Mask Set Errata for Mask 0N14Y

This report applies to mask 0N14Y for these products:

- MIMX8MN6DVTJZAA
- MIMX8MN6DVTJZCA
- MIMX8MN6DVTJZDA
- MIMX8MN5DVTJZAA
- MIMX8MN4DVTJZAA
- MIMX8MN3DVTJZAA
- MIMX8MN2DVTJZAA
- MIMX8MN1DVTJZAA
- MIMX8MN6CVTIZAA
- MIMX8MN5CVTIZAA
- MIMX8MN4CVTIZAA
- MIMX8MN3CVTIZAA
- MIMX8MN2CVTIZAA
- MIMX8MN1CVTIZAA

Table 1. Errata and Information Summary

Erratum ID	Erratum Title
ERR003774	AIPS: Unaligned access causes abort on writes to the internal registers
ERR050358	BSDL: The GPIO1_IO02 used as WDOG_B is set output low when entering boundary scan mode
ERR050310	CM7 Icache/Dcache are not operational
ERR050381	DRAM: DRAM data may be lost while exiting from DDR IO retention 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
ERR009165	ECSPI: TXFIFO empty flag glitch can cause the current FIFO transfer to be sent twice
ERR050226	GPU: Texture L2 Cache idle signal may incorrectly clock gate the texture engine in GPU
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
ERR050045	IOMUX: Setting ODE control bit of I2C IOs causes malfunction
ERR050350	ROM: Exception raised when ROM accesses a reserved region when Field Return fuse is enabled
ERR050359	ROM: USB Serial Download mode supports maximum 3 devices per USB host
ERR050362	TCM: AXI2AHB cannot handle partial write and causes redundant write operations to TCM

Table 2. Revision History

Revision	Changes
0	Initial revision

ERR003774: AIPS: Unaligned access causes abort on writes to the internal registers

Description: Unaligned access to AIPS can be driven high by SAHARA, DAP, and FEC. If they access the AIPS internal registers during an unaligned access, an ABORT occurs.

Workaround: Make only aligned accesses to the AIPS internal registers.

ERR050358: BSDL: The GPIO1_IO02 used as WDOG_B is set output low when entering boundary scan mode

Description: Only GPIO1_IO02 can be multiplexed as WDOG_B to toggle PMIC. When entering boundary scan mode, the GPIO1_IO02 is always low. If this pin is connected with PMIC WGOD_B during boundary scan mode, WDOG_B low will reset the PMIC and prevent normal system boot-up.

Workaround: If the PMIC supports WDOG reset by default, the PMIC WDOG_B pin cannot be connected to GPIO1_IO02 and should be pulled-up to a 100K ohm resistor during boundary scan test. Otherwise, use the WDOG timer buffer circuit. The related reference workaround circuit can be found from i.MX8M Nano Hardware Developer's Guide.

ERR050310: CM7 Icache/Dcache are not operational

Description: CM7 Icache/Dcache tag memories were incorrectly integrated, preventing the cache memories from working properly.

If the I-cache is enabled, there will be a slight performance degradation (based on cache not-enabled), with no other system implications. However, if the D-cache is enabled the system will return false cache hits and may cause unexpected system behavior.

Workaround: Do not enable the caches on the CM7 core. If the CM7 code size is below 256K, load and run the code from Tightly Coupled Memory (TCM).

The following steps outline loading and running CM7 code from the TCM memory:

Reg 0x30340054 defines the "CM7 Vector table offset register out of reset" and should be set as default 0x00000000 (ITCM) in this case.

Step 1: Reg 0x3039000c 0xaa // release reset, enable TCM

Step 2: Loadb program bin file to TCM

Step 3: Reg 0x30340058 0x0 //disable CPUWAIT, set CPU in run mode

Note: In CA53/system perspective: DTCM address is from 0x0080_0000 to 0x0081_FFFF, and ITCM address is from 0x007E_0000 to 0x007F_FFFF.

In CM7 perspective: DTCM address is from 0x2000_0000 to 0x2001_FFFF, and ITCM address is from 0x0000_0000 to 0x0001_FFFF.

ERR050381: DRAM: DRAM data may be lost while exiting from DDR IO retention mode

Description: There are two DDR PHY input signals PwrOkIn and atpg_mode that have combinational logic powered by VDD_DRAM. When DDR PHY exits from IO retention mode, VDD_DRAM will ramp-up during the time it is OFF, therefore the two signals cannot to be assured as 0 and will not meet the PHY requirement. The issue may cause the IO retention mode to work improperly and data in DRAM might be lost.

Workaround: Keep the VDD_DRAM on while in DDR IO retention mode.

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,…).

ERR009165: ECSPI: TXFIFO empty flag glitch can cause the current FIFO transfer to be sent twice

Description: When using DMA to transfer data to the TXFIFO, if the data is written to the TXFIFO during an active ECSPI data exchange, this can cause a glitch in the TXFIFO empty signal, resulting in the TXFIFO read pointer (TXCNT) not updating correctly, which in turn results in the current transfer getting resent a second time.

Workaround: This errata is only seen when the SMC (Start Mode Control) bit is set. A modified SDMA script with TX_THRESHOLD = 0 and using only the XCH (SPI Exchange) bit to initiate transfers prevents this errata from occurring. There is an associated performance impact with this workaround. Testing transfers to a SPI-NOR flash showed approximately a 5% drop in write data rates and a 25% drop in read data rates.

