Mask Set Errata for Mask 1P33A

This report applies to mask 1P33A for these products:

- MIMX8ML8CVNKZAB
- MIMX8ML8DVNLZAB
- MIMX8ML6CVNKZAB
- MIMX8ML6DVNLZAB
- MIMX8ML4CVNKZAB
- MIMX8ML4DVNLZAB
- MIMX8ML3CVNKZAB
- MIMX8ML3DVNLZAB

Table 1. Errata and Information Summary

Erratum ID	Erratum Title
ERR003774	AIPS: Unaligned access to AIPS internal registers will result in an abort response.
ERR050461	Cortex-A53 incorrect exclusive access to DDR when using non-cached variables
ERR050711	DDR: Activate Commands blocked in Per-Bank Refresh Mode for LPDDR4.
ERR050712	DDR: Register corruption possible when software triggered mode register (MR) operations performed in DDR4 mode.
ERR009535	ECSPI: Burst completion by SS signal in slave mode is not functional
ERR009606	ECSPI: In master mode, burst lengths of 32n+1 will transmit incorrect data
ERR050697	ENET_QOS: Failure to generate Fatal Bus Error interrupt when descriptor posted write is enabled
ERR050694	ENET_QOS: MAC incorrectly discards the received packets when Preamble Byte does not precede SFD or SMD
ERR050698	ENET_QOS: Scheduled transmit packet not sent in the allotted slot or the remaining fragment of a Preempted Packet incorrectly dropped due to scheduling timeout in the EST GCL
ERR011543	FlexCAN: Nominal Phase SJW incorrectly applied at CRC Delimiter
ERR050246	FlexCAN: Receive Message Buffers may have its Code Field corrupted if the Receive FIFO function is used
ERR050537	FlexSPI: Read timing sequence mismatches with several existing SPI NOR devices in dual, quad, and octal modes
ERR007805	I2C: When the I2C clock speed is configured for 400 kHz, the SCL low period violates the I2C spec of 1.3 uS min

Table continues on the next page...

Table 1. Errata and Information Summary (continued)

Erratum ID	Erratum Title
ERR050531	NOC: VPU_NOC power down handshake may hang during VC8000E/VPUMIX power up/down cycling.
ERR050717	PCIe: LTSSM L2 state is not supported
ERR051128	PCIe: PCI Express PHY is subject to accelerated aging in lower power states
ERR050144	SAI: Setting FCONT=1 when TMR>0 may not function correctly
ERR050542	SAI: The Bit Count Timestamp Register (TBCTR, RBCTR) may return a live rather than latched Timestamp
ERR050462	SDMA: SDMA3 can’ t work when SDMA2 clock is OFF
ERR050714	USB: HOST Stream IN issue if received short packet
ERR050689	USB: USB3 device immediate wakeup in low power mode

Table 2. Revision History

Revision	Changes
1.0	Initial revision
2.0	Errata added: ERR051128

ERR003774: AIPS: Unaligned access to AIPS internal registers will result in an abort response.

Description: Unaligned access to AIPS internal registers will return an abort response.

Workaround: Only aligned AIPS internal register access is supported. Software should not issue unaligned accesses to AIPS internal registers.

ERR050461: Cortex-A53 incorrect exclusive access to DDR when using non-cached variables

Description: 1.The exclusive monitor used for transactions to DRAM has a bug which in ~ 1/1000000 cases allows a competing master to read old data with an EXOK status.

2. When DRAM memory region is marked as “cacheable”, the exclusive monitor inside the Cortex-A cluster is used and the exclusive accesses from Cortex-A cores do not reach the system interconnect or DRAM Controller, and so there is no issue.

3. However, in that case, the Cortex-M core cannot share use of the same locks/semaphores in DRAM as Cortex-A cores are using, because there is no coherency support between Cortex-A core cluster caches and the rest of the system.

4. Hence, if the Cortex-M and Cortex-A cores have to share the locks, then OCRAM region can be used, because it has an external monitor which is fully functional.

Workaround: Workaround 1. Use OCRAM with external monitor for spinlocks, instead of DDR.

Might not be suitable for all the codebase. Supports both Cortex-A and Cortex-M EXCL.

