Document Number: IMXRT1050CE
Rev. 3, 11/2021

Chip Errata for the i.MX RT1050

This document details the silicon errata known at the time of publication for the i.MX RT1050 crossover processors.

Table 1 provides a revision history for this document.

Table 1. Document Revision History

Rev. Number	Date	Substantive Changes
Rev. 3	11/2021	• Added following errata: – ERR050606 – ERR050607 – ERR050538 – ERR011573 • Removed the following erratum: – ERR006281
Rev. 2.3	12/2020	• Added following errata: – ERR050143 – ERR050577 • Removed the following errata: – ERR007265 (replaced ERR007265 with ERR050143) • Updated the Figure 1, "Revision Level to Part Marking Cross-Reference,"
Rev. 2.2	01/2020	• Added following errata: – ERR050235 • Updated the ERR050101 solution in the Table 2
Rev. 2.1	08/2019	• Updated following errata: – ERR050194

Table 1. Document Revision History (continued)

Rev. Number	Date	Substantive Changes
Rev. 2	06/2019	• Added following errata: – ERR011572 – ERR050101 – ERR050130 – ERR050144 – ERR050194
Rev. 1.1	08/2018	• Added following errata: – ERR006032 – ERR009527 – ERR009595 – ERR011207 – ERR011377
Rev. 1	03/2018	• Added following errata: – ERR006223 – ERR011225 – ERR011262 • Updated following errata: – ERR011164 – ERR011092 – ERR011093 – ERR011091 – ERR011138 – ERR011111 – ERR011165 – ERR011110 – ERR011119 – ERR011120
Rev. 0	10/2017	• Initial version

Figure 1 provides a cross-reference to match the revision code to the revision level marked on the device.

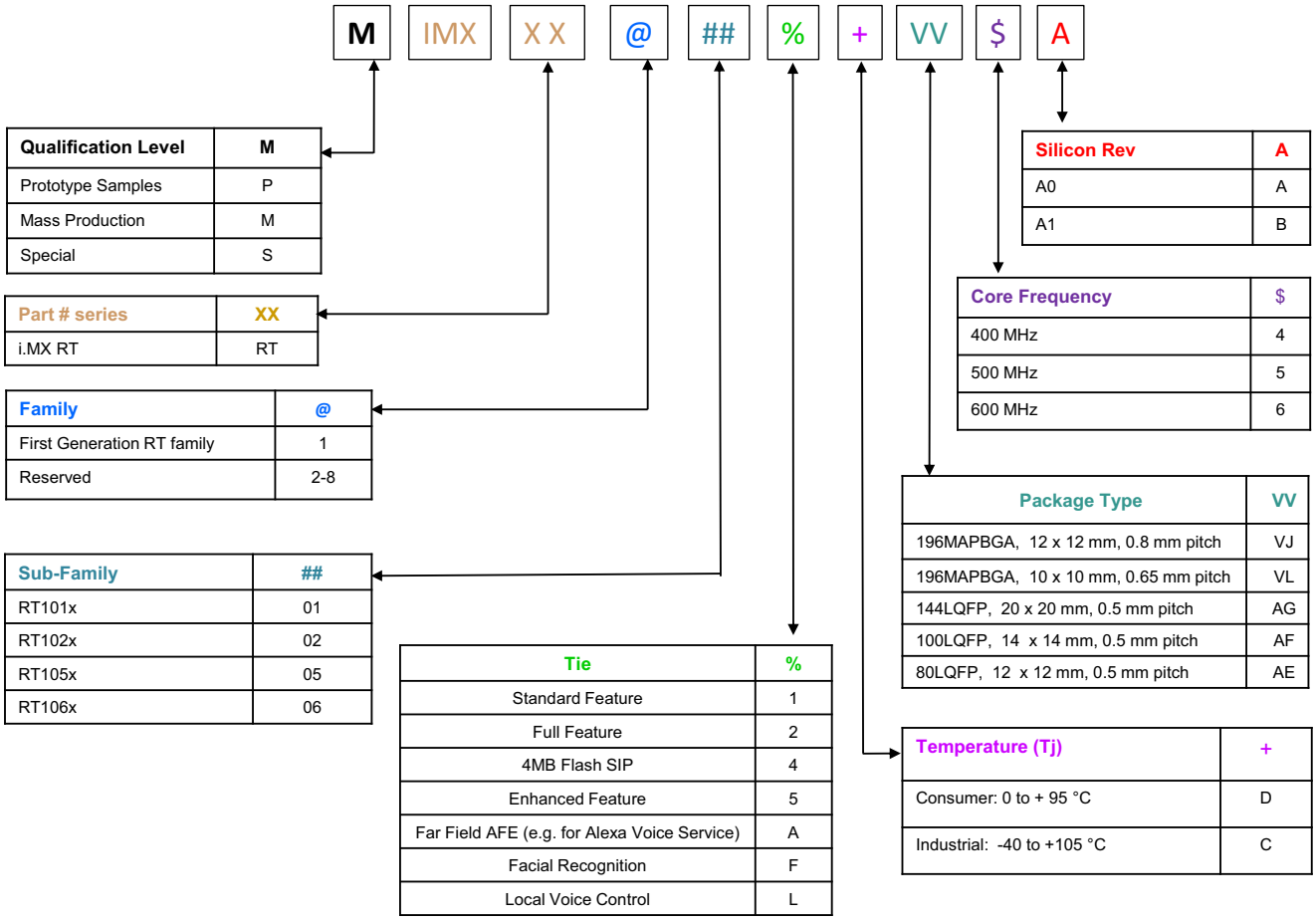

Figure 1. Revision Level to Part Marking Cross-Reference

For details on the Arm® configuration used on this chip (including Arm module revisions), please see the "Platform configuration" section of the "Arm Cortex®-M7 Platform" chapter of the *i.MX RT1050 Series Reference Manual* (IMXRT1050RM).

Table 2 summarizes errata on the i.MX RT1050.

