STM32G071x8/xB device errata

Applicability

This document applies to the part numbers of STM32G071x8/xB 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 RM0444.

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
STM32G071x8	STM32G071C8, STM32G071G8, STM32G071K8, STM32G071R8
STM32G071xB	STM32G071CB, STM32G071EB, STM32G071GB, STM32G071KB, STM32G071RB

Table 2. Device variants

Reference	Silicon revision codes	
	Device marking(1)	REV_ID(2)
STM32G071xx	B	0x2000
	Y	0x2002

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 STM32G071x8/xB device limitations and their status:

A = limitation present, workaround available

N = limitation present, no workaround available

P = limitation present, partial workaround available

"-" = limitation absent

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

Table 3. Summary of device limitations

Function	Section	Limitation	Status	
			Rev. B	Rev. Y
System	2.2.1	Unstable LSI when it clocks RTC or CSS on LSE	P	P
	2.2.2	WUFx wakeup flag wrongly set during configuration	A	A
	2.2.3	Under Level 1 read protection, booting from main flash memory selected through PA14-BOOT0 pin is not functional	N	-
	2.2.4	DMAMUX cannot be synchronized or triggered by EXTI	N	-
	2.2.5	Overwriting with all zeros a flash memory location previously programmed with all ones fails	N	N
	2.2.6	Wakeup from Stop not effective under certain conditions	N	N
	2.2.7	Flash memory PCROP area weakness	N	-
	2.2.8	PC13 signal transitions disturb LSE	N	N
	2.2.9	Device lock upon mismatch of option bytes	P	-
	2.2.11	Corrupted content of the RTC domain due to a missed power-on reset after this domain supply voltage drop	A	A
GPIO	2.3.1	GPIO assigned to DAC cannot be used in output mode when the DAC output is connected to on-chip peripheral	N	N
DMA	2.4.1	DMA disable failure and error flag omission upon simultaneous transfer error and global flag clear	A	A
DMAMUX	2.5.1	SOFx not asserted when writing into DMAMUX_CFR register	N	N
	2.5.2	OFx not asserted for trigger event coinciding with last DMAMUX request	N	N
	2.5.3	OFx not asserted when writing into DMAMUX_RGCFR register	N	N
	2.5.4	Wrong input DMA request routed upon specific DMAMUX_CxCR register write coinciding with synchronization event	A	A
ADC	2.6.1	Overrun flag is not set if EOC reset coincides with new conversion end	P	P
	2.6.2	Writing ADC_CFGR1 register while ADEN bit is set resets RES[1:0] bitfield	A	A
	2.6.3	Out-of-threshold value is not detected in AWD1 Single mode	A	A
	2.6.4	Writing ADC_CFGR2 register while ADEN bit is set resets CKMODE[1:0] bitfield	A	A
	2.6.5	ADC sampling time might be one cycle longer	N	N
	2.6.7	ADC offset may be out of specification	A	-
DAC	2.7.1	Invalid DAC channel analog output if the DAC channel MODE bitfield is programmed before DAC initialization	A	A

