STM32H5Exxx/5Fxxx device errata

Applicability

This document applies to the part numbers of STM32H5Exxx/5Fxxx 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 RM0517.

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
STM32H5E4xx	STM32H5E4AJ, STM32H5E4AK, STM32H5E4IJ, STM32H5E4IK, STM32H5E4VJ, STM32H5E4VK, STM32H5E4ZJ, STM32H5E4ZK
STM32H5E5xx	STM32H5E5IJ, STM32H5E5IK, STM32H5E5LJ, STM32H5E5LK, STM32H5E5VJ, STM32H5E5VK, STM32H5E5ZJ, STM32H5E5ZK
STM32H5F4xx	STM32H5F4AJ, STM32H5F4IJ, STM32H5F4VJ, STM32H5F4ZJ
STM32H5F5xx	STM32H5F5LJ, STM32H5F5IJ, STM32H5F5VJ, STM32H5F5ZJ

Table 2. Device variants

Reference	Silicon revision codes	
	Device marking(1)	REV_ID(2)
STM32H5Exxx/5Fxxx	A	0x1000
	Z	0x1001

1. *Refer to the device datasheet for how to identify this code on different types of package.*
2. *REV_ID[15:0] bitfield of register.*

1 Summary of device errata

The following table gives a quick reference to the STM32H5Exxx/5Fxxx device limitations and their status:

A Limitation applicable. Workaround available

N Limitation applicable. No workaround available

P Limitation applicable. Partial workaround available

- Limitation not applicable.

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. A	Rev. Z
Core	2.1.1	Access permission faults are prioritized over unaligned device memory faults	N	N
System	2.2.1	LSE crystal oscillator may be disturbed by transitions on PC13	N	N
	2.2.2	SRAMx_RST option bits have an immediate effect	A	A
	2.2.3	Full JTAG configuration without NJTRST pin cannot be used	A	A
	2.2.4	Incorrect backup domain reset	A	A
	2.2.5	Reset vector catch feature cannot be used when debugging is disabled for secure code	A	A
	2.2.6	Debug not available when the PRODUCT_STATE is iROT-Provisioned	A	A
	2.2.7	Clearing WWDG_SW might result in debug authentication failure	A	A
	2.2.8	Invalid DAC output voltage for several DAC kernel clocks	A	A
	2.2.9	ADC, COMP, OPAMP inputs voltage limited to VDD + 0.3 V	A	A
	2.2.10	CPU execution freeze upon first erase or program operation after power-on or wake-up from Standby	A	A
	2.2.11	V_{DDCORE} voltage in VOS0 and VOS1 is higher than specified in RM0517	N	-
	2.2.12	Inaccurate ECC error address reporting for SRAM2	A	A
	2.2.13	FDCAN TrustZone® and privilege protection misalignment with memory boundaries	A	A
	2.2.14	PWR_CSLEEP and PWR_CSTOP alternate functions are not operational	N	N
FMC	2.3.1	Dummy read cycles inserted when reading synchronous memories	N	N
	2.3.2	Wrong data read from a busy NAND memory	A	A
OCTOSPI	2.4.1	Deadlock can occur under certain conditions	A	A
	2.4.2	Transactions are limited to 8 Mbytes in OctaRAM™ memories	N	N
	2.4.3	Variable latency is not supported when a refresh collision occurs during a write access to some OctaRAM™ memories	P	P
	2.4.4	Octo-SPI memory data failure when using HCLK as kernel clock	A	A
SDMMC	2.5.1	Command response and receive data end bits not checked	N	N
ADC	2.6.1	New context conversion initiated without waiting for trigger when writing new context in ADC_JSQR with JQDIS = 0 and JQM = 0	A	A

