update write additionally by CPU the new desired value of CM0 or CM1 to corresponding work register CM0 or CM1 to activate comparison and to reset ACBO bits.

**GTM_AI.140** ATOM SOMC mode: a write access to ATOM_CH_CTRL sets WRF if CCU0 compare match already occurred but CCU1 compare match open

Under following configuration:
• ATOM SOMC mode,
• ARU_EN=1
For compare strategy 'serve last', if after CCU0 compare match and before CCCU1 compare match a write access to register ATOM_CH_CTRL is done, WRF bit is set independent of written bit WR_REQ.

**Scope**
ATOM SOMC mode.
Effects
In the described case the WRF flag may be set erroneously.

Workaround
If ATOM[i]_CH[x]_CTRL is written without the intention to set WR_REQ while there may be a comparison active on this channel x, reset afterwards erroneously set WRF flag by writing a '1' to WRF bit of register ATOM[i]_CH[x]_STAT.

GTM.AI.141 TIM: Incorrect data captured to GPR registers and routed via ARU when EGPRi_SEL,GPRi_SEL= 100 in TIM channel mode TIEM, TP-WM, TIPM, TPIM, TGPS

In case of a TIM channel capture event issued by a rising edge at TIM[i]_CH[x]_FOUT the capturing of the TIM[i]_CH[x]_ECNT register to the TIM[i]_CH[x]_GPRi register is incorrect. The captured value will be ECNT_REG+2; bit 0 (signal level) will be 0. The correct operation would be to capture ECNT_REG+1; bit 1 (signal level) would be 1.

Scope
TIM.

Effects
a) Inconsistency of ARU signal level bit and bit[0] of ARU word which shows the captured ECNT.
b) Reading of TIM[i]_CH[x]_GPRi shows inconsistency when comparing bits [31:24] to [7:0]. At the point in time of capture event the bits [31:24] contain the correct value and are subject to be changed with new incoming edge.

Workaround
a) When using captured data via ARU routing the correct data can be reconstructed by:
IF ARU_SIGNAL_LEVEL ==1 AND ARU_DATA[0] == 0 THEN ARU_DATA = ARU_DATA -1;
b) When reading TIM[i]_CH[x]_GPRi by configuration interface the data can be corrected as long as there is no GPR overflow and no new edge by:

IF TIM[i]_CH[x]_GPRi[24] == 1 AND TIM[i]_CH[x]_GPRi[0] == 0 THEN
TIM[i]_CH[x]_GPRi[23:0] = TIM[i]_CH[x]_GPRi[23:0] -1

**GTM_AI.142** TIM: Incorrect data captured to GPR registers and routed via ARU when EGPRi_SEL,GPRi_SEL= 100 in TIM channel mode TBCM

In case of a TIM channel capture event issued by an input pattern match to condition TIM[i]_CH[x]_CNTS the capturing of the TIM[i]_CH[x]_ECNT register to the TIM[i]_CH[x]_GPRi register can be incorrect. Starting at t=0 with counter value ECNT_REG(t=0), the captured values of two consecutive edges can be ECNT_REG(t=0)+2 followed by ECNT_REG(t=0)+2 instead of ECNT_REG(t=0)+1 followed by ECNT_REG(t=0)+2.

**Scope**

TIM.

**Effects**

a) In 2 following ARU transfers the ARU word which shows the captured ECNT do not increment by 1.

b) Reading of TIM[i]_CH[x]_GPRi shows inconsistency between [31:24] and [7:0]

**Workaround**

a) Ignore captured data via ARU and build with MCS independent counter which increments on each ARU transfer.

b) When reading TIM[i]_CH[x]_GPRi by configuration interface use only TIM[i]_CH[x]_GPRi[31:24] as EDGE counter; don't use TIM[i]_CH[x]_GPRi[23:0].
GTM_AI.143  GTM_TOP level: AEI pipelined write to GTM_BRIDGE_MODE register directly after setting aei_reset='0' can result in blocking of AEI configuration interface

If the GTM bus bridge is reset with aei_reset= '0' and the next AEI transfer is a write command to GTM_BRIDGE_MODE register the AEI configuration interface can be blocked.

Scope
AEI pipelined protocol.

Effects
GTM Bus interface does not issue aei_ready which could lead to bus timeout of the serving bus master.

Workaround
Ensure that after setting aei_reset to inactive state the next command must be a read to any other register except GTM_BRIDGE_MODE. Issue desired write to GTM_BRIDGE_MODE register afterwards.

GTM_AI.146  ATOM SOMC mode: compare match does not clear WR_REQ

If an ATOM channel is operating in SOMC mode, ARU is enabled and, initiated by setting WR_REQ=1, a late update of CM0/CM1 register and/or compare strategy (i.e. ACB[4..0]) was successfully done, then, after final compare match the WR_REQ bit should be reset.
This is erroneously not done.

Scope
ATOM SOMC mode.

Effects
In the described case the bit WR_REQ is not reset. As a result no new ARU read request is set up after final compare match.
**Workaround**
Reset WR_REQ by software after late update (after forced update).

**GTM_A1.150 TIM: Valid edge after Timeout**

Assume that a TIM timeout event triggers an ARU write request with timeout information ACB2=1 and ACB1=0. If this request is acknowledged by the ARU while a new valid edge occurs, the valid edge is neither signalized by setting the bits ACB2=1 and ACB1=1 within the acknowledged transfer nor by setting up a new subsequent ARU write request for the new valid edge with ACB2=0 and ACB1=0.

**Scope**
TIM timeout detection in combination with ARU transfers.

**Effects**
If a valid edge occurs after a timeout event, the valid edge is not signalized reliably via the ACB bits over the ARU.

**Workaround**
The workaround for this issue requires an additional plausibility check within the MCS or CPU via FIFO:

1. Always store the received data ARUDATA(47:0)n and ACB0n in temporary variables.
2. If an ARU transfer with ACB2n+1=1 and ACB1n+1=0 is received also check the following:
   - If ACB0n+1 ≠ ACB0n OR ARUDATA(47:0)n+1 ≠ ARUDATA(47:0)n then a timeout with subsequent valid edge has occurred, which means ACB1 must be corrected to 1.
GTM_AI.152  DPLL: THVAL value not immediately available at inactive trigger slope

According to the specification chapter x.16.8.6\(^1\) it is specified that “for each invalid trigger slope....store this value to THVAL”. The value THVAL is calculated correctly but this value is stored into the THVAL memory location with every new active edge of the trigger signal.

**Scope**
DPLL storage of value THVAL into memory.

**Effects**
The value THVAL is not available in the memory at the specified point in time.

**Workaround**
If the THVAL value is needed immediately with the inactive trigger edge it is necessary to calculate the THVAL value by an TIM_CHO/1 to obtain the active and inactive slopes in input event mode. With this timestamps the CPU is able to calculate the time span within the CPU.

GTM_AI.153  TIM: Incorrect data captured to CNTS register when TIM channel operates in mode TPWM or TPIM and CNTS_SEL = 1 and selected CMU_CLK ≠ sys_clk

In case of CNTS_SEL = 1 and TIM_MODE = TPWM or TPIM in the CNTS_REG register the value of TBU_TS0 shall be captured. This does not happen when the selected CMU_CLK ≠ sys_clk.

**Scope**
TIM.

---

1) Section “Scheduling of the Calculation”, Table “State description of the State Machine”
Effects
Unexpected values in CNTS_REG.

Workaround
Setup the TIM channel to operate on a CMU_CLK (Divider =1) which is identical to sys_clk. Please notice that the measurement with TIM_CNT has resolution of sys_clk.

GTM_AI.154  TOM: Incorrect duty cycle in PCM mode (bit reversed mode)

The generated duty cycle on the TOM output in PCM mode is always one smaller than the configured value in the CM1 register. So if the value 1 is configured, a duty cycle of 0% will be generated. Configuring the max value (0xFFFF) in the CM1 register results in a duty cycle of max-1. Expected is 100% duty cycle in this case. A zero in CM1 register results in 100% duty cycle.

Scope
TOM.

Effects
Unexpected duty cycle in PCM mode.

Workaround
Configure always the value for the expected duty cycle in the CM1 register with expected duty cycle + 1.
To get 0% duty cycle, value 1 has to be configured. To get 100% duty cycle, 0 has to be configured to CM1 register while CM0 is always configured with max. value of 0xFFFF. Configuring CM0=0x1000 and CM1=0xFFFF will also get a duty cycle of 100%.
GTM_Ai.158  DPLL: Reset of pcm1/pcm2 bits in relation to an interrupt

The PCM1/2 bits are reset after the correction values MPVAL1/2 are used to calculate the number of sub_incs for the next increment and to calculate the add_in values. See specification chapter x.16.8.6\(^1\) (States 5, 25). The problem is that the PCM1 bit is transferred with an active edge into the dedicated shadow registers, but cleared some time later. If the PCM1/2 bits are written by the CPU in between the point of time of the transfer to the shadow register and the point of time were the PCM1/2 bits are cleared, the bits are cleared and never used. This is not what one should expect from a properly defined user interface and to prevent additional expenditure to calculate the correct point of time for writing the PCM1/2 bits.

From application point of view the desired behavior is that the PCM1/2 bits are cleared when transferred to their shadow registers (not in state 5, 25). The proposed workaround would fit to this described modification.

Scope
DPLL.

Effects
When the PCM1/2 bits are written in the critical timeframe the bits are cleared before they are used.

Workaround
The point of time when the PCM1/2 bits are written by the CPU must be around 750 system clocks after the TASI interrupt. This time could be derived by a GTM resource like an ATOM channel.

\(^1\) Section “Scheduling of the Calculation”, table “State description of the State Machine”
GTM.AI.161  DPLL MTI/TORI-IRQ’s are not activated when low_res='1' and ts0_hrt='1'; MSI/SORI-IRQ's are not activated when low_res='1' and ts0_hrs='1'

The DPLL Interrupts MTI/TORI are not raised when the DPLL is configured with low_res='1' and ts0_hrt='1' when the upper three bits of the tbu_ts0 are not equal to “000”.

The DPLL Interrupts MSI/SORI are not raised when the DPLL is configured with low_res='1' and ts0_hrs='1' when the upper three bits of the tbu_ts0 are not equal to “000”.

**Scope**

DPLL in mode low_res='1' and tso_hrt='1' ts0_hrs='1'.

**Effects**

When this effect is activated by the configuration and when the upper tbu_ts0 bits are not equal to “000” the interrupts MTI/TORI or MSI/SORI are not activated. A consequence of this is that the lock1/2 bits and the mti/msi flags in the DPLL_STATUS register are not operating correctly.

**Workaround**

Don't use the configuration low_res='1' and ts0_hrt='1'.

The signals are working correctly for the configurations low_res='0' and low_res='1' and ts0_hrt/s='0'.

GTM.AI.162  DPLL: Input signal (active edge) which is rejected by PVT-check occurring at a gap in the profile causes that the MTI_IRQ is not activated within this gap

The DPLL interrupt MTI_IRQ is not activated when during a gap in the profile an active input signal edge is rejected by the PVT check. In this case the internal check for the MTI-IRQ is done when the invalid active event is coming so that the check for the first and valid active event is not done for this gap.
Scope
DPLL.

Effects
For the gap where the described situation occurs the mti interrupt is not activated. In this moment the lock1/2 signals are unaffected. The possible problem is that in case of monitoring the DPLL synchronization e.g. with the use of the MTI_IRQ in a gap such monitoring may report a synchronization problem which is not real.

Workaround
The activated PVT check is reported by the activation of the PWI interrupt. This interrupt can be used to check if a gap condition in the profile has occurred. This information can be used to correct the wrong information out of the DPLL.

**GTM_AI.163  TIM: timeout signaled when TDU unit is reenabled**

In the following situation an undesired timeout event is signaled:
After stopping the TDU the TO_CNT bitfield will have an arbitrary value TO_CNT0 <= TOV0 bitfield. Assume TOV will be reconfigured to value TOV1 with TOV1 <= TO_CNT0. If the TDU will be enabled again by writing to TOCTRL a value !=0 and at the same time the TCS selected CMU_CLK has an active edge an unintended timeout is signaled. This results due to the fact that for one clock cycle TO_CNT0 >= TOV1.

Scope
TIM.

Effects
Unexpected timeout event when TIM TDU is enabled.
Workaround

If TDU unit has to be reenabled with a TOV value TOV1 which is less than the previous one in use TOV0 (2 alternatives are available):

a) Wait with disabling TDU until condition TOV1 > TO_CNT is fulfilled. Configure TOV with TOV1 reenable TDU Unit.
b) Disable TDU; if TOV1 <= TO_CNT write TOV with FF\textsubscript{H}; enable TDU unit; reconfigure TOV to desired value TOV1.

**GTM\_AI.164** TIM: capturing of data into TIM[i]\_CH[x]\_CNTS with setting CNTS\_SEL=1 not functional in TPWM and TPIM mode

If CNTS\_SEL=1 is selected and a new input edge is signaled by the TIM Filter unit while the selected CMU\_CLK has no rising edge the register TIM[i]\_CH[x]\_CNTS will capture data TIM[i]\_CH[x]\_CNT instead of TBU\_TS0.

Scope

TIM.

Effects

Captured data in TIM[i]\_CH[x]\_CNTS is not as expected.

Workaround

a) Select with CLK\_SEL a CMU\_CLK which is identical to sys\_clk (clock divider=1 applied in CMU channel and for global fractional divider).
b) Use TIEM mode to capture TBU\_TS0 for rising and falling input edges.  
c) PWM mode: Use CNTS\_SEL=0 with CMU\_CLK source selected as in use for TBU\_TS0 counting. Capture with EGPR0\_SEL=0, GPR0\_SEL=0 in GPR0\_REG TBU\_TS0 and with EGPR1\_SEL=0, GPR1\_SEL= 3 in GPR1\_REG CNT. Calculate the desired timestamp with GPR0\_REG - GPR1\_REG + CNTS\_REG.

**GTM\_AI.166** DPLL: The Content of registers DPLL\_apt\_sync.APT\_2b\_ext and DPLL\_aps\_sync.APS\_1c2\_ext is added independently of the state of
DPLL_apt_sync.APT_2b_status or DPLL_aps_sync.APS_1c2_status to the pointers apt_2b/aps_1c2

If during synchronization the registers DPLL_apt_sync.APT_2b_ext and DPLL_aps_sync.APS_1c2_ext are loaded with non zero values they are added to the pointers apt_2b/aps_1c2 independently from the status of the control bits DPLL_apt_sync.APT_2b_status or DPLL_aps_sync.APS_1c2_status. Correctly this should happen only when the control signals DPLL_apt_sync.APT_2b_status or DPLL_aps_sync.APS_1c2_status are set to “1”.

Scope
DPLL.

Effects
Wrong status of pointers apt_2b or aps_1c2 after synchronization has been executed.

Workaround
If the pointers apt_2b/aps_1c2 should remain unchanged after synchronization the registers DPLL_apt_sync.APT_2b_ext and DPLL_aps_sync.APS_1c2_ext must be set to zero before synchronization is performed.

**GTM_A1.167 ATOM SOMP mode**: for RST_CCU0=1 and ARU_EN=1, if CN0 reaches CM0 an update of the register SRx is requested

For the configuration ATOM SOMP mode, ARU_EN=1, RST_CCU0=1 an update of SR0/SR1 register via ARU is requested erroneously any time CN0 reaches CM0.

Because of RST_CCU0=1, if CN0 reaches CM0, CN0 is not reset but counting until it is reset be the trigger of a preceding channel. Therefore, it may not be the end of a period if CN0 reaches CM0.
The expected point in time for a new ARU read request to update the shadow register SR0/SR1 would be the trigger to reset CN0 which triggers also the update of CM0/CM1 with the value of SR0/SR1.

**Scope**

ATOM SOMP mode.

**Effects**

For the described configuration, the ATOM channel requests and updates the SR0/SR1 register not only after the update of CM0/CM1.

Depending on time between CM0 of this channel and the value of CN0 in case of reset by the trigger, the SR0/SR1 register may be updated two times between two triggers to reset CN0.

**Workaround**

1. If new data via ARU is provided by FIFO, avoid for ATOM SOMP mode the combination of configuration ARU_EN=1 and RST_CCU0=1
2. If new data is provided by MCS, ensure by MCS that only one time per period new data for SR0/SR1 register can be read. This can be reached by starting the 'master period' which triggers the reset of CN0 on a time base value and provide to the MCS the start value and the period. Then, the MCS can calculate a time for providing new ARU data.

---

**GTM_AI.168** DPLL: CPU read / write accesses to RAM2 in competition to DPLL accesses to RAM2 may lead to wrong SYN_T data read by DPLL

If at a dedicated point in time during sub increment calculation the DPLL TRIGGER processing unit reads a profile value out of RAM2 and in competition a second read/write operation is scheduled on the RAM2 via CPU/DMA interface, there is a dedicated state and signal constellation that leads to the effect that the RAM2 output data belonging to the CPU/DMA access is used as read data for the internal TRIGGER processing unit. This can lead to a wrong internal syn_t, syn_t_old value leading to a de-synchronization of the DPLL.
Scope
DPLL.

Effects
DPLL TRIGGER processing unit reads out from RAM2 wrong syn_t, syn_t_old data. As a result sub increment calculations of the DPLL are wrong.
This leads to loss of synchronization.

Workaround
The application SW has to avoid any access (CPU or DMA) to DPLL RAM2 in the time window starting with the active TRIGGER edge and ending with the TASI interrupt.

Workaround 1
Synchronization of CPU/DMA accesses to phases where DPLL is not accessing RAM2.
This can be reached by synchronizing DPLL RAM2 accesses to TRIGGER signal using the TASI interrupt and checking continuously if the RAM2 access is finished before next active TRIGGER edge.
As an alternative for TASI interrupt one can start with the TIM0_CH0 active edge interrupt an ATOM pulse (SOMP mode, one shot mode) of the length 200 SYS_CLK periods. With the CCU1 interrupt of the ATOM channel the critical phase of DPLL internal RAM2 accesses is finished and now the CPU/DMA can access DPLL RAM2.

Workaround 2
Asynchronous CPU/DMA accesses to phases where DPLL is not accessing RAM2.
This can be achieved by using MCS to calculate and set flags that indicate the uncritical phase of DPLL RAM2 accesses.
GTM_A1.169  DPLL: no TORI/SORI interrupt in case low_res = 1 AND ts0_hrt/s = 0

If the described configuration is chosen there is no TORI/SORI interrupt raised at all.

Scope
DPLL.

Effects
The TORI/SORI interrupt is not coming in that configuration.

Workaround
For the configuration low_res=1 and ts0_hrt/s = 0 use TOM or ATOM to generate an interrupt on time out of TRIGGER/STATE:
With every TRIGGER/STATE edge adapt (A)TOM period to current speed and reset CN0. If CN0 is not reset by next TRIGGER/STATE event, (A)TOM raises an edge interrupt at the end of the period.

GTM_A1.170  DPLL: Action calculation: requested action not always calculated immediately

If the action calculation by DPLL was interrupted due to a new input event it may happen that with the next TRIGGER/STATE input event, after sub increment calculation is finished, the action calculation starts again at the same internal action number which has been interrupted before. If in between new action data arrives where the action number is above the currently calculated action this new action data is only calculated after the next input event. The reason for that behavior is that if action calculation was interrupted the action calculation starts with the internal action address which was stored at the end of the event cycle before. New PMT data for action with higher action number are not recognized. The action calculation stops if the action number zero is reached.
Generally: The calculation of sub increments and PMT cannot be done in parallel due to resource sharing. This leads to the behavior that PMT calculation is interrupted if a new input event (TRIGGER/STATE) occurs.

When DPLL is doing the action calculations the DPLL has exclusive access rights to RAM1a which contains the PMT request values. Then the DPLL cannot accept new PMT requests via ARU.

**Scope**

DPLL.

**Effects**

Requested actions are not calculated regularly with every tooth (as long as they are not in the past).

**Workaround**

Request actions which are not “past” so far with every new tooth. The synchronization of the MCS task to TIM input event can be done by routing the TIM edge capture event value via ARU to MCS.

Then, if new PMT data is arriving after the action number has reached the value zero, the action is calculated immediately starting with the highest action number again.

As a workaround one can request the action calculation tooth by tooth until action runs into past. An additionally PMT request can be placed earlier after new input event (TRIGGER/STATE) while DPLL is doing sub increment calculations because then RAM1a can be handled exclusively for updating PMT requests via ARU.

Generally it is recommended to sent PMT requests at least 3 teeth before action has to be executed. This ensures that even under presence of the erratum the MCS, ATOM are getting calculated action results at least from a calculation of the action in an input event cycle before.
GTM_AI.172  TIM: overflow bit in TIM ARU data not set; signal level bit in ARU data has opposite value

Relevant mode TIEM with ISL=1 and ARU_EN=1.
In case of 2 input signal changes with distance smaller than ARU routing time the overflow Bit ACB1 might not be set.
The erroneous behavior occurs, if an edge (first_level) starts an ARU transfer and one system clock before the ARU request is serviced the input signal changes (! first_level). In this case the overflow bit ACB1 is not set (keeps ACB1=0), and the signal level bit ACB0 will be incorrect ACB0= first_level.
Note that the irq_notify(3) bit (gpr_overflow) is set correctly.

Scope
TIM.

Effects
The overflow information in the ARU ACB1 bit is not set, ARU ACB0 signal level incorrect.

Workaround A
Ensure with TIM filter that input signal changes smaller than ARU Routing Time will be removed. Configure FLT_FE/FLT_RE with filter delay which is greater than ARU Routing time.

Workaround B
Select ECNT or CNT to be transferred in ARU_DATA. Next is shown a pseudo code which can be used as a workaround:

Last_CNT = -1
For each ARU_DATA
  If ARU_DATA(ACB1) ==0
    If Last_CNT != -1
      If Last_CNT+1 != ARU_DATA(CNT)
        Message(Hit on ERRATA: Detected overflow condition)
ARU_DATA(ACB1) = 1
ARU_DATA(ACB0) = not ARU_DATA(ACB0)
else
    Message(No signal level present yet, cannot apply workaround)

Last_CNT = ARU_DATA(CNT)

**GTM_A1.173**  DPLL: new PMT data not received

The root cause of the problem in a dedicated constellation of time is an action calculation with the result “past” although a pending data transfer to the DPLL via ARU with new input data on the same PMT channel cannot be executed. So the data transfer of the new action data starts after the action calculation so that first the action is finished e.g. with the result past before the new input data can be used.

When the DPLL receives PMT requests after a new input slope, only that requests can be considered, which are transferred during a simple ARU routing cycle. The DPLL blocks new PMT requests when there is a time of about 200 ns since the last PMT request is passed. New PMT requests are only accepted after the calculation of the pending action calculations are performed. This calculation starts in step 13 (33) of the state machine, about 10 µs after the input event and ends depending on the number x of actions to be calculated x*3.7 µs later. After this time a single new PMT request is accepted, but there is no possibility to stop an action calculation with an update of data. The “old” value is always calculated.

**Scope**

DPLL.

**Effects**

PMT result calculated on “older” PMT input data because a pending data transfer with newer input data to the DPLL cannot be executed.
Workaround

When the calculated action is transmitted to the MCS check, if there is a ARU transfer with new data of this action was blocked by the ARU, because the DPLL was not ready to receive new data within these increment. Also in the case the ARU transfer was just performed, the corresponding action contains only the “old” PMT requirements. Ignore this action value and wait for the new value which appears about 3.7 µs after the PMT requirement update was transmitted.

New action values relating to new PMT requests are considered in the states 18 to 20 (38 to 40) of the state machine. Typically one PMTR update is transmitted and then corresponding action is calculated until a new PMTR is accepted (when not transmitted in a block with directly succeeding ARU transmissions).

GTM_AI.174 DPLL: PMT result not sent to ARU

The root cause for the problem is that before reaching a dedicated state of the DPLL there is a gap in time in which the DPLL.act_n(i) bit of an action is reset before the act_n_shd_reg signal (shadow register) is set to “1” which starts the transfer of the output data via ARU. If in this gap a new input event is arriving the act_n(i) and the internal state controller changes to the processing of this new input event the signal act_n_shd(i) is not set and so there is no request for transmitting the output data to the ARU placed. This leads to the situation in which an action calculation is finished without transferring the data via the ARU.

PMT calculations were the result is not “PAST” are not affected by this issue. The time frame in which a incoming input signal is causing the misbehavior is about 25 system clock cycles.

Scope

DPLL.

Effects

The results of a PMT calculation are not transferred to their target. This can only happen if the result of the PMT is “PAST”.

**Workaround**

In general a workaround has to take into account that a message ending in “past” is going to have the issue when during action calculation at a dedicated point of time a new input event occurs.

For the MCS program it is therefore necessary that the MCS program is reading the PMT data from DPLL via non blocking ARU reads to prevent that the MCS program is blocked. Additionally, the MCS program should make inside the loop that is doing the non-blocking ARU reads a plausibility check if the requested action is 'out of time' or 'out of angle'.

The MCS can read at any time from TBU the time base tbu_ts0 and the angle from tbu_ts1/tbu_ts2. This information should be used to determine if there is a requested action pending or out of date. If the MCS program did not get back a result in the expected time window, it could either request the old value again or, if the requested event is in the past, request a new value.

Additionally the TIM0 interrupt could be routed to the MCS to check if an active edge occurred. In this case all actions which delivered a result so far must not be checked again independently if their result was “PAST” or not.

If the PMT is used such that the PMT result is transferred directly from the DPLL to the ATOM, there should be a default assignment to the ATOM which is not in PAST to make sure that, even if the DPLL fails to sent the PMT result to the ATOM, the ATOM is not missing an event completely.

---

**GTM_A1.178** **MCS: Evaluation of CAT bit after blocking ARU instruction**

The specification for the instructions ARD, AWR, ARDI, and AWRI claims that the CAT bit can be evaluated by the MCS program in order to check if the last ARU transfer was successful (CAT=0) or cancelled by Software (CAT=1). However, since the CAT bit can be set directly by Software to cancel an ARU transfer at any time the bit does not reflect the status information reliably. Bad case: If the CPU software is setting CAT between the time of ARU data arrival and evaluation of CAT bit.

---

**Scope**

MCS.

---

TC29x, ES-BB, BB 119/291 Rel. 1.6, 2018-02-23
Effects
If the mechanism for cancelling blocking ARU transfers by CPU is used the MCS may signalize an aborted ARU transfer by a set CAT bit although the transfer has finished successfully.

Workaround
If the mechanism for cancelling blocking ARU transfers by CPU is used and data consistency by ARU transfers is important, a possible workaround may check the consistency by inspection of the transferred data (e.g. checking for linear increment of ECNT for data transfers from TIM to MCS).

GTM_AI.181 TIM: Incorrect signal level bit ECNT[0] in mode TIEM, TPWM, TIPM, TPIM, TGPS

In case of re-enabling a previously disabled TIM channel the bit ECNT[0] might not reflect the actual signal level of the corresponding input TIM[i]_CH[x]_FOUT until the next input edge occurs. This situation can only occur if between disabling and re-enabling the ECNT register is not read.

Scope
TIM.

Effects
Inconsistency of input signal level with ECNT bit[0].

Workaround
• After disabling the TIM channel, ensure that the ECNT register is read at least once and afterwards the TIM channel can be re-enabled.
• Before re-enabling a TIM channel, issue a TIM channel reset and reconfigure the TIM channel control registers.
**GTM.AI.202** (A)TOM: no CCU1 interrupt in case of CM1=0 or 1 and RST_CCU0=1

In case of channel x has configuration of RST_CCU0=1 (i.e. CN0 is reset by trigger input) and CN0 counts from 0 to MAX:
- if CM1=0, CM0>0 -> no CCU1 interrupt is generated
- if CM1=1, CM0=MAX+1 -> only one time a CCU1 interrupt is generated

**Scope**
TOM / ATOM SOMP mode.

**Effects**
For the described configuration no CCU1 interrupt is generated.

**Workaround**
Use for triggering channel y (i.e. the channel that triggers on channel x the reset of counter CN0) the configuration of CM0=MAX, CM1=1.
In case of duty cycle configuration of CM1=0 and CM0>0 on channel x use instead of CCU1 interrupt on channel x the CCU0 interrupt of triggering channel y.
In case of duty cycle configuration of CM1=1 and CM0=MAX+1 on channel x use instead of CCU1 interrupt on channel x the CCU1 interrupt of triggering channel y.

**GTM.AI.204** TIM: incorrect signal level on TIM_MODE change if TIM channel is disabled

If TIM_EN=0 and TIM_MODE="100" (TBCM) and corresponding channel input signal is high any write of TIM_MODE!="100" while TIM_EN=0 will not update the signal level bit ECNT[0]. Expected operation is that ECNT[0] will be set to the actual channel input value on TIM_MODE change.

**Scope**
TIM.
Effects
Unexpected signal level.

Workaround
Never set unnecessary TIM_MODE="100" followed by TIM_MODE!="100" while TIM_EN=0.

**GTM_AI.205 TIM: unexpected CNTS register update in TPWM OSM mode**

If OSM=1 and TIM_MODE="000" (TPWM) an active edge defined by DSL will stop the measurement. In case of an inactive edge following after 1 GTM system clock cycle the active edge the CNTS register will be reset unexpected.

Scope
TIM.

Effects
Unexpected CNTS register content.

Workaround
a) Use CMU clock in TIM channel with frequency lesser than system clock.
b) Enable filter and configure filter parameter in a way that two consecutive edges will never occur with distance of GTM system clock.

**GTM_AI.208 DPLL: Start of sub-increment generation and action calculation delayed by one input event if PCM1/2 bits are set and DPLL_STATUS.FTD = '0'**

The DPLL is delaying the start of sub-increment generation and the action calculation by one input event cycle if the DPLL starts after activation (DPLL_CTRL1.DEN= 0 ->1) when the flag DPLL_STATUS.FTD = '0'. In these situations and when additionally PCM1/2 was activated just before or remains
(DPLL_CTRL1.PCM1/2 = '1') the sub-increment generation is starting delayed by one input event cycle.
This results in a wrong state of the TBU_TS1 angle clock. The start of action calculation (PMT) could be delayed by one input event cycle as well.

Scope
DPLL.

Effects
Delayed start of sub-increment generation and action calculation (PMT) by one input event.

Workaround
The issue can happen only when the DPLL starts after activation (DPLL_CTRL1.DEN= 0 ->1) when the Flag DPLL_STATUS.FTD = '0'. In these situations and when additionally PCM1/2 was activated just before (DPLL_CTRL1.PCM1/2 = '1') the DPLL_CTRL1.PCM1/2 bits must be set to '0' before the DPLL is activated again.

**GTM_AL.209**  TOM/ATOM: no update of CM0/CM1/CLK_SRC via trigger signal from preceding instance if selected CMU_CLKx is not SYS_CLK

The trigger signal between (A)TOM instances (e.g. signal TOM_TRIG_[i]) is registered between each TOM and between each 2nd ATOM and with this delayed by one SYS_CLK period to break long combinational path.
For each register in the trigger path between (A)TOM instance i and the succeeding (A)TOM instance i+1, this trigger from instance i does not trigger the update of register CM0, CM1 and CLK_SRC with content of SR0, SR1 and CLK_SRC_SR if the triggered channel of instance i+1 is not running with a selected CMU_CLKx = SYS_CLK.

Scope
TOM/ATOM.
Effects
In the described configuration no update of CM0, CM1 and CLK_SRC is done although the update is enabled by register TOM[i]_TGC[y]_GLB_CTRL / ATOM[i]_AGC_GLB_CTRL.

Workaround
For each register in trigger path between (A)TOM instance i and (A)TOM instance i+1, the channel of instance i+1 that should be triggered has to use a clock of period identical to SYS_CLK period.

A second workaround could be to set up on instance i+1 a redundant channel to trigger other channel of instance i+1 like it was set up on instance i to trigger other channel. Then, start both instances synchronously by using the TBU time base comparator of AGC/TGCx unit (i.e. the ATOM[i]_AGC_ATC_TB / TOM[i]_TGC[y]_ACT_TB register).

GTM_AI.210 ATOM: data loss in SOMS one-shot mode if ARU is enabled and the period of the selected CMU_CLKx is greater than ARU-cycle-time/2

ATOM in SOMS one-shot mode starts to request new data from ARU with ARU_EN = 1. If new data is delivered by ARU and stored into SR0/1 register, the data will be transferred to CM0/1 register and the ATOM starts to shift with next selected CMU_CLKx. In parallel ATOM requests immediately new data from ARU. If ARU will deliver next data before the first bit of the first data is shifted out which means before the next CMU_CLKx takes place, the data will be stored into SR0/1 register but it will not be marked as valid (bit DV not set) and therefore it will be ignored.

Scope
ATOM.

Effects
Delivered data from ARU is not marked as valid (bit DV not set) and will be ignored.
**Workaround**

It has to be ensured, that the time between delivering of two new data from ARU is greater than CMU_CLKx periods. This can be reached by delivering the data by MCS instead of by FIFO.

The issue can only occur if the ARU roundtrip time is greater than 2 CMU_CLKx periods.

**GTM_AI.212  F2A: stream data register will not be deleted after disabling stream**

Disabling a data stream inside the F2A will not delete existing valid data inside F2A. So after re-enabling the disabled stream, F2A will deliver the old data - independent of the configured data transfer direction.

**Scope**

F2A.

**Effects**

Delivering unexpected data by F2A after stream enable.

**Workaround**

Before enabling a data stream, the F2A has to be emptied. After disabling the stream, the ARU read address has to be set to reset value 0x1FE (always empty address). Then the F2A stream has to be configured into the direction ARU to FIFO. After this the stream can be enabled, so that old data will be transported into FIFO. At last the FIFO channel should be flushed.

**GTM_AI.215  FIFO: read pointer will be incremented in ring buffer mode on empty FIFO channel with read access from AFD_CHx_BUF_ACC**

If an empty FIFO channel x is configured into ring buffer mode an then a read access to AFD_CH[x]_BUF_ACC is executed, the read pointer of this FIFO channel x will be incremented.
Scope
FIFO.

Effects
FIFO channel delivers undefined data to ARU.

Workaround
There are 2 possibilities to avoid this erratum:
1. Do not execute a read access to AFD_CH[x]_BUF_ACC after setting the corresponding FIFO channel into ring buffer mode while the FIFO channel is empty.
   In general there are no real application to read a FIFO channel from CPU side (AFD_CH[x]_BUF_ACC) while the FIFO channel is in ring buffer mode.
2. Do not set a FIFO channel into ring buffer mode while the FIFO channel is empty. First fill the FIFO channel and afterwards configure them into ring buffer mode.

GTM_AI.218  DPLL: PWI-IRQ permanently activated

When the DPLL is activated (DPLL_CTRL_1.den= '1') and after that
a) the register DPLL_CTRL_0 is written and
b) the STATE input signals (emergency mode) is activated,
it happens that after the activation of the PWI-IRQ (active input signal event is rejected by negative PVT check) the PWI-IRQ is again and again activated.

Scope
DPLL after reactivation.