Function	Section	Limitation	Status	
			Rev. B	Rev. Y
DAC	2.7.2	DMA underrun flag not set when an internal trigger is detected on the clock cycle of the DMA request acknowledge	N	N
TIM	2.8.1	One-pulse mode trigger not detected in master-slave reset + trigger configuration	P	P
	2.8.2	Consecutive compare event missed in specific conditions	N	N
	2.8.3	Output compare clear not working with external counter reset	P	P
	2.8.4	Bidirectional break mode not working with short pulses	N	N
	2.8.5	TIM1 and TIM15 synchronization trigger might be missed	N	N
	2.8.6	TIM16 and TIM17 are unduly clocked by SYSCLK	N	-
LPTIM	2.9.1	Device may remain stuck in LPTIM interrupt when entering Stop mode	A	A
	2.9.2	Device may remain stuck in LPTIM interrupt when clearing event flag	P	P
RTC and TAMP	2.10.1	Calendar initialization may fail in case of consecutive INIT mode entry	A	A
	2.10.2	Tamper flag not set on LSE failure detection	N	N
I2C	2.11.1	Wrong data sampling when data setup time ($t_{SU;DAT}$) is shorter than one I2C kernel clock period	P	P
	2.11.2	Spurious bus error detection in master mode	A	A
	2.11.3	Spurious master transfer upon own slave address match	P	P
	2.11.5	OVR flag not set in underrun condition	N	N
	2.11.6	Transmission stalled after first byte transfer	A	A
	2.11.7	SDA held low upon SMBus timeout expiry in slave mode	A	A
USART	2.12.1	Anticipated end-of-transmission signaling in SPI slave mode	A	A
	2.12.2	Data corruption due to noisy receive line	A	A
	2.12.4	Received data may be corrupted upon clearing the ABREN bit	A	A
	2.12.5	Noise error flag set while ONEBIT is set	N	N
LPUART	2.13.1	Possible LPUART transmitter issue when using low BRR[15:0] value	P	P
SPI	2.14.1	BSY bit may stay high when SPI is disabled	A	A
	2.14.2	BSY bit may stay high at the end of data transfer in slave mode	A	A
CEC	2.16.1	Missed CEC messages in normal receiving mode	A	A
	2.16.2	Unexpected TXERR flag during a message transmission	A	A
UCPD	2.15.1	TXHRST upon write data underflow corrupting the CRC of the next packet	A	A
	2.15.2	Ordered set with multiple errors in a single K-code is reported as invalid	N	N
	2.15.3	UCPD wrong R_{pdb} default trimming value after power-on	N	N

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

Table 4. Summary of device documentation errata

Function	Section	Documentation erratum
System	2.2.10	Boot select after debug interface connection
ADC	2.6.6	ADC trigger latency parameter
I2C	2.11.4	START bit is cleared upon setting ADDRCF, not upon address match
USART	2.12.3	USART prescaler feature missing in USART implementation section

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®-M0+ core revision r0p1 is available from https://developer.arm.com/documentation/.

2.2 System

2.2.1 Unstable LSI when it clocks RTC or CSS on LSE

Description

The LSI clock can become unstable (duty cycle different from 50 %) and its maximum frequency can become significantly higher than 32 kHz, when:

- LSI clocks the RTC, or it clocks the clock security system (CSS) on LSE (which holds when the LSECSSON bit set), and
- the V_{DD} power domain is reset while the backup domain is not reset, which happens:
  - upon exiting Shutdown mode
  - if V_{BAT} is separate from V_{DD} and V_{DD} goes off then on
  - if V_{BAT} is tied to V_{DD} (internally in the package for products not featuring the VBAT pin, or externally) and a short (< 1 ms) V_{DD} drop under V_{DD}(min) occurs

Workaround

Apply one of the following measures:

- Clock the RTC with LSE or HSE/32, without using the CSS on LSE
- If LSI clocks the RTC or when the LSECSSON bit is set, reset the backup domain upon each V_{DD} power up (when the PWRRSTF flag is set). If V_{BAT} is separate from V_{DD}, also restore the RTC configuration, backup registers and anti-tampering configuration.

2.2.2 WUFx wakeup flag wrongly set during configuration

Description

Upon configuring a wakeup pin (WKUPx), the corresponding wakeup flag (WUFx) might spuriously go high depending on the state and configuration of the wakeup pin.

Workaround

After configuring a wakeup pin, clear its corresponding WUFx flag.

2.2.3 Under Level 1 read protection, booting from main flash memory selected through PA14-BOOT0 pin is not functional

Description

With the flash memory read protection set to level 1 and the boot mode selected through the PA14-BOOT0 pin (BOOT0 function of the pin), an attempt to boot from main flash memory can wrongly be interpreted by the read protection mechanism as an unauthorized access, preventing the user code execution. Booting from main flash memory operates correctly if selected through option bits (nBOOT_SEL and nBOOT0 both set).

Workaround

None.