Table 2. Summary of Silicon Errata

Errata	Name	Solution	Page
ADC			
ERR011164	ADC: ADC_ETC fails to clear the ADC_ETC request signals automatically after receiving DMA ack	A0 Erratum, fixed in A1 silicon	7
CCM			
ERR006223	CCM: Failure to resume from Wait/Stop mode with power gating	No fix scheduled	8
ERR050143	CCM: SoC will enter low power mode before the Arm CPU excutes WFI when improper low power sequence is used	No fix scheduled	9
ERR050235	CCM: Incorrect clock setting for CAN affects UART clock gating	No fix scheduled	10
Cortex-M7			
ERR011572	Cortex-M7: Write-Trough stores and loads may return incorrect data	No fix scheduled	11
ERR011573	Core: Speculative access might be performed to memory unmapped in MPU	No fix scheduled	12
DCDC			
ERR011092	DCDC: Some power up sequence may result in DCDC startup failure	A0 Erratum, fixed in A1 silicon	14
ERR011093	DCDC: Unexpected DCDC reset occurs on some chips	A0 Erratum, fixed in A1 silicon	15
FlexCAN			
ERR005829	FlexCAN: FlexCAN does not transmit a message that is enabled to be transmitted in a specific moment during the arbitration process	No fix scheduled	16
ERR006032	FlexCAN: A frame with wrong ID or payload is transmitted into the CAN bus when the Message Buffer under transmission is either aborted or deactivated while the CAN bus is in the Bus Idle state	No fix scheduled	18
ERR009527	FlexCAN: The transmission abort mechanism may not work properly	No fix scheduled	20
ERR009595	FlexCAN: Corrupt frame possible if the Freeze Mode or the Low-Power Mode are entered during a Bus-Off state	No fix scheduled	21
FlexSPI			
ERR011207	FlexSPI: In rare condition when FLEXSPI_AHBCR[PREFETCHEN] is set, incorrect data can be returned	A0 Erratum, fixed in A1 silicon	23
ERR011377	FlexSPI: FlexSPI DLL lock status bit not accurate due to timing issue	No fix scheduled	24
IO			
ERR011091	IO: High POR current if NVCC_EMC/NVCC_GPIO/NVCC_SD0/NVCC_SD1 is powered ahead of VDD_SOC_IN	A0 Erratum, fixed in A1 silicon	25
LCDIF			

Table 2. Summary of Silicon Errata (continued)

Errata	Name	Solution	Page
ERR011138	LCDIF: LUT consecutive programming may fail in case two writes are close	A0 Erratum, fixed in A1 silicon	26
LPSPI			
ERR011097	LPSPI: Command word does not load properly when TXMSK = 1	No fix scheduled	27
ERR050606	LPSPI: TCR value does not get resampled when polling the register	No fix scheduled	28
ERR050607	LPSPI: TCR[FRAMSZ] can be ignored when TCR[TXMSK] = 1b	No fix scheduled	29
QTMR			
ERR050194	QTMR: Overflow flag and related interrupt cannot be generated when the timer is configured as upward count mode	No fix scheduled	30
PIT			
ERR050130	PIT: Temporary incorrect value reported in LMTR64H register in lifetimer mode	No fix scheduled	31
RTWDOG			
ERR011111	RTWDOG: Chip stuck in reset when RTWDOG low byte test mode is enabled	A0 Erratum, fixed in A1 silicon	32
SAI			
ERR011096	SAI: The internal bit clock cannot be generated when BCI = 1	No fix scheduled	33
ERR050144	SAI: Setting FCONT = 1 when TMR > 0 may not function correctly	No fix scheduled	34
SEMC			
ERR011225	SEMC: CPU AXI writes to SEMC NAND memory may cause incorrect data programmed into NAND memory	No fix scheduled	35
ERR050577	SEMC: Auto-refresh can fail to be triggered during long back to back write (or read)	No fix scheduled	36
SOC			
ERR050538	SOC: Potential boot failure on system reset if SJC_DISABLE fuse is blown	No fix scheduled	37
SNVS			
ERR011165	SNVS: Invalid ECC check failure	A0 Erratum, fixed in A1 silicon	38
System Boot			
ERR011262	System Boot: ROM cannot boot from SEMC NAND with default fuse setting	Fixed in A1 silicon	39
ERR011110	System Boot: SEMC NOR boot cannot support the signed image authentication under HAB closed mode	A0 Erratum, fixed in A1 silicon	40
ERR011119	System Boot: FlexSPI NOR encrypted XIP boot fails after system reset if the FAC region number is less than 2	A0 Erratum, fixed in A1 silicon	41
ERR011120	System Boot: FlexSPI NOR encrypted XIP boot fails after system reset if the IOMUXC_GPR18 to IOMUXC_GPR21 are locked	A0 Erratum, fixed in A1 silicon	42
USB			

Table 2. Summary of Silicon Errata (continued)

Errata	Name	Solution	Page
ERR010661	USB: VBUS leakage occurs if USBOTG1 VBUS is on and USBOTG2 VBUS transitions from on to off	No fix scheduled	43
ERR050101	USB: Endpoint conflict issue in device mode	Migration to i.MX RT1060 new revision	44

ERR011164 ADC: ADC_ETC fails to clear the ADC_ETC request signals automatically after receiving DMA ack

Description:

If enable ADC_ETC to trigger DMA transfer, the DMA transfer data send ack out. It does not clear the request signals automatically and continue to trigger DMA.

Projected Impact:

This issue can lead to DMA failure when working with ADC_ETC.

Workarounds:

Configuring two DMA channels for ADC_ETC data transfer. The first DMA channel with low priority triggered by ADC_ETC request is to transfer ADC_ETC data. The second DMA channel with high priority links to the first channel is to clear request of ADC_ETC by writing DMA_CTRL register. Both channel's priority need to be higher than any channel used by other peripherals. This solution is result in DMA to transfer ADC_ETC data twice for one request signal and application need handle the redundant data properly.

Proposed Solution:

Fixed in A1 silicon

Software Status:

Software workaround is not in SDK.