Workaround 2. For spinlocks used by Cortex-A cores in DDR, mark the memory as cacheable.

Utilize internal ARM exclusive monitor in the Cortex-A clusters. Does not support both Cortex-A and Cortex-M exclusiveaccess. To support both Cortex-A and Cortex-M sharing of data, it is recommended to use other mechanisms such as MU rather than exclusive access transactions

ERR050711: DDR: Activate Commands blocked in Per-Bank Refresh Mode for LPDDR4.

Description: When the optional per-bank refresh mode is enabled in the DDR Controller, activate commands are unnecessarily blocked in a rank until all postponed refresh commands are issued, leading to a potential impact on DRAM utilization.

This issue happens only when the all the following conditions are satisfied:

1. LPDDR4 mode enabled (MSTR.lpddr4=1).

2. Per-bank refresh is enabled. (RFSHCTL0.per_bank_refresh=1).

3. One or more per-bank refresh commands are critical in a rank. Critical here means (RFSHCTL0.refresh_burst + 1) number of per-bank refresh commands have been postponed.

Workaround: Set the DDR Controller register RFSHCTL.refresh_burst = 0 to minimize the performance impact when per-bank refresh mode is enabled (RFSHCTL0.per_bank_refresh=1).

ERR050712: DDR: Register corruption possible when software triggered mode register (MR) operations performed in DDR4 mode.

Description: The DDR controller allows user software to manually trigger a mode register read or write operation by setting the MRCTRL0.mr_wr=1. Under certain specific conditions listed below it is possible that DDR controller registers can be corrupted while an internal hardware driven MR access is occurring. The impact of the corruption depends on the registers being accessed.

This issue can only occur when:

1. DDR controller is configured in DDR4 mode

2. A mode register read or write operation is performed by user software by programming the register MRCTRL0.mr_wr=1

3. Separate DDR controller register R/W (or R/W1S or R/W1C) APB write accesses occur close together and

4. An internal hardware driven MR access occurs concurrently

4.a. Entering and exiting Self-Refresh or MPSM or when

4.b. Per DRAM Addressability mode (PDA) or Per Buffer Addressability(PBA) mode is enabled

Workaround: When performing a software driven MR access, the following polling sequence must be done automatically before performing other DDR controller register accesses:

1. Set the register MRCTRL0.mr_wr=1 (When the MR operation is complete, the DDRC automatically clears this bit)

2. Check the DDR Controller register MRSTAT.mr_wr_busy = 0 If not, go to step (2)

3. Check the DDR Controller register MRSTAT.mr_wr_busy = 0 again (for the second time). If not, go to step (2)

ERR009535: ECSPI: Burst completion by SS signal in slave mode is not functional

Description: According to the eCSPI specifications, when eCSPI is set to operate in the Slave mode (CHANNEL_MODE[x] = 0), the SS_CTL[x] bit controls the behavior of burst completion.

In the Slave mode, the SS_CTL bit should control the behavior of SPI burst completion as follows:

• 0—SPI burst completed when (BURST_LENGTH + 1) bits are received

• 1—SPI burst completed when the SS input is negated

Also, in BURST_LENGTH definition, it is stated “In the Slave mode, this field takes effect in SPI transfer only when SS_CTL is cleared.”

However, the mode SS_CTL[x] = 1 is not functional in Slave mode. Currently, BURST_LENGTH always defines the burst length.

According to the SPI protocol, negation of SSB always causes completion of the burst. However, due to the above issue, the data is not sampled correctly in RxFIFO when {BURST_LENGTH+1}mod32 is not equal to {actual burst length}mod32.

Therefore, setting the BURST_LENGTH parameter to a value greater than the actual burst does not resolve the issue.

Workaround: Do not use the SS_CTL[x] = 1 option in the Slave mode. The accurate burst length should always be specified using the BURST_LENGTH parameter.

ERR009606: ECSPI: In master mode, burst lengths of 32n+1 will transmit incorrect data

Description: When the ECSPI is configured in master mode and the burst length is configured to a value 32n+1 (where n=0,1, 2,…), the ECSPI will transmit the portions of the first word in the FIFO twice.