2.2.4 DMAMUX cannot be synchronized or triggered by EXTI

Description

The EXTI-related DMAMUX synchronization and trigger inputs are wrongly routed to the *it_exti_per(y)* output instead of being routed to the *exti[15:0]* output lines.

The *it_exti_per(y)* signals are not usable for synchronizing and triggering DMAMUX.

Workaround

None.

2.2.5 Overwriting with all zeros a flash memory location previously programmed with all ones fails

Description

Any attempt to re-program with all zeros (0x0000 0000 0000 0000) a flash memory location previously programmed with 0xFFFF FFFF FFFF FFFF fails and the PROGERR flag of the FLASH_SR register is set.

Note: Flash memory locations in the erased state (that is, not programmed) are not affected by this failure. They can be programmed with any value.

Workaround

None.

2.2.6 Wakeup from Stop not effective under certain conditions

Description

With the HSI clock divider bitfield HSIDIV[2:0] set to a value different from 000, the device fails to enter Stop mode when SYSCLK is set to HSE clock.

With the HSI clock divider bitfield HSIDIV[2:0] set to a value different from 000, peripherals with clock request capability fail to wake the device up from Stop modes.

Workaround

None.

2.2.7 Flash memory PCROP area weakness

Description

When the CPU accesses PCROP-protected flash memory areas:

- Fetch requests are allowed and are responded to normally.
- Read access are properly discarded. However, the bus holds and returns the value read during previous successful access.

Workaround

None.

Note: We recommend to use the PCROP protection in the following RDP and PCROP_RDP configurations:

- *RDP = Level 1 and PCROP_RDP = 1*
- *RDP = Level 2*

2.2.8 PC13 signal transitions disturb LSE

Description

The PC13 port toggling disturbs the LSE clock.

Workaround

None.

2.2.9 Device lock upon mismatch of option bytes

Description

An option byte mismatch (for example, due to its failing modification attempt) detection leads to setting the BOOT_LOCK register bit high and the RDP to Level 1 or higher. This disables the debug interface, which then makes the device reprogramming impossible.

Note: On the latest device versions, an option byte mismatch detection results in setting the device in RDP Level 1 with BOOT_LOCK register bit low. This keeps the debug connection available for reprogramming the device.

Workaround

Ensure a safe environment when modifying the option bytes, to prevent any option byte corruption.

2.2.10 Boot select after debug interface connection

Description

Some revisions of the reference manual may omit the following information.

After connecting the debug interface and until the device power-down, the boot source upon reset or wakeup from a low-power mode is determined by the PA14-BOOT0 pin level before connecting the debug interface (stored by the device), as opposed to the actual PA14-BOOT0 pin level. The device power-up restores the operation of the PA14-BOOT0 pin as direct boot source selector.

This is a documentation issue rather than a device limitation.

Workaround

No application workaround is required or applicable.

2.2.11 Corrupted content of the RTC domain due to a missed power-on reset after this domain supply voltage drop

Description

The RTC domain reset may be missed upon a power-on following a power-off, if its supply voltage drops during the power-off phase hitting a window, which is few mV wide before it starts to rise again. In this critical window, the flip-flops are no longer able to safely retain the information and the RTC 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.

This missed reset results in unpredictable values of the RTC domain registers. This may cause a spurious behavior (such as driving the LSCO output pin on PA2 or influencing RTC functions).

Workaround

Apply one of the following measures:

- In the application, let the V_{DD} and V_{BAT} supply voltages fall to a level below 100 mV for more than 200 ms before a new power-on.

