STM32H7Pxxx device errata

Applicability

This document applies to the part numbers of STM32U2H7Pxxx 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 RM0533.

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
STM32H7P4xx	STM32H7P4VG, STM32H7P4QG, STM32H7P4ZG, STM32H7P4IG, STM32H7P4VI, STM32H7P4QI, STM32H7P4ZI, STM32H7P4II,
STM32H7P5xx	STM32H7P5VG, STM32H7P5QG, STM32H7P5ZG, STM32H7P5IG, STM32H7P5VI, STM32H7P5QI, STM32H7P5ZI, STM32H7P5II,

Table 2. Device variants

Reference	Silicon revision codes	
	Device marking(1)	REV_ID
STM32H7Pxxx	Z	0x1001

1. *Refer to the device datasheet for information on how to identify this code on different package types.*

1 Summary of device errata

The following table gives a quick reference to the STM32U2H7Pxxx 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. Z
Core	2.1.1	PLD might perform linefill to address that would generate a MemManage Fault	A
	2.1.2	Software programming errors might not be reported for online MBIST access to the ICACHE	N
	2.1.3	ECC error causes data corruption when the data cache error bank registers are locked	A
	2.1.4	Store after cache invalidate without intervening barrier might cause inconsistent memory view	A
System	2.2.1	SWD debug connection is lost when NJTRST is driven low	A
FMC	2.3.1	Dummy read cycles inserted when reading synchronous memories	N
	2.3.2	Wrong data read from a busy NAND memory	A
	2.3.3	Unsupported read access with unaligned address	P
	2.3.4	Duty cycle variation on memory clock while accessing PSRAM in continuous clock mode	A
XSPI	2.4.1	Possible deadlock when a request arrives during the disabling process	A
	2.4.2	A read mismatch occurs at the last memory address if a new request arrives before data is read from the I/O pins	A
	2.4.3	A read mismatch occurs after disabling the controller during a read transaction	A
SDMMC	2.5.1	Command response and receive data end bits not checked	N
ADC	2.6.1	In certain dual modes, the fixed trigger latency for the injected conversions may not be respected	A
	2.6.2	In simultaneous regular mode, stopping an injected conversion may shift the next regular conversion master and slave timing by one clock cycle	A
HRTIM	2.7.1	Delayed idle protection malfunction when IDLESx = 1	N
	2.7.2	Deadtime insertion mode not functional	N
	2.7.3	Negative deadtime not supported in push-pull mode	N
	2.7.4	Deadtime slightly shortened in push-pull mode	N
LPTIM	2.8.1	Device may remain stuck in LPTIM interrupt when entering Stop mode	A
	2.8.2	ARRM and CMPM flags are not set when APB clock is slower than kernel clock	A
	2.8.3	Interrupt status flag is cleared by hardware upon writing its corresponding bit in LPTIM_DIER register	N

Function	Section	Limitation	Status Rev. Z
RTC and TAMP	2.9.1	Alarm flag may be repeatedly set when the core is stopped in debug	N
	2.9.2	RTC calendar read may return transient incorrect value when BYPSHAD = 0	A
	2.9.3	System stuck if some RTC or TAMP registers are written when write protection is enabled	N
I2C	2.10.1	Wrong data sampling when data setup time ($t_{SU;DAT}$) is shorter than one I2C kernel clock period	P
	2.10.2	Spurious bus error detection in controller mode	A
USART	2.11.1	Received data may be corrupted upon clearing the ABREN bit	A
	2.11.2	Noise error flag set while ONEBIT is set	N
LPUART	2.12.1	Possible LPUART transmitter issue when using low BRR[15:0] value	P
SPI	2.13.1	RDY output failure at high serial clock frequency	N
FDCAN	2.14.1	Desynchronization under specific condition with edge filtering enabled	A
	2.14.2	Tx FIFO messages inverted under specific buffer usage and priority setting	A
OTG_FS	2.15.1	Potential unexpected transfer on the USB bus instead of a zero-length packet	A
UCPD	2.16.1	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
FMC	2.3.5	SDRAM recovery delay inconsistent when TWR[3:0] = 0000

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®-M7 core revision r1p2 is available from http://infocenter.arm.com.

2.1.1 PLD might perform linefill to address that would generate a MemManage Fault

Description

If the MPU is present and enabled, then it can be programmed so that loads to certain addresses generate a MemManage Fault. This could be because:

- The address is unmapped, that is, it is not in an enabled region and the default memory map is not being used.
- The address cannot be accessed at the current privilege level.
- The address cannot be accessed at any privilege level.