ERR006223 CCM: Failure to resume from Wait/Stop mode with power gating

Description:

When entering Wait/Stop mode with power gating of the Arm core(s), if an interrupt arrives during the power-down sequence, the system could enter an unexpected state and fail to resume.

Projected Impact:

Device might fail to resume from low-power state.

Workarounds:

Use REG_BYPASS_COUNTER (RBC) to hold off interrupts when the PGC unit is in the middle of the power-down sequence. The counter needs to be set/cleared only when there are no interrupts pending. The counter needs to be enabled as close to the WFI (Wait For Interrupt) state as possible.

The PREG_BYPASS_COUNT value is equal or greater than 2.

Proposed Solution:

No fix scheduled

Software Status:

Software workaround in SDK

ERR050143 CCM: SoC will enter low power mode before the Arm CPU excutes WFI when improper low power sequence is used

Description:

When software tries to enter the low power mode with the following sequence, SoC enters the low power mode before the Arm CPU executes the WFI instructions.

- Set CCM_CLPCR[1:0] to 2’b00
- Arm CPU enters WFI
- Arm CPU wakes up from an interrupt event, which is masked by GPC or not visible to GPC, such as an interrupt from local timer.
- Set CCM_CLPCR[1:0] to 2’b01 or 2’b10
- Arm CPU executes WFI

Before the last step, SoC enters the WAIT mode if CCM_CLPCR[1:0] is set to 2’b01, or enters the STOP mode if CCM_CLPCR[1:0] is set to 2’b10.

Workarounds:

Software workaround:

1) Trigger IRQ #41 (IOMUX), which is always pending by setting IOMUXC_GPR_GPR1[12] bit

2) Unmask IRQ #41 in GPC before setting the CCM low power mode

3) Mask IRQ #41 right after the CCM low power mode is set (set CCM_CLPCR[1:0])

Proposed Solution:

No fix scheduled

Software Status:

Software workaround in SDK

ERR050235 CCM: Incorrect clock setting for CAN affects UART clock gating

Description:

When selecting the CCM CAN clock source with CAN_CLK_SEL set to 2, the UART clock gate will not open and CAN_CLK_ROOT will be off. To avoid this issue, set CAN_CLK_SEL to 0 or 1 for CAN clock selection, or open the UART clock gate by configuring the CCM_CCGRx register.

Workarounds:

There are two workarounds:

- Set CAN_CLK_SEL to 0 or 1 for CAN clock selection, or
- If CAN_CLK_SEL is set to 2, then the CCM must open any of UART clock gate by configuring the CCM_CCGRx register.

Proposed Solution:

No fix scheduled

Software Status:

Software workaround in SDK

ERR011572 Cortex-M7: Write-Trough stores and loads may return incorrect data

Description:

Arm errata 1259864

If a particular sequence of stores and loads is performed on the Cortex-M7 core to Write-Through memory, and some timing-based internal conditions are met, then a load may not have the last data stored to that address.

This erratum can only occur if the loads and stores are to Write-Through memory. The following methods enable write-through mode of the cache:

1. The Memory Protection Unit (MPU) has been programmed to set this address as Write-Through.
2. The default memory map is being used, and this address is Write-Through in the default memory map.
3. The memory is cacheable, and the CM7_CACR.FORCEWT bit is set.
4. The memory is cacheable, shared, and the CM7_CACR.SIWT bit is set.

Following sequence is required for this erratum to occur:

1. The address of interest must be in the data cache.
2. A Write-Through store is performed to the same double-word as the address of interest.
3. One of the following:

- A linefill is started (to a different cacheline to the address of interest) that allocates to the same set and way as the address of interest.
- An Error Correcting Code (ECC) error is observed anywhere in the data cache.
- A data cache maintenance operation without a following Data Synchronization Barrier (DSB).

4. A store to the address of interest.
5. A load to the address of interest.

If certain specific timing conditions are met, the load get the data from the first store, or from what was in the cache at the start of the sequence instead of the data from the second store.

Under these conditions, a load can return incorrect data.

Workarounds:

There is no direct workaround for this erratum.

Where possible, Arm is recommended that using the MPU to change the attributes on any Write-Through memory to Write-Back memory. If this is not possible, it might be necessary to disable the cache for sections of code that access Write-Through memory.

Proposed Solution:

No fix scheduled

ERR011573 Core: Speculative access might be performed to memory unmapped in MPU

Description:

Arm errata 1013783-B

Cortex®–M7 can perform speculative memory accesses to Normal memory for various reasons. All other type of memory should never be subject to speculative accesses.

The memory attributes for a given address are defined by the settings of the MPU when it is enabled. Regions that are not mapped in the MPU do not have any explicit attributes and should not be subject to any speculative accesses.

Because of this erratum, Cortex®–M7 can incorrectly perform speculative accesses to such unmapped regions.

Conditions:

To trigger this erratum, the data cache must be enabled and the MPU must be enabled with the default memory map disabled. That is :

- CCR.DC = 1; data cache is enabled.
- MPU_CTRL.ENABLE = 1; MPU is enabled.
- If MPU_CTRL.PRIVDEFNA = 1, then this erratum cannot occur from privileged mode.
- If MPU_CTRL.HFNMIENA = 1, then this erratum cannot occur from the NMI or HF handlers or exception handlers when FAULTMASK = 1.

In these situations, a PLD instruction targeting an unmapped region might result in an incorrect speculative access. The PLD instruction itself could be speculative because of branch prediction. Even a literal data value that corresponds to a PLD encoding could theoretically cause this issue. This makes it difficult to scan code to check if these conditions apply.

Therefore, Arm recommends that any software with the MPU and data cache configured as mentioned in the conditions above uses the workaround below.

Implications:

Processor execution is not directly affected by this erratum. The data returned from the speculative access is never used and if the access is inferred by the program, then an abort will be taken as required.

The only implications of this erratum are the access itself which should not have been performed. This might have an impact on memory regions with side-effects on reads or on memory which never returns a response on the bus.