- If the above workaround cannot be applied, and the boot follows a power-on reset, erase the RTC domain by software.
  When the application is using shutdown mode, user needs to discriminate between the power-on reset or an exit from a shutdown mode.
  For this purpose, at least one backup register must have been 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 of calculating the CRC of the backup registers: RCC_BDCR and RTC registers, excluding bits modified by HW.
  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 the BORRSTF flag of the RCC_CSR register. If set, the reset is caused by a power on, or is exiting from shutdown mode.
  2. If BORRSTF flag is true, and the shutdown mode is used in the application, check that the backup register value is different from BKP_REG_VAL. When tamper detection is enabled,check that no tamper flag is set. If both conditions are met then the reset is caused by a power-on.
  3. If the reset is caused by a power-on, apply the following sequence:
     a. Enable the PWR clock in the RCC, by setting the PWREN bit.
     b. Enable the RTC domain access in the PWR, by setting the DBP bit.
     c. Reset the RTC domain, by:
        i. Writing 0x0001 0000 in the RCC_BDCR register, which sets the BDRST bit and clears other register bits that might not be reset.
        ii. reading the RCC_BDCR register, to make the reset time long enough
        iii. writing 0x0000 0000 in the RCC_BDCR register, to clear the BDRST bit
     d. Clear the BORRSTF flag by setting the RMVF bit of the RCC_CSR register.

2.3 GPIO

2.3.1 GPIO assigned to DAC cannot be used in output mode when the DAC output is connected to on-chip peripheral

Description

When a DAC output is connected only to an on-chip peripheral, the corresponding GPIO is expected to be available as an output for any other functions.

However, when the DAC output is configured for on-chip peripheral connection only, the GPIO output buffer remains disabled and cannot be used in output mode (GPIO or alternate function). It can still be used in input or analog mode.

Workaround

None.

2.4 DMA

2.4.1 DMA disable failure and error flag omission upon simultaneous transfer error and global flag clear

Description

Upon a data transfer error in a DMA channel x, both the specific TEIFx and the global GIFx flags are raised and the channel x is normally automatically disabled. However, if in the same clock cycle the software clears the GIFx flag (by setting the CGIFx bit of the DMA_IFCR register), the automatic channel disable fails and the TEIFx flag is not raised.

This issue does not occur with ST's HAL software that does not use and clear the GIFx flag when the channel is active.

Workaround

Do not clear GIFx flags when the channel is active. Instead, use HTIFx, TCIFx, and TEIFx specific event flags and their corresponding clear bits.

2.5 DMAMUX

2.5.1 SOFx not asserted when writing into DMAMUX_CFR register

Description

The SOFx flag of the DMAMUX_CSR status register is not asserted if overrun from another DMAMUX channel occurs when the software writes into the DMAMUX_CFR register.

This can happen when multiple DMA channels operate in synchronization mode, and when overrun can occur from more than one channel. As the SOFx flag clear requires a write into the DMAMUX_CFR register (to set the corresponding CSOFx bit), overrun occurring from another DMAMUX channel operating during that write operation fails to raise its corresponding SOFx flag.

Workaround

None. Avoid the use of synchronization mode for concurrent DMAMUX channels, if at least two of them potentially generate synchronization overrun.

2.5.2 OFx not asserted for trigger event coinciding with last DMAMUX request

Description

In the DMAMUX request generator, a trigger event detected in a critical instant of the last-generated DMAMUX request being served by the DMA controller does not assert the corresponding trigger overrun flag OFx. The critical instant is the clock cycle at the very end of the trigger overrun condition.

Additionally, upon the following trigger event, one single DMA request is issued by the DMAMUX request generator, regardless of the programmed number of DMA requests to generate.

The failure only occurs if the number of requests to generate is set to more than two (GNBREQ[4:0] > 00001).

Workaround

Make the trigger period longer than the duration required for serving the programmed number of DMA requests, so as to avoid the trigger overrun condition from occurring on the very last DMA data transfer.

2.5.3 OFx not asserted when writing into DMAMUX_RGCFR register

Description

The OFx flag of the DMAMUX_RGSR status register is not asserted if an overrun from another DMAMUX request generator channel occurs when the software writes into the DMAMUX_RGCFR register. This can happen when multiple DMA channels operate with the DMAMUX request generator, and when an overrun can occur from more than one request generator channel. As the OFx flag clear requires a write into the DMAMUX_RGCFR register (to set the corresponding COFx bit), an overrun occurring in another DMAMUX channel operating with another request generator channel during that write operation fails to raise the corresponding OFx flag.

