SAMA7G5 Series Silicon Errata and Data Sheet Clarifications

SAMA7G5 Series

Product Page Links

Scope

The SAMA7G5 Series device that you have received conforms functionally to the current SAMA7G5 Series device data sheet (DS60001765) or SAMA7G5 Series System-in-Package (SiP) data sheet (DS50003577), except for the anomalies described in this document.

The silicon issues discussed in the following pages are for silicon revisions with the device revision and device identification listed in the following table. The silicon issues are summarized in Silicon Issue Summary.

Data Sheet clarifications and corrections (if applicable) are located in Data Sheet Clarifications.

The silicon device revisions and device IDs are shown in the following table.

Table 1. SAMA7G5 Series Device Identification

Ordering Code	Device Revision	Device Identification
		CHIPID_CIDR[31:0]
SAMA7G54(T)-V/4HB	A0	0x80162110
SAMA7G54(T)-V/4HB	A1	0x80162111
SAMA7G54(T)-E/4HB(VAO)		
SAMA7G54D1G(T)-I/4UB	A1-D1G	
SAMA7G54D2G(T)-I/4UB	A1-D2G	
SAMA7G54D4G(T)-I/4UB	A1-D4G	

Note: Refer to the “Chip Identifier (CHIPID)” and “Product Identification System” sections in the current device data sheet for detailed information on chip identification for your specific device.

1. Silicon Issue Summary

In this table and in subsequent sections, the following applies:

- "X" means the device revision is affected by the erratum.
- "–" means the device revision is not affected by the erratum.

Table 1-1. Silicon Issue Summary

Module	Erratum	Affected Device Revisions				
		A0	A1	A1-D1G	A1-D2G	A1-D4G
ROM Code	NAND Flash and Octal SPI boot not supported	X	X	X	X	X
	Boot failure on e.MMC memories	X	X	X	X	X
Cortex-A7	PMU interrupt spurious rise	X	X	X	X	X
	AXIERRIRQ interrupt spurious rise	X	X	X	X	X
EIC	WPVS bit incorrect behavior	X	X	X	X	X
XDMAC	Data corrupted when number of AXI outstanding transactions differs from 1	X	X	X	X	X
	Some XDMAC0 and XDMAC1 channels ineffective	X	–	–	–	–
RSTC	RSTC_SR.RSTTYP not showing GENERAL_RST	X	X	X	X	X
RTC	RTC_TSTR0 timestamping error	X	X	X	X	X
CHIPID	CHIPID_EXID may report a wrong value	–	–	X	X	–
OTPC	OTPC limited number of packets	X	X	X	X	X
	OTPC restricted operating range in Write mode	X	X	X	X	X
	OTPC wrong default configuration	X	X	X	X	X
PMC	MCKRDY flag error	X	X	X	X	X
	Delay to first establish PCK	X	X	X	X	X
	PCK and GCLK Ready status issue	X	X	X	X	X
	Processor (CPU_CLK0) and main system bus clock (MCK0) source selection	X	X	X	X	X
PIO	Open drain management limitation	X	X	X	X	X
ADC	Spurious effect when zeros written to ADC_EOC_IDR	X	X	X	X	X
	EOC interrupts not disabled when ones written to ADC_EOC_IDR	X	X	X	X	X
	ADC_IMR interrupts enabled when ones written to ADC_EOC_IDR	X	X	X	X	X
	Temperature sensor still enabled when stopped without conversion	X	X	X	X	X
	Temperature sensor spurious activation with CH30	X	X	X	X	X
	Sleep mode ineffective	X	X	X	X	X
ISC	Spurious DMA descriptor writing	X	X	X	X	X
	Incoming pixels corrupted after overload	X	X	X	X	X
	Frequency limitation	X	–	–	–	–
SSC	Inverted left/right channels	X	X	X	X	X
	TD output delay	X	X	X	X	X
SPDIFRX	SPDIFRX left/right inversion	X	X	X	X	X
AES	SPLIP mode does not work with some header sizes	X	X	X	X	X

Table 1-1. Silicon Issue Summary (continued)

Module	Erratum	Affected Device Revisions				
		A0	A1	A1-D1G	A1-D2G	A1-D4G
SECUMOD	Dynamic detection intrusion (PIOBU) alarm issue	X	X	X	X	X
	Tamper timestamping polarity error	X	X	X	X	X
	SECUMOD registers BMPR and WKPR reading issue	X	X	X	X	X
GMAC	GMAC0 not functional with multiple queues in 10/100 Half Duplex mode	X	X	X	X	X
	Incorrect reading of Specific Address filter registers on GMAC0 and GMAC1	X	–	–	–	–
	Incorrect reading of Type 1 Screener registers on GMAC0 and GMAC1	X	–	–	–	–
	Incorrect reading of Type 2 Screener registers on GMAC0 and GMAC1	X	–	–	–	–
	GTSUCOMP ineffective connection to TC1	X	X	X	X	X
FLEXCOM	Write Protection ineffective on FLEXCOM8 to FLEXCOM11	X	X	X	X	X
SDMMC	SDMMC failure when changing speed mode or performing ALL soft reset on the fly	X	X	X	X	X
	SDR104, HS200, HS400 modes are not functional	X	X	X	X	X
	GCLK cannot be stopped	X	X	X	X	X
	SDHC blocked after switch from high-speed mode	X	X	X	X	X
MCAN	Edge filtering causes mis-synchronization when falling edge at Rx input pin coincides with end of integration phase	X	X	X	X	X
	Configuration of MCAN_NBTP.NTSEG2 = '0' not allowed	X	X	X	X	X
	Retransmission in DAR mode due to lost arbitration at the first two identifier bits	X	X	X	X	X
	Tx FIFO message sequence inversion	X	X	X	X	X
	Unexpected High Priority Message (HPM) interrupt	X	X	X	X	X
	Issue message transmitted with wrong arbitration and control fields	X	X	X	X	X
	Debug message handling state machine not reset to Idle when CCCR.INIT is set	X	X	X	X	X
	Message order inversion when transmitting from dedicated Tx buffers configured with same message ID	X	X	X	X	X
	Frame transmitted despite confirmed transmit cancellation for CAN-FD messages with more than 8 data bytes	X	X	X	X	X
	MCAN_TSU_TSCFG reset after read	X	X	X	X	X
	MCAN_TSU_TSS1 not reset after a MCAN_TSU_TSx read	X	X	X	X	X
	MCAN_TSU_ATB read resets the timebase value	X	X	X	X	X
TC	TC0 Channel 2 registers incorrect reading	X	–	–	–	–
UDPHS	EHCI spurious stop when Suspend mode occurs on port A	X	X	X	X	X

Table 1-1. Silicon Issue Summary (continued)

Module	Erratum	Affected Device Revisions				
		A0	A1	A1-D1G	A1-D2G	A1-D4G
Low Power Modes	ULP2 mode does not work	X	X	X	X	X

2. ROM Code

2.1. NAND Flash and Octal SPI boot not supported

The ROM code does not support booting out of NAND Flash or Octal SPI.