For example, if the transmit FIFO is loaded with:

[0] 0x00000001

[1] 0xAAAAAAAA

And the burst length is configured for 33 bits (ECSPIx_CONREG[BURST_LENGTH]=0x020), the ECSPI will transmit the first bit of word [0] followed by the entire word [0], then transmit the data as expected.

The transmitted sequence in this example will be:

[0] 0x00000001

[1] 0x00000001

[2] 0x00000000

[3] 0xAAAAAAAA

Workaround: Do not use burst lengths of 32n+1 (where n=0,1, 2,…).

ERR050697: ENET_QOS: Failure to generate Fatal Bus Error interrupt when descriptor posted write is enabled

Description: When a bus error occurs, DWC_ether_qos sets the Fatal Bus Error (FBE) status field in the DMA_CH(#i)_Statusregister of the corresponding DMA channel. The Fatal Bus Error interrupt is generated if the FBEE field of theDMA_CH(#i)_Interrupt_Enable register is set.

When posted write is enabled by setting the DSPW field of the DMA_Mode register to 1, the write channel of the AXI Master interface transfers an OKAY response to the DMA as soon as the last beat of the data is accepted by the AXI Bus interface. To improve the performance or throughput, the AXI Master interface sends this response without waiting for the write response from the AXI Bus interface.

However due to the defect, when the descriptor posted write is enabled, if a bus error occurs for a write transfer, the bus error response is undetected by the AXI Master interface. This can occur only when a write response from AXI Bus coincides with an internal dummy write response from the AXI Master interface to the DMA. Therefore, the Fatal Bus Error status is not set in the DMA_CH(#i)_Status register of corresponding DMA channel.

Impacted Use Cases:

The descriptor posted write is enabled in the configuration and AXI Bus/Slave/Interconnect to which theDWC_ether_qos is interfaced supports bus error generation for an incorrect write transfer.

Consequence:

As the bus error is neither detected, nor reported by the DWC_ether_qos, the software cannot take the required corrective action, which can lead to serious system errors.

Workaround: The software must not enable descriptor posted writes. Program the DSPW field to 0 in the DMA_Mode register.

ERR050694: ENET_QOS: MAC incorrectly discards the received packets when Preamble Byte does not precede SFD or SMD

Description: The IEEE 802.3 standard states that, in MII/GMII modes, the byte preceding the SFD (0xD5), SMD-S (0xE6,0x4C, 0x7F, or 0xB3), or SMD-C (0x61, 0x52, 0x9E, or 0x2A) byte can be a non-PREAMBLE byte or there can be no preceding preamble byte. The MAC receiver must successfully receive a packet without any preamble(0x55) byte preceding the SFD, SMD-S, or SMD-C byte.

However due to the defect, in configurations where frame preemption is enabled, when preamble byte does not precede the SFD, SMD-S, or SMD-C byte, the received packet is discarded by the MAC receiver. This is because, the start-of-packet detection logic of the MAC receiver incorrectly checks for a preamble byte.

Impacted Use Cases:

The frame preemption feature is enabled in the configuration and a packet is received without a preamble byte preceding the SFD, SMD-S, or SMD-C byte. The preamble byte might be absent due to any of the following reasons:

Remote transmitter transmits packets without preamble (chip-to-chip applications).

Preamble byte corrupted in transit.

Repeater removes the preamble byte.

Consequence:

The MAC receiver discards the received packet for which preamble byte does not precede the SFD, SMD-S, or SMD-C bytes, resulting in loss of data. However, applications that support retransmission can retransmit the packet. Retransmissions impact performance.

Workaround: If the remote transmitter (chip-to-chip application) does not transmit preamble byte, configure remote transmitter to transmit at the least one preamble byte preceding the SFD, SMD-S, or SMD-C byte.

If the preamble byte is corrupted in transit, it is a transient issue; only a few packets are impacted. No workaround is required.

A workaround not available for other cases.

ERR050698: ENET_QOS: Scheduled transmit packet not sent in the allotted slot or the remaining fragment of a Preempted Packet incorrectly dropped due to scheduling timeout in the EST GCL

Description: The EST (Enhancements to Scheduled Traffic) scheduler schedules the packet for transmission in a GCL slot, only when the sum of packet size (adjusted for change in the packet size due to offloads enabled in the MAC),fixed packet overhead, and the scheduler delay is less than the duration available in the GCL slot.

