STM32U59xxx STM32U5Axxx

Errata sheet

STM32U59xxx and STM32U5Axxx device errata

Applicability

This document applies to the part numbers of STM32U59xxx and STM32U5Axxx 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 RM0456.

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
STM32U59xxx	STM32U595AI, STM32U595AJ,STM32U595QI, STM32U595QJ, STM32U595RI, STM32U595RJ, STM32U595VI, STM32U595VJ, STM32U595ZI, STM32U595ZJ, STM32U599BJ, STM32U599NI, STM32U599NJ, STM32U599VI, STM32U599VJ, STM32U599ZI, STM32U599ZJ
STM32U5Axxx	STM32U5A5AJ, STM32U5A5QJ, STM32U5A5RJ, STM32U5A5VJ, STM32U5A5ZJ, STM32U5A9BJ, STM32U5A9NJ, STM32U5A9VJ, STM32U5A9ZJ

Table 2. **Device variants**

Reference	Silicon revision codes	
	Device marking(1)	REV_ID(2)
STM32U59xxx and STM32U5Axxx	X	0x3001

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

1 Summary of device errata

The following table gives a quick reference to the STM32U59xxx and STM32U5Axxx 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. X
Core	2.1.1	Access permission faults are prioritized over unaligned Device memory faults	N
System	2.2.1	LSE disturbed by RTC/TAMP OUT function on PC13	N
	2.2.2	PWR_BDCR1 is not write-protected by DBP	N
	2.2.3	The PWR_S3WU interrupt is generated for internal wakeup sources (WUSELx = 11)	A
	2.2.4	MSIK clock cannot be stopped when used as kernel clock by MDF1 or ADF1	A
	2.2.5	Too low MSI frequency upon exit from Standby or Stop 3 mode	A
	2.2.6	HardFault on wakeup from Stop mode may occur in debug mode	N
	2.2.7	Full JTAG configuration without NJTRST pin cannot be used	A
	2.2.8	Incorrect backup domain reset with V_{BAT} and V_{DD} supplied by different power sources	A
	2.2.9	EXTI LOCK bit does not lock privilege configuration	N
	2.2.10	Device may be locked upon system reset under Stop 2 mode	P
GPIO	2.3.1	Software can modify GPIOs configuration not available in WLCSP150 packages	A
FMC	2.4.1	Dummy read cycles inserted when reading synchronous memories	N
	2.4.2	Wrong data read from a busy NAND memory	A
OCTOSPI	2.5.1	Memory-mapped write error response when DQS output is disabled	P
	2.5.2	Byte possibly dropped during an SDR read in clock mode 3 when a transfer gets automatically split	A
	2.5.3	Received data corrupted after arbitration ownership toggles when using clock mode 3 and no DQS for read direction	A
	2.5.4	Deadlock can occur under certain conditions	A
	2.5.5	Deadlock or write-data corruption after spurious write to a misaligned address in OCTOSPI_ AR register	N
	2.5.6	Read data corruption after a few bytes are skipped when crossing a four-byte boundary	A
	2.5.7	At least six cycles memory latency must be set when DQS is used for HyperBus™ memories	A
	2.5.8	Data write discarded in memory-mapped mode if a write to a misaligned address is directly followed by a request to the same address	A
HSPI	2.6.1	Memory wrap instruction not enabled when DQS is disabled	N

Function	Section	Limitation	Status
			Rev. X
HSPI	2.6.2	Deadlock or write-data corruption after spurious write to a misaligned address in HSPI_ AR register	N
	2.6.3	Deadlock on consecutive out-of-range memory-mapped write operations	P
	2.6.4	HSPI deadlock or RAM content corrupted on CSBOUND split during prefetch, when DQS is disabled	A
	2.6.5	Indirect write mode limited to 256 Mbytes	N
	2.6.6	Read-modify-write operation does not clear the MSEL bit	A
ADC12	2.7.1	Unexpected regular conversion when two consecutive injected conversions are performed in Dual interleaved mode	A
	2.7.2	ADC_AWDy_OUT reset by non-guarded channels	A
	2.7.3	Injected data stored in the wrong ADC_JDRx registers	A
	2.7.4	ADC slave data may be shifted in Dual regular simultaneous mode	A
ADC4	2.8.1	ADC4 conversion error when used simultaneously with ADC1 or ADC2	P
VREFBUF	2.9.1	V_{REFBUF_OUT} voltage overshoots in Range 4, Stop 1 or Stop 2 mode	A
DSI	2.10.1	DSI automatic clock lane control not functional	P
LPTIM	2.11.1	Device may remain stuck in LPTIM interrupt when entering Stop mode	A
	2.11.2	ARRM and CMPM flags are not set when APB clock is slower than kernel clock	P
	2.11.3	Interrupt status flag is cleared by hardware upon writing its corresponding bit in LPTIM_DIER register	N
IWDG	2.12.1	IWDG is stopped when BDRST is set	P
RTC and TAMP	2.13.1	Alarm flag may be repeatedly set when the core is stopped in debug	N
	2.13.2	Binary mode: SSR is not reloaded with 0xFFFF FFFF when SSCLR = 1	A
	2.13.3	Parasitic tamper detection when debugger is used in RDP Level 0	A
I2C	2.14.1	Wrong data sampling when data setup time ($t_{SU;DAT}$) is shorter than one I2C kernel clock period	P
	2.14.2	Spurious bus error detection in master mode	A
USART	2.15.1	Data corruption due to noisy receive line	N
	2.15.2	Wrong data received by SPI slave receiver in Autonomous mode with CPOL = 1	A
	2.15.3	Received data may be corrupted upon clearing the ABREN bit	A
LPUART	2.16.1	Possible LPUART transmitter issue when using low BRR[15:0] value	P
SPI	2.17.1	Possible corruption of last-received data depending on CRCSIZE setting	A
	2.17.2	MODF flag cannot generate interrupt	A
	2.17.3	RDY output failure at high serial clock frequency	N
	2.17.4	Master communication suspension fails in Autonomous mode	N
	2.17.5	SPE may not be cleared upon MODF event	A
	2.17.6	SPI slave stalls with masters not providing extra SCK periods upon *Not ready* signalling	A
	2.17.7	Truncation of SPI output signals after EOT event	A
FDCAN	2.18.1	Desynchronization under specific condition with edge filtering enabled	A
	2.18.2	Tx FIFO messages inverted under specific buffer usage and priority setting	A
UCPD	2.19.1	TXHRST upon write data underflow corrupting the CRC of the next packet	A
	2.19.2	Ordered set with multiple errors in a single K-code is reported as invalid	N