Work Around

None

Affected Device Revisions

A0	A1	A1-D1G	A1-D2G	A1-D4G			
X	X	X	X	X			

2.2. Boot failure on e.MMC memories

The device fails to load a bootstrap program (boot.bin) from an e.MMC **USER** partition.

Work Around

Always use the e.MMC **BOOT** partition to store the boot.bin file and enable the e.MMC **BOOT** partition feature.

Affected Device Revisions

A0	A1	A1-D1G	A1-D2G	A1-D4G			
X	X	X	X	X			

3. Cortex-A7 Processor (Arm)

3.1. PMU interrupt spurious rise

The Performance Monitoring Unit (PMU) interrupt rises as soon as enabled.

Work Around

Do not enable the PMU interrupt.

Affected Device Revisions

A0	A1	A1-D1G	A1-D2G	A1-D4G			
X	X	X	X	X			

3.2. AXIERRIRQ interrupt spurious rise

The AXIERRIRQ interrupt rises as soon as enabled.

Work Around

Do not enable the AXIERRIRQ interrupt.

Affected Device Revisions

A0	A1	A1-D1G	A1-D2G	A1-D4G			
X	X	X	X	X			

4. External Interrupt Controller (EIC)

4.1. WPVS bit incorrect behavior

The EIC_WPSR.WPVS bit never rises to 1.

Work Around

None

Affected Device Revisions

A0	A1	A1-D1G	A1-D2G	A1-D4G			
X	X	X	X	X			

5. DMA Controller (XDMAC)

5.1. Data corrupted when number of AXI outstanding transactions differs from 1

Data corruption may occur when the number of AXI outstanding transactions differs from 1.

Work Around

Limit to 1 the number of AXI outstanding transactions to access the AHB part in NICGPV. Performance of multichannel DMA transfers to the AHB part (SRAM, EBI, QSPI) is slightly impacted. Performance of transfers to the DDR memory is not affected. Apply the following settings:

```
NICGPV->NICGPV_AMIB[6].NICGPV_AMIB_FN_MOD = 0x3;
NICGPV->NICGPV_AMIB[13].NICGPV_AMIB_FN_MOD = 0x3;
```

Affected Device Revisions

A0	A1	A1-D1G	A1-D2G	A1-D4G			
X	X	X	X	X			

5.2. Some XDMAC0 and XDMAC1 channels ineffective

Reading any register of channels 0, 1, 18, 19, 26 and 27 of XDMAC0 and XDMAC1 may return corrupted values.

Work Around

Do not use XDMAC0 and XDMAC1 channels 0, 1, 18, 19, 26 and 27.

Affected Device Revisions

A0	A1	A1-D1G	A1-D2G	A1-D4G			
X	–	–	–	–			

6. Reset Controller (RSTC)

6.1. RSTC_SR.RSTTYP not showing GENERAL_RST

In the Status register (RSTC_SR), the RSTTYP field shows BACKUP_RST instead of GENERAL_RST.

Work Around

None

Affected Device Revisions

A0	A1	A1-D1G	A1-D2G	A1-D4G			
X	X	X	X	X			

7. Real-time Clock (RTC)

7.1. RTC_TSTR0 timestamping error

RTC_TSTR0.TEVCNT fails to report the correct number of tamper event occurrences.

Work Around

None

Affected Device Revisions

A0	A1	A1-D1G	A1-D2G	A1-D4G			
X	X	X	X	X			

8. Chip Identifier (CHIPID)

8.1. CHIPID_EXID may report a wrong value

The CHIPID_EXID register for SAMA7G54D1G and SAMA7G54D2G may report a wrong value for devices with lot traceability code 2306W9M and 2330WHM, respectively (refer to the section “Marking” of the data sheet).

The value reported is 0 instead of 0x00000018 (SAMA7G54D1G) and 0x00000020 (SAMA7G54D2G).

Work Around

None

Affected Device Revisions

A0	A1	A1-D1G	A1-D2G	A1-D4G			
–	–	X	X	–			

9. OTP Controller (OTPC)

9.1. OTPC limited number of packets

The number of OTP packets allowed to be written in the user area, in addition to those necessary to configure the ROM code boot features, is limited to 2. The maximum size of the payload for each packet is 8192 bits.

Work Around

None

Affected Device Revisions

A0	A1	A1-D1G	A1-D2G	A1-D4G			
X	X	X	X	X			

9.2. OTPC restricted operating range in Write mode

The write operations in the OTPC cannot be performed over the full temperature and VDDIN33 power supply ranges specified.

Work Around

The write operations in the OTPC are restricted to the following ambient temperature and VDDIN33 power supply ranges:

- T_A = [0°C to 50°C]
- VDDIN33 = [3.15V to 3.6V]

Affected Device Revisions

A0	A1	A1-D1G	A1-D2G	A1-D4G			
X	X	X	X	X			

9.3. OTPC wrong default configuration

The default configuration of the OTPC cannot be used to access the OTP memory in Write mode.

Work Around

Prior to any write operation in the OTPC, the OTPC must be configured using the following code. This operation needs to be performed only once before the first write operation and whenever the peripheral reset (signal periph_nreset) is asserted.

```
#define ARRAY_SIZE(a) (sizeof(a) / sizeof((a)[0]))

/*
* writing one word lasts 350us
* the timeout was chosen to be enough for writing 10 words  **/
#define TIMEOUT 500000
#define OTPC_0    (0x1u << 0)
#define OTPC_1    16
#define OTPC_2    (0xffffu << OTPC_1)
#define OTPC_3    (0x4391u << OTPC_1)

static void otp_sama7g5_fixup(void)
{
       static const uint32_t fixup0[4] = {0x04194801, 0x01000000, 0x00000008, 0x00000000};
       static const uint32_t fixup1[4] = {0xfb164801, 0x4c017d12, 0x02120e01, 0x00004000};
       __IO uint32_t *OTPC_4   = (__IO uint32_t *)((uint8_t *)OTPC + 0x090);
       __IO uint32_t *OTPC_5 = (__IO uint32_t *)((uint8_t *)OTPC + 0x0A0);
       __IO uint32_t *OTPC_6 = (__IO uint32_t *)((uint8_t *)OTPC + 0x0B0);
       uint32_t timeout;
       int i;

       timeout = TIMEOUT;
       *OTPC_4 = OTPC_0 | OTPC_3;
```

```
        while (!(OTPC->OTPC_SR & OTPC_SR_UNLOCK) && --timeout > 0);

        for (i = 0; i < ARRAY_SIZE(fixup0); i++)
               OTPC_5[i] = fixup0[i];

        for (i = 0; i < ARRAY_SIZE(fixup1); i++)
               OTPC_6[i] = fixup1[i];

        timeout = TIMEOUT;
        *OTPC_4 = OTPC_3;
        while ((OTPC->OTPC_SR & OTPC_SR_UNLOCK) && --timeout > 0); }
```

Affected Device Revisions

A0	A1	A1-D1G	A1-D2G	A1-D4G			
X	X	X	X	X			

10. Power Management Controller (PMC)

10.1. MCKRDY flag error

When PMC_CPU_CKR.MDIV is greater than 1, if the PMC_CPU_CKR.CSS field is modified, the MCKRDY signal may be stuck at 0.