Workarounds:

Instead of leaving memory unmapped, software should use MPU region 0 to cover all unmapped memory and make this region execute-never and inaccessible. That is, MPU_RASR0 should be programmed with:

- MPU_RASR0.ENABLE = 1; MPU region 0 enable.
- MPU_RASR0.SIZE = b11111; MPU region 0 size = 2^32 bytes to cover entire memory.

- MPU_RASR0.SRD = b00000000; All sub-regions enabled.
- MPU_RASR0.XN = 1; Execute-never to prevent instruction fetch.
- MPU_RASR0.AP = b000; No read or write access for any privilege level.
- MPU_RASR0.TEX = b000; Attributes = Strongly-ordered.
- MPU_RASR0.C = b0; Attributes = Strongly-ordered.
- MPU_RASR0.B = b0; Attributes = Strongly-ordered.

Note that the MPU supports addressing hitting in multiple regions with the highest numbered region taking priority.

Therefore, use of MPU region 0 in this way does not affect the existing organization and use of MPU regions.

Software Status:

Software workaround is in SDK

ERR011092 DCDC: Some power up sequence may result in DCDC startup failure

Description:

When NVCC_SNVS_IN is powered earlier than or with DCDC_IN/DCDC_IN_Q, DCDC may not be able to start up and no output on DCDC_LP.

Projected Impact:

This issue leads chip failed to power up when using on-chip DCDC.

Workarounds:

Apply power to DCDC_IN/DCDC_IN_Q with DCDC_PSWITCH pin, 1ms ahead of VDD_SNVS_IN.

Proposed Solution:

Fixed in A1 silicon

Software Status:

No software workaround available

ERR011093 DCDC: Unexpected DCDC reset occurs on some chips

Description:

The possibility and frequent of the occurrence increases with DCDC_IN/DCDC_IN_Q voltage. When the unexpected reset happens, DCDC output drops first, then recovers after unexpected reset completed. In rare case, DCDC might not be able to recover and have output on DCDC_LP. This issue affects both CCM and DCM mode.

Projected Impact:

This issue results in system error, lock up reset, or power loss of VDD_SOC_IN during chip running when using on-chip DCDC.

Workarounds:

Power supply to the DCDC_IN/DCDC_IN_Q should be >= 2.8 V and <= 3.0 V

Proposed Solution:

Fixed in A1 silicon

Software Status:

No software workaround available

ERR005829 FlexCAN: FlexCAN does not transmit a message that is enabled to be transmitted in a specific moment during the arbitration process

Description:

FlexCAN does not transmit a message that is enabled to be transmitted in a specific moment during the arbitration process. The following conditions are necessary for the issue to occur:

- Only one message buffer is configured to be transmitted.
- The write which enables the message buffer to be transmitted (write on Control/Status word) happens during a specific clock during the arbitration process.
- After this arbitration process occurs, the bus goes to the Idle state and no new message is received on the bus.

For example:

1. Message buffer 13 is deactivated on RxIntermission (write 0x0 to the CODE field from the Control/Status word) [First write to CODE]
2. Reconfigure the ID and data fields
3. Enable the message buffer 13 to be transmitted on BusIdle (write 0xC on CODE field) [Second write to CODE]
4. CAN bus keeps in Idle state
5. No write on the Control/Status from any message buffer happens.

During the second write to CODE (step 3), the write must happen one clock before the current message buffer 13 to be scanned by arbitration process. In this case, it does not detect the new code (0xC) and no new arbitration is scheduled.

The problem can be detected only if the message traffic ceases and the CAN bus enters into Idle state after the described sequence of events.

There is no issue if any of the conditions below holds:

- Any message buffer (either Tx or Rx) is reconfigured (by writing to its CS field) just after the Intermission field.
- There are other configured message buffers to be transmitted.
- A new incoming message sent by any external node starts just after the Intermission field.

Projected Impact:

FlexCAN does not transmit a message that is enabled to be transmitted in a specific moment.

Workarounds:

To transmit a CAN frame, the CPU must prepare a message buffer for transmission by executing the following standard 5-step procedure:

1. Check if the respective interrupt bit is set and clear it.

2. If the message buffer is active (transmission pending), write the ABORT code (0b1001) to the CODE field of the Control/Status word to request an abortion of the transmission. Wait for the corresponding IFLAG to be asserted by polling the IFLAG register or by the interrupt request if enabled by the respective IMASK. Then read back the CODE field to check if the transmission was aborted or transmitted. If backwards compatibility is desired (MCR[AEN] bit negated), just write the INACTIVE code (0b1000) to the CODE field to inactivate the message buffer, but then the pending frame might be transmitted without notification.
3. Write the ID word.
4. Write the data bytes.
5. Write the DLC, Control, and CODE fields of the Control/Status word to activate the message buffer.
6. The workaround consists of executing two extra steps:
7. Reserve the first valid mailbox as an inactive mailbox (CODE = 0b1000). If RX FIFO is disabled, this mailbox must be message buffer 0. Otherwise, the first valid mailbox can be found using the “RX FIFO filters” table in the FlexCAN chapter of the chip reference manual.
8. Write twice INACTIVE code (0b1000) into the first valid mailbox.

NOTE

The first mailbox cannot be used for reception or transmission process.

Proposed Solution:

No fix scheduled

Software Status:

Software workaround is not in SDK.

ERR006032 FlexCAN: A frame with wrong ID or payload is transmitted into the CAN bus when the Message Buffer under transmission is either aborted or deactivated while the CAN bus is in the Bus Idle state

Description:

The FlexCAN module may transmit an incorrect message if one or more Message Buffers (MBs) are configured for transmission while FlexCAN is in the Bus Idle state, and the MB selected for transmission is either aborted or deactivated at the exact moment it starts to be transmitted. This will cause FlexCAN to transmit a syntactically correct message, but with either incorrect ID or data field. The CRC information will be calculated over the incorrect data (in case data is affected) and all other fields of the frame will be correct.