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

Table 4. **Summary of device documentation errata**

Function	Section	Documentation erratum
System	2.2.11	SRAM ECC error flags and addresses are updated only if interrupt is enabled
ADC12	2.7.5	14-bit ADC offset error and integral linearity error values are increased in datasheets

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 Core

Reference manual and errata notice for the Arm® Cortex®-M33 core revision r0p4 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 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.

2.2 System

2.2.1 LSE disturbed by RTC/TAMP OUT function on PC13

Description

LSE frequency can be incorrect on LQFP packages when PC13 is used as output for RTC_OUT1 or TAMP_OUT2 function.

Workaround

None.

The RTC_OUT2 function must be used on PB2 and TAMP_OUT functions must be used on other I/Os on LQFP packages.

2.2.2 PWR_BDCR1 is not write-protected by DBP

Description

When the DBP bit of PWR_DBPR register is cleared, the write access to backup domain content is expected to be disabled. However, PWR_BDCR1 register is not write-protected when DBP = 0.

Workaround

None.

2.2.3 The PWR_S3WU interrupt is generated for internal wakeup sources (WUSELx = 11)

Description

According to some reference manual versions, the PWR_S3WU interrupt is not generated for internal wakeup sources (when WUSELx = 11 in PWR_WUCR3 register). However, upon wakeup from Stop 3 mode with WUSELx = 11, two interrupts are generated: the PWR_S3WU and the internal wakeup source (RTC or TAMP) interrupts.

Note: The PWR_S3WU flag (WUFx in PWR_WUSR) is automatically cleared when clearing the internal wakeup source flag.

Workaround

Set the PWR_S3WU interrupt with a lower priority than the internal source interrupt (RTC or TAMP). Consequently, upon wakeup from Stop 3 mode, the RTC or TAMP interrupt is serviced first. Then, the PWR_S3WU interrupt is entered without the corresponding flag (WUFx in PWR_WUSR) being set. This interrupt service routine can be returned without clearing any flag.

2.2.4 MSIK clock cannot be stopped when used as kernel clock by MDF1 or ADF1

Description

MSIK can be used as kernel clock by the MDF1 and ADF1 by configuring the MDF1SEL and ADF1SEL bitfields of RCC_CCIPRx registers.

Once selected as kernel clock source for MDF1 or ADF1 peripherals, MSIK can no longer be disabled using the MSIKON bits of the RCC_CR register. This is because MDF1 and ADF1 request their kernel clock by default after reset.

If the application requests an entry in Stop 0, Stop 1, or Stop 2 mode, the selected MSIK kernel clock remains enabled leading to an overconsumption in this mode. The Stop 3 mode is not concerned by this limitation. MDF1 is not concerned for Stop 2 mode.

Workaround

Before disabling the MSIK clocks, configure MDF1SEL and ADF1SEL to HCLK.

In case the application wants to enter Stop 0 or Stop 1 mode, and the MDF1 and ADF1 are not used during Stop mode, then configure MDF1SEL and ADF1SEL bitfields to HCLK before entering Stop mode. Upon Stop mode exit, restore the required MSIK kernel clock with the MDF1SEL and ADF1SEL bitfields. In case the application wants to enter Stop 2 mode and the ADF1 is not used during Stop mode, the same procedure is needed only for ADF1SEL.

In case the application wants to enter Stop 0, Stop 1, or Stop 2 mode and the MDF1 or ADF1 peripheral is used during Stop mode (Stop 0 and Stop 1 modes only for MDF1), then simply configure the MDF1SEL and ADF1SEL bitfields to required MSIK kernel clock. These peripherals request their kernel clock when they need it.