Function	Section	Limitation	Status	
			Rev. A	Rev. Z
ADC	2.6.2	Two consecutive context conversions fail when writing new context in ADC_JSQR just after previous context completion with JQDIS = 0 and JQM = 0	A	A
	2.6.3	Unexpected regular conversion when two consecutive injected conversions are performed in Dual interleaved mode	A	A
	2.6.4	ADC_AWDy_OUT reset by non-guarded channels	A	A
	2.6.5	Injected data stored in the wrong ADC_JDRx registers	A	A
	2.6.6	ADC slave data may be shifted in Dual regular simultaneous mode	A	A
PSSI	2.7.1	Output mode not usable with both PSSI_RDY and PSSI_DE signals enabled	N	N
RNG	2.8.1	RNG may report continuous seed errors in specific environmental conditions	N	N
TIM	2.9.1	Unexpected PWM output when using ocref_clr	N	N
LPTIM	2.10.1	Device may remain stuck in LPTIM interrupt when entering Stop mode	A	A
	2.10.2	ARRM and CMPM flags are not set when APB clock is slower than kernel clock	A	A
	2.10.3	Interrupt status flag is cleared by hardware upon writing its corresponding bit in LPTIM_DIER register	N	N
RTC and TAMP	2.11.1	Alarm flag may be repeatedly set when the core is stopped in debug	N	N
	2.11.2	RTC calendar read may return transient incorrect value when BYPSHAD = 0	A	A
	2.11.3	Timestamp flag unexpectedly raised when disabling timestamp	A	A
I2C	2.12.1	Wrong data sampling when data setup time ($t_{SU;DAT}$) is shorter than one I2C kernel clock period	P	P
	2.12.2	Spurious bus error detection in controller mode	A	A
USART	2.13.1	Received data may be corrupted upon clearing the ABREN bit	A	A
	2.13.2	Noise error flag set while ONEBIT is set	N	N
LPUART	2.14.1	Possible LPUART transmitter issue when using low BRR[15:0] value	P	P
SPI	2.15.1	RDY output failure at high serial clock frequency	N	N
FDCAN	2.16.1	Desynchronization under specific condition with edge filtering enabled	A	A
	2.16.2	Tx FIFO messages inverted under specific buffer usage and priority setting	A	A
OTG_FS	2.17.1	Potential unexpected transfer on the USB bus instead of a zero-length packet	A	A
OTG_HS	2.18.1	Potential unexpected transfer on the USB bus instead of a zero-length packet	A	A
	2.18.2	False detection of chirp-K when a FS device is connected	A	A
UCPD	2.19.1	Ordered set with multiple errors in a single K-code is reported as invalid	N	N
CEC	2.20.1	Missed CEC messages in normal receiving mode	A	A
	2.20.2	Unexpected TXERR flag during a message transmission	A	A

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

Table 4. Summary of device documentation errata

Function	Section	Documentation erratum
FMC	2.3.3	SDRAM timing discrepancies

2 Description of device errata

The following sections describe the errata of the applicable devices and provide a workaround where available. They are grouped by device functions.

The applicable devices are based on Arm® Cortex® core.

arm

Note: Arm and Cortex are registered trademarks of Arm Limited (or its subsidiaries or affiliates) in the US and/or elsewhere.

The Arm word and logo are trademarks of Arm Limited (or its subsidiaries) in the US and/or elsewhere. All rights reserved.

2.1 Core

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

2.1.1 Access permission faults are prioritized over unaligned device memory faults

Description

A load or store which causes an unaligned access to device memory results 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 is 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 is 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.

2.2 System

2.2.1 LSE crystal oscillator may be disturbed by transitions on PC13

Description

On LQFP packages, the LSE crystal oscillator clock frequency can be incorrect when PC13 is toggling in input or output (for example when used for RTC_OUT1).

The external clock input (LSE bypass) is not impacted by this limitation.

The WLCSP and UFBGA packages are not impacted by this limitation.

Workaround

Avoid toggling PC13 when LSE is used on LQFP packages.

2.2.2 SRAMx_RST option bits have an immediate effect

Description

Clearing the SRAMx_RST option bit immediately triggers an SRAMx erase operation. This might lead to a CPU exception or unpredictable behavior if the erased SRAMx is being used by the application.

Note: This reported SRAM erase is performed only once after system reset, and only if the SRAMxRST option bit is set to 1 during system reset.

Workaround

The application must be reset immediately after SRAMx_RST clear.

2.2.3 Full JTAG configuration without NJTRST pin cannot be used

Description

When using the JTAG debug port in Debug mode, the connection with the debugger is lost if the NJTRST pin (PB4) is used as a GPIO or for an alternate function other than NJTRST. Only the 4-wire JTAG port configuration is impacted.

Workaround

Use the SWD debug port instead of the full 4-wire JTAG port.

2.2.4 Incorrect backup domain reset

Description

The backup domain reset may be missed upon a backup domain power-on following a V_{BAT} power-off in VBAT mode, if the V_{BAT} voltage drops during the power-off phase hitting a window, which is a few mV wide before it starts to rise again. This window is located in the range between 100 mV and 700 mV, the exact position depending mainly on the device and on the temperature.

The missed reset results in unpredictable values of the backup domain registers. This may lead to raising an unexpected tamper event preventing the access to SRAM2 and PKA, or influencing any of the backup domain functions.

Workaround

Apply one of the following measures to avoid an incorrect backup domain reset:

- Before performing a new power-on, let the V_{BAT} supply voltage fall to a level below 100 mV for more than 200 ms.
- If none of the previous workarounds can be applied, and the boot follows a backup domain power-on reset, erase the backup domain by software. In order to discriminate the backup domain power-on reset from a power-on reset, at least one backup register (called, for example, BackupTestRegister) must be previously programmed with a BKP_REG_VAL value containing 16 bits set and 16 bits cleared. The robustness of this workaround can be significantly improved by using a CRC rather than registers, since the registers are subject to backup domain reset.
  The workaround consists in calculating the CRC of the backup domain registers, RCC_BDCR and RTC/TAMP registers, excluding the bits modified by hardware. The CRC result can be stored in the backup register, instead of a fixed BKP_REG_VAL value. The CRC result needs to be updated for each modification of values covered by the CRC, for example when the CRC peripheral is used.
  Insert the following software sequence at the very beginning of the boot code:
  1. Check if the BORRSTF flag of the RCC_RSR register is set (the reset is caused by a power-on).
  2. If it is set, check that the BackupTestRegister content is different from BKP_REG_VAL, or that the new CRC calculated value is different from stored results, depending on the chosen workaround implementation.
  3. If this is the case and if no tamper flag is set (when the tamper detection is enabled), the reset is caused by a backup domain power-on. Then apply the following sequence:
     a. Enable backup domain access by setting the DBP bit of the PWR_DBPCR register.
     b. Reset the backup domain by applying the following sequence:
        i. Write 0x0001 0000 to the RCC_BDCR register, which sets the VSWRST bit and clears the other register bits that may not be cleared.
        ii. Read the RCC_BDCR register to make the reset time long enough.
        iii. Write 0x0000 0000 to the RCC_BDCR register to clear the VSWRST bit.
     c. Clear the BORRSTF flag by setting the RMVF bit of the RCC_RSR register.

2.2.5 Reset vector catch feature cannot be used when debugging is disabled for secure code

Description

If the product state (PRODUCT_STATE[7:0] bitfield of the FLASH_OPTSR_CUR register) is different from the open state, debugging is not allowed except by resetting the CPU by mean of a reset vector catch. This may lead to the CPU not halting before executing the first instruction of the nonsecure exception handler.

Consequently, nonsecure code execution may not stop after switching to the nonsecure state, and the CPU may proceed executing code in the nonsecure area.

Workaround

To halt the CPU in the reset handler, insert a software break point in the first nonsecure instruction of the user application. The address of the reset handler can be read from the NSBOOTADD[23:8] bitfield of the FLASH_NSBOOTR_CUR register.

2.2.6 Debug not available when the PRODUCT_STATE is iROT-Provisioned

Description

When the PRODUCT_STATE is iROT-Provisioned, the debug must be available for a code executed from an HDPL 3 area. However, in this case, the code cannot be debugged.

Workaround

If PRODUCT_STATE = iROT-Provisioned, force the debug opening in the first stage of the boot sequence software.

Below an example of code:

```
/* This code ensures that in PRODUCT_STATE = IROT_PROVISIONED, the Debug is opened for HDPL3
*/
 /* This code must be integrated in a HDPL1 code */
if (READ_BIT(FLASH->OPTSR_CUR, FLASH_OPTSR_PRODUCT_STATE_Msk) == OB_PROD_STATE_IROT_PROVISION
ED)
 {
 __HAL_RCC_SBS_CLK_ENABLE();
 ((SBS_TypeDef *)SBS_BASE)->DBGCR = 0x006FB4B4U; // 6F value to open debug when HDPL3 is reac
hed
 }
 /* To be able to attach the debugger, the code to debug must be executed in HDPL3. */
 /* The code must call the below function twice before reaching the code to debug: */
 /* HAL_SBS_IncrementHDPLValue(); */
```

2.2.7 Clearing WWDG_SW might result in debug authentication failure

Description

If the WWDG_SW configuration option bit is cleared (window watchdog controlled by hardware), the WWDG is always enabled after a reset, and cannot be disabled. As a result:

- The ST-DA debug authentication sequence might fail, preventing any possibility to securely launch regressions on secure products.

Workaround

Do not enable the *"WWDG controlled by hardware"* configuration. Instead, use the *"WWDG controlled by software"* configuration (WWDG_SW configuration option bit set).

2.2.8 Invalid DAC output voltage for several DAC kernel clocks

Description

The DAC output voltage might be incorrect when the DAC kernel clock is different from rcc_hclk (ADCDACSEL[2:0] bits equal to 000 in RCC_CCIPR5 register) or sys_ck (ADCDACSEL[2:0] bits equal to 001 in RCC_CCIPR5 register).

Workaround

When DAC is used, use ADCDACSEL[2:0] = 000 or 001 in the RCC_CCIPR5 register.

2.2.9 ADC, COMP, OPAMP inputs voltage limited to VDD + 0.3 V

Description

The ADC, COMP, and OPAMP inputs listed below are clamped to VDD instead of VDDA. This limits the input voltage to VDD + 0.3 V.

The impacted analog peripheral inputs are:

- ADC1: ADC1_INP18, ADC1_INN18, ADC1_INP19
- ADC2: ADC2_INP18, ADC2_INN18, ADC2_INP19
- ADC3: ADC3_INP18, ADC3_INP19
- COMP1_INM: COMP1_INM4 (dac1_out2), COMP1_INM5 (dac1_out1), and COMP1_INM7 (PA6)
- COMP2_INM: COMP2_INM4 (dac1_out2), COMP2_INM5 (dac1_out1)
- COMP1_INP2 (dac1_out1)
- COMP2_INP2 (dac1_out2)
- OPAMP: OPAMP1_INP1 (dac1_out1), OPAMP1_INP3 (PE12)

Workaround

Apply one of the following measures:

- Use VDDA lower than VDD + 0.3 V.
- Use other ADC, COMP, and OPAMP inputs.