The probability of the problem occuring is limited to one CAN bit during the transmission of one frame, how ever under a very specific combination of simultaneous events:

1. Bug event may take place in one specific CAN bit per frame.
2. The CAN bus must be in Bus Idle state.
3. The CPU must be triggered to configure one or more MBs for transmission while in the Bus Idle state.
4. The CPU must be triggered to remove the MBs just configured, by abortion or deactivation, in a short period after starting the configuration in step 3.

In summary, the probability of occurrence is very low, in the order of 1 per 10 million. Moreover, the procedure of configuring a MB followed by abortion or deactivation of the same MB in a short intervals is unlikely to occur in normal applications.

In practice, there is no issue if the CPU guarantees that any MB configured for transmission will be aborted or deactivated just in the next frame.

Workarounds:

The user can avoid the error by preventing to make MB configurations for transmission when the CAN bus is in the Bus Idle state.

There are bits in a FlexCAN debug register can be used to determine when the CAN bus is in the Idle state.

This debug register is located at: FlexCAN Debug 1 Register (CAN_DBG1) - Base + 0x0058

The CAN Finite State Machine (CFSM) bits of CAN_DBG1 register monitor the FlexCAN's internal state. The CFSM is the six least significant bits of the CAN_DBG1 register. The CAN Bit Number (CBN) is the five bits long field at bit position 3 to 7 in the CAN_DBG1 register that indicates the current bit number in a given CFSM state value.

CAN_DBG1.CFSM = 0x0000_003F

CAN_DBG1.CBN = 0x1F00_0000

There are several internal states values that need to be looked for, listed below with their corresponding CFSM value.

RXINTERMISSION - 0x2F

TXINTERMISSION - 0x14

BUSIDLE - 0x02

The following procedure must be performed to configure a MB transmission:

1. Disable all interrupt
2. Read CAN_DBG1.CFSM and CAN_DBG1.CBN fields.
3. Check if CFSM value is either BUSIDLE, RXINTERMISSION or TXINTERMISSION.

For the later two values, also check if CBN value is 3, to determine the paired conditions RXINTERMISSION bit 3 or TXINTERMISSION bit 3, and processed as described below.

3.1. If CAN_DBG1 fields indicate BUSIDLE, wait N CPU clocks.

3.2. Else if CAN_DBG1 fields indicate either RXINTERMISSION bit 3 or TXINTERMISSION bit 3 wait until CFSM is different from either RXINTERMISSION or TXINTERMISSION.

3.3. Check again CAN_DBG1 fields, if they indicate BUSIDLE, wait for DELAY time.

4. Write 0x0 into Code field of CS word.
5. Enable all interrupts.
6. Write the ID word.
7. Write the DATA words.
8. Write 0xC into Code field of CS word.

NOTE

DELAY = {2 x (MAXMB + 1) + 18} x peripheral_clock_period + 3 x PE_clock_period + 1 x CAN_bit_period

The Number of the Last Message Buffer (MAXMB) are the 7 least significant bits in the Module Configuration Register (CAN_MCR: base + 0x0).

Proposed Solution:

No fix scheduled

Software Status:

Software workaround is not in SDK.

ERR009527 FlexCAN: The transmission abort mechanism may not work properly

Description:

The Flexible Controller Area Network (FlexCAN) is not able to abort a transmission frame and the abort process may remain pending on the following cases:

1. If a pending abort request occurs while the FlexCAN is receiving a remote frame.
2. When a frame is aborted during an overload frame after a frame reception.
3. When an abort is requested while the FlexCAN has just started a transmission.
4. When the Freeze Mode request occurs and the FlexCAN has just started a transmission.

Workarounds:

Use the Mailbox Inactivation mechanism instead of the transmission abort mechanism. The Abort Enable (AEN) bit of the Module Configuration Register can be kept cleared and the abort code value “0b1001” cannot be written into the CODE field of the Message Buffer Control and Status word.

Proposed Solution:

No fix scheduled

Software Status:

Software workaround is not in SDK.

ERR009595 FlexCAN: Corrupt frame possible if the Freeze Mode or the Low-Power Mode are entered during a Bus-Off state

Description:

If the Freeze Enable bit (FRZ) of the Module Configuration Register (MCR) is asserted and the Freeze Mode is requested by asserting the Halt bit (HALT) of the MCR register during the Bus Off state, the transmission after exiting the Bus-Off condition will be corrupted. The issue occurs only if a transmission is pending before the Freeze Mode request. In addition, the same issue can happen if the Low-Power Mode is requested instead of the Freeze Mode.

Workarounds:

The workaround depends on whether the Bus-Off condition occurs prior to requesting Freeze Mode or the Low-Power Mode.

Procedure to enter the Freeze Mode:

1. Set the Freeze Enable bit (FRZ) in the Module Control Register (MCR).
2. Check if the Module Disable bit (MDIS) in the MCR register is set. If yes, clear the MDIS bit.
3. Poll the MCR register until the Low-Power Mode Acknowledge (LPMACK) bit in the MCR is cleared (timeout for software are implementation is two CAN bits length).
4. Read the Fault Confinement State (FLTCONF) field in the Error and Status 1 Register (ESR1) to check if FlexCAN is in bus-off state. If yes, go to the step 5A. Otherwise, go to step 5B.

5A. Set the Soft Reset bit ((SOFTRST) in the MCR.

6A. Poll the MCR register until the Soft Reset (SOFTRST) bit is cleared (timeout for software are implementation is two CAN bits length.

7A. Poll the MCR register until the Freeze Acknowledge (FRZACK) bit is set (timeout for software are implementation is two CAN bit length).

8A. Reconfigure the MCR.

9A. Reconfigure all the Interrupt Mask Registers (IMASKn).

5B. Set the Halt Flex CAN (HALT) bit in the MCR.

6B. Poll the MCR register until the Freeze Acknowledge (FRZACK) bit is set (timeout for software are implementation is 178 CAN bits length).

NOTE

The time between step 4 and step 5B must be less than 1353 CAN bit periods.