2.2.5 Too low MSI frequency upon exit from Standby or Stop 3 mode

Description

The MSI clock frequency can be up to 25% lower than expected upon exit from Standby or Stop 3 mode, for a maximum of 200 µs.

Workaround

In case accuracy is needed, wait for 200 µs after exiting Standby or Stop 3 mode before using the MSI.

2.2.6 HardFault on wakeup from Stop mode may occur in debug mode

Description

A HardFault may occur at wakeup from Stop mode when the following conditions are met:

- Device is in debug mode.
- DBG_STOP bit is set in DBGMCU_CR.
- A wakeup event/interrupt from an SRD peripheral (except EXTI) occurs in a timing window of four clock cycles during Stop mode entry sequence. SRD peripherals are the ones connected to AHB3 and APB3.

Workaround

None.

2.2.7 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.

2.2.8 Incorrect backup domain reset with V_{BAT} and V_{DD} supplied by different power sources

Description

The backup domain reset may be missed upon backup domain power-on subsequent to a V_{BAT} power-off, in V_{BAT} mode, if the V_{BAT} voltage during the power-off phase drops to a few mV window before starting to rise again. In this critical window, the flip-flops are no longer able to safely retain the information, and the backup domain reset has not yet been triggered. This window is located in the range between 100 mV and 700 mV, with the exact position depending mainly on the device and on the temperature. The issue only occurs when MONEN = 0 in the PWR_BDCR1 register (backup domain supply and temperature monitoring are disabled).

This missed reset results in unpredictable values of the backup domain registers, which may cause a spurious behavior such as driving the LSCO output pin on PA2, raising an unexpected tamper event preventing the SRAM2 or PKA access, or influencing any of the backup domain functions.

Workaround

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

- If the extra consumption is acceptable, enable backup domain supply and temperature monitoring (MONEN = 1 in PWR_BDCR1). The maximum extra consumption is 1.2 µA from V_{BAT} when V_{DD} is above V_{BOR0}, and around 3.6 µA from V_{BAT} in V_{BAT} mode.
- In the application, let the V_{BAT} supply voltage fall to a level below 100 mV for more than 200 ms, before a new power-on.

If the two previous workarounds are not applicable 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 or exit from Shutdown mode, at least one backup register (called, for example, BackupTestRegister) must be previously programmed with a BKP_REG_VAL value with 16 bits set and 16 bits cleared. Robustness of this workaround can be significantly improved by using a CRC rather than registers. 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 bits modified by hardware.

The CRC result can be stored in the backup register instead of a fixed value. This value needs to be updated for each modification of values covered by CRC, such as by using CRC peripheral.

At the very beginning of the boot code, insert the following software sequence:

1. Check if the BORRSTF flag of RCC_CSR is set (the reset is caused by a power-on).
2. If true, check that the BackupTestRegister content is different from BKP_REG_VAL, or the CRC recalculation is different from stored results, accordingly the chosen workaround implementation.
3. If true and if no tamper flag is set (when tamper detection is enabled), the reset is caused by a backup domain power-on. Apply the following sequence:
    a. Enable PWR clock in RCC, by setting the PWREN bit of RCC_AHB3ENR.
    b. Enable backup domain access, by setting the DBP bit of PWR_DBPR.
    c. Reset backup domain, by:
        i. writing 0x0001 0000 in RCC_BDCR, which sets the BDRST bit and clears other register bits that might not be reset
        ii. reading RCC_BDCR, to make the reset time long enough
        iii. writing 0x0000 0000 in RCC_BDCR, to clear the BDRST bit
    d. Clear BORRSTF, by setting the RMVF bit of RCC_CSR.

2.2.9 EXTI LOCK bit does not lock privilege configuration

Description

Both security and privilege configuration of the extended interrupts and event controller (EXTI) must be locked when the LOCK bit of the EXTI lock register (EXTI_LOCKR) is set. The EXTI security configuration register (EXTI_SECCFGR1) is locked as expected, but the EXTI privilege configuration register (EXTI_PRIVCFGR1) can still be modified when the LOCK bit is set.

Workaround

None.

2.2.10 Device may be locked upon system reset under Stop 2 mode

Description

A system reset occurs when the MCU is in Stop 2 mode on LDO regulator, with at least, one RAM in power-down (SRAMx, Caches, or peripheral SRAMs). This may lock the device in a high consumption state (dozens of mA). The IWDG, the external NRST pin, and the BOR thresholds 1 to 4 are the reset sources that create this limitation.

Only a power-on sequence recovers from this state: V_{DD} must drop below V_{BOR0} (brownout reset minimum threshold value), then raise to its expected value.

This limitation is not present:

- upon wake-up from Stop 2 mode with other wake-up sources (GPIO or peripheral interrupt).
- if the device is supplied by the SMPS regulator during Stop 2 mode.
- in Stop 0, Stop 1, and Stop 3 modes.

Workaround

Keep all SRAMs powered on during Stop 2 mode that is all xRAMxPD/PDS bits in PWR_CRx registers must be kept at their reset state.