Work Around

Use a software timeout of 64 cycles of CPU clock instead of polling the MCKRDY bit.

Affected Device Revisions

A0	A1	A1-D1G	A1-D2G	A1-D4G			
X	X	X	X	X			

10.2. Delay to first establish PCK

When enabling a PCK after a reset, the delay before establishing the PCK with the correct frequency is 255 cycles of the PCK source clock. Once this delay has elapsed, and as long as the core reset is not asserted, there is no more additional delay when disabling/enabling the PCK.

Work Around

None

Affected Device Revisions

A0	A1	A1-D1G	A1-D2G	A1-D4G			
X	X	X	X	X			

10.3. PCK and GCLK Ready status issue

The PCK and GCLK Ready signals are only affected by the enable/disable of the corresponding clock (PMC_SCER.PCKx, PMC_SCDR.PCKx or PMC_SR.GCLKEN).

A Ready signal at '1' does not imply the clock is correctly established with the required frequency, hence the Ready status is not affected by the modification of the source or the dividing ratio of the clock. This means that:

1. modifying PMC_PCKx.CSS or PMC_PCKx.PRES does not make PMC_SR.PCKRDYx fall,
2. modifying PMC_PCR.GCLKCSS or PMC_PCR.GCLKDIV does not make PMC_SR.GCLKRDY fall.

Work Around

None

Affected Device Revisions

A0	A1	A1-D1G	A1-D2G	A1-D4G			
X	X	X	X	X			

10.4. Processor (CPU_CLK0) and main system bus clock (MCK0) source selection

When changing the fields CSS or CPCSS in the CPU Clock register (PMC_CPU_CKR) from any PLL source clocks (PLLxCKx) to Slow Clock source (SLOW_CLK), the clock switching circuitry first switches from the PLL source to MAINCK source, then to Slow Clock source.

There is no impact on the clock switching sequence or device behavior. This intermediate step can be observed when the main system bus clock is output on a PCK pin.

Work Around

None

Affected Device Revisions

A0	A1	A1-D1G	A1-D2G	A1-D4G			
X	X	X	X	X			

11. Parallel Input/Output Controller (PIO)

11.1. Open drain management limitation

PIOC does not allow open drain configuration (PIO_CFGRx.OPD=1) when a peripheral is selected (PIO_CFGRx.FUNC different from 0).

TWI/TWIHS peripherals are not affected.

Work Around

None

Affected Device Revisions

A0	A1	A1-D1G	A1-D2G	A1-D4G			
X	X	X	X	X			

12. Analog-to-Digital Converter (ADC) Controller

12.1. Spurious effect when zeros written to ADC_EOC_IDR

Writing 0s to ADC_EOC_IDR enables EOC interrupts instead of having no effect.

Work Around

None

Affected Device Revisions

A0	A1	A1-D1G	A1-D2G	A1-D4G			
X	X	X	X	X			

12.2. EOC interrupts not disabled when ones written to ADC_EOC_IDR

Writing 1s to ADC_EOC_IDR does not disable the EOC interrupts as it should.

Work Around

The channels can be disabled from ADC_CHDR if unused.

Affected Device Revisions

A0	A1	A1-D1G	A1-D2G	A1-D4G			
X	X	X	X	X			

12.3. ADC_IMR interrupts enabled when ones written to ADC_EOC_IDR

Writing 1s to ADC_EOC_IDR enables interrupts in ADC_IMR. If interrupts are pending in ADC_ISR, an interrupt is triggered to the interrupt controller.

Writing to ADC_EOC_IDR is not recommended.

Work Around

Immediately after writing 1s to ADC_EOC_IDR, write 1s to ADC_IDR to disable the unwanted interrupts (store previous value, disable all, reenable previous value).

Affected Device Revisions

A0	A1	A1-D1G	A1-D2G	A1-D4G			
X	X	X	X	X			

12.4. Temperature sensor still enabled when stopped without conversion

The temperature sensor remains active even when ADC_TEMPMR.TEMPON is set to 0.

Work Around

To stop the temperature sensor and save its power consumption, perform a conversion prior to writing ADC_TEMPMR.TEMPON=0.

Affected Device Revisions

A0	A1	A1-D1G	A1-D2G	A1-D4G			
X	X	X	X	X			

12.5. Temperature sensor spurious activation with CH30

Enabling ADC channel 30 enables the temperature sensor.

Work Around

To stop the temperature sensor and save its power consumption, perform a conversion prior to writing ADC_TEMPMR.TEMPON=0.

Affected Device Revisions

A0	A1	A1-D1G	A1-D2G	A1-D4G			
X	X	X	X	X			

12.6. Sleep mode ineffective

Even if Sleep mode is selected by setting ADC_MR.SLEEP, ADC does not enter Sleep mode after a conversion.

Work Around

Reset the ADC Controller with ADC_CR.SWRST.

Affected Device Revisions

A0	A1	A1-D1G	A1-D2G	A1-D4G			
X	X	X	X	X			

13. Image Sensor Controller (ISC)

13.1. Spurious DMA descriptor writing

Due to AXI transaction reordering, DMA descriptors writing may occur before the last image is written in the memory, even if the DONE flag is set. The user cannot read the DMA descriptors before the full image is written.

Work Around

1. Poll the bit ISC_INTSR.DDONE.
2. Perform an extra read of ISC_INTSR to enable DDR Controller writing.
3. Poll the bit UDDRC_PSTAT.WR_PORT_BUSY_3 with:
   ```
   while (DDRUMCTL_REGS->UDDRC_PSTAT & UDDRC_PSTAT_WR_PORT_BUSY_3(1));
   ```

Once done, the full image is written in the memory and the DMA descriptors can be written.

Affected Device Revisions

A0	A1	A1-D1G	A1-D2G	A1-D4G			
X	X	X	X	X			

13.2. Incoming pixels corrupted after overload

In case of overload, incoming pixels may be corrupted.

Work Around

Discard the last frame when the ISC_INTSR.DAOV bit is set.

Affected Device Revisions

A0	A1	A1-D1G	A1-D2G	A1-D4G			
X	X	X	X	X			

13.3. Frequency limitation

Timing issues may occur in harsh conditions (low voltage and/or high temperature).

Work Around

In harsh conditions, limit the operating frequency to f_{MCK3} < 184 MHz.

Affected Device Revisions

A0	A1	A1-D1G	A1-D2G	A1-D4G			
X	–	–	–	–			

14. Synchronous Serial Controller (SSC)

14.1. Inverted left/right channels

When the SSC is in Client mode, the TF signal is derived from the codec and not controlled by the SSC. The SSC transmits the data when detecting the falling edge on the TF signal after the SSC transmission is enabled. In some overflow cases, a left/right channel inversion may occur and may require SSC reinitializing.

Work Around

Use the SSC in Host mode so that TF is controlled by the SSC. If the SSC must be used in TF Client mode, start the SSC by writing TXEN and RXEN synchronously with the TXSYN flag rising.