Procedure to enter the Low-Power Mode:

1. Enter the Freeze Mode (execute the procedure A).
2. Request the Low-Power Mode.

3. Poll the MCR register until the Low-Power Mode Acknowledge (LPMACK) bit in the MCR is set (timeout for software are implementation is two CAN bit length).

Proposed Solution:

No fix scheduled

Software Status:

Software workaround is not in SDK.

ERR011207 FlexSPI: In rare condition when FLEXSPI_AHBCR[PREFETCHEN] is set, incorrect data can be returned

Description:

When prefetching is enabled (FLEXSPI_AHB[PREFETCHEN]) for non-cacheable space, there are conditions write-read order may not be guaranteed. The problem can occur if data is written and then read back using the AHB interface, while a region containing the data location is in the process of being loaded into the FlexSPI's AHB Rx buffer.

Workarounds:

There are two workarounds:

- If FlexSPI space is not cached (configured as device or strongly-ordered type in the MPU), then FLEXSPI_AHBRXBUFnCR0[PREFETCHEN] should be cleared.
- If the write is critical and the following read is to the same address, FlexSPI_STS0[SEQIDLE] bit can be checked to make sure the write has completed (SEQIDLE is 1) before issuing the subsequent read.

Proposed Solution:

Fixed in A1 silicon

Software Status:

Software workaround is not in SDK.

ERR011377 FlexSPI: FlexSPI DLL lock status bit not accurate due to timing issue

Description:

After configuring DLL and the lock status bit is set, the data may be wrong if read/write immediately from FLEXSPI based external flash due to timing issue.

Workarounds:

Add delay time (100 NOP) again after the DLL lock status is set.

Proposed Solution:

No fix scheduled

Software Status:

Software workaround is in SDK.

ERR011091 IO: High POR current if NVCC_EMC/NVCC_GPIO/NVCC_SD0/NVCC_SD1 is powered ahead of VDD_SOC_IN

Description:

When NVCC_EMC/NVCC_GPIO/NVCC_SD0/NVCC_SD1 is powered and VDD_SOC_IN is not powered, it will be high leakage current during the period of VDD_SOC_IN absence. The leakage current on NVCC_EMC/NVCC_GPIO could be high enough to damage the chip permanently. There could also be abnormal leakage on VDD_SNVS_IN when it is powered up, however the period is very short and the current is not high enough to damage the chip.

Projected Impact:

This issue causes high leakage power on IO supply, even permanent chip damage during power up stage, or anytime VDD_SOC_IN is absent but IO is powered.

Workarounds:

When using on-chip DCDC, apply power to DCDC_IN/DCDC_IN_Q 2ms ahead of NVCC_EMC/NVCC_GPIO/NVCC_SD0/NVCC_SD1. If on-chip DCDC is not used, apply power to VDD_SOC_IN ahead of NVCC_EMC/NVCC_GPIO/NVCC_SD0/NVCC_SD1.

Proposed Solution:

Fixed in A1 silicon

Software Status:

No software workaround available

ERR011138 LCDIF: LUT consecutive programming may fail in case two writes are close

Description:

When two writes to LCDIF_LUT0/1_DATA register are very close, data may not be programmed into LUT successfully. It is related with CPU, IP bus, and LDCIF DISPLAY CLOCK frequency. More faster the CPU runs, more slower the DISPLAY CLOCK is, more likely this issue happens.

Projected Impact:

LUT may not be programmed successfully and LCDIF output data is incorrect.

Workarounds:

Delay need be inserted between two LUT writes. The safe delay should be 6 BUS CLOCK (apb_clk) cycles + 3 DISPLAY CLOCK (pix_clk) cycles.

Proposed Solution:

Fixed in A1 silicon

Software Status:

Software workaround in SDK

ERR011097 LPSPI: Command word does not load properly when TXMSK = 1

Description:

When writing the TCR[TXMSK] = 1 in the Transmit Command Register and next write to the TX FIFO is another command, the first command may not load properly.

Projected Impact:

Command word does not load properly and LPSPI does not work properly.

Workarounds:

When writing the TCR[TXMSK] = 1 in the Transmit Command Register, wait for the TX FIFO to empty (FSR[TXCOUNT] = 0) before writing another command to the Transmit Command Register.

Proposed Solution:

No fix scheduled

Software Status:

Software workaround is not in SDK.

ERR050606 LPSPI: TCR value does not get resampled when polling the register

Description:

Following a write to the Transmit Command register (TCR), if the user continuously reads the TCR (polls the register), then the read content no longer represents the contents of the Transmit Command register if it updates due to internal logic following the first read. The same value shall continue to be read.

Workarounds:

After reading the Transmit Command Register must always access a different register in between subsequent reads from TCR.

Proposed Solution:

No fix scheduled

Software Status:

Software workaround is not in SDK.

ERR050607 LPSPI: TCR[FRAMSZ] can be ignored when TCR[TXMSK] = 1b

Description:

Transmit Command Register (TCR) is used to write new command word to the LPSPI transmit FIFO.

TCR[FRAMESZ] configures the frame size of the data to be transmitted in number of bits equal to (FRAMESZ + 1). When TCR[TXMSK] is set, transmit data is masked (no data is loaded from transmit FIFO and output pin is tristated). In master mode, the Transmit Data Mask bit will initiate a new transfer which cannot be aborted by another command word; the Transmit Data Mask bit will be cleared by hardware at the end of the transfer. TCR[CONTC] controls the continuous transfer mode. TCR[CONTC]=1b1 enables continuous transfer. In master mode, continuous transfer will keep the PCS asserted at the end of the frame size, until a command word is received that starts a new frame.

If command word is written with TCR[TXMSK]=1 and TCR[FRAMESZ]>32 and the next command word with TCR[CONTC]=0 is in the FIFO then at the end of any 32-bit word of the first command, the frame will terminate early and negate PCS.

Workarounds:

There are two workarounds:

1. Do not write a second command word after writing command word with TXMSK = 1 and FRAMESZ > 32 until the first one has completed. OR
2. Divide the command word into multiple command words with TXMSK = 1 and FRAMESZ = 32 (or remainder) using a continuous transfer.