2.2.11 SRAM ECC error flags and addresses are updated only if interrupt is enabled

Description

In the reference manual RM0456, up to version 4, it is stated that:

- If a single error is detected, SEDC and CSEDC flags are set in RAMCFG_MxISR and RAMCFG_MxICR respectively. The ECC single error address is updated in RAMCFG_MxSEAR.
- If a double error is detected, DED and CDED flags are set in RAMCFG_MxISR and RAMCFG_MxICR respectively. The ECC double error address is updated in RAMCFG_MxDEAR.

However, the real behavior is that:

- If a single error is detected, SEDC and CSEDC flags are set, and the associated ECC single error address is updated only if the single error interrupt is enabled (SEIE bit of RAMCFG_MxIER is set).
- If a double error is detected, DED and CDED flags are set, and the associated ECC double error address is updated only if double error interrupt or NMI is enabled (DEIE bit or ECCNMI bit of RAMCFG_MxIER is set).

This is a documentation error rather than a device limitation.

Workaround

When the application needs to get the ECC error flags and addresses status without servicing the associated interrupts, the RAMCFG SEIE/DEIE interrupts must be enabled with the associated NVIC RAMCFG vector 5 disabled.

2.3 GPIO

2.3.1 Software can modify GPIOs configuration not available in WLCSP150 packages

Description

Software can configure the unbonded GPIOs in WLCSP150 packages. This can lead to an extra consumption if they are floating with Schmitt trigger ON.

Workaround

Configure all unbonded GPIOs as analog input or as output push-pull 0 state.

2.4 FMC

2.4.1 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.

2.4.2 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.

2.5 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 Byte possibly dropped during an SDR read in clock mode 3 when a transfer gets automatically split

Description

When reading a continuous stream of data from sequential addresses in a serial memory, the OCTOSPI can interrupt the transfer and automatically restart it at the next address when features generating transfer splits (CSBOUND, REFRESH, TIMEOUT or MAXTRAN) are active. Thus, a single continuous transfer can effectively be split into multiple smaller transfers.

When the OCTOSPI is configured to use clock mode 3 (CKMODE bit of the OCTOSPI_DCR1 register set) and a continuous stream of data is read in SDR mode (DDTR bit of the OCTOSPI_CCR register cleared), the last byte sent by the memory before an automatic split gets dropped, thus causing all the subsequent bytes to be seen one address earlier.

Workaround

Use clock mode 0 (CKMODE bit of the OCTOSPI_DCR1 register cleared) when in SDR mode.

2.5.3 Received data corrupted after arbitration ownership toggles when using clock mode 3 and no DQS for read direction

Description

When two OCTOSPI peripherals compete the ownership for the same external bus through the I/O manager and the following conditions are met:

- at least one of the OCTOSPIs operating in read mode
- at least one of the OCTOSPIs set with clock mode 3

the received data is corrupted due to a spurious sampling event when the ownership of the external bus toggles.

Workaround

Use clock mode 0, by setting the bit CKMODE of the OCTOSPI_DCR1 register (all memories known to date support clock mode 0).

2.5.4 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.

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.

2.5.5 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.

2.5.6 Read data corruption after a few bytes are skipped when crossing a four-byte boundary

Description

A memory-mapped read is corrupted when the following sequence occurs:

1. An 8- or 16-bit read is performed in a four-byte window, and the last byte of this window is not read.
2. Any read in the next four-byte window is done before the completion of the previous read memory prefetch.

OCTOSPI immediately responds to this read request, even before the data are read from memory, thus resulting in incorrect data read.

If the next read operations are issued to consecutive addresses, then the read data are also corrupted.

This limitation is not present when the SDMMC accesses the OCTOSPI since the SDMMC performs only 32-bit read accesses.

Workaround

Apply one of the following measures:

- Perform only 32-bit memory-mapped read accesses. For CPU read accesses, enable ICACHE and/or DCACHE and configure the OCTOSPI memory as cacheable (CACHE only performs 32-bit read accesses).
  For system DMA, use only 32-bit data size for read accesses.
  If the DMA2D uses 4-, 8-, 16- or 24-bpp picture format, add dummy pixels at the end of each picture line to increase the number of bytes that are skipped when performing read operations. As an example, a 128 x 28 x 24 bpp picture can be extended to 132 x 128 x 24 bpp to ensure that any sequence skipping bytes skips a minimum of two words on each line.
- If this is not possible and an 8-bit or 16-bit read is done at the beginning of a four-byte window, ensure that the next access is not a read from the next four-byte window or that the second access occurs after the data at the skipped addresses are prefetched from memory.

2.5.7 At least six cycles memory latency must be set when DQS is used for HyperBus™ memories

Description

For HyperBus™ memories, the TACC[7:0] bitfield of the OCTOSPI_HLCR register enables the setting of the memory latency in number of clock cycles. These dummy cycles are inserted between the address and the data phases during read operations.

When the DQS signal is used for HyperBus™ memories, and the number of latency clock cycles programmed in TACC[7:0] is lower than six, a deadlock occurs during read operations.

Workaround

Configure the memory and the octo-SPI controller to have at least six clock cycles of latency.

2.5.8 Data write discarded in memory-mapped mode if a write to a misaligned address is directly followed by a request to the same address

Description

In memory-mapped mode with DQS enabled, a data write is discarded if the write targets a misaligned address and is directly followed by a request (cycle by cycle on AHB) to the same address.