Affected Device Revisions

A0	A1	A1-D1G	A1-D2G	A1-D4G			
X	X	X	X	X			

14.2. TD output delay

The TD output is delayed by two or three extra system clock cycles when SSC is configured with the following conditions:

- RCMR.START = Start on falling edge/Start on rising edge/Start on any edge
- RFMR.FSOS = None (input)
- TCMR.START = Receive Start

Work Around

None

Affected Device Revisions

A0	A1	A1-D1G	A1-D2G	A1-D4G			
X	X	X	X	X			

15. Sony/Philips Digital Interface Receiver (SPDIFRX)

15.1. SPDIFRX left/right inversion

The SPDIFRX can always provide the left channel sample in the first DMA buffer location when SPDIFRX_MR.SBMODE=1. If the Soft Reset command is applied, the right channel sample may be located in the first DMA buffer location.

Work Around

Perform actions in the following order:

1. Disable the DMA channel.
2. Apply the Soft Reset command.
3. Perform a dummy read in the Mode register (SPDIFRX_MR).
4. Read the SPDIFRX_ISR.RXRDY flag. If set to 1, read the SPDIF Receiver Holding register (SPDIFRX_RHR).

Affected Device Revisions

A0	A1	A1-D1G	A1-D2G	A1-D4G			
X	X	X	X	X			

16. Advanced Encryption Standard (AES)

16.1. SPLIP mode does not work with some header sizes

The Secure Protocol Layers Improved Performances (SPLIP) mode does not work when the ESP header is not an integer multiple of 4 words.

Work Around

When the ESP header is not an integer multiple of 4 words, disable SPLIP mode to stop AES from automatically uploading the encrypted payload into SHA, and use the central DMA to feed SHA with the encrypted payload.

Affected Device Revisions

A0	A1	A1-D1G	A1-D2G	A1-D4G			
X	X	X	X	X			

17. Security Module (SECUMOD)

17.1. Dynamic detection intrusion (PIOBU) alarm issue

The error counter fails to reinitialize after a dynamic detection intrusion (PIOBU) alarm.

Work Around

None

Affected Device Revisions

A0	A1	A1-D1G	A1-D2G	A1-D4G			
X	X	X	X	X			

17.2. Tamper timestamping polarity error

The tamper detection signal polarity is inverted, with the following consequences:

- Key erasing in TZAEB, AES and TDES if the respective Clear On Tamper features are enabled, with TZAESB_MR.TAMPCLR = 1, AES_MR.TAMPCLR = 1 and TDES_MR.TAMPCLR = 1.
- Scrambling key erasing in QSPI0 or QSPI1 if Clear On Tamper is enabled with QSPI0_MR.TAMPCLR = 1 or QSPI1_MR.TAMPCLR = 1.
- SHA locking if Tamper Lock is enabled with SHA_MR.TMPLCK = 1. SHA is locked until SHA_CR.UNLOCK is written to 1.

Work Around

Do not enable the following bits:

- TZAESB_MR.TAMPCLR
- AES_MR.TAMPCLR
- TDES_MR.TAMPCLR
- QSPI0_MR.TAMPCLR
- QSPI1_MR.TAMPCLR
- SHA_MR.TMPLCK

Affected Device Revisions

A0	A1	A1-D1G	A1-D2G	A1-D4G			
X	X	X	X	X			

17.3. SECUMOD registers BMPR and WKPR reading issue

The bits 18, 19, 20 and 21 (DET0, DET1, DET2, DET3) are functional in Write mode. When they are written, the corresponding PIOBU bits are enabled as wake-up sources but, when read, the registers show those bits shifted two steps to the right, to positions 16, 17 , 18 and 19, respectively.

Work Around

None

Affected Device Revisions

A0	A1	A1-D1G	A1-D2G	A1-D4G			
X	X	X	X	X			

18. Gigabit Ethernet MAC (GMAC)

18.1. GMAC0 not functional with multiple queues in 10/100 Half Duplex mode

When operating in 10/100 Half Duplex mode, GMAC0 does not work with multiple queues.

Work Around

Configure the controller for transmission over a single queue.

Affected Device Revisions

A0	A1	A1-D1G	A1-D2G	A1-D4G			
X	X	X	X	X			

18.2. Incorrect reading of Specific Address filter registers on GMAC0 and GMAC1

On GMAC0 and GMAC1, Specific Address filter register reading may be corrupted. This may impact in particular the device MAC address.

Affected registers are GMAC0_SABx, GMAC0_SATx, GMAC1_SABx, GMAC1_SATx.

Work Around

Use a shadow copy of the registers and fake the read operation using the shadow copy.

WARNING With the debugger, a value read in a register may be corrupted.

Affected Device Revisions

A0	A1	A1-D1G	A1-D2G	A1-D4G			
X	–	–	–	–			

18.3. Incorrect reading of Type 1 Screener registers on GMAC0 and GMAC1

On GMAC0 and GMAC1, Type 1 Screener register reading may be corrupted. This may impact QoS-based applications using DS/TC fields or TCP/UDP ports.

Affected registers are GMAC0_ST1RPQx and GMAC1_ST1RPQx.

Work Around

Use a shadow copy of the registers and fake the read operation using the shadow copy.

WARNING With the debugger, a value read in a register may be corrupted.

Affected Device Revisions

A0	A1	A1-D1G	A1-D2G	A1-D4G			
X	–	–	–	–			

18.4. Incorrect reading of Type 2 Screener registers on GMAC0 and GMAC1

On GMAC0 and GMAC1, Type 2 Screener register reading may be corrupted. This may impact QoS applications based on source/destination IP, source/destination TCP/UDP, Ethertype or VLAN.

Affected registers are GMAC0_ST2RPQx, GMAC0_ST2ERx, GMAC0_ST2CWxRy, GMAC1_ST2RPQx, GMAC1_ST2ERx, GMAC1_ST2CWxRy.

Work Around

Use a shadow copy of the registers and fake the read operation using the shadow copy.

WARNING With the debugger, a value read in a register may be corrupted.

Affected Device Revisions

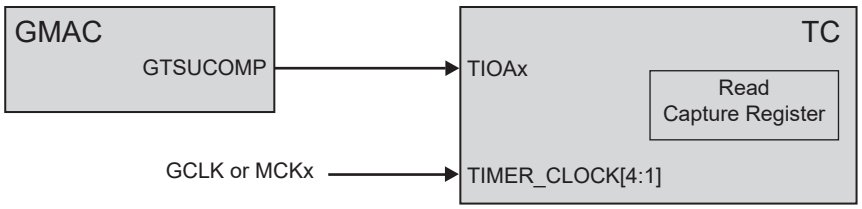
A0	A1	A1-D1G	A1-D2G	A1-D4G			
X	–	–	–	–			

18.5. GTSUCOMP ineffective connection to TC1

The trigger/capture input B of TC1.TIOB1 is driven internally by the GTSUCOMP signal of the Ethernet MAC (GMAC), but the feature does not work.

Work Around

Routing the GTSUCOMP signal to one timer counter input can be done at PCB level. GTSUCOMP is multiplexed on the PB2 line, therefore PB2 can be externally looped back to one of the timer counter inputs, as shown in the following figure.