Because of this erratum, a PLD to such an address might incorrectly cause a data cache line-fill.

Conditions:

- The data cache is enabled and the MPU is enabled.
- A PLD is executed, and either:
  - The PLD is to an address not mapped in the MPU, which requires that:
    - The MPU is enabled.
    - The default memory map is not being used.
    - The default memory map is cacheable at that address.
    - The PLD does not hit an enabled MPU region.
  - The PLD is to a region that has permission requirements that the PLD does not meet, which requires that:
    - The MPU is enabled.
    - The default memory map is not being used.
    - The region that the PLD hits is cacheable.
    - The region that the PLD hits would generate a MemManage fault for a load. This requires either:
      - The region cannot be accessed by a read at any privilege level.
      - The region only has read access for privileged code and the PLD is unprivileged.

Note that in rare cases, a PLD instruction can be speculatively executed in the shadow of a mispredicted branch. This can even theoretically be a literal value that decodes to a PLD.

Processor execution is not affected by this erratum. The data returned from the line-fill is not directly consumed by the PLD. Any subsequent load to that address can only access the data if it has permission to do so. This erratum does not permit software to access data that it does not have permissions for.

The only implications of this erratum are the access itself that should not have been performed. This might have an impact on memory regions with side-effects on reads or on memory, which never returns a response on the bus.

Workaround

Accesses to memory that is not mapped in the MPU can be avoided by using MPU region 0 to cover all unmapped memory and make this region execute-never and inaccessible. That is, MPU_RASR0 must be programmed with:

- The ENABLE bit of the MPU_RASR0 register = 0b1; MPU region 0 enable
- The SIZE bit of the MPU_RASR0 register = 0b11111; MPU region 0 size = 2^32 bytes to cover entire memory
- The SRD bit of the MPU_RASR0 register = 0b0000 0000; All sub-regions enabled
- The XN bit of the MPU_RASR0 register = 0b1; Execute-never to prevent instruction fetch
- The AP bit of the MPU_RASR0 register = 0b000; No read or write access for any privilege level
- The TEX bit of the MPU_RASR0 register = 0b000; Attributes = Strongly-ordered
- The C bit of the MPU_RASR0 register = 0b0; Attributes = Strongly-ordered
- The B bit of the MPU_RASR0 register = 0b0; Attributes = Strongly-ordered

Accesses to memory that is mapped in the MPU, but should not be accessed at the current privilege level can be avoided by making the region noncacheable. That is, MPU_RASR0 should be programmed with:

- The TEX bit of the MPU_RASR0 register = 0b000; Attributes = Strongly-ordered
- The C bit of the MPU_RASR0 register = 0b0; Attributes = Strongly-ordered
- The B bit of the MPU_RASR0 = 0b0; Attributes = Strongly-ordered

2.1.2 Software programming errors might not be reported for online MBIST access to the ICACHE

Description

The online MBIST interface provides access to the cache and TCM RAMs to allow in-field memory testing during normal operation of the processor. Because of this erratum, errors in the software that works with the memory testing might not be indicated on the MBISTERR output signal as intended for ICACHE tests.

Note that this erratum does not affect the detection of faults in the memories under test, but affects only the feature that helps to indicate errors in software used during testing.

There are two online MBIST use cases: software transparent and software assisted.

In the software transparent use case, software running on the processor is not involved in or aware of the memory testing being carried out. See the Cortex®-M7 safety manual for more details. In this case, the target memory is automatically locked by the MBIST controller, which causes the processor pipeline to stall if it attempts to access this memory. Testing is carried out using short bursts of accesses, which last for less than 20 clock cycles and do not corrupt the memory contents. For this reason, the memory is locked only for a very short period of time and the gap between bursts is very large.

In the software-assisted use case, the target memory is still locked by the MBIST controller, but the software running on the processor disables the target memory before testing commences. This prevents any software access to this memory during testing. See the Cortex®-M7 safety manual for more details. For this reason, software accesses go to another memory instead of the target memory and the pipeline does not stall. This is important because the software-assisted use case is intended to be used for production MBIST algorithms, which take a long time to run. For example, if the ICACHE were disabled then software might still execute using the main memory or the TCMs.