An address is misaligned if the address is odd and the OCTOSPI is configured to send two bytes of data to the memory on every cycle that is targeted by one of the following modes:

- Octal DTR
- Dual-quad DTR

Workaround

Use one of the following measures:

- Configure the OCTOSPI to issue commands to aligned addresses, that is to an even address when two bytes are transferred during each clock cycle.
- Avoid consecutive back-to-back (AHB cycle by cycle) accesses to the memory after a write to a memory mapped at the same address. Instead, insert a NOP (no operation) or a software delay between the two accesses.

2.6 HSPI

2.6.1 Memory wrap instruction not enabled when DQS is disabled

Description

Memory wrap instruction (as configured in the HSPI_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.

2.6.2 Deadlock or write-data corruption after spurious write to a misaligned address in HSPI_ AR register

Description

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

- FMODE = 00 in HSPI_CR (Indirect write mode)
- DQSE = 1 in HSPI_CCR (DQS active)

then, the HSPI 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 HSPI 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 HSPI is configured to send four bytes of data to the memory (16-bit DTR mode or dual-octal DTR mode).

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

Workaround

None.

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

Description

The DEVSIZE[4:0] bitfield of the HSPI_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 HSPI 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.

2.6.4 HSPI deadlock or RAM content corrupted on CSBOUND split during prefetch, when DQS is disabled

Description

Depending on the HSPI configuration and sequence of operations, HSPI may be deadlocked after an abort or the RAM content corrupted when a REFRESH, TIMEOUT or MAXTRAN event occurs.

When the HSPI is configured as follows:

- 16-bit DTR mode or dual-octal DTR mode enabled
- DQS input disabled (DQSE bit cleared into HSPI_CCR register)

and the following sequence occurs:

1. The last byte is read from the AMBA interface at an address that is 65-72 bytes before a CSBOUND split.
2. There are no more memory-mapped requests (or no more reads from HSPI_DR register in indirect read mode) for long enough that the prefetch mechanism fills up the FIFO.

then, the issue is encountered if either of the two events occurs:

- An abort is requested before another memory-mapped request is issued.
  In such case, HSPI becomes thoroughly deadlocked, and only a reset enables to recover from deadlock.
- No new memory-mapped requests are issued for so long that the command to the memory can be interrupted (chip select released) due to a REFRESH, TIMEOUT or MAXTRAN event.
  In such case, the chip select is not released and the RAM content may get corrupted (for instance if no refresh is performed).

Workaround

Use the DQS output of the memory (by setting the DQSE bit of the HSPI_CCR register) when using 16-bit memories or octal memories in dual-octal mode.

2.6.5 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 HSPI_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.

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

Description

When the MSEL bit of the HSPI_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 HSPI_CR register, otherwise, the MSEL bit remains set.

2.7 ADC12

2.7.1 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.

2.7.2 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.

2.7.3 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).

2.7.4 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.

2.7.5 14-bit ADC offset error and integral linearity error values are increased in datasheets

Description

STM32U59x and STM32U5Ax datasheets revision 1 (DS13633 and DS13543) specify that:

- The offset error maximum value is ±5 LSB in singled ended mode, while the correct value is ±12 LSB.
- The integral linearity error maximum value is ±5 LSB in singled ended mode, while the correct value is ±7 LSB.

This is a documentation error rather than a device limitation. This error has no impact on the ADC accuracy.

Workaround

None.

2.8 ADC4

2.8.1 ADC4 conversion error when used simultaneously with ADC1 or ADC2

Description

The ADC4 conversion is disturbed when carried out simultaneously with an ADC1 or ADC2 conversion and results in an ADC4 conversion error. The error is due to a V_{REF+} disturbance during an ADC1 or ADC2 sampling phase, and occurs regardless of which V_{REF+} is provided: either by external reference, or by VREFBUF.

Workaround

ADC1 and ADC4, or ADC2 and ADC4 analog converter clocks must be identical (same clock source, same frequency, and same phase) to avoid any disturbance. This is possible only when both ADC prescalers are programmed without any division factor. The bitfields of the both registers listed below must be set to 0b0000:

- ADC1 or ADC2 prescaler: PRESC[3:0] in the ADC12_CCR register
- ADC4 prescaler: PRESC[3:0] in the ADC4_CCR register

ADC1 and ADC4, or ADC2 and ADC4 analog converter clock frequencies must not exceed 55 MHz and the ADC4 clock duty cycle must be set to between 45% and 55%. Selecting AHB clock as the ADC kernel clock limits the whole AHB to 55 MHz. Other independent clock sources can be used to avoid this drawback. As a result, the triggers from timers, that are clocked by AHB clock, have an uncertainty due to trigger synchronization delay from timers AHB clock to ADC asynchronous kernel clock. In case PLL output pll2_r_ck is used as ADC kernel clock, the PLL division (controlled by the PLL2R[6:0] bitfield in the RCC_ RCC_PLL2_DIVR register) must be set to an even division (division by 2, 4, and so on) to guarantee a 50% ratio.