Effects
PWI-IRQ permanently activated.
Workaround
The issue can happen only when the DPLL starts after activation (DPLL_CTRL1.DEN= 0 ->1) when the control register DPLL_CTRL_0 is written after that. If this is prevented the issue will not occur.

**GTM_AI.219** DPLL: Wrong internal pointer calculation in case of backwards direction can lead to wrong PMT calculation results (PMT in PAST)

A DPLL internal pointer register is calculated wrong in backwards direction. In this case the PMT calculations leading to wrong results e.g. PMT in “past”. This can only happen in backwards direction.

Scope
PMT computation of GTM/DPLL in backwards direction.

Effects
Wrong PMT computation results by DPLL when DPLL is operating in backwards direction.

Workaround
Combustion engine:
a) don't use PMT calculation in backwards direction.

or

b) If PMT calculations needed even in backwards direction the PMT results must be sent from DPLL to MCS to verify that PMT result is not erroneously in PAST before sent to ATOM.

**GTM_AI.220** DPLL: PVT check is deactivated in case of direction change; Behaviour implemented but not documented in specification so far

The behaviour that the parameter PVT is set to zero after a direction change has occurred is implemented but so far not described in the specification in an adequate manner.
Scope
DPLL-PVT parameter.

Effects
Described and implemented behaviour not documented in specification.

GTM_AI.221  DPLL: Possible inconsistency of internal pointers and parameter NUTE/NUSE when NUTE/NUSE modified in dedicated time window

The parameters NUTE/NUSE are DPLL internally used to modify pointers as well as to decide which data to be used for doing the prediction of the next increment or the selection of the algorithm of PMT calculation to be used. After a new input signal reaches the DPLL either on TRIGGR or STATE processing unit the internal pointers are updated shortly after the TASI/SASI-irq's. If the NUTE/NUSE parameter is changed after that point of time the pointers are not updated until the next input event such that the described inconsistency may occur. This inconsistency may lead to the use of wrong data which can corrupt the results of the increment prediction and the frequency calculation as well as the calculation of PMT.

Scope
DPLL increment prediction and PMT calculation.

Effects
This inconsistency may lead to the use of wrong data which can corrupt the results of the increment prediction and the frequency calculation as well as the calculation of PMT.

Workarround
Modification of NUSE/NUTE, VTN/VSN parameters must be done in uncritical time windows:

a) after new input signal (TIM0_CH0_irq) before TASI/SASI-irq.
b) after PMT calculation has finished for a dedicated increment: e.g. number of active PMT \( n \) that small THVAL > 10µs \( +n*3\mu s \); in this case the parameters NUTE/VTN may be modified after the TISI irq.

**GTM.AI.222** DPLL: TAXI-irq not deactivated for THMA=0

TAXI-irq not deactivated for THMA=0; The internal interrupt signal is not set correctly such that the notify bit of the DPLL_IRQ_NOTIFY.TAXI bit can only be reset if the taxi-irq is internally deactivated with a next input event which does not cause an activation of this interrupt.

**Scope**
DPLL-TAXI-irq.

**Effects**
TAXI-irq is activated even if parameter THMA set to zero.

**Workaround**
Use DPLL_IRQ_EN to deactivate TAXI-irq if not needed.

**GTM.AI.223** DPLL: discontinuities in the sub increments when DPLL_NUTC/S.FST/FSS=1; set to full scale

When the physical deviations are used (DPLL_CTRL_1.AMT/AMS=1) or higher accelerations are happening and at the same time NUTE/FST, NUSE/FSS are set to full scale it happens that the sub increment generation is showing irregular behaviour. This means that the pulse generator frequency is not calculated correctly which ends up in either too fast or too slow generated micro ticks.

**Scope**
DPLL sub increments.
**Effects**

Not well distributed sub increments in between two teeth.

**Workaround**

Don't use DPLL in “full scale” mode, when NUTE/NUSE is set to maximum and FST/FSS is set to one, when stronger accelerations exist or physical deviation with significant deviation is used.

It is possible as well that for the phase of acceleration or the place in the profile, when a physical deviation is relevant for equation DPLL-2c, DPLL_2c1 or DPLL-7c, DPLL-7c1 that just the control bit DPLL_FST/FSS is set to '0'. In this case the error calculation EDT_T, MEDT_T must be observed and checked if not getting too high. If so, this value can be modified via CPU.

**GTM_AI.247  DPLL: Input event not served after DPLL_CTRL_1.DEN is activated**

After the DPLL is enabled by setting DPLL_CTRL_1.DEN = 0 --> 1 there is a time frame in which a new input signal either TRIGGER (i.e. Crank) or STATE (i.e. Cam) is not recognized and not stored.

After power on reset or DPLL software reset this timeframe is about 140 clock cycles.

When the DPLL is enabled after the module was disabled the timeframe is 20 clock cycles for a STATE signal and about 45 clock cycles for a TRIGGER input signal. In case of the TRIGGER input signal the time window can be longer if there are accesses to memory RAM1b in parallel.

Each RAM1b access will lengthen the time window by 10 clock cycles.

**Scope**

DPLL

**Effects**

Input events on TRIGGER/STATE input not served, and synchronization process can take longer.
Workaround

a) Input event will be neglected, DPLL calculations will start with one event delayed.

b) Reenabling of DPLL during operation: Within the time frame after the DPLL is enabled the TIM inputs must be observed if an input event has arrived. To adopt the angle clock the missing pulses must be repeated by the PCM1/2 mechanism.

c) Reenabling of DPLL during operation: Within the time frame after the DPLL is enabled the TIM inputs must be observed if an input event has arrived. Repeat the missing event/pulses by insertion of a TIM input event by writing to configuration register TIM0_IN_SRC.

GTM_AI.250  DPLL: DPLL_STATUS.BWD1/2 not reset after DPLL_CTRL_1.DEN = 1->0->1, when DPLL_CTRL_0 has been written some time before

If the DPLL is disabled and enabled again it happens that the DPLL_STATUS.BWD1/2 flags are not reset.

There are 2 conditions in which the DPLL is disabled for a too short timeframe.

a) If no active input signal is processed in the DPLL at reenabling within a timeframe smaller than ~1.2 µs (@100 MHz GTM clk frequency or 120 system clock cycles) after the register DPLL_CTRL_0 has been written.

b) If an active input signal is processed in the DPLL at reenabling within a timeframe smaller than 8.6 µs (@100 MHz GTM clk frequency or 860 system clock cycles) after the register DPLL_CTRL_0 has been written.

Scope

DPLL

Effects

Incorrect status of DPLL_STATUS.BWD1/2 and wrong angle clock because of opposite direction dependent calculation
Workaround

When the DPLL is disabled there should be at least

a) a time of 1,2 µs or 120 system clock cycles until the DPLL is enabled again (DPLL_CTRL_1.DEN = 1), when no active input signal is processed/expected in this situation.

b) a time of 8,6 µs or 860 system clock cycles until the DPLL is enabled again (DPLL_CTRL_1.DEN = 1), when an active input signal is processed or expected in this situation.

GTM_AI.260  TOM/ATOM: Async. update in SOMP mode with CM1=0 and selected CMU clock unequal sys_clk not functional

An asynchronous update of the duty cycle by writing value 0 to CM1 register while a CMU clock unequal sys_clk is selected is not working. It is expected that the output signal level is set immediately to inactive level but it will remain at actual level.

Scope

TOM/ATOM.

Effects

The output signal level is not set to inactive level. It will remain at actual level.

Workaround

Writing value 1 instead of 0 to CM1 register.

GTM_AI.270 (A)TOM: output signal is postponed one period for the values CM0=1 and CM1>CM0 if CN0 is reset by the trigger of a preceding channel (RST_CCU0=1)

If counter CN0 is reset by the trigger of a preceding channel (bit RST_CCU0 of register TOM[i]_CH[x]_CTRL/ATOM[i]_CH[x]_CTRL is set), then the value of
CM0 defines the signal edge to SL (signal level), whereas CM1 defines the edge to !SL (inverted signal level).

If - in this case - the value 1 is configured for the output edge to SL (CM0=1) and CM1 is configured to greater than CM0 (CM1>CM0) the expected output edge will be postponed by one period.

Scope
TOM, ATOM SOMP mode

Effects
The expected output edge will be postponed by one period.

Workaround
Instead of configuring CM0=1 it is also possible to configure CM1=1 and to invert SL to get the expected edge at counter value 1 (CN0=1).

GTM.AI.271 DPLL: No DCGI-irq after direction change and DPLL_CTRL_0 has been written

If the DPLL is running in normal mode and a direction change is detected after the register dpll_ctrl_0 has been written the DCGI-Interrupt does not occur for following direction changes until both a TRIGGER and a STATE input signal has been arrived at the DPLL inputs.

Scope
DPLL

Effects
DCGI-irq does not occur.

Workaround 1
Don't write to DPLL_CTRL_0 until both an active TRIGGER, STATE input signal has occurred in case of an direction change within this time frame.
How to prevent this, under need of changes of:

- **a)** Need to deactivate Trigger/state input signal (replacing changes on DPLL_CTRL_0.SEN, DPLL_CTRL_0.TEN):
  - Switch according TIM input channels receiving the TRIGGER/STATE input signal by:
  - Modification of TIM[i]_CH[x]_IN:SRC.MODE[i]="10", VAL[i].
- **b)** Need to change from normal mode to emergency mode by change of DPLL_CTRL_0.RMO:
  - If this is necessary the write operation to DPLL_CTRL_0 cannot be prevented.
- **c)** Changes of DPLL_CTRL_0.TNU, SNU. MLT:
  - Such changes should not be necessary, if not the write operation to DPLL_CTRL_0 cannot be prevented.
- **d)** Changes of Adaption modes TRIGGER, STATE (replacing changes of DPLL_CTRL_0.AMT, AMS):
  - Activate/Deactivate AMT, AMS after power up, activate mode by writing adapt data to RAM1c PD(ADT_S) and or RAM2 PD(ADT_T).
- **e)** Changes of Input Delay TRIGGER,STATE (replacing changes of DPLL_CTRL_0.IDT, IDS):
  - Activate/Deactivate TIM[i]_CH[x]_CTRL.FLT_EN to enable or disable filter input data within the dedicated TIM input channel.
- **f)** Changes of DPLL_CTRL_0.IFP:
  - If such changes are necessary the write operation to DPLL_CTRL_0 cannot be prevented.

If Workaround 1 is not doable:

**Workaround 2**

After writing to DPLL_CTRL_0: Check for direction change by evaluating the register DPLL_STATUS.BWD1 when an inactive edge occurred on TRIGGER (TISI-irq) until both an active TRIGGER, STATE input signal has occurred.

The pulse corrections and pointer modifications of the direction change are operated correctly!
GTM_AI.272  DPLL: No update of DPLL_RAM1b.PSTC after direction change and DPLL_CTRL_0 has been written

If the DPLL is running in normal mode and a direction change is detected after the register dpll_ctrl_0 has been written the DPLL_RAM1b.PSTC value is not updated for the following active input signal and keeps the difference for the following input signals.

Scope
DPLL

Effects
Incorrect PSTC value, incorrect PMT/action results. At the tooth with the incorrect PSTC only it can be observed that the subincrements are generated at highest speed.

Workaround 1
Don't write to DPLL_CTRL_0 until both an active TRIGGER, STATE input signal has occurred in case of an direction change within this time frame.

How to prevent this, under need of changes of:

• a) Need to deactivate Trigger/state input signal (replacing changes on DPLL_CTRL_0.SEN, DPLL_CTRL_0.TEN):
  – Switch according TIM input channels receiving the TRIGGER/STATE input signal by:
  – Modification of TIM[i]_CH[x]_IN:SRC.MODE[i]="10", VAL[i].
• b) Need to change from normal mode to emergency mode by change of DPLL_CTRL_0.RMO:
  – If this is necessary the write operation to DPLL_CTRL_0 cannot be prevented.
• c) Changes of DPLL_CTRL_0.TNU, SNU. MLT:
  – Such changes should not be necessary, if not the write operation to DPLL_CTRL_0 cannot be prevented.
• d) Changes of Adaption modes TRIGGER, STATE (replacing changes of DPLL_CTRL_0.AMT, AMS):
– Activate/Deactivate AMT, AMS after power up, activate mode by writing adapt data to RAM1c PD(ADT_S) and or RAM2 PD(ADT_T)

• e) Changes of Input Delay TRIGGER,STATE (replacing changes of DPLL_CTRL_0.IDT, IDS):
  – Activate/Deactivate TIM[i]_CH[x]_CTRL.FLT_EN to enable or disable filter input data within the dedicated TIM input channel.

• f) Changes of DPLL_CTRL_0.IFP:
  – If such changes are necessary the write operation to DPLL_CTRL_0 cannot be prevented.

If Workaround 1 is not doable:

**Workaround 2**

In this case (direction change after DPLL_CTRL_0 has been written before both a TRIGGER and STATE input event has occurred) the PSTC value has to be corrected via CPU access from outside the DPLL.

To achieve this the calculation PSTC_new=PSTC_old +/- nmb_t_tar (+ forward(dir1=0); - backward(dir1=0)) has to be performed and stored to RAM1b.PSTC earlier as 1000 system clock cycles after the active input event, or 850 system clock cycles after the TASI-Irq has occurred.

The PSTC value is internally of the DPLL used for PMT calculations. If no PMT calculation is ongoing in the tooth after direction change and the PSTC value is not needed in GTM external processes the described timing constraint for the PSTC correction can be relaxed until before the next PMT calculations are requested or the PSTC value is needed otherwise.

**GTM_A1.278 FIFO: Restoring of F2A (ARU to FIFO interface) read access to FIFO after GTM_HALT condition not functional**

GTM_HALT is activated while the submodule F2A is executing a read access to a FIFO channel buffer.

Then the F2A read access has to be stopped and restored after GTM_HALT is deactivated.

The restoring of the F2A read access will hand back false data to F2A.
Scope
FIFO

Effects
False data are read from FIFO.

Workaround
No workaround available.

GTM_AI.292  DPLL: pulse correction at direction change incompletely for DPLL_CTRL_1.SMC='1'

Under the assumption of DPLL_CTRL_1.SMC=1 the pulse correction at direction change is done incompletely such that some pulses may be not placed immediately after the direction change. Because the status of the register DPLL_INC_CNT1/2 for automatic end mode (DPLL_CTRL_1.DMO = 0) is correct the pulses can be placed for the next active input signal event.

Scope
DPLL

Effects
Under the assumption of DPLL_CTRL_1.SMC=1 the pulse correction at direction change is done incompletely such that some pulses may be not placed immediately after the direction change.

Workaround
No action, wait for repeating missed pulses in automatic end mode at the next active input event.
GTM_AI.300  DPLL: Change to forward operation when DPLL_THMI is set to zero does not work correctly

If direction control is set up via the TRIGGER input signal (DPLL_CTRL_1.IDDS=0, DPLL_CTRL_1.SMC=0) and DPLL_THMI is set to zero the direction does not change to forward (BWD1=0) when the current direction is backward (BWD1=1). Instead, when DPLL_THMI=0, the direction set latest is hold.

Scope
DPLL

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

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

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

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

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

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

**Scope**
DPLL

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

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

1. Adding an additional input signal (active edge followed by inactive edge while not exceeding the THMI limit) to the trigger input which switches the DPLL back to forward direction.

2. Switching to the direction control mode DPLL_CTRL_1.IDDS=1 and to control the direction by setting the GTM input signal TIM0_IN6 to e.g. zero (forward direction). For combustion engine operation and MAP_CTRL.TSEL=0 the TDIR/SDIR signals can be used to control the direction with the TIM0_IN6 input signal. This TIM0_IN6 signal must be set directly on the GTM input pin by the mechanisms provided by the semiconductor supplier who integrated the GTM. This mechanism is bound to the resource of the TIM0_IN6 input channel.

**GTM_AI.302** DPLL: Pulse generation ongoing for DPLL_CTRL_1.DMO=1 (continuous mode) if DPLL_CTRL_1.sge1/2=0

In continuous mode (DPLL_CTRL_1.DMO=1) the pulse generation cannot be switched off by setting DPLL_CTRL_1.SGE1/2=0. The pulse generation is ongoing independently from the chosen mode (DPLL_CTRL_0.RMO, DPLL_CTRL_1.SMC).

**Scope**

DPLL

**Effects**

Pulse generator cannot be switched off by setting DPLL_CTRL_1.SGE1=0.

**Workarounds**

Set number of pulses to DPLL_CNT_NUM1/2 =0 to suppress pulse generation for DPLL_CTRL_1.DMO=1.
GTM_AI.306  DPLL: DPLL_NUTC.syn_t_old, DPLL_NUSC.syn_s_old not updated according specification

The DPLL specification defines for DPLL_NUTC.WSYN=1 that an update of register DPLL_NUTC allows writing of the bits DPLL_NUTC.syn_t while DPLL_NUTC.syn_t_old inherits the previous value of DPLL_NUTC.syn_t. Differing from the specified behavior the actual hardware does not update the value of DPLL_NUTC.syn_t_old with the previous value of DPLL_NUTC.syn_t but instead updates DPLL_NUTC.syn_t_old according to the corresponding bits of the write operation executed by the CPU.

The DPLL specification defines for DPLL_NUTC.WSYN=1 that an update of register DPLL_NUSC allows writing of the bits DPLL_NUSC.syn_s while DPLL_NUSC.syn_s_old inherits the previous value of DPLL_NUSC.syn_s. Differing from the specified behavior the actual hardware does not update the value of DPLL_NUSC.syn_s_old with the previous value of DPLL_NUSC.syn_s but instead updates DPLL_NUSC.syn_s_old according to the corresponding bits of the write operation executed by the CPU.

Scope
DPLL

Effects
The registers bits DPLL_NUTC.syn_t_old are not updated with the previous value of DPLL_NUTC.syn_t but by the bits of the input data word.
The registers bits DPLL_NUSC.syn_s_old are not updated with the previous value of DPLL_NUSC.syn_s but by the bits of the input data word.

Workaround
If the update of syn_t/s_old shall be done like described in the specification the register DPLL_NU(T/S)C.syn_t/s must be read first, then the DPLL_NU(T/S)C.syn_(t/s) can be used to modify the bits which are written to DPLL_NU(T/S)C.syn_(t/s)_old.
As the current behavior of DPLL_NUT/SC.syn_s/t_old is in use by and can be advantageous for certain applications, there is no intend to change the current
hardware behavior at this point in time. Instead a specification update to align the specification with the current hardware behavior is planned for future GTM generations.

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

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

**Scope**

ATOM

**Effects**

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

**Workaround**

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

**GTM_TC.010** Effects of GTM Resets

The following recommendations to avoid side effects of GTM resets should be considered.

*Note: These effects have not been seen yet in real applications, but have been reported by static timing analysis as potential failure.*
GTM Kernel Reset

The GTM module (including the implementation wrapper) can be reset via software by a kernel reset (bit RST in registers GTM_KRST0/1). Potential side effects are:

• The GTM SRAM contents may be unexpectedly modified.
• False alarms might be generated after restarting the GTM sub-modules (FIFO, DPLL, MCS) as a result of it in case the SRAM contents was not re-initialized.

Workaround

Initialize the GTM SRAMs after a GTM kernel reset is issued by the application. The SRAM contents must not be read by the GTM modules (FIFO, DPLL, MCS) before it is overwritten by initialization (i.e. none of the mentioned GTM modules must be switched on before).

GTM Global Reset

The GTM module (except the implementation wrapper) can be reset via software by a GTM global reset (bit RST in registers GTM_RST). Potential side effects include:

• The GTM SRAM contents may be unexpectedly modified.
• PSI5 functionality might be unreliable while the GTM global reset is performed.
• Starting/ongoing MSC transmissions might be unreliable while the GTM global reset is performed.

Workaround

The GTM global reset must not be used. Use the GTM kernel reset instead.

GTM TC.012 Read Access Control by Register ODA

Specific GTM registers have by default “destructive read” behavior as their normal read behavior (see section “GTM Software Debugger Support” in the GTM chapter of the User’s Manual for further details.)
Depending on the reading master and the configuration of bits DREN and DDREN in register GTM_ODA (OCDS Debug Access Register), the read can be performed “non-destructive” for debug related read operation.

According to the User’s Manual the read is performed “non-destructive” (i.e. debug related read operation)

• for all masters when ODA.DREN = 1B,
• for the Cerberus (OCDS) FPI master when ODA.DREN = 0B and ODA.DDREN = 0B.

Problem Description

In the current implementation the read is performed “non-destructive” (i.e. debug related read operation)

• for all masters when ODA.DREN = 1B,
• for the DMA Partition 2 FPI master when ODA.DREN = 0B and ODA.DDREN = 0B.

Workaround

The problem described above has 2 aspects:

1. For DMA Partition 2 Access to GTM

When the DMA Partition 2 FPI master is used to perform a normal (“destructive”) read of the GTM registers that by default have “destructive read” behavior as their normal read behavior, setting ODA.DREN = 0B and ODA.DDREN = 1B is required to avoid an unintended debug related (“non-destructive”) read access that would be caused by this issue.

2. For Cerberus (OCDS) Access to GTM

When ODA.DREN = 0B and ODA.DDREN = 0B, any read access of the Cerberus (OCDS) FPI master to the registers that by default have “destructive read” behavior as their normal read behavior will cause the normal (“destructive”) read behavior. To get the intended debug related (“non-destructive”) read behavior, ODA.DREN needs to be set to 1B before each access of the Cerberus and set back to 0B afterwards to not affect the access of other FPI masters on the registers described above.
HSCT_TC.007  RX_FIFO overflow interrupt

If a receive path FIFO overflow condition occurs, the corresponding event is not indicated via bit IRQ.SFO (Synchronization FIFO overflow in RX direction), and no interrupt (HSCT Service Request) is generated.

*Note: This interrupt would be an indication about a too slow SRI clock in relation to the Physical layer clock, which results in an overflow situation.*

*(Minimum SRI frequency 40 MHz @ a 320 Mbit baud rate.)*

HSCT_TC.009  Sleep mode not to be used

Due to problems with the wake-up functionality, sleep mode is not to be used.

HSCT_TC.010  Master Mode Interface Test Mode not working

The HSCT Master Mode Interface Test Mode to send out a 101010101.. pattern continuously does not work correctly.

**Workaround**

Do not enable the HSCT Master Mode Interface Test Mode, i.e. leave bit IFCTRL.IFTESTMD = 0B (default after reset). Instead, send out a test pattern under software control.

I2C_TC.001  I2C FIFO data buffer does not support double buffering

Double buffering in the FIFO data buffer allows:

- The TPS value and characters of the next data packet to be written while data transmission on a previous data packet is still ongoing.
- The MPRS value of the next data packet to be written and its characters received into the RXFIFO while the previous data packet is still not completely read out of the RXFIFO.

However in the current implementation, this feature is not supported.
Workaround

For data transmission, the TPS value of the next data packet can be programmed only after the previous data packet has been successfully transmitted and the TXFIFO is empty again. After the new TPS value is programmed, the I2C module generates a data transfer request and the next data transmission will start when new data is available.

Similarly for data reception, the MPRS value of the next data packet can be programmed only after the full reception of the previous data packet. This means after all requests are handled and cleared, and data has been read out of the RXFIFO. The next data reception is then possible.

I2C_TC.003  Limits on selectable INC and DEC values

The two most significant bits of bitfield INC in registers FDIVCFG.[23:22] and FDIVHIGHCFG.[23:22], and the most significant bit of bitfield DEC in register FDIVHIGHCFG.10 cannot be used.

Workaround

When configuring the fractional divider for baud rate generation:

- Use only the least significant six bits of INC; i.e. values equal or less than 63D
- Use only the least significant ten bits of DEC in register FDIVHIGHCFG; i.e. values equal or less than 1023D

Example

The following Table 11 shows some examples for achieved baud rates and their deviation from the target baud rate depending on the settings of INC and DEC in registers FDIVCFG and FDIVHIGHCFG, respectively.
Note: As described in the User’s Manual (section Baudrate Generation), the actual baud rate depends on further factors and differs from these theoretical calculations. The final settings of INC and DEC for a given system should be optimized by measurements.

I2C_TC.004  High speed mode: SCL clock ratio 1:2

The standard for the clock ratio of the I2C high speed mode is 1:2. The ratio is programmed in the DEC/INC bit fields of the FDIVHIGHCFG register.

Recommendation

In order to achieve a 3.4 MHz frequency for high speed mode with the best deviation case for the 1:2 duty cycle ratio, DEC and INC have to be programmed with the following values. Note that not for every $f_{\text{baud1}}$/DEC/INC combination 0% deviation can be achieved.

### Table 11  Baud Rate Deviation depending on INC / DEC

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>100</td>
<td>0.01</td>
<td>$2^D$</td>
<td>997$_D$</td>
<td>100</td>
<td>0.00%</td>
</tr>
<tr>
<td>400</td>
<td>100</td>
<td>0.01</td>
<td>$2^D$</td>
<td>247$_D$</td>
<td>400</td>
<td>0.00%</td>
</tr>
<tr>
<td>3400</td>
<td>100</td>
<td>0.01</td>
<td>$31^D$</td>
<td>170$_D$</td>
<td>3399.1</td>
<td>-0.03%</td>
</tr>
</tbody>
</table>

### Table 12  1:2 Duty Cycle Ratio

<table>
<thead>
<tr>
<th>$f_{\text{baud1}}$ (MHz)</th>
<th>INC</th>
<th>DEC</th>
<th>$f_{\text{SCL}}$ (MHz)</th>
<th>Duty Cycle(%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>85</td>
<td>466</td>
<td>3.4</td>
<td>25.44</td>
</tr>
<tr>
<td>90</td>
<td>1</td>
<td>5</td>
<td>3.33</td>
<td>25.92</td>
</tr>
<tr>
<td>90</td>
<td>85</td>
<td>416</td>
<td>3.4</td>
<td>26.04</td>
</tr>
<tr>
<td>61</td>
<td>5</td>
<td>16</td>
<td>3.39</td>
<td>33.33</td>
</tr>
</tbody>
</table>
I2C TC.005  Hold Time Start violation in Multi-Master Mode

The I2C Standard defines the parameter $t_{HD,STA}$ (Hold Time (repeated) START condition) as 4.0 µs for Standard mode and 0.6 µs for Fast mode. After this period, the first clock pulse is generated.

The parameter $t_{HD,STA}$ is represented in the Infineon TC2xx Data Sheets as $t_7$ (Hold Time for the (repeated) START condition).

In a special situation this design step of the I2C module violates this timing specification by ~30% for Standard mode and by ~18.6% for Fast mode (min. 2.8 µs instead 4.0 µs for Standard mode and min. 0.488 µs instead 0.6 µs for Fast mode).

This occurs when data is written into TX FIFO, the I2C module is ready to start transmission and another master starts driving its startbit shortly before the I2C module starts driving. In this case the I2C Finite State Machine tries to win arbitration by reloading approximately half period in baudrate generator, so next SCL clock edge of the I2C module comes earlier than defined by $t_7$.

IOM TC.002  Missed or spurious IOM events when pulse length exceeds Event Window counter range

When using the Logic Analyzer Module (LAM) of the IOM, if the 24-bit counter for the Event Window exceeds its maximum value (0xFFFFFFFF) it wraps around and starts counting again from 0x0.

If the Event Window is not inverted (LAMCFG.IW = 0B), for example for measuring long pulses, and the edge that generates an event comes after the counter exceeded its maximum value, the event will not be generated if the counter, due to the rollover, is again below the threshold value (LAMEWS.THR), outside of the Event Window.

As an additional side effect of the wraparound, spurious events may be generated when expecting an alarm only in case of pulses that are too short, if a pulse is longer than the counter can handle.

Workaround

Avoid measuring pulses longer than the Event Window counter range.
IOM_TC.003 Unexpected Event upon Kernel Reset

If a kernel reset (via bits RST in registers KRST0/1) is performed on the IOM, an unexpected event may be signalled to the SMU.

**Workaround**

Before triggering a kernel reset via software, set the alarm reaction in SMU to “No Action” to avoid reaction on the unexpected event.

IOM_TC.004 Write to IOM register space when IOM_CLC.RMC > 1

If a clock divider value RMC > 1 is selected in register IOM_CLC, more than one write access may be performed to the IOM register address space within one IOM clock cycle.

This will cause unpredictable effects on the internal state for the following scenarios where two (or even multiples of 2) write accesses are performed within one IOM clock cycle to the following register groups:

- ECM registers ECMCCFG and/or ECMSELr, or
- ECM Event Trigger History registers ECMETH0 and/or ECMETH1, or
- FPC registers FPCESR, FPCCTRk and/or FPCTIMk, or
- LAM registers LAMCFGm and/or LAMEWSm.

*Note:* No problem will occur for read accesses.

**Workaround**

Set IOM_CLC.RMC = 1 when configuring (writing to the registers of) the IOM.

During runtime (not configuring IOM) IOM_CLC.RMC > 1 is not an issue.

miniMCDS_TC.002 Incorrect nested interrupt trace indication

*Note: This problem is only relevant for debugging (tracing).*

The TriCore does not provide an indication that program code of Interrupt Service Routines is executed.
The miniMCDS reconstructs that information. The ISR bit of the TCXDCSTS register respectively of a generated DCU message is set when the TriCore starts to service an interrupt request.

The ISR bit is cleared with the first executed RFE (Return From Exception) instruction.

For nested Interrupts this is incorrect. It should be cleared with the last executed RFE instruction of the nested interrupts.

**Workaround**

No workaround possible.

**MSC_TC.012 Increased Jitter for Data Frame Transmission in Repetition Mode with ABRA**

When the MSC module is configured in repetition mode with ABRA active, command frames can be inserted at any time, starting at equidistant time reference points (TRPs).

The length of a command frame (including passive phase) is defined as follows:

- \( \text{cmd\_len} = (1 + \text{DSC.NBC} + \text{DSTE.PPCE} + 2) \times \text{bit time}^{1) } \)

The time between two TRPs is defined in this context as

- \( \text{tframe\_len} = \text{number of bit times between 2 TRPs} \)

Depending on the relation between cmd\_len and tframe\_len, two specific scenarios can occur that lead to an increased jitter for data frame transmission:

**Scenario 1: cmd\_len = tframe\_len -1**

If the length of the command frame (cmd\_len) is equal to the number of bit times between two TRPs minus 1 (tframe\_len - 1), then the data frame is not started at the next free TRP, but at the TRP when all passive time frames are finished (in equidistant raster).

---

1) where DSC.NBC and DSTE.PPCE is the (decimal) contents of the respective bit fields in the corresponding registers
This means there is an increased jitter for the equidistance of data frame starts, because there is a complete passive time frame sequence (NTPF) without any data frame transmission.

**Scenario 2: cmd_len = tframe_len**

If the length of the command frame (cmd_len) is equal to the number of bit times between two TRPs (tframe_len), then the next data frame is started correctly at the next TRP, but the passive phase of this data frame will erroneously be increased by one bit time, also increasing the jitter for TRPs in repetition mode.

**Workaround**

Avoid the two length configurations for the command frame including its passive phase that are equal to the two scenarios described above:

- cmd_len = tframe_len -1, or
- cmd_len = tframe_len.

**MSC_TC.013 Missing Chip Select for Command Frame with Length zero**

When the MSC module is configured in repetition mode, and the asynchronous ABRA mode is enabled, and the serial clock output FCL is in permanent mode (OCR.CLKCTRL = 1B), then the following problem may occur for command frames with length zero:

If a command frame is started with zero bit length (DSC.NBC = 000000B), then the transmission of one bit (selection bit SEL) is started, and a command interrupt is generated correctly. In some cases, however, the chip select signal for the command frame may not be generated.

**Workaround**

Configure the MSC module such that not all of the following settings are active at the same time:

- DSC.TM = 1B (data repetition mode enabled)
- OCR.CLKCTRL = 1B (FCL always active)
- DSC.NBC = 000000B (0 bits shifted during command frame transmission)
- ABC.ABB = 1B (asynchronous block active, not bypassed)
If one of the conditions listed above is not valid, then the problem cannot occur.

**MSC_TC.014 Upstream Timeout Interrupt cannot be issued at Service Request Output SR4**

When the watchdog timer (defined by USTE.USTOPPRE and USTE.USTOVAL) of the upstream channel is decremented to zero the timeout flag (USTE.USTF) will be set and an interrupt should be issued at one of the 5 service request outputs. The pointer USCE.USTOIP directs the interrupt correctly to the service request lines 0,1,2,3, but if the alternate service request output SR4 is configured (USCE.UTASR = 1B), then erroneously no interrupt occurs if the ABRA overflow interrupt is also directed to SR4 (i.e. ABC.OASR = 1B).

**Workaround**

Do not use the alternate service request output SR4 (USCE.UTASR = 1B) for the upstream timeout interrupt if the ABRA overflow interrupt is also directed to SR4 (i.e. ABC.OASR = 1B).

Instead, select one of the service request outputs SR0..SR3 via the SR multiplexer for the upstream timeout interrupt (USCE.UTASR = 0B).

**MSC_TC.015 Emergency Stop not effective at Injected Bit Positions in Downstream Frame**

Before a data frame on the downstream channel is started, the configured data bits are loaded into the shift register (SRL, SRH).

If an emergency stop condition is active (signal EMGSTOPMSC = 1) during the load operation, all SRL[x] /SRH[y] bits that are enabled for the emergency stop feature in register ESR/ESRE are loaded directly with the corresponding bits of the downstream data registers DD/DDE.

However, the emergency stop feature is not effective for bits that are enabled in addition for an external injection (DSCE.INJENP0/1 =1, position defined by DSCE.INJPOS0/1). This means the injection feature will not be overruled by the emergency stop feature.
Workaround
Do not enable the same data bits with DSCE.INJPOS0/1 for injection and with ESR/ESRE for emergency stop.
Or disable injection via software when an emergency stop condition has occurred.

MSC_TC.016  MSC Spikes on Data and Enable Signals

A potential problem has been identified originating in the MSC ABRA block and involving the strongest output pad driver settings.
The following application settings might lead to spikes visible on the pins:
• ABRA block active
• MPRss, MP+ss or MPss pads (Speed Grade 0) used for the enable signals ENx

Description
The ABRA block of the MSC generates spikes in the serial output data (SO) and enable (EN0 to EN3) signals, that under some worst case conditions might become visible at the output of the corresponding pins.
The spikes would appear as shown in Figure 5, within the normal signal transition timing intervals, as defined with the data sheet timings t44 (SOPx output delay) and t45 (ENx output delay). The spikes would not cause violation of the timings.
Each MSC module instance (MSC0, MSC1 ...) of each device (TC26x, TC27x, TC29x) shows specific behavior regarding the width of the spikes, depending on the on-chip synthesis of the particular instance.

The data signals are sampled with the latching edge, and the spikes may appear around the shift edge. Therefore, this effect is considered to be important only for the enable signals ENx, not for the data signals, therefore in the rest of this document only the ENx signals are being considered. The spikes could appear only on the single ended pads, not on the differential LVDS pads. Additionally, risk exists for MP+ss and MPss pads only for generating positive spikes (00100), not for negative (11011) spikes. For MPRss pads risk exists for both positive and negative spikes.