This erratum only affects the software-assisted use case, when the ICACHE RAMs are tested. An error indication is sent back to the MBIST controller if software attempts to access the target memory while it is locked for testing. Because of this erratum, an error is not indicated back to the MBIST controller on the MBISTERR[0] output signal when software performs a lookup to the ICACHE during MBIST testing.

The error indication is correctly asserted for all the other types of ICACHE access during MBIST testing:

- A cache line invalidates because of an ECC error.
- A cache invalidates by MVA.
- A cache invalidates all operation.
- A cache line-fill allocation.

Note that this erratum only affects the MBIST software-assisted use case error indication for the ICACHE and the MBISTERR[0] signal functions correctly for the DCACHE, ITCM, and DTCM.

The following conditions are required to cause this erratum:

- The software intends to use the software assisted online MBIST use case.
- The ICACHE is not disabled by software running on the Cortex®-M7 before testing commences.
- The MBIST controller selects an ICACHE memory array for testing, locks the target memory, and testing commences.

This erratum could result in an error not being indicated back to the MBIST controller on the MBISTERR[0] output signal when software assisted use case is used and the ICACHE is not disabled by software before testing commences. This could result in the processor unexpectedly stalling for a long period of time during MBIST testing of the ICACHE memories, without there being a clear indication of the cause of the stall. For this reason, the processor might not make progress as expected, because of the software error, during ICACHE testing.

Workaround

There is no workaround for this erratum.

2.1.3 ECC error causes data corruption when the data cache error bank registers are locked

Description

The data cache contains two error bank registers, DEBR0 and DEBR1. These registers store the locations in the cache that error correcting code (ECC) errors affect and prevent future allocations to those locations.

Software can lock each DEBR, and this prevents the DEBR from being automatically updated when a data cache ECC error is detected.

Because of this erratum, if both DEBR0 and DEBR1 are locked and an ECC error is detected on a cacheable store, then the store data is written onto the bus, but not written into the data cache. This might result in the data cache containing stale data.

Conditions:

- DEBR0 and DEBR1 are locked.
- The wanted address has been allocated to the cache.
- A cacheable store to the wanted address looks up in the cache, and an ECC error is found in the cache set that the store addresses.

This erratum can cause data corruption in the data cache.

Workaround

Software must avoid locking both error bank registers.

2.1.4 Store after cache invalidate without intervening barrier might cause inconsistent memory view

Description

If a cache invalidate operation is followed by a write-through store to an address affected by that operation and a line-fill to that address occurs, then the line-fill might allocate to the cache without the data from the store. Subsequently, that store writes to the bus and leaves the cache with stale data.

The following sequence is required for this erratum to occur:

1. The address of interest is in the cache.
2. One of the following data cache maintenance operations that affects the same cache line as the wanted address is performed.
   - DCCIMVAC.
   - DCCISW.
   - DCIMVAC.
   - DCISW.
3. A write-through store is performed to the wanted address
4. A line-fill to the same cache line of the wanted address occurs for any reason.

There must be no DSB or DMB between the maintenance operation and the store.

If this sequence occurs and certain very specific internal timing conditions are met, then the store data is not merged into the line-fill, but it writes out to the bus. After this has occurred, the line-fill buffer or cache contains stale data.

A subsequent load to the same address of the store might observe stale data in the cache.

Workaround

A DMB must be inserted between the cache maintenance operation and the store.

It is expected that all code should already have this DMB or DSB because there is no implicit ordering between cache maintenance operations and stores.

2.2 System

2.2.1 SWD debug connection is lost when NJTRST is driven low

Description

A low level on the NJTRST input (PB4) also resets the serial wire debug (SWD) unit. If this occurs during a debug session, the connection to the debugger can be lost.

Workaround

Configure PB4 as a GPIO pin at boot time. This configuration disconnects the nJTRST signal from the pin. However, PB4 reconnects to nJTRST if a system reset occurs. To avoid losing the debug connection during a system reset, maintain the PB4 input at a logic high level while the reset is active until the software reconfigures the pin as a GPIO.

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 Unsupported read access with unaligned address

Description

Read access with unaligned address, such as a half-word read access starting at odd address, is not supported.