2.2.10 CPU execution freeze upon first erase or program operation after power-on or wake-up from Standby

Description

The first flash memory erase or program operation performed after power-on sequence or after wake-up from Standby makes the flash memory inaccessible for about 240 µs. Any CPU fetch or read access to the flash memory during this period freezes the CPU execution until the flash memory becomes accessible again.

This also concerns the read-while-write operation where the code executes in one bank, and it writes and reads another bank of the flash memory.

Note: Any subsequent flash memory erase or program operation is free of this issue.

Workaround

If the application is sensitive to this limitation, apply the following measure:

1. Relocate the interrupt vector table and critical interrupt handlers to the SRAM.
2. Execute the code for the first flash memory erase or program operation in the SRAM.

Ensure that the code executed in the SRAM does not access the flash memory during the period of its inaccessibility.

Once the first erase or program operation is completed, the software can resume execution in the flash memory.

2.2.11 V_{DDCORE} voltage in VOS0 and VOS1 is higher than specified in RM0517

Description

The V_{DDCORE} voltage is higher than specified in RM0517 for all voltage scaling (VOS) levels (VOS0/1/2/3 and SVOS3/4/5). The voltage shift can be up to 40 mV.

This deviation results in slightly higher current consumption in Run, SLEEP and Stop modes, while Standby and VBAT modes are not impacted.

Workaround

None.

2.2.12 Inaccurate ECC error address reporting for SRAM2

Description

In the RAMCFG_M2SEAR (single ECC error) and RAMCFG_M2DEAR (double ECC error) registers of SRAM2, the ESEA and EDEA bitfields are incorrectly composed of [13:0] bits instead of [14:0]. As a result, in the event of a single or double ECC error, the address of the error is incomplete, missing the bit 14. This bit is always read as 0.

Workaround

To accurately determine the failing address when an ECC error is detected, check the two possible fail addresses following these steps:

1. Read data from SRAM2 at the reported address with the bit 14 at 0b0, and check the ECC status.
2. Read data from SRAM2 at the alternate address with the bit 14 at 0b1, and check the ECC status.
3. The address with the ECC error status is the failing address.

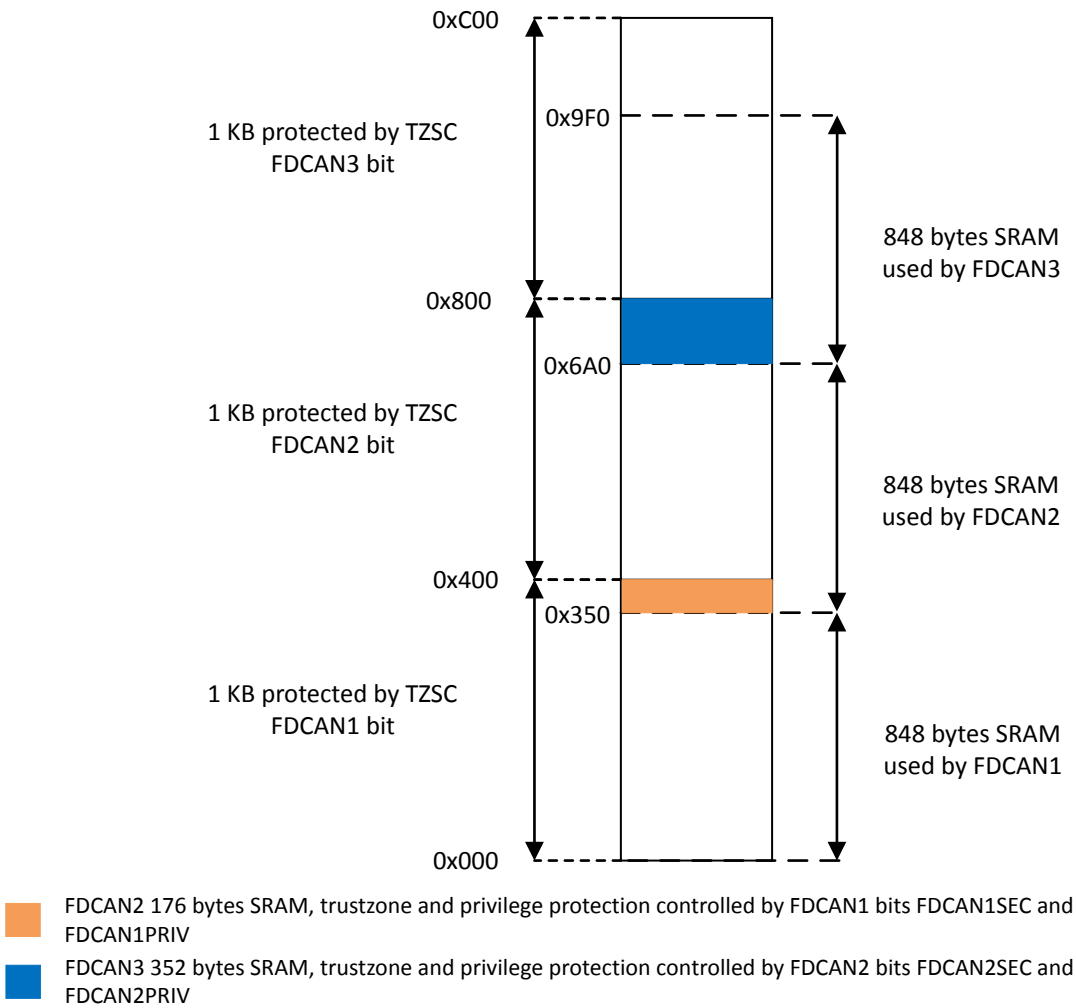
2.2.13 FDCAN TrustZone® and privilege protection misalignment with memory boundaries