Note: *The use of ADC kernel clock division factor is possible if ADC1 and ADC4, or ADC2 and ADC4 do not convert simultaneously.*

2.9 VREFBUF

2.9.1 V_{REFBUF_OUT} voltage overshoots in Range 4, Stop 1 or Stop 2 mode

Description

The V_{REFBUF_OUT} output voltage overshoots when started:

- while the regulator is in Range 4.
- while the device is in Stop 1 or Stop 2 mode.

Workaround

Modify the regulator voltage range to Range 1, 2, or 3 before enabling the voltage reference buffer.

2.10 DSI

2.10.1 DSI automatic clock lane control not functional

Description

The automatic mechanism used to manage the clock activity on the clock lane can cause a D-PHY timing violation between clock lane start/stop and data lane HS-to-LP and LP-to-HS transition.

The automatic clock lane control (ACR) bit of the DSI Host clock lane configuration register (DSI_CLCR) must always be kept at 0 (reset value).

Workaround

In video mode, the clock has to be provided continuously.

In command mode, the clock signal on the clock line can be interrupted when there is no ongoing HS transmission. This is done by clearing the D-PHY clock control (DPCC) bit of the DSI Host clock lane configuration register (DSI_CLCR). The DPCC bit must be set before any HS transmission.

2.11 LPTIM

2.11.1 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 RCC_APByRSTRz register.

2.11.2 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.*

2.11.3 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.

2.12 IWDG

2.12.1 IWDG is stopped when BDRST is set

Description

IWDG, once started, is expected to stop only in case of system reset. However, the LSI (IWDG clock) is stopped when BDRST is set in the RCC_BDCR register.
In addition, the BDRST bit is not protected against non-secure access when LSI or IWDG is secure.

Workaround

If a Backup domain reset must be done, set BDRST and clear it right after to minimize the duration LSI is stopped.
BDRST can be protected against non-secure access by configuring at least one function of RTC or TAMP as secure.

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.

2.13.2 Binary mode: SSR is not reloaded with 0xFFFF FFFF when SSCLR = 1

Description

When SSCLR bit of the RTC_ALRMxSSR register is set when in binary mode, SSR is reloaded with 0xFFFF FFFF at the end of the ck_apre cycle when RTC_SSR is set to RTC_ALRxBINR (*x* stands for either A or B)

RTC_SSR is not reloaded with 0xFFFF FFFF if RTC_ALRxBINR is modified while RTC_SSR is set to RTC_ALRxBINR. Rather, SSR continues to decrement.

Workaround

The workarounds are described for alarm A, and can be applied in the same manner for alarm B. Two workarounds are proposed, the second one requires to use the second alarm.

- Wait for one ck_apre cycle after an alarm A event before changing the RTC_ALRABINR register value.
- Do not reprogram RTC_ALRABINR following the alarm A event itself. Instead, use alarm B configured with RTC_ALRBBINR set to 0xFFFF FFFF, and reprogram RTC_ALRABINR after the alarm B event. This ensures that one ck_apre cycle elapses following the alarm A event.

2.13.3 Parasitic tamper detection when debugger is used in RDP Level 0

Description

The internal tamper 6 flag (ITAMP6F) can be unexpectedly set in the TAMP status register (TAMP_SR) when a debugger is connected in RDP Level 0, in case a switch to V_{BAT} occurs (V_{DD} is below the BOR0 threshold).

Workaround

Keep internal tamper 6 flag disabled as long as debug is needed, and enable it once development phase is complete. The tamper flag cannot be set if no debug access is done.

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^{2}C-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^{2}C-bus SDA line when $t_{SU;DAT}$ is smaller than one I2C kernel clock (I^{2}C-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 I^{2}C-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 I^2C-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 USART

2.15.1 Data corruption due to noisy receive line

Description

In UART mode with oversampling by 8 or 16 and with 1 or 2 stop bits, the received data may be corrupted if a glitch to zero shorter than the half-bit occurs on the receive line within the second half of the stop bit.

Workaround

None.

2.15.2 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.

2.15.3 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 LPUART

2.16.1 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 LPUART_BRR register (BRR[15:0]) is not an integer, and is in the three to four range. A typical example is when the LSE 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:

- Increase the ratio between the LPUART kernel clock and the baud rate. To do so:
  - Increase the LPUART kernel clock frequency by using a clock different from the LSE clock, or
  - Decrease the baud rate.
- Generate the baud rate on the receiver side by using a higher frequency and applying oversampling techniques if supported.

2.17 SPI

2.17.1 Possible corruption of last-received data depending on CRCSIZE setting

Description

With the CRC calculation disabled (CRCEN = 0), the transfer size bitfield set to a value greater than zero (TSIZE[15:0] > 0), and the length of CRC frame set to less than 8 bits (CRCSIZE[4:0] < 00111), the last data received in the RxFIFO may be corrupted.

Workaround

Keep the CRCSIZE[4:0] bitfield at its default setting (00111) during the data reception if CRCEN = 0 and TSIZE[15:0] > 0.

2.17.2 MODF flag cannot generate interrupt

Description

Mode fault detection results in disabling SPI. With the MODFIE bit of the SPI_IER register set, the mode fault flag (MODF) going high is expected to trigger an interrupt. However, disabling SPI unduly blocks this interrupt request.