Workaround

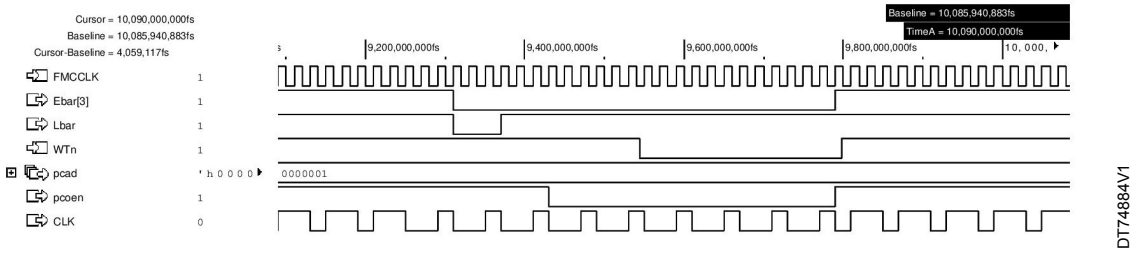
Compile the software that accesses the fmc region with a compiler option that ensures data alignment, such as – *no_unaligned_access*.

2.3.4 Duty cycle variation on memory clock while accessing PSRAM in continuous clock mode

Description

With a clock division of 3 (CLKDIV = 0x02), the FMC continuous clock duty cycle, which is 2/3 high and 1/3 low during read or write transfers, changes to 1/3 high and 2/3 low on the first clock after transfer. The duty cycle change creates a narrow pulse after the data transfer, as illustrated by Figure 1 below.

Figure 1. RTL simulation timing diagram



Workaround

The narrow pulse after a data transfer is not expected to cause a timing violation for PSRAM, as CLK may be off and irrelevant after the transfer is complete.

In case PSRAM needs a continuous clock and $tCLK_{min}$ is not fullfilled, the clock division must be set to 4 (CLKDIV = 0x03).

2.3.5 SDRAM recovery delay inconsistent when TWR[3:0] = 0000

Description

When the application software configures the SDRAM recovery delay TWR[3:0] to 0000, the recovery delay is two clock cycles instead of one when a burst-write access is interrupted by an auto-refresh command.

The behavior is as expected (one-clock cycle delay) when the burst write is interrupted by a precharge command.

This is a documentation error rather than a device limitation.

Workaround

None.

2.4 XSPI

2.4.1 Possible deadlock when a request arrives during the disabling process

Description

If a new transaction request arrives during the disabling process (the ENABLE bit is cleared and the BUSY bit is still set), the following behaviors may be observed, potentially resulting in a deadlock situation:

- The CPU waits for an error response to the transaction, which never arrives.
- The controller waits for the acknowledgment of the new request, which does not arrive since the controller is in the process of being disabled.

Workaround

Before clearing the ENABLE bit, ensure that all current transactions are completed using synchronization barriers.

2.4.2 A read mismatch occurs at the last memory address if a new request arrives before data is read from the I/O pins

Description

The last access to the memory (last memory address) is fetched directly from the I/O pins thanks to an internal signal indicating that the current read transaction has requested access to this last memory address. If a new request arrives before the I/O data pin fetch occurs, the previous read transaction is considered finished (NCS high) and the I/O pin data are flushed, which corrupts the last data read.

Workaround

Avoid accessing the last memory address (last 8-bit aligned memory address in octal mode or last 16-bit aligned memory address in 16-bit mode).

2.4.3 A read mismatch occurs after disabling the controller during a read transaction

Description

If the controller is disabled (the ENABLE bit is cleared) after a read request and before receiving data (the BUSY bit is still set), the FIFO read pointer may be outdated, resulting in a data mismatch after reenabling the controller.

Workaround

Before clearing the ENABLE bit, ensure that all transactions are completed using the synchronization barriers.

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 In certain dual modes, the fixed trigger latency for the injected conversions may not be respected

Description

If the application operates in regular simultaneous or interleaved mode (DUAL[4:0] = 0x06 or 0x07 in the ADCC_CCR register), the injected conversions can be activated either on the master or on the slave ADC. When a fixed trigger latency is configured for injected conversions, the regular conversions on both ADCs are interrupted by the injected trigger. In this case, the previous conversion trigger latency and the current conversion trigger latency may differ by one clock cycle.

Workaround

If the regular simultaneous mode or the interleaved mode is used (DUAL[4:0]=0x06 or 0x07), and the application needs injected conversions with a fixed trigger latency for both ADCs, apply one of the following measures:

- Select another dual mode, either DUAL[4:0] = 0x01 or 0x03.
- Make sure that no injected trigger event occurs when regular conversions are stopped, by setting the ADSTP bit in the ADC_CR register.