Also, when the first fragment or a few fragments of the preempted packet are transmitted, the MAC does not drop the remaining fragment because of the scheduling timeout or expire due to excessive express packet transmission. However due to the defect, the boundary size packets (packets that almost or exactly fit the GCL slot) are not scheduled in the expected GCL slot. This is because, the GCL slot is generated in the PTP clock domain and the scheduler operates in in the application clock domain. The sync delay of this asynchronous path introduces minor inaccuracies in the computation of the time available in the GCL slot. This results in the packet not being scheduled in the allotted GCL slot. The remaining fragment of the preempted packet is dropped because of scheduling timeout or expiry due to excessive express packet transmission.

Impacted Use Cases:

The GCL slot intervals are computed and programmed to exactly fit the packets to be scheduled by the EST scheduler. The DFBS field of the MTL_EST_Control register is set to 1, to drop packets that encounter scheduling error.

Consequences:

The packet is not scheduled in the expected GCL slot. Even if the packet is subsequently scheduled and transmitted, the remote receiver drops the packet, as the packet is not received at the expected time. This results in loss of data.

The remaining fragment of the preempted packet is dropped. The remote receiver drops the packet as it does not receive the complete packet. This results in loss of data.

Workaround: Program the GCL slot interval to be larger than the sum of the packet size (adjusted for packet size change due to offloads enabled in MAC), fixed packet overhead, and scheduler delay, by at least the maximum sync delay between the PTP clock domain and application clock domain (for example, 1 clock period of PTP clock and 3 clock periods of application clock).

Disable the dropping of packets that encounter scheduling error, by programming the DFBS field of the MTL_EST_Control register to 0.

ERR011543: FlexCAN: Nominal Phase SJW incorrectly applied at CRC Delimiter

Description: During the reception of a CAN-FD frame when the Bit Rate Switch (BRS) is enabled, the Synchronization Jump Width (SJW) for the CRC Delimiter bit is incorrectly defined by the Nominal Phase SJW. The CAN specification stipulates that the CRC Delimiter bit should have a SJW set by the Data Phase SJW.

When a resynchronization event is triggered for the CRC delimiter bit (recessive in correct operation), the sample point will be adjusted by an amount as defined by the Nominal Phase SJW rather than the specified Data Phase SJW. This may result in the incorrect detection of a dominant bit leading to a CAN error frame. However, as the CRC delimiter bit position will only apply the SJW upon the detection of an unexpected dominant bit on the CAN bus, an error frame is already likely. For the case the SJW is applied at the CRC delimiter and a recessive bit is not detected, the receiving node will issue an error frame.

The CAN protocol is designed to handle resynchronization errors and hence the CAN bus will recover from the insertion of the incorrect SJW at the CRC delimiter. Upon detecting the error frame the transmitting node will re-transmit the frame.

The following FlexCAN configurations are not affected:

- Classical CAN frames (CAN 2.0B)
- CAN FD frames with bit rate switch disabled (BRS = 0)
- CAN FD frames with Nominal Phase SJW equal to Data Phase SJW
- CAN FD transmissions

Configuration for the FlexCAN:

- Nominal Phase SJW is configured by the Resync Jump Width bit in the CAN Control Register 1 (CAN_CTRL1[RJW]) or by the Extended Resync Jump Width bit in the CAN Bit Timing Register (CAN_CBT[ERJW])
- Data Phase SJW is configured by the Fast Resync Jump Width bit in the CAN FD Bit Timing Register (CAN_FDCBT[FRJW])

Workaround: The robustness of the CAN protocol ensures that the receiver automatically recovers from the application of the incorrect SJW. The CAN protocol is designed to recover from resynchronization errors and hence any frame that is not correctly received will be re-sent by the transmitting node.

ERR050246: FlexCAN: Receive Message Buffers may have its Code Field corrupted if the Receive FIFO function is used

Description: If the Code Field of a Receive Message Buffer is corrupted it may deactivate the Message Buffer, so it is unable to receive new messages. It may also turn a Receive Message Buffer into any type of Message Buffer as defined in the Message buffer structure section in the device documentation.