Workaround

To detect a mode fault event, poll the MODF flag by software.

2.17.3 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.

2.17.4 Master communication suspension fails in Autonomous mode

Description

The SPI peripheral is blocked regardless of the completion of the ongoing data frame transaction, and the SUPSF flag is never set, when:

- the master provides a communication triggered in Autonomous mode (TRIGEN=1 of the SPI_AUTOCR register), and
- the suspension of the ongoing transaction is applied by setting the CSUSP bit through the smart DMA in Stop mode.

Workaround

None.

Note: *The user software must avoid any master suspension in Stop mode while the master operates in Autonomous mode and waits for EOT if TSIZE is greater than 0. If an endless transaction is applied (TSIZE = 0), the suspension is the only way to stop the ongoing transaction. Then to unblock the peripheral, the software must disable SPI then apply the hardware reset. Otherwise, the system cannot proceed to the next transaction.*

2.17.5 SPE may not be cleared upon MODF event

Description

The failure described applies to multi-master topology when the device is configured to monitor the SS input signal by hardware (SSM = 0, SSOE = 0 of the SPI_CFG2 register).

If the software sets the SPE (SPI enable) bit of the SPI_CR1 register at the instant of the SS signal transiting to its active logical level, the resulting MODF event duly switches the SPI into slave mode, but it fails to clear the SPE bit and thus disable the SPI.

Note: *The SS active logical level is the one that matches the SSIOP bit of the SPI_CFG2 register.*

Workaround

Whenever MODF event fails to clear the SPE bit, do it by software.

2.17.6 SPI slave stalls with masters not providing extra SCK periods upon *Not ready* signalling

Description

In Stop mode, the device SPI operating as slave with the *Ready* signalling enabled (the RDIOM of the SPI_CFG2 register set) may stall and never retrieve the *Ready* state. This occurs when SCK stops immediately after *Not ready* status.

Note: *STM32 devices supporting the Ready signaling and operating as SPI master provide some extra SCK periods upon detecting Not ready signal, thus allowing the SPI slaves to operate correctly.*

Workaround

If in the application, there is an SPI master that stops SCK immediately upon *Not ready* signal, without providing some extra SCK periods, do not enable the *Ready* signalling.

2.17.7 Truncation of SPI output signals after EOT event

Description

After an EOT event signaling the end of a non-zero transfer size transaction (TSIZE > 0) upon sampling the last data bit, the software may disable the SPI peripheral. As expected, disabling SPI deactivates the SPI outputs (SCK, MOSI and SS when the SPI operates as a master, MISO when as a slave), by making them float or statically output their by-default levels, according to the AFCNTR bit of the SPI_CFG2 register.

With fast software execution (high PCLK frequency) and slow SPI (low SCK frequency), the SPI disable occurring too fast may result in truncating the SPI output signals. For example, the device operating as a master then generates an asymmetric last SCK pulse (with CPHA = 0), which may prevent the correct last data bit reception by the other node involved in the communication.

Workaround

Apply one of the following measures or their combination:

- Add a delay between the EOT event and SPI disable action.
- Decrease the ratio between PCLK and SCK frequencies.

2.18 FDCAN

2.18.1 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.

2.18.2 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 UCPD

2.19.1 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.

2.19.2 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.

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
17-Jan-2023	1	Initial release
30-Mar-2023	2	Added errata: • **System:** Device may be locked upon system reset under Stop 2 mode • EXTI LOCK bit does not lock privilege configuration • SRAM ECC error flags and addresses are updated only if interrupt is enabled • **ADC12:** 14-bit ADC offset error and integral linearity error values are increased in datasheets • **UCPD:** TXHRST upon write data underflow corrupting the CRC of the next packet • Ordered set with multiple errors in a single K-code is reported as invalid Renamed LPUART limitation: Wrong data received when the communication nodes are two LPUART instances into Possible LPUART transmitter issue when using low BRR[15:0] value Removed errata not applicable for this product: • **TIM:** Consecutive compare event missed in specific conditions • Output compare clear not working with external counter reset