**Workaround**

- If possible, do not use the ABRA block. If the block is bypassed, there is no risk of spikes
- When using the ABRA block, follow these guidelines:
  - Never use the MPRss (strong sharp) pad setting. These are the fastest pads with the strongest driver setting. MPR pad is used for MSC0 on the port P11.2 for EN1 signal. For this use-case, the risk of spikes is the highest
– Avoid using the MP+ss or MPss pads with MSC instances which are not listed as spike-safe. See the Table 13 below for spike-safe MSC instances

The Table 13 shows the MSC spike-safe and not-safe instances, using the following convention:

- **“safe”** instances are safe for:
  - MP+ss and MPss pad setting with 5V and 3.3V pad supply
- **“3.3V safe”** instances are safe for:
  - all pad types with strong-sharp driver strength supplied with 3.3V, but not with 5V supply, except the MPR pad type.
- **“not safe”** instances are not safe with strong sharp pad setting, neither with 5V, nor with 3.3V supply
- all pad settings slower than strong sharp: strong-medium and medium of all pad types for all MSC instances are always safe, as well as the MP+ss and MPss settings for negative spikes

### Table 13  Spike-Safe MSC Instances for Strong Sharp Pad Setting

<table>
<thead>
<tr>
<th>Device (investigated)</th>
<th>Step</th>
<th>MSC0</th>
<th>MSC1</th>
<th>MSC2</th>
</tr>
</thead>
<tbody>
<tr>
<td>TC26x all devices</td>
<td>BB</td>
<td>safe</td>
<td>3.3V safe</td>
<td>n.a.</td>
</tr>
<tr>
<td>TC27x Standard Variants</td>
<td>BC</td>
<td>not safe</td>
<td>not safe</td>
<td>n.a.</td>
</tr>
<tr>
<td>TC27x ED</td>
<td>BC</td>
<td>not safe</td>
<td>not safe</td>
<td>n.a.</td>
</tr>
<tr>
<td>TC27x Standard Variants</td>
<td>CA</td>
<td>safe</td>
<td>not safe</td>
<td>n.a.</td>
</tr>
<tr>
<td>TC27x ED</td>
<td>CA</td>
<td>safe</td>
<td>3.3V safe</td>
<td>n.a.</td>
</tr>
<tr>
<td>TC27x Standard Variants</td>
<td>DB</td>
<td>3.3V safe</td>
<td>3.3V safe</td>
<td>n.a.</td>
</tr>
<tr>
<td>TC27x ED</td>
<td>DB</td>
<td>not safe</td>
<td>not safe</td>
<td>n.a.</td>
</tr>
<tr>
<td>TC29x Standard Variants</td>
<td>BB</td>
<td>not safe</td>
<td>not safe</td>
<td>not safe</td>
</tr>
<tr>
<td>TC29x ED, ADAS and extended SRAM types</td>
<td>BB</td>
<td>safe</td>
<td>not safe</td>
<td>not safe</td>
</tr>
</tbody>
</table>

### MTU_TC.005  Access to MCx_ECCD and MCx_ETRRi while MBIST disabled

It is possible to access the memory controller registers MCx_ECCD and MCx_ETRRi without the need of the MBIST mode being enabled (i.e. without
MTU_MEMTEST.MEMxEN = 1\textsubscript{B}). This may be used to avoid a complete SRAM initialization on certain security relevant SRAMs.

However, when a MBIST controller is disabled (MTU_MEMTEST.MEMxEN = 0\textsubscript{B}), there is an inevitable corner case that causes the value read/written from/to registers MC\textsubscript{x} ECCD and MC\textsubscript{x} ETRRi of a disabled MBIST controller to be wrong. There is also a possibility that an SPB error is triggered when accessing the MC\textsubscript{x} ECCD and MC\textsubscript{x} ETRRi registers if other masters concurrently use the SPB bus in this situation.

Note: No workaround is required to access the registers of an enabled MBIST controller.

Workaround

When MBIST mode is disabled (MTU_MEMTEST.MEMxEN = 0\textsubscript{B}) for a MBIST controller,

- ensure that the module kernel clock is enabled for the access to MC\textsubscript{x} ECCD and MC\textsubscript{x} ETRRi,
- and perform a dummy write to MC\textsubscript{x} ECCD with value 780FH before any read/write access to MC\textsubscript{x} ECCD or MC\textsubscript{x} ETRRi.

Note: The module kernel clock (of the module in which the SRAM is present) does not need to be enabled if it can be ensured that no concurrent SPB bus accesses by other masters (CPU, DMA, HSM, debugger, ..) to other modules are performed during the MC\textsubscript{x} ECCD/ETRRi access while the module kernel clock is disabled.

The module kernel clock is enabled under the following conditions:

1. For CPU memories, the clock is enabled after reset (for CPU\textsubscript{x} with x>0 even when CPU\textsubscript{x} is still in BOOT-HALT mode), when the CPU is not explicitly put into IDLE mode by software.
2. For SRAMs in peripherals, the module kernel clock is enabled when the module clock is enabled via the CLC register.

The value 780FH has been chosen as an example based on the following use cases and assumptions:

- If error reporting is turned on (i.e. notification enable bits *ENE are set), it does not disturb the system to write back 780FH to register ECCD (write back...
of reset values, write to read-only bits and write of 1\textsubscript{B} to error indication bits has no effect).

- If error reporting is turned off (i.e. notification enable bits \*ENE are cleared), write back of 780F\textsubscript{H} to register ECCD may trigger SMU alarms (if SMU is configured). It is assumed that the corresponding errors are already known by the system since error reporting had previously been deactivated.

**MTU TC.006** Bus error generation upon access to non-existing MBIST registers

Access to non-existing addresses in the MTU Memory Control register block might not deliver a bus error in all cases.

**Example:**

The number of Error Tracking Registers (ETRR) is 1 for all memory controllers, except for Data/Program Scratch Pad RAM (DSPR/PSPR) with 5 ETRR registers and Data/Program Cache Tag SRAM (DTAG/PTAG) with 2 ETRR registers.

This means that for all modules except D/PSPR and D/PTAG, no bus error will be generated for an access to addresses > 0x12 + 0x2 within the respective memory controller.

For D/PSPR, no bus error is generated for an access to addresses > 0x12 + 0xA, and for D/PTAG for addresses > 0x12 + 0x4 within the respective memory controller.

**MTU TC.011** MBIST Bitmap not working for w0 - r1

The simple test case of writing all 0 and checking for 1 should return a full bitmap.

However, in this device step, only one (the last) address of the SRAM is returned.
Workaround
Use the reverse test w1 - r0, which is working as expected and returns the full bitmap.

MTU TC.012 Security of CPU Cache Memories During Runtime is Limited

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

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

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

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

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

MultiCAN.AI.047 Transmit Frame Corruption after Protocol Exception (CAN FD only)

Incorrect behaviour:
It is observed that the MultiCAN+ transmits a corrupted frame after a protocol exception event (An FD enabled receiver shall detect a protocol exception
event when it detects the RES bit to be recessive instead of the expected dominant value).

The transmit frame corruption happens under the following conditions:
A MultiCAN+ node starts transmitting a CAN FD frame and it loses arbitration and hence becomes a receiver. During the reception of RES bit a protocol exception event occurs and the node enters bus integration state as expected. After bus integration is done, the node starts the re-transmission of frame which was cancelled previously because of an arbitration loss. However it is observed that the frame being transmitted is corrupt. The ID as well as the data field will be falsified.

**Correct behaviour:**
In case of a protocol exception event the node becoming the receiver is transmitting the message saved properly.

**Workaround:**
None. The frame can be detected by the fact that the contents of DATAL can be found in the arbitration field.

**OCDS_TC.038 Disconnecting a debugger without device reset ("hot detach") may require reading of OCS registers**

If a debugger disconnects, it should activate at least the Debug Reset. This will reset all the main OCDS resources like CPUs, Cerberus, etc. However for peripherals having a BPI interface, there is the following issue: The Debug Reset is implemented as a synchronous clear on this level. If the OCDS registers are not clocked (e.g. for power saving reasons), the effect of this synchronous clear will be delayed to the next activation of the clock.

In general this will be more a theoretical problem. It’s very unlikely that there is a use case, where a hot detach is required and critical OCDS resources of peripherals were used before. In nearly all cases this effect is invisible for a user, since any register access of the peripheral will generate the clock cycles which are required for the synchronous clear.
Workaround

In case of a hot detach, a tool should - after the Debug Reset activation - read the OCS registers of all peripherals where it used critical OCDS resources. These reads will initiate the required peripheral kernel clocks for the synchronous clear of the OCDS resources.

OCDS TC.040  DAP turn_off to JTAG telegram not working properly

In case of an unsolicited PORST, a JTAG tool will usually not be able to activate the TRST pin as well and by this enable the JTAG interface again after this PORST is released. So the device interface is in DAP mode and not in JTAG mode afterwards.

For this situation the DAP telegram turn_off to JTAG exists, which is JTAG compliant in the sense, that it is a valid and uncritical JTAG TMS control pattern.

The erratum is that the telegram can’t be used directly after a PORST, but requires a dapisc telegram before. This is no problem for DAP tools, but it requires a workaround for JTAG tools.

Please note that this behavior is still much more robust than regular JTAG behavior, where the JTAG operation of the device is completely uncontrolled (e.g. boundary scan enabled), when the tool continues to communicate without noticing that an unsolicited PORST occurred.

Workaround A

Use DAP instead of JTAG.

Workaround B

Recover with another PORST controlled by the JTAG tool.

Workaround C

A fast tool hardware can automatically activate TRST when a PORST is sensed. However this does not work e.g. for the TriBoard since there is a level shifter between OCDS L1 connector and device.
Workaround D

Apply an extended TMS pattern, which includes the dapisc telegram

1. Make sure that the JTAG interface is NOT enabled.
   Trying to read the JTAG ID with the robust sequence JTAG TAP reset followed by a DR scan.
   If the device is in DAP mode this request is ignored because it is an invalid telegram.
2. Make sure that the device is in the DAP state `waiting for start bit`.
   Apply 64 clock cycles with TMS low to potentially drive the DAP module through a CRC error recovery.
3. Apply dapisc telegram with a specific DAPISC.MODE value
   The DAPISC.MODE value is chosen that the reply telegram is output on TDO (DAP2).
   The complete dapisc TMS pattern are the 66 bits 2A4ABBAF530F20C23H, output with LSB first.
   This pattern has to be followed by 30 or more clocks with TMS low (device responds on TDO).
4. Apply turn_off to JTAG telegram
   The complete turn_off to JTAG TMS pattern are the 56 bits FFFF0F8FCEF83FH, output with LSB first.
   After this the JTAG TAP controller is in reset state and JTAG communication can be started.

OCDS_TC.042 OTGS capture registers can miss single clock cycle triggers

The Cerberus OTGS capture registers (TCTL, TCCB, TCCH, TCIP, TCTGB, TCM) can fail to capture a trigger if the trigger is of single clock cycle duration and arrives in the same cycle as the same trigger register is being read by the bus.

Workaround

Avoid polling of OTGS capture registers while the system is running.
If polling while running can't be avoided use TLCCx counters for capturing critical Trigger Lines.

**OCDS_TC.043 Read-Modify-Write Bus Transactions to Cerberus Registers**

During read-modify-write (RMW) bus transactions to writable registers in the Cerberus (CBS), the target register is incorrectly updated with an undefined value during the Read-part. The correct value is always returned to the bus master for the Read-part, and the correct value is written to the register when the Write-part completes. But the register may contain an undefined value for a number of clock cycles between the Read-part and the Write-part.

The bus master (CPU) will see the RMW complete normally, but any logic driven by the hardware register's writable bits may be unexpectedly toggled.

This effects all registers that can be written by the SPB (using the FPI protocol) in the CBS block. It does not effect external access from the tool via JTAG/DAP.

**Workaround**

Do not use RMW bus operations targeting the CBS registers.

**PLL_ERAY_TC.001 PLL_ERAY Initialization after Cold Power-up or Wake-up from Standby mode**

When the PLL_ERAY is configured by the application software after cold power-on reset or wake-up from Standby mode, it may not always reach the intended target frequency (either lock at a lower frequency, or go into unlock state), in particular at high temperature.

**Workaround**

The following code sequence, executed after power-on reset or wake-up from Standby mode and before initializing the PLL_ERAY, avoids the problem:

```
SCU_PLLERAYCON0.B.PLLPWD = 0;  // set PLL_ERAY to power saving mode
wait(10);                      // wait 10µs
```
PLL_TC.005  PLL Initialization after Cold Power-up or Wake-up from Standby mode

When the system PLL is configured by the application software after cold power-on reset or wake-up from Standby mode, it may not always reach the intended target frequency (either lock at a lower frequency, or go into unlock state), in particular at high temperature.

Workaround

The following code sequence, executed after power-on reset or wake-up from Standby mode and before initializing the system PLL, avoids the problem:

```
SCU_CCUCON0.B.CCLKSEL = 0;  // switch system clock to another source different from PLL, e.g. back-up clock
SCU_CCUCON0.B.UP = 1;       // request update
SCU_PLLCON0.B.PLLPWD = 0;   // set PLL to power saving mode
wait(10);                   // wait 10µs
SCU_PLLCON0.B.PLLPWD = 1;   // set PLL to normal behavior
...                        // initialize PLL
```

*Note: For devices with PLL_ERAY, see also problem PLL_ERAY_TC.001*

PORTS_TC.002  Behavior of P21 Port Pins upon Power-on Reset

The following problem affects port pins with LVDSH RX pads. For TC29x, these are P21.[3:0].

As specified, port pins P21.[3:0] are switched to non-LVDS input mode during cold and warm Power-on Reset.

However, in addition the 100 Ohm receiver internal termination between P21.0 and P21.1 or P21.2 and P21.3 is switched on during Power-on Reset.
While no application impact is expected if P21.[3:0] will be used in LVDS mode, this behavior needs to be considered if one or more pins of P21.[3:0] will be used in non-LVDS mode.

If non-LVDS mode with 100 Ohm receiver internal termination is active during Power-on reset, a max. current of 5 mA is allowed without damaging the device (defining the stress for the 100 Ohm receiver internal termination).

*Note:* After power-on reset, the 100 Ohm receiver internal termination is only active if the corresponding bits RX_DIS = 0\(_B\) and TERM = 1\(_B\) in register P21_LPCR0 or P21_LPCR1, respectively.

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

According to the specification, a baud rate error is detected if the incoming shift clock supplied by the master has less than half or more than double the expected baud rate (determined by bit field GLOBALCON.TQ).

However, in this design step, a baud rate error is detected not only if the incoming shift clock has less than half the expected baud rate (as specified), but also already when the incoming shift clock is somewhat (i.e. less than double) higher than the expected baud rate.

In this case, the baud rate error is indicated in the current frame.

**Workaround**

It is recommended not to rely on the baud rate error detection feature, and not to use the corresponding automatic reset enable feature (i.e. keep GLOBALCON.AREN=0\(_B\)).

The baud rate error detection feature in slave mode is of conceptually limited use and is not related to data integrity. Data integrity can be ensured e.g. by parity, CRC, etc., while clocking problems of an AURIX™ master are detected by mechanisms implemented in the master.

Protection against the effects of high frequency glitches is provided by the spike detection feature in slave mode.
RESET_TC.005  Indication of Power Fail Events in SCU_RSTSTAT

In case of consecutive cold resets triggered by EVR13, EVR33 or SWD power fail events, then only the last power fail event is registered in register SCU_RSTSTAT. It is not possible to distinguish individually between EVR13, EVR33 or SWD power fail events from RSTSTAT information.

Workaround

In case any power fail reset indication bit is set among EVR13, EVR33 or SWD power fail events in register SCU_RSTSTAT, it has to be assumed that all power fail events may have happened before.

SMU_TC.005  Unexpected/Incorrect Reset caused by SMU Alarms

A sporadic problem, resulting in generation of unexpected/incorrect resets upon any SMU alarm may occur if at least one alarm is configured to cause a reset.

The probability of occurrence depends on the device version.

*Note: No problem will occur if (at least) one of the following conditions is true:*
- all alarms are configured to cause a reset, i.e. all alarm codes in registers SMU_AGnCFx = 0x6, or
- all alarms are configured to NOT cause a reset, i.e. all alarm codes in registers SMU_AGnCFx ≠ 0x6, or
- the reset request from SMU to SCU is suppressed, i.e. bit field SCU_RSTCON.SMU = 00B, or
- the device is clocked with the back-up clock f_BACK as clock source

Problem Description

The SMU collects all the error flags (called alarms) from all the hardware monitors (safety mechanisms) defined by the safety concept. An internal state machine scans the alarm inputs organized into alarm groups in a defined linear sequence, and triggers the actions (reset, NMI, etc.) selected for each alarm.

If at least one alarm (of any group) is configured to cause a reset (SMU_AGnCFx = 0x6), and at least one other alarm is configured to NOT cause a reset (SMU_AGnCFx ≠ 0x6), it may happen that upon an alarm a
wrong combination of alarm group/index is internally present for less than one clock cycle. If this intermediate group/index combination matches an alarm configured for reset, a spike on the reset request line to the SCU may occur. In case the reset request is not suppressed in the SCU (SCU_RSTCON.SMU ≠ 00B), the following problems may occur, depending on whether the request issued by the SMU to the SCU is configured to cause a System Reset or an Application Reset.

**Case 1: Reset configured as System Reset (SCU_RSTCON.SMU = 01B)**

Upon any SMU alarm, an unexpected System Reset may be performed, although the corresponding active alarm identified during the current alarm scan sequence was configured for a different action. Bit SCU_RSTSTAT.SMU is not set in this case, i.e. this effect can e.g. be used by SW to identify the problematic case after the restart from this System Reset.

The predefined shutdown sequence to achieve a stable state before the reset is internally applied is not correctly executed. Bits CSSx (CPUx Safe State Reached) in register SCU_RSTCON2 keep their old values, i.e. they can not be used to identify this situation.

For further effects resulting from this unexpected reset see section **Additional Effects** below.

**Case 2: Reset configured as Application Reset (SCU_RSTCON.SMU = 10B)**

Upon any alarm, the configured action (e.g. NMI) is performed correctly, although the wrong group/index combination (causing the spike) was temporarily present. In contrast to Case 1, no unexpected (Application) Reset is executed, and bit SCU_RSTSTAT.SMU will not be set in this case.

However, upon any following event (from SMU or other source) configured to cause an Application/System or warm Power-on Reset, the predefined shutdown sequence to achieve a stable state before the reset is internally applied is not correctly executed. Bits SCU_RSTCON2.CSSx of not halted CPUs are cleared to 0B (CPUx safe state not achieved prior to last reset). This effect can e.g. be used by SW to identify the problematic case after restart from this reset.
For further effects resulting from this (intended) reset see section Additional Effects below.

Additional Effects

• As the shutdown sequence to sequentially bring the CPUs into halt state is not executed properly prior to the reset, the difference in $I_{DD}$ supply current between active and reset state (“current jump”) is increased as shown in the following table:

Table 14  $I_{DD}$ Current Difference caused by Reset [values in mA]

<table>
<thead>
<tr>
<th>Device (real / max power pattern)</th>
<th>Step</th>
<th>$I_{DD}$</th>
<th>$I_{DPPORST}$ (150°C)</th>
<th>Largest $I_{DD}$ Jump without correct Shutdown Sequence</th>
<th>Largest $I_{DD}$ Jump with correct Shutdown Sequence</th>
</tr>
</thead>
<tbody>
<tr>
<td>TC29xED (real)</td>
<td>BB</td>
<td>656</td>
<td>290</td>
<td>366</td>
<td>~220</td>
</tr>
<tr>
<td>TC29xED (max)</td>
<td>BB</td>
<td>900</td>
<td>290</td>
<td><a href="mailto:610@1.43V">610@1.43V</a>, <a href="mailto:630@1.485V">630@1.485V</a></td>
<td>~270</td>
</tr>
<tr>
<td>TC29x (real)</td>
<td>BB</td>
<td>567</td>
<td>220</td>
<td>347</td>
<td>~200</td>
</tr>
<tr>
<td>TC29x (max)</td>
<td>BB</td>
<td>800</td>
<td>220</td>
<td><a href="mailto:580@1.43V">580@1.43V</a>, <a href="mailto:600@1.485V">600@1.485V</a></td>
<td>~240</td>
</tr>
</tbody>
</table>

1) Values for $I_{DD}$ and $I_{DPPORST}$ are taken from corresponding product Data Sheet. For ED devices, values are calculated for use case without MCDS.
2) Calculated from $I_{DD} - I_{DPPORST}$
3) Typical value

• **Recommendations:**
  - In case external core voltage ($V_{DD}$) regulator is used, the corresponding load transient response shall be limited within the bounds of operating conditions. In case of overshoots beyond operating conditions, it should be ensured that absolute maximum ratings are met.
  - In case internal EVR13 core voltage regulator is used, the corresponding SMPS regulator load transient response (see specification of $dV_{out}/dl_{out}$ in Data Sheet) shall be considered.
• The duration of the ESR0 pulse to indicate the reset may be shortened to ~2 µs upon Application Reset / ~13 µs upon System Reset in default configuration.
  – To compensate for this effect, increase the value of the Flash Config Sector setting “ESR0CNT” which is copied by Boot Code into FLASH0_PROCOND.ESR0CNT on a cold power-on reset. Increasing the value for ESR0CNT by 0x8 extends the ESR0 pulse by 8 * 10 µs.

• In case a Data Flash (EEPROM) write access is active when the reset is finally executed, data integrity for the selected word line cannot be guaranteed.
  – In general, robust adequate counter measures should be implemented in the EEPROM emulation software, e.g. as in Infineon’s MCAL FEE driver.

• In case an SRAM write access is active when the reset is finally executed:
  – Multi-cycle or unaligned write accesses to SRAMs may not be completed, i.e. they may only write part of their data. **Note:** In almost all cases a mis-aligned 32-bit write to local DSPR is still a single-cycle access. The only exception is the case where a mis-aligned access using Circular Addressing mode wraps at the limit (top/bottom) of the circular buffer. A 32-bit access to SRI is single-cycle if naturally aligned, multi-cycle if mis-aligned.
    A 64-bit write to local DSPR is usually a single-cycle access. Exception is with circular addressing mode wrapping as above. A 64-bit access to SRI is single-cycle if naturally aligned, multi-cycle if mis-aligned or targeting the FPI.
  – Write bursts on busses may transfer only part of their data. **Note:** For the CPUs burst writes occur as a result of one of the following: - Certain mis-aligned 64-bit writes (others result in sequence of single transactions)
    - CSA related instructions / events: CALL{A,I}, FCALL{A,I}, SVLCX, BISR, Interrupts, Traps (local DSPR only allowed for TC1.6E, local DSPR or bus for TC1.6P)
    - Context Store instructions: STLCX, STUCX (local DSPR only allowed for TC1.6E, local DSPR or bus for TC1.6P)
- Cache line writeback to SRI for TC1.6P if cache line contains dirty data and is evicted from cache.
  - Example: In case of a 32-bit write that is 16-bit aligned only the first 16 bits might be written before the reset hits. Post-reset reads from this (unaligned) address will read scrambled 32-bit data (16-bit old, 16-bit new).
  
  Note: Data integrity for single-cycle write accesses is not affected!

Example for Critical Scenario

The problem has been reproduced with the following test scenario:

- **Preconditions:**
  - Configure Recovery Timer 1 alarm (AG2[30]) to generate a reset
  - Configure SCU to generate System Reset upon SMU reset request
  - Configure AG2[21] (EDC Address phase error) and AG3[30] (SRI Bus error event) to generate e.g. NMI or No Action
  - Force an SRI address error (set CPUx_SEGEN = 0x80000001, read from LMU (0xB0000000))

- **Expected result:**
  - Alarm AG2[21] (EDC Address phase error) and AG3[30] (SRI Bus error event) are raised almost in parallel
  - Alarm scanning state machine writes AG2[21] and AG3[30] to group/index registers to activate their configured action.

- **Error situation:**
  - AG3[30] is identified first and group 3, index 30 are written to internal registers
  - AG2[21] is next identified alarm and shall be written as next pair.
  - When writing this second pair of group/index, the combination AG2[30] (i.e. new group, but old index value!) appears for short time and triggers the alarm action for this combination AG2[30] =>> unexpected reset trigger generated, leading to incorrect System Reset.

**Workaround 1**

Configure all alarms that should cause a reset to cause an NMI instead. In the NMI trap routine, software may trigger the intended reset via register
SCU_SWRSTCON. The reset type (System or Application Reset) is selected in bit field SCU_RSTCON.SW.

Workaround 2
Configure all alarms that should cause a reset to trigger the FSP Error Pin instead. The signal on this pin may be used to directly assert an ESRx/PORST reset.

Workaround 3
If an incorrect reset has been identified - either via bit SCU_RSTSTAT.SMU (case 1) or bits SCU_RSTCON2.CSSx (case 2) - appropriate measures have to be taken to ensure memory integrity. E.g. perform a system initialization as after a cold power-on reset.

SMU_TC.006 OCDS Trigger Bus OTGB during Application Reset
The SMU provides an alarm trigger and trace interface (Trigger Set TS16_SMU) using the OCDS Trigger Bus OTGB.
While the Application Reset is active, the SMU outputs the reset state of the OTGB interface instead of TS16_SMU.
This OTGB interface reset state is identical to TS16_SMU when no alarm is active.
After the Application Reset TS16_SMU is output again.

Workaround
Just ignore the phase in the OTGB trace where an alarm seems to become inactive while the Application Reset is active.

SMU_TC.007 Size and Position of Field ACNT in Register SMU_AFCNT
Note: This erratum might affect the SFR C Header Definitions. In such cases, SFR usage in the software shall be analyzed within the applications for their correct handling.

TC29x, ES-BB, BB 170/291 Rel. 1.6, 2018-02-23
In the SMU chapter of the User’s Manual, in the description of register SMU_AFCNT (Alarm and Fault Counter),

- Size and position of field ACNT (Alarm Counter) are incorrectly described as SMU_AFCNT.[15:8], and
- Bits SMU_AFCNT.[7:4] are incorrectly shown as “Reserved; read as 0”.

The correct size and position of field ACNT (Alarm Counter) in register SMU_AFCNT is SMU_AFCNT.[15:4], as shown in the following Table 15. The position of the “Reserved” bits is aligned accordingly.

### Table 15  Field ACNT in Register SMU_AFCNT - Correction

<table>
<thead>
<tr>
<th>Field</th>
<th>Bits</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACNT</td>
<td>[15:4]</td>
<td>rh</td>
<td>Alarm Counter</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>This field is incremented by hardware when the SMU processes an internal action related to an alarm event (see Figure “Alarm operation”). The counter value holds if the maximum value is reached.</td>
</tr>
<tr>
<td>0</td>
<td>[29:16]</td>
<td>r</td>
<td>Reserved</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Read as 0; should be written with 0.</td>
</tr>
</tbody>
</table>

Note: The other fields (ACO, FCO, FCNT) of register SMU_AFCNT are correctly described in the User’s Manual.

### SMU_TC.008  Behavior of Action Counter ACNT

Register SMU_AFCNT (Alarm and Fault Counter) implements a Fault Counter (FCNT) that counts the number of transitions from the RUN state to the FAULT state. Register AFCNT is only reset by a power-on-reset.

Whenever a pending alarm event is processed, the corresponding status bit is set to $1_B$ by hardware in the Alarm Status register AG<$x$>.

If an internal SMU action is configured for this alarm, the Action Counter (ACNT) in register AFCNT is incremented anytime the SMU processes this internal action.
Corner Case
In this device step, some of the alarm signals may increment the Action Counter ACNT multiple times for a single alarm event.

Workaround
Do not rely on the value in the action counter ACNT.

SMU TC.010 Transfer to SMU_AD register not triggered correctly

Background
The SMU contains Alarm Debug registers which can be used for diagnostic purposes. If an alarm which is configured to generate a reset (application or system reset) is sent to the SMU, a copy of the Alarm Status registers – AGi – into the Alarm Debug registers – ADi – is automatically triggered. The AGi are reset by Application reset while the ADi are reset only by power-on reset.

Corner Case
In the case that a first SMU alarm AGi[j] generates a reset request, and a second alarm AGx[y] (where x=i and y=j is possible) configured for a reset occurs a few cycles before the reset is actually executed, then the reset values of the AGi registers will be transferred to the ADi register. In this case, the ADi registers will not reflect the root cause that lead to a SMU alarm/reset.

Note: This corner case will always be met for level alarms.
3 Deviations from Electrical- and Timing Specification

ADC_TC.P007 Additional Parameter for Data Sheet: Wakeup Time $t_{\text{WU}}$

As mentioned in section “Wakeup Time from Analog Powerdown” of the VADC chapter in the User’s Manual, when the converter is activated, it needs a certain wakeup time to settle before a conversion can be properly executed.

In the Data Sheet (section VADC) the corresponding parameter *Wakeup Time* is missing:

Table 16 Wakeup Time - Addendum to Tables VADC, VADC_33

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Values</th>
<th>Unit</th>
<th>Note / Test Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Min.</td>
<td>Typ.</td>
<td>Max.</td>
</tr>
<tr>
<td>Wakeup Time</td>
<td>$t_{\text{WU}}$</td>
<td>-</td>
<td>-</td>
<td>12 µs</td>
</tr>
</tbody>
</table>

ADC_TC.P011 Leakage current for ADC reference pins VAREF, VAGND

The values of the leakage current for the VADC reference pins ($I_{\text{OZ2}}$ at VAREF and $I_{\text{OZ3}}$ at VAGND) need to be slightly corrected as shown in the tables below (changes marked in **bold**).

As a result, the adjusted numbers provide a coherent picture for the members of the AURIX™ family while ensuring a stable production.
## Table 17: Leakage Current for TC29x VADC Reference Pins - $V_{DDM} = 4.5$ V to 5.5 V

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Values</th>
<th>Unit</th>
<th>Note / Test Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Positive reference</td>
<td>$I_{OZ2}$ CC</td>
<td>-7</td>
<td>7 μA</td>
<td>$V_{AREF} &gt; V_{DDM}$; $T_J &gt; 150°C$</td>
</tr>
<tr>
<td>VAREF2 pin leakage</td>
<td></td>
<td>-4</td>
<td>4 μA</td>
<td>$V_{AREF} &gt; V_{DDM}$; $T_J \leq 150°C$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-1 instead of -2</td>
<td>4 μA</td>
<td>$V_{AREF} \leq V_{DDM}$; $T_J &gt; 150°C$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-1 instead of -2</td>
<td>2 μA</td>
<td>$V_{AREF} \leq V_{DDM}$; $T_J \leq 150°C$</td>
</tr>
<tr>
<td>Negative reference</td>
<td>$I_{OZ3}$ CC</td>
<td>-17 instead of -13</td>
<td>7 μA</td>
<td>$V_{AGND} &lt; V_{SSM}$; $T_J &gt; 150°C$</td>
</tr>
<tr>
<td>VAGND2 pin leakage</td>
<td></td>
<td>-8 instead of -7</td>
<td>7 μA</td>
<td>$V_{AGND} &lt; V_{SSM}$; $T_J \leq 150°C$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-5 instead of -4.5</td>
<td>1 μA</td>
<td>$V_{AGND} \geq V_{SSM}$; $T_J &gt; 150°C$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-3 instead of -2.5</td>
<td>1 μA</td>
<td>$V_{AGND} \geq V_{SSM}$; $T_J \leq 150°C$</td>
</tr>
</tbody>
</table>

1) For TC290 (bare die version), the VADC positive reference VAREF leakage current is the sum of the leakage currents on pads VAREF2, VAREF3, VAREF4.

2) For TC290 (bare die version), the VADC negative reference VAGND leakage current is the sum of the leakage currents on pads VAGND2, VAGND3, VAGND4.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Values</th>
<th>Unit</th>
<th>Note / Test Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Min.</td>
<td>Typ.</td>
<td>Max.</td>
</tr>
<tr>
<td>Positive reference VAREF2 pin leakage(^1)</td>
<td>(I_{OZ2}) CC</td>
<td>-2</td>
<td>-</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-3.5</td>
<td>-</td>
<td>3.5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-1</td>
<td>-</td>
<td>3</td>
</tr>
<tr>
<td>Negative reference VAGND2 pin leakage(^2)</td>
<td>(I_{OZ3}) CC</td>
<td>-12</td>
<td>-</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-6.5</td>
<td>-</td>
<td>6.5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-3.5</td>
<td>-</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-2</td>
<td>-</td>
<td>1</td>
</tr>
</tbody>
</table>

1) For TC290 (bare die version), the VADC positive reference VAREF leakage current is the sum of the leakage currents on pads VAREF2, VAREF3, VAREF4.

2) For TC290 (bare die version), the VADC negative reference VAGND leakage current is the sum of the leakage currents on pads VAGND2, VAGND3, VAGND4.
CHIP TC.P001 TC290 Chip Dimensions

The actual chip dimensions A and B are slightly smaller than listed in table “TC290 Chip Dimensions” of the Data Sheet. The updated values are listed in Table 19 below.

Table 19 TC290 Chip Dimensions

<table>
<thead>
<tr>
<th>Device</th>
<th>A</th>
<th>B</th>
<th>T</th>
</tr>
</thead>
<tbody>
<tr>
<td>TC290</td>
<td>8.770 mm</td>
<td>9.357 mm</td>
<td>0.3 mm</td>
</tr>
</tbody>
</table>

Note: The X/Y coordinates are not affected, they are correctly listed in table “TC29x Bare Die Pad List” of the Data Sheet.

EBU TC.P002 EBU Timing Deviations

The actual EBU timings partially deviate from the values listed in the Data Sheet. The deviating parameters and values are summarized and marked in bold in Table 20 .. Table 23.