Affected Device Revisions

A0	A1	A1-D1G	A1-D2G	A1-D4G			
X	X	X	X	X			

19. Flexible Serial Communication Controller (FLEXCOM)

19.1. Write Protection ineffective on FLEXCOM8 to FLEXCOM11

On FLEXCOM8, FLEXCOM9, FLEXCOM10 and FLEXCOM11, TWIx_WPMR and TWIx_WPSR reading may return corrupted values.

Work Around

Do not use the Write Protection feature on FLEXCOM8, FLEXCOM9, FLEXCOM10 and FLEXCOM11.

Affected Device Revisions

A0	A1	A1-D1G	A1-D2G	A1-D4G			
X	X	X	X	X			

20. Secure Digital MultiMedia Card Controller (SDMMC)

20.1. SDMMC failure when changing speed mode or performing ALL soft reset on the fly

Changing speed mode or performing an ALL soft reset while SDCK is active may lead to block the SDMMC.

Work Around

Stop SDCLK before changing speed mode or performing an ALL soft reset, then re-enable SDCLK by writing SDMMC_CCR.SDCLKEN to 0 before writing SDMMC_MC1R/SDMMC_HC1R/SDMMC_HC2R.

Example:

```
//FIX : stop SDCLK
pSDMMC->SDMMC_CCR = pSDMMC->SDMMC_CCR & ~SDMMC_CCR_SDCLKEN;
switch (speed_mode) {
case DS : pSDMMC->SDMMC_HC1R = pSDMMC->SDMMC_HC1R & ~SDMMC_HC1R_HSEN;
break;
case HS : pSDMMC->SDMMC_HC1R = pSDMMC->SDMMC_HC1R | SDMMC_HC1R_HSEN;
break;
case SDR12 : pSDMMC->SDMMC_HC2R = (pSDMMC->SDMMC_HC2R & ~SDMMC_HC2R_UHSMS_Msk) |
SDMMC_HC2R_UHSMS_SDR12;
break;
case SDR25 : pSDMMC->SDMMC_HC2R = (pSDMMC->SDMMC_HC2R & ~SDMMC_HC2R_UHSMS_Msk) |
SDMMC_HC2R_UHSMS_SDR25;
break;
case SDR50 : pSDMMC->SDMMC_HC2R = (pSDMMC->SDMMC_HC2R & ~SDMMC_HC2R_UHSMS_Msk) |
SDMMC_HC2R_UHSMS_SDR50;
break;
case SDR104 : pSDMMC->SDMMC_HC2R = (pSDMMC->SDMMC_HC2R & ~SDMMC_HC2R_UHSMS_Msk) |
SDMMC_HC2R_UHSMS_SDR104;
break;
case DDR50 : pSDMMC->SDMMC_HC2R = (pSDMMC->SDMMC_HC2R & ~SDMMC_HC2R_UHSMS_Msk) |
SDMMC_HC2R_UHSMS_DDR50;
break;
}
//FIX : re-start SDCLK
pSDMMC->SDMMC_CCR = pSDMMC->SDMMC_CCR | SDMMC_CCR_SDCLKEN;
```

Affected Device Revisions

A0	A1	A1-D1G	A1-D2G	A1-D4G			
X	X	X	X	X			

20.2. SDR104, HS200, HS400 modes are not functional

Using mode SDR104, HS200 or HS400 may lead to tuning issues, data read errors or clock switching failures.

Work Around

None

Affected Device Revisions

A0	A1	A1-D1G	A1-D2G	A1-D4G			
X	X	X	X	X			

20.3. GCLK cannot be stopped

If a speed mode other than DefaultSpeed or SDR12 is used, GCLK cannot be stopped, leading to unpredictable behavior.

Work Around

Perform an ALL soft reset before any operation to ensure the internal clock DLL can be stopped properly.

Affected Device Revisions

A0	A1	A1-D1G	A1-D2G	A1-D4G			
X	X	X	X	X			

20.4. SDHC blocked after switch from high-speed mode

If the current speed mode is Default Speed or SDR12, the SDCLK frequency must be higher than 25 MHz before switching to another mode, otherwise SDHC is blocked.

Work Around

Set the SDCLK frequency to a value higher than 25 MHz before switching to another mode.

Affected Device Revisions

A0	A1	A1-D1G	A1-D2G	A1-D4G			
X	X	X	X	X			

21. Controller Area Network (MCAN)

21.1. Edge filtering causes mis-synchronization when falling edge at Rx input pin coincides with end of integration phase

Attention: This erratum is not relevant for CAN 2.0.

When edge filtering is enabled (MCAN_CCCR.EFBI = ’1’) and when the end of the integration phase coincides with a falling edge at the Rx input pin, it may happen that the MCAN synchronizes itself wrongly and does not correctly receive the first bit of the frame. In this case the CRC will detect that the first bit was received incorrectly; it will rate the received FD frame as faulty and an error frame will be sent.

The issue only occurs when there is a falling edge at the Rx input pin (CANRX) within the last time quantum (tq) before the end of the integration phase. The last time quantum of the integration phase is at the sample point of the 11th recessive bit of the integration phase. When the edge filtering is enabled, the bit timing logic of the MCAN sees the Rx input signal delayed by the edge filtering. When the integration phase ends, the edge filtering is automatically disabled. This affects the reset of the FD CRC registers at the beginning of the frame. The Classical CRC registers are not affected, so this issue does not affect the reception of Classical frames.

In CAN communication, the MCAN may enter integrating state (either by resetting MCAN_CCCR.INIT or by protocol exception event) while a frame is active on the bus. In this case the 11 recessive bits are counted between the Acknowledge bit and the following start of frame. All nodes have synchronized at the beginning of the dominant Acknowledge bit. This means that the edge of the following Start-of-Frame bit cannot fall on the sample point, so the issue does not occur. The issue occurs only when the MCAN is, by local errors, mis-synchronized with regard to the other nodes, or not synchronized at all.

Glitch filtering as specified in ISO 11898-1:2015 is fully functional.

Edge filtering was introduced for applications where the data bit time is at least two tq (of the nominal bit time) long. In that case, edge filtering requires at least two consecutive dominant time quanta before the counter counting the 11 recessive bits for idle detection is restarted. This means edge filtering covers the theoretical case of occasional 1-tq-long dominant spikes on the CAN bus that would delay idle detection. Repeated dominant spikes on the CAN bus would disturb all CAN communication, so the filtering to speed up idle detection would not help network performance.

When this rare event occurs, the MCAN sends an error frame and the sender of the affected frame retransmits the frame. When the retransmitted frame is received, the MCAN has left the integration phase and the frame will be received correctly. Edge filtering is only applied during integration phase; it is never used during normal operation. As the integration phase is very short with respect to “active communication time”, the impact on total error frame rate is negligible. The issue has no impact on data integrity.

The MCAN enters integration phase under the following conditions:

- when MCAN_CCCR.INIT is set to ’0’ after start-up
- after a protocol exception event (only when MCAN_CCCR.PXHD = ’0’)

Work Around

Disable edge filtering or wait on retransmission in case this rare event happens.

Affected Device Revisions

A0	A1	A1-D1G	A1-D2G	A1-D4G			
X	X	X	X	X			

21.2. Configuration of MCAN_NBTP.NTSEG2 = ’0’ not allowed

Attention: This erratum is applicable for CAN 2.0.