Description

The FDCAN instances (FDCAN1, FDCAN2, and FDCAN3) use contiguous SRAM regions. However, since each instance uses 848 bytes, their memory regions do not align with the 1 KB boundaries required by TrustZone® and privilege protection. This misalignment does not impact FDCAN1 SRAM protection but has the following implications:

- **FDCAN2:** The lower part of FDCAN2 RAM (176 bytes in orange) is controlled by FDCAN1 TrustZone/ privilege settings (bits FDCAN1SEC and FDCAN1PRIV).
  - The lower SRAM might become inaccessible if FDCAN1 is configured as secure (or privileged) and FDCAN2 as nonsecure (or nonprivileged).
  - Conversely, if FDCAN1 is nonsecure (or nonprivileged) and FDCAN2 is secure (or privileged), the lower part of FDCAN2 memory might still be accessible, compromising the security of FDCAN2.
- **FDCAN3:** The lower part of FDCAN3 RAM (352 bytes in blue) is controlled by FDCAN2 TrustZone/ privilege settings (bits FDCAN2SEC and FDCAN2PRIV).
  - The lower SRAM might become inaccessible if FDCAN2 is configured as secure (or privileged) and FDCAN3 as nonsecure (or nonprivileged).
  - Conversely, if FDCAN2 is nonsecure (or nonprivileged) and FDCAN3 is secure (or privileged), the lower part of FDCAN3 memory might still be accessible, compromising the security of FDCAN3.

Figure 1. FDCAN instances



Workaround

Configure FDCAN1, FDCAN2, and FDCAN3 to have the same security and/or privilege level. This ensures that the TrustZone® and/or privilege protection is uniformly applied across all FDCAN instances, preventing any unintended access or restriction.

2.2.14 PWR_CSLEEP and PWR_CSTOP alternate functions are not operational

Description

The alternate functions PWR_CSLEEP and PWR_CSTOP, mapped to pins PC2 and PC3 respectively, are not operational.

Workaround

None.

2.3 FMC

2.3.1 Dummy read cycles inserted when reading synchronous memories

Description

When performing a burst read access from a synchronous memory, two dummy read accesses are performed at the end of the burst cycle whatever the type of burst access.

The extra data values read are not used by the FMC and there is no functional failure.

Workaround

None.

2.3.2 Wrong data read from a busy NAND memory

Description

When a read command is issued to the NAND memory, the R/B signal gets activated upon the de-assertion of the chip select. If a read transaction is pending, the NAND controller might not detect the R/B signal (connected to NWAIT) previously asserted and sample a wrong data. This problem occurs only when the MEMSET timing is configured to 0x00 or when ATTHOLD timing is configured to 0x00 or 0x01.

Workaround

Either configure MEMSET timing to a value greater than 0x00 or ATTHOLD timing to a value greater than 0x01.

2.3.3 SDRAM timing discrepancies

Description

The following delays are longer than expected:

- The delay between a Load mode register command and an Activate command is two cycles longer than the value programmed in the TMRD[3:0] bitfield of the FMC_SDTRx register. For example, it equals three clock cycles for TMRD[3:0] = 0000.
- The exit from Self-refresh delay, the row cycle delay, the recovery delay, the row precharge delay, and the self-refresh time, are one clock cycle longer than the values programmed in the TXSR[3:0], TRC[3:0], TWR[3:0], TRP[3:0], and TRAS[3:0] bitfields of the FMC_SDTRx register.

This is a documentation error rather than a device limitation.

Workaround

None.

2.4 OCTOSPI

2.4.1 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.

2.4.2 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

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

2.4.3 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.

2.4.4 Octo-SPI memory data failure when using HCLK as kernel clock

Description

When the HCLK clock (rcc_hclk4) frequency is lower than the SYSCLK clock frequency, the HCLK clock duty cycle is not 50 %. In this condition, selecting HCLK as Octo-SPI kernel clock may lead to memory data failure.

Note: The HCLK clock is prescaled by a ratio controlled with the HPRE[3:0] bitfield of the RCC_CFGR2 register. Any value exceeding 0b0111 makes the HCLK frequency lower than the SYSCLK frequency.