Contents

1 Summary of device errata . . . 2

2 Description of device errata . . . 5

2.1 Core . . . 5

2.1.1 Access permission faults are prioritized over unaligned Device memory faults . . . 5

2.2 System . . . 5

2.2.1 LSE disturbed by RTC/TAMP OUT function on PC13 . . . 5

2.2.2 PWR_BDCR1 is not write-protected by DBP . . . 5

2.2.3 The PWR_S3WU interrupt is generated for internal wakeup sources (WUSELx = 11) . . . 6

2.2.4 MSIK clock cannot be stopped when used as kernel clock by MDF1 or ADF1 . . . 6

2.2.5 Too low MSI frequency upon exit from Standby or Stop 3 mode . . . 6

2.2.6 HardFault on wakeup from Stop mode may occur in debug mode . . . 7

2.2.7 Full JTAG configuration without NJTRST pin cannot be used . . . 7

2.2.8 Incorrect backup domain reset with V_{BAT} and V_{DD} supplied by different power sources . . . 7

2.2.9 EXTI LOCK bit does not lock privilege configuration . . . 8

2.2.10 Device may be locked upon system reset under Stop 2 mode . . . 8

2.2.11 SRAM ECC error flags and addresses are updated only if interrupt is enabled . . . 8

2.3 GPIO . . . 9

2.3.1 Software can modify GPIOs configuration not available in WLCSP150 packages . . . 9

2.4 FMC . . . 9

2.4.1 Dummy read cycles inserted when reading synchronous memories . . . 9

2.4.2 Wrong data read from a busy NAND memory . . . 9

2.5 OCTOSPI . . . 10

2.5.1 Memory-mapped write error response when DQS output is disabled . . . 10

2.5.2 Byte possibly dropped during an SDR read in clock mode 3 when a transfer gets automatically split . . . 10

2.5.3 Received data corrupted after arbitration ownership toggles when using clock mode 3 and no DQS for read direction . . . 10

2.5.4 Deadlock can occur under certain conditions . . . 10

2.5.5 Deadlock or write-data corruption after spurious write to a misaligned address in OCTOSPI_ AR register . . . 11

2.5.6 Read data corruption after a few bytes are skipped when crossing a four-byte boundary . 11

2.5.7 At least six cycles memory latency must be set when DQS is used for HyperBus™ memories . . . 12

2.5.8 Data write discarded in memory-mapped mode if a write to a misaligned address is directly followed by a request to the same address . . . 12

2.6 HSPI . . . 13

2.6.1 Memory wrap instruction not enabled when DQS is disabled . . . 13

2.6.2 Deadlock or write-data corruption after spurious write to a misaligned address in HSPI_ AR register 13
2.6.3 Deadlock on consecutive out-of-range memory-mapped write operations. 13
2.6.4 HSPI deadlock or RAM content corrupted on CSBOUND split during prefetch, when DQS is disabled 14
2.6.5 Indirect write mode limited to 256 Mbytes 14
2.6.6 Read-modify-write operation does not clear the MSEL bit. 14
2.7 ADC12 15
2.7.1 Unexpected regular conversion when two consecutive injected conversions are performed in Dual interleaved mode 15
2.7.2 ADC_AWDy_OUT reset by non-guarded channels 15
2.7.3 Injected data stored in the wrong ADC_JDRx registers. 15
2.7.4 ADC slave data may be shifted in Dual regular simultaneous mode 15
2.7.5 14-bit ADC offset error and integral linearity error values are increased in datasheets . . . 16
2.8 ADC4 16
2.8.1 ADC4 conversion error when used simultaneously with ADC1 or ADC2 16
2.9 VREFBUF 16
2.9.1 V_{REFBUF_OUT} voltage overshoots in Range 4, Stop 1 or Stop 2 mode 16
2.10 DSI 17
2.10.1 DSI automatic clock lane control not functional 17
2.11 LPTIM 17
2.11.1 Device may remain stuck in LPTIM interrupt when entering Stop mode 17
2.11.2 ARRM and CMPM flags are not set when APB clock is slower than kernel clock 17
2.11.3 Interrupt status flag is cleared by hardware upon writing its corresponding bit in LPTIM_DIER register 18
2.12 IWDG 18
2.12.1 IWDG is stopped when BDRST is set 18
2.13 RTC and TAMP 18
2.13.1 Alarm flag may be repeatedly set when the core is stopped in debug 18
2.13.2 Binary mode: SSR is not reloaded with 0xFFFF FFFF when SSCLR = 1 19
2.13.3 Parasitic tamper detection when debugger is used in RDP Level 0 19
2.14 I2C 19
2.14.1 Wrong data sampling when data setup time ($t_{SU;DAT}$) is shorter than one I2C kernel clock period 19
2.14.2 Spurious bus error detection in master mode 20
2.15 USART 20
2.15.1 Data corruption due to noisy receive line. 20
2.15.2 Wrong data received by SPI slave receiver in Autonomous mode with CPOL = 1. 20
2.15.3 Received data may be corrupted upon clearing the ABREN bit. 20

2.16 LPUART 20
2.16.1 Possible LPUART transmitter issue when using low BRR[15:0] value. 20
2.17 SPI 21
2.17.1 Possible corruption of last-received data depending on CRCSIZE setting. 21
2.17.2 MODF flag cannot generate interrupt 21
2.17.3 RDY output failure at high serial clock frequency 21
2.17.4 Master communication suspension fails in Autonomous mode 22
2.17.5 SPE may not be cleared upon MODF event 22
2.17.6 SPI slave stalls with masters not providing extra SCK periods upon *Not ready* signalling . 22
2.17.7 Truncation of SPI output signals after EOT event 22
2.18 FDCAN 23
2.18.1 Desynchronization under specific condition with edge filtering enabled. 23
2.18.2 Tx FIFO messages inverted under specific buffer usage and priority setting 23
2.19 UCPD 24
2.19.1 TXHRST upon write data underflow corrupting the CRC of the next packet 24
2.19.2 Ordered set with multiple errors in a single K-code is reported as invalid 24
Important security notice 25
Revision history 26

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. 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.

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 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.

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.

© 2023 STMicroelectronics – All rights reserved