STM32MP23xx/25xx device errata

Applicability

This document applies to the part numbers of STM32MP23xx/25xx devices and the device variants as stated in this page.

It gives a summary and a description of the device errata, with respect to the device datasheet and reference manual RM0457.

Deviation of the real device behavior from the intended device behavior is considered to be a device limitation. Deviation of the description in the reference manual or the datasheet from the intended device behavior is considered to be a documentation erratum. The term *"errata"* applies both to limitations and documentation errata.

Table 1. Device summary

Reference	Part numbers
STM32MP231x	STM32MP231A, STM32MP231C, STM32MP231D, STM32MP231F
STM32MP233x	STM32MP233A, STM32MP233C, STM32MP233D, STM32MP233F
STM32MP235x	STM32MP235A, STM32MP235C, STM32MP235D, STM32MP235F
STM32MP251x	STM32MP251A, STM32MP251C, STM32MP251D, STM32MP251F
STM32MP253x	STM32MP253A, STM32MP253C, STM32MP253D, STM32MP253F
STM32MP255x	STM32MP255A, STM32MP255C, STM32MP255D, STM32MP255F
STM32MP257x	STM32MP257A, STM32MP257C, STM32MP257D, STM32MP257F

Table 2. Device variants

Reference	Silicon revision codes	
	Device marking(1)	REV_ID(2)
STM32MP23xx	Y	0b010001
STM32MP25xx	B	0b000000
	Y	0b010001

1. *Refer to the device datasheet for how to identify this code on different types of package.*
2. *REV_ID[5:0] bitfield of BSEC_FVR102 fuse.*

Note: BSEC_FVR102 register is not automatically shadowed with OTP content, so a fuse read sequence must be issued to get the register updated once (clear after reading). Refer to product reference manual - BSEC section "Operations on fuses".

1 Summary of device errata

The following table gives a quick reference to the STM32MP23xx/25xx device limitations and their status:

A = limitation present, workaround available

N = limitation present, no workaround available

P = limitation present, partial workaround available

"-" = limitation absent

Applicability of a workaround may depend on specific conditions of target application. Adoption of a workaround may cause restrictions to target application. Workaround for a limitation is deemed partial if it only reduces the rate of occurrence and/or consequences of the limitation, or if it is fully effective for only a subset of instances on the device or in only a subset of operating modes, of the function concerned.

Table 3. Summary of device limitations

Function	Section	Limitation	Status	
			Rev. B	Rev. Y
Arm Cortex-A35 core	2.1.1	Speculative AT instruction using out-of-context translation regime could cause subsequent request to generate an incorrect translation	A	A
	2.1.2	Some AT instructions executed from EL3 might incorrectly report a domain fault	A	A
	2.1.3	ATB flush response may be delayed	A	A
	2.1.4	PMU counter might be inaccurate when monitoring BUS_ACCESS and BUS_ACCESS_ST	A	A
	2.1.5	Mismatch between EDPRSR.SR and EDPRSR.R	A	A
	2.1.6	ATS12NSOPR instruction might incorrectly translate when the HCR.TGE bit is set	A	A
Arm Cortex-M33 core	2.2.1	Access permission faults are prioritized over unaligned Device memory faults	N	N
System	2.3.1	ADF1/MDF1 kernel clock not provided in autonomous mode	A	A
	2.3.2	Debug port unavailable during backup domain software reset VSWRST	A	A
	2.3.3	Incorrect JEDEC ID on AP2 (Cortex-M0+)	A	A
	2.3.4	Wrong SYSCFG_IPIDR reset value	A	A
	2.3.5	Compartment filtering of PWR_CR11 and PWR_CR12 registers is not functional	A	A
	2.3.6	STGEN is reset when D1 domain is in DStandby low power mode	N	N
	2.3.7	Unwanted IP reset when D1 domain exit from DStandby	A	-
	2.3.8	LPLV-STOP2 exit failed if D3 using LSE or LSI clock	A	-
	2.3.9	GPU reset randomly not released	A	A
	2.3.11	Boot fails after wakeup from STANDBY when booting on SNOR, SNAND and HYPERFLASH	N	-
	2.3.12	Boot ROM hangs when system reset is applied while D1 domain is in DStandby state	P	-
	2.3.13	Boot ROM writes in SYSRAM during LPLV-Stop2 wakeup	A	A
	2.3.14	Instruction fetch access to PWR register lead to unwanted write	A	A
	2.3.15	CPU2 (Cortex-M33) does not support debug in non-secure only	A	A
	2.3.16	LSE function can be impacted if there is negative current injection in GPIOs	A	A
	2.3.17	ETHx kernel clock is gated if ETHxMACEN register bit is not set	A	A

Function	Section	Limitation	Status	
			Rev. B	Rev. Y
System	2.3.18	RISAF wrong SIDR value	N	N
	2.3.19	STOP and Standby entry failed when DDR is in shared mode	A	A
	2.3.20	Cortex-A35 not restarted after system reset in TDCID Cortex-M33 configuration	A	A
	2.3.21	AHB RISAB3/4/5 illegal access due to ghost CID0 detection	A	A
	2.3.22	ETM timestamp is exported with zero value	A	A
FMC	2.4.1	NOR flash memory/PSRAM incorrect bus turnaround timing	A	A
	2.4.2	Incorrect FMC_CLK clock period when CLKDIV value is changed on-the-fly in Continuous clock mode	A	A
OCTOSPI	2.5.1	Memory-mapped write error response when DQS output is disabled	P	P
	2.5.2	Deadlock can occur under certain conditions	A	A
	2.5.3	Memory wrap instruction not enabled when DQS is disabled	N	N
	2.5.4	Deadlock or write-data corruption after spurious write to a misaligned address in OCTOSPI_AR register	N	N
	2.5.5	Deadlock on consecutive out-of-range memory-mapped write operations	P	P
	2.5.6	Indirect write mode limited to 256 Mbytes	N	N
	2.5.7	Read-modify-write operation does not clear the MSEL bit	A	A
	2.5.9	Setting the ABORT bit does not generate an error on the AHB bus for undefined-length incremental burst transfers	P	P
	2.5.10	Read data corruption when a wrap transaction is followed by a linear read to the same MSB address	N	N
	2.5.11	Transactions are limited to 8 Mbytes in OctaRAM™ memories	N	N
	2.5.12	Variable latency is not supported when a refresh collision occurs during a write access to some OctaRAM™ memories	P	P
OCTOSPIM	2.6.1	Certain quad memories may be reset during arbitration while in single-SPI mode	A	A
SDMMC	2.7.1	Command response and receive data end bits not checked	N	N
ADC	2.8.1	JEOS may be set before the last injected data are available in ADC_JDRx	A	A
	2.8.2	In combined regular simultaneous plus alternate trigger mode, stopping injected conversion may delay regular conversion	A	A
	2.8.3	When the ADC clock is derived from the AHB clock, the injected conversion latency is not respected if the injected trigger coincides with the stopping of the regular conversion	A	A
LTDC	2.9.1	Ongoing AXI write never completes if disabling LTDC	A	A
	2.9.2	Layers cannot read YUV420 multibuffer data	N	-
	2.9.3	Rotation for wide landscape display can causes artifacts	N	N
	2.9.4	Layer 1 cannot read YUV420 multibuffer data	-	N
VENC	2.10.1	VENC hardware self-reset after internal timeout is unstable	A	A
VDEC	2.11.1	VDEC hardware self-reset after internal timeout is unstable	A	A
LPTIM	2.12.1	Device may remain stuck in LPTIM interrupt when entering Stop mode	A	A
	2.12.2	ARRM and CMPM flags are not set when APB clock is slower than kernel clock	A	A
	2.12.3	Interrupt status flag is cleared by hardware upon writing its corresponding bit in LPTIM_DIER register	N	N

Function	Section	Limitation	Status	
			Rev. B	Rev. Y
RTC and TAMP	2.13.1	Alarm flag may be repeatedly set when the core is stopped in debug	N	N
I2C	2.14.1	Wrong data sampling when data setup time ($t_{SU;DAT}$) is shorter than one I2C kernel clock period	P	P
	2.14.2	Spurious bus error detection in master mode	A	A
I3C	2.15.1	I3C controller: unexpected read data bytes during a legacy I^2C read	A	A
	2.15.2	I3C controller: SCL clock is not stalled during address ACK/NACK phase following a frame start, when enabled through I3C_TIMINGR2 register	A	A
	2.15.3	I3C controller: unexpected first frame with a 0x7F address when the I3C peripheral is enabled	A	A
	2.15.4	I3C controller: no timestamp on IBI acknowledge when timing control is used in Asynchronous mode 0	A	A
USART	2.16.1	Wrong data received by SPI slave receiver in autonomous mode with CPOL = 1	A	A
	2.16.2	Received data may be corrupted upon clearing the ABREN bit	A	A
	2.16.3	Noise error flag set while ONEBIT is set	N	N
LPUART	2.17.1	Possible LPUART transmitter issue when using low BRR[15:0] value	P	P
SPI	2.18.1	RDY output failure at high serial clock frequency	N	N
FDCAN	2.19.1	Desynchronization under specific condition with edge filtering enabled	A	A
	2.19.2	Tx FIFO messages inverted under specific buffer usage and priority setting	A	A
	2.19.3	DAR mode transmission failure due to lost arbitration	A	A
UCPD	2.20.1	TXHRST upon write data underflow corrupting the CRC of the next packet	A	A
	2.20.2	Ordered set with multiple errors in a single K-code is reported as invalid	N	N
	2.20.3	UCPDPHY specification marginality for ZDRIVER	A	A
ETH	2.21.1	Incorrect gate control list switching for intermediate cycles when CTR is less than the GCL execution time	A	A
ETHSW	2.22.1	Reported bridge delays do not match measured values	P	P
	2.22.2	Tagged link local frames are discarded if the port is not part of the VLAN	N	N
	2.22.3	TSN switch controller tags management traffic when the PVID is configured differently on internal and external ports	P	P
	2.22.4	Express/preemptable selection must be per priority, not per queue (traffic class)	P	P
	2.22.6	Higher priority frame may overtake lower priority frame	N	N

The following table gives a quick reference to the documentation errata.