When MCAN_NBTP.NTSEG2 is configured to zero (Phase_Seg2(N) = 1), and when there is a pending transmission request, a dominant third bit of Intermission may cause the MCAN to wrongly transmit the first identifier bit dominant instead of recessive, even if this bit was configured as ’1’ in the MCAN’s Tx Buffer Element.

A phase buffer segment 2 of length ’1’ (Phase_Seg2(N) = 1) is not sufficient to switch to the first identifier bit after the sample point in Intermission where the dominant bit was detected.

The CAN protocol according to ISO 11898-1 defines that a dominant third bit of Intermission causes a pending transmission to be started immediately. The received dominant bit is handled as if the MCAN has transmitted a Start-of-Frame (SoF) bit.

The ISO 11898-1 specifies the minimum configuration range for Phase_Seg2(N) to be 2..8 tq. Therefore excluding a Phase_Seg2(N) of ’1’ will not affect MCAN conformance.

Work Around

Use the range 1..127 for MCAN_NBTP.NTSEG2 instead of 0..127.

Affected Device Revisions

A0	A1	A1-D1G	A1-D2G	A1-D4G			
X	X	X	X	X			

21.3. Retransmission in DAR mode due to lost arbitration at the first two identifier bits

Attention: This erratum is applicable for CAN 2.0.

When the MCAN is configured in DAR mode (MCAN_CCCR.DAR = ’1’) the Automatic Retransmission for transmitted messages that have been disturbed by an error or have lost arbitration is disabled. When the transmission attempt is not successful, the Tx Buffer’s transmission request bit (MCAN_TXBRP.TRPxx) shall be cleared and its Cancellation Finished bit (MCAN_TXBCF.CFxx) shall be set.

When the transmitted message loses arbitration at one of the first two identifier bits, it may happen that instead of the bits of the actually transmitted Tx Buffer, the MCAN_TXBRP.TRPxx and MCAN_TXBCF.CFxx bits of the previously started Tx Buffer (or Tx Buffer 0 if there is no previous transmission attempt) are written (MCAN_TXBRP.TRPxx = ’0’, MCAN_TXBCF.CFxx = ’1’).

If in this case the MCAN_TXBRP.TRPxx bit of the Tx Buffer that lost arbitration at the first two identifier bits has not been cleared, retransmission is attempted.

When the MCAN loses arbitration again at the immediately following retransmission, then actually and previously transmitted Tx Buffers are the same and this Tx Buffer’s MCAN_TXBRP.TRPxx bit is cleared and its MCAN_TXBCF.CFxx bit is set.

Work Around
None

Affected Device Revisions

A0	A1	A1-D1G	A1-D2G	A1-D4G			
X	X	X	X	X			

21.4. Tx FIFO message sequence inversion

Attention: This erratum is applicable for CAN 2.0.

Assume the case that there are two Tx FIFO messages in the output pipeline of the Tx Message Handler. Transmission of Tx FIFO message 1 is started:

- Position 1: Tx FIFO message 1 (transmission ongoing)
- Position 2: Tx FIFO message 2
- Position 3: --

Now a non-Tx FIFO message with a higher CAN priority is requested. Due to its priority it will be inserted into the output pipeline. The TxMH performs so called "message scans" to keep the output pipeline up to date with the highest priority messages from the Message RAM. After the following two message scans, the output pipeline has the following content:

- Position 1: Tx FIFO message 1 (transmission ongoing)
- Position 2: non-Tx FIFO message with higher CAN priority
- Position 3: Tx FIFO message 2

If the transmission of Tx FIFO message 1 is not successful (lost arbitration or CAN bus error) it is pushed from the output pipeline by the non-Tx FIFO message with higher CAN priority. The following scan re-inserts Tx FIFO message 1 into the output pipeline at position 3:

- Position 1: non-Tx FIFO message with higher CAN priority (transmission ongoing)
- Position 2: Tx FIFO message 2
- Position 3: Tx FIFO message 1

Now Tx FIFO message 2 is in the output pipeline in front of Tx FIFO message 1 and they are transmitted in that order, resulting in a message sequence inversion.

Work Around
1. First Work Around

Use two dedicated Tx Buffers, e.g. use Tx Buffers 4 and 5 instead of the Tx FIFO. The pseudo-code below replaces the function that fills the Tx FIFO.

Write message to Tx Buffer 4.

Transmit loop:

- Request Tx Buffer 4 - write MCAN_TXBAR.A4
- Write message to Tx Buffer 5
- Wait until transmission of Tx Buffer 4 completed - MCAN_IR.TC, read MCAN_TXBTO.TO4
- Request Tx Buffer 5 - write MCAN_TXBAR.A5
- Write message to Tx Buffer 4
- Wait until transmission of Tx Buffer 5 is completed - MCAN_IR.TC, read MCAN_TXBTO.TO5

2. Second Work Around

Make sure that only one Tx FIFO element is pending for transmission at any time. The Tx FIFO elements may be filled at any time with messages to be transmitted, but their transmission requests are handled separately. Each time a Tx FIFO transmission has completed and the Tx FIFO gets empty (MCAN_IR.TFE = '1'), the next Tx FIFO element is requested.

3. Third Work Around

Use only a Tx FIFO. Send the message with the higher priority also from Tx FIFO.

One drawback is that the higher priority message has to wait until the preceding messages in the Tx FIFO have been sent.

Affected Device Revisions

A0	A1	A1-D1G	A1-D2G	A1-D4G			
X	X	X	X	X			

21.5. Unexpected High Priority Message (HPM) interrupt

Attention: This erratum is applicable for CAN 2.0.

This issue occurs in two configurations:

Configuration A:

- At least one Standard Message ID Filter Element is configured with Priority flag set (S0.SFEC = "100"/"101"/"110").
- No Extended Message ID Filter Element is configured.
- Non-matching extended frames are accepted (MCAN_GFC.ANFE = "00"/"01").

The HPM Interrupt flag MCAN_IR.HPM is set erroneously on reception of a non-high-priority extended message under the following conditions:

1. A standard HPM frame is received, and accepted by a filter with Priority flag set. Then, Interrupt flag MCAN_IR.HPM is set as expected.
2. Next, an extended frame is received and accepted due to the MCAN_GFC.ANFE configuration. Then, Interrupt flag MCAN_IR.HPM is set erroneously.

Configuration B:

- At least one Extended Message ID Filter Element is configured with Priority flag set (F0.EFEC = "100"/"101"/"110").
- No Standard Message ID Filter Element is configured.
- Non-matching standard frames are accepted (MCAN_GFC.ANFS = "00"/"01").

The HPM Interrupt flag MCAN_IR.HPM is set erroneously on reception of a non-high-priority standard message under the following conditions:

1. An extended HPM frame is received, and accepted by a filter with Priority flag set. Then, Interrupt flag MCAN_IR.HPM is set as expected.
2. Next, a standard frame is received and accepted due to the MCAN_GFC.ANFS configuration. Then, Interrupt flag MCAN_IR.HPM is set erroneously.