Workaround

None. Avoid the use of request generator mode for concurrent DMAMUX channels, if at least two channels are potentially generating a request generator overrun.

2.5.4 Wrong input DMA request routed upon specific DMAMUX_CxCR register write coinciding with synchronization event

Description

If a write access into the DMAMUX_CxCR register having the SE bit at zero and SPOL[1:0] bitfield at a value other than 00:

- sets the SE bit (enables synchronization),
- modifies the values of the DMAREQ_ID[5:0] and SYNC_ID[4:0] bitfields, and
- does not modify the SPOL[1:0] bitfield,

and if a synchronization event occurs on the previously selected synchronization input exactly two AHB clock cycles before this DMAMUX_CxCR write, then the input DMA request selected by the DMAREQ_ID[5:0] value before that write is routed.

Workaround

Ensure that the SPOL[1:0] bitfield is at 00 whenever the SE bit is 0. When enabling synchronization by setting the SE bit, always set the SPOL[1:0] bitfield to a value other than 00 with the same write operation into the DMAMUX_CxCR register.

2.6 ADC

2.6.1 Overrun flag is not set if EOC reset coincides with new conversion end

Description

If the EOC flag is cleared by an ADC_DR register read operation or by software during the same APB cycle in which the data from a new conversion are written in the ADC_DR register, the overrun event duly occurs (which results in the loss of either current or new data) but the overrun flag (OVR) may stay low.

Workaround

Clear the EOC flag, by performing an ADC_DR read operation or by software within less than one ADC conversion cycle period from the last conversion cycle end, in order to avoid the coincidence with the end of the new conversion cycle.

2.6.2 Writing ADC_CFGR1 register while ADEN bit is set resets RES[1:0] bitfield

Description

Modifying the ADC_CFGR1 register while ADC is enabled (ADEN set in ADC_CR) resets RES[1:0] to 00 whatever the bitfield previous value.

Workaround

Apply the following sequence:

1. Set ADDIS to disable the ADC, and wait until ADEN is cleared.
2. Program the ADC_CFGR1 register according to the application requirements.
3. Set ADEN bit.

2.6.3 Out-of-threshold value is not detected in AWD1 Single mode

Description

AWD1 analog watchdog does not detect that the result of a converted channel has reached the programmed threshold when the ADC operates in Single mode, performs a sequence of conversions, and one of the converted channels other than the first one is monitored by the AWD1 analog watchdog.

Workaround

Apply one of the following measures:

- Use a conversion sequence of one single channel.
- Configure the monitored channel as the first one of the sequence.

2.6.4 Writing ADC_CFGR2 register while ADEN bit is set resets CKMODE[1:0] bitfield

Description

Modifying the ADC_CFGR2 register while ADC is enabled (ADEN set in ADC_CR) resets CKMODE[1:0] to 00 whatever the bitfield previous value.

Workaround

Apply the following sequence:

1. Set ADDIS to disable the ADC, and wait until ADEN is cleared.
2. Program the ADC_CFGR2 register according to the application requirements.
3. Set ADEN bit.

2.6.5 ADC sampling time might be one cycle longer

Description

For sampling time set to 1.5 or 3.5 cycles, the sampling in a single ADC conversion or in the first conversion of a sequence takes one extra cycle.

Workaround

None.

2.6.6 ADC trigger latency parameter

Description

Some datasheet revisions state incorrectly ADC trigger latency values for CKMODE set to 01, 10, and 11, as 2.75, 2.63, and 3 ADC clock cycles at maximum, respectively.

The correct specification is 6.5, 12.5, and 3.5 PCLK cycles typical, respectively.

This is a documentation issue rather than a product limitation.

Workaround

No application workaround is applicable.

.