Table 4. Summary of device documentation errata

Function	Section	Documentation erratum
System	2.3.10	LSEDRV description is swapped
OCTOSPI	2.5.8	Automatic status-polling mode cannot be used with HyperFlash™ memories
ETHSW	2.22.5	Priority queue drop is not working as specified

2 Description of device errata

The following sections describe the errata of the applicable devices with Arm® core and provide workarounds if available. They are grouped by device functions.

Note: Arm is a registered trademark of Arm Limited (or its subsidiaries) in the US and/or elsewhere.

2.1 Arm Cortex-A35 core

Reference manual and errata notice for the Arm® Cortex®-A35 core revision r1p0 is available from http://infocenter.arm.com. Only applicable information from the Arm errata notice is replicated in this document.

2.1.1 Speculative AT instruction using out-of-context translation regime could cause subsequent request to generate an incorrect translation

This limitation is registered under Arm ID number 1608096 as Cat B, with significant impact to the silicon devices using this Arm core.

Description

A speculative *address translation* (AT) instruction translates using registers that are associated with an out-of-context translation regime and caches the resulting translation in the TLB. A subsequent translation request that is generated when the out-of-context translation regime is current uses the previous cached TLB entry producing an incorrect virtual to physical mapping.

The wrong speculative address translation may occur when the following conditions are all met:

1. A speculative AT instruction performs a table walk, translating a virtual address to a physical address using registers associated with an out-of-context translation regime.
2. Address translation data that is generated during the walk is cached in the TLB.
3. The out-of-context translation regime becomes current and a subsequent memory access is translated using previously cached address translation data in the TLB, resulting in an incorrect virtual to physical mapping.

When these conditions are met, the resulting translation is incorrect.

Workaround

When context-switching the register state for an out-of-context translation regime, system software at EL2 or above must ensure that all intermediate states during the context switch would report a level 0 translation fault in response to an AT instruction targeting the out-of-context translation regime.

A workaround is only required if the system software contains an AT instruction as part of an executable page.

2.1.2 Some AT instructions executed from EL3 might incorrectly report a domain fault

This limitation is registered under Arm ID number 799764 as Cat B, with significant impact to the silicon devices using this Arm core.

Description

Address translation instructions executed from EL3 and targeting EL1 or EL0 might report an incorrect result in the PAR when the HCR_EL2. DC bit is set.

The failure occurs when the following conditions are all met:

1. The core is executing at *exception* level 3 in AArch64.
2. The core executes one of the following address translation instructions:
   - AT S1E0R, AT S1E0W
   - AT S1E1R, AT S1E1W
   - AT S12E0R, AT S12E0W
   - AT S12E1R, AT S12E1W
3. The Exception level targeted by the address translation instruction is Non-secure and AArch32.
4. HCR_EL2.DC is set.

5. The DACR is programmed so that domain 0 would cause a domain fault if the HCR_EL2.DC bit had not been set. Note that this is the default value out of reset.

When these conditions are met, the PAR register incorrectly reports that a domain fault occurred.

Workaround

Secure software can set the DACR[1:0] to 0b01 before executing the address translation instruction. It should restore the previous DACR value before returning to a lower *exception* level.

2.1.3 ATB flush response may be delayed

This limitation is registered under Arm ID number 2252746 as Cat C, with minor impact to the silicon devices using this Arm core.

Description

The *embedded trace macrocell* (ETM) supports an external flush request for each ATB bus. Under certain timing conditions, an AFREADY response from the processor may be delayed until a new ATB transfer is generated by the processor.

The erratum occurs when the following conditions are met:

1. The ETM is enabled.
2. Trace data is generated on the ATB bus.
3. An external flush request is generated.
4. Trace data stops being generated due to filtering in the ETM or the ETM being disabled.

When these conditions are met, external trace infrastructure which is waiting for trace to be captured may stall forever (for example in a *flush and stop* scenario). All of the trace data are captured, but it is not possible to identify this by polling the *trace capture device*. If the flush is acknowledged, this can be treated as a reliable indication.

If the ETM is enabled again after the erratum has been triggered, the flush logic should become active again. If a flush is generated while trace is being generated, the only effect will be a delay in acknowledging the flush. This should not have any observable impact.

In a system with multiple trace sources, the delayed flush response may prevent other trace sources from accessing the ATB bus if they are generating trace while the flush is in progress. This could cause trace to be lost from these other sources.

Workaround

There is no workaround to reliably avoid this erratum.

As an alternative to waiting for the flush to be acknowledged, a sufficiently long timeout can be used if it is likely that trace generation has stopped. If ATB upsizers are present in the system, this workaround will not be effective.

2.1.4 PMU counter might be inaccurate when monitoring BUS_ACCESS and BUS_ACCESS_ST

This limitation is registered under Arm ID number 1010162 as Cat C, with minor impact to the silicon devices using this Arm core.

Description

The Cortex-A35 processor implements a *performance monitor unit* (PMU). The PMU allows programmers to gather statistics on the operation of the processor during runtime. Because of this erratum, the PMU counter values might be inaccurate when monitoring BUS_ACCESS and BUS_ACCESS_ST events.

This limitation occurs under the following conditions:

1. A performance counter is enabled and configured to count BUS_ACCESS or BUS_ACCESS_ST events.
2. A write or eviction occurs on the bus.

When these conditions are met, the PMU counter will erroneously increment or erroneously fail to increment. The inaccuracy varies between cores. Some bus accesses for core 0 might be attributed to core 1. The impact on the final counter value in each core is high.

This might lead to inaccurate results when using the PMU to debug or profile code.

Workaround

The BUS_ACCESS and BUS_ACCESS_ST events can be counted accurately for core 0 by enabling the counter on both core 0 and core 1 and taking the total. This workaround requires core 1 to be present in the configuration and idle during testing.

The event L2D_CACHE_WB counts write-backs from the L2 cache that are attributable to a core. For a test focused on cacheable memory bandwidth, this might be a suitable replacement for BUS_ACCESS_ST. This event includes:

- write-backs as a result of evictions and cache maintenance instructions
- writes in read allocate mode (write-streaming mode)

L2D_CACHE_WB does not include:

- write-backs as a result of snoop requests from outside of the cluster
- write-backs as a result of ACP interface accesses

2.1.5 Mismatch between EDPRSR.SR and EDPRSR.R

This limitation is registered under Arm ID number 857487 as Cat C, with minor impact to the silicon devices using this Arm core.

Description

The processor provides access to the EDPRSR through the APB interface. If this access is done at the same time as the core leaves a *warm* reset, then a subsequent read of the same register reads an incorrect value of the SR field.

The error occurs when the following conditions are met:

1. The core is in *warm* reset.
2. A debugger reads the EDPRSR register over the APB interface.
3. The core comes out of *warm* reset during the APB read.
4. A second APB read is made to the EDPRSR register. One or more exception types are routed to *monitor* mode by setting one or more of SCR.{EA, FIQ, IRQ} bits.

When these conditions are met then the exception mask bit CPSR.{A, F, I} is cleared for each exception type that meets the conditions 1. and 2.. The affected mask bits are cleared together regardless of the exception type in the condition 3..

The implications of this erratum are:

- The first read of the EDPRSR reads the SR field and R field both as 0b1.
- The second read of the EDPRSR.SR field reads 0b0 whereas the previous read of the EDPRSR.SR was 0b1 while in *warm* reset. Because the first read took place while in *warm* reset, the sticky bit must still be set on the second read.

Workaround

If the debugger reads the EDPRSR and sees both the SR and R fields set, then it must remember this result and on the next EDPRSR read treat the SR bit as if it was set.

2.1.6 ATS12NSOPR instruction might incorrectly translate when the HCR.TGE bit is set

This limitation is registered under Arm ID number 801757 as Cat C, with minor impact to the silicon devices using this Arm core.

Description

An ATS12NSOPR address translation instruction executed from EL3 might report an incorrect result in the PAR when both the HCR.TGE bit is set and the SCTLR.M is set.

The error occurs when the following conditions are met:

1. The core is executing at *exception* level 3 in AArch32.
2. SCR.NS = 0
3. HCR.TGE = 1
4. SCTLR(ns).M = 1
5. The core executes an ATS12NSOPR instruction.

If the above conditions are met, the PAR register incorrectly reports the translation as if the stage 1 MMU was enabled.

Note: The combination of both HCR.TGE and SCTLR.M bits being set was unpredictable in earlier versions of the architecture, and was only given a defined behavior to reduce the unpredictable space. It is not expected to be a useful combination for software.

Workaround

Secure software can clear the SCTLR(ns).M bit before executing the address translation instruction, if the HCR.TGE bit is set. It should restore the previous SCTLR(ns).M value before returning to a lower *exception* level.

2.2 Arm Cortex-M33 core

Reference manual and errata notice for the Arm® Cortex®-M33 core revision r0p4 is available from http://infocenter.arm.com.

2.2.1 Access permission faults are prioritized over unaligned Device memory faults

Description

A load or store which causes an unaligned access to Device memory will result in an UNALIGNED UsageFault exception. However, if the region is not accessible because of the MPU access permissions (as specified in MPU_RBAR.AP), then the resulting MemManage fault will be prioritized over the UsageFault.

The failure occurs when the MPU is enabled and:

- A load/store access occurs to an address which is not aligned to the data type specified in the instruction.
- The memory access hits one region only.
- The region attributes (specified in the MAIR register) mark the location as Device memory.
- The region access permissions prevent the access (that is, unprivileged or write not allowed).

The MemManage fault caused by the access permission violation will be prioritized over the UNALIGNED UsageFault exception because of the memory attributes.

Workaround

None. However, it is expected that no existing software is relying on this behavior since it was permitted in Armv7-M.

2.3 System

2.3.1 ADF1/MDF1 kernel clock not provided in autonomous mode

Description

When hse_ker_ck/msi_ker_ck clock is selected and HSEKERON/MSIKERON bit is not set in RCC_OCENSETR/RCC_D3DCR registers then ADF1/MDF1 kernel clock is not provided when the peripheral is working in autonomous mode and the bus clock request is active.

Workaround

Set HSEKERON/MSIKERON bit in RCC_OCENSETR/RCC_D3DCR registers.