Work Around

Configuration A:

Set up an Extended Message ID Filter Element with the following configuration:

- F0.EFEC = "001"/"010" - select Rx FIFO for storage of extended frames
- F0.EFID1 = any value - value not relevant as all ID bits are masked out by F1.EFID2
- F1.EFT = "10" - classic filter, F0.EFID1 = filter, F1.EFID2 = mask
- F1.EFID2 = zero - all bits of the received extended ID are masked out

Now, all extended frames are stored in Rx FIFO 0 respectively Rx FIFO 1 depending on the configuration of F0.EFEC.

Configuration B:

Set up a Standard Message ID Filter Element with the following configuration:

- S0.SFEC = "001"/"010" - select Rx FIFO for storage of standard frames
- S0.SFID1 = any value - value not relevant as all ID bits are masked out by S0.SFID2
- S0.SFT = "10" - classic filter, S0.SFID1 = filter, S0.SFID2 = mask
- S0.SFID2 = zero - all bits of the received standard ID are masked out

Now, all standard frames are stored in Rx FIFO 0 respectively Rx FIFO 1 depending on the configuration of S0.SFEC.

Affected Device Revisions

A0	A1	A1-D1G	A1-D2G	A1-D4G			
X	X	X	X	X			

21.6. Issue message transmitted with wrong arbitration and control fields

Attention: This erratum is applicable for CAN 2.0.

When the following conditions are met, a message with wrong ID, format, and DLC is transmitted:

- M_CAN is in state "Receiver" (PSR.ACT = "10") and there is no pending transmission.
- A new transmission is requested before the third Intermission bit is reached.
- The CAN bus is sampled dominant at the third Intermission bit which is treated as SoF (see ISO11898-1:2015 Section 10.4.2.2).

Then, it can happen that:

- The Shift register is not loaded with the ID, format and DLC of the requested message.
- The MCAN starts arbitration with wrong ID, format, and DLC on the next bit.
- If the ID wins arbitration, a CAN message with valid CRC is transmitted.
- If this message is acknowledged, the ID stored in the Tx Event FIFO is the ID of the requested Tx message, and not the ID of the message transmitted on the CAN bus, and no error is detected by the transmitting MCAN.

Work Around

Request a new transmission only if another transmission is already pending or when the MCAN is not in "Receiver" state (when PSR.ACT ≠ "10").

To avoid activating the transmission request in the critical time window between the sample points of the second and third Intermission bits, the application software can evaluate the Rx Interrupt flags IR.DRX, IR.RF0N and IR.RF1N, which are set at the last EoF bit when a received and accepted message becomes valid.

The last EoF bit is followed by three Intermission bits. Therefore, the critical time window has safely terminated three bit times after the Rx interrupt. Now a transmission can be requested by writing to TXBAR.

After the interrupt, the application has to take care that the transmission request for the CAN Protocol Controller is activated before the critical window of the following reception is reached.

A checksum covering the arbitration and control fields can be added to the data field of the message to be transmitted, to detect frames transmitted with wrong arbitration and control fields.

Affected Device Revisions

A0	A1	A1-D1G	A1-D2G	A1-D4G			
X	X	X	X	X			

21.7. Debug message handling state machine not reset to Idle when CCCR.INIT is set

When the host sets the MCAN_CCCR.INIT bit through the MCAN_CCCRn register, or when the CAN enters Bus Off state, the debug message handling state machine stays in its current state instead of resetting to Idle state. Setting MCAN_CCCR.CCE does not change MCAN_RXF1S.DMS.

Work Around

If the debug message handling state machine stopped while MCAN_RXF1S.DMS="01" or MCAN_RXF1S.DMS="10", it can be reset to Idle state by hardware reset or by reception of debug messages after MCAN_CCCR.INIT is reset to zero.

Affected Device Revisions

A0	A1	A1-D1G	A1-D2G	A1-D4G			
X	X	X	X	X			

21.8. Message order inversion when transmitting from dedicated Tx buffers configured with same message ID

When several Tx buffers are configured with the same message ID, transmission of these Tx buffers is requested sequentially with a delay between the individual Tx requests. They are transmitted in ascending order of their numbers, so the Tx buffer with the lowest number and pending Tx request is transmitted first.

However, depending on the delay between the individual Tx requests, it can happen that the lowest Tx buffer number is not transmitted first and that the message order is inverted.

Work Around

First write the group of Tx messages having the same message ID to the message RAM, and then request transmission of all these messages concurrently by a single write access to MCAN_TXBAR.

Affected Device Revisions

A0	A1	A1-D1G	A1-D2G	A1-D4G			
X	X	X	X	X			

21.9. Frame transmitted despite confirmed transmit cancellation for CAN-FD messages with more than 8 data bytes

Attention: This erratum is not relevant for CAN 2.0.

In case the transmission of Tx Buffer nn was not successful and is restarted immediately afterwards by automatic retransmission, and the software requests a Tx cancellation for this Tx Buffer by

setting the cancellation request bit MCAN_TXBCR.CRnn during transmission of the first 4 identifier bits, a successful cancellation is incorrectly signalled by setting MCAN_TXBCF.CFnn = '1' and by clearing MCAN_TXBRP.TRPnn. In addition, the respective Transmission Occurred bit remains zero (MCAN_TXBTO.TOnn = '0'), incorrectly indicating that the frame was not transmitted on the bus.

Other than signalled by MCAN_TXBCF.CFnn and MCAN_TXBTO.TOnn, the transmission continues until the complete frame has been sent on the CAN bus. If the transmission is successful, MCAN_TXBTO.TOnn will be set.

If in this case new data is written to Tx Buffer nn while the transmission is still ongoing, a frame with inconsistent data may appear on the bus.

This problem is limited to the case of transmit cancellation of CAN-FD messages with more than 8 data bytes while automatic retransmission is enabled (MCAN_CCCR.DAR = '0').

Transmit cancellation of Classical CAN messages and CAN-FD messages with up to 8 data bytes is not affected.

CAN 2.0 operation is not impacted.

Work Around

Do not use transmit cancellation for CAN-FD messages with more than 8 data bytes.

Alternatively, wait for the duration of the expected transmission time of the cancelled Tx Buffer before writing new data to that Tx Buffer.

Affected Device Revisions

A0	A1	A1-D1G	A1-D2G	A1-D4G			
X	X	X	X	X			

21.10. MCAN_TSU_TSCFG reset after read

When a write is issued to configure the TSU Timestamp Configuration register (MCAN_TSU_TSCFG), any attempt to read it resets the register content.

Work Around

Save the content of MCAN_TSU_TSCFG in memory to access the value without reading the register.

Affected Device Revisions

A0	A1	A1-D1G	A1-D2G	A1-D4G			
X	X	X	X	X			

21.11. MCAN_TSU_TSS1 not reset after a MCAN_TSU_TSx read

TSL[15:0] and TSN[15:0] in MCAN TSU Timestamp Status 1 (MCAN_TSU_TSS1) are not reset after reading a MCAN TSU Timestamp Status (MCAN_TSU_TSx) register.

Work Around

Proceed as follows:

1. Read MCAN_TSU_TSx.
2. For each bit set, read the corresponding MCAN_TSU_TSx and save the values.
3. Write the same MCAN_TSU_TSx register with value '0' to reset the corresponding bit in MCAN_TSU_TSS1.