Table 20 Deviations of Common Asynchronous Timings

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Values</th>
<th>Unit</th>
<th>Note / Test Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Min.</td>
<td>Typ.</td>
<td>Max.</td>
</tr>
<tr>
<td>AD(31:0), deviation of address output valid time from the ideal programmed value</td>
<td>$t_{13} \text{ CC}$</td>
<td>1.9</td>
<td>-</td>
<td>19.8</td>
</tr>
<tr>
<td>AD(31:0) output delay to ADV# rising edge, multiplexed read / write</td>
<td>$t_{14} \text{ CC}$</td>
<td>1.9</td>
<td>-</td>
<td>19.8</td>
</tr>
</tbody>
</table>
### Deviations of Common Asynchronous Timings (cont’d)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Values</th>
<th>Unit</th>
<th>Note / Test Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pulse width deviation from the ideal programmed width due to pad asymmetry, rise delay - fall delay</td>
<td>$t_a$ CC</td>
<td>-1.8</td>
<td>0.3</td>
<td>edge = medium</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-1.0</td>
<td>0.6</td>
<td>edge = sharp</td>
</tr>
</tbody>
</table>

### Deviations of Asynchronous Read Timings

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Values</th>
<th>Unit</th>
<th>Note / Test Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data input Setup to CS rising edge</td>
<td>$t_{19}$ CC</td>
<td>14.6</td>
<td>-</td>
<td>ns</td>
</tr>
<tr>
<td>ADV rising edge to RD rising edge, deviation from the ideal programmed value</td>
<td>$t_3$ CC</td>
<td>-1.7</td>
<td>-</td>
<td>ns</td>
</tr>
<tr>
<td>WAIT input setup to RD rising edge</td>
<td>$t_5$ SR</td>
<td>13</td>
<td>-</td>
<td>ns</td>
</tr>
<tr>
<td>Data input setup to RD rising edge</td>
<td>$t_7$ SR</td>
<td>15</td>
<td>-</td>
<td>ns</td>
</tr>
</tbody>
</table>
# Deviations of Asynchronous Write Timings

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Values</th>
<th>Unit</th>
<th>Note / Test Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS rising edge to WR rising edge, deviation from the ideal programmed value</td>
<td>$t_{32}$ CC</td>
<td>-2</td>
<td>-</td>
<td>2.1</td>
</tr>
<tr>
<td>WAIT input setup to WR rising edge, deviation from the ideal programmed value</td>
<td>$t_{35}$ SR</td>
<td>13</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Write data output hold after WR rising edge, deviation from the ideal programmed value</td>
<td>$t_{38}$ CC</td>
<td>-</td>
<td>-</td>
<td>20</td>
</tr>
<tr>
<td>MR / W output delay to WR rising edge, deviation from the ideal programmed value</td>
<td>$t_{39}$ CC</td>
<td>- removed (signal not existing on TC29x)</td>
<td>-</td>
<td></td>
</tr>
</tbody>
</table>
Table 23  Deviations of Burst Read Timings

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Values</th>
<th>Unit</th>
<th>Note / Test Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Output delay from BFCLKO rising edge</td>
<td>$t_{10}$ CC</td>
<td>-2</td>
<td>Typ.</td>
<td>2.1 ns</td>
</tr>
<tr>
<td>D(31:0) Output delay from BFCLKO rising edge</td>
<td>$t_{10a}$ CC</td>
<td>-2</td>
<td></td>
<td>20 ns</td>
</tr>
<tr>
<td>CSx output delay from BFCLKO active edge</td>
<td>$t_{21}$ CC</td>
<td>-2.5</td>
<td></td>
<td>1.9 ns</td>
</tr>
<tr>
<td>Data setup to BFCLKI rising edge</td>
<td>$t_{23}$ SR</td>
<td>4</td>
<td></td>
<td>ns</td>
</tr>
</tbody>
</table>

**FlexRay TC.P002**  Pad Configuration for E-Ray Parameters

The following sentence is missing at the beginning of section “E-Ray Parameters” in the Data Sheet:

“The timings of this section are valid for the strong driver sharp edge settings (speed grade 1) of the output drivers with $C_L = 25\ pF$. For the inputs the hysteresis has to be configured to inactive.”

**IDD TC.H001**  IPC Limits used in Production Test for IDD Max Power Pattern

Instructions per cycle for a CPU is measured by dividing ICNT instruction counter value with the CCNT clock counter value.

*Note: For a complete description of registers ICNT and CCNT refer to the TriCore Architecture Manual, chapter “Performance Counter Registers”.*
Parameters using the max power pattern for device individual testing of power consumption limits (IDD) are tested for a maximum IPC rate of 1.2 for all CPUs available in the device.

**PADS_TC.P002 Restrictions for P00.1 .. P00.12 if $V_{DDM}$ is lower than $V_{EXT}$**

Each input pin of the AURIX™ devices is equipped with ESD protection circuitry.

For the mixed analog/digital signal pins P00.1 .. P00.12, there is both ESD protection of the digital part to $V_{EXT}$, and ESD protection of the analog VADC inputs to $V_{DDM}$. P00.1, P00.2, P00.7 and P00.8 have additional ESD protection to $V_{DDM}$ for the analog DSADC inputs.

In case $V_{DDM}$ is lower than $V_{EXT}$ (e.g. $V_{DDM} = 3.3\ \text{V}$ and $V_{EXT} = 5.0\ \text{V}$), an increased cross current can be observed if the input or output voltage on these pins is above $V_{DDM}$, e.g. if:

- (one or more of) the internal pull-ups on P00.1 .. P00.12 are enabled, or
- (one or more of) the pins P00.1 .. P00.12 are driven high as output, or
- the input voltage on (one or more of) the pins P00.1 .. P00.12 is above $V_{DDM}$.

The cross current observed on P00.1, P00.2, P00.7 and P00.8 is higher than on other P00 pins due to the dual ESD protection structure (VADC and DSADC inputs).

*Note: This effect does not occur if $V_{EXT}$ is lower than $V_{DDM}$.***

**Workaround**

Design the system such that $V_{DDM} \geq V_{EXT}$.

Otherwise, ensure that the (input or output) voltage on P00.1 .. P00.12 is lower or equal to $V_{DDM}$, and additionally disable pull-ups on P00.1 .. P00.12, and do not drive P00.1 .. P00.12 as (push/pull) outputs.

**PADS_TC.P003 Input Frequency $f_{IN}$ for Class S Pads**

For the class S pads parameter “Input frequency” (symbol $f_{IN}$), only the “Hysteresis active” mode is available.
The “Hysteresis inactive” mode is not available for this pad type, therefore the corresponding row in the Data Sheet with “Hysteresis inactive” in column “Note/Test Condition” for $f_{IN}$ does not apply for this pad type.

**PADS TC.P004 Input Low / High Voltage for LP Pad $V_{ILHLP}$ - Condition**

For the class LP (5V / 3.3V) pads parameter “Input low / high voltage for LP pad” (symbol $V_{ILHLP}$) the condition should read:

- Hysteresis inactive; not available for P14.2, P14.4, P15.1, P15.10, and P15.11

i.e. P15.10 and P15.11 have to be added to this list in the Data Sheet.

**PADS TC.P006 P21.6/P21.7 Pull-up Reset Behavior**

In Table “Port 21 Functions” in the Data Sheet, the pull-up behavior of P21.6 and P21.7 is defined with symbol PU in column “Type”, and in Table “Pull-up/Pull-Down Reset Behavior of the Pins” in the Data Sheet, the behavior of TDI (P21.6) and TDO (P21.7) is described to be independent of the state of HWCFG[6].

However, the actual pull-up behavior for P21.6 and P21.7 is as defined for symbol PU1, i.e.

- P21.6 (TDI) and P21.7 (TDO) are pulled up during and after PORST if pin HWCFG[6] (P14.4) is pulled high or left unconnected,
- P21.6 (TDI) and P21.7 (TDO) are High-Z during and after PORST if pin HWCFG[6] (P14.4) is pulled low.

**PADS TC.P009 Bonding of VGATE1P on Bare Die Variants**

*Note: This information is only relevant for the TC290 Bare Die variant.*

On packaged devices, pads VGATE1P (SMPS) and VGATE1P (LDO) are internally connected together, ensuring identical levels on both pads.
On the bare die variant, these pads should consequently be bonded to the same level, depending on the selected supply configuration.

The recommended connections of the supply pads for the individual supply options are summarized in table “Supply Mode and Topology selection” in the User’s Manual and described in detail therein.

The documentation in table “TC29x Bare Die Pad List” of the Data Sheet has to be updated for pads VGATE1P (SMPS) and VGATE1P (LDO) as shown in the following table.

Table 24 TC290 VGATE1P Pads - Documentation Update

<table>
<thead>
<tr>
<th>Number</th>
<th>Pad Name</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>310</td>
<td>VGATE1P (SMPS)</td>
<td>Connect to same level as VGATE1P (LDO).</td>
</tr>
<tr>
<td>311</td>
<td>VGATE1P (LDO)</td>
<td>Connect to same level as VGATE1P (SMPS).</td>
</tr>
</tbody>
</table>

PWR_TC.P013 EVR Supply Voltage $V_{\text{EXT}}$ Ramp-up

The device may not start in case of specific “non-monotonous” EVR supply voltage ($V_{\text{EXT}}$) ramp-up scenarios slower than 5V/10ms if the internal EVR33 voltage regulator is used. Characteristics of these scenarios are (one or more) plateaus of $> 100 \mu s$ in the range of $2.65 \text{ V} < V_{\text{EXT}} < 2.85 \text{ V}$ at the $V_{\text{EXT}}$ pins (see Figure 6).

In fail case, the start-up sequence is stopped, the device remains in cold PORST reset state and the internal EVR33 regulator may not start though activated via HWCFG pins. The device in fail state does not recover after a PORST assertion, but a renewed power-off/on cycle is required.

Note:

- The fail behavior is observed only if the internal EVR33 regulator is used.
- The fail behavior is not observed together with TLF35584 regulator (owing to fast ramp-up slopes).
**Workaround**

Ensure that the minimum slope for the EVR Supply Voltage (VEXT) is faster than 5V/10ms (see also Data Sheet, symbol dVin/dT).

*Note: Power Sequencing rules for the individual supply voltages remain unchanged and need to be considered as described in the Data Sheet.*

**Additional Information**

- The startup issue is seen to happen only when there is at least one stationary non-monotonous step plateau greater than 100 µs (without buffer capacitors), with leading/trailing steps of >15 mV amplitude each, in the critical range as shown in **Figure 6**.
- The fail behavior is observed mainly at cold temperatures up to room temperature.
- This issue is not confirmed by any observation in field.
4 Application Hints

ADC_A1.H003 Injected conversion may be performed with sample time of aborted conversion

For specific timing conditions and configuration parameters, a higher prioritized conversion $c_i$ (including a synchronized request from another ADC kernel) in cancel-inject-repeat mode may erroneously be performed with the sample time parameters of the lower prioritized cancelled conversion $c_c$. This can lead to wrong sample results (depending on the source impedance), and may also shift the starting point of following conversions.

The conditions for this behavior are as follows (all 3 conditions must be met):

1. **Sample Time setting**: injected conversion $c_i$ and cancelled conversion $c_c$ use different sample time settings, i.e. bit fields $STC^*$ in the corresponding Input Class Registers for $c_c$ and for $c_i$ ($GxICLASS0/1$, $GLOBICLASS0/1$) are programmed to different values.
2. **Timing condition**: conversion $c_i$ starts during the first $f_{ADCI}$ clock cycle of the sample phase of $c_c$.
3. **Configuration parameters**: the ratio between the analog clock $f_{ADCI}$ and the arbiter speed is as follows:
   \[ N_A > N_D \times (N_{AR} + 3), \]
   with
   a) $N_A = \text{ratio } f_{ADC}/f_{ADCI}$ ($N_A = 1 \ldots 32$, as defined in bit field $DIVA$),
   b) $N_D = \text{ratio } f_{ADC}/f_{ADCD} = \text{number of } f_{ADC} \text{ clock cycles per arbitration slot}$ ($N_D = 1 \ldots 4$, as defined in bit field $DIVD$),
   c) $N_{AR} = \text{number of arbitration slots per arbitration round}$ ($N_{AR} = 4, 8, 16, \text{ or } 20$, as defined in bit field $GxARBCFG.ARBND$).

Bit fields $DIVA$ and $DIVD$ mentioned above are located in register $GLOBCFG$.

As can be seen from the formula above, a problem typically only occurs when the arbiter is running at maximum speed, and a divider $N_A > 7$ is selected to obtain $f_{ADCI}$. 
**Recommendation 1**
Select the same sample time for injected conversions $c_i$ and potentially cancelled conversions $c_c$, i.e. program all bit fields $STC^*$ in the corresponding Input Class Registers for $c_c$ and for $c_i$ ($GxICLASS0/1, GLOBICLASS0/1$) to the same value.

**Recommendation 2**
Select the parameters in register $GLOBCFG$ and $GxARBCFG$ according to the following relation:

$$N_A \leq N_D(N_{AR}+3).$$

**ADC_TC.H014  VADC Start-up Calibration**

The formula for the duration of the start-up calibration in some versions of the TC2x User’s Manuals is incorrect with respect to the used frequency, or missing.

In the following, the contents of chapter “Calibration” is reprinted, including the correct **Formula for Start-up Calibration** below.

**Calibration**
Calibration automatically compensates deviations caused by process, temperature, and voltage variations. This ensures precise results throughout the operation time.

An initial start-up calibration is required once after a reset for all converters. All converters must be enabled ($ANONS = 11B$). The start-up calibration is initiated globally by setting bit $SUCAL$ in register $GLOBCFG$. Conversions may be started after the initial calibration sequence. This is indicated by bit $CALS = 1B$ AND bit $CAL = 0B$.

**Formula for Start-up Calibration**
The start-up calibration phase takes $4352 f_{ADCI}$ cycles ($4352 \times 50 \text{ ns} = 217.6 \mu\text{s}$ for $f_{ADCI} = 20 \text{ MHz}$).

After that, postcalibration cycles will compensate the effects of drifting parameters. The postcalibration cycles can be disabled.
Note: The ADC error depends on the temperature. Therefore, the calibration must be repeated periodically.

ADC TC.H015 Conversion Time with Broken Wire Detection

As described in a note in section “Broken Wire Detection” of the User’s Manual, the duration of the complete conversion is increased by the preparation phase (same as the sample phase) if the broken wire detection is enabled, i.e. the sample time doubles for standard conversions when broken wire detection is enabled (GxCHCTRY.BWDEN = 1B):

Formula for Standard Conversions without Broken Wire Detection
• \( t_{CN} = t_s + (N + PC) \times t_{ADCI} + 2 \times t_{VADC} \) (see also User’s Manual/Data Sheet)

Formula for Standard Conversions with Broken Wire Detection
• \( t_{CN} = 2 \times t_s + (N + PC) \times t_{ADCI} + 2 \times t_{VADC} \)

where:
\[ t_s = \begin{cases} (2 + STC) \times t_{ADCI} & \text{for } STC \leq 15, \text{ and} \\ (2 + (STC-15) \times 16) \times t_{ADCI} & \text{for } STC \geq 16; \end{cases} \]

\( N = \text{result width (8/10/12 bits)}; \)

\( PC = 2 \text{ if post-calibration selected, } PC = 0 \text{ otherwise.} \)

Examples
Conversion times for different configurations are shown in the following Table 25 (without broken wire detection) and Table 26 (with broken wire detection):
Table 25  Conversion Time for Standard Conversions - Without Broken Wire Detection - Examples

<table>
<thead>
<tr>
<th>Result</th>
<th>Symbol</th>
<th>Time</th>
<th>Conditions</th>
</tr>
</thead>
<tbody>
<tr>
<td>12-bit result</td>
<td>( t_{C_{12}} )</td>
<td>((16 + \text{STC}) \times t_{ADC_I} + 2 \times t_{VADC})</td>
<td>Post-calibration enabled, STC (\leq 15)</td>
</tr>
<tr>
<td>10-bit result</td>
<td>( t_{C_{10}} )</td>
<td>((12 + \text{STC}) \times t_{ADC_I} + 2 \times t_{VADC})</td>
<td>Post-calibration disabled, STC (\leq 15)</td>
</tr>
<tr>
<td>8-bit result</td>
<td>( t_{C_{8}} )</td>
<td>((10 + \text{STC}) \times t_{ADC_I} + 2 \times t_{VADC})</td>
<td>Post-calibration disabled, STC (\leq 15)</td>
</tr>
</tbody>
</table>

Table 26  Conversion Time for Standard Conversions - With Broken Wire Detection - Examples

<table>
<thead>
<tr>
<th>Result</th>
<th>Symbol</th>
<th>Time</th>
<th>Conditions</th>
</tr>
</thead>
<tbody>
<tr>
<td>12-bit result</td>
<td>( t_{C_{12B}} )</td>
<td>((18 + 2 \times \text{STC}) \times t_{ADC_I} + 2 \times t_{VADC})</td>
<td>Post-calibration enabled, STC (\leq 15)</td>
</tr>
<tr>
<td>10-bit result</td>
<td>( t_{C_{10B}} )</td>
<td>((14 + 2 \times \text{STC}) \times t_{ADC_I} + 2 \times t_{VADC})</td>
<td>Post-calibration disabled, STC (\leq 15)</td>
</tr>
<tr>
<td>8-bit result</td>
<td>( t_{C_{8B}} )</td>
<td>((12 + 2 \times \text{STC}) \times t_{ADC_I} + 2 \times t_{VADC})</td>
<td>Post-calibration disabled, STC (\leq 15)</td>
</tr>
</tbody>
</table>

ADC_TC.H016  P02 Output Driver Setting for External Multiplexer Control

Short intermediate values can appear on outputs EMUX0y on P02 when the subchannel number changes, if the respective port drivers operate in strong driver / fast edge mode.

This may lead to unintended drawing of charge from a connected analog signal source.

*Note:* Strong driver / fast edge mode can only be selected for MP pads (speed grade 1), i.e. for EMUX0y outputs on port P02. EMUXxy outputs on ports P00 and P33 use LP pads that only feature medium driver mode, i.e. EMUXxy outputs on these ports are not affected.
 Recommendation
Avoid strong driver / fast edge mode, i.e. speed grade 1 on EMUX0y outputs on P02 pins. Select strong driver / medium edge mode (speed grade 2) instead to avoid the unwanted intermediate states.

ADC_TC.H019 G6ARBCNT Connection to GTM

The connection G6ARBCNT to GTM_TIM1_CH3 is erroneously listed in table “Digital Connections in the TC29x”. It does not exist on silicon. G6ARBCNT is only connected to GTM_TIM1_CH6.

ADC_TC.H020 Minimum/Maximum Detection Compares 12 Bits Only

In minimum or maximum detection mode (FEN = 11_B or 10_B) new results are compared to the lower 12 bits of the respective result register bitfield RESULT. Therefore, a value RESULT = XFFF_H (X > 0_H) will not be updated for a new result value of 0FFF_H in minimum detection mode.

In a real application, this should be no problem, as the minimum detection usually sees values below 0FFF_H.

 Recommendation
For minimum detection, use the start value 0FFF_H (instead of FFFF_H as mentioned in the User’s Manual).

For maximum detection, use the start value 0000_H as mentioned in the User’s Manual.

ADC_TC.H022 Sample Time Control - Formula

Table “Sample Time Coding” in section “Input Class Registers” of the VADC chapter in the User’s Manual describes the additional clock cycles (selected in bit fields STCS and STCE) to be added to the minimum sample time of two analog clock cycles.
As can be seen from the table in the User’s Manual, the step width in the coding depends on the MSB of STCi (i = S or E). The following Table 27 has been copied from the User’s Manual, with the corresponding formula added in the last column:

**Table 27 Sample Time Coding**

<table>
<thead>
<tr>
<th>STCS / STCE</th>
<th>Additional Clock Cycles(^1)</th>
<th>Resulting Sample Time</th>
<th>Clock Cycle Formula</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0000(_B)</td>
<td>0</td>
<td>2 / (f_{ADCi})</td>
<td>2 + STCi</td>
</tr>
<tr>
<td>0 0001(_B)</td>
<td>1</td>
<td>3 / (f_{ADCi})</td>
<td></td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td></td>
</tr>
<tr>
<td>0 1111(_B)</td>
<td>15</td>
<td>17 / (f_{ADCi})</td>
<td>2 + (STCi - 15) x 16</td>
</tr>
<tr>
<td>1 0000(_B)</td>
<td>16</td>
<td>18 / (f_{ADCi})</td>
<td></td>
</tr>
<tr>
<td>1 0001(_B)</td>
<td>32</td>
<td>34 / (f_{ADCi})</td>
<td></td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td></td>
</tr>
<tr>
<td>1 1110(_B)</td>
<td>240</td>
<td>242 / (f_{ADCi})</td>
<td></td>
</tr>
<tr>
<td>1 1111(_B)</td>
<td>256</td>
<td>258 / (f_{ADCi})</td>
<td></td>
</tr>
</tbody>
</table>

\(^1\) The number of resulting additional clock cycles listed in this column corresponds to the term “STC” used in the conversion timing formulas in the Data Sheet.

ADC TC.H024 Documentation: Filter control only in registers \(GxRCR7/GxRCR15\)

In sections “Finite Impulse Response Filter Mode (FIR)” and “Infinite Impulse Response Filter Mode (IIR)” of the VADC chapter in the User’s Manual,

- replace this sentence:
  “Several predefined sets of coefficients can be selected via bitfield DRCTR (coding listed in Table xx-6) in registers G0RCRy (y = 0 - 15)ff and GLOBRCR.”
- with this sentence:
  “Several predefined sets of coefficients can be selected via bitfield DRCTR (coding listed in Table xx-6) in registers \(GxRCR7\) and \(GxRCR15\).”
**ASCLIN_TC.H001** Bit field **FRAMECON.IDLE** in LIN slave mode

In LIN slave mode, bit field **FRAMECON.IDLE** has to be set to 000\textsubscript{B} (default after reset), i.e. no pause will be inserted between transmission of bytes.

For **FRAMECON.IDLE** > 000\textsubscript{B}, the inter-byte spacing of the ASCLIN module is not working properly in all cases in LIN slave mode (no bit errors are detected by the ASCLIN module within the inter-byte spacing).

**ASCLIN_TC.H003** Behavior of LIN Autobaud Detection Error Flag

**Expected Behavior**

In ASCLIN, when auto baud detection (LINCON.ABD) is deactivated, the auto baud measurement should still be active and the Autobaud Detection Error Flag **FLAGS.LA** should be set when the value measured is outside the BRD.LOWERLIMIT and BRD.UPPERLIMIT range.

**Actual Behavior**

The Autobaud Detection Error Flag **FLAGS.LA** is not set, as the auto baud measurement is not active when auto baud detection is deactivated (LINCON.ABD = 0).

**ASCLIN_TC.H004** Changing the Transmit FIFO Inlet Width / Receive FIFO Outlet Width

**Expected Behavior**

The Transmit FIFO should write the data to intended location of TxFIFO, even though the Transmit FIFO inlet width TXFIFOCON.INW is changed between the write operations.

The Receive FIFO should read the data from intended location, even though the Receive FIFO outlet width RXFIFOCON.OUTW is changed between the read operations.


**Actual Behavior (Transmit FIFO)**

The Transmit FIFO does not write the data in the intended location when TXFIFOCON.INW is changed in an increasing order (from 1 to 2 to 4) between write operations.

The Transmit FIFO writes the data only to aligned write index based on the number of bytes to be written (TXFIFOCON.INW).

**Example:** Assuming that the write index of TxFIFO is from 0 to 15 (16 bytes), when TXFIFOCON.INW = 2, the TxFIFO writes two bytes of data starting only from half-word aligned write index (0, 2, 4, …, 14). Similarly when TxFIFO writes four bytes of data starting only from word aligned write index (0, 4, 8, 12).

*Note:* This misbehavior is seen only when TXFIFOCON.INW is changed in-between write operations.

**Actual Behavior (Receive FIFO)**

The Receive FIFO does not read the data from intended location when RXFIFOCON.OUTW is changed in an increasing order (from 1 to 2 to 4) between read operations.

The Receive FIFO reads the data only from aligned read index based on the number of bytes to be read (RXFIFOCON.OUTW).

**Example:** Assuming that the read index of RxFIFO is from 0 to 15 (16 bytes), when RXFIFOCON.OUTW = 2, the RxFIFO reads two bytes of data starting only from half-word aligned write index (0, 2, 4, …, 14). Similarly when RxFIFO reads four bytes of data starting only from word aligned read index (0, 4, 8, 12).

*Note:* This misbehavior is seen only when RXFIFOCON.OUTW is changed in-between read operations.

**Effect**

Previously written data in TxFIFO will be over-written by the new data, when the TxFIFO write index is not aligned with number of data bytes to be written.

Previously read data will be read again, when the RxFIFO read index is not aligned with number of data bytes to be read.
Recommendation

Flush the TxFIFO (TXFIFOCON.FLUSH) or RxFIFO (RXFIFOCON.FLUSH) before TXFIFOCON.INW or RXFIFOCON.OUTW is changed respectively.

**ASCLIN TC.H005 Collision detection error reported twice in LIN slave mode**

An ASCLIN module configured as LIN slave node could report a wrong collision detection error during reception of LIN header after detecting a first correct collision detection error during the transmission of a response field of the previous LIN frame.

This misbehavior is observed under the following sequence:

- The LIN slave node detects a collision detection error when there is a bit error in its transmitted response frame, and then it goes to the idle state as expected.
- The master transmits a header onto the LIN bus, and the LIN slave node receives header and tries to capture the identifier inside the header.
- Then the LIN slave node reports another collision error which is wrongly detected during the reception of identifier although there is no corruption of LIN header on the bus.

Recommendation

Ignore the collision detection error which happened during reception phase of a LIN slave node.

**BCU TC.H001 HSM Transaction Information not captured**

No HSM transaction information is captured by the System Bus Control Unit (SBCU). Therefore the following HSM related control/status register bits in the SBCU do not have any function:

- Register **SBCU_DBGRNT** (SBCU Debug Grant Mask Register):
  - **HSMCMi**: this control bit has no function. Behavior as described for SBCU_DBGRNT.ONE0.
- **HSMRMI**: this control bit has no function. Behavior as described for SBCU_DBGRNT.ONE0.

- **Register SBCU_DGNNT** (SBCU Debug Trapped Master Register):
  - **HSMCMI**: this control bit has no function. Behavior as described for SBCU_DBGNTT.ONE0.
  - **HSMRMI**: this control bit has no function. Behavior as described for SBCU_DBGNTT.ONE0.

---

### BROM_TC.H003 Information related to Register FLASH0_PROCOND

Chapters “TC2x BootROM Content” of the User’s Manuals contain a description of parts of the FLASH0_PROCOND register as used by the firmware. This description in subchapter “Configuration by Boot Mode Index (BMI)” shows an incorrect address F800 1030H.

Correct is the description of this register in the PMU chapter with address F800 2030H (FLASH0 base address F800 1000H + offset 1030H).

---

### BSL_TC.H002 Bootstrap Loaders need initialized Memory

The Bootstrap Loaders write the received data with word writes ("st.w") to PSPR memory of CPU0. These write accesses fail if the target addresses contain ECC incorrect data. This is the case after power-on reset if the user has disabled the automatic RAM initialization via PROCOND.

**Recommendation**

Ensure that at least CPU0 RAMs are initialized at least for cold power-on resets by proper PROCOND configuration (bit fields RAMIN and RAMINSEL).

---

### CCU6_AI.H001 Update of Register MCMOUT

At every correct Hall event (CM_CHE), the next Hall patterns are transferred from the shadow register MCMOUTS into MCMOUT (Hall pattern shadow transfer HP_ST), and a new Hall pattern with its corresponding output pattern can be loaded (e.g. from a predefined table in memory) by software into
MCMOUTS. For the Modulation patterns, signal MCM_ST is used to trigger the transfer.

Loading this register can also be done by writing MCMOUTS.STRHP = 1\(_B\) (for EXPH and CURH) or MCMOUTS.STRMCMP = 1\(_B\) (for MCMP).

**Note:** If in a corner case a hardware event occurs simultaneously with a software write where MCMOUTS.STRHP = 1\(_B\) or MCMOUTS.STRMCMP = 1\(_B\), the current contents of MCMOUTS is copied to the corresponding bit fields of MCMOUT. The new value written to MCMOUTS will be loaded upon the next event.

### CCU6.AI.H002 Description of Bit RWHE in Register ISR

Register ISR (Interrupt Status Reset Register) contains bits to individually clear the interrupt event flags by software. Writing a 1\(_B\) clears the bit(s) in register IS at the corresponding bit position(s), writing a 0\(_B\) has no effect.

In some versions of the User’s Manual, the description of bit RWHE (Reset Wrong Hall Event Flag) in column “Description” of register ISR is wrong (description for status 0\(_B\) and 1\(_B\) inverted).

The correct description for bit RWHE is (like for all other implemented bits in register ISR) as shown in the following **Table 28**:

<table>
<thead>
<tr>
<th>Field</th>
<th>Bits</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RWHE</td>
<td>13</td>
<td>w</td>
<td><strong>Reset Wrong Hall Event Flag</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0(_B) No action</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1(_B) Bit WHE will be cleared</td>
</tr>
</tbody>
</table>

### CCU6.AI.H003 Bit TRPCTR.TRPM2 in Manual Mode - Documentation Update

In CCU6 chapter “Trap Control Register” of the User’s Manual, the description for bit TRPCTR.TRPM2 = 1\(_B\) (Manual Mode) incorrectly states:
“Manual Mode:
Bit TRPF stays 0 after the trap input condition is no longer valid. It has to be cleared by SW by writing ISR.RTRPF = 1.”

**Correction**
The correct description is as follows:
Manual Mode:
Bit TRPF stays 1 after the trap input condition is no longer valid. It has to be cleared by SW by writing ISR.RTRPF = 1.

**CCU_TC.H001 Clock Monitor Check Limit Values**
The values for the check limits of the clock monitor have been updated as shown in **Table 29**. This table replaces the corresponding table in chapter “Clock Monitors” of the User’s Manual.

**Table 29**    Target trimmed Check limits

<table>
<thead>
<tr>
<th>Target Frequency</th>
<th>LOWER value</th>
<th>UPPER value</th>
<th>SELXXX 1)</th>
<th>Error can be detected for min. deviation</th>
<th>Error is detected for min. deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>7.5 MHz</td>
<td>0x24</td>
<td>0x27</td>
<td>11B</td>
<td>-1.26% +1.54%</td>
<td>-6.45% +6.35%</td>
</tr>
<tr>
<td>6.6 MHz</td>
<td>0x20</td>
<td>0x23</td>
<td>10B</td>
<td>-0.91% +2.75%</td>
<td>-6.09% +7.50%</td>
</tr>
<tr>
<td>6 MHz</td>
<td>0x1C</td>
<td>0x1F</td>
<td>01B</td>
<td>-3.35% +1.54%</td>
<td>-8.43% +6.35%</td>
</tr>
<tr>
<td>5 MHz</td>
<td>0x17</td>
<td>0x1A</td>
<td>00B</td>
<td>-2.76% +4.07%</td>
<td>-9.41% +7.50%</td>
</tr>
</tbody>
</table>

1) refers to corresponding bit field xxxSEL in respective CCUCON register
CCU_TC.H002 Oscillator Gain Selection via OSCCON.GAINSEL

The reset value of OSCCON.GAINSEL = 11\textsubscript{B} provides the default and recommended setting for the oscillator gain. It is not required to modify this value, as the adaptation to a crystal frequency is done via the external circuitry. Therefore, all other gain selections should be regarded as reserved for special application topics, as shown in the following Table 30.

Table 30 Oscillator Gain Selection via OSCCON.GAINSEL

<table>
<thead>
<tr>
<th>Field</th>
<th>Bits</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GAINSEL</td>
<td>[4:3]</td>
<td>rw</td>
<td>Oscillator Gain Selection</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>This value should not be changed from the reset value 11\textsubscript{B}.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>00\textsubscript{B} Low gain 1: reserved for adaptations</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>01\textsubscript{B} Low gain 2: reserved for adaptations</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>10\textsubscript{B} Low gain 3: reserved for adaptations</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>11\textsubscript{B} Maximum gain: default setting</td>
</tr>
</tbody>
</table>

Recommendation

Always to keep the default configuration of OSCCON.GAINSEL = 11\textsubscript{B}.

CCU_TC.H005 References to f\textsubscript{PLL2}, f\textsubscript{PLL2_ERAY} and K3 Divider in User’s Manual

The VADC incorporated in this device uses clocks derived from f\textsubscript{SPB}. Previous design steps (e.g. TC27x Bx, TC26x Ax, TC29x Ax) incorporated a different VADC module also clocked by f\textsubscript{ADC}, which could be derived via the K3 divider from f\textsubscript{PLL2}, f\textsubscript{PLL2_ERAY}. These clocks were selected in CCUCON0.[27:26], which is described as “Reserved/Should be written with 0” in the present version of the User’s Manual.

Clocks f\textsubscript{PLL2}, f\textsubscript{PLL2_ERAY} and the K3 divider are still described in the present version of the User’s Manual.
Recommendation

- New software implementations should not consider $f_{PLL2}$, $f_{PLL2\_ERAY}$ and the K3 divider.
- Software ported from previous design steps with a VADC module clocked by $f_{ADC}$ may be reused on this device step (see also SMU\_TC.H004).

**CCU\_TC.H006 Clock Monitor Support - Documentation Update**

The note at the end of section “Operating the Clock Monitors” in chapter “Clock Monitors”:

*Note: This feature is supported by the Infineon safety driver [safTlib] and there is no additional customer software required.*

should state more precisely:

*Note: The Infineon SafeTlib provides a test for the clock monitor. The clock monitor shall be configured by the application software.*

**CCU\_TC.H007 Oscillator Watchdog Trigger Conditions for ALM3[0]**

As described in the User’s Manual in section “Oscillator Watchdog”, the divider value OSCCON.OSCVAL has to be selected in a way that $f_{OSCREF}$ is within the range of 2 MHz to 3 MHz, and should be as close as possible to 2.5 MHz.

The Oscillator Watchdog (OSC\_WDT) will trigger the “input clock out of range” alarm ALM3[0] under the following conditions:

- **Boundary for too high** frequencies:
  - for $(\text{OSCVAL}+1) \times 6.25 \leq f_{OSC} [\text{MHz}] \leq (\text{OSCVAL}+1) \times 7.5$, an alarm can be generated, but there is no guarantee that it is generated,
  - for $f_{OSC} [\text{MHz}] > (\text{OSCVAL}+1) \times 7.5$, an alarm is always generated.

- **Boundary for too low** frequencies:
  - for $(\text{OSCVAL}+1) \times 1.25 \leq f_{OSC} [\text{MHz}] \leq (\text{OSCVAL}+1) \times 1.67$, an alarm can be generated, but there is no guarantee that it is generated,
  - for $f_{OSC} [\text{MHz}] < (\text{OSCVAL}+1) \times 1.25$, an alarm is always generated.

The accuracy of these limits [in %] depends on the variation [in %] of the back up clock (see specification of $f_{BACKUT}$ and $f_{BACKT}$ in the Data Sheet).
Example

- For $f_{\text{OSC}} = 20$ MHz, selecting $\text{OSCVAL} = 7$ results in $f_{\text{OSC}} = 2.5$ MHz.
  - An alarm for too high frequencies can be generated for $f_{\text{OSC}} \geq 50$ MHz,
  - An alarm for too high frequencies is always generated for $f_{\text{OSC}} > 60$ MHz.
  - An alarm for too low frequencies can be generated for $f_{\text{OSC}} \leq 13.36$ MHz,
  - An alarm for too low frequencies is always generated for $f_{\text{OSC}} < 10$ MHz.

CPU_TC.H006 Store Buffering in TC1.6/P/E Processors

Overview

Store buffering is a method of increasing processor performance by decoupling memory write operations from the instruction execution flow within the CPU. All write data is placed in a FIFO buffer (known as the store buffer) by the CPU prior to being read by the memory/bus interfaces and written to memory. This allows the processor to continue execution without waiting for the write data to be written to the target memory location. Data is written to the store buffer at processor speed and read from the store buffer at memory/bus speed. Typically the read bandwidth from the store buffer will exceed the write bandwidth from the processor, only if the store buffer fills will the processor stall.