2.3.2 Debug port unavailable during backup domain software reset VSWRST

Description

When RCC_BDCR.VSWRST is set, the debug port becomes unavailable. This causes two types of issues:

- The backup domain cannot be reset through the debugger (the debugger can set RCC_BDCR.VSWRST, but then the debugger looses control and cannot clear the bit);
- The debugger looses control during system boot, when the secure OS resets the backup domain, logging many errors.

Workaround

RCC_BDCR.VSWRST must not be set by the debug port.

2.3.3 Incorrect JEDEC ID on AP2 (Cortex-M0+)

Description

The value of the JEDEC ID from AP2 (Cortex-M0+) is not compliant with ST rules. ST tools (ST-Link, CubeIDE) check the JEDEC ID to verify if the device belongs to ST. Having non-ST JEDEC ID blocks the tools.

Workaround

For debugging Cortex-M0+ through the main JTAG/SWD, a special debug initialization sequence must be used to first detect the device through AP0 (that has correct ST JEDEC ID), then switch to AP2 for Cortex-M0+.

2.3.4 Wrong SYSCFG_IPIDR reset value

Description

Value if SYSCFG_IPIDR is 0x00030001 whereas it should be 0x00030003.

Workaround

This value is the same than STM32MP15 and might lead to confusion. If needed, DEV_ID should be used to distinguish the different products.

2.3.5 Compartment filtering of PWR_CR11 and PWR_CR12 registers is not functional

Description

Once the compartment filtering is enabled on PWR_CR11 (RCC internal resource 104) and PWR_CR12 (RIFSC peripheral #79), the registers are no writeable by any CIDs even the permitted one.

Workaround

CID filtering should be disabled before accessing PWR_CR11 and PWR_CR12 registers.

2.3.6 STGEN is reset when D1 domain is in DStandby low power mode

Description

The STGEN is reset by a cpu1_rstn signal. As a consequence, STGEN is reset whenever CPU1 is woken-up from any of the RUN2, STOP2, LP-STOP2, or LPLV-STOP2 modes.

Workaround

None.

2.3.7 Unwanted IP reset when D1 domain exit from DStandby

Description

A reset glitch could occur on the VDDCORE domain when the D1 domain exit from DStandby. Some peripherals could be unduly reset. This depends on the device, the VDDCORE voltage, and the temperature.

Workaround

Do not use low power modes involving D1 DStandby, such as RUN2, STOP2, LP-STOP2, or LPLV-STOP2 modes.

2.3.8 LPLV-STOP2 exit failed if D3 using LSE or LSI clock

Description

When selecting a slow clock such as LSE or LSI for the D3 domain (RCC_D3DCR.D3PERCKSEL), the boot ROM is badly detecting an issue and going to a deadlock state on wake-up from LPLV-STOP2.

Workaround

Always use MSI (4 MHz or 16 MHz) for D3 domain clocking if LPLV-STOP2 is going to be used.

2.3.9 GPU reset randomly not released

Description

Performing a GPU software reset when the PLL3 reference clock frequency is bigger than any of GPU or RCC clock frequencies lead to an unpredictable GPU state.

Workaround

Application must respect the following constraints whenever a GPU reset is issued (RCC_GPUCFGR.GPURST=1):

- PLL3 reference clock frequency (ck_pll3_ref) must be lower than GPU clocks (ck_icn_p_gpu, ck_icn_m_gpu and ck_ker_gpu) and RCC clock (ck_icn_p_rcc, which is ck_icn_ls_mcu) frequencies.

2.3.10 LSEDRV description is swapped

Description

Some reference manual and datasheet revisions provide wrong RCC_BDCR.LSEDRV[1:0] description for value 0x1 and 0x2. The correct description is :

- 0x0: Lowest drive
- 0x1: Medium-high drive (recommended setting with most crystals)
- 0x2: Medium-low drive (default after backup domain reset)
- 0x3: Highest drive

It should be noted that the recommended setting is not the default field value after reset.

This is a documentation issue rather than a device limitation.

Workaround

No application workaround is required if the LSEDRV setting is conform to the latest documentation.

2.3.11 Boot fails after wakeup from STANDBY when booting on SNOR, SNAND and HYPERFLASH

Description

When booting using OCTOSPI interface (Serial-NOR, Serial-NAND, HYPERFLASH) the wakeup from STANDBY fails.

Workaround

None.

2.3.12 Boot ROM hangs when system reset is applied while D1 domain is in DStandby state

Description

When D1 domain is in DSTANDBY state, regardless of the TDCID value for CPU1 or CPU2 , if one of the following reset is applied, the system hangs:

- NRST input reset
- SYSC2RST (CPU2 system reset)
- RETCRCERRRST (RETRAM CRC error reset)
- RETECCFAILCRCRST (RETRAM ECC failure reset flag during the CRC computation phase)
- RETECCFAILRESTRST (RETRAM ECC failure reset flag during the system restoration phase)
- HCSSRST (HSE CSS reset)

Workaround

Issue a second NRST input reset sequence if this is possible on the platform.

2.3.13 Boot ROM writes in SYSRAM during LPLV-Stop2 wakeup

Description

During LPLV-Stop2 wake-up, the boot ROM writes dummy 32-bits word values at address 0x0E000080 and 0x0E000084, thus modifying SYSRAM content.

Workaround

The firmware in SYSRAM shall not use these addresses.

2.3.14 Instruction fetch access to PWR register lead to unwanted write

Description

In case of instruction fetch within peripheral space, a response error is generated by all peripherals. In case of PWR, even if the response error is well generated, an unwanted write access is perform on the latest PWR register access, thus corrupting the value.

Workaround

No workaround. Ensure instruction fetch within peripheral area is not possible by correct programming of Cortex-A MMU or Cortex-M MPU when available.

2.3.15 CPU2 (Cortex-M33) does not support debug in non-secure only

Description

The Cortex-M33 access port AHB-AP (AP8) is not accessible when “microcontroller debug / non-secure / full debug” profile is selected (BSEC_DENR[15:0] = 0x0398). This mean the debug and trace features integrated in the Cortex-M33 are not accessible. Debug activity of the device can still be done using system interconnect AXI-AP (AP4).

Workaround

Use “microcontroller debug / secure / full debug” profile (BSEC_DENR[15:0] = 0x0F98).

2.3.16 LSE function can be impacted if there is negative current injection in GPIOs

Description

Negative current injection on V_{SW} domain can interfere with very low power circuit, for instance, it can stop LSE oscillator. Negative current injection on V_{SW} domain can occur when there is undershoot on GPIO pins. It can affect any V_{SW} domain GPIO or V_{DD} domain GPIO that share its function with TAMPER_IN or TAMPER_OUT.

Workaround

The PCB design must ensure there is no negative current injection in related GPIOs (for example, series resistor on far side output, filtering capacitors on input, clamping Schottky diodes can be used).

2.3.17 ETHx kernel clock is gated if ETHxMACEN register bit is not set

Description

In registers RCC_ETHxCFGR, when setting only ETHxEN or ETHxLPEN bits, the kernel clock is not enabled. The ETHxMACEN bit controlling the gating of the bus clock must also be set to allow enabling the kernel clock (x = 1, 2 or SW for respectively ETH1, ETH2 and ETHSW).

Workaround

Each time the ck_ker_ethx kernel clock is required, the corresponding ETHxMACEN bit must be set together with regular kernel clock enable bit.

2.3.18 RISAF wrong SIDR value

Description

The value of RISAF_SIDR is 0xA3C5 DD01 whereas it should be 0xA3C5 DD04. 4 Kbytes are allocated to RISAF in the memory map.

Workaround

None.

2.3.19 STOP and Standby entry failed when DDR is in shared mode

Description

Once RCC_DDRITFCFGR.DDRSHR is set to 1, the system cannot enter in low power mode (Stop and Standby modes) if the DDRSS bus clocks are kept enabled.

Workaround

The DDRSS bus clocks must be disabled by software before entering low power mode. The bus clocks are disabled by clearing RCC_DDRCPCFGR.DDRCPEN, RCC_DDRPHYCAPBCFGR.DDRPHYCAPBEN and RCC_DDRCFGR.DDRCFGEN registers bits.

2.3.20 Cortex-A35 not restarted after system reset in TDCID Cortex-M33 configuration

Description

In TDCID = 0x2 (Cortex-M33) configuration, the first boot is successful (Cortex-M33 or Cortex-A35 flash boot), but if the TAMP_BKP11R has been set with a value corresponding to a SYSRAM address (intended for CPU1 CStandby exit), then in case of system reset, the Cortex-A35 does not restart. If the system reset is done with backup domain kept switched on during the power cycle, the system does not boot until the backup domain is switched off.

Workaround

The software must update the configuration after cold-boot (system reset applied), standby wakeup, Cortex-A35 reset initiated by Cortex-M33 or Watchdog reset with TAMP_BKP11R = 0x00000000.

2.3.21 AHB RISAB3/4/5 illegal access due to ghost CID0 detection

Description

When an AHB busy signal is inserted during a transaction, a ghost CID0 is generated on the bus. If the compartment filtering is enabled on RISAB3/4/5, this transient CID0 is interpreted as a fault access by RISAB3/4/5 which aborts current access and returns an IAC.

Workaround

CID0 must be “reserved”: not assigned to any master at RISFC RIMU level.

- RIFSC_RIMC_ATTRx.MCID must not be programmed with CID0 value.

CID0 must be “reserved”: not assigned to any master at RIFSC RISUP level.

- RIFSC_RISC_PERy_CIDCFGR.CFEN must be set to 1 (compartment filtering enabled) to avoid use of CID0 default at RISUP level.
- RIFSC_RISC_PERy_CIDCFGR.SCID must not be programmed with CID0 value.
- RIFSC_RISC_PERy_CIDCFGR.SEMWLC0 must not be set to 1.

CID0 must be “reserved” for HPDMAx channel.

- HPDMA_CxCIDCFGR.CFEN must be set to 1 (compartment filtering enabled) to avoid use of CID0 default value.
- HPDMA_CxCIDCFGR.SCID must not be programmed with CID0 value.
- HPDMA_CxCIDCFGR.SEM_WLIST_CID0 must not be set to 1.