2.6.7 ADC offset may be out of specification

Description

When the V_{REF+} voltage is lower than 3.0 V, the ADC hardware calibration may not fully compensate for the offset error caused by the diffusion process variation. When this occurs, the CALFACT register value is either 0x00 or 0x7F and the EO parameter is out of the maximum stated in the device data sheet. To reflect this, the following specification substitutes the one in the device datasheet:

Symbol	Parameter	Condition	Min	Typ	Max	Unit
EO	Offset error	$2\ V < V_{DDA} = V_{REF+} < 3\ V$ $f_{ADC} = 35\ MHz$, $f_s \leq 2.5\ Msps$ T_A = entire range	-	1.5	31	LSB

Symbol	Parameter	Condition	Min	Typ	Max	Unit
EO	Offset error	1.65 V < V_{DDA} = V_{REF+} < 3 V f_{ADC} = 35 MHz, f_s ≤ 2.2 Msps f_{ADC} = 16 MHz; f_s ≤ 1.1 Msps T_A = entire range	-	1.5	50	LSB
ET	Total unadjusted error	2 V < V_{DDA} = V_{REF+} < 3 V f_{ADC} = 35 MHz, f_s ≤ 2.5 Msps T_A = entire range	-	3	33	
		1.65 V < V_{DDA} = V_{REF+} < 3 V f_{ADC} = 35 MHz, f_s ≤ 2.2 Msps f_{ADC} = 16 MHz; f_s ≤ 1.1 Msps T_A = entire range	-	3	52	

Workaround

Apply one of the following measures:

- Use V_{DDA} and V_{REF+} higher than 3V.
- Use the ADC for measuring a difference between two DC voltages or for measuring AC voltages with a head room to V_{REF+} and V_{SSA} greater than the voltage corresponding to EO(max).
- In the hardware application, make V_{REF+} and V_{SSA} available on the ADC GPIO inputs or internally through the DAC output. Then if CALFACT register value indicates 0x00 or 0x7F after the hardware calibration, compensate A/D conversions by software for the ADC offset remaining after the hardware compensation as follows:
  - For CALFACT equal to 0x00:
    1. With the ADC, measure the V_{REF+} voltage through the GPIO channel or DAC channel. Subtract the A/D conversion result from the full-scale value plus one (4096 for a 12-bit ADC), to obtain the offset value. For example, if the A/D conversion result is 4093, the offset value is 3 (3 LSBs).
    2. Add the offset value to any further A/D conversion result. For example, if the A/D conversion result is 1550, the result after this compensation is 1553.
  - For CALFACT equal to 0x7F:
    1. With the ADC, measure the V_{SSA} voltage through the GPIO channel or DAC channel, to obtain the offset value, for example 4 (4 LSBs).
    2. Subtract the offset value from any further A/D conversion result. For example, if the A/D conversion result is 1550, the result after this compensation is 1546.

Note: For better accuracy of the remaining offset measurement, it is recommended to use an average of multiple (for example eight) A/D conversions.

Note: The compensation for the offset remaining after the hardware calibration reduces the effective ADC conversion range.

2.7 DAC

2.7.1 Invalid DAC channel analog output if the DAC channel MODE bitfield is programmed before DAC initialization

Description

When the DAC operates in Normal mode and the DAC enable bit is cleared, writing a value different from 000 to the DAC channel MODE bitfield of the DAC_MCR register before performing data initialization causes the corresponding DAC channel analog output to be invalid.

Workaround

Apply the following sequence:

1. Perform one write access to any data register.
2. Program the MODE bitfield of the DAC_MCR register.

2.7.2 DMA underrun flag not set when an internal trigger is detected on the clock cycle of the DMA request acknowledge

Description

When the DAC channel operates in DMA mode (DMAEN of DAC_CR register set), the DMA channel underrun flag (DMAUDR of DAC_SR register) fails to rise upon an internal trigger detection if that detection occurs during the same clock cycle as a DMA request acknowledge. As a result, the user application is not informed that an underrun error occurred.

This issue occurs when software and hardware triggers are used concurrently to trigger DMA transfers.

Workaround

None.