2.6.2 In simultaneous regular mode, stopping an injected conversion may shift the next regular conversion master and slave timing by one clock cycle

Description

In simultaneous regular mode (DUAL[4:0] = 0x06 in the ADCC_CCR register), injected conversions can be activated either on ADC1 or ADC2. When regular conversions are ongoing, if an injected conversion stop (JADSTP) occurs at the same time as an injected trigger event, then ADC1 and ADC2 timing may be shifted by one clock cycle.

Workaround

Apply one of the following measures:

- Avoid the following events from occurring simultaneously:
  - Triggering injected conversions
  - Stopping injected conversions
  - Enabling regular conversion in simultaneous regular mode (DUAL[4:0] = 0x06)
- Stop regular conversions before stopping injected conversions.
- Stop injected trigger source before stopping injected conversions.
- Configure DUAL[4:0] to 0x01 or 0x02.

2.7 HRTIM

2.7.1 Delayed idle protection malfunction when IDLESx = 1

Description

Upon delayed idle or balanced idle mode entry, the outputs are forced in their IDLESx state.

However, due to the defect, the IDLESx state may not be applied if the delayed protection is entered during the deadtime duration.

This defect is present only when both:

- The deadtime logic is enabled.
- One of the IDLESx bit is set to 1 (IDLES1 = 0/IDLES2 = 1, or IDLES1 = 1/ IDLES2 = 0), to have one output active during the delayed protection.

The consequence differs with respect to the deadtime polarity:

- With positive deadtime, both outputs may become inactive (as if IDLES1 = IDLES2 = 0).
- With negative deadtime, both outputs may become active potentially creating cross-conduction.

Workaround

None.

2.7.2 Deadtime insertion mode not functional

Description

When the deadtime insertion mode is enabled (DTM[1:0] = 01 or 10 in the HRTIM_DTxR register), deadtime durations are not respected in certain conditions, potentially creating deadtime violations.

The deadtime insertion modes must not be used, and it is mandatory to have DTM[1:0] = 00.

This defect does not affect the legacy deadtime mode (DTM[1:0] = 00 and DTEN = 1).

Workaround

None.

The deadtime insertion mode can be used for temporarily disabling an output. For example, when a half-bridge buck converter starts operating in discontinuous mode, the low side switch must no longer be switched. In this case, it is possible to dynamically update the DTEN bit to disable the complementary operation, since the DTEN bit can be preloaded.

For implementing a cycle-by-cycle protection without deadtime insertion, refer to the example application note AN4539 (HRTIM cookbook).

2.7.3 Negative deadtime not supported in push-pull mode

Description

In push-pull mode, the negative deadtime configuration (STRx = 1 and/or SDTFx = 1 in HRTIM_TIMxCR) provides in a null deadtime.

Workaround

None.

2.7.4 Deadtime slightly shortened in push-pull mode

Description

In push-pull mode, with positive deadtime configuration, the effective deadtime falling duration may be slightly reduced compared to the value programmed in DTFx[8:0] bitfield.

The maximum error is strictly below1 hrtim_ker_ck clock period (31/32 of hrtim_ker_ck).

Workaround

1. The deadtime falling value must be increased by 1 hrtim_ker_ck. This guarantees a safe minimum deadtime value and prevents any risk of cross-conduction. Still, the transition from inactive to active of output 2 may jitter (+0/-1 hrtim_ker_ck).
2. The push-pull guarantees nonoverlapping signals by design. Another option to insert deadtime is to use a compare event to set the output at the beginning of the push or pull phase, after a delay corresponding to the deadtime value.

2.8 LPTIM

2.8.1 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.

2.8.2 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.

2.8.3 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.

2.9 RTC and TAMP

2.9.1 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.

2.9.2 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.

2.9.3 System stuck if some RTC or TAMP registers are written when write protection is enabled

Description

The system becomes stuck if some RTC or TAMP registers bits are written to 1 whereas the write protection is enabled in the PWR controller (= 0). The bits creating the failure are:

- Any bit of the RTC status clear register (RTC_SCR)
- Any bit of the TAMP status clear register (TAMP_SCR)
- The BKERASE bit of the TAMP control register 2 (TAMP_CR2)

This does not occur if the software sequence described in reference manual is correctly executed.

Workaround

None. Only a system reset can recover from this stuck state.