For RISAB3/4/5, CID0 read and write accesses must be enabled for all blocks/pages.

- RISAB3/4/5_CID0RDCFGR = 0xFFFFFFFF.
- RISAB3/4/5_CID0WRCFGR = 0xFFFFFFFF.

2.3.22 ETM timestamp is exported with zero value

Description

When attempting to use ETM traces, the timestamp value exported by default is zero. This is because STM clock is needed for ETM trace timestamps.

Workaround

Set RCC_STMCFGR.STMEN bit to 1.

2.4 FMC

2.4.1 NOR flash memory/PSRAM incorrect bus turnaround timing

Description

The delays between consecutive device accesses, programmed through the BUSTURN[3:0] bitfield of the FMC_BTRx and FMC_BWTRx registers, have no effect. Instead systematic delays are applied:

- t_{IDLE2}: 2 * fmc_ker_ck cycles between accesses to two NOR/PSRAM devices
- t_{IDLE1}: 1 * fmc_ker_ck cycle between accesses to a NOR/PSRAM and a NAND flash memory

Workaround

Extend the bus turnaround delays to satisfy bus turnaround constraints. Three cases need to be considered:

1. Two consecutive accesses to non-multiplexed NOR/PSRAM devices:
   Program t_{ADDSET} (NOR/PSRAM address setup phase) as needed (see Figure 1).
2. Access to a non-multiplexed NOR/PSRAM followed by an access either to a NOR/PSRAM device with multiplexed A/DQ signals or to a synchronous device:
   Decrease the FMC kernel clock frequency in order to meet the timing constraints (see Figure 2).
3. Access to a non-multiplexed NOR/PSRAM followed by an access to a NAND flash memory device
   Program t_{MEMSET} (NOR/PSRAM address setup phase) as needed (see Figure 3).

Figure 1. Bus turn timing recovery - asynchronous accesses to NOR/PSRAM devices (case 1)

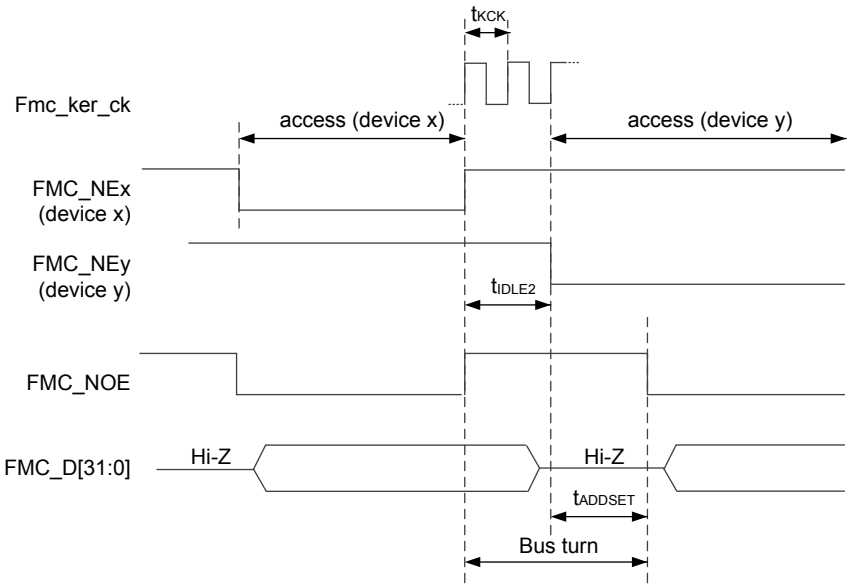

Figure 2. Bus turn timing recovery - access to NOR/PSRAM followed by access to multiplexed A/DQ synchronous device (case 2)

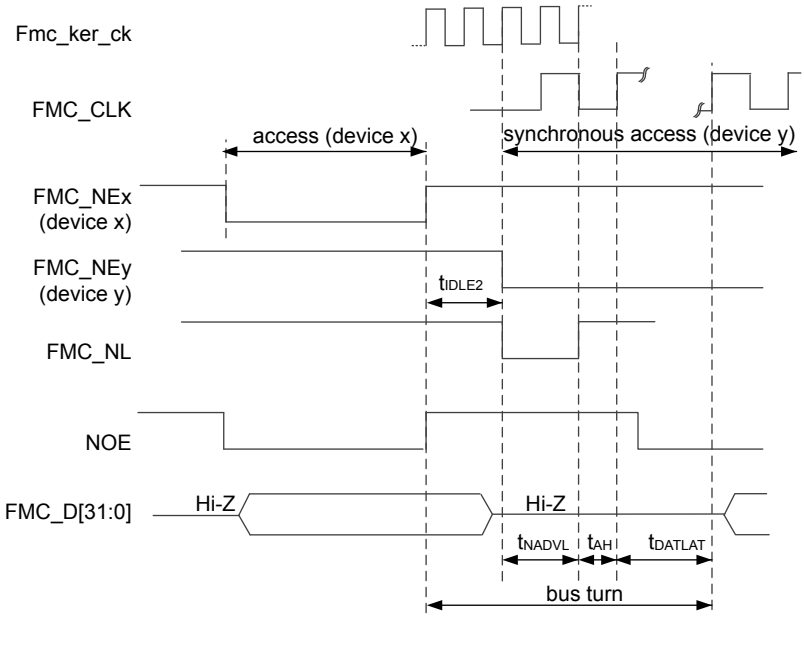

Figure 3. Bus turn timing recovery - access to NOR/PSRAM followed by access to NAND Flash device (case 3)

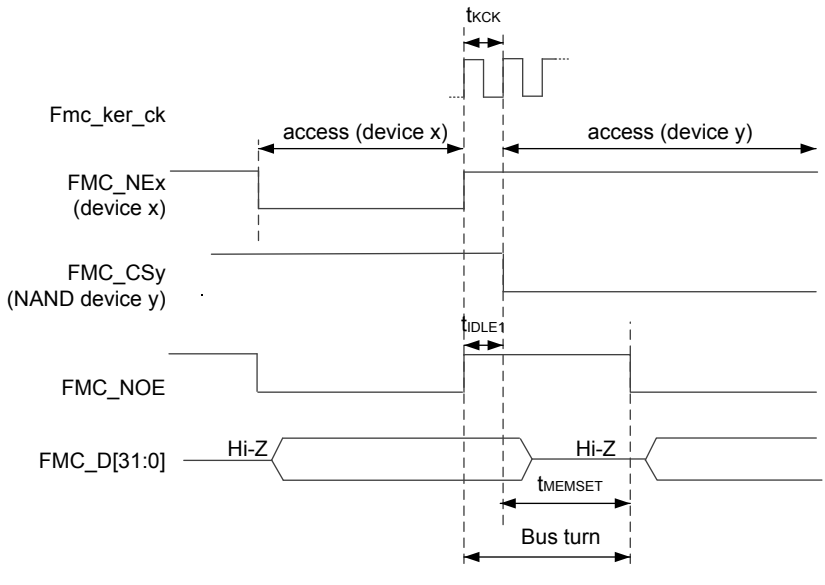

Table 5 gives the internal latencies that are not mentioned in the product datasheets. Refer to the datasheets for more details about the others timing values.

Table 5. Timing parameter description

Parameter	Description	Minimum value
t_{KCK}	FMC kernel clock period	See product datasheet

Parameter	Description	Minimum value
t_{IDLE1}	NEx high to CSy low (switching to a NAND flash memory device)	1 * t_{KCK}
t_{IDLE2}	NEx high to NEy low (NOR/PSRAM devices)	2 * t_{KCK}
t_{ADDSET}	NOR/PSRAM address setup phase	See Ref. Man.
t_{NADVL}	Address valid low pulse duration in synchronous mode	(CLKDIV+1) * t_{KCK}
t_{AH}	Address hold in synchronous mode	((CLKDIV+1) * t_{KCK}) / 2
t_{DATLAT}	NOR/PSRAM data latency in synchronous mode	See product datasheet
$t_{MEMxSET}$	NAND flash memory address setup phase	See product datasheet

2.4.2 Incorrect FMC_CLK clock period when CLKDIV value is changed on-the-fly in Continuous clock mode

Description

When the FMC operates in Continuous clock mode (CCLKEN is set in FMC_BCRx register), a new clock division factor is applied by changing the value of CLKDIV[3:0] in FMC_CFGR while the FMC is disabled (FMCEN cleared in FMC_BCRx), there is one FMC_CLK clock cycle during which the FMC_CLK period is not as expected: for example the clock low pulse duration matches the previous CLKDIV[3:0] value whereas the clock high pulse duration matches the new CLKDIV[3:0] value.

Workaround

Use the following sequence:

1. Stop the memory traffic for all devices.
2. Disable the FMC (refer to the disabling sequence described in the product reference manual).
3. Change CCLKEN for 1 to 0 in the FMC_BCRx register to stop the clock generation.
4. Program the desired CLKDIV[3:0] value in the FMC_CFGR register.
5. Change back CCLKEN from 0 to 1.
6. Enable the FMC.

2.5 OCTOSPI

2.5.1 Memory-mapped write error response when DQS output is disabled

Description

If the DQSE control bit of the OCTOSPI_WCCR register is cleared for memories without DQS pin, it results in an error response for every memory-mapped write request.

Workaround

When doing memory-mapped writes, set the DQSE bit of the OCTOSPI_WCCR register, even for memories that have no DQS pin.

2.5.2 Deadlock can occur under certain conditions

Description

A deadlock can occur when all the following conditions are met:

- The product communicates through an I/O manager in multiplexed mode with an single external memory or an external combo featuring two memories, directly or through a high-speed interface.
- The external memory(ies) is(are) accessed in indirect mode or memory-mapped mode.

The deadlock can happen when the two following conditions occur at the same time:

- The Octo-SPI interface that currently owns the external bus (for example OCTOSPI1) waits for a transfer to occur with the external memory, to complete its transfer on the internal interconnect matrix bus.
- A data transfer request on the internal interconnect matrix bus arrives to the other Octo-SPI interface (for example OCTOSPI2).