The Octo-SPI kernel clock is selected with the OCTOSPI1SEL[1:0] bitfield of the RCC_CCIPR4 register.

Workaround

When the HCLK and SYSCLK frequencies differ, do not select rcc_hclk4 as Octo-SPI kernel clock. Instead, select another clock such as pll1_q_ck or pll2_r_ck.

2.5 SDMMC

2.5.1 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.

2.6 ADC

2.6.1 New context conversion initiated without waiting for trigger when writing new context in ADC_JSQR with JQDIS = 0 and JQM = 0

Description

Once an injected conversion sequence is complete, the queue is consumed and the context changes according to the new ADC_JSQR parameters stored in the queue. This new context is applied for the next injected sequence of conversions.

However, the programming of the new context in ADC_JSQR (change of injected trigger selection and/or trigger polarity) may launch the execution of this context without waiting for the trigger if:

- the queue of context is enabled (JQDIS cleared to 0 in ADC_CFGR), and
- the queue is never empty (JQM cleared to 0 in ADC_CFGR), and
- the injected conversion sequence is complete and no conversion from previous context is ongoing

Workaround

Apply one of the following measures:

- Ignore the first conversion.
- Use a queue of context with JQM = 1.
- Use a queue of context with JQM = 0, only change the conversion sequence but never the trigger selection and the polarity.

2.6.2 Two consecutive context conversions fail when writing new context in ADC_JSQR just after previous context completion with JQDIS = 0 and JQM = 0

Description

When an injected conversion sequence is complete and the queue is consumed, writing a new context in ADC_JSQR just after the completion of the previous context and with a length longer that the previous context, may cause both contexts to fail. The two contexts are considered as one single context. As an example, if the first context contains element 1 and the second context elements 2 and 3, the first context is consumed followed by elements 2 and 3 and element 1 is not executed.

This issue may happen if:

- the queue of context is enabled (JQDIS cleared to 0 in ADC_CFGR), and
- the queue is never empty (JQM cleared to 0 in ADC_CFGR), and
- the length of the new context is longer than the previous one

Workaround

If possible, synchronize the writing of the new context with the reception of the new trigger.

2.6.3 Unexpected regular conversion when two consecutive injected conversions are performed in Dual interleaved mode

Description

In Dual ADC mode, an unexpected regular conversion may start at the end of the second injected conversion without a regular trigger being received, if the second injected conversion starts exactly at the same time than the end of the first injected conversion. This issue may happen in the following conditions:

- two consecutive injected conversions performed in Interleaved simultaneous mode (DUAL[4:0] of ADC_CCR = 0b00011), or
- two consecutive injected conversions from master or slave ADC performed in Interleaved mode (DUAL[4:0]of ADC_CCR = 0b00111)

Workaround

- In Interleaved simultaneous injected mode: make sure the time between two injected conversion triggers is longer than the injected conversion time.
- In Interleaved only mode: perform injected conversions from one single ADC (master or slave), making sure the time between two injected triggers is longer than the injected conversion time.

2.6.4 ADC_AWDy_OUT reset by non-guarded channels

Description

ADC_AWDy_OUT is set when a guarded conversion of a regular or injected channel is outside the programmed thresholds. It is reset after the end of the next guarded conversion that is inside the programmed thresholds.

However, the ADC_AWDy_OUT signal is also reset at the end of conversion of non-guarded channels, both regular and injected.

Workaround

When ADC_AWDy_OUT is enabled, it is recommended to use only the ADC channels that are guarded by a watchdog.

If ADC_AWDy_OUT is used with ADC channels that are not guarded by a watchdog, take only ADC_AWDy_OUT rising edge into account.

2.6.5 Injected data stored in the wrong ADC_JDRx registers

Description

When the AHB clock frequency is higher than the ADC clock frequency after the prescaler is applied (ratio > 10), if a JADSTP command is issued to stop the injected conversion (JADSTP bit set to 1 in ADC_CR register) at the end of an injected conversion, exactly when the data are available, then the injected data are stored in ADC_JDR1 register instead of ADC_JDR2/3/4 registers.

Workaround

Before setting JADSTP bit, check that the JEOS flag is set in ADC_ISR register (end of injected channel sequence).

2.6.6 ADC slave data may be shifted in Dual regular simultaneous mode

Description

In Dual regular simultaneous mode, ADC slave data may be shifted when all the following conditions are met:

- A read operation is performed by one DMA channel,
- OVRMOD = 0 in ADC_CFGR register (Overrrun mode enabled).

Workaround

Apply one of the following measures:

- Set OVRMOD = 1 in ADC_CFGR. This disables ADC_DR register FIFO.
- Use two DMA channels to read data: one for slave and one for master.

2.7 PSSI

2.7.1 Output mode not usable with both PSSI_RDY and PSSI_DE signals enabled

Description

In output mode, when both the PSSI_RDY and PSSI_DE signals are enabled (DERDYCFG[2:0] bitfield set to 011, 100, or 111), the PSSI_DE signal may fail to indicate data validity.

