NXP Semiconductors
Chip Errata

Document Number: IMXRT1064CE
Rev. 0, 10/2018

Chip Errata for the i.MX RT1064

This document details the silicon errata known at the time of publication for the i.MX RT1064 crossover processors.

Table 1 provides a revision history for this document.

Table 1. Document Revision History

Rev. Number	Date	Substantive Changes
Rev. 0	10/2018	• Initial version

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

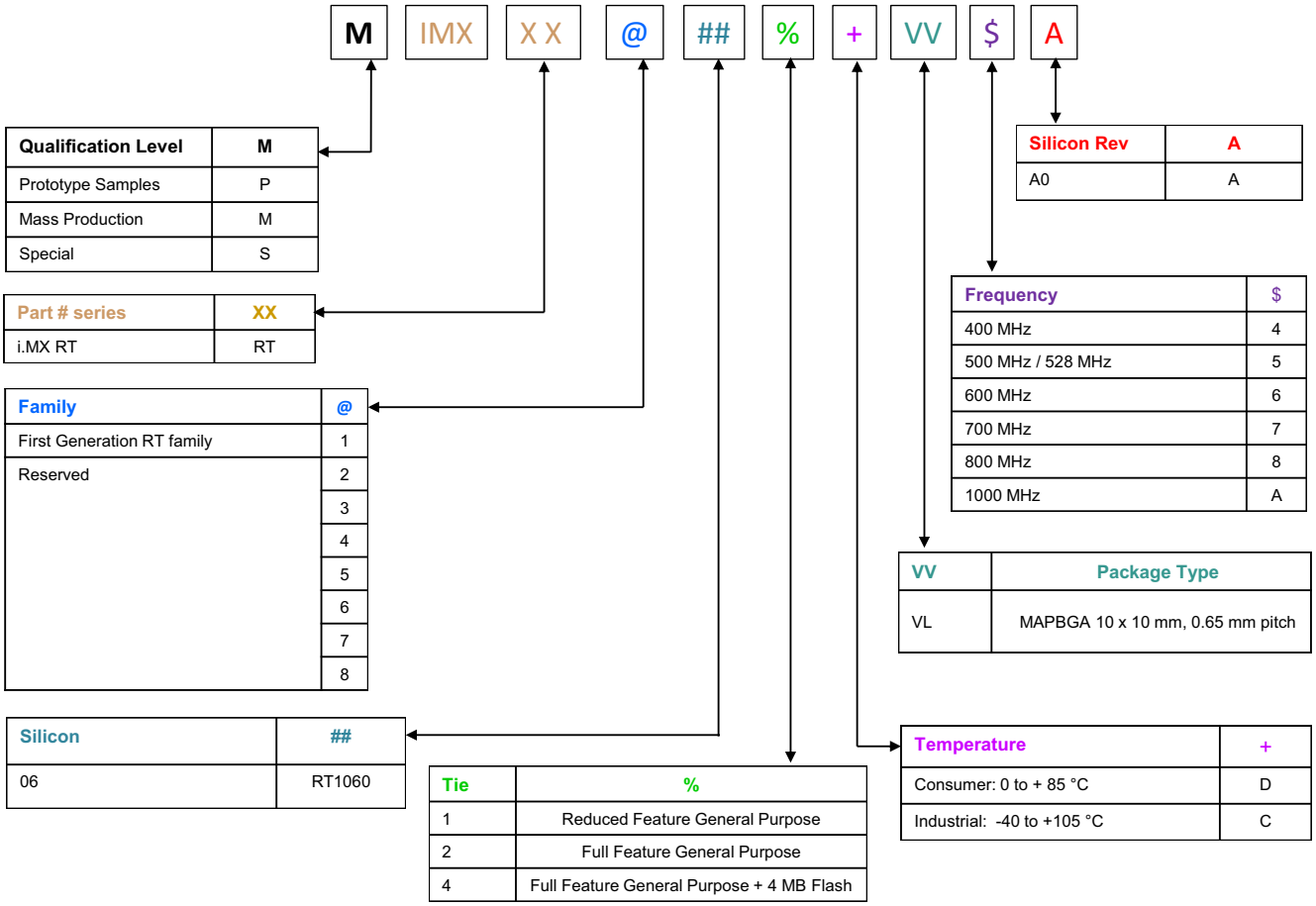

Figure 1. Revision Level to Part Marking Cross-Reference

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 RT1064 Series Reference Manual* (IMXRT1064RM).

Table 2 summarizes errata on the i.MX RT1064.

Table 2. Summary of Silicon Errata

Errata	Name	Solution	Page
CCM			
ERR006223	CCM: Failure to resume from Wait/Stop mode with power gating	No fix scheduled	4
ERR007265	CCM: When improper low-power sequence is used, the SoC enters low power mode before the Arm core executes WFI	No fix scheduled	5
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	6
ERR009527	FlexCAN: The transmission abort mechanism may not work properly	No fix scheduled	8
ERR009595	FlexCAN: Corrupt frame possible if the Freeze Mode or the Low-Power Mode are entered during a Bus-Off state	No fix scheduled	9
FlexSPI			
ERR011377	FlexSPI: FlexSPI DLL lock status bit not accurate due to timing issue	No fix scheduled	11
SEMC			
ERR011225	SEMC: CPU AXI writes to SEMC NAND memory may cause incorrect data programmed into NAND memory	No fix scheduled	12
USB			
ERR006281	USB: Incorrect DP/DN state when only VBUS is applied	No fix scheduled	13
ERR010661	USB: VBUS leakage occurs if USBOTG1 VBUS is on and USBOTG2 VBUS transitions from on to off	No fix scheduled	14

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

ERR007265 CCM: When improper low-power sequence is used, the SoC enters low power mode before the Arm core executes WFI

Description:

When software tries to enter Low-Power mode with the following sequence, the SoC enters Low-Power mode before the Arm core executes the WFI instruction:

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

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

Projected Impact:

This issue can lead to errors ranging from module underrun errors to system hangs depending on the specific use case.

Workarounds:

Software workaround:

1. Software should trigger IRQ #41 (GPR_IRQ) to be always pending by setting IOMUXC_GPR_GPR1_GINT.
2. Software should then unmask IRQ #41 in GPC before setting CCM Low-Power mode.
3. Software should mask IRQ #41 right after CCM Low-Power mode is set (set bits 0-1 of CCM_CLPCR).

Proposed Solution:

No fix scheduled

Software Status:

Software workaround in SDK

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.

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.

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:

No software workaround available

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.

ERR006281 USB: Incorrect DP/DN state when only VBUS is applied

Description:

When VBUS is applied without any other supplies, incorrect communication states are possible on the data (DP/DN) signals. If VDDHIGH_IN is supplied, the problem is removed.

Projected Impact:

This issue primarily impacts applications using charger detection to signal power modes to a PMIC in an undercharged battery scenario where the standard USB current allotment is not sufficient to boot the system.

Workarounds:

Apply VDDHIGH_IN if battery charge detection is needed. Otherwise, disable charger detection by setting the EN_B bit in USB_ANALOG_USBx_CHRG_DETECTn to 1.

Proposed Solution:

No fix scheduled

Software Status:

Software workaround is not in SDK.

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

How to Reach Us:

Home Page:
nxp.com

Web Support:
nxp.com/support

Document Number: IMXRT1064CE
Rev. 0
10/2018