This leads to an ownership conflict where:

- OCTOSPI2 cannot get ownership of the external bus which is currently in use by OCTOSPI1.
- OCTOSPI1 cannot get ownership of the internal interconnect matrix bus which is currently in use by OCTOSPI2.

Workaround

Apply one of the following measures:

- If any of the features generating automatic transfer split (MAXTRAN, REFRESH, CSBOUND, TIMEOUT) is set, OCTOSPI1 splits its transfer at some point in time, releasing the bus. OCTOSPI2 can then process its data, and when OCTOSPI1 gets ownership back again, it resumes its transfer thanks to its embedded capability to restart at the address following the last address accessed. In this case, the deadlock is resolved.
  Limitation of the workaround: The automatic resume of the transfer does not work with certain flash memories in write direction only. These memories require an extra "write enable" command before resuming a write transfer. This "write enable" command is not generated by the OCTOSPI.
- The application must ensure that it has sufficient room left in the OCTOSPI internal FIFO for each and every transfer before launching it. The internal interconnect matrix bus activity no longer depends on what happens on external bus side, and the deadlock condition is avoided.

2.5.3 Memory wrap instruction not enabled when DQS is disabled

Description

Memory wrap instruction (as configured in the OCTOSPI_WPxxx registers) is not generated when DQS is disabled. The memory wrap instruction is replaced by two regular successive read instructions to ensure the correct data ordering: this split has very limited impact on performance.

Workaround

None.

2.5.4 Deadlock or write-data corruption after spurious write to a misaligned address in OCTOSPI_AR register

Description

Upon writing a misaligned address to OCTOSPI_AR just before switching to memory-mapped mode (without first triggering the indirect write operation), with the OCTOSPI configured as follows:

- FMODE = 00 in OCTOSPI_CR (indirect write mode)
- DQSE = 1 in OCTOSPI_CCR (DQS active)

then, the OCTOSPI may be deadlocked on the first memory-mapped request or the first memory-mapped write to memory (and any sequential writes after it) may be corrupted.

An address is misaligned if:

- the address is odd and the OCTOSPI is configured to send two bytes of data to the memory every cycle (octal-DTR mode or dual-quad-DTR mode), or
- the address is not a multiple of four when the OCTOSPI is configured to send four bytes of data to the memory (16-bit DTR mode or dual-octal DTR mode).

If the OCTOSPI_AR register is reprogrammed with an aligned address (without triggering the indirect write between the two writes to OCTOSPI register), the data sent to the memory during the indirect write operation are also corrupted.

Workaround

None.

2.5.5 Deadlock on consecutive out-of-range memory-mapped write operations

Description

The DEVSIZE[4:0] bitfield of the OCTOSPI_DCR1 register indicates that the size of the memory is 2 ^ [DEVSIZE + 1] bytes, and thus any memory-mapped access to address 2 ^ [DEVSIZE + 1] or above should get an error response.

However, no error response may be returned and the OCTOSPI may become deadlocked after the following sequence of events:

1. A memory-mapped write operation is ongoing on the AHB bus.
2. A second memory-mapped write is requested to an address close to the end of the memory but not consecutive to the address targeted by the first write operation.
3. A third memory-mapped write operation is requested, this time to an address consecutive to the address targeted by the second write, and the address of this third write is 2 ^ [DEVSIZE + 1] or an address consecutive to 2 ^ [DEVSIZE + 1].
   If the first write command has not completed writing data, then the write to 2 ^ [DEVSIZE + 1] does not return any error response and the next memory-mapped request gets stalled indefinitely.

Workaround

Ensure that no sequences of consecutive memory-mapped write operations pass the memory boundary.

2.5.6 Indirect write mode limited to 256 Mbytes

Description

In indirect write mode, if the address is greater than 256 Mbytes, the indirect write is not performed at the targeted address, even if it is located inside the allowed memory space configured through the device size (DEVSIZE[4:0] of OCTOSPI_DCR1). Actually, this write operation takes place within the 256-Mbyte memory space, thus corrupting the memory content.

Indirect read operations are not impacted.

Workaround

Indirect write operations have to be performed inside the first 256 Mbytes of the memory space.

2.5.7 Read-modify-write operation does not clear the MSEL bit

Description

When the MSEL bit of the OCTOSPI_CR register is set, it remains set even if the software attempts to clear it by performing a read-modify-write operation.

Workaround

To clear the MSEL bit, clear in a single write access bit 7 and bit 30 of the OCTOSPI_CR register, otherwise, the MSEL bit remains set.

2.5.8 Automatic status-polling mode cannot be used with HyperFlash™ memories

Description

Some reference manuals mention that the automatic status-polling mode can be used with the HyperBus™ protocol. This is not possible since HyperFlash™ memories require two steps to read the status register (a write operation followed by a read command), while the automatic status-polling mode, already implemented in the regular-command protocol, requires a single read instruction to read back the status register.

This is a documentation issue rather than a product limitation.

Workaround

None.

2.5.9 Setting the ABORT bit does not generate an error on the AHB bus for undefined-length incremental burst transfers

Description

An AHB error is expected to be generated when the ABORT bit of the OCTOSPI_CR register is set while a request is ongoing.

Instead, the controller does not trigger any AHB error if the ongoing request is an undefined-length incremental burst AHB transfer.

An AHB error is generated for all other transfer types.

Workaround

When possible, wait for the end of the transfer before setting the ABORT bit.

2.5.10 Read data corruption when a wrap transaction is followed by a linear read to the same MSB address

Description

If a wrap transaction is followed by a linear read having the same MSB start address as the wrap (), then the linear read is wrongly considered as a sequential transaction to the previous one, taking back the prefetched data and causing data corruption.

Notice that for a wrap transaction, the prefetch starts after the last address of the wrap window.

Workaround

As prefetch cannot be disabled, there is no workaround. However, the issue is seldom encountered since wrap operations are mostly initiated by the internal cache to refresh its cacheline. All the other masters must avoid retrieving data by using a linear read access to the same MSB address as the wrap, which has been just completed.

2.5.11 Transactions are limited to 8 Mbytes in OctaRAM™ memories

Description

When the controller is configured in Macronix OctaRAM™ mode, by setting the MTYP[2:0] bitfield of the OCTOSPI_DCR1 register to 011, only 13 bits of row address are decoded and sent to the memory, meaning that only 8 K of 1-Kbyte blocks can be accessed (8 Mbytes).

Workaround

None.

This limitation is not present for PSRAMs or HyperRAM™ memories.

2.5.12 Variable latency is not supported when a refresh collision occurs during a write access to some OctaRAM™ memories

Description

When the memory type (MTYP[2:0] bitfield of the OCTOSPI_CR register) is configured to 0b011 to target an OctaRAM™ memory, the host controller does not support the variable latency requested by the external memory if a refresh collision occurs during the write access. For example, some OctaRAM™ memories, such as ISSI memories, request extra latency cycles for write accesses during refresh collision. In this case, the controller does not sample the DQS input signal during the instruction phase, and cannot detect the extra latency requested by the external memory for the refresh operation.This results in data corruption.

Some OctaRAM™ memories do not request any additional latency for write access during refresh cycles. It is required only when the refresh occurs during a read access. In this case, no issue can be observed.

Workaround

When the application targets an OctaRAM™ memory that requests extra latency cycles for write access during refresh collision, force the fixed latency mode in the configuration register of the external memory. There is no constraint about read access, since both variable and fixed latency modes are supported.

2.6 OCTOSPIM

2.6.1 Certain quad memories may be reset during arbitration while in single-SPI mode

Description

The OCTOSPI I/O manager allows two OCTOSPIs to be mapped on the same I/Os, in which case the OCTOSPIs arbitrate for use of the multiplexed port. This arbitration introduces a glitch on the data lines when the arbitration passes the ownership of the port from one OCTOSPI to the other.

External quad memories, having their asynchronous $\overline{\text{RESET}}$ pin (when selected by default by the memory) multiplexed with an SO data line and operating in single-SPI mode, may be asynchronously reset due to the glitch on the data line when the ownership of the port is transferred.

This problem typically occurs when the memory defaults to operate in single-bit mode and the application reconfigures the memory in quad mode, while arbitrating and transferring the port ownership.

Workaround

Ensure that the ownership of the port does not change while an OCTOSPI is configuring its memory to operate in quad mode:

1. Configure the first memory to quad mode, while clearing MUXEN of OCTOSPIM_CR.
2. Switch the ownership of the port by inverting the MODE bit value in the OCTOSPIM_CR register (even if a glitch is present on the corresponding data line, the reset is applied to the memory that is not yet configured)
3. Configure the second memory to quad mode, while clearing MUXEN of OCTOSPIM_CR.
4. Write the MODE bit to the correct value and set the MUXEN bit of the OCTOSPIM_CR register to come back to the multiplexed mode.

2.7 SDMMC

2.7.1 Command response and receive data end bits not checked

Description

The command response and receive data end bits are not checked by the SDMMC. A reception with only a wrong end bit value is not detected. This does not cause a communication failure since the received command response or data is correct.

Workaround

None.

2.8 ADC

2.8.1 JEOS may be set before the last injected data are available in ADC_JDRx

Description

When the ADC peripheral clock (adc_hclk) is slower than the ADC kernel clock (adc_ker_ck), the JEOS flag of the ADC_ISR register may be set before the last data of the injected sequence are available in the ADC_JDRx register.

Workaround

Apply one of the following measures:

- Ensure adc_hclk is faster than adc_ker_ck.
- Select the discontinuous conversion mode for regular channels by setting the DISCEN bit of the ADC_CFGR1 register.
- Use oversampling for injected conversions.
- Count the number of injected data conversions, by monitoring the JEOC flag of the ADC_ISR register.

2.8.2 In combined regular simultaneous plus alternate trigger mode, stopping injected conversion may delay regular conversion

Description

In dual ADC combined regular simultaneous plus alternate trigger mode, the resumption of an active regular conversion may be delayed by a few ADC clock cycles when an injected trigger event coincides with stopping, by application, an ongoing injected conversion.

Note: The dual ADC combined regular simultaneous plus alternate trigger mode is selected by setting the DUAL[4:0] bitfield of the ADCC_CCR register to 0x02. To stop an ongoing injected conversion, the software sets the JADSTP bit of the ADC_CR register.