As a result, output mode cannot be used when both PSSI_RDY and PSSI_DE signals are enabled.

Workaround

None.

2.8 RNG

2.8.1 RNG may report continuous seed errors in specific environmental conditions

Description

Under certain combinations of supply voltage, temperature, and process variation, one of the RNG analog noise sources may provide insufficient entropy. As a consequence, the RNG generates seed errors continuously.

If the application correctly implements the error handling procedure described in the “Error Management” section of the RNG chapter, it can reliably detect this condition and recover from continuous seed error generation.

Workaround

No application workaround is required or applicable as it is handled by the standard error management procedure.

2.9 TIM

2.9.1 Unexpected PWM output when using ocref_clr

Description

In combined PWM mode 1, asymmetric PWM mode 1, or asymmetric PWM mode 2, using ocref_clr can cause the tim_ocxrefc output to be unexpectedly re-enabled or disabled. This behavior depends on the timing of when ocref_clr is activated and deactivated.

Workaround

None.

2.10 LPTIM

2.10.1 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.

2.10.2 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
- The repetition counter is set with REP[7:0] = 0x3 (odd value)

The above example is subject to issues, 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 the LPTIM_RCR register (=3, odd) to assess the risk of issue.

2.10.3 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.

2.11 RTC and TAMP

2.11.1 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.

2.11.2 RTC calendar read may return transient incorrect value when BYPSHAD = 0

Description

When the BYPSHAD control bit in the RTC_CR register is cleared, reading the RTC calendar registers (RTC_SSR, RTC_TR, and RTC_DR) may seldom produce incorrect values. This issue occurs because internal timing can cause the asynchronous RTC calendar registers to be copied into their shadow registers during a transition of the asynchronous registers.

Since this copying process occurs every RTC kernel clock cycle, any erroneous value persists for only one RTC kernel clock cycle. The likelihood of this failure is very low and depends on several factors, including:

- RTC software configuration, such as the RTC kernel clock source, asynchronous prescaler factor, and calibration settings.
- APB clock frequency.
- Operating voltage and temperature.
- The timing of the register read operation.

Additionally, process variations may cause timing differences between samples, which can also influence the probability of this failure.

Workaround

To avoid this condition and ensure consistent calendar readings:

- Set the BYPSHAD control bit to 1 in the RTC_CR register to bypass shadow registers.
- Read the RTC calendar registers twice.
- Compare the two readings. If the values differ, repeat the read and compare sequence until two consecutive readings match.

This software method ensures a reliable calendar value under all operating conditions. For further details, see section *Reading the calendar – When the BYPSHAD control bit is set in the RTC_CR register (bypass shadow registers)* in the reference manual.

2.11.3 Timestamp flag unexpectedly raised when disabling timestamp

Description

The TSF flag of RTC_SR is wrongly set when disabling the timestamp. This issue occurs when the following conditions are met:

- timestamp negative edge detection is requested, and
- no edge has occurred

The other detection configurations are not impacted.

Workaround

After clearing the TSE bit of the RTC_CR register:

- in polling mode, wait for 7 ms to allow the TSF flag to be set. Then clear it.
- in interrupt mode: clear the TSF flag as soon as it is set.

2.12 I2C

2.12.1 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 I^2C-bus SDA line when $t_{SU;DAT}$ is smaller than one I2C kernel clock (I^2C-bus peripheral clock) period: the previous SDA value is sampled instead of the current one. This can result in a wrong receipt of target 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 I^2C-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.

2.12.2 Spurious bus error detection in controller mode

Description

In controller 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 I^2C-bus transfer in controller mode and any such transfer continues normally.

Workaround

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

2.13 USART

2.13.1 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.

2.13.2 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.

2.14 LPUART

2.14.1 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.

2.15 SPI

2.15.1 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.

2.16 FDCAN

2.16.1 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.