To further increase performance memory read operations are prioritised ahead of memory write operations from the store buffer. This ensures that the processor does not stall on data loads while data writes are pending in the store buffer. A side effect of this prioritising is that memory may not be accessed in program order.

Operational Details

The function of the store buffer is designed to be invisible to the end user under normal operation:

- All CPU load operations are checked against the store buffer contents. Data for matching load addresses is either immediately forwarded to the CPU from the store buffer (TC1.6, TC1.6P) or written to memory prior to the load operation proceeding (TC1.6E).
• All loads and store operations to peripheral regions (typically segments $E_H$ and $F_H$) are performed in strict program order (no load prioritisation).

The operation of the store buffer can become visible when in-order memory access is required to non-peripheral segments. This can occur under the following circumstances:

• When programming flash memory.
• When performing memory testing with the processor.
• When data is required to be in memory for inter-core/inter-module communication.

In such cases the following solutions may be employed:

• The store buffer may be explicitly flushed by use of a DSYNC instruction.
• The store buffer may be disabled by setting $SMACON.IODT$. This should not be done during normal operation as it significantly impacts performance.

**Examples**

The following examples refer to memory accesses to non-peripheral regions (i.e. segments $0_H .. D_H$):

**Example-1a** Out of order memory access due to load prioritisation

<table>
<thead>
<tr>
<th>Program Flow</th>
<th>Memory Access</th>
</tr>
</thead>
<tbody>
<tr>
<td>st-1</td>
<td>ld-4</td>
</tr>
<tr>
<td>st-2</td>
<td>ld-5</td>
</tr>
<tr>
<td>st-3</td>
<td>ld-6</td>
</tr>
<tr>
<td>ld-4</td>
<td>st-1</td>
</tr>
<tr>
<td>ld-5</td>
<td>st-2</td>
</tr>
<tr>
<td>ld-6</td>
<td>st-3</td>
</tr>
</tbody>
</table>

**Example-1b** In order memory access enforced by DSYNC

<table>
<thead>
<tr>
<th>Program Flow</th>
<th>Memory Access</th>
</tr>
</thead>
<tbody>
<tr>
<td>st-1</td>
<td>st-1</td>
</tr>
<tr>
<td>st-2</td>
<td>st-2</td>
</tr>
<tr>
<td>st-3</td>
<td>st-3</td>
</tr>
<tr>
<td>dsync</td>
<td></td>
</tr>
</tbody>
</table>
Example-2a  Load forwarding from store buffer - no memory read (TC1.6/1.6P)

Program Flow    -    Memory Access
st.w [a0], d0
ld.w d1,  [a0]       st.w [a0], d0

Example-2b  In order memory access enforced by DSYNC (TC1.6/1.6P)

Program Flow    -    Memory Access
st.w [a0], d0        st.w [a0], d0
dsync
ld.w d1,  [a0]       ld.w d1,  [a0]

**CPU TC.H008  Instruction Memory Range Limitations**

To ensure the processor cores are provided with a constant stream of instructions the Instruction Fetch Units will speculatively fetch instructions from up to 64 bytes ahead of the current Program Counter (PC).

If the current PC is within 64 bytes of the top of an instruction memory the Instruction Fetch Unit may attempt to speculatively fetch instructions from beyond the physical range. This may then lead to error conditions and alarms being triggered by the bus and memory systems.

**Recommendation**

It is therefore recommended that either the MPU is used to define the allowable executable range or that the upper 64 bytes of any memory be initialized but unused for instruction storage for the TC1.6.* class processors. For TC1.3.* class processors this may be reduced to 32 bytes.
CPU TC.H009 Details on CPU Clock Control

As described in chapter “Clock Control Unit” of the User’s Manual, the effective CPU execution frequency may be reduced by programming the associated bit field CPUxDIV in register CCUCONn (where x is the core number, and n = x+6).

The effective execution frequency $f_{CPUx}$ seen by CPUx is given by the following equation (where $f_{SRI}$ is the base SRI frequency):

- $f_{CPUx} = f_{SRI} \times \frac{(64 - CPUxDIV)}{64}$

A CPUxDIV value of 0 results in the core CPUx being clocked at the SRI frequency (no frequency reduction).

To avoid synchronisation issues typically associated with clock division the clock control mechanism stalls the issue of instructions into the processor pipeline rather than by modifying the actual applied clock. An incoming instruction fetch packet is stalled for the number of cycles required to approximate the required execution frequency. The stall is seen by the processor as a stall in the instruction stream in the same way a stalling instruction memory would be seen.

In most scenarios this mechanism provides a good approximation to clock division based control. The actual reduction in effective frequency will be dependent on the code executed.

When determining IPC rates as described in AP32168 (Application Performance Optimization for TriCore V1.6 Architecture), note that for CPUxDIV > 0, field Count Value in register CCNT still represents SRI clock cycles.

CPU TC.H010 External Accesses to CPU Local Memory may delay CPU Execution Progress

A sequence of contiguous external accesses to the CPUx local memory (DSPR/DCache, PSPR/PCache) may delay the CPUx execution progress for a potentially long time.

External accesses to the CPUx local memory may arrive from several agents (CPUy, DMA, etc.), therefore, the resulting sequence of external accesses to
the CPUx local memory may be contiguous, even if each of the agents is leaving some gaps between its requests.

Note: The CPUx execution continues when the external access sequence is finished. There is no impact on the correctness of code execution.

Known Cases

• An external access to DSPR memory may delay CPUx execution progress, if CPUx is accessing its local DSPR memory or using Data Cache. Local accesses to DSPR memory include: data load and store, context save and restore operations.
• An external access to PSPR memory may delay CPUx execution progress, if CPUx is executing code from a memory other than its own local PSPR and Program Cache is enabled.

Recommendations

• Reduce the frequency of external accesses to DSPR and PSPR memory.
• Introduce gaps in long access sequences to DSPR and PSPR memory.

CPU TC.H012 Behavior of bit-wise operations on certain peripheral register bits which need to be written back with the same value

The LDMST, ST.T, CMPSWAP.W, SWAPMSK.W and SWAP.W instructions in the AURIX™ microcontrollers are instructions intended to provide atomicity as well as bit-wise operations to a targeted memory location or peripheral register. They are also referred to as Read-Modify-Write (RMW) instructions.

In some registers in certain modules, a bit has to be written with the same value (e.g. a bit set to 1B has to be written with a 1B to perform an operation). When using a RMW instruction to write to such a bit, the write is masked away and will not happen at all.

Note: Writing a different value (e.g. writing a 1B to a bit currently at 0B) is not affected, and works as expected to modify only the selected bit.

Example: Consider the GxVFR register in the VADC module:
Figure 7  Register GxVFR in the VADC Module of TC2xx Devices

The bits in the GxVFR register have to be written with 1B to clear a valid flag VFy indicating a valid result. Assuming VFy = 1B, if one of the RMW instructions listed above is used, the write to VFy would never happen since VFy is already set to 1B. This means that the next read of VFy may lead to incorrect conclusions by software.

Affected Modules and Registers in the AURIX™ Platform

- CCU6: IMON
- VADC: GxVFR, GxSEFLAG, GxCEFLAG, GxREFLAG, GLOBEFLAG.

Note: VADC is located outside the addressable range of ST.T, so ST.T need not be considered in the context of VADC.

Recommendation

In the affected modules, use only direct writes (i.e., write the whole register as a 32-bit word), and do not use RMW operations to write to such bits.

For example, to clear bit VF0 in the GxVFR register, the software should write:
VADC_GxVFR.U = 0x00000001;
Here .U implies writing the whole 32-bit register as an unsigned integer.

**CPU_TC.H014**  ACCEN* Protection for Write Access to Safety Protection Registers - Documentation Update

The access protection symbol ‘P’ to indicate protection by the ACCEN* register mechanism is missing in column “Access Mode - Write” in table “Safety Protection Registers” in the CPU chapter of the User’s Manual for RGN*x registers with an index x ≥4.
Actually, these registers also have write access attribute ‘P’.

**CPU_TC.H015**  Register Access Modes for Safety Protection Registers - Documentation Update

The access protection symbol ‘U’ is erroneously included and should be removed in column “Access Mode - Write” for all registers in table “Safety Protection Registers” in the CPU chapter of the User’s Manual.
The note below this table is rephrased as follows:
Note: A disallowed access to any CPU register (e.g. attempted write to non-existent register, attempted write to read only register, attempted access to E without Endinit, etc.) will NOT result in a Bus Error

**DAP_TC.H002**  DAP client_blockread in Combination with TGIP and all Parcels with CRC6

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

When issuing a DAP client_blockread telegram together with the TGIP (Trigger in Protocol) option (DAPISC.TGIP = 1) the TGIP extra bit is appended for each parcel in case “all parcels with CRC6” is enabled. This causes a slight increase in the communication length compared to the correct behavior of having a TGIP bit only for the last parcel.
**Recommendation**

Do not use the TGIP and “CRC6 for all parcels” features together in case this extra bit can not be tolerated. If the Trigger in Protocol and increased communication safety is required TGIP can be used together with the CRC32 option (see also DAP_TC.002 DAP client_blockread has Performance issue in Specific Operation Modes).

**DAP_TC.H003 Not acknowledged DAP telegrams in noisy environments**

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

DAP telegrams always follow a request-reply scheme. The request is driven by the tool, the reply by the AURIX™. The AURIX™ acknowledges a correctly received telegram always by a reply, which consists at least of a start-bit. DAP communication in noisy environments might result in invalid telegrams. This can leave the IOClient in an intermediate state which requires an IOClient reset.

If AURIX™ receives an invalid telegram with a wrong CRC6 or length field, it does not reply at all and in some cases the selected IOClient might be left in an intermediate state in case of a detected client_write/blockwrite/readwrite tool request.

**Recommendation**

If a tool does not receive a start bit as an acknowledge for an IOClient request, a client_reset must be sent as the next telegram for the selected IOClient. Tool interaction with the DAP module itself is not affected and can be done in between.

**DMA_TC.H002 Bit CHCSRz.BUFFER can be toggled when not in Double Buffer Mode**

The purpose of bit CHCSRz.BUFFER is to indicate which buffer is read or filled during DMA double buffering (selected in bitfield ADICRz.SHCT).
However, bit CHCSRz.BUFFER can also be toggled by writing bit CHCSRz.SWB = 1B when not in Double Buffer Mode.

Recommendation
Do not write bit CHCSRz.SWB = 1B when not in Double Buffer Mode.

**DMA TC.H003  Spurious Error Interrupt Service Requests after Transaction Lost Event in Double Buffer Mode**

When a DMA channel is configured for any double buffering operation (ADICRz.SHCT[3:2] = 10B) then there is a possibility of spurious error interrupt service requests.

If a Transaction Request Lost event occurs (TSRz.TRL = 1B) AND if the transaction request lost interrupt is enabled (ADICRz.ETRL = 1B) then there is a possibility that spurious error interrupt service requests will be generated.

No Transaction Request Lost (TRL) events will be missed, but one TRL event may result in multiple error interrupt service requests.

Recommendation
It is recommended that if an error interrupt service request is triggered then bit TRL should be cleared immediately by writing TSRz.CTL = 1B to prevent further spurious error interrupt service requests.

**DMA TC.H004  Transaction Request Lost upon software trigger with pattern match**

If a DMA channel is configured for pattern detection and software triggering of each DMA transfer (CHCSRz.RROAT = 0B), then if there is a new DMA software request received while a DMA transfer is executing then a Transaction Request Lost event may be lost.

Recommendation
The loss of TRL status is a debug feature. A DMA channel should be used such that TRL is not set.
The user must ensure that the CPU triggers a new DMA software request when no DMA access is pending. The software could poll the TSRz.CH bit to confirm it is $0_B$ before issuing a DMA software trigger.

**DMA_TC.H005  Linked List Transfer leading to loading of non-Linked List TCS causes corruption**

If on completion of a Linked List (LL) a non-LL Transaction Control Set (TCS) is loaded with shadow address buffering enabled (read only and direct write) then the new non-LL TCS can be corrupted.

**Recommendation**

Shadow address buffering must be disabled in the non-LL TCS ($SHCT[3:0] = 0000_B$)

**DMA_TC.H006  Clearing of HTRE when DMA channel is configured for Single Mode**

The DMA may be used to support a peripheral with a high interrupt rate where the interrupts are generated in quick succession (e.g. a QSPI filling a TXFIFO). The DMA channel z is configured with the following settings:

- Single Mode (HTRE is reset by hardware on completion of a DMA transaction)
  - $TSRz.CHMODE = 0_B$
- Request required for each DMA Transfer
  - $TSRz.RROAT = 0_B$

If the DMA channel is configured to execute a DMA transaction of 1 x DMA transfer of 2 x DMA moves:

- Block Mode: 2 x DMA Move per DMA transfer
  - $DMA\_CHCFGRz.BLKM = 001_B$
- Transfer Reload Value: 1 x DMA transfer
  - $DMA\_CHCFGRz.TREL = 1_B$

then additional DMA moves are executed unexpectedly.
Explanation of Effect

If the peripheral generates two interrupt service requests in relatively quick succession then the first DMA hardware request is serviced by the DMA and performs one DMA transfer comprising two DMA moves. The second DMA hardware request arrives before the completion of the first DMA transfer (i.e. before the clearing of HTRE at the end of the DMA transaction). The second hardware request is serviced by the DMA and performs a second DMA transfer comprising two DMA moves.

Recommendation

If the second DMA hardware request arrives before completion of the first DMA transfer then the DMA channel Block Mode must limit a DMA transfer to one DMA move:

- DMA_CHCFGRz.BLKM = 000B; //1 x DMA move/DMA transfer

The total number of DMA moves must be defined by the Transfer Reload Value DMA_CHCFGRz.TREL.

DMA_TC.H007 Selecting the Priority for DMA Channels

All used DMA channels should be configured with the highest priority on SPB in respect to other used SPB master agents (CPUs, HSSL, ETH) to enable a robust execution of the configured DMA transactions.

The DMA channels are configured per default with the lowest priority on SPB:

- DMA_CHCFGRz.DMAPRIO = 00B --> maps DMA channel z SPB requests to SPB priority DMAL
- SBCU PRIOH.DMAL = 1111B --> configures DMAL with the lowest priority on SPB

Recommendation

There are several ways to configure used DMA channels with the highest priority on SPB with respect to other SPB master agents. Two examples follow:
Example1
Map the used DMA channels to SPB priority DMAH by setting DMA_CHCFGRz.DMAPRIO = 11B and keep the configuration of the DMAH priority (SBCU_PRIOL.DMAH = 0000B).

Example2
Keep the mapping of the used DMA channels to DMAL (DMA_CHCFGRz.DMAPRIO = 00B) and change the priority configuration of DMAL (e.g. set SBCU_PRIOH.DMAL = 0001B).

Background
The DMA can request for SPB access with three different requests (DMAH, DMAM, DMAL) that are configured with different SPB priorities with respect to the other SPB master agents (CPUx, HSCT, ETH). The priority of the DMA requests DMAH, DMAM and DMAL on the SPB in respect to the priority of other SPB master agents can be configured via the SBCU registers SBCU_PRIOL / SBCU_PRIOH.
Each DMA channel z can be configured via DMA_CHCFGRz.DMAPRIO regarding which of three priorities (DMAH, DMAM or DMAL) it uses for SPB access.
The default configuration of DMA_CHCFGRz.DMAPRIO = 00B. This means that the channels will request for SPB access with the DMAL priority.
The priority of a DMAL request on SPB is configured per default with the lowest priority (SBCU_PRIOH.DMAL = 1111B).

DMA_TC.H008 Transaction Request State
The DMA Transaction Request State bit DMA_TSRz.CH is cleared when the DMA transfer starts (RROAT = 0B) or at the end of a DMA transaction (RROAT = 1B).
Figure “Channel Request Control” and RROAT bit field description of register DMA_MExCHCR in chapter “Register Description” of the User's Manual are wrong.
**DMA_TC.H009 Resetting Bits ICH and IPM in register CHCSRz**

The Clear Interrupt from Channel bit (CICH) is accessible via the DMA channel CHCSR register.

The AURIX™ TC2xx User Manuals are incorrect with respect to the following statement:

- The DMA channel DMA_CHCSRz ICH and IPM bit field description states:
  “is reset by software when writing a 1 to ADICRz.CICH”.

**Correction**

- The text should read:
  “is reset by software when writing a 1 to CHCSRz.CICH”.

**DMA_TC.H010 Calculation of DMA Address Checksum for DMA read moves to Cacheable Addresses**

The DMA Move Engine (ME) stores the DMA read move data in eight 32-bit read registers. If a DMA read move is to a cached address (Segment 8 or 9), the ME shall translate the DMA read move access to the on chip bus into an SRI BTR4 access to a 32-byte aligned address. The DMA shall calculate the DMA address checksum from the on chip bus address i.e. the 32-byte aligned address. The DMA shall store the DMA address checksum in the SDCRCR.

**Recommendation**

If an expected DMA address checksum is pre-calculated to test the DMA address generation, the user shall take note of the address translation to 32-byte aligned addresses when calculating the expected DMA address checksum from a cacheable DMA source address.

Alternatively, DMA read moves should be performed to non-cacheable source addresses (segments A and B).
DMA_TC.H011 DMA_ADICRz.SHCT - Reserved Values

The DMA channel shadow control bit field DMA_ADICRz.SHCT controls the function of the shadow address register. If software programs a reserved value in DMA_ADICRz.SHCT, the DMA may deadlock the operation of the DMA. Therefore, software shall not program DMA_ADICRz.SHCT with the following reserved values:

- \(0011_B\) Reserved
- \(0100_B\) Reserved
- \(0111_B\) Reserved.

DMA_TC.H012 TCS Update in Halt State

If a DMA channel is in halt state,

- The DMA shall stop performing DMA moves to the destination location.
- Software may perform a background test on the destination location.
- Software may modify the DMA channel Transaction Control Set (TCS).

Recommendation

If software modifies the DMA channel TCS, software shall only modify the DMA channel source address (DMA_SADRz.SDAR) and the DMA channel destination address (DMA_DADRz.DADR).

DSADC_TC.H002 Influence of Temperature on DC Offset Error EDOFF (calibrated)

The performance of the DSADC can be improved by applying some calibration techniques to compensate temperature effects.

E.g. parameter “DC Offset Error” (symbol EDOFF) may exceed the specified Data Sheet value of ±5 mV (test condition = calibrated) when temperature has changed by more than approximately 20 °C after calibration.
Recommendation
To compensate temperature effects it is recommended to repeat the calibration sequence when the device temperature has changed by approximately 20 °C. (see section “Calibration Support” in DSADC chapter of the User's Manual).

**DSADC_TC.H003**  FIR Filters not reset when Integration starts

When the integration window is started, the CIC filter and the integrator are reset. However, the FIR filters are not reset in this case.

If the FIR filters are active, the time delay to the first result value is not constant, but is shortened by 1/4, 2/4, or 3/4 of a result data period, depending on the current state of the FIR filters.

For repeated measurements, this may cause a timing jitter from the start of the integration window until the first result is available.

Recommendation
To compensate the effect on the discard phase, the number of values discarded may be increased by 1 (bit field NVALDIS in register IWCTRx).

In case the jitter in the result timing is not tolerable,

- either do not use the FIR filters and perform integration by software,
- or force a reset of the entire chain including the FIR filters by switching off/on the corresponding CHxRUN bit in register GLOBRC e.g. via DMA.

**DSADC_TC.H004**  Full-scale Values produced by On-chip Modulator

Due to SNR improvements, the full-scale values produced by the on-chip modulator in this device step differ by a factor of about 2 from previous device steps, as shown in the following table. See also chapter “Filter Configuration and Control”, subchapter “Recommended Settings” in the User's Manual.
Recommendation

For this device step, use the value of 3800\text{D} to calculate the setup of the filter chain. To avoid overflow and clipping of values within the filter chain, the magnitude of the result values must not exceed \( \pm 2^{15} \) at any stage.

When migrating from previous design steps, the increased output amplitude of this design step may be compensated by the data shifter setting (FCFGMx.FSH = 0\text{B} instead of 1\text{B} in the example given in the User’s Manual).

**DSADC_TC.H005 Data Strobe Setting for On-chip Modulator**

In this device step, the improved on-chip modulator uses the rising edge of the modulator clock to transfer the data values to the digital filter chain. To ensure proper reception of this data, the filter chain must evaluate the data values with the **falling** edge of the modulator clock.

**Recommendation**

Bitfield STROBE in register DICFGx must be set to 0010\text{B}.

The description of bitfield DICFGx.STROBE will be adjusted accordingly in the next revision of the User’s Manual as shown in **Table 32**.

**Table 31 Full-scale Values produced by On-chip Modulator**

<table>
<thead>
<tr>
<th>Full-Scale Values</th>
<th>Device Step ≥ BA</th>
<th>Device Step = Ax</th>
</tr>
</thead>
<tbody>
<tr>
<td>Uncalibrated average value</td>
<td>( \pm 3600\text{D} \ (0E10\text{H} / F1F0\text{H}) )</td>
<td></td>
</tr>
<tr>
<td>Value used in example calculation to avoid filter overflows</td>
<td>( \pm 3800\text{D} \ (0ED8\text{H} / F129\text{H}) )</td>
<td>( \pm 1900\text{D} \ (076C\text{H} / F894\text{H}) )</td>
</tr>
</tbody>
</table>
Avoiding Intermediate States

The DSADC may experience unintended intermediate states in the two scenarios identified below. To avoid these states, consider the following recommendations:

**Intermediate States due to External Signals**

External control signals are not specially filtered. Therefore, glitches on those external signals may also affect the internal functions controlled by them.

To ensure proper operation of the externally controlled functions, it is recommended to provide glitch-free control signals.

The following input signals should be considered:

- Trigger inputs (ITRxy)
- External carrier sign (SGNA, SGNB)

Alternatively, internal signal sources can be selected for the respective functions.

**Intermediate States during Configuration**

Similarly, it is recommended to change configurations only while modulator and channel are stopped.

This avoids unintended intermediate states.

Recommended sequence:

### Table 32 DICFGx (Demodulator Input Configuration Register x), Bitfield STROBE

<table>
<thead>
<tr>
<th>Field</th>
<th>Bits</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>STROBE</td>
<td>[23:20]</td>
<td>rw</td>
<td>Data Strobe Generation Mode</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0000B</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0010B</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Other combinations are reserved</td>
</tr>
</tbody>
</table>

**DSADC_TC.H006 Avoiding Intermediate States**

The DSADC may experience unintended intermediate states in the two scenarios identified below. To avoid these states, consider the following recommendations:

**Intermediate States due to External Signals**

External control signals are not specially filtered. Therefore, glitches on those external signals may also affect the internal functions controlled by them.

To ensure proper operation of the externally controlled functions, it is recommended to provide glitch-free control signals.

The following input signals should be considered:

- Trigger inputs (ITRxy)
- External carrier sign (SGNA, SGNB)

Alternatively, internal signal sources can be selected for the respective functions.

**Intermediate States during Configuration**

Similarly, it is recommended to change configurations only while modulator and channel are stopped.

This avoids unintended intermediate states.

Recommended sequence:
1. Write the static configuration while modulator and demodulator are disabled (GLOBRC.MxRUN = 0B, GLOBRC.CHxRUN = 0B)
2. Enable the modulator (GLOBRC.MxRUN = 1B) and wait for the modulator to settle (see Data Sheet, parameter “Modulator settling time” t_MSET)
3. Enable the demodulator (GLOBRC.CHxRUN = 1B)

**DSADC_TC.H007  Dithering Control**

The dithering feature reduces the idle tones caused by low-frequency input signals and minimizes the dead-zone.

After reset, dithering is enabled, but the dithering trim value is 000_B (minimum intensity).

**Recommendation**

To optimize the effect, it is recommended to select a higher dithering intensity:

- DITRIM = 001_B (low intensity) ensures the conversion performance (SNR) in all cases, but leaves a residual dead-zone of approximately 2 mV.
- DITRIM = 011_B (medium intensity) reduces the residual dead-zone below -80 dB.

In this case, the voltage of the input signal must not exceed 90% of the reference voltage and an oversampling rate of OSR ≥ 200 is required to achieve an SNR of 80 dB.

**DSADC_TC.H008  DSADC Gain Calibration Procedure**

In order to improve the overall accuracy of the DSADC, an algorithm for the gain calibration using the High-Precision Square wave Generator (HPSG) is proposed in section “Gain Calibration Support” of the User’s Manual.

The calibration is done by following the sequence for measuring the output signal of the HPSG as described in the User’s Manual. After enabling the calibration mode, the HPSG needs up to half a period for settling. In order to achieve a reproducible result it is highly recommended to read a sequence of values for at least two full periods. Then evaluate a complete period e.g.
between two rising edges of the square wave for calculating the amplitude of the waveform and go on with the calculation as proposed in the User's Manual. For your reference, the corresponding bullet points are copied from the description in the Users Manual and extended accordingly:

- Enable gain calibration mode (MODCFGx.GCEN = 1)
- Determine the actual amplitude by converting the high level and the low level of the square wave signal. Result = AM.
  - Read a sequence of result values for at least two full periods.
  - Then evaluate a complete period e.g. between two rising edges of the square wave for calculating the amplitude of the waveform
  - Ignore the values close to the signal transitions to exclude the overshoots/undershoots caused by the Gibbs phenomenon (see Figure in User’s Manual).
  - ...

**DTS_TC.H001 Update of Bit DTSSTAT.BUSY**

The following statement in the description of bit BUSY in register DTSSTAT in the SCU chapter “Die Temperature Measurement” is incorrect:

*Note: This bit is updated 2 cycles after bit DTSCON.START is set.*

**Correction**

The correct description is as follows:

*Note: This bit is updated 7 cycles after bit DTSCON.START is set.*

**EBU TC.H013 EBU_USERCON Upper Half-word - Documentation Update**

In EBU chapter “Test/Control Configuration Register, USERCON” of the User’s Manual, the correlation between the upper half-word EBU_USERCON.[31:16] of the register image and its description should be as follows:

- ADDIO should be bits USERCON.[19:16] in the register image (rw field)
- Bits USERCON.[27:20] in the register image should be reserved (r field)
- Bits USERCON.[29:28] should be BCEN in the register description (rw field)
The updated register image of the upper half-word of EBU_USERCON is shown in Table 33, the updated description in Table 34:

### Table 33  EBU_USERCON - Register Image Upper Half-word

<table>
<thead>
<tr>
<th>31</th>
<th>30</th>
<th>29</th>
<th>28</th>
<th>27</th>
<th>26</th>
<th>25</th>
<th>24</th>
<th>23</th>
<th>22</th>
<th>21</th>
<th>20</th>
<th>19</th>
<th>18</th>
<th>17</th>
<th>16</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADD</td>
<td>AD</td>
<td>LSW</td>
<td>VIO</td>
<td>BCEN</td>
<td>RES</td>
<td>ADDIO</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>rw</td>
<td>rw</td>
<td>rw</td>
<td>rw</td>
<td>r</td>
<td>rw</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Table 34  EBU_USERCON - Register Description Upper Half-word

<table>
<thead>
<tr>
<th>Field</th>
<th>Bits</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| ADDIO | 19:16 | rw | Address Pins to GPIO Mode  
- Description see User’s Manual - |
| RES | 27:20 | r | Reserved |
| BCEN | 29:28 | rw | Byte Control Enable  
- Description see User’s Manual - |
| ADVIO | 30 | rw | ADV Pin to GPIO Mode  
- Description see User’s Manual - |
| ADDLSW | 31 | rw | Enable Address LSW, A(15:0) for GPIO  
- Description see User’s Manual - |

**EBU_TC.H014  Reset of EBU Registers**

In chapter “EBU Registers” of the User’s Manual, the sentence “All registers are reset by the module reset” is incorrect.

Unlike other modules, the EBU does not have KRST* registers and thus no module reset functionality. All EBU registers are reset by Application Reset.

**EMEM_TC.H002  EMEM will raise ECC errors when not properly initialized**

*Note: This application hint only applies to ADAS or Emulation Devices (ED).*
After power-on the RAM contents is random. This causes ECC errors when data is read from a 256 bit wide RAM line, which was not initialized by writing before.

**Recommendation**

Initialize 256-bit EMEM RAM lines by writing to them. The full 32-byte address aligned line can be written in an arbitrary way. The minimum initialization is at least one write (e.g. a byte) in the lower 128-bit and one in the upper 128-bit part of the 256 bit wide RAM line.

**EMEM_TC.H003  Handling of Register TILECONFIGXM**

*Note: The following description only applies to ADAS and Extended SRAM versions of TC29x.*

The register description for the TILECONFIGXM register in the Extended Memory (EMEM) section of the TC29x User’s Manual is missing some bit fields. These are the bit fields corresponding to the following bit fields in an Emulation Device:

<table>
<thead>
<tr>
<th>Field</th>
<th>Bits</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>XTM0</td>
<td>[17:16]</td>
<td>w</td>
<td>see Note</td>
</tr>
<tr>
<td>XTM1</td>
<td>[19:18]</td>
<td>w</td>
<td>see Note</td>
</tr>
<tr>
<td>0</td>
<td>[31:20]</td>
<td>r</td>
<td><strong>Reserved</strong>; should be written with 0</td>
</tr>
</tbody>
</table>

*Note: User software should only write the value 01B to avoid interference with debug tool operation.*
**ENDINIT_TC.H001  Endinit Protection for Registers KRST0, KRST1, KRSTCLR**

The access protection symbol ‘E’ to indicate Endinit-protection is missing in column “Access Mode - Write” in table “Register Overview” in the User’s Manual for the following registers:

- KRST0, KRST1, KRSTCLR

of the following modules (if implemented):

- E-Ray, ETH, PSI5.

**ETH_AI.H001  Sequence for Switching between MII and RMII Mode**

When switching between MII and RMII mode is required, the ETH module must be clocked (MII: RXCLK and TXCLK; RMII: REFCLK) and be in a defined state to avoid unpredictable behavior.

Therefore, it is recommended to use the defined sequence listed below:

1. Finish running transfers and make sure that transmitters and receivers are set to stopped state:
   a) Check the RS and TS status bit fields in register ETH_STATUS.
   b) Check that ETH_DEBUG register content is equal to zero. Note: it may be required to wait 70 f_{SPB} cycles after the last reset before checking if ETH_DEBUG.RXFSTS is zero.

2. Wait until a currently running interrupt is finished and globally disable interrupts.

3. Apply kernel reset to ETH module:
   a) Deactivate Endinit protection, as registers KRST0/1 and KRSTCLR can only be written in Supervisor Mode and when Endinit protection is not active.

   Write to corresponding RST bits of KRST0/1 registers to request a kernel reset. The reset status flag KRST0.RSTSTAT may be cleared afterwards by writing to bit CLR in the KRSTCLR register.

   Re-activate Endinit protection.

   b) Wait 70 f_{SPB} cycles, then check if ETH_DEBUG.RXFSTS is zero.

4. Initialize the new mode (MII or RMII) in bit field GPCTL.EPR.
5. Apply software reset by writing to the ETH_BUS_MODE.SWR bit.
   Wait 4 $f_{\text{SPB}}$ cycles, then check if ETH_BUS_MODE.SWR = 0B.

**ETH_TC.H001 ETHMDIO on P21.1 not to be used for productive systems**

Unlike the other mapping options for signal ETHMDIO, P21.1 does not have the hardware direction control functionality to automatically switch the direction of ETHMDIO.

Therefore, do not use ETHMDIO on P21.1 in productive systems.

Instead, use the ETHMDIO mappings on P00.0, P12.1, or P21.3 (availability depends on device version and package pin count, see product documentation).

Changing the driver setting of pin P21.1 to speed grade 4 (in P21 Pad Driver Mode register) may allow the external PHY to overdrive the MDIO line. However, this configuration must not be used for productive systems.

**ETH_TC.H002 Minimum operation frequency for Ethernet MAC**

When using the Ethernet MAC module, $f_{\text{RAM}}$ must not be lower than 120 MHz.

**Recommendation**

Use $f_{\text{SPB}} \geq 60$ MHz and do not enable the Module Clock Divider, i.e. leave bit ETH_GPCTL.DIV = 0B (default after reset).

**ETH_TC.H003 Interrupt Generation by Wake-up or Magic Packet Frames**

In order to properly wake up by network (remote) wake-up frames or AMD Magic Packet, the SPB clock must not be switched off.

**Recommendation**

Therefore, keep the Module Disable Request Bit CLC.DISR = 0B.
FLASH_TC.H007 Advice for using Suspend and Resume

As documented in the User's Manual section “Operation Suspend and Resume”, an operation is suspended by writing ‘1’ to MARD.SPND. The Flash operation stops when it reaches an interruptible state. After that the flag FSR.SPND is set and BUSY is cleared.

The 1-to-0 transition of MARD.SPND alone is not indicating if the suspend request has been executed and the Flash can accept a new command. The BUSY flags have to be checked to determine if the Flash is still busy with the current operation. Only after the 1-to-0 transition of the BUSY flags the flag FSR.SPND indicates if the operation has finished or if it is in suspended state.

The following recipe describes the best practice for using suspend and resume.

Suspending an Erase Operation

In case of a request for suspending an ongoing erase operation:

As documented in the User's Manual: Please ensure that between start or resume of an erase process and the suspend request normally at least ~1 ms erase time can pass.

- Check if the corresponding BUSY flag has already cleared. If yes, no suspend is necessary.
- Request the suspend with control flag MARD.SPND = 1B.
- Wait until the BUSY flag clears.
- After that check FSR.SPND. If this is 1B then the operation was suspended and needs to be resumed later. If this is 0B the operation has already finished, therefore no resume is necessary.
- Now new Flash operations are allowed with the restrictions documented in User's Manual section “Operation Suspend and Resume”.

Note for PFlash erase operations in bank x that PxBUSY and D0BUSY are set at the beginning. The D0BUSY is cleared early after updating the Erase Counters, and PxBUSY is cleared when the erase operation has finished. Therefore, for PFlash the PxBUSY flag has to be used. (Polling for PxBUSY and DxBUSY can be a generic solution for suspend sequences before checking the SPND state.) Interrupt driven software receives two interrupts!
Resuming a Suspended Erase Operation

The resume of the suspended erase operation is done in these steps:

- Resume the operation with the command sequence “Resume Prog/Erase”.
- Wait until FSR.SPND is $0_B$.
- After that wait for the end of the operation signalled by BUSY going to $0_B$.

Suspending a Program Operation

In case of a request for suspending an ongoing programming operation:

- Request the suspend with control flag MARD.SPND = $1_B$.
- Wait until the BUSY flag clears.
- After that check FSR.SPND. If this is $1_B$ then the operation was suspended and needs to be resumed later. If this is $0_B$ the operation has already finished, therefore no resume is necessary.
- Now new Flash operations are allowed with the restrictions documented in User's Manual section “Operation Suspend and Resume”.

Resuming a Suspended Program Operation

The resume of the suspended programming operation is done in these steps:

- Resume the operation with the command sequence “Resume Prog/Erase”.
- Wait until FSR.SPND is $0_B$.
- After that wait for the end of the operation signalled by BUSY going to $0_B$.