Workaround

- Design the application so as to avoid injected trigger events from coinciding with stopping, by software or DMA, an ongoing regular conversion.
- Stop the regular conversion before stopping the injected conversion.

2.8.3 When the ADC clock is derived from the AHB clock, the injected conversion latency is not respected if the injected trigger coincides with the stopping of the regular conversion

Description

When the ADC clock is derived from the AHB clock, and the analog-to-digital conversion is triggered by a timer, the latency between the trigger and the start of the ADC sampling is fixed. When both injected and regular conversions are enabled, if the injected trigger coincides with the stopping of regular conversion, the latency of injected conversions becomes one clock cycle shorter than the expected latency.

Note: To stop an ongoing regular conversion, the software sets the ADSTP bit of the ADC_CR register.

Workaround

Apply one of the following measures:

- Avoid triggering injected conversions when the ADSTP bit is set.
- When an injected trigger is expected, keep the ADSTP bit cleared.

2.9 LTDC

2.9.1 Ongoing AXI write never completes if disabling LTDC

Description

If LTDC is disabled during an AXI write transaction, the AXI transaction never completes. This violates the AXI rules.

Workaround

To disable LTDC, apply the following sequence:

1. Disable the display, to avoid visual artifacts on the screen.
2. Disable the rotation, by clearing the ROTEN bit of the LTDC_GCR register.
3. Wait till the end of the next frame, to ensure that no write traffic is ongoing.
4. Disable LTDC, by clearing the LTDCEN bit of the LTDC_GCR register.

2.9.2 Layers cannot read YUV420 multibuffer data

Description

The YUV semiplanar and full-planar modes of the layers are not functional.
The YUV coplanar and the RGB modes can be used.

Workaround

None.

2.9.3 Rotation for wide landscape display can causes artifacts

Description

The rotation feature is limited by the display width, and the pixel clock and the AXI bus frequencies:

- **Display width of 1080 pixels**
  The rotation (landscape to portrait) of a 1920x1080 frame buffer to a 1080x1920 display with the pixel clock of 150 MHz and the AXI clock of 400 MHz is functional.
- **Display width of 1280 pixels**
  The rotation (landscape to portrait)of a 720x1280 frame buffer to a 1280x720 display with the pixel clock of 75 MHz and the AXI clock of 400 MHz is functional.
- **Display width of 1366pixels**
  The rotation (portrait to landscape) of a 768x1366 frame buffer to a 1366x768 display with the pixel clock of 85 MHz and the AXI clock of 400 MHz is functional.
- **Display width exceeding 1366 pixels**
  The rotation can cause artifacts.

Workaround

None.

2.9.4 Layer 1 cannot read YUV420 multibuffer data

Description

The YUV semiplanar and the full planar mode of layer 1 are not functional.
The YUV coplanar or RGB mode can be used.

Workaround

None.

2.10 VENC

2.10.1 VENC hardware self-reset after internal timeout is unstable

Description

The VENC may crash when receiving a corrupted video stream. A timeout is implemented to detect such a crash and self-reset the whole VENC.

The implementation of the self-reset is unstable, and may result in a deadlock of the VENC, which cannot be exited by a hardware asynchronous reset.

Workaround

The software must disable the automatic self-reset feature and use a software timeout instead.

2.11 VDEC

2.11.1 VDEC hardware self-reset after internal timeout is unstable

Description

The VDEC may crash when receiving a corrupted video stream. A timeout is implemented to detect such a crash and self-reset the whole VDEC.

The implementation of the self-reset is unstable, and may result in a deadlock of the VDEC, which cannot be exited by a hardware asynchronous reset.

Workaround

The software must disable the automatic self-reset feature and use a software timeout instead.

2.12 LPTIM

2.12.1 Device may remain stuck in LPTIM interrupt when entering Stop mode

Description

This limitation occurs when disabling the low-power timer (LPTIM).

When the user application clears the ENABLE bit in the LPTIM_CR register within a small time window around one LPTIM interrupt occurrence, then the LPTIM interrupt signal used to wake up the device from Stop mode may be frozen in active state. Consequently, when trying to enter Stop mode, this limitation prevents the device from entering low-power mode and the firmware remains stuck in the LPTIM interrupt routine.

This limitation applies to all Stop modes and to all instances of the LPTIM. Note that the occurrence of this issue is very low.

Workaround

In order to disable a low power timer (LPTIMx) peripheral, do not clear its ENABLE bit in its respective LPTIM_CR register. Instead, reset the whole LPTIMx peripheral via the RCC controller by setting and resetting its respective LPTIMxRST bit in the relevant RCC register.

2.12.2 ARRM and CMPM flags are not set when APB clock is slower than kernel clock

Description

When LPTIM is configured in one shot mode and APB clock is lower than kernel clock, there is a chance that ARRM and CMPM flags are not set at the end of the counting cycle defined by the repetition value REP[7:0]. This issue can only occur when the repetition counter is configured with an odd repetition value.

Workaround

To avoid this issue the following formula must be respected:

{ARR, CMP} ≥ KER_CLK / (2* APB_CLK),

where APB_CLK is the LPTIM APB clock frequency, and KER_CLK is the LPTIM kernel clock frequency. ARR and CMP are expressed in decimal value.

Example: The following example illustrates a configuration where the issue can occur:

- APB clock source (MSI) = 1 MHz , Kernel clock source (HSI) = 16 MHz
- Repetition counter is set with REP[7:0] = 0x3 (odd value)

The above example is subject to issue, unless the user respects:

{CMP, ARR} ≥ 16 MHz / (2 * 1 MHz)

→ ARR must be ≥ 8 and CMP must be ≥ 8

Note: REP set to 0x3 means that effective repetition is REP+1 (= 4) but the user must consider the parity of the value loaded in LPTIM_RCR register (=3, odd) to assess the risk of issue.

2.12.3 Interrupt status flag is cleared by hardware upon writing its corresponding bit in LPTIM_DIER register

Description

When any interrupt bit of the LPTIM_DIER register is modified, the corresponding flag of the LPTIM_ISR register is cleared by hardware.

Workaround

None.

2.13 RTC and TAMP

2.13.1 Alarm flag may be repeatedly set when the core is stopped in debug

Description

When the core is stopped in debug mode, the clock is supplied to subsecond RTC alarm downcounter even when the device is configured to stop the RTC in debug.

As a consequence, when the subsecond counter is used for alarm condition (the MASKSS[3:0] bitfield of the RTC_ALRMASSR and/or RTC_ALRMBSSR register set to a non-zero value) and the alarm condition is met just before entering a breakpoint or printf, the ALRAF and/or ALRBF flag of the RTC_SR register is repeatedly set by hardware during the breakpoint or printf, which makes any attempt to clear the flag(s) ineffective.

Workaround

None.

2.14 I2C

2.14.1 Wrong data sampling when data setup time ($t_{SU;DAT}$) is shorter than one I2C kernel clock period

Description

The I^2C-bus specification and user manual specify a minimum data setup time ($t_{SU;DAT}$) as:

- 250 ns in Standard mode
- 100 ns in Fast mode
- 50 ns in Fast mode Plus

The device does not correctly sample the I²C-bus SDA line when $t_{SU;DAT}$ is smaller than one I2C kernel clock (I²C-bus peripheral clock) period: the previous SDA value is sampled instead of the current one. This can result in a wrong receipt of slave address, data byte, or acknowledge bit.

Workaround

Increase the I2C kernel clock frequency to get I2C kernel clock period within the transmitter minimum data setup time. Alternatively, increase transmitter's minimum data setup time. If the transmitter setup time minimum value corresponds to the minimum value provided in the I²C-bus standard, the minimum I2CCLK frequencies are as follows:

- In Standard mode, if the transmitter minimum setup time is 250 ns, the I2CCLK frequency must be at least 4 MHz.
- In Fast mode, if the transmitter minimum setup time is 100 ns, the I2CCLK frequency must be at least 10 MHz.
- In Fast-mode Plus, if the transmitter minimum setup time is 50 ns, the I2CCLK frequency must be at least 20 MHz.

2.14.2 Spurious bus error detection in master mode

Description

In master mode, a bus error can be detected spuriously, with the consequence of setting the BERR flag of the I2C_SR register and generating bus error interrupt if such interrupt is enabled. Detection of bus error has no effect on the I²C-bus transfer in master mode and any such transfer continues normally.

Workaround

If a bus error interrupt is generated in master mode, the BERR flag must be cleared by software. No other action is required and the ongoing transfer can be handled normally.

2.15 I3C

2.15.1 I3C controller: unexpected read data bytes during a legacy I²C read

Description

Under specific conditions, unexpected data bytes are read during a legacy I²C read transfer.

The issue occurs when all the following conditions are met:

- I3C acts as controller
- a legacy I²C read message is generated
- the STALLT bit of I3C_TIMINGR2 register is set to request the SCL clock to be stalled at low level on the 9th T-bit phase of data bytes (also known as ACK/NACK phase)
- instead of releasing the SDA line, the I²C target incorrectly drives SDA low on the 9th T-bit phase of the end of read from the I3C controller

To end a legacy I²C read, the I3C controller is supposed not to drive SDA low on the 9th T-bit, and to emit a NACK. If the STALLT bit of I3C_TIMINGR2 is set, the controller does not NACK for the purpose of ending the data read transfer.

During the same clock cycle, if the I²C target, instead of releasing the SDA line, incorrectly drives SDA low on this 9th T-bit phase of the end of read from the controller, then the controller detects an incorrect ACK on the I3C bus and keeps SCL clock running.

After 8 clock cycles, the I3C controller generates again an ACK instead of a NACK, and an unexpected dummy data byte is transferred to the RX-FIFO.

Then the target continues transferring data or releases the SDA line, thus causing additional dummy bytes to be received. The transfer can be stopped only when an overrun error occurs.