The Code Field of the FlexCAN Receive Message Buffers (MB) may get corrupted if the following sequence occurs.

1- A message is received and transferred to an MB (i.e. MBx)

2- MBx is locked by software for more than 20 CAN bit times (time determines the probability of erratum to manifest).

3- SMB0 (Serial Message Buffer 0) receives a message (i.e. message1) intended for MBx, but destination is locked by the software (as depicted in point 2 above) and therefore NOT transferred to MBx.

4- A subsequent incoming message (i.e. message2) is being loaded into SMB1 (as SMB0 is full) and is evaluated by the FlexCAN hardware as being for the FIFO.

5- During the message2, the MBx is unlocked. Then, the content of SMB0 is transferred to MBx and the CODE field is updated with an incorrect value.

The problem does not occur in cases when only Rx FIFO or only a dedicated MB is used (i.e. either RX MB or Rx FIFO is used). The problem also does not occur when the Enhanced Rx FIFO and dedicated MB are used in the same application. The problem only occurs if the FlexCAN is programmed to receive in the Legacy FIFO and dedicated MB at the same application.

Workaround: This defect only applies if the Receive FIFO (Legacy Rx FIFO) is used. This feature is enabled by RFEN bit in the Module Control Register (MCR). If the Rx FIFO is not used, the Receive Message Buffer Code Field is not corrupted.

If available on the device, use the enhanced Rx FIFO feature instead of the Legacy Rx FIFO. The Enhanced Rx FIFO is enabled by the ERFEN bit in the Enhanced Rx FIFO Control Register (ERFCR).

The defect does not occur if the Receive Message Buffer lock time is less than or equal to the time equivalent to 20 x CAN bit time.

The recommended way for the CPU to service (read) the frame received in a mailbox is by the following procedure:

1. Read the Control and Status word of that mailbox.

2. Check if the BUSY bit is deasserted, indicating that the mailbox is not locked. Repeat step 1) while it is asserted.

3. Read the contents of the mailbox.

4. Clear the proper flag in the IFLAG register.

5. Read the Free Running Timer register (TIMER) to unlock the mailbox

In order to guarantee that this procedure occurs in less than 20 CAN bit times, the MB receive handling process in software (step 1 to step 5 above) should be performed as a ‘critical code section’ (interrupts disabled before execution) and should ensure that the MB receive handling occurs in a deterministic number of cycles.

ERR050537: FlexSPI: Read timing sequence mismatches with several existing SPI NOR devices in dual, quad, and octal modes

Description: The FlexSPI controller expects every read command has at least one latency cycle between address phase and data phase to account for turnaround time on the IO bus. In multiple IO modes such as dual, quad, and octal modes, the FlexSPI controller inserts one additional clock cycle following the address (or command modifier) phase in order to prevent contention on bidirectional IO pins.

It will cause drive conflict if the SPI NOR device’s timing sequence does not contain dummy cycles after the command/address cycles. Such drive conflict might result in reading wrong data value. The problem usually happens when reading a SPI slave’s register space.

Workaround: For FlexSPI memory device that supports multi IO Read command with zero latency cycle between address phase and data phase, use single line mode for read command, or use different data line to issue commands and read data.

The official NXP BSP release uses a signal line (1S-1S-1S) mode, but not multiple IO modes when access FlexSPI device registers.

ERR007805: I2C: When the I2C clock speed is configured for 400 kHz, the SCL low period violates the I2C spec of 1.3 uS min

Description: When the I2C module is programmed to operate at the maximum clock speed of 400 kHz (as defined by the I2C spec), the SCL clock low period violates the I2C spec of 1.3 uS min. The user must reduce the clock speed to obtain the SCL low time to meet the 1.3us I2C minimum required. This behavior means the SoC is not compliant to the I2C spec at 400kHz.

Workaround: To meet the clock low period requirement in fast speed mode, SCL must be configured to 384KHz or less.

ERR050531: NOC: VPU_NOC power down handshake may hang during VC8000E/ VPUMIX power up/down cycling.

Description: VC8000E reset de-assertion edge and AXI clock may have a timing issue.