FLASH_TC.H008 Understanding Flash Retention/Endurance Figures in the Data Sheet

Flash retention/endurance is documented in the Data Sheet by the following parameters:

- Program Flash Retention Time $t_{RET}$ for PFlash,
- UCB Retention Time $t_{RTU}$ for the UCBs,
- Data Flash Endurance per EEPROMx sector $N_{E,EPP10}$ for DFlash0,
- Data Flash Endurance per HSMx sector $N_{E,HSM}$ for DFlash1 (if available).
Retention
To emphasize the importance of retention, the PFlash and UCB parameters are described as retention time under the condition of a maximum number of cycles.

The value “Min. x years” has to be interpreted as: the data retention is at least x years, i.e. x years or longer after the last programming data stays readable.

The condition “Max. y erase/program cycles” means: this data retention figure is valid if there were not more than y erase/program cycles.

Endurance
For the DFlash the endurance is most important, therefore as parameter the number of cycles under the condition of the retention is given.

The value “Min. x cycles” has to be interpreted as: at least x cycles can be applied.

The condition “Max. data retention time y years” means: this endurance figure is valid if the expected data retention after the last programming is maximum y years.

Note: As general remark, these figures are only valid if the parameters given in the Data Sheet are adhered to in their entirety.

FLASH_TC.H009  PFlash Operations Concurrent to DF1 Operations

Note: The effect described in the following affects only devices with activated HSM.

In the PMU chapter, the User's Manual describes under the headline “Time Slice Control” in chapter “HSM Command Interface” how time slicing is used to share common Flash resources between operations by the host in DF0 and by the HSM in DF1.

In order to ensure isolation of HSM DF1 operations from the host side, the PMU doesn't prevent concurrent operations of PFlash and DF1. This mode is however considered as operation “out of specification”. The correct functionality of the operations is ensured but their timing can deviate from Data Sheet values.
Notably parameter “PFlash suspend to read latency” \( t_{SPNDP} \) may increase to \(~5\) ms. Due to time slicing the duration of PFlash operations is increased as described in the User's Manual for the DF0/DF1 concurrency.

**Recommendation**

Don't let HSM perform DF1 operations while the host performs Flash operations in the PFlash. In typical applications this is anyhow ensured because HSM needs to execute from its RAM during ongoing PFlash operations as these prevent read access to its code sectors.

**FlexRay_AI.H004  Only the first message can be received in External Loop Back mode**

If the loop back (TXD to RXD) will be performed via external physical transceiver, there will be a large delay between TXD and RXD. A delay of two sample clock periods can be tolerated from TXD to RXD due to a majority voting filter operation on the sampled RXD. Only the first message can be received, due to this delay. To avoid that only the first message can be received, a start condition of another message (idle and sampling '0' -> low pulse) must be performed.

The following procedure can be applied at one or both channels:

- wait for no activity (\( TEST1.AOx=0 \) -> bus idle)
- set Test Multiplexer Control to I/O Test Mode (\( TEST1.TMC=2 \)), simultaneously \( TXDx=TXENx=0 \)
- wait for activity (\( TEST1.AOx=1 \) -> bus not idle)
- set Test Multiplexer Control back to Normal signal path (\( TEST1.TMC=0 \))
- wait for no activity (\( TEST1.AOx=0 \) -> bus idle)

Now the next transmission can be requested.
**FlexRay_AI.H005** Initialization of internal RAMs requires one eray_bclk cycle more

The initialization of the E-Ray internal RAMs as started after hardware reset or by CHI command CLEAR_RAMS (SUCC1.COMD[3:0] = 1100B) 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 1B to 0B is correct.

**FlexRay_AI.H006** Transmission in ATM/Loopback mode

When operating the E-Ray in ATM/Loopback mode there should be only one transmission active at the same time. Requesting two or more transmissions in parallel is not allowed.

To avoid problems, a new transmission request should only be issued when the previously requested transmission has finished. This can be done by checking registers TXRQ1/2/3/4 for pending transmission requests.

**FlexRay_AI.H007** Reporting of coding errors via TEST1.CERA/B

When the protocol engine receives a frame that contains a frame CRC error as well as an FES decoding error, it will report the FES decoding error instead of the CRC error, which should have precedence according to the non-clocked SDL description.

This behaviour does not violate the FlexRay protocol conformance. It has to be considered only when TEST1.CERA/B is evaluated by a bus analysis tool.

**FlexRay_AI.H009** Return from test mode operation

The E-Ray FlexRay IP-module offers several test mode options

- Asynchronous Transmit Mode
- Loop Back Mode
• RAM Test Mode
• I/O Test Mode

To return from test mode operation to regular FlexRay operation we strongly recommend to apply a hardware reset via input eray_reset to reset all E-Ray internal state machines to their initial state.

Note: The E-Ray test modes are mainly intended to support device testing or FlexRay bus analyzing. Switching between test modes and regular operation is not recommended.

**FlexRay_TC.H002 Initialization of E-Ray RAMs**

After Power-on reset the ECC codes in the E-Ray RAMs may be set to an arbitrary state. Therefore the E-Ray RAM must be cleared and the ECC codes set to a defined state to avoid unintended traps.

To achieve this the following alternative methods are proposed:

**Method 1 using the MTU/MBIST:**

• Clear all E-Ray RAMs and the related ECC code storage by executing writes to all RAM locations using the AURIX MBIST engine. The MBIST engine supports filling the E-Ray RAM with ECC-correct patterns. For this purpose the AURIX MBIST auto-initialization algorithm can be used. See section “Filling a Memory with Defined Contents” in the corresponding User’s Manual/Target Specification.

The following E-Ray RAM blocks have to be initialized with correct data:

- Output Buffer
- Input Buffer
- Message Buffers

The MBIST function to be executed for each buffer is the same, only the function parameters have to be adapted.

• Execute one read from each E-Ray RAM block using the AURIX MBIST engine (reading from all E-Ray RAM locations is an alternative but not necessary solution). For this purpose the AURIX MBIST engine can also be used.
• Insert at the end of all MBIST function calls a status check, which makes sure that the launched MBIST tests are finished (check MSTATUS.DONE status flag).

• Clear all ECC error flags in the E-Ray module: these are flag EERR in register EIR, flags EIBF, EOBF, EMR, ETBF1, ETBF2 in register MHDS. The flags are cleared by writing a ‘1’ to the according bit position in the flag register.

After these steps the E-Ray RAM can be used for further operation, for example for initialization of the E-Ray buffer.

**Method 2 using “CLEAR RAMS” Command:**

Step 1 to 4: Enable the clock of the module:

• 1. Remove EINIT protection for the writing of the CLC register.
• 2. Enable the clock in the CLC register.
• 3. Read the CLC register.
• 4. Enable the EINIT protection.

Enable the test mode, check if the state of the module is according to the expected settings and start clearing the RAMs.

• 5. Take care of the unlock sequence. See description of LCK.TMK and TEST1.WRTEN in User’s Manual:
  – Test Mode Key: To set bit TEST1.WRTEN the write operation has to be directly preceded by two consecutive write accesses to the Test Mode Key.
  – If the write sequence is interrupted by other write accesses between the second write to the Test Mode Key and the write access to the TEST1 register, bit TEST1.WRTEN is not set to 1 and the sequence has to be repeated.
    First write: LCK.TMK = 75\text{H} = 0111 0101\text{B}
    Second write: LCK.TMK = 8A\text{H} = 1000 1010\text{B}
    Second write: TEST1.WRTEN = 1\text{B}
• 6. Check if CCSV.POCS is either 0x0 (DEFAULT_CONFIG) or 0xF (CONFIG). If not in any of these states, perform the according command to get to CONFIG state.
• 7. Check if SUCC1.PBSY is equal 0x0. If 0x1 wait until 0x0.
• 8. Set SUCC1.CMD to 0xC meaning that the CLEAR_RAMS command is entered.
• 9. Read SUCC1.CMD. If 0x0 the command has not been accepted. Repeat up from step 7. Otherwise continue.
• 10. Wait 1024 module cycles.
• 11. Enable RAM Test mode: TEST1.TMC = 01B. This mode enables access of all RAM blocks in E-Ray modules to the host.
• 12. CUST1.IBF1PAG := 1B
• 13. CUST1.IBF2PAG := 1B.
• 14. Repeat steps 7 to 10.
• 15. Read at least one address in all the RAM blocks within E-Ray module.
• 16. Switch off Test mode: TEST1.WRTEN = 0B
• 17. Clear ECC error flags in MHDS and EIR registers
• 18. From here you can start the normal initialization process of the module.

**FPI_TC.H002 Write Access to Register ACCEN1**

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

*Note: For a write access to the ACCEN1 registers in the following modules, a bus error will be generated: MTU, SMU, ETH, I2C, FFT, CIF.*

**GPT12_TC.H001 Timer T5 Run Bit T5R - Documentation Correction**

In the current version of the User’s Manual, the lines for T5R=0B and T5R=1B in the register description of the Timer T5 Run Bit (T5R) erroneously have been swapped.

**Correction**

The correct behavior of bit T5R is as shown in Table 36: T5R=0B (Timer T5 stops; default after reset), T5R=1B (Timer T5 runs).
**Table 36** Timer T5 Control Register T5CON, Bit T5R - Correction

<table>
<thead>
<tr>
<th>Field</th>
<th>Bits</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>T5R</td>
<td>6</td>
<td>rw</td>
<td>Timer T5 Run Bit</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0&lt;sub&gt;B&lt;/sub&gt; Timer T5 stops</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1&lt;sub&gt;B&lt;/sub&gt; Timer T5 runs</td>
</tr>
</tbody>
</table>

*Note: This bit only controls timer T5 if bit T5RC = 0.*

**GTM_TC.H003** Typo: GTM Chapters “Multi Channel Sequencer (MCS)” and “Memory Configuration (MCFG)” sometimes use “MSC” instead of “MCS”

Due to an editorial issue, in chapters “Multi Channel Sequencer (MCS)” and “Memory Configuration (MCFG)” sometimes the term “MSC” is erroneously used instead of “MCS” within the text and in figure/section titles.

*Note: The register names defined in these chapters are correct, such that register definition files extracted from these chapters will include the correct register names.*

**GTM_TC.H004** Correction to Bit Fields GTM_TIMi_IN_SRC.VAL_x

In the description of bit field VAL_0 in register GTM_TIMi_IN_SRC in the User’s Manual, the encoding 01<sub>B</sub> was erroneously repeated while 10<sub>B</sub> and 11<sub>B</sub> were missing.

The correct description is included in the following Table 37. As the description of bit fields VAL_x, x>0 refers to VAL_0, this description is valid for all VAL_x bit fields in registers GTM_TIMi_IN_SRC.
In table “TIM integration Mode” in section “External Capture in TIM Pulse Integration Mode (TPIM)” of the GTM chapter in the User’s Manual, the information that CNT is cleared upon external capture is missing in column “Action description”.

The corrected Table 38 is shown below:

<table>
<thead>
<tr>
<th>Input signal F_OUTx</th>
<th>selected CMU clock</th>
<th>External capture</th>
<th>ISL</th>
<th>DSL</th>
<th>Action description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>-</td>
<td>0</td>
<td>CNT++</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>-</td>
<td>0</td>
<td>no</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>-</td>
<td>1</td>
<td>CNT++</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>-</td>
<td>1</td>
<td>no</td>
</tr>
<tr>
<td>-</td>
<td>-</td>
<td>rising edge</td>
<td>-</td>
<td>-</td>
<td>do GPRx capture; issue NEWVAL_IRQ; CNT = 0</td>
</tr>
<tr>
<td>-</td>
<td>0</td>
<td>0</td>
<td>-</td>
<td>-</td>
<td>no</td>
</tr>
</tbody>
</table>
GTM_TC.H006  AEI to FIFO Data Interface (AFD1) Registers

The description of the registers GTM_AFD1_CHx_BUF_ACC (x=0-7) of the second instance (AFD1) of the AEI to FIFO Data Interface (AFD) is missing in the GTM chapter of the User’s Manual.

The registers of AFD1 and their function are identical to the registers of AFD0. The only difference is the difference in the offset address by 0x4000 as described in Table “Address Overview” in the GTM chapter of the User’s Manual (see extract in Table 39 below):

Table 39  Address Overview - AFD Submodules

<table>
<thead>
<tr>
<th>Submodule</th>
<th>Base address (to be added to GTM based address)</th>
</tr>
</thead>
<tbody>
<tr>
<td>AFD0</td>
<td>0x00018000</td>
</tr>
<tr>
<td>AFD1</td>
<td>0x0001C000</td>
</tr>
</tbody>
</table>

GTM_TC.H007  GTM to CAN Timer Triggers

The CAN transmit trigger inputs of the individual CAN nodes are connected to GTM trigger outputs as specified in table “CAN Transmit Trigger Inputs” in the MultiCAN+ chapter of the User's Manual.

The corresponding GTM TOM/ATOM channel is selected in register GTM_CANOUTSEL as specified in tables “CAN Timer Triggers” in the GTM chapter. Note that not all specified SELx bit fields in register CANOUTSEL are used for trigger selection.

The following GTM to CAN connections are implemented:

Table 40  GTM to CAN Connections in TC29x

<table>
<thead>
<tr>
<th>CAN Node</th>
<th>GTM Trigger Selection via Bit Field</th>
</tr>
</thead>
<tbody>
<tr>
<td>CAN Node 0</td>
<td>CANOUTSEL.SEL0</td>
</tr>
<tr>
<td>CAN Node 1</td>
<td>CANOUTSEL.SEL1</td>
</tr>
<tr>
<td>CAN Node 2</td>
<td>CANOUTSEL.SEL2</td>
</tr>
<tr>
<td>CAN Node 3</td>
<td>CANOUTSEL.SEL3</td>
</tr>
</tbody>
</table>
Table 40  GTM to CAN Connections in TC29x (cont’d)

<table>
<thead>
<tr>
<th>CAN Node</th>
<th>GTM Trigger Selection via Bit Field</th>
</tr>
</thead>
<tbody>
<tr>
<td>CANR Node 0</td>
<td>No GTM trigger</td>
</tr>
<tr>
<td>CANR Node 1</td>
<td>No GTM trigger</td>
</tr>
</tbody>
</table>

**GTM_TC.H008  Correction to Figure “SPE to TOM Connections”**

In figure “SPE to TOM Connections” in the GTM chapter of the TC29x User’s Manual, the signal originating from block TOM_CH2 is incorrectly shown as TOM[i]_CH2_TRIG_CCU0.

The correct signal originating from block TOM_CH2, as shown in figure “SPE Submodule architecture” and documented in the description of register GTM_SPEi_CTRL_STAT for TRIG_SEL = 11B, is TOM[i]_CH2_TRIG_CCU1.

**GTM_TC.H011  First CM0 updates in case of SR0=1 and (A)TOM used as Triggered Channel**

In case the CM0 register should be updated from the shadow register with 1, the Force Update mechanism (FUPD(x) signal) has to be enabled on the (A)TOM channel. Otherwise the first edge triggered from CM0 will not be generated after 1 appears in CM0.

**GTM_TC.H014  Synchronous Bridge Mode Restrictions**

The reset value for register GTM_BRIDGE_MODE is specified as 0400 1001H, and should never be changed according to the User’s Manual, i.e. the AEI bridge should always operate in async_bridge mode.

**Exception**

In order to improve access latency, operation in synchronous bridge mode is possible if it is ensured that the SPB frequency is identical to the GTM frequency:
**f_{SPB} == f_{GTM}**

Sequence to configure the bridge in synchronous mode (pseudocode):

```c
/* ensure that no data are read or written in the GTM */
if (f_{SPB} == f_{GTM})
{
    GTM_BRIDGE_MODE = 0x04011000; /* switch to sync mode, reset bridge*/
    while (GTM_BRIDGE_MODE & 0x100) /* wait till mode change completed */
    {
    }
} else
;
```

**GTM_TC.H015 Register TIMi_CHx_CTRL - Correction to Register Image**

The register image of register TIMi_CHx_CTRL (i=0) erroneously shows bit 19 as “Reserved” with type “r” (read only).

The register image of register TIMi_CHx_CTRL (i>0) erroneously shows bit 19 with type “r” (read only).

**Correction**

Actually, bit 19 has type “rw” and is correctly described in the register table as copied from the User’s Manual in **Table 41** below:

<table>
<thead>
<tr>
<th>Field</th>
<th>Bits</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>EXT_CAP_EN</td>
<td>19</td>
<td>rw</td>
<td>Enables external capture mode</td>
</tr>
</tbody>
</table>

The selected TIM mode is only sensitive to external capture pulses, the input event changes are ignored.
GTM_TC.H016 Evaluating DSADC Signals SAULx/SBLLx

The DSADC provides the following signals to indicate whether the results of the parallel filter are outside the limits defined in the BOUNDSELx registers:

- Signal SAULx is active while the results are above the upper limit,
- Signal SBLLx is active while the results are below the lower limit.

In the GTM, these signals can be selected as TIM inputs in the DSADCINSEL register(s).

The rising edge of these signals is transformed to a pulse inside the GTM, so that it is not possible to directly measure the high/low phase of SAULx/SBLLx via TIM, but the duration between two rising edges.

GTM_TC.H017 Bit DXINCON.24 (DSS10) - Documentation Correction

In the register image of register DXINCON in the User's Manual, bit 24 is erroneously named DSS00 instead of DSS10.

Correction

The correct symbolic name for bit DXINCON.24 is DSS10, as listed in the table below the register image in the User's Manual:

<table>
<thead>
<tr>
<th>Field</th>
<th>Bits</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DSS1x</td>
<td>x+24</td>
<td>rw</td>
<td>Data Source Select 1x Control</td>
</tr>
</tbody>
</table>

For TC29x, n = 5.
Note: The SFR C Header Definitions are not affected, as they are generated from the table (not from the register image).

**GTM_TC.H018** TIM4 Mapping for CH7SEL = 1010B - Correction

In tables “TIM4 Mapping for BGA-nnn” (nnn = 292, 416, 516) in the GTM chapter of the User’s Manual, for setting CH7SEL = 1010B erroneously P02.11 is listed in column “Pad / Input”.

**Correction**

In this device step, for TIM4, setting CH7SEL = 1010B is “Reserved”:

<table>
<thead>
<tr>
<th>CH7SEL</th>
<th>Pad / Input</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>1010B</td>
<td>Reserved</td>
<td>-</td>
</tr>
</tbody>
</table>

Note: The connections for P02.11 listed in other parts of the TC29x documentation (e.g. tables “GTM to Port Mapping” of the GTM chapter, in the Ports chapter, and in the TC29x Data Sheet) are correct.

**HSCT_TC.H003** Functionality of bit **TX_PWDPD**

Bit **TX_PWDPD** in register **P21_LPCR2** directly disables or enables the LVDS pull down.

The application software must disable the TX power down pull down after power-up. With a LVDS power down configuration, the pull down function must be enabled, if required.

**HSCT_TC.H005** Access to reserved address 0xF009 0060 when f_SPB = f_SRI

Unlike an access to other reserved addresses within the HSCT, an access a_x to address 0xF009 0060 will not result in a bus error when f_SPB = f_SRI.
If another HSCT access $a_y$ follows back-to-back to $a_x$, a bus error will be generated for $a_y$, even if the access is to a valid address.

Note: With the default reset value of register CCUCON0 = 0202 0112H, i.e. $f_{SRI} = 2 \times f_{SPB}$, these effects will not occur.

**HSCT_TC.H007  HSSL Integrated Phase Noise**

The diagram below shows the phase noise characteristics at the SysClk output of the HSCT PHY in the master device.

- The Integrated Phase Noise $I_{PN}$ limit is violated. Target value is:
  - $I_{PN} = -58$ dBc, corresponding to $J_{ABS20} = 14$ ps

- The achieved value with max power pattern running on the master microcontroller is:
  - $I_{PN} = -43$ dBc, corresponding to $J_{ABS20} = 80$ ps

- The achieved value with no application running on the master microcontroller is:
  - $I_{PN} = -49$ dBc, corresponding to $J_{ABS20} = 40$ ps

Nevertheless, such a target value on the random jitter of the SysClk signal is only an intermediate specification. The real target to be respected in order to assure $BER_{20}$ of $10^{-12}$ is the total jitter of the link. The total jitter target is met. Consistently, measurements on the HSSL/HSCT communication channel using the SysClk signal with $I_{PN}$ from above show that the ultimate target of $BER_{20} = 10^{-12}$ is met. In such a measurement setup, both the master and the slave are microcontrollers of the AURIX™ family.

The diagram below shows the phase noise density on the SysClk pin when no application pattern but only the HSSL/HSCT subsystem is running:

---

1) integration range from 10 kHz to 10 MHz
**Figure 8  Phase Noise Density on SysClk Output Pin**

**Recommendation**

When designing the PLL for an HSSL/HSCT ASIC, the values for $I_{PN}$ and $J_{ABS20}$ above and ultimately $BER_{20} = 10^{-12}$ can be achieved with the following recommendations:

- DCO frequency = 640 MHz
- DCO free running phase noise @1 MHz ≤ -98 dBc
- PLL bandwidth = 1 MHz

Comprehensive information on this topic is provided in Application Note AP32292 “HSCT Jitter Considerations”.

---

1) the bandwidth can go as low as 200 kHz, at which point the DCO noise exceeds the reference noise
**HSCT_TC.H008  Details on PLL Lock-in Time**

The HSCT parameter “PLL lock-in time” $t_{\text{LOCK}} \leq 50 \, \mu\text{s}$ defined in the Data Sheet refers to the pure HSCT PLL lock time not including the internal voltage regulator (IVR) start-up time $t_{\text{IVR}}$.

In case of Master Mode the PLL reference clock is SysClk = 20 MHz. The total PLL lock time including the IVR start-up time is $\leq 70 \, \mu\text{s}$.

In case of a HSCT slave receiving SysClk = 20 MHz, the total PLL lock time including the IVR start-up time is $\leq 70 \, \mu\text{s}$.

In case of a HSCT slave receiving SysClk = 10 MHz, the PLL lock time is longer and the total time including the IVR start-up time is $\leq 100 \, \mu\text{s}$ (see Table 44).

**Table 44  Total PLL Lock-in Time for SysClk = 20 MHz and SysClk = 10 MHz**

<table>
<thead>
<tr>
<th>SysClk</th>
<th>Total PLL Lock-in Time (max.)</th>
</tr>
</thead>
<tbody>
<tr>
<td>20 MHz</td>
<td>Master: 70 µs  Slave: 70 µs</td>
</tr>
<tr>
<td>10 MHz</td>
<td>not applicable 100 µs</td>
</tr>
</tbody>
</table>

**I2C_TC.H001  I2C Module Behavior in OCDS Suspend Mode**

The register bit CLC1.FSOE selects between Secure Clock Shut Off (FSOE=0B) and Fast Clock Shut Off (FSOE=1B) when entering the OCDS Suspend Mode.

In the current implementation, the behavior of the module upon an OCDS suspend request is as follows:

- **In master** mode, where the master generates the clock, the I2C module always stops immediately, independent of the setting of bit FSOE. The bus lines (SDA and SCL) are stalled, and it is likely that I2C protocol (e.g. SCL_LOW / SCL_HIGH) is not fulfilled in this case.
- **In slave** mode with FSOE=0B, the I2C module stops after an acknowledge has been sent on the bus, and then drives SCL low to delay operation.
- **In slave** mode with FSOE=1B, the I2C module immediately stops in any state, without generating/waiting for an acknowledge.
Recommendation
It is recommended to reset the system after the clock shut off and restart the I2C sequence again for further debugging and analysis.

I2C_TC.H002 Initialization of INC/DEC values in Slave mode

Baudrate generation is mainly used for master mode, but there is one corner case where a support of the baudrate generation is required in slave mode: when I2C is in slave mode with a pending transmit data request and FIFO is currently empty, then SCL is kept low until FIFO is filled after some time. Then transmit data is sent out immediately. SCL is kept low for a min. time of $t_{SU;DAT}$.

Recommendation
It is recommended to program INC/DEC in slave mode also with values to ensure a suitable setup time for transmit data according $t_{SU;DAT}$ of I2C standard. The parameter $t_{SU;DAT}$ is represented in the Infineon TC2xx Data Sheets as $t_4$. In order to keep the set-up time, $t_4$, according to I2C standard, the baudrate generation is used to guarantee the delay on SCL. The formulas in the 3rd and the 4th column of Figure 9 show how to configure DEC and INC for a given $f_{I2C}$ and the intended set-up time. The formulas in column Resulting $t_4$ [µs] define the real value of $t_4$ with the selected DEC and INC values.

The same value for DEC and INC as in master mode is not recommended and would lead to additional delay, as requested by the standard, in most cases.
Figure 9  I2C Baudrate Generation Configuration for Slave Mode

In the following table some example settings are given to program INC / DEC. The used abbreviation INT is the integer function.

Table 45  INC/DEC Settings for Slave Mode

<table>
<thead>
<tr>
<th>kernel _clk [MHz]</th>
<th>Standard Mode</th>
<th>Fast Mode</th>
<th>High-Speed Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>INC</td>
<td>DEC</td>
<td>t₄</td>
</tr>
<tr>
<td>10</td>
<td>75</td>
<td>250</td>
<td>275 ns</td>
</tr>
<tr>
<td>50</td>
<td>20</td>
<td>249</td>
<td>260 ns</td>
</tr>
<tr>
<td>100</td>
<td>10</td>
<td>240</td>
<td>260 ns</td>
</tr>
</tbody>
</table>

I2C_TC.H003  DMA Channel Configuration

The I2C module expects an acknowledge from the DMA after a data transfer to FIFO/from FIFO. But the DMA implemented in AURIX does not provide the
acknowledge. If the CPU would have to provide it, this would make the use of a DMA pointless.

**Recommendation**

A linked list should be used to avoid that the CPU has to get active to provide the acknowledge. The DMA channel that serves I2C has to be configured with a linked list that first clears the interrupt by writing 0x0 to register ICR (Interrupt Clear Register) and second makes the FIFO-TX/RX transfer.

**I2C TC.H004 Transfers of more than 32 Bytes**

When the I2C module FIFO isn't serviced in time (send: filled with transmit data, receive: read the received data), an underflow/overflow event will lead to (TX_END) abortion of the transmission, like specified in the User's Manual.

**Recommendation**

To avoid this behaviour the software shall be configured to transfer a maximum of 32 bytes per transfer.

If more than 32 bytes should be transmitted, the transmissions should be divided in maximum 32 data bytes per transfer.

**I2C TC.H005 FIFO Data is lost during Transaction RX->TX**

When the I2C module is changing the state from receive to transmit mode, the FIFO is “flushed”. This is needed in many cases due to the half duplex nature of the FIFO.

If the software does not proceed in the right sequence, this “flush” can lead to data loss.

**Recommendation**

To avoid loosing data when the FIFO is “flushed” the software should proceed as follows:
In a scenario where the device is addressed as slave and is asked to return data, this new data must be entered in the FIFO only after detection of the address and “end” indication, so the software shall wait for AM (Address Match) and TX_END (Transmission End) interrupt requests and then can transfer the data to the FIFO or can trigger the DMA that fills the FIFO for the TX transfer.

**I2C_TC.H008 Handling of RX FIFO Overflow in Slave Mode**

If the I2C kernel has detected a RX FIFO overflow in slave mode, a RX_OFL_srq request is generated, the incoming character is discarded, and the kernel puts a not-acknowledge on the bus and changes to listening state. However, it does not generate an EORXP_ind signal, so that the remaining characters in the FIFO can not be moved out by means of data transfer requests.

**Recommendation**

Upon an RX FIFO overflow in slave mode, received data may be invalid. However, they may be read from the FIFO e.g. for analysis if required.

In order to flush the FIFO and correctly resume communication

- set bit RUNCTRL.RUN = 0B (switch to configuration mode),
- set bit RUNCTRL.RUN = 1B (participate in I2C communication).

**IOM_TC.H001 How to clear the IOM_LAMEWCm register**

The Logic Analyzer Module Event Window Count Status register IOM_LAMEWCm stores the window count value reached prior to being cleared in the LAM block once an event has been generated.

Writing to IOM_LAMEWCm by software will result in a bus error.

The IOM_LAMEWCm register can be reset (cleared) by software with a write to the IOM_LAMCFGm or IOM_LAMEWSm registers, e.g. by writing the same configuration data that have been read to either of these registers.
Note: The clock divider should be set to IOM_CLC.RMC = 1 when configuring the IOM (see issue IOM_TC.004 “Write to IOM register space when IOM_CLC.RMC > 1”).

IOM_TC.H002  IOM Clock Control

Contrary to the named clocks given within the subsections of the IOM chapter, the entire IOM operates at the higher of the SPB or GTM clock frequencies. This may be further divided via the RMC bit field of the IOM_CLC register, where the physical RMC value represents the divisor. For example, RMC = 00000001_B divides clock by 1, RMC = 00000010_B divides clock by 2, and so on. Note that RMC = 00000000_B disables the clock.

See also the following revised description of the IOM_CLC register.

IOM Clock Control Register (IOM_CLC)

The Clock Control Register CLC allows the programmer to adapt the functionality and power consumption of the module to the requirements of the application. The description below shows the clock control register functionality which is implemented in the BPI_FPI for the module. Where a module kernel is connected to the CLC clock control interface, CLC controls the f_IOM module clock signal, sleep mode and disable mode for the module.

Table 46 Description of Fields in IOM Clock Control Register (IOM_CLC)

<table>
<thead>
<tr>
<th>Field</th>
<th>Bits</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DISR</td>
<td>0</td>
<td>rw</td>
<td>Module Disable Request Bit</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Used for enable/disable control of the module.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0_B Module disable is not requested</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1_B Module disable is requested</td>
</tr>
<tr>
<td>DISS</td>
<td>1</td>
<td>rh</td>
<td>Module Disable Status Bit</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Bit indicates the current status of the module.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0_B Module is enabled</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1_B Module is disabled</td>
</tr>
</tbody>
</table>
As shown in figure “Logic Analyzer Module (LAM) block diagram” in the IOM chapter of the User’s Manual, an EVENT will be generated if the required edge is detected and the XOR between the Event Window value and the invert bit (LAMCFG.IVW) is 1.

When the edge to be detected arrives at LAMEWSn.THR value of the counter, the EVENT will be generated depending on LAMCFG.IVW value:

- If LAMCFG.IVW==0 event will be generated,
- if LAMCFG.IVW==1 event will not be generated.
Taking this behavior into account, the description of the LAMCFG.IVW and/or LAMEWS.THR configuration in examples 2, 4, 5 and 6 of section “Example Monitor/Safety Measures” is misleading.

**Correction**

The corrected description, including the case “equal to”, is as follows (only modified lines are printed):

**Example 2 - Pulse or duty cycle too long**

LAMCFG.IVW: 0x0 ; don’t invert window, capture events when the counter is equal or above the threshold.

LAMEWS.THR: select appropriate threshold (maximum duty cycle length required. If duty cycle is longer than this value then an event will be triggered).

**Example 4 - Period too long**

LAMCFG.IVW: 0x0 ; don’t invert window, capture events when the counter is equal or above the threshold.

LAMEWS.THR: select appropriate threshold (maximum period length required. If period is longer than this value then an event will be triggered).

**Example 5 - Diagnosis of Command and Feedback - acceptable propagation window and/or signal consistency check**

LAMCFG.IVW: 0x0 ; don’t invert window, capture events when the counter is equal or above the threshold.

LAMCFG.THR: set to max delay allowed (if the delay between corresponding edges of reference and monitor signals is longer than this value, the event will be triggered).

**Example 6 - Diagnosis of Set-up and Hold times**

- **Example settings for LAM block registers for Set-up**
  LAMCFG.IVW: 0x0 ; don’t invert window, capture events when the counter is equal or above the threshold.

- **Example settings for LAM block registers for Hold**
LAMCFG.IVR: 0x1; invert reference signal (use for gating).
LAMCFG.THR: Acceptable Hold (ref Threshold 2 on waveforms shown, changes in monitor signal will generate an alarm if they occur inside the “THR” cycles after a falling edge in the reference signal).

**IOM_TC.H004  Behavior of LAMEWCn.CNT when LAMEWSn.THR is 0**

When LAMEWSn.THR is set to 0, no event will be sent from the Logic Analyzer Module (LAM) to the Event Combiner Module (ECM) and no ALARM towards the SMU will be generated.

The rest of the effects derived from the cause generating the event inside the LAM will be maintained, for instance copying the counter to LAMEWCn.CNT (this means LAMEWCn.CNT also may change when LAMEWSn.THR is 0).

**IOM_TC.H006  ACCEN* Protection for Write Access to IOM Registers**

The access protection symbol ‘P’ to indicate protection by the ACCEN* register mechanism is missing in column “Access Mode - Write” in table “Register Overview” in the User’s Manual for IOM registers with an offset address ≥ 30H.

Actually, these registers have write access attributes ‘U,SV,P’.

**Exception**

In this design step, a write access to register LAMEWCm will result in a bus error, as correctly reflected by symbol ‘BE’ in column “Access Mode - Write” in table “Register Overview” in the User’s Manual.

**IOM_TC.H007  Write Access to FPCESR**

The Filter and Prescaler Edge Status Register FPCESR stores the state of detected rising and falling edges from each of the Filter and Prescaler Channels k (k = 0..15).

The flags in this register can be selectively cleared by writing a 0 in the respective bitfield.
However, writing to register FPCESR with a sub-word granularity (e.g. byte or half-word) leads to undefined behavior.

**Recommendation**

Individual bits for channel k in FPCESR are cleared with a write to the control register (FPCCTRk) or timer register (FPCTIMk).

Writing to FPCESR directly shall be done always to the whole register (32-bit writes), with bits that should not be modified set to 1B.

In particular, LDMST or SWAPMSK.W should be used only with bit mask enabled for all ‘rwh’ bits in register FPCESR.

**LMU TC.H002 On-the-fly BBB:SRI clock ratio switching**

*Note: This problem only occurs in an ADAS or Emulation Device (ED), but may already need to be considered during software development for the target device.*

When switching the clock ratio for fBBB relative to fSRI, make sure that no MMES (Memory Mapped Emulation System) access to EMEM is performed by an SRI master via the LMU. Otherwise, data read/written may be incorrect.

**Recommendation**

After a MMES read is complete, allow at least 12 SRI clock cycles before initiating a clock ratio change.

After a MMES write is complete, allow at least 20 SRI clock cycles before initiating a clock ratio change.

After a clock ratio change, allow the clock ratio change to become effective before performing any MMES transfer (e.g. read back control register that was written for the clock ratio change).
**LMU_TC.H003 Function of Bit MEMCON.PMIC (Protection Bit for Memory Integrity Control Bit)**

In the LMU chapter of the User’s Manual, the following text (last paragraph in section “Local Memory (LMU SRAM)”) is incorrect: Some bitfields of the LMU_MEMCON register are protected by LMU_MEMCON.PMIC bit. If the data written to the register has the bitfield set to 0B, no change will be made to bits 15D to 9D of the register regardless of the data written to these fields.