Proposed Solution:

No fix scheduled

Software Status:

Software workaround is not in SDK.

ERR050194 QTMR: Overflow flag and related interrupt cannot be generated when the timer is configured as upward count mode

Description:

1. Overflow flag and related interrupt cannot be generated successfully in upward count mode.
2. When TMR_CTRL[OUTMODE] is set to 110b, OFLAG output is not cleared on counter rollover when the timer counts upward.

Workarounds:

For item 1, using compare interrupt instead of overflow interrupt by setting compare value to 0xFFFF. The compare interrupt has the same timing effect as overflow interrupt in this way.

For item 2, there is no workaround.

Proposed Solution:

No fix scheduled

Software Status:

Software workaround is not in SDK.

ERR050130 PIT: Temporary incorrect value reported in LMTR64H register in lifetimer mode

Description:

When the Programmable interrupt timer (PIT) module is used in lifetimer mode, timer 0 and timer 1 are chained and the timer load start value (LDVAL0[TSV] and LDVAL1[TSV]) are set according to the application need for both timers. When timer 0 current time value (CVAL0[TVL]) reaches 0x0 and subsequently reloads to LDVAL0[TSV], then timer 1 CVAL1[TVL] should decrement by 0x1.

However this decrement does not occur until one cycle later, therefore a read of the PIT upper lifetime timer register (LTMR64H) is followed by a read of the PIT lower lifetime timer register (LTMR64L) at the instant when timer 0 has reloaded to LDVAL0[TSV] and timer 1 is yet to be decremented in next cycle then an incorrect timer value in LTMR64H[LTH] is expected.

Workarounds:

In lifetimer mode, if the read value of LTMR64L[LTL] is equal to LDVAL0[TSV], then read both LTMR64H and LTMR64L registers for one additional time to obtain the correct lifetime value.

Proposed Solution:

No fix scheduled

Software Status:

Software workaround is not in SDK.

ERR011111 RTWDOG: Chip stuck in reset when RTWDOG low byte test mode is enabled

Description:

When the TST bits of the RTWDOG CS register are set to b10, the watchdog low byte test mode will be enabled. In this mode, CNT[CNTLOW] is compared with TOVAL[TOVALLOW]. System reset happens when they are equal. Then the chip stuck in the reset.

Projected Impact:

RTWDOG test mode cannot be used to test low byte of the watchdog counter.

Workarounds:

Do not set the TST bits of the RTWDOG CD register to b10. If watchdog test is required, set the TST bits to b10 in the User mode to test low byte of the watchdog counter.

Proposed Solution:

Fixed in A1 silicon

Software Status:

No software workaround available

ERR011096 SAI: The internal bit clock cannot be generated when BCI = 1

Description:

When SAI transmitter or receiver is configured for the internal bit clock with BCI = 1, the bit clock cannot be generated for either of the following two configurations:

1. SYNC = 00 and BCS = 0
2. SYNC = 01 and BCS = 1

Projected Impact:

The SAI bit clock cannot be generated properly.

Workarounds:

When SAI transmitter or receiver is configured for the internal bit clock with BCI = 1, using one of the following two configurations:

1. SYNC = 01 and BCS = 0
2. SYNC = 00 and BCS = 1

Proposed Solution:

No fix scheduled

Software Status:

Software workaround is not in SDK.

ERR050144 SAI: Setting FCONT = 1 when TMR > 0 may not function correctly

Description:

When FCONT = 1 the transmitter will recover after a FIFO error when the FIFO is no longer empty and starting again from the same word in the following frame where the error occurred.

Configuring TMR > 0 will configure one or more words in the frame to be masked (nothing transmitted during that slot). If anything other than the last word(s) in the frame are masked when FCONT = 1 and a FIFO Error Flag is set, then the transmitter will not recover and will set FIFO Error Flag during each frame.

Workarounds:

To avoid this issue, set FCONT in TCR4 to be 0.

Proposed Solution:

No fix scheduled

Software Status:

Software workaround is not in SDK.

ERR011225 SEMC: CPU AXI writes to SEMC NAND memory may cause incorrect data programmed into NAND memory

Description:

When SEMC NAND memory region is Normal type, non-cacheable, cacheable write-through, or writeback, non-allocate, and not hit, CM7 AXI writes to the region could program incorrect data to the NAND memory.

Projected Impact:

CPU cannot perform AXI write to SEMC NAND memory when it is the Normal memory type.

Workarounds:

1. Set SEMC NAND memory region to Device type or Strongly-ordered type in MPU, and CPU only perform 32-bit write to SEMC NAND memory region or;
2. Use eDMA to perform 64-bit AXI write to SEMC NAND memory region or;
3. Use IP command to program SEMC NAND memory.

Proposed Solution:

No fix scheduled

Software Status:

Software workaround is not in SDK.

ERR050577 SEMC: Auto-refresh can fail to be triggered during long back to back write (or read)

Description:

Auto-refresh command can fail to be triggered during long time back to back write (or read) when SDRAM controller burst length is greater than 1. Missing an auto-refresh command can lead to corruption of SDRAM.

Workarounds:

Configure burst length to 1 by writing register SDRAMCR0.

Proposed Solution:

No fix scheduled

Software Status:

Software workaround is in SDK.

ERR050538 SOC: Potential boot failure on system reset if SJC_DISABLE fuse is blown

Description:

By default, the JTAG/SWD clock is pulled high reset. When the SJC_DISABLE fuse is blown the clock is low. The fuses are reloaded during a system reset, so there is a window between the system reset assertion and fuse loading completion, during which the clock is high. When the fuses are loaded the clock will go low, causing a transition from high to low. There is another clock transition from low to high on a subsequent system reset. The clock toggles can cause the system to think JTAG/SWD is active. This causes a security violation leading to HAB boot failure.

Workarounds:

Configure the appropriate IOMUXC_SW_PAD_CTL register's PUE and PUS fields to enable a pull resistor on one of the following signals:

- Pull JTAG_TCK/SWD_CLK low
- Pull JTAG_TRST low
- Pull JTAG_TMS/SWD_DIO high

The IOMUXC registers retain state on a system reset, so this only needs to be done one time after each POR.