Workaround: Set bit2 (vc8000e_clk_en) of BLK_CLK_EN_CSR to 0 to gate off both AXI clock and VC8000E clock sent to VC8000E and AXI clock sent to VPU_NOC m_v_2 interface during VC8000E power up(VC8000E reset is de-asserted by hardware).

ERR050717: PCIe: LTSSM L2 state is not supported

Description: PCIe cannot be configured to enter to LTSSM L2. This function is not supported by PCIe Phy IP.

Workaround: If PCIe operation is ever required for a given system design, all clocks to PCIe should remain active whenever the i.MX 8M Plus is powered. The lowest PCIe power mode supported by reference BSP is L1.

ERR051128: PCIe: PCI Express PHY is subject to accelerated aging in lower power states

Description: According to the PCI Express base specification, the PCI Express power management (PCI Express-PM) defines Link Power Management (Link PM) states, including L0, L0s, L1, L2/L3 Ready, L2 and L3. The power saving increases as the state transitions from L0 to L3. A PCI Express physical link is permitted to enter the Link PM states in response to either the legacy PCI-PM software-driven D-state transitions or native PCI Express Active State link Power Management (ASPM) activities. Unless otherwise stated, the states described in this erratum refer to the PCI Express Link PM states other than the Physical Layer Link Training and Status State Machine (LTSSM) states.

Among the Link PM states, the L1 refers to the lower power standby state while the L2 is defined as the low power sleep state.

As indicated in the L1 PM Substates Capabilities Register of the PCI Express controller, the device supports ASPM L1.1, PCI-PM L1.1 and PCI-PM L1.2, which are the substates of the L1 Link PM main state in addition to the L1 entry substate L1.0. The L1 Link PM state can be entered (when considered to be in L1.0 substate) with either of the following conditions:

- Entering PCI-PM L1.0 when the Physical Layer LTSSM state entered L1 through legacy PCI-PM mechanism, as the result of the controller's downstream link partner to be programmed to any D-state (PCI Device PM state) other than the fully-on D0 state.
- Entering ASPM L1.0 when the Physical Layer LTSSM state entered L1 through ASPM, as the result of automatic hardware handshake when the L1 Entry Enable bit is set in the Link Control Register of both link partners.

Once the Link enters L1.0 (regardless it's in PCI-PM L1.0 or ASPM L1.0), the PCI Express controller automatically transitions the Link to either L1.1 or L1.2 accordingly based on whichever is set among the ASPM L1.1 Enable, PCI-PM L1.1 Enable and PCI-PM L1.2 Enable bits in the controller's L1 PM Substates Control 1 Register, in response to the de-assertion of the CLKREQ# signal. As defined in the PCI Express base specification, the reference clock may be shut off in both L1.1 and L1.2 substates for further power saving, although the power of the PCI Express controller and PHY remains on.

The device also directs the PCI Express controller to transition the Link PM state to L2 whenever the device's Suspend Mode (also called Deep Sleep Mode (DSM)) is entered. The reference clock is shut off upon the entry of the L2 state, although the power of the PCI Express controller and PHY remains on since the device does not support the optional PCI Express auxiliary power rail (Vaux).

Due to this erratum, whenever the reference clock to the PCI Express PHY is shut off, the PLL of the PHY is turned off accordingly. Since the static power of the PCI Express PHY (VDD_PCI_0P8 and VDD_PCI_1P8) remains on at this time, the PHY is subject to accelerated aging which leads to the potential loss of the functionality specified in the datasheet.

Regardless of the source of the reference clock, the PCI Express PHY is subjected to accelerated aging in lower power states.

The accelerated aging means that due to this erratum, the affected PCI Express PHY of the device under the above stated conditions will reduce the device lifetime to less than that specified in the document AN13214 Rev0, i.MX 8M Plus Product Lifetime Usage.

Workaround: Below workaround must be followed to avoid the aging impact to the PCI Express PHY.