**Correct Description**

For the correct description (only bit 9 (ERRDIS) is protected) see the description of bit PMIC in the LMU Memory Control Register in section “LMU Registers”, copied in **Table 47** below:

<table>
<thead>
<tr>
<th>Field</th>
<th>Bit</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| PMIC  | 8   | w    | Protection Bit for Memory Integrity Control Bit  
Will always return 0B when read  
0B Bit Protection: Bit 9 remains unchanged after  
LMU_MEMCON write.  
1B Bit 9 will be updated by the current write to  
LMU_MEMCON |
| ERRDIS| 9   | rw   | ECC Error Disable  
When set SRI bus errors caused by ECC errors in data read from the SRAM will be disabled.  
... |

**LMU_TC.H004 FFT Accelerator Interface**

*Note: This is a documentation problem, only relating to ADAS devices of TC29x and TC26x when used in combination with User’s Manual V1.3.*

*Note: This issue might affect the SFR C Header Definitions. In such cases, SFR usage in the software shall be analyzed within the applications for their correct handling.*
A problem during document generation has resulted in the following text being omitted from the LMU Chapter of the User Manual V1.3:

- Chapter “FFT Accelerator Interface”
- Description of bit 10 (FFTPFT) in register MEMCON (see Table 48)

These parts are included below.

11.4 FFT Accelerator Interface

The LMU maps the resources of the hardware based FFT accelerator into the system address space. For full information on using the FFT accelerator, see the dedicated chapter of this specification.

Accesses to the registers and the data load/unload ports of the FFT accelerator are handled separately.

The LMU provides two functions related to the data load/unload ports:

- Any access to the data load/unload ports which is not completed by the FFT accelerator will time out after 255 clock cycles.
- Read accesses to the data unload port are augmented by a prefetch function. Any read access using a BTR4, SRI opcode will automatically trigger another access of the same size to the next contiguous address. The prefetch is enabled by writing bit 10 (FFTPFT) of the MEMCON register.

<table>
<thead>
<tr>
<th>Field</th>
<th>Bits</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>FFTPFT</td>
<td>10</td>
<td>rw</td>
<td>FFT Accelerator Prefetch Disable</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0B</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1B</td>
</tr>
</tbody>
</table>
**MSC_TC.H010 Configuration of SCU.EMSR for the EMGSTOPMSC Signal**

The emergency stop input signal EMGSTOPMSC of the MSC module is connected to the output signal of the emergency stop control logic located in the SCU. Its functionality is controlled by the SCU emergency stop register SCU.EMSR.

The emergency stop input line EMGSTOPMSC is used to indicate an emergency stop condition of a power device. In emergency case, shift register bits can be loaded bit-wise from the downstream data register instead from the ALTINL and ALTINH buses.

The emergency stop frame is sent out at each trigger event as long as the emergency stop signal is active. This means that in data repetition mode the emergency stop frame is repeatedly sent as long as the emergency stop signal is active.

*Note:* If the emergency stop signal is used by the MSC module with setting SCU.EMSR.MODE = 1B (Asynchronous Mode), there is some low probability that the first emergency stop frame could be corrupted, but the following emergency stop frames will be correct.

**Recommendation**

- If the emergency stop signal is used by the MSC module, setting SCU.EMSR.MODE = 0B (Synchronous Mode) is mandatory.
- Setting SCU.EMSR.MODE = 1B (Asynchronous Mode) is not allowed to be used with the MSC module.

**MSC_TC.H011 Effect of kernel reset on MSC0_FCLP when selected in Event Trigger Logic**

If a kernel reset of the MSC0 module is performed, and signal MSC0_FCLP is selected to trigger an event via the Event Trigger Logic (ETL), an unintended trigger may be generated.
Recommendation

Disable edge detection for both the rising and the falling edge of MSC0_FCLP in the SCU, i.e. set bits EICR0.REN0 = 0B and EICR0.FEN0 = 0B, before performing the kernel reset on MSC0.

After the MSC0 kernel reset, restore the intended settings in register EICR0 as part of the MSC re-initialization.

MSC_TC.H012 Handling the overflow interrupt of the ABRA block

The configuration of the ABRA block and the MSC kernel is static and the timing behavior is deterministic. Therefore, an overflow/underflow event primarily signals some configuration error resulting in an unadjusted input/output baud rate ratio of MSC and ABRA. In normal operation with correctly configured baud rates this error mechanism does not occur.

However, due to an internal synchronisation problem, in very rare cases an overflow interrupt might occur with the overflow flag ABC.OVF set to 1B despite correct configuration (baud rate ratio, length of passive phase, ...).

Recommendation

The probability of the synchronisation problem is low enough to allow evaluation of the overflow interrupt and the overflow flag ABC.OVF during the software development and debugging phase to identify incorrect MSC/ABRA configurations.

In the final software implementation, disable the overflow interrupt via bit ABC.OIE = 0B, and do not evaluate the overflow flag ABC.OVF.

MSC_TC.H013 Empty Data Frames not supported with ABRA

When using the Asynchronous Baud Rate Adjustment block (ABRA), transmission of empty data frames (consisting only of a selection bit and no data bits) is not supported.

In this corner case, enable signals (ENL, ENH) may not be correctly activated, and a SYNC FIFO underflow may erroneously be signalled.
**Recommendation**

Do not use configurations where empty data frames (DSC.NDBL/NDBH = 00000B) are sent in combination with ABRA.

**MTU TC.H003  AURIX™ Memory Tests using the MTU**

The use of destructive tests such as March-U and Checkerboard etc. in conjunction with FAILDMP mode to get detailed failure information (errors, fail addresses) will cause the SRAM redundancy information to be overwritten. Therefore, the MTU/MBIST module effectively only supports the Non-Destructive Inversion Test (NDIT).

**Recommendation**

To avoid overwriting the SRAM redundancy information, only use Non-Destructive Inversion Test. In this case, failure is detected by ECC and the detailed information can be obtained from ETRR and ECCD registers.

Refer to the latest version of Application Note AP32197 “AURIX™ Memory Tests using the MTU” for more details on MTU/MBIST usage and fault coverage.

**MTU TC.H004  Handling the Error Tracking Registers ETRR**

CPU and on-chip peripheral SRAMs are capable of detecting errors and generating SMU alarms for correctable, uncorrectable, and address errors. The failing addresses are stored in Error Tracking Registers (ETRR), and the corresponding indicator (CERR/UERR/AERR and SERR) and valid bits (VAL) are set in the Memory ECC Detection Register (ECCD). Only new errors will be considered, i.e. errors at already stored addresses will be ignored. In case the maximum number of ETRR for a memory is used up and a new error occurs, the error overflow bit ECCD.EOV is set, and the corresponding “address buffer overflow” SMU alarm is generated. For peripheral SRAMs, the second error will cause a buffer overflow, and for CPU SRAMs, up to five errors can be registered before the buffer overflow alarm is triggered.
Bit ECCD.TRC (Tracking Clear) allows to clear the EOV and VAL bits in register ECCD and the associated ETRR registers, e.g. in response to a tolerated corrected single bit error.

Corner Case

If in an exceptional corner case software would set TRC at the same time an error overflow occurs, then the EOV bit is not set, and the SMU alarm is not generated.

Recommendation

• It is not necessary to clear the Error Tracking Registers ETRR by software as part of an SRAM error handling concept. For correctable errors, the application software should only react on the address buffer overflow alarm (e.g. with a reset). Single correctable error events may be ignored (within limits) to increase the fault tolerance of the system without impacting the safety.
• If a different concept is used requiring clearing of the ETRR registers by software via ECCD.TRC, make sure that the corresponding SRAM instance is not functionally accessed while the application software writes ECCD.TRC, so that an overflow error cannot be generated during the clear operation.

Information on using the MTU for memory diagnosis is given in Application Note AP32197 “AURIX™ Memory Tests using the MTU”.

MTU_TC.H005 Handling SRAM Alarms

Alarms are generated for CPU and on-chip peripheral SRAMs when correctable, uncorrectable, and address errors are detected. The failing addresses are stored in Error Tracking Registers (ETRR), and information on the error type is stored in the Memory ECC Detection Register (ECCD). Only new errors will be considered, i.e. errors at already stored addresses will be ignored. In case the maximum number of ETRR for a memory is used up and a new error occurs, the error overflow bit ECCD.EOV is set, and the corresponding “address buffer overflow” SMU alarm is generated.
For peripheral SRAMs, the second error will cause a buffer overflow, and for CPU SRAMs, up to five errors can be registered before the buffer overflow alarm is triggered.

In addition, traps and bus errors are generated for uncorrectable errors, depending on the bus master and type of access.

**Corner Case**

If in an exceptional corner case

- two errors at different locations are present in the same SRAM
- and accesses are made to both locations within a time window of ~ 10 CPU clock cycles,

then the first access to the location with an error will correctly trigger an SMU alarm, while the second access to the other location with an error will not trigger an SMU alarm. In the worst case, a correctable error may thus mask an uncorrectable or address error.

*Note:* In case the second error would result in an address buffer overflow, the corresponding bit ECCD.EOV is set and the “address buffer overflow” SMU alarm is correctly generated.

Therefore, this problem is not relevant for peripheral SRAMs that only have one ETRR, as the second error will always cause an SMU alarm.

**Recommendations**

- As recommended in Application Hint MTU_TC.H004 (Handling the Error Tracking Registers ETRR), for correctable errors, the application software should only react on the address buffer overflow alarm (e.g. with a reset). Single correctable error events may be ignored (within limits) to increase the fault tolerance of the system without impacting the safety.
- In case an uncorrectable error for a CPU SRAM would neither generate an “address buffer overflow” nor an “uncorrectable” or “address error” SMU alarm, the error handling (typically resulting in a reset) should be performed in the corresponding trap routine.
- In particular for EMEM or FFT SRAMs used in Emulation, ADAS or Extended SRAM devices of the AURIX™ family, a workaround is possible by triggering a correctable error before application startup. This would result in the ECCD.CERR bit of the corresponding MBIST to be set. Any future
correctable alarms will not be forwarded\(^1\) and this issue can be avoided completely.

**MTU_TC.H006  Alarm Propagation to SMU via Error Flags in MCx_ECCD**

Upon any correctable, un-correctable or address error alarm in an SRAM, the corresponding error flags (CERR, UERR or AERR bits) in the MCx_ECCD register are set, and the corresponding alarm is forwarded to the SMU.

However, in case these bits are set to 1\(_B\), and a further error of the same type occurs, then the corresponding alarm is no longer forwarded to the SMU.

If in a corner case software writes to Mx_ECCD in the same cycle where an error event would set one of the CERR, UERR or AERR bits from 0\(_B\) to 1\(_B\), the software write has priority and the status flags remain at 0\(_B\). In this case, however, the alarm is correctly propagated to the SMU.

*Note:* This behavior does not endanger the concept recommended in Application Hints MTU_TC.H004 and MTU_TC.H005 (ignore correctable errors, react on first uncorrectable/address error/buffer overflow alarm).

**Recommendation**

Upon any alarm from an SRAM/MBIST, if a further alarm of the same type is required to be sent to the SMU and processed, then the software shall clear the error flag (CERR, UERR, AERR) in the ECCD register.

The flags can be cleared by writing MCx_ECCD.CERR (or UERR or AERR, respectively) with 0\(_B\).

**MTU_TC.H007  Reset Values of Bit ECCS.TRE**

The default reset value of bit MTU_ECCS.TRE (Tracking Enable) is 0\(_B\).

A special reset value of 1\(_B\) is implemented for the MTU_ECC.TRE bit of MCs of all TriCore Memories. In this context, ‘TriCore Memories’ means all available

---

\(^1\) see MTU_TC.H006 (Alarm Propagation to SMU via Error Flags in MCx_ECCD)
DTAG, PTAG, PSPR, DSPR and DSPR2 Memory Controllers of all CPUs implemented in the product.

MTU_TC.H008 Memory Controllers for DSPR

Due to their implementation, the Data Scratch Pad RAMs (DSPR) of CPU1 and CPU2 each have two Memory Controller instances, described in the MTU chapter of the User’s Manual as:

- MC6 (MC_CPU1 TC16P_DSPR), MC20 (MC_CPU1 TC16P_DSPR2) and
- MC0 (MC_CPU2 TC16P_DSPR), MC21 (MC_CPU2 TC16P_DSPR2).

Each Memory Controller covers one half of the SRAM. In order to fully test the DSPR, the test\(^1\) has to be executed once on each Memory Controller (i.e. only one of the two Memory Controllers is enabled at a time).

As both Memory Controllers share the same ECC decoders, any error detected by a test executed on one of the Memory Controllers will be logged in the Error Tracking Registers of both Memory Controllers.

Note that once an error status bit is set, further alarms of the same type are not forwarded to the SMU until the flag is cleared (see MTU_TC.H006 “Alarm Propagation to SMU via Error Flags in MCx_ECCD”).

Recommendation

- Enable only one of the two Memory Controllers at a time.
- Before executing a test on CPUx_DSPR (respectively CPUx_DSPR2), clear the Error Tracking Registers and the error status bits of CPUx_DSPR (respectively CPUx_DSPR2), so that the test reflects the results of only the one memory which is being tested.
- It is also recommended to clear the Error Tracking Registers and the error status bits of both CPUx_DSPR and CPUx_DSPR2 after executing each test.
- Alternatively, before executing a test on CPUx_DSPR (respectively CPUx_DSPR2), disable the error notifications in CPUx_DSPR2

\(^1\) Test in this context means Non-Destructive Inversion Test (NDIT, see also MTU_TC.H003).
(respectively CPUx_DSPR) and reenable them after the test. It is also recommended to clear the Error Tracking Registers and the error status bits of CPUx_DSPR (respectively CPUx_DSPR2) after executing a test on it.

Regarding configuration and use of the two Memory Controllers, see also the latest version of Application Note AP32197 “AURIX™ Memory Tests using the MTU”.

**MTU_TC.H009 Reset Value for Register ECCD**

The reset value of the ECC Detection Register ECCD is documented as 7800\(_H\) in the User’s Manual. This is always the case for the SRAMs listed in Table 49 below (if available in the corresponding product).

<table>
<thead>
<tr>
<th>Memory Controller No.</th>
<th>Associated SRAM</th>
<th>Comments / Memory available in</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>CPU2 TC16P_DTAG</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>CPU2 TC16P_PTAG</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>CPU1 TC16P_DTAG</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>CPU1 TC16P_PTAG</td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>CPU0 PTAG</td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>CPU0 TC16P_DTAG</td>
<td></td>
</tr>
<tr>
<td>30</td>
<td>GTM MCS1</td>
<td></td>
</tr>
<tr>
<td>31</td>
<td>GTM DPLL RAM1A</td>
<td></td>
</tr>
<tr>
<td>32</td>
<td>GTM DPLL RAM1B</td>
<td></td>
</tr>
<tr>
<td>34</td>
<td>PSI5</td>
<td></td>
</tr>
<tr>
<td>38</td>
<td>ERAY0 OBF</td>
<td></td>
</tr>
<tr>
<td>39</td>
<td>ERAY0 IBF_TBF</td>
<td></td>
</tr>
<tr>
<td>80</td>
<td>CIF1</td>
<td>ADAS products only</td>
</tr>
<tr>
<td>81</td>
<td>CIF2</td>
<td>ADAS products only</td>
</tr>
<tr>
<td>83</td>
<td>DMA</td>
<td></td>
</tr>
</tbody>
</table>
For other SRAMs the ECCD reset value may either be \text{7C00}_{16} or \text{7800}_{16}.

Bit ECCD.10 is marked as ‘Reserved’ in the User’s Manual:

- When writing to ECCD, bit ECCD.10 should be written as \text{0B}.
- When reading register ECCD, bit ECCD.10 should not be evaluated.

Memory errors will be reported by the notification bits CERR, UERR, AERR and EOV in register ECCD.

MTU TC.H010 Register MCONTROL - Bit Field Res4

The position of the 3-bit field Res4 within register MCONTROL is incorrectly described as [14:10] in the register description of the User’s Manual.

The correct position of the 3-bit field Res4 is MCONTROL.[14:12], as shown in the register image in the User’s Manual, and in the following Table 50:

Table 50 Register MCONTROL - Position of Bit Field Res4

<table>
<thead>
<tr>
<th>Field</th>
<th>Bits</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Res</td>
<td>15</td>
<td>r</td>
<td>Reserved</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Read returns \text{0B}, should be written with \text{0B}</td>
</tr>
<tr>
<td>Res4</td>
<td>14:12</td>
<td>rw</td>
<td>Reserved</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Read returns \text{0x4}</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Must always be written with \text{0x4}</td>
</tr>
<tr>
<td>Res</td>
<td>11:10</td>
<td>r</td>
<td>Reserved</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Read returns \text{00B}, should be written with \text{00B}</td>
</tr>
</tbody>
</table>

MTU TC.H011 Access Protection for Memory Control Registers

The access protection symbol ‘P’ to indicate Access Enable Register protection is missing in column “Access Mode - Write” in table “Register Overview of each MTU Memory Control register block” of the MTU chapter in the User’s Manual.

The MTU Memory Control register block actually has protection via the Access Enable registers (ACCEN0/1).
MTU_TC.H012  Kernel Reset triggers Reset of MBIST Registers

When a kernel reset is executed (via bit RST in registers KRST0/1) for a module equipped with Memory Controllers (MC) for its internal RAMs, also the corresponding MTU Memory Control (MBIST) registers are reset.

Recommendation

If required, analyze/save the contents of the MBIST registers before executing a kernel reset.

After a kernel reset, reconfigure the MBIST registers.

MTU_TC.H014  Access to SRAM while MTU operations are underway

When MTU operations on the SRAM are underway, the memories cannot be accessed. MTU operations in this context include:

1. Running an MBIST test (e.g. Non-destructive test).
2. Performing an SRAM initialization using the MTU.
3. When an Auto-data-initialization is underway.

During these operations, the SRAM shall not be accessed. If the SRAM is accessed during this time, unexpected behavior may occur (e.g. access timeout).

Cases 1. and 2. are easily identified, i.e. whenever the application has triggered an MBIST test or SRAM initialization.

Case 3. occurs whenever bit field PROCOND.RAMIN is not equal to 0x3. Whenever this is the case in specific MBIST controllers, the SRAM is fully or partially cleared under certain conditions:

• When MTU_MEMTEST.*EN bit is enabled or disabled.
• When MTU_MEMMAP.*MAP bit is set or cleared (applicable only to cache memories).

This means, when the above mentioned bits are set or cleared, it takes some time (~hundreds of clock cycles) for the associated SRAMs to be (fully or partially) initialized. During this time the SRAM is not accessible.

Affected SRAMs are:
• CPUx DMEM (DSPR+DCACHE)
• CPUx PMEM (PSPR + PCACHE)

**Recommendation**

• For all memories, ensure that the SRAM is not accessed when any MTU operation is underway.
• For the specific memories listed above, ensure that the SRAM is not accessed:
  – When setting MTU_MEMTEST.*EN bit: as long as MEMSTAT.*AIU bit is set or as long as the MEMTEST.*EN bit is not yet set.
  – When clearing MTU_MEMTEST.*EN bit: as long as MEMSTAT.*AIU bit is set or as long as the MEMTEST.*EN bit is not yet cleared.
  – When setting or clearing MTU_MEMMAP.*MAP bit for DMEM/PMEM: as long as MEMSTAT.*AIU bit is set.

**MultiCAN_AI.H005  TxD Pulse upon short disable request**

If a CAN disable request is set and then canceled in a very short time (one bit time or less) then a dominant transmit pulse may be generated by MultiCAN module, even if the CAN bus is in the idle state.

Example for setup of the CAN disable request:

\[
\text{CAN_CLC.DISR} = 1 \text{ and then CAN_CLC.DISR} = 0
\]

**Workaround**

Set all INIT bits to 1 before requesting module disable.

**MultiCAN_AI.H006  Time stamp influenced by resynchronization**

The time stamp measurement feature is not based on an absolute time measurement, but on actual CAN bit times which are subject to the CAN resynchronization during CAN bus operation. The time stamp value merely indicates the number of elapsed actual bit times. Those actual bit times can be shorter or longer than nominal bit time length due to the CAN resynchronization events.
Workaround
None.

**MultiCAN_Al.H007  Alert Interrupt Behavior in case of Bus-Off**

The MultiCAN module shows the following behavior in case of a bus-off status:

When the threshold for error warning (EWRN) is reached (default value of Error Warning Level EWRN = 0x60), then the EWRN interrupt is issued. The bus-off (BOFF) status is reached if TEC > 255 according to CAN specification, changing the MultiCAN module with REC and TEC to the same value 0x1, setting the INIT bit to 1B, and issuing the BOFF interrupt. The bus-off recovery phase starts automatically. Every time an idle time is seen, REC is incremented. If REC = 0x60, a combined status EWRN+BOFF is reached. The corresponding interrupt can also be seen as a pre-warning interrupt, that the bus-off recovery phase will be finished soon. When the bus-off recovery phase has finished (128 times idle time have been seen on the bus), EWRN and BOFF are cleared, the ALERT interrupt bit is set and the INIT bit is still set.

**MultiCAN_TC.H003  Message may be discarded before transmission in STT mode**

If MOFCRn.STT=1 (Single Transmit Trial enabled), bit TXRQ is cleared (TXRQ=0) as soon as the message object has been selected for transmission and, in case of error, no retransmission takes places.

Therefore, if the error occurs between the selection for transmission and the real start of frame transmission, the message is actually never sent.
Workaround

In case the transmission shall be guaranteed, it is not suitable to use the STT mode. In this case, \texttt{MOFCRn.STT} shall be 0.

\textbf{MultiCAN TC.H004 Double remote request}

Assume the following scenario: A first remote frame (dedicated to a message object) has been received. It performs a transmit setup (\texttt{TXRQ} is set) with clearing \texttt{NEWDAT}. MultiCAN starts to send the receiver message object (data frame), but loses arbitration against a second remote request received by the same message object as the first one (\texttt{NEWDAT} will be set).

When the appropriate message object (data frame) triggered by the first remote frame wins the arbitration, it will be sent out and \texttt{NEWDAT} is not reset. This leads to an additional data frame, that will be sent by this message object (clearing \texttt{NEWDAT}).

There will, however, not be more data frames than there are corresponding remote requests.

\textbf{Figure 11 Loss of Arbitration}
MultiCAN_TC.H007 Oscillating CAN Bus may Disable the CAN Interface

If the connected CAN network is in an unspecified oscillating state for more than 512 cycles this can result in disabling the CAN interface of the device. Enabling the CAN interface again requires then a Power-on Reset.

Recommendation
Please refer to application note AP32264 “DXCPL DAP over CAN Physical Layer” for further information and how this situation can be prevented.

OCDS_TC.H011 Application Reset during CIF Transactions while OCDS is enabled

Note: This problem may only occur on TC26x and TC29x ADAS or Emulation devices, and on TC29x Extended SRAM devices.

If OCDS is enabled, then an Application Reset during CIF to EMEM transactions could potentially result in the CIF being reset without a corresponding reset of the bus infrastructure between CIF and EMEM. This could result in disruption to further use of the CIF and IOC32 on the extension part of the device, requiring a device reset (PORST).
Note: This problem will not occur if OCDS is disabled.

Recommendation
Do not perform an Application Reset if OCDS is enabled and CIF to EMEM transactions may be ongoing.

OCDS_TC.H012 Minimum Hold Time for Inputs OCDS_TGIx

Inputs OCDS_TGIx (x=0..7, depending on device/package type) may be used to trigger the On-Chip Debug System (OCDS) e.g. for break or interrupt from an external source.
To ensure the external trigger is sampled correctly and not missed, the trigger should be asserted for a minimum of two SPB clock cycles.
PADS TC.H001  Hysteresis Inactive Function

The following sentence in the first section of chapter “Pad Driver Mode Register” in the User’s Manual is partially incorrect:

“For port lines configured as input (PCx), the PDx fields determines if hysteresis is active or inactive for the input function (hysteresis function is only available for MP, MP+, MPR and LP pads).”

Correction

The correct description is as follows:

For port lines configured as input (PCx), the PDx fields determine if hysteresis is active or inactive for the input function (hysteresis inactive function is only available for MP, MP+, MPR and most LP pads). A2, F, and S pads do not support a hysteresis inactive function.

PADS TC.H002  Write Access to Register PMSWCR0 when HWCFG[6] = 0

When option HWCFG[6] = 0 is selected (i.e. default pad behavior is tristate), bit PMSWCR0.TRISTREQ has to be reconfigured to TRISTREQ = 1_B via PMSWCR0.TRISTEN = 1_B with the first write operation to register PMSWCR0 following a power-on.

Otherwise, with TRISTEN = 0_B and TRISTREQ = X_B on the first write to PMSWCR0 after power-on, the pad default behavior (unexpectedly) changes to ‘pull-up’ with the next warm PORST, system, or application reset assertion/deassertion.

For any subsequent write to PMSWCR0 after power-on, the protection for bit TRISTREQ works as specified (i.e. pad behavior unchanged after write with TRISTEN = 0_B).

Note: No special considerations are required when HWCFG[6] = 1, or when PMSWCR0 is not modified.
Recommendation

When option HWCFG[6] = 0 is selected (i.e. default pad behavior is tristate), ensure to write PMSWCR0.[22:21] = 11_B (i.e. TRISTREQ = 1_B, TRISTEN = 1_B) with the first write operation to register PMSWCR0 following a power-on.

PADS_TC.H003 Terminology Mapping: Emulation/Production Device

The definition of the terms “Emulation Device (ED)” and “Production Device (PD)” in chapter “Package and Pinning Definitions” of the Data Sheet and in table “Tool Relevant Device Pins of AURIX Family” in the User’s Manual, in particular for pins VDD/VDDSB and NC/VDDPSB, applies to the individual TC29x device variants¹) as shown in the following tables:

### Table 51  TC299 Supply Pins - Mapping to Device Variants

<table>
<thead>
<tr>
<th>Pin</th>
<th>Symbol</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>N12,</td>
<td>VDD / VDDSB</td>
<td>• TC299 Emulation Device or TC299TA/TB/TX/TY: Standby Power Supply (1.3 V) for Emulation SRAM or Extended Memory (EMEM).</td>
</tr>
<tr>
<td>M13</td>
<td></td>
<td>• TC299T/TP/TC: VDD (1.3 V)</td>
</tr>
<tr>
<td>R19</td>
<td>NC / VDDPSB</td>
<td>• TC299 Emulation Device or TC299TA/TB/TX/TY: Power Supply (3.3V) for DAP¹)/JTAG pad group.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• TC299T/TP/TC: This pin is not connected on package level. It can be connected on PCB level to VDDP3 or Ground or can be left unsupplied.</td>
</tr>
</tbody>
</table>

¹) 2-/3-pin DAP

---

¹) For availability see the corresponding “AURIX™ TC29x Variants / Data Sheet Addendum”.
### Table 52  TC298 Supply Pins - Mapping to Device Variants

<table>
<thead>
<tr>
<th>Pin</th>
<th>Symbol</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>R1, R4</td>
<td>VDD / VDDSB</td>
<td>• TC298 Emulation Device: Standby Power Supply (1.3 V) for Emulation SRAM.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• TC298TP: Not connected on package level.</td>
</tr>
<tr>
<td>P17</td>
<td>NC / VDDPSB</td>
<td>• TC298 Emulation Device: Power Supply (3.3V) for DAP(^1)/JTAG pad group.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• TC298TP: This pin is not connected on package level. It can be connected on PCB level to VDDP3 or Ground or can be left unsupplied.</td>
</tr>
</tbody>
</table>

\(^1\) 2-/3-pin DAP

### Table 53  TC297 Supply Pins - Mapping to Device Variants

<table>
<thead>
<tr>
<th>Pin</th>
<th>Symbol</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>G8, H7</td>
<td>VDD / VDDSB</td>
<td>• TC297 Emulation Device or TC297TA/TB/TX/TY: Standby Power Supply (1.3 V) for Emulation SRAM or Extended Memory (EMEM).</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• TC297T/TP/TC: VDD (1.3 V)</td>
</tr>
<tr>
<td>K14</td>
<td>NC / VDDPSB</td>
<td>• TC297 Emulation Device or TC297TA/TB/TX/TY: Power Supply (3.3V) for DAP(^1)/JTAG pad group.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• TC297T/TP/TC: This pin is not connected on package level. It can be connected on PCB level to VDDP or Ground or can be left unsupplied.</td>
</tr>
</tbody>
</table>

\(^1\) 2-/3-pin DAP
**PLL_ERAY_TC.H002** Correction in Figure “PLL_ERAY Block Diagram”

The signal originating from block “K2-Divider” in figure “PLL_ERAY Block Diagram” in chapter “ERAY Phase-Locked Loop” of the User’s Manual is incorrectly labeled as PLLERAYSTAT.K1RDY.

**Correction**

The correct name of the signal originating from block “K2-Divider” is PLLERAYSTAT.K2RDY.

**PMC_TC.H001** Check for permanent Overvoltage during Power-up

After an initial power-on with a permanent overvoltage condition on either $V_{_{\text{EXT}}}$, $V_{_{\text{DDP3}}}$ or $V_{_{\text{DD}}}$ supply rails, no overvoltage alarm may be generated by the SMU after configuration of the alarms, as the threshold transition condition has already happened.

However, in case an overvoltage condition was present, it will be indicated by flags OV13, OV33, and OVSWD, respectively, in register EVRSTAT.

**Recommendation**

Check the OV13, OV33, and OVSWD flags in register EVRSTAT by software at start-up to identify an overvoltage condition.

**PMC_TC.H002** Description of Register PMSWSTAT

In the description of the Standby and Wake-up Status Flag Register (PMSWSTAT) in the User’s Manual, bit PMSWSTAT.16 is erroneously shown as 0B.

As the reset value of PMSWSTAT.16 = 1B, the correct description of bit 16 in PMSWSTAT is as shown in Table 54 below:
PORTS_TC.H006 Using P33.8 while SMU is disabled

Per default, the SMU is enabled (SMU_CLC = 0x0) and collects the alarms from the safety mechanisms defined by the safety concept. The SMU may optionally use P33.8 to output the Fault Signaling Protocol (FSP), selectable via register SMU_PCTL. To satisfy safety requirements, it is ensured that the pad configuration of this pin is not affected by an application or system reset after the first 0-to-1 transition of bit SMU_PCTL.PCS.

If the SMU is enabled, but is not using P33.8 for the FSP function, this pin may be used as general purpose input/output (GPIO) or alternate function input/output, controlled via the corresponding P33 registers.

However, if the SMU is disabled by software (SMU_CLC.DISR = 1B, i.e. not clocked), configuration of P33.8 (pull devices, driver settings, selection of alternate function, etc.) requires special considerations as described in the following, otherwise the configuration change may not become effective.

Recommendations

- If P33.8 shall be used as GPIO or alternate function input/output, do not disable the SMU, i.e. keep SMU_CLC = 0x0 (default after reset). In this case, the configuration of P33.8 may be changed by software at any time.
- Alternatively, configure P33.8 before the SMU is disabled by software (SMU_CLC.DISR = 1B). After the SMU is disabled, the configuration of P33.8 can no longer be modified by software.
- Alternatively, if the SMU is disabled by software (SMU_CLC.DISR = 1B, i.e. not clocked), clear bit position 8 at address 0xF003 D364 in the P33 address space once after any reset (Application, System Reset, PORST) before

---

Table 54 Register PMSWSTAT - Bits 19:16

<table>
<thead>
<tr>
<th>Field</th>
<th>Bits</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Res</td>
<td>16</td>
<td>r</td>
<td>Reserved</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Read as 1</td>
</tr>
<tr>
<td>0</td>
<td>19:17</td>
<td>r</td>
<td>Reserved</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Read as 0</td>
</tr>
</tbody>
</table>
configuring P33.8. Controlling P33.8 as FSP by SMU is possible only once after a reset.

*Note: Write access to address 0xF003 D364 is Safety ENDINIT protected.*

**PORTS_TC.H008  Emergency Stop for LVDS TX Pads in LVDS Mode**

The Emergency Stop function allows to force GPIOs (General Purpose Inputs/Outputs) into a defined state (input with pull-up or High-Z), either via an external signal (EMGSTOPA or EMGSTOPB) or the SMU Port Emergency Stop feature (PES).

However, on pins with LVDSM/LVDSH TX pads, the Emergency Stop function affects only the CMOS driver, not the LVDS driver.

For TC29x, these are P22.[3:0], P21.[5:4], P13.[7:0].

Thus, for LVDSM/H pads, only when CMOS mode is selected the output is switched off. When LVDS mode is selected the output is not switched off by the Emergency Stop function.

**Recommendation**

In case these LVDS TX pads are used in LVDS mode, and an Emergency Stop event occurs, switch them to the desired state via software.

**PORTS_TC.H009  Port Input/Output Control Register listed for P01.12**

The Port Input/Output Control Register (P01_IOCR8) listed for P01.12 in Table “Port 01 Functions” of the TC29x User's Manual is incorrect.

**Correction**

P01.12 is controlled by P01_IOCR12.

**QSPI_TC.H005  Stopping Transmission in Continuous Mode**

The QSPI module supports the following mechanisms to (temporarily) suspend its operation:
• Pause by setting bit GLOBALCON.EN = 0_B via software
• Disable by setting bit CLC.DISR = 1_B via software
• Sleep Mode (enabled with CLC.EDIS = 0) requested by hardware
• Suspend Mode requested by hardware (debugger)

These modes and their handling is described in detail in section “Operation Modes” of the QSPI chapter in the User’s Manual.

In **Continuous Mode**, the following specific behavior of QSPI module has to be considered:

• In case the QSPI module is put into **Pause** state by setting bit GLOBALCON.EN = 0_B via software, it continues transmission until the end of the TRAIL phase of the frame with BACON.LAST = 1_B.
• In case the QSPI module is put into **Disable**, **Sleep**, or **Suspend** mode, the frame is stopped after the next trailing delay (character n). In case BACON.LAST was not =1_B at that time, transmission continues with character n+2 when operation from Disable/Sleep/Suspend state is resumed, i.e. data loss (character n+1) will occur.

**Recommendation**

Ensure that software does not put the QSPI module into Pause or Disable state (via GLOBALCON.EN or CLC.DISR) while a transmission in Continuous Mode is ongoing.

If Sleep Mode is used in the system, disable acceptance of sleep requests (set CLC.EDIS = 1_B) before starting data transmission in Continuous Mode.

During debugging, ensure that the QSPI is not suspended while it is transmitting in Continuous Mode.

**QSPI_TC.H006  Corrections to Figures “QSPI - Frequency Domains” and “Phase Duration Control, Overview”**

In the current version of the User's Manual,

• Figure “QSPI - Frequency Domains” erroneously uses the term “f_{PER}” instead of “f_{BAUD2}”, and
• Figure “Phase Duration Control, Overview” erroneously uses the term “T_{PER}” instead of T_{BAUD2}.”
Correction

- \( f_{SCLK} = \frac{1}{f_{BAUD2}} \) in Figure “QSPI - Frequency Domains”, and
- \( T_{BAUD2} = \frac{1}{f_{BAUD2}} \) in Figure “Phase Duration Control, Overview”.