ERR050226: GPU: Texture L2 Cache idle signal may incorrectly clock gate the texture engine in GPU

Description: While running certain graphics cases, the Texture Engine's L2 Cache is waiting too long for more data to be returned from the AXI bus, hence the L2 cache will become idle and incorrectly clock gate the Texture Engine, leading to an eventual hang.

Workaround: GC7000UL GPU module level clock gating enables by default. As a workaround, add an exception in the GPU software to disable TX module level clock gating. The GPU has multiple levels of power management one at the module clock gating and GPU driver level power management. The GPU driver level power management will be used with this workaround.

The related driver is located in the file "drivers/mxc/gpu-viv/hal/kernel/arch/gc_hal_kernel_hardware.c". The code detail are as follows:

```
gcmkONERROR(
gckOS_WriteRegisterEx(Hardware->os,
Hardware->core,
0x00414,
data));
}
```

```
if (_IsHardwareMatch(Hardware, gcv4000, 0x5222) || _IsHardwareMatch(Hardware, gcv2000,
0x5108) || _IsHardwareMatch(Hardware, gcv7000, 0x6202) || _IsHardwareMatch(Hardware,
gcv7000, 0x6203) || (gckHARDWARE_IsFeatureAvailable(Hardware,
gcvFEATURE_TX_DESCRIPTOR) && !gckHARDWARE_IsFeatureAvailable(Hardware,
gcvFEATURE_TX_DESC_CACHE_CLOCKGATE_FIX) ) ) { if (regPMC == 0)
{ gcmkONERROR( gckOS_ReadRegisterEx(Hardware->os, Hardware->core,
```

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.

ERR050045: IOMUX: Setting ODE control bit of I2C IOs causes malfunction

Description: The I2C module supports open drain. The I2C module drives the open-drain signal of the output data. However, setting the ODE bit in the I2C IOMUXC registers results in malfunctions due to internal logic.

Workaround: Do not set the ODE bit in the I2C IOMUX registers because I2C module already supports open drain.

ERR050350: ROM: Exception raised when ROM accesses a reserved region when Field Return fuse is enabled

Description: A Slave Error exception (SLVERR) is generated by the hardware when attempting to access a reserved region in field return mode. Because the exception interrupt is masked in the ROM the pending error defers to the bootloader and will cause the system to hang if not handled correctly.

Workaround: Because the Arm architecture does not provide any mechanisms to clear pending exceptions before they are taken, users must correctly handle the exception. Hence the initial bootloader is required to handle the exception generated by the core before proceeding further execution. The NXP provided SPL bootloader handles this exception.

Below are the steps a typical application may have to handle for the SError exception:

1. Add an exception handler for system error (SError) in the vector table. The offset 0x380 of the vector table should be used to set the handler for SError exception. Exceptions raised while rom is in execution happens at exception Level 3 (EL3).

2. Clear the SError exception mask. By default, ROM masks the asynchronous exceptions. The mask bit should be cleared at an early stage of application startup code so that exception handler routine gets executed to take the exception.

3. The exception handler routine should return for the very first SError exception if the field return fuse bit is blown, this condition guarantees that the exception is raised due to ROM execution. For all other conditions, application should handle the exception as the case may be.

ERR050359: ROM: USB Serial Download mode supports maximum 3 devices per USB host

Description: The ROM USB HID driver supports up to a maximum of 3 devices per host for simultaneous download, when the SOC is in Serial Download mode.

Workaround: Serial Download Mode allows up to 3 devices to be connected to a single USB host channel. If more than 3 devices are required, the user must have additional USB host channels available. Limit the number of devices simultaneously connected on PC to 3 per USB host.

ERR050362: TCM: AXI2AHB cannot handle partial write and causes redundant write operations to TCM

Description: The AXI2AHB bridge is used to access the TCM in CM7. On the AXI side, the bus data width is 64-bit with 8-bit write strobe signals.

For normal write operations at least one bit of the WSTB signal is asserted on the AXI side, therefore, AHB can handle the related write operations correctly.

For burst write operations there are no write strobe signals ASSERTED for some beats such as in the SDMA case. In this case, the first beats with write strobe asserted the write operations from AXI to AHB are correct; however, for the remaining beats without write strobe asserted in AXI (WSTB=0x00), AHB will write data from the invalid beats to TCM and cause error.

This issue impacts all masters that access TCM through system bus.

Workaround: Set bit-1 of register 0x32504044 to 1.

This enables the bridge to correctly handle the situation when there is write request but no write strobe is asserted in AXI side.

How to Reach Us:

Home Page:
nxp.com

Web Support:
nxp.com/support

Information in this document is provided solely to enable system and software implementers to use NXP products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits based on the information in this document. NXP reserves the right to make changes without further notice to any products herein.

NXP makes no warranty, representation, or guarantee regarding the suitability of its products for any particular purpose, nor does NXP assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. “Typical” parameters that may be provided in NXP data sheets and/or specifications can and do vary in different applications, and actual performance may vary over time. All operating parameters, including “typicals,” must be validated for each customer application by customer’s technical experts. NXP does not convey any license under its patent rights nor the rights of others. NXP sells products pursuant to standard terms and conditions of sale, which can be found at the following address: nxp.com/SalesTermsandConditions.

While NXP has implemented advanced security features, all products may be subject to unidentified vulnerabilities. Customers are responsible for the design and operation of their applications and products to reduce the effect of these vulnerabilities on customer's applications and products, and NXP accepts no liability for any vulnerability that is discovered. Customers should implement appropriate design and operating safeguards to minimize the risks associated with their applications and products.