1. The following two guidelines must be followed regardless of the PCI Express link speed:

a. The PCI Express L1.1 and L1.2 Link PM substates must be disabled for all link speeds from Gen1 to Gen3. To disable the L1.1 and L1.2 Link PM substates, the software must not set any of the ASPM L1.1 Enable, PCI-PM L1.1 Enable and PCI-PM L1.2 Enable bits in the controller's L1 PM Substates Control 1 Register.

b. For any end product system that does not have EP connected currently however plans to use PCI Express in the future, the PCI Express clocks and power supplies must be kept on by default as described in the Known Issues/Limitations chapter of the NXP IMXLXRN i.MX Linux® Release Notes Rev. LF5.10.52_2.1.0 and beyond.

2. The device's Suspend Mode usage must follow the requirements below, which vary depending on the PCI Express link speed:

a. For systems with the PCI Express controller's link operating at Gen3 speed, the Suspend Mode is not supported.

b. For systems operating only at Gen1 and Gen2 speeds, Suspend Mode can only be used when the supply voltage is 0.85V (Typical) for both the VDD_PCI_0P8 and VDD_SOC.

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.

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

ERR050542: SAI: The Bit Count Timestamp Register (TBCTR, RBCTR) may return a live rather than latched Timestamp

Description: A SAI Timestamp Counter instance implements independent 32-bit counters for BCLK and a Timestamp based on the sub-system clock (AUDIO_AHB_CLK_ROOT, typically 400MHz). The current value of the timestamp count is latched on a BCLK edge and the contents of that latch is further latched into the xBCTR register whenever the BCLK count is read (xBCR). However, reading xBCR sometimes results in xBCTR latching the current value of the timestamp count, not the value latched on the most recent BCLK edge. This introduces uncertainty in the timestamp of up to 1 BCLK period.

Workaround: A BCLK period is sampling frequency and format dependent e.g. 142 sub-system clocks for 44.1kHz I2S or 33 sub-system clocks for 48kHz TDM8. These represent 3.5ppm or 825ppb respectively when measuring at 10Hz, compared to the 25ppb design aim. The uncertainty in the timestamp is instantaneous not accumulating and should be considered when designing any PLL or ASRC correction.

ERR050462: SDMA: SDMA3 can’ t work when SDMA2 clock is OFF

Description: SDMA_EVENTS_LOGIC is used to synchronize async event inputs to the SDMA clock domain. These synchronized events are shared by SDMA2 and SDMA3.

The clock input of “SDMA_EVENTS_LOGIC” is shared with SDMA2, so it will be gated off once AUDIOMIX_CLKEN0[SDMA2] is 0, then impacts async events to SDMA3.

For SDMA3 to work normally, the AUDIOMIX_CLKEN0[SDMA2] (bit-26) must be enabled, otherwise, the SDMA3 does NOT transfer the data.

Workaround: AUDIOMIX_CLKEN0[SDMA2] (bit-26) should be set for SDMA3 to work.

ERR050714: USB: HOST Stream IN issue if received short packet

Description: An issue may happen for IN Stream capable Bulk Endpoints, if the received wither short packet on TD where TRBs bufsize is more than 1 packet (1KB) or if received short packet with EOB=1 and TRB bufsize is more than received packet size.

When a streamID is selected for IN Endpoint, i.e. there is ongoing transfer on selected StreamID, and Host is receiving Short Data Packet (packet length is less than 1KB) and the packet receiver of that DP is TRB which has buffer more than 1KB, then if the Device selects another StreamID by sending ERDY as a StartStream transaction, the Host may not respond to the newly selected StreamID, even though SW(xHCI driver) has prepared a TRB for the new StreamID and the doorbell is rung.

System Usage Scenario:

When SW prepared TRB to receive data from device, host may not receive data as a result of this issue.

Consequence(s):

IN Stream transfers malfunction

Workaround: No software workaround.

ERR050689: USB: USB3 device immediate wakeup in low power mode

Description: When High Speed Input/Output (HSIO) sub-system goes into suspend low power mode, it immediately initiates the wakeup sequence. Thus, the USB3 cannot sustain low power mode for the proper duration until the next valid wakeup event

Workaround: Before the HSIO sub-system powers down, the USB3 wakeup interrupt can be disabled. Thus, when the USB3 is suspended, the wake up event will not propagate to initiate the wakeup sequence.

How to Reach Us:

Home Page:
nxp.com

Web Support:
nxp.com/support