**QSPI_TC.H007 RXFIFO Overflow Bit Behavior in Slave Mode**

In slave mode, if no data word has been written to TXFIFO during initialization before the master starts sending data, the error flag corresponding to an RXFIFO overflow (bit STATUS.5) is set to 1B.

**Recommendation**

To avoid this RXFIFO overflow event, write (at least) one word to TXFIFO during initialization and after each reset in slave mode. For following transmissions, no data need to be written to TXFIFO to avoid this effect.

**RESET_TC.H002 Unexpected SMU Reset Indication in SCU_RSTSTAT**

Under certain conditions the Reset Status Register SCU_RSTSTAT can show an SMU reset indication in addition to the real reset trigger (e.g. a SW reset). The explanation of this behavior refers to section “Reset Generation” and following pages in chapter “RCU” of the User’s Manual.

Figure “Reset Overview” shows that all warm resets are executed in a defined sequence. This sequence ensures that first the active CPUs are ramped down, then at 80µs the Flash receives an idle request and at 180µs the reset is executed.

The idle request to the Flash makes it immediately busy, all read requests after this point fail with a bus error. All non-CPU masters (HSM, Ethernet, HSSL, DMA and DAM) however continue operation from 80µs to 180µs. When one of these masters reads the busy Flash, a bus error is signaled to the SMU as alarm ALM3[30] (SRI) and/or ALM3[31] (SPB).

If the SMU is configured to react on this by a reset request, this will be noted in the SCU_RSTSTAT register in addition to the original warm reset.

This applies mainly to the master HSM which fetches its code from PFlash.
Recommendations

- Generally a different alarm handling can be configured in the SMU for the mentioned alarms, e.g. trigger an NMI trap but not a reset.
- When the application detects after reset that SCU_RSTSTAT has an additional SMU reset indication it might ignore it and proceed based on the other reset indication.
- In case of SW resets the application can prepare the system just before activating the reset:
  - The non-CPU masters can be disabled or in case of HSM it can be informed about the imminent SW reset and continue execution from RAM.
  - The mentioned alarms can be disabled or the alarm reaction can be changed to trigger an NMI trap.
  - The SMU module reset can be used to reconfigure the SMU into its initial state in which only watchdog timeout alarms are handled.

**RESET_TC.H003 Usage of the Prolongation Feature for ESR0 as Reset Indicator Output**

The ESR0 pin can be used as reset indicator output and in such a case its active low state can be prolonged upon user-configurable selection as described in section “ESRx as Reset Output” of chapter “Reset Control Unit (RCU) in the User’s Manual.

According to this description, an ESR0CNT value of 0 defines “as soon as possible after start of Boot Code execution”, where “as soon as possible” means:

- about 500 µs after cold power-on,
- not less than 20 µs after other types of reset.

**Warning**

In case of ESR0CNT = 2, the ESR0 pin will never be released by the device and the user code will never start.
Note: On the other hand - as explained before - configuring an ESR0CNT value of 1 or 2 would anyhow not be effective as a prolongation time below 20 µs is conceptually unachievable.

Recommendation

Do not configure ESR0CNT = 2.
If prolongation of about 20 µs or below is needed, configure ESR0CNT = 3 or 0 instead.

**RESET_TC.H004 Effect of Power-on and System Reset on DSPR**

The following part of footnote 3) on Table “Effect of Reset on Device Functions” in the RCU chapter “Module Reset Behavior” regarding the effect of startup firmware on Data Scratchpad RAM (DSPR): “DSPR is partially used as a scratchpad by the startup firmware. Previous data stored in the upper 32kB will be overwritten on start-up” is incorrect.

The correct effect is described in the Boot ROM chapter “RAM overwrite during start-up“:
Start-up procedure upon power-on and system reset can overwrite up to 8 Kbyte at the beginning of CPU0 DSPR.

**SCU_TC.H009 LBIST Influence on Pad Behavior**

The behavior of the GPIO and ESR0/1 pads during LBIST execution is as follows:

- ESR0 is switched to input direction during LBIST with weak pull-up and pull-down driver disabled (i.e. pad is tri-stated).
- ESR1 is switched to input direction during LBIST with weak pull-down driver enabled.
- Other GPIO pins are switched to input direction with weak pull-up devices either stable active or inactive (depending on LBIST user configuration).
SCU_TC.H013 Correction to Register References in Chapter “Watchdog Timers”

Some references to register names in chapter “Watchdog Timers” of the User’s Manual are incorrect.

The corrected references and their section headers are listed in **bold** below.

**Section Password Access to WDTxCON0**

.. To ensure that a CPU fault could not allow a fault to be ignored an option is provided to prevent watchdog unlocking if the Safety Management Unit (SMU) is not in the RUN state. This option may be enabled by bit WDTxCON1.UR. If the password is valid and the SMU state meets the requirements of the WDTxSR.US bit then WDTxCON0 will be unlocked as soon as the Password Access is completed. ..

**Section Timer Operation**

.. The parameter divider represents the user-programmable source clock division selected by WDTxCON1.IRx, which can be 64, 256 or 16384.

**Section Watchdog Timer Registers**

- **WDTSCON1** - Safety WDT Control Register 1:
  - References to WDTxCON0 and WDTxSR should be consequently to WDTSCON0 and WDTSSR in the context of WDTSCON1.
- **WDTCPUxCON1** - CPUx WDT Control Register 1:
  - References to WDTSCON0 and WDTSSR should be consequently to WDTCPUxCON0 and WDTCPUxSR in the context of WDTCPUxCON1.

**SCU_TC.H014 Reset Value of Bit Field IOCR.PC1 - Control for Pin ESR1**

The reset value of register SCU_IOCR is documented as 0000 20E0H in chapter “Reset Control Units” of the User’s Manual, i.e. the reset value of bit field PC1 = 2H.

This is not always correct under all circumstances:
The actual SCU_IOCR reset value should be considered as 0000 X0E0H with the explanations given in the following Documentation Update.

**Documentation Update**

The reset value of bit field SCU_IOCR.PC1 is influenced by pin HWCFG6 and bit PMSWCR0.TRISTREQ:

- When a cold reset is activated and HWCFG6=1 then PC1 is reset to 2H and pin ESR1 will have input pull-up mode.
- If HWCFG6=0 then PC1 is reset to 0H and ESR1 will have tri-state mode.

PC1 and the ESR1 reset state can also be configured by software with the PMSWCR0.TRISTREQ bit. PMSWCR0.TRIST REQ is not affected by warm reset or wake-up from standby so the IOCR.PC1 reset value is configured as per the state of the TRISTREQ bit prior to the warm reset.

**SENT_TC.H002 SENT Nibble Tolerance**

The length of a nibble in the SENT protocol determines the value of this nibble. For each value this length can vary, but it has to be at least inside a given range. This range is given by the SENT standard.

- The lower border is nominal_value-(tick_length/2) µs+100 ns
- The upper border is nominal_value+(tick_length/2) µs-100 ns.

![SENT Standard Tolerance Diagram](image)

**Figure 12 SENT Standard Tolerance**

In this design step the range is delimited as follows:
The lower border is nominal_value - (tick_length/2) µs + 100 ns
The upper border is nominal_value + (tick_length/2) µs - 145 ns.

Figure 13  SENT Tolerance of this Design Step

Recommendation
To compensate this 45 ns difference so that there is no deviation from the SENT Standard, use the classified port pins as described in AppNote AP32286 “Parameter V_{ILSD} for LP and MPx Pads relevant for SENT”.

SENT_TC.H003  First Write Access to Registers FDR and TPD after ENDINIT Status Change

Due to an extra registering stage of the ENDINIT signal from the SCU inside the SENT kernel, the behavior of the first write access to SENT registers FDR and TPD protected by the Endinit write protection scheme after an ENDINIT status change is as follows:

- After unlocking protection (ENDINIT change from 1 to 0), if the first access to the SENT module is a write to FDR or TPD, it will still view ENDINIT as locked (value 1). The contents of FDR or TPD is not changed, but no BCU alarm will be generated, as the ENDINIT does not indicate a protected status in case of the access.
- By setting protection again (ENDINIT change from 0 to 1), if the first access to the SENT module is a write to FDR or TPD, it will still be effective, i.e., the value will be written. Nevertheless a SMU alarm through BCU will be generated as the protection status is ENDINIT.
Note: After the first read of any SENT register, or first write to any SENT register, the ENDINIT change will be correctly considered for all following accesses. The CLC, KRST0/1 and KRSTCLR registers (that also have Endinit protection) are not affected at all. An initial value of 0 for ENDINIT is seen by SENT after reset before the first access.

Recommendation
After a change of the ENDINIT protection status, first perform a read of any SENT register or a write to a non-Endinit-protected SENT register. The second access is then always equipped with correct information of ENDINIT.

SENT_TC.H004  Short Serial Message - Figure Correction

In Figure “Short Serial Message, Serial Data Encoding over 16 messages” of the SENT chapter, the arrows originating from bits 2 and 3 of the Status & Comm Nibble are routed incorrectly and must be swapped.

Correction
Figure 14 shows a corrected version of this figure.

Figure 14  Short Serial Message, Serial Data Encoding over 16
**SENT_TC.H005 Interface Connections of the SENT Module - Documentation Correction**

The following corrections apply to chapter “Interface Connections of the SENT Module” in the SENT chapter of the User’s Manual:

**Figure “SENT Module Implementation and Interconnections”**
The range of index \( n \) for connected trigger inputs \( TRIG_n \) in TC29x is \( n = 0..10 \).

**Interrupt and DMA Controller Service Requests**
Table “Service Request Lines of SENT” in the TC29x User’s Manual specifies 10 request lines (SR0..3 and TRIGO0..5) as connected to the Interrupt Router.

In table “Registers Overview - Service Request Control Registers” in the Interrupt Router chapter of the TC29x User’s Manual, registers SRC_SENT10..14 should be regarded as “Reserved”.

**Trigger Inputs**
In TC29x, the SENT module has 15 SENT channels and 11 trigger inputs \( TRIG[10:0] \) selectable in bit field IOCRx.ETS. Values \( \geq 1010_B \) are reserved and should not be used for bit field ETS in register IOCRx.

The trigger inputs (TRIG[10:0]) of the SENT module are connected to the GTM. The rows shown in the table below replace the rows in the corresponding table of the User’s Manual:

<table>
<thead>
<tr>
<th>Table 55 Trigger Input Lines of SENT - Correction for TC29x</th>
</tr>
</thead>
<tbody>
<tr>
<td>Request Line</td>
</tr>
<tr>
<td>---------------</td>
</tr>
<tr>
<td>TRIG9</td>
</tr>
<tr>
<td>TRIG10</td>
</tr>
<tr>
<td>TRIG11..TRIG15</td>
</tr>
</tbody>
</table>
SMU_TC.H001 Write all bit fields of SMU_PCTL with one write access

When configuring the FSP pin (e.g. P33.8), all bit fields (HWDIR, HWEN and PCS) of register SMU_PCTL must be written with the same write access. Otherwise, when first writing a 1B to HWEN before writing a 1B to PCS, the pad configuration will be modified to push/pull configuration before it is latched into field PCFG.

*Note*: When PCS = 1B, the bit fields PCFG and PCS are protected against any changes until the next power on reset. HWEN and HWDIR may still be modified by SW, unless locked via register SMU_KEYS.

SMU_TC.H003 Alarm Table Corrections

Some alarm tables were unintentionally changed between User's Manual (UM) version V1.2 and V1.3. In the following, the issues are described and the correct table parts are included.

- Table 9-3 HwAlarmOut[7:4]: CPU1 DCACHE/DSPR SRAM
  The PreAlarms 9, 11, 13 and 15 were accidentally removed in UM V1.3. The following Table 56 copied from UM V1.2 is correct:

<table>
<thead>
<tr>
<th>Function</th>
<th>HwAlarmOut[4] = PreAlarm[8=CPU1.DMI.DSPR.(ECC single bit correction)] or PreAlarm[9=CPU1.DMI.DSPR2.(ECC single bit correction)]</th>
</tr>
</thead>
</table>
Table 9-4 HwAlarmOut[15:8]: CPU2 SRAMs

The PreAlarms 17, 19, 21 and 23 were accidentally removed in UM V1.3. The following Table 57 copied from UM V1.2 is correct:

Table 57 Table 9-4 HwAlarmOut[15:8]: CPU2 SRAMs

<table>
<thead>
<tr>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>HwAlarmOut[8]=</td>
</tr>
<tr>
<td>PreAlarm[16=CPU2.DMI.DSPR.(ECC single bit correction)] or</td>
</tr>
<tr>
<td>PreAlarm[17=CPU2.DMI.DSPR2.(ECC single bit correction)]</td>
</tr>
<tr>
<td>HwAlarmOut[9]=</td>
</tr>
<tr>
<td>PreAlarm[18=CPU2.DMI.DSPR.(ECC uncorrectable error)] or</td>
</tr>
<tr>
<td>PreAlarm[19=CPU2.DMI.DSPR2.(ECC uncorrectable error)]</td>
</tr>
<tr>
<td>HwAlarmOut[10]=</td>
</tr>
<tr>
<td>PreAlarm[20=CPU2.DMI.DSPR.(Address error)] or</td>
</tr>
<tr>
<td>PreAlarm[21=CPU2.DMI.DSPR2.(Address error)]</td>
</tr>
<tr>
<td>HwAlarmOut[11]=</td>
</tr>
<tr>
<td>PreAlarm[22=CPU2.DMI.DSPR.(Address buffer overflow)] or</td>
</tr>
<tr>
<td>PreAlarm[23=CPU2.DMI.DSPR2.(Address buffer overflow)]</td>
</tr>
</tbody>
</table>

Table 9-6 HwAlarmOut[23:20]: ERAY (FLEXRAY) SRAMs

The PreAlarms of the second ERAY module RAMs were accidentally removed in UM V1.3; these are: 57, 59, 61, 63, 65, 67, 69, 71, 73, 55, 77, 79.
The following Table 58 copied from UM V1.2 is correct:

<table>
<thead>
<tr>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>HwAlarmOut[20]=</td>
</tr>
<tr>
<td>PreAlarm[56=ERAY.SRAM.OBF0.(ECC single bit correction)] or</td>
</tr>
<tr>
<td>PreAlarm[57=ERAY.SRAM.OBF1.(ECC single bit correction)] or</td>
</tr>
<tr>
<td>PreAlarm[58=ERAY.SRAM.TBF_IBF0.(ECC single bit correction)] or</td>
</tr>
<tr>
<td>PreAlarm[59=ERAY.SRAM.TBF_IBF1.(ECC single bit correction)] or</td>
</tr>
<tr>
<td>PreAlarm[60=ERAY.SRAM.MBF0.(ECC single bit correction)] or</td>
</tr>
<tr>
<td>PreAlarm[61=ERAY.SRAM.MBF1.(ECC single bit correction)]</td>
</tr>
</tbody>
</table>

| HwAlarmOut[21]= |
| PreAlarm[62=ERAY.SRAM.OBF0.(ECC uncorrectable error)] or |
| PreAlarm[63=ERAY.SRAM.OBF1.(ECC uncorrectable error)] or |
| PreAlarm[64=ERAY.SRAM.TBF_IBF0.(ECC uncorrectable error)] or |
| PreAlarm[65=ERAY.SRAM.TBF_IBF1.(ECC uncorrectable error)] or |
| PreAlarm[66=ERAY.SRAM.MBF0.(ECC uncorrectable error)] or |
| PreAlarm[67=ERAY.SRAM.MBF1.(ECC uncorrectable error)] |

| HwAlarmOut[22]= |
| PreAlarm[68=ERAY.SRAM.OBF0.(Address error)] or |
| PreAlarm[69=ERAY.SRAM.OBF1.(Address error)] or |
| PreAlarm[70=ERAY.SRAM.TBF_IBF0.(Address error)] or |
| PreAlarm[71=ERAY.SRAM.TBF_IBF1.(Address error)] or |
| PreAlarm[72=ERAY.SRAM.MBF0.(Address error)] or |
| PreAlarm[73=ERAY.SRAM.MBF1.(Address error)] |

| HwAlarmOut[23]= |
| Pre Alarm[74=ERAY.SRAM.OBF0.(Address buffer overflow)] or |
| PreAlarm[75=ERAY.SRAM.OBF1.(Address buffer overflow)] or |
| PreAlarm[76=ERAY.SRAM.TBF_IBF0.(Address buffer overflow)] or |
| PreAlarm[77=ERAY.SRAM.TBF_IBF1.(Address buffer overflow)] or |
| PreAlarm[78=ERAY.SRAM.MBF0.(Address buffer overflow)] or |
| PreAlarm[79=ERAY.SRAM.MBF1.(Address buffer overflow)] |

- Table 9-7 HwAlarmOut[27:24]: CAN SRAM
The PreAlarms of the second CAN module RAMs were accidentally removed in UM V1.3; these are: 81, 83, 85, 87.

The following Table 59 copied from UM V1.2 is correct:

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HwAlarmOut[24]=</td>
<td>PreAlarm[80=CAN.SRAM.MCAN0.(ECC single bit correction)] or PreAlarm[81=CAN.SRAM.MCAN1.(ECC single bit correction)]</td>
</tr>
<tr>
<td>HwAlarmOut[25]=</td>
<td>PreAlarm[82=CAN.SRAM.MCAN0.(ECC uncorrectable error)] or PreAlarm[83=CAN.SRAM.MCAN1.(ECC uncorrectable error)]</td>
</tr>
<tr>
<td>HwAlarmOut[26]=</td>
<td>PreAlarm[84=CAN.SRAM.MCAN0.(Address error)] or PreAlarm[85=CAN.SRAM.MCAN1.(Address error)]</td>
</tr>
<tr>
<td>HwAlarmOut[27]=</td>
<td>PreAlarm[86=CAN.SRAM.MCAN0.(Address buffer overflow)] or PreAlarm[87=CAN.SRAM.MCAN1.(Address buffer overflow)]</td>
</tr>
</tbody>
</table>

- **Table 9-8 HwAlarmOut[31:28]: LMU sub-system SRAMs**

The PreAlarms of the MiniMCDS RAMs (TC29x PD only) and the FFT RAMs (TC29x ADAS only) were accidentally removed in UM V1.3; these are: 230, 231, 232, 233, 234, 235, 236, 237, 238, 239, 240, 241.

The following Table 60 copied from UM V1.2 is correct:
Table 9-11 HwAlarmOut[44:41]: Misc. SRAMs

HWAlarmOut[41] is defined twice in UM V1.3: both in the first and in the last row of Table 9-11. The definition in the last row is incorrect and has to be ignored.

The part containing the correct definition of HWAlarmOut[41] is copied from UM V1.2 into the following Table 61:
Table 61  Part of Table 9-11 with correct HwAlarmOut[41] Definition

<table>
<thead>
<tr>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>HwAlarmOut[41]=PreAlarm[142=PMU.FSI.SRAM(ECC single bit correction)] orPreAlarm[143=DMA.SRAM(ECC single bit correction)] orPreAlarm[144=Ethernet.SRAM(ECC single bit correction)] orPreAlarm[145=PSI5.SRAM(ECC single bit correction)] orPreAlarm[146=MCDS.SRAM(ECC single bit correction)] orPreAlarm[147=CIF.SRAM(ECC single bit correction)] orPreAlarm[148=CIF.JPEG1_4.SRAM(ECC single bit correction)] orPreAlarm[149=CIF.JPEG3.SRAM(ECC single bit correction)]</td>
</tr>
</tbody>
</table>

Table 62  Table 9-13 HwAlarmOut[50:46]: PMU Alarms

The PreAlarms of PFlash2 and PFlash3 were accidentally removed in UM V1.3; these are: 180, 181, 188, 189, 196, 197, 204, 205, 212, 213.

The following Table 62 copied from UM V1.2 is correct:

<table>
<thead>
<tr>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>HwAlarmOut[46]= PreAlarm[178=PMU.PFLASH0.(ECC single bit correction notification)] or PreAlarm[179=PMU.PFLASH1.(ECC single bit correction notification)] or PreAlarm[180=PMU.PFLASH2.(ECC single bit correction notification)] or PreAlarm[181=PMU.PFLASH3.(ECC single bit correction notification)]</td>
</tr>
<tr>
<td>HwAlarmOut[47]= PreAlarm[186=PMU.PFLASH0.(ECC double bit correction notification)] or PreAlarm[187=PMU.PFLASH1.(ECC double bit correction notification)] or PreAlarm[188=PMU.PFLASH2.(ECC double bit correction notification)] or PreAlarm[189=PMU.PFLASH3.(ECC double bit correction notification)]</td>
</tr>
</tbody>
</table>
Table 9-13 HwAlarmOut[50:46]: PMU Alarms (cont’d)

<table>
<thead>
<tr>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>HwAlarmOut[48] = PreAlarm[194=PMU.PFLASH0.(ECC non correctable multiple bit error)] or PreAlarm[195=PMU.PFLASH1.(ECC non correctable multiple bit error)] or PreAlarm[196=PMU.PFLASH2.(ECC non correctable multiple bit error)] or PreAlarm[197=PMU.PFLASH3.(ECC non correctable multiple bit error)]</td>
</tr>
<tr>
<td>HwAlarmOut[49] = PreAlarm[202=PMU.PFLASH0.(Addressing error)] or PreAlarm[203=PMU.PFLASH1.(Addressing error)] or PreAlarm[204=PMU.PFLASH2.(Addressing error)] or PreAlarm[205=PMU.PFLASH3.(Addressing error)]</td>
</tr>
<tr>
<td>HwAlarmOut[50] = PreAlarm[210=PMU.PFLASH0.(Address buffer full)] or PreAlarm[211=PMU.PFLASH1.(Address buffer full)] or PreAlarm[212=PMU.PFLASH2.(Address buffer full)] or PreAlarm[213=PMU.PFLASH3.(Address buffer full)]</td>
</tr>
</tbody>
</table>

- **Table 9-17 TC29x Alarm Mapping related to Alarm 0 group**

ALM0[0] incorrectly shows “Lockstep CPU” in UM V1.3. This alarm actually is reserved as there is no lockstep in CPU0.

The entries for ALM0[14:17] are incorrectly marked as reserved in UM V1.3. The part containing the correct table rows is copied from UM V1.2 into the following **Table 63**:

Table 63  
Part of Table 9-17 with correct TC29x Alarm Mapping related to Alarm 0 group

<table>
<thead>
<tr>
<th>Alarm Index</th>
<th>Module</th>
<th>Safety Mechanism &amp; Alarm Indication</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALM0[0]</td>
<td>Reserved</td>
<td>Reserved</td>
</tr>
<tr>
<td>ALM0[14]</td>
<td>CPU0</td>
<td>Safety Mechanism: SRAM ECC Alarm: DCACHE TAG SRAM ECC correction</td>
</tr>
</tbody>
</table>
Table 63 Part of Table 9-17 with correct TC29x Alarm Mapping related to Alarm 0 group (cont’d)

<table>
<thead>
<tr>
<th>Alarm Index</th>
<th>Module</th>
<th>Safety Mechanism &amp; Alarm Indication</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALM0[15]</td>
<td>CPU0</td>
<td>Safety Mechanism: SRAM ECC Alarm: DCACHE TAG SRAM ECC uncorrectable error</td>
</tr>
<tr>
<td>ALM0[16]</td>
<td>CPU0</td>
<td>Safety Mechanism: SRAM Address Monitor Alarm: DCACHE TAG SRAM address error</td>
</tr>
<tr>
<td>ALM0[17]</td>
<td>CPU0</td>
<td>Safety Mechanism: SRAM Monitor Alarm: DCACHE TAG SRAM address buffer overflow</td>
</tr>
</tbody>
</table>

• Table 9-19 TC29x Alarm Mapping related to ALM2 group

Note: The change of ALM2[24] to “Reserved” is correct in UM V1.3. The intended alarm event has been merged with the other DMA SRI alarms.

• Table 9-20 TC29x Alarm Mapping related to ALM3 group

Note: The change of ALM3[28] to “Reserved” is correct in UM V1.3. This alarm shall not be evaluated.

SMU_TC.H004 Alarm Mapping related to ALM3[9] in ALM3 Group

The VADC incorporated in this device uses clocks derived from f_{SPB}. The fault type “clock out of range” for the VADC is therefore covered by ALM3[7] “SPB clock out of range frequency”.

Previous design steps (e.g. TC27x Bx, TC26x Ax, TC29x Ax) incorporated a different VADC module (clocked by f_{ADC}) using ALM3[9] to signal faults of f_{ADC}.

However, clock f_{ADC} and its monitor still exist in the present design and are connected to ALM3[9] listed as “ADC clock out of range frequency” alarm in table “Alarm Mapping related to ALM3 group” of the SMU chapter.
Recommendation

- New software implementations should treat ALM3[9] as “Reserved”, i.e. software should configure the behavior to “No Action”.
- Software ported from previous design steps with a VADC module clocked by \( f_{\text{ADC}} \) may be reused on this device step. However, alarms from ALM3[9] should be ignored.

SMU_TC.H005 Correction to Figure “SMU Register Map”

The start address “@SMU + 0x0E0” for the SMU System Registers shown in the lower part of figure “SMU Register Map” in the SMU chapter of the User’s Manual is incorrect.

The correct start address is “@SMU + 0x7E0”.

Addresses listed in table “Registers Overview” of the SMU chapter are correct.

SMU_TC.H006 Description of Bit EFRST in Register SMU_AGC

In the SMU chapter of the User's Manual, the description of the encoding of bit EFRST (Enable FAULT to RUN State Transition) in register SMU_AGC (Alarm Global Configuration) is missing.

The complete description should be as shown in Table 64:

<table>
<thead>
<tr>
<th>Table 64</th>
<th>Bit EFRST in Register SMU_AGC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Field</td>
<td>Bits</td>
</tr>
<tr>
<td>EFRST</td>
<td>29</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
SMU_TC.H007  SPB Bus Control Unit (SBCU) Alarm Signalling to SMU

ALM3[31] is dedicated to System Peripheral Bus (SPB) alarms. As described in table “Alarm Mapping related to ALM3 group” in the SMU chapter of the User’s Manual, an SPB bus error can result from multiple root causes, including protocol violation, incorrect address, register access protection violation. More details on the SPB related error conditions can be found in the “On-Chip Bus System” chapter:

The SBCU signals an alarm to the SMU whenever it detects

- a SPB transaction that was finished with a Bus Error (Error Acknowledge)
- an un-implemented Address (no slave responds to a transaction request)
- a SPB transaction that was finished by a Time-out.

The alarm signaling to the SMU is independent of the BCU configuration (e.g. BCU interrupt configuration, BCU debug status).

SMU_TC.H010  Clearing individual SMU flags: use only 32-bit writes

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

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

Recommendation (Examples in C Language)

- **Example 1**: To clear status flag SF2 in register AG0, use:
  - `SMU_AG0.U = 0x0000 0004;`

- **Example 2**: To clear status flags EF2 in register RMEF and RMSTS, use:
  - `SMU_RMEF.U = 0xFFFF FFFB;`
  - `SMU_RMSTS.U = 0xFFFF FFFB;`

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

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

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

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

**SRI_TC.H001 Using LDMST and SWAPMSK.W instructions on SRI mapped Peripheral Registers (range 0xF800 0000-0xFFFF FFFF)**

The LDMST and SWAPMSK.W instructions in the AURIX™ microcontrollers are intended to provide atomicity as well as bit-wise operations to a targeted memory location or peripheral register. They are also referred to as Read-Modify-Write (RMW) instructions.

The bit-manipulation functionality is intended to provide software a mechanism to write to individual bits in a register, without affecting other bits. The bits to be written can be selected via a mask in the instruction. Please refer to the TriCore Architecture Manual for further information about these instructions and their formats.

**Restrictions for SRI mapped Peripherals**

The bit-manipulation functionality is supported only on registers accessed via the SPB bus, and is not supported on the SRI mapped peripheral range (i.e. address range 0xF800 0000 to 0xFFFF FFFF, including (if available) EBU, PMU0, SRI Crossbar, LMU, DAM, FFT, CPUx SFRs and CSFRs, MCDS, miniMCDS); see table “On Chip Bus Address Map of Segment 15” in chapter “Memory Map”.

On the SRI mapped peripherals, usage of these instructions always results in all the bits of a register being written, and not just specific individual bits.
Note: The instructions are still executed atomically on the bus – i.e the SRI is locked between the READ and the WRITE transaction.

STM_TC.H001 Effect of kernel reset on interrupt outputs STMIR0/1

The clock ratio \( f_{STM} : f_{SPB} \) is determined by the settings of bit fields STMDIV and SPBDIV in registers CCUCON1 and CCUCON0, respectively.

If \( f_{STM} \leq f_{SPB} \), and a kernel reset of the STM module is performed in the same clock cycle where a compare match of the STM with the CMP0 or CMP1 registers occurs, a transition on the interrupt outputs STMIR0 or STMIR1 may occur. This may e.g. trigger the External Request Unit (ERU), or set the corresponding Service Request flags SRC_STMmSR0.SRR or SRC_STMmSR1.SRR in the Interrupt Router (\( m = 0, 1, 2 \), depending on number of CPUs).

Note: For \( f_{STM} > f_{SPB} \), this effect will not occur.

Recommendation

If \( f_{STM} \leq f_{SPB} \), set bits ICR.CMP0EN = 0B and ICR.CMP1EN = 0B to disable the compare match interrupts before performing the STM kernel reset.

STM_TCH002 Access Protection for STM Control Registers

The access protection symbol ‘P’ to indicate Access Enable Register protection is missing in table “Registers Overview - STM Control Registers” of the STM chapter in the User’s Manual for the STM registers CMP0, CMP1, CMCON, ICR, ISCR.

The STM registers CMP0, CMP1, CMCON, ICR, ISCR actually have protection via the Access Enable registers (ACCEN0/1), as shown in the following Table 65.
### Table 65  Correction to Table Registers Overview - STM Control Registers

<table>
<thead>
<tr>
<th>Short Name</th>
<th>Description</th>
<th>Offset Addr.</th>
<th>Access Mode</th>
<th>Reset</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>Read</td>
<td>Write</td>
</tr>
<tr>
<td>CMP0</td>
<td>Compare Register 0</td>
<td>30&lt;sub&gt;H&lt;/sub&gt;</td>
<td>U, SV</td>
<td>U, SV, P</td>
</tr>
<tr>
<td>CMP1</td>
<td>Compare Register 1</td>
<td>34&lt;sub&gt;H&lt;/sub&gt;</td>
<td>U, SV</td>
<td>U, SV, P</td>
</tr>
<tr>
<td>CMCON</td>
<td>Compare Match Control Register</td>
<td>38&lt;sub&gt;H&lt;/sub&gt;</td>
<td>U, SV</td>
<td>U, SV, P</td>
</tr>
<tr>
<td>ICR</td>
<td>Interrupt Control Register</td>
<td>3C&lt;sub&gt;H&lt;/sub&gt;</td>
<td>U, SV</td>
<td>U, SV, P</td>
</tr>
<tr>
<td>ISCR</td>
<td>Interrupt Set/Clear Register</td>
<td>40&lt;sub&gt;H&lt;/sub&gt;</td>
<td>U, SV</td>
<td>U, SV, P</td>
</tr>
<tr>
<td>Sales Name</td>
<td>SP number</td>
<td>OPN</td>
<td>Package</td>
<td></td>
</tr>
<tr>
<td>-------------------</td>
<td>------------</td>
<td>----------------------</td>
<td>-------------------</td>
<td></td>
</tr>
<tr>
<td>SAK-TC297T-128F300S BB</td>
<td>SP001265596</td>
<td>TC297T128F300SBBKXUMA1</td>
<td>PG-LFBGA-292-6</td>
<td></td>
</tr>
<tr>
<td>SAK-TC297T-96F300S BB</td>
<td>SP001265616</td>
<td>TC297T96F300SBBKXUMA1</td>
<td>PG-LFBGA-292-6</td>
<td></td>
</tr>
<tr>
<td>SAK-TC297TA-128F300S BB</td>
<td>SP001130460</td>
<td>TC297TA128F300SBBKXUMA1</td>
<td>PG-LFBGA-292-6</td>
<td></td>
</tr>
<tr>
<td>SAK-TC297TA-64F300S BB</td>
<td>SP001265584</td>
<td>TC297TA64F300SBBKXUMA1</td>
<td>PG-LFBGA-292-6</td>
<td></td>
</tr>
<tr>
<td>SAK-TC297TC-96F300S BB</td>
<td>SP001265598</td>
<td>TC297TC96F300SBBKXUMA1</td>
<td>PG-LFBGA-292-6</td>
<td></td>
</tr>
<tr>
<td>SAK-TC297TP-128F300S BB</td>
<td>SP001207656</td>
<td>TC297TP128F300SBBKXUMA1</td>
<td>PG-LFBGA-292-6</td>
<td></td>
</tr>
<tr>
<td>SAK-TC297TP-96F300S BB</td>
<td>SP001001102</td>
<td>TC297TP96F300SBBKXUMA1</td>
<td>PG-LFBGA-292-6</td>
<td></td>
</tr>
<tr>
<td>SAK-TC297TX-128F300S BB</td>
<td>SP001265592</td>
<td>TC297TX128F300SBBKXUMA1</td>
<td>PG-LFBGA-292-6</td>
<td></td>
</tr>
<tr>
<td>SAK-TC297TY-128F300S BB</td>
<td>SP001265594</td>
<td>TC297TY128F300SBBKXUMA1</td>
<td>PG-LFBGA-292-6</td>
<td></td>
</tr>
<tr>
<td>SAK-TC299TC-96F300S BB</td>
<td>SP001265604</td>
<td>TC299TC96F300SBBKXUMA1</td>
<td>PG-LFBGA-516-5</td>
<td></td>
</tr>
<tr>
<td>SAK-TC299TP-128F300S BB</td>
<td>SP001207664</td>
<td>TC299TP128F300SBBKXUMA1</td>
<td>PG-LFBGA-516-5</td>
<td></td>
</tr>
<tr>
<td>SAK-TC299TX-128F300S BB</td>
<td>SP001265588</td>
<td>TC299TX128F300SBBKXUMA1</td>
<td>PG-LFBGA-516-5</td>
<td></td>
</tr>
<tr>
<td>SAL-TC290TP-128F300 BB</td>
<td>SP001001126</td>
<td>TC290TP128F300BBLX2SA1</td>
<td>S-BDWBT-1</td>
<td></td>
</tr>
<tr>
<td>SAL-TC290TP-128F300N BC</td>
<td>SP001397610</td>
<td>TC290TP128F300NBCLX2SA1</td>
<td>S-BDWBT-1</td>
<td></td>
</tr>
<tr>
<td>SAL-TC298TP-128F300L BB</td>
<td>SP001207660</td>
<td>TC298TP128F300LBBLXUMA1</td>
<td>PG-BGA-416-26</td>
<td></td>
</tr>
<tr>
<td>SAL-TC299TP-128F300S BB</td>
<td>SP001260190</td>
<td>TC299TP128F300SBBLXUMA1</td>
<td>PG-LFBGA-516-5</td>
<td></td>
</tr>
</tbody>
</table>