Proposed Solution:

No fix scheduled

Software Status:

Software workaround is not in SDK.

ERR011165 SNVS: Invalid ECC check failure

Description:

When setting LPMKCR[ZMK_ECC_EN] bit, it may generate ZMK ECC Check Failure Violation even the ZMK and its ECC values are correct.

Projected Impact:

ZMK is not usable in case the ZMK ECC check is enabled.

Workarounds:

Not enable ZMK ECC check

Proposed Solution:

Fixed in A1 silicon

Software Status:

Software workaround is not in SDK.

ERR011262 System Boot: ROM cannot boot from SEMC NAND with default fuse setting

Description:

ROM cannot boot from SEMC NAND with default fuse setting because the parameter configuration of SEMC AXI read is incorrect.

For SEMC NAND AXI read, tWHR is not the one defined in the ONFI standard. Instead of tWHR, it is tCSS, which needs to be read from device. It is unique to device, there is no uniform standard.

Projected Impact:

ROM cannot boot from SEMC NAND with default fuse setting.

Workarounds:

Configure FUSE to select non-ONFI device (BOOT_CFG2[0] = 1’b1) and enable EDO mode (0x6E0[4] = 1’b1), then AXI read can work properly (because the ONFI timing mode is selected by default, and tWHR in the ONFI timing mode 0 is long enough for AXI read). The NAND configuration block in FCB is able to adjust tWHR parameter. So it can switch to any timing mode.

Proposed Solution:

Fixed in A1 silicon

ERR011110 System Boot: SEMC NOR boot cannot support the signed image authentication under HAB closed mode

Description:

When the device is configured as HAB closed mode, BootROM cannot boot from SEMC NOR device because the signed image authentication is not supported.

Projected Impact:

This issue primarily impacts the application that requires Parallel NOR boot.

Workarounds:

Do not boot application via Parallel NOR under HAB closed mode.

Proposed Solution:

Fixed in A1 silicon

Software Status:

No software workaround available

ERR011119 System Boot: FlexSPI NOR encrypted XIP boot fails after system reset if the FAC region number is less than 2

Description:

If the total FAC region number in PRDB0 and PRDB1 is less than 2, ROM fails to boot from FlexSPI NOR on RT1051 and RT1052 A0 chip after system reset.

Conditions:

There are two conditions cause a boot failure:

- Only PRDB0 is available and FAC region is 1 in PRDB0.
- Only PRDB1 is available and FAC region is 1 in PRDB1.

Projected Impact:

User application cannot run after WDOG reset or system reset.

Workarounds:

There are two workarounds:

- Set IOMUXC_GPR20 = 0 and IOMUXC_GPR21 = 0 in user application after boot.
- Enable more than one FAC region in PRDB0 and PRDB1.

Proposed Solution:

Fixed in A1 silicon

Software Status:

Software workaround is not in SDK.

ERR011120 System Boot: FlexSPI NOR encrypted XIP boot fails after system reset if the IOMUXC_GPR18 to IOMUXC_GPR21 are locked

Description:

If the IOMUXC_GPR18 to IOMUXC_GPR_21 are locked by either BootROM or the user application, ROM fails to boot from FlexSPI NOR on RT1051 and RT1052 A0 chip after system reset.

Conditions:

There are two conditions cause a boot failure:

- Enable the lock option in PRDB0 or PRDB1.
- Enable the lock bit for IOMUXC_GPR18 to IOMUXC_GPR21 in user application.

Projected Impact:

User application cannot run after WDOG reset or system reset.

Workarounds:

Not available

Proposed Solution:

Fixed in A1 silicon

Software Status:

No software workaround available

ERR010661 USB: VBUS leakage occurs if USBOTG1 VBUS is on and USBOTG2 VBUS transitions from on to off

Description:

When two USB ports work as OTG or device simultaneously. One VBUS (selected by PMU_REG_3P0.vbus_sel bit) voltage will not drop after cable unplug, causing the port to fail to detect the cable detach. If these two ports do not need to support detach detection, simultaneously using two OTGs or devices can be supported.

Conditions:

When two USB ports work as OTGs or devices simultaneously.

Projected Impact:

Do not use two OTGs or devices simultaneously. Only four scenarios are supported:

- One for OTG/Device, another for Host.
- One for OTG/Device, another is un-used.
- One for Host, another for Host.
- One for Host, another is un-used.

Workarounds:

Only one port can be used as OTG or device. The other port must be used as host. Set the PMU_REG_3P0.vbus_sel bit to select the host port.

Proposed Solution:

No fix scheduled

Software Status:

No software workaround available

ERR050101 USB: Endpoint conflict issue in device mode

Description:

An endpoint conflict occurs when the USB is working in device mode and an isochronous IN endpoint exists.

When the endpointA IN direction is an isochronous IN endpoint, and the host sends an IN token to endpointA on another device, then the OUT transaction may be missed regardless the OUT endpoint number. Generally, this occurs when the device is connected to the host through a hub and other devices are connected to the same hub.

The affected OUT endpoint can be either control, bulk, isochronous, or an interrupt endpoint.

After the OUT endpoint is primed, if an IN token to the same endpoint number on another device is received, then the OUT endpoint may be unprimed (Cannot be detected by SW), which causes this endpoint to no longer respond to the host OUT token, and thus, no corresponding interrupt occurs.

Workarounds:

Do not connect to a hub in the case when ISO IN endpoint(s) is used. When the hub(s) must be connected in this scenario, the endpoint number(s) of the ISO IN endpoint(s) should be different from the endpoint number(s) of any type of IN endpoint(s) used in any other device(s) connected to the same host.

Proposed Solution:

Migration to i.MX RT1060 new revision

How to Reach Us:

Home Page:
nxp.com

Web Support:
nxp.com/support

Document Number: IMXRT1050CE
Rev. 3
11/2021

arm

NXP