Workaround

Apply the following measures:

- If the I3C controller is configured with S-FIFO mode enabled (SMODE bit set in I3C_CFGR), the transfer goes on until RX-FIFO is full. Then ERRF = 1 in I3C_EVR (an error occurred), PERR = 1 in I3C_SER (protocol error), DOVR = 1 in I3C_SER (RX-FIFO overrun), and CODERR[3:0] = 001 in I3C_SER (CE1 error).
  It is recommended to enable the error interrupt by setting ERRIE in I3C_IER. When DOVR = 1 and CODERR[3:0] = 0001, flush the RX-FIFO inside the error interrupt service routine by setting RXFLUSH in I3C_CFGR, then clear the CERRF error flag.
- If the I3C controller is configured with S-FIFO mode disabled (SMODE bit cleared in I3C_CFGR), the I3C status register (I3C_SR) may be overwritten by the hardware if unread, thus failing to report any status overrun. An overrun can occur only as a data overrun if the DMA or the software stops reading the RX-FIFO during enough time for the RX-FIFO to be full with dummy bytes. Then both CE1 and DOVR flags are set and an error is reported (ERRF = 1, PERR = 1, CODERR[3:0] = 0001 and DOVR = 1).

Whatever S-FIFO configuration, implement a software timeout to inform that neither FCF nor ERRF error bit was raised during an acceptable time. Then, stop reading RX-FIFO to cause a data overrun to be reported. When an error is reported, if both CE1 and DOVR flags are set (ERRF = 1, PERR = 1, CODERR[3:0] = 0001 and DOVR = 1), flush the RX-FIFO.

2.15.2 I3C controller: SCL clock is not stalled during address ACK/NACK phase following a frame start, when enabled through I3C_TIMINGR2 register

Description

Under specific conditions, the I3C controller does not stall the SCL clock during the address ACK/NACK phase when this feature is configured through I3C_TIMINGR2 register.

The issue occurs when all the following conditions are met:

- I3C acts as controller
- I3C is programmed to stall the SCL clock low during the address ACK/NACK phase (STALLA bit of I3C_TIMINGR2 set to 1 and STALL[7:0] bitfield of I3C_TIMINGR2 set to a non-null value)
- the address emitted by the controller follows a frame start and not a repeated start

The purpose of this programmed SCL clock stall time is to add an additional duration for the I3C target(s) to respond on the address ACK/NACK phase. However, the SCL clock is not stalled on this address ACK/NACK phase.

Workaround

Set NOARBH = 0 in I3C_CFGR in order to insert the arbitrable header between the frame start and the emitted address.

If the I^2C/I3C target has still not enough time to respond to the emitted static/dynamic address, increase the SCL low duration for any open-drain phase by increasing SCLL_OD[7:0] value in I3C_TIMINGR0.

2.15.3 I3C controller: unexpected first frame with a 0x7F address when the I3C peripheral is enabled

Description

After I3C has been initialized as controller, an unexpected frame is generated when the I3C peripheral is enabled. The issue occurs after the following sequence:

1. I3C is initialized as I3C controller (CRINIT bit is set in I3C_CFGR whereas EN bit is kept cleared in I3C_CFGR).
2. I3C is enabled (EN bit set in I3C_CFGR).

As a result, the I3C controller can incorrectly detect that the SDA line has been driven low by a target, interpret it as a start request, activate the SCL clock, and generate a 0x7F address followed by RNW bit = 1 that is not acknowledged.

This first frame completes without any other impact than this unexpected I3C bus activity.

Workaround

Respect the sequence below during I3C controller initialization:

1. Instead of configuring the alternate GPIO of the SDA line without any pull-up, temporary enable the GPIO pull-up.
2. After a delay of 1 ms, disable GPIO pull-up.
3. Initialize I3C as I3C controller by setting CRINIT in I3C_CFGR whereas EN bit is kept cleared in I3C_CFGR.
4. Enable I3C by setting EN bit in I3C_CFGR.

As a result the I3C controller does not detect SDA low when it is enabled, and no unexpected frame is generated.

2.15.4 I3C controller: no timestamp on IBI acknowledge when timing control is used in Asynchronous mode 0

Description

When I3C acts as controller, it cannot provide a timestamp on an IBI acknowledge (named C_REF in MIPI I3C v1.1 specification).

As a result, when timing control is used in Asynchronous mode 0, the controller software cannot calculate the timestamp of the sampled data of the target(s) following a received and acknowledged IBI using payload data for timing control (T_C1 and T_C2) (see MIPI formula: $\mathrm{C\_TS} = \mathrm{C\_REF} - \mathrm{C\_C2} \times \mathrm{T\_C1}/\mathrm{T\_C2}$), despite the fact that the controller software can compute the duration C_C2 by using the formula:

$$\mathrm{C\_C2} = 9 \times (\mathrm{I3C\_TIMINGR0.SCLL\_PP[7:0]} + 1 + \mathrm{I3C\_TIMINGR0.SCLH\_I3C[7:0]} + 1) \times T_{I3CCLK}$$

When operating in Asynchronous mode 0, the sampled data received from the target(s) cannot be associated with a computed timestamp, and on controller side, they can not be time-correlated.

Workaround

Follow the sequence below:

1. Allocate an available product timer by software and approximate the IBI acknowledge moment by when the timer is notified by an interrupt of a received and complete IBI.
2. Program a broadcast/direct SETXTIME CCC with subcommand byte 0xDF to enter Asynchronous mode 0.
3. After being notified of the command completion by the flag and/or the related interrupt (FCF flag is set in I3C_EVR), reset and enable the timer to start the counter.
4. After being notified that an IBI is complete by the flag and/or the related interrupt (IBIF flag is set in I3C_EVR), read the value of the timer as C_TIM. The timestamp of the sampled data can then be approximated by using the formula:

   $$\mathrm{C\_TS} = \mathrm{C\_TIM} - \mathrm{C\_C2} \times (\mathrm{T\_C1}/\mathrm{T\_C2} + 4)$$

   knowing that

   $$\mathrm{C\_C2} = 9 \times (\mathrm{I3C\_TIMINGR0.SCLL\_PP[7:0]} + 1 + \mathrm{I3C\_TIMINGR0.SCLH\_I3C[7:0]} + 1) \times \mathrm{T}_{I3CCLK}$$

   and that the IBI is complete after a 4-byte payload.
5. Generate a broadcast/direct SETXTIME CCC with subcommand byte 0xFF to exit Asynchronous mode 0 to disable/deallocate the timer resource.

2.16 USART

2.16.1 Wrong data received by SPI slave receiver in autonomous mode with CPOL = 1

Description

The SPI slave receiver device receives wrong data when all the following conditions are met:

- The USART is used in SPI master transmitter mode
- The autonomous mode is used
- The CPOL bit of the USART_CR2 register is set

Workaround

When the autonomous mode is used, do not set the CPOL bit in USART_CR2.

2.16.2 Received data may be corrupted upon clearing the ABREN bit

Description

The USART receiver may miss data or receive corrupted data when the auto baud rate feature is disabled by software (ABREN bit cleared in the USART_CR2 register) after an auto baud rate detection, while a reception is ongoing.

Workaround

Do not clear the ABREN bit.

2.16.3 Noise error flag set while ONEBIT is set

Description

When the ONEBIT bit is set in the USART_CR3 register (one sample bit method is used), the noise error (NE) flag must remain cleared. Instead, this flag is set upon noise detection on the START bit.

Workaround

None.

Note: Having noise on the START bit is contradictory with the fact that the one sample bit method is used in a noise free environment.

2.17 LPUART

2.17.1 Possible LPUART transmitter issue when using low BRR[15:0] value

Description

The LPUART transmitter bit length sequence is not reset between consecutive bytes, which could result in a jitter that cannot be handled by the receiver device. As a result, depending on the receiver device bit sampling sequence, a desynchronization between the LPUART transmitter and the receiver device may occur resulting in data corruption on the receiver side.

This happens when the ratio between the LPUART kernel clock and the baud rate programmed in the LPUART_BRR register (BRR[15:0]) is not an integer, and is in the three to four range. A typical example is when the 32.768 kHz clock is used as kernel clock and the baud rate is equal to 9600 baud, resulting in a ratio of 3.41.

Workaround

Apply one of the following measures:

- On the transmitter side, increase the ratio between the LPUART kernel clock and the baud rate. To do so:
  - Increase the LPUART kernel clock frequency, or
  - Decrease the baud rate.
- On the receiver side, generate the baud rate by using a higher frequency and applying oversampling techniques if supported.

2.18 SPI

2.18.1 RDY output failure at high serial clock frequency

Description

When acting as slave with RDY alternate function enabled through setting the RDIOM bit of the SPI_CFG2 register, the device may fail to indicate its *Not ready* status in time through the RDY output signal to suspend communication. This may then lead to data overrun and/or underrun on the device side. The failure occurs when the serial clock frequency exceeds:

- twice the APB clock frequency, with data sizes from 8 to 15 bits
- six times the APB clock frequency, with data sizes from 16 to 23 bits
- fourteen times the APB clock frequency, with data sizes from 24 to 32 bits

Workaround

None.

2.19 FDCAN

2.19.1 Desynchronization under specific condition with edge filtering enabled

Description

FDCAN may desynchronize and incorrectly receive the first bit of the frame if:

- the edge filtering is enabled (the EFBI bit of the FDCAN_CCCR register is set), and
- the end of the integration phase coincides with a falling edge detected on the FDCAN_Rx input pin

If this occurs, the CRC detects that the first bit of the received frame is incorrect, flags the received frame as faulty and responds with an error frame.

Note: This issue does not affect the reception of standard frames.

Workaround

Disable edge filtering or wait for frame retransmission.

2.19.2 Tx FIFO messages inverted under specific buffer usage and priority setting

Description

Two consecutive messages from the Tx FIFO may be inverted in the transmit sequence if:

- FDCAN uses both a dedicated Tx buffer and a Tx FIFO (the TFQM bit of the FDCAN_TXBC register is cleared), and
- the messages contained in the Tx buffer have a higher internal CAN priority than the messages in the Tx FIFO.