Affected Device Revisions

A0	A1	A1-D1G	A1-D2G	A1-D4G			
X	X	X	X	X			

21.12. MCAN_TSU_ATB read resets the timebase value

Each access to the Actual Timebase register (MCAN_TSU_ATB) resets the Timebase Prescaler MCAN_TSU_TSCFG.TBPRE.

Work Around

None

Affected Device Revisions

A0	A1	A1-D1G	A1-D2G	A1-D4G			
X	X	X	X	X			

22. Timer Counter (TC)

22.1. TC0 Channel 2 registers incorrect reading

Reading the following registers (TC0 Channel 2 registers) may return corrupted values: TC0_CCR2, TC0_CMR2, TC0_SMMR2, TC0_RAB2, TC0_CV2, TC0_RA2, TC0_RB2, TC0_RC2, TC0_SR2, TC0_IER2.

Work Around

Do not use TC0 Channel 2.

Affected Device Revisions

A0	A1	A1-D1G	A1-D2G	A1-D4G			
X	–	–	–	–			

23. USB Device High Speed Port (UDPHS)

23.1. EHCI spurious stop when Suspend mode occurs on port A

If port A enters Suspend mode (via the USB device controller or the USB host controller), the EHCI clock stops, which can block the EHCI if it uses the other ports (B/C). This occurs after about 2 µs (≈ 120 clock cycles).

Work Around

Set the SFR_EHCIOHCI.PHYCLK bit so that the EHCI clock remains activated in Suspend mode, with no power saving.

Affected Device Revisions

A0	A1	A1-D1G	A1-D2G	A1-D4G			
X	X	X	X	X			

24. Low Power Modes

24.1. ULP2 mode does not work

Operation of the device at ULP2 wake-up is not predictable. Do not use this low-power mode.

Work Around

None

Affected Device Revisions

A0	A1	A1-D1G	A1-D2G	A1-D4G			
X	X	X	X	X			

25. Data Sheet Clarifications

There are no known data sheet clarifications as of this publication date.

26. Revision History

26.1. DS80001016F - 06/2025

Extended scope to the SAMA7G54D4G(T)-I/4UB device
Removed SAMA7G54(T)-V/7EW from scope
Table 1: corrected ordering codes from SAMA7G54DxG(T)-I/4TB to SAMA7G54DxG(T)-I/4UB

26.2. DS80001016E - 03/2025

- Throughout:
  - Extended scope to the SAMA7G54(T)-V/7EW device
  - Changed terminology from “silicon revision” to “device revision”
- Added Boot failure on e.MMC memories

26.3. DS80001016D - 11/2024

Added:
- MCAN_TSU_TSCFG reset after read
- MCAN_TSU_TSS1 not reset after a MCAN_TSU_TSx read
- MCAN_TSU_ATB read resets the timebase value
- ULP2 mode does not work

Updated MCKRDY flag error, GTSUCOMP ineffective connection to TC1

26.4. DS80001016C - 12/2023

Added references to SAMA7G5 Series SiP devices.
Added:
- CHIPID_EXID may report a wrong value
- Delay to first establish PCK
- PCK and GCLK Ready status issue
- Processor (CPU_CLK0) and main system bus clock (MCK0) source selection
- SPLIP mode does not work with some header sizes
- SECUMOD registers BMPR and WKPR reading issue

Updated MCKRDY flag error
Removed “Temperature sensor wrong parameter value” data sheet clarification

26.5. DS80001016B - 08/2022

Added:
- Temperature sensor still enabled when stopped without conversion
- Temperature sensor spurious activation with CH30
- Sleep mode ineffective
- Frame transmitted despite confirmed transmit cancellation for CAN-FD messages with more than 8 data bytes
- “Temperature sensor wrong parameter value” in Data Sheet Clarifications

Updated Spurious DMA descriptor writing
Removed ACC erratum “WPVS bit incorrect behavior”

26.6. DS80001016A - 03/2022

First issue.

Microchip Information

Trademarks

The “Microchip” name and logo, the “M” logo, and other names, logos, and brands are registered and unregistered trademarks of Microchip Technology Incorporated or its affiliates and/or subsidiaries in the United States and/or other countries (“Microchip Trademarks”). Information regarding Microchip Trademarks can be found at https://www.microchip.com/en-us/about/legal-information/microchip-trademarks.

ISBN: 979-8-3371-1362-3

Legal Notice

This publication and the information herein may be used only with Microchip products, including to design, test, and integrate Microchip products with your application. Use of this information in any other manner violates these terms. Information regarding device applications is provided only for your convenience and may be superseded by updates. It is your responsibility to ensure that your application meets with your specifications. Contact your local Microchip sales office for additional support or, obtain additional support at www.microchip.com/en-us/support/design-help/client-support-services.

THIS INFORMATION IS PROVIDED BY MICROCHIP “AS IS”. MICROCHIP MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND WHETHER EXPRESS OR IMPLIED, WRITTEN OR ORAL, STATUTORY OR OTHERWISE, RELATED TO THE INFORMATION INCLUDING BUT NOT LIMITED TO ANY IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY, AND FITNESS FOR A PARTICULAR PURPOSE, OR WARRANTIES RELATED TO ITS CONDITION, QUALITY, OR PERFORMANCE.

IN NO EVENT WILL MICROCHIP BE LIABLE FOR ANY INDIRECT, SPECIAL, PUNITIVE, INCIDENTAL, OR CONSEQUENTIAL LOSS, DAMAGE, COST, OR EXPENSE OF ANY KIND WHATSOEVER RELATED TO THE INFORMATION OR ITS USE, HOWEVER CAUSED, EVEN IF MICROCHIP HAS BEEN ADVISED OF THE POSSIBILITY OR THE DAMAGES ARE FORESEEABLE. TO THE FULLEST EXTENT ALLOWED BY LAW, MICROCHIP’S TOTAL LIABILITY ON ALL CLAIMS IN ANY WAY RELATED TO THE INFORMATION OR ITS USE WILL NOT EXCEED THE AMOUNT OF FEES, IF ANY, THAT YOU HAVE PAID DIRECTLY TO MICROCHIP FOR THE INFORMATION.

Use of Microchip devices in life support and/or safety applications is entirely at the buyer’s risk, and the buyer agrees to defend, indemnify and hold harmless Microchip from any and all damages, claims, suits, or expenses resulting from such use. No licenses are conveyed, implicitly or otherwise, under any Microchip intellectual property rights unless otherwise stated.

Microchip Devices Code Protection Feature

Note the following details of the code protection feature on Microchip products:

- Microchip products meet the specifications contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is secure when used in the intended manner, within operating specifications, and under normal conditions.
- Microchip values and aggressively protects its intellectual property rights. Attempts to breach the code protection features of Microchip products are strictly prohibited and may violate the Digital Millennium Copyright Act.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of its code. Code protection does not mean that we are guaranteeing the product is “unbreakable”. Code protection is constantly evolving. Microchip is committed to continuously improving the code protection features of our products.

Product Page Links

SAMA7G54, SAMA7G54D1G, SAMA7G54D2G, SAMA7G54D4G