2.10 I2C

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

Description

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

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

The device does not correctly sample the I²C-bus SDA line when $t_{SU;DAT}$ is smaller than one I2C kernel clock (I²C-bus peripheral clock) period: the previous SDA value is sampled instead of the current one. This can result in a wrong receipt of 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 I²C-bus standard, the minimum I2CCLK frequencies are as follows:

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

2.10.2 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 I²C-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.

2.11 USART

2.11.1 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.

2.11.2 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.

2.12 LPUART

2.12.1 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.

2.13 SPI

2.13.1 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.

2.14 FDCAN

2.14.1 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.

2.14.2 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
```

2.15 OTG_FS

2.15.1 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).

2.16 UCPD

2.16.1 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.

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
23-Oct-2026	1	Initial release.

Contents

1 **Summary of device errata** 2
2 **Description of device errata** 4
2.1 Core 4
2.1.1 PLD might perform linefill to address that would generate a MemManage Fault 4
2.1.2 Software programming errors might not be reported for online MBIST access to the ICACHE 5
2.1.3 ECC error causes data corruption when the data cache error bank registers are locked . . . 6
2.1.4 Store after cache invalidate without intervening barrier might cause inconsistent memory view 6
2.2 System 7
2.2.1 SWD debug connection is lost when NJTRST is driven low 7
2.3 FMC 7
2.3.1 Dummy read cycles inserted when reading synchronous memories 7
2.3.2 Wrong data read from a busy NAND memory 7
2.3.3 Unsupported read access with unaligned address 7
2.3.4 Duty cycle variation on memory clock while accessing PSRAM in continuous clock mode . 8
2.3.5 SDRAM recovery delay inconsistent when TWR[3:0] = 0000 8
2.4 XSPI 8
2.4.1 Possible deadlock when a request arrives during the disabling process 8
2.4.2 A read mismatch occurs at the last memory address if a new request arrives before data is read from the I/O pins 9
2.4.3 A read mismatch occurs after disabling the controller during a read transaction 9
2.5 SDMMC 9
2.5.1 Command response and receive data end bits not checked 9
2.6 ADC 9
2.6.1 In certain dual modes, the fixed trigger latency for the injected conversions may not be respected 9
2.6.2 In simultaneous regular mode, stopping an injected conversion may shift the next regular conversion master and slave timing by one clock cycle 10
2.7 HRTIM 10
2.7.1 Delayed idle protection malfunction when IDLESx = 1 10
2.7.2 Deadtime insertion mode not functional 10
2.7.3 Negative deadtime not supported in push-pull mode 11
2.7.4 Deadtime slightly shortened in push-pull mode 11
2.8 LPTIM 11
2.8.1 Device may remain stuck in LPTIM interrupt when entering Stop mode 11
2.8.2 ARRM and CMPM flags are not set when APB clock is slower than kernel clock 12

2.8.3 Interrupt status flag is cleared by hardware upon writing its corresponding bit in LPTIM_DIER register . . . 12
2.9 RTC and TAMP . . . 12
2.9.1 Alarm flag may be repeatedly set when the core is stopped in debug . . . 12
2.9.2 RTC calendar read may return transient incorrect value when BYPSHAD = 0 . . . 13
2.9.3 System stuck if some RTC or TAMP registers are written when write protection is enabled . . . 13
2.10 I2C . . . 13
2.10.1 Wrong data sampling when data setup time ($t_{SU;DAT}$) is shorter than one I2C kernel clock period . . . 13
2.10.2 Spurious bus error detection in controller mode . . . 14
2.11 USART . . . 14
2.11.1 Received data may be corrupted upon clearing the ABREN bit . . . 14
2.11.2 Noise error flag set while ONEBIT is set . . . 14
2.12 LPUART . . . 15
2.12.1 Possible LPUART transmitter issue when using low BRR[15:0] value . . . 15
2.13 SPI . . . 15
2.13.1 RDY output failure at high serial clock frequency . . . 15
2.14 FDCAN . . . 15
2.14.1 Desynchronization under specific condition with edge filtering enabled . . . 15
2.14.2 Tx FIFO messages inverted under specific buffer usage and priority setting . . . 16
2.15 OTG_FS . . . 16
2.15.1 Potential unexpected transfer on the USB bus instead of a zero-length packet . . . 16
2.16 UCPD . . . 17
2.16.1 Ordered set with multiple errors in a single K-code is reported as invalid . . . 17
Important security notice . . . 18
Revision history . . . 19

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.

© 2026 STMicroelectronics – All rights reserved