Workaround

Apply one of the following measures:

- Ensure 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 (TFE bit of FDACN_IR set to 1) the next Tx FIFO element is requested.
- Use only a Tx FIFO:
  Send both messages from a Tx FIFO, including the message with the higher priority. This message has to wait until the preceding messages in the Tx FIFO have been sent.
- Use two dedicated Tx buffers (for example, use Tx buffer 4 and 5 instead of the Tx FIFO). The following pseudo-code replaces the function in charge of filling the Tx FIFO:

```
Write message to Tx Buffer 4
Transmit Loop:
    Request Tx Buffer 4 - write AR4 bit in FDCAN_TXBAR
    Write message to Tx Buffer 5
    Wait until transmission of Tx Buffer 4 complete (IR bit in FDCAN_IR),
    read TO4 bit in FDCAN_TXBTO
    Request Tx Buffer 5 - write AR5 bit of FDCAN_TXBAR
    Write message to Tx Buffer 4
    Wait until transmission of Tx Buffer 5 complete (IR bit in FDCAN_IR),
    read TO5 bit in FDCAN_TXBTO
```

2.19.3 DAR mode transmission failure due to lost arbitration

Description

In DAR mode, the transmission may fail due to lost arbitration at the first two identifier bits.

Workaround

Upon failure, clear the corresponding Tx buffer transmission request bit TRPx of the FDCAN_TXBRP register and set the corresponding cancellation finished bit CFx of the FDCAN_TXBCF register, then restart the transmission.

2.20 UCPD

2.20.1 TXHRST upon write data underflow corrupting the CRC of the next packet

Description

TXHRST command issued at the instant of detecting write data underflow during a packet transmission can cause a corrupt CRC of the following packet.

Workaround

Use DMA (TXDMAEN) rather than software writing to UCPD_TXDR. Normally, this prevents write data underflow. Should a corrupt CRC event still occur, the DMA transfer method retransmits the packet until the CRC is correct and the packet acknowledged by the receiver.

2.20.2 Ordered set with multiple errors in a single K-code is reported as invalid

Description

The Power Delivery standard allows considering a received ordered set as valid even if it contains errors, provided that they only affect a single K-code of the ordered set.

In the reference manual, the RXSOP3OF4 flag is specified to signal errors affecting a single K-code, the RXERR flag to signal errors in multiple K-codes.

However, the behaviour does not conform with the reference manual. The RXSOP3OF4 flag is only raised in the case of a single error. The RXERR flag is raised in the case of multiple errors, regardless of whether they affect a single K-code or multiple K-codes. As a consequence, ordered sets with multiple errors in a single K-code are reported by the device as invalid although the Power Delivery standard allows considering them as valid.

Despite this non-conformity versus its reference manual, the device remains compliant with the Power Delivery standard.

Workaround

None.

2.20.3 UCPDPHY specification marginality for ZDRIVER

Description

ZDRIVER value could be above the maximum limit of USB standard requirements at high junction temperatures (T_J).

Note: The USB certifications are not affected by this limitation since tests are conducted under typical operating conditions.

Workaround

ZDRIVER value is ensured to be below 100 Ω when T_J is kept below 90 °C. Within this conditions, no issues have been detected during thorough UCPDPHY characterization and functional tests.

2.21 ETH

2.21.1 Incorrect gate control list switching for intermediate cycles when CTR is less than the GCL execution time

Description

The EST (enhancements to scheduled traffic) scheduler switches to the next gate control list (GCL) after executing the current GCL, regardless of the difference between the start time of the next GCL and the time when the current GCL rows are completely executed.

If the GCL execution takes longer than the cycle time, the GCL is truncated at the cycle time, and the subsequent loop begins at BTR + N × cycle time, where N is an integer that represents the iteration number.

However, the GCL incorrectly updates the internal BTR twice, and skips the execution of the next GCL loop. This issue arises if one of the following conditions is met:

- CTR value < sum of time intervals of fully executed GCL rows, or
- CTR value > sum of time intervals of fully executed GCL rows, and
  CTR value < sum of time intervals of fully executed GCL rows + 8 PTP clock periods

Note: The CTR is the value initially configured by the software.

Workaround

Apply one of the following measures:

- CTR value > sum of time intervals of fully executed GCL rows + 8 PTP clock periods
- CTR = sum of time intervals of fully executed GCL rows

2.22 ETHSW

2.22.1 Reported bridge delays do not match measured values

Description

Maximum input-to-gate and gate-to-output delay values are incorrect when traffic exceeds 90% line rate. This is also true for bursts where the line rate exceeds 90% during bursts.

Maximum input-to-gate and gate-to-output delay values can be read from the following registers:

- ETHSW_FES_I_TO_G_MAX_10R
- ETHSW_FES_I_TO_G_MAX_100R
- ETHSW_FES_I_TO_G_MAX_1000R
- ETHSW_FES_G_TO_O_MAX_10R
- ETHSW_FES_G_TO_O_MAX_100R
- ETHSW_FES_G_TO_O_MAX_1000R

Workaround

Make sure the line rate does not exceed 90%.

2.22.2 Tagged link local frames are discarded if the port is not part of the VLAN

Description

An incoming link local frame from an external port is discarded if it has a VLAN tag and the VID in the VLAN tag is not enabled on the respective port.

Workaround

None. VLAN tagging should not be used for the link local frames.

2.22.3 TSN switch controller tags management traffic when the PVID is configured differently on internal and external ports

Description

If the "Port Default VLAN" (PVID) is configured differently on the internal and external ports, management traffic gets tagged on the external port too. This affects all management protocols (such as PTP, MSTP, and LLDP) and they may stop operating.

Workaround

Make sure that the configuration of the PVID is exactly the same for all switch ports.

2.22.4 Express/preemptable selection must be per priority, not per queue (traffic class)

Description

According to the IEEE 802.1Qbu standard, the express/preemptable selection must be per priority, but for the TSN switch controller the express/preemptable selection is per queue (traffic class).

Workaround

Ensure that the express and preemptable frames use different queues (traffic classes).

2.22.5 Priority queue drop is not working as specified

Description

The specification says "the oldest frame in the priority queue is dropped to make room for the new frame". The switch may drop the newest frame instead if the queue is full.

Workaround

None. Specification updated to match the design.

2.22.6 Higher priority frame may overtake lower priority frame

Description

It is possible that in some full-bandwidth corner cases a higher priority frame overtakes a lower priority frame before arriving in data memory. This may cause a change in frame order.

Workaround

None.

Important security notice

The STMicroelectronics group of companies (ST) places a high value on product security, which is why the ST product(s) identified in this documentation may be certified by various security certification bodies and/or may implement our own security measures as set forth herein. However, no level of security certification and/or built-in security measures can guarantee that ST products are resistant to all forms of attacks. As such, it is the responsibility of each of ST's customers to determine if the level of security provided in an ST product meets the customer needs both in relation to the ST product alone, as well as when combined with other components and/or software for the customer end product or application. In particular, take note that:

- ST products may have been certified by one or more security certification bodies, such as Platform Security Architecture (www.psacertified.org) and/or Security Evaluation standard for IoT Platforms (www.trustcb.com). For details concerning whether the ST product(s) referenced herein have received security certification along with the level and current status of such certification, either visit the relevant certification standards website or go to the relevant product page on www.st.com for the most up to date information. As the status and/or level of security certification for an ST product can change from time to time, customers should re-check security certification status/level as needed. If an ST product is not shown to be certified under a particular security standard, customers should not assume it is certified.
- Certification bodies have the right to evaluate, grant and revoke security certification in relation to ST products. These certification bodies are therefore independently responsible for granting or revoking security certification for an ST product, and ST does not take any responsibility for mistakes, evaluations, assessments, testing, or other activity carried out by the certification body with respect to any ST product.
- Industry-based cryptographic algorithms (such as AES, DES, or MD5) and other open standard technologies which may be used in conjunction with an ST product are based on standards which were not developed by ST. ST does not take responsibility for any flaws in such cryptographic algorithms or open technologies or for any methods which have been or may be developed to bypass, decrypt or crack such algorithms or technologies.
- While robust security testing may be done, no level of certification can absolutely guarantee protections against all attacks, including, for example, against advanced attacks which have not been tested for, against new or unidentified forms of attack, or against any form of attack when using an ST product outside of its specification or intended use, or in conjunction with other components or software which are used by customer to create their end product or application. ST is not responsible for resistance against such attacks. As such, regardless of the incorporated security features and/or any information or support that may be provided by ST, each customer is solely responsible for determining if the level of attacks tested for meets their needs, both in relation to the ST product alone and when incorporated into a customer end product or application.
- All security features of ST products (inclusive of any hardware, software, documentation, and the like), including but not limited to any enhanced security features added by ST, are provided on an "AS IS" BASIS. AS SUCH, TO THE EXTENT PERMITTED BY APPLICABLE LAW, ST DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE, unless the applicable written and signed contract terms specifically provide otherwise.

Revision history

Table 6. **Document revision history**

Date	Version	Changes
22-Mar-2024	1	Initial release.
27-Jun-2024	2	Added errata in Summary of device limitations and Summary of device limitations: • LSEDRV description is swapped • Boot fails after wakeup from STANDBY when booting on SNOR, SNAND and HYPERFLASH • Boot ROM hangs when system reset is applied while D1 domain is in DStandby state • Boot ROM writes in SYSRAM during LPLV-Stop2 wakeup • Instruction fetch access to PWR register lead to unwanted write • CPU2 (Cortex-M33) does not support debug in non-secure only • LSE function can be impacted if there is negative current injection in GPIOs • ETHx kernel clock is gated if ETHxMACEN register bit is not set • RISAF wrong SIDR value • STOP and Standby entry failed when DDR is in shared mode • Variable latency is not supported when a refresh collision occurs during a write access to some OctaRAM™ memories • Reported bridge delays do not match measured values • Tagged link local frames are discarded if the port is not part of the VLAN • TSN switch controller tags management traffic when the PVID is configured differently on internal and external ports • Express/preemptable selection must be per priority, not per queue (traffic class) • Priority queue drop is not working as specified • Higher priority frame may overtake lower priority frame Updated status of ARRM and CMPM flags are not set when APB clock is slower than kernel clock in Summary of device limitations.
30-Oct-2024	3	Introduced STM32MP23xx microprocessors. Added errata in Summary of device limitations: • Cortex-A35 not restarted after system reset in TDCID Cortex-M33 configuration • AHB RISAB3/4/5 illegal access due to ghost CID0 detection • ETM timestamp is exported with zero value • UCPDPHY specification marginality for ZDRIVER

Contents