2.16.2 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
```

2.17 OTG_FS

2.17.1 Potential unexpected transfer on the USB bus instead of a zero-length packet

Description

In Device mode, when a zero-length packet must be transmitted on the USB bus, an incorrect data packet might be sent. This issue depends on several events occurring within a very short period and has never been observed in a real end application. The sequence that can generate this issue occurs when the device endpoint is enabled simultaneously or very shortly after setting the CNAK (clear NAK). Additionally, this must happen two AHB clock cycles before the end of the token is received from the host.

Workaround

1. Program DIEPCTLx.SNAK = 1 and DIEPCTLx.CNAK = 0.
2. Wait for 20 AHB clock cycles (this value is determined by considering the fastest AHB clock and the slowest phy_clk combination).
3. Program DIEPCTLx.EPENA = 1.
4. Wait for 15 AHB clock cycles (fixed safe delay).
5. Program DIEPCTLx.CNAK = 1 and DIEPCTLx.SNAK = 0.

The above steps must be performed for all IN transfers involving a zero-length packet transmission in Device mode (DIEPTSIZx.XFRSIZ = 0 and DIEPTSIZx.PKTCNT = 1).

2.18 OTG_HS

2.18.1 Potential unexpected transfer on the USB bus instead of a zero-length packet

Description

In both buffer DMA and slave modes, when a zero-length packet must be transmitted on the USB bus, an incorrect data packet can be sent on the USB bus.

Workaround

In buffer DMA mode:

1. Program DIEPCTLx.EPENA=1, DIEPCTLx.SNAK=1, DIEPCTLx.CNAK=0.

2. Wait for 15 AHB clock cycles.
3. Program DIEPCTLx.CNAK=1, DIEPCTLx.SNAK=0.

The above steps must be performed for all IN transfers that involve a zero-length packet transmission.

In target mode:

1. Program DIEPCTLx.SNAK=1, DIEPCTLx.CNAK=0.
2. Wait for 20 AHB clock cycles (this value is determined by considering the fastest AHB clock and slowest phy_clk combination).
3. Program DIEPCTLx.EPENA=1.
4. Wait for 15 AHB clock cycles (fixed safe delay).
5. Program DIEPCTLx.CNAK=1 and DIEPCTLx.SNAK=0.

The above steps must be performed for all IN transfers that involve a zero-length packet transmission in device mode (DIEPTSIZx.XFRSIZ=0 and DIEPTSIZx.PKTCNT=1).

2.18.2 False detection of chirp-K when a FS device is connected

Description

When the host controller is programmed in high-speed mode and the port reset (OTG_HPRT.PRST) is programmed within 150 PHY clock cycles after a full-speed device connection is detected, the controller can detect a false chirp-K. This causes the controller to switch to high-speed operation even though no chirp is received from the connected device.

Workaround

1. Program the OTG_GINTMSK.PRTIM bit to unmask.
2. Configure the OTG_HCFG register to select either the full-speed host or high-speed host.
3. Set the OTG_HCFG.PPWR bit to 1.
4. Wait for the OTG_HPRT.PCDET interrupt, which indicates that a device is connected to the port.
5. Wait for 150 PHY clock cycles.
6. Set the OTG_HPRT.PRST bit to 1 to start the reset process.

2.19 UCPD

2.19.1 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 CEC

2.20.1 Missed CEC messages in normal receiving mode

Description

In normal receiving mode, any CEC message with destination address different from the own address should normally be ignored and have no effect to the CEC peripheral. Instead, such a message is unduly written into the reception buffer and sets the CEC peripheral to a state in which any subsequent message with the destination address equal to the own address is rejected (NACK), although it sets RXOVR flag (because the reception buffer is considered full) and generates (if enabled) an interrupt. This failure can only occur in a multi-node CEC framework where messages with addresses other than own address can appear on the CEC line.

The listen mode operates correctly.

Workaround

Use listen mode (set LSTEN bit) instead of normal receiving mode. Discard messages to single listeners with destination address different from the own address of the CEC peripheral.

2.20.2 Unexpected TXERR flag during a message transmission

Description

During the transmission of a 0 or a 1, the HDMI-CEC drives the open-drain output to high-Z, so that the external pull-up implements a voltage rising ramp on the CEC line.

In some load conditions, with several powered-off devices connected to the HDMI-CEC line, the rising voltage may not drive the HDMI-CEC GPIO input buffer to V_{IH} within two HDMI-CEC clock cycles from the high-Z activation to TXERR flag assertion.

Workaround

Limit the maximum number of devices connected to the HDMI-CEC line to ensure the GPIO V_{IH} threshold is reached within a time of two HDMI-CEC clock cycles (~61 µs).

The maximum equivalent 10%-90% rise time for the HDMI-CEC line is 111.5 µs, considering a V_{IH} threshold equal to 0.7 x V_{DD}.

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 5. **Document revision history**

Date	Version	Changes
19-Feb-2026	1	Initial release.

Contents

IMPORTANT NOTICE – READ CAREFULLY

STMicroelectronics NV and its subsidiaries (“ST”) reserve the right to make changes, corrections, enhancements, modifications, and improvements to ST products and/or to this document at any time without notice.

In the event of any conflict between the provisions of this document and the provisions of any contractual arrangement in force between the purchasers and ST, the provisions of such contractual arrangement shall prevail.

The purchasers should obtain the latest relevant information on ST products before placing orders. ST products are sold pursuant to ST’s terms and conditions of sale in place at the time of order acknowledgment.

The purchasers are solely responsible for the choice, selection, and use of ST products and ST assumes no liability for application assistance or the design of the purchasers’ products.

No license, express or implied, to any intellectual property right is granted by ST herein.

Resale of ST products with provisions different from the information set forth herein shall void any warranty granted by ST for such product.

If the purchasers identify an ST product that meets their functional and performance requirements but that is not designated for the purchasers' market segment, the purchasers shall contact ST for more information.

ST and the ST logo are trademarks of ST. For additional information about ST trademarks, refer to www.st.com/trademarks. All other product or service names are the property of their respective owners.

Information in this document supersedes and replaces information previously supplied in any prior versions of this document.