2.8 TIM

2.8.1 One-pulse mode trigger not detected in master-slave reset + trigger configuration

Description

The failure occurs when several timers configured in one-pulse mode are cascaded, and the master timer is configured in combined reset + trigger mode with the MSM bit set:

OPM = 1 in TIMx_CR1, SMS[3:0] = 1000 and MSM = 1 in TIMx_SMCR.

The MSM delays the reaction of the master timer to the trigger event, so as to have the slave timers cycle-accurately synchronized.

If the trigger arrives when the counter value is equal to the period value set in the TIMx_ARR register, the one-pulse mode of the master timer does not work and no pulse is generated on the output.

Workaround

None. However, unless a cycle-level synchronization is mandatory, it is advised to keep the MSM bit reset, in which case the problem is not present. The MSM = 0 configuration also allows decreasing the timer latency to external trigger events.

2.8.2 Consecutive compare event missed in specific conditions

Description

Every match of the counter (CNT) value with the compare register (CCR) value is expected to trigger a compare event. However, if such matches occur in two consecutive counter clock cycles (as consequence of the CCR value change between the two cycles), the second compare event is missed for the following CCR value changes:

- in edge-aligned mode, from ARR to 0:
  - first compare event: CNT = CCR = ARR
  - second (missed) compare event: CNT = CCR = 0
- in center-aligned mode while up-counting, from ARR-1 to ARR (possibly a new ARR value if the period is also changed) at the crest (that is, when TIMx_RCR = 0):
  - first compare event: CNT = CCR = (ARR-1)
  - second (missed) compare event: CNT = CCR = ARR

- in center-aligned mode while down-counting, from 1 to 0 at the valley (that is, when TIMx_RCR = 0):
  - first compare event: CNT = CCR = 1
  - second (missed) compare event: CNT = CCR = 0

This typically corresponds to an abrupt change of compare value aiming at creating a timer clock single-cycle-wide pulse in toggle mode.

As a consequence:

- In toggle mode, the output only toggles once per counter period (squared waveform), whereas it is expected to toggle twice within two consecutive counter cycles (and so exhibit a short pulse per counter period).
- In center mode, the compare interrupt flag does note rise and the interrupt is not generated.

Note: The timer output operates as expected in modes other than the toggle mode.

Workaround

None.

2.8.3 Output compare clear not working with external counter reset

Description

The output compare clear event (ocref_clr) is not correctly generated when the timer is configured in the following slave modes: Reset mode, Combined reset + trigger mode, and Combined gated + reset mode.

The PWM output remains inactive during one extra PWM cycle if the following sequence occurs:

1. The output is cleared by the ocref_clr event.
2. The timer reset occurs before the programmed compare event.

Workaround

Apply one of the following measures:

- Use BKIN (or BKIN2 if available) input for clearing the output, selecting the Automatic output enable mode (AOE = 1).
- Mask the timer reset during the PWM ON time to prevent it from occurring before the compare event (for example with a spare timer compare channel open-drain output connected with the reset signal, pulling the timer reset line down).

2.8.4 Bidirectional break mode not working with short pulses

Description

The TIM_BKIN and TIM_BKIN2 I/Os can be configured in bidirectional mode using the BKBID and BK2BID bits in the TIMx_BDTR register, to be forced to 0 when a break/break2 event occurs. The bidirectional break/break2 mode is not functional when the pulse width on break/break2 input is lower than two tim_ker_clk periods.

This limitation is also valid when software break events are generated (the break event is correctly generated internally but not reflected on break inputs).

Workaround

None.

For applications that can afford some latency in bidirectional break mode, the break interrupt can eventually be enabled, for the CPU to verify the break input state and force it to zero when a break/break2 event occurred.

2.8.5 TIM1 and TIM15 synchronization trigger might be missed

Description

A slave timer may miss synchronization trigger signal generated by its respective master timer TIM1 or TIM15 if the master timer clock is faster than the slave timer clock.

Workaround

None.

2.8.6 TIM16 and TIM17 are unduly clocked by SYSCLK

Description

The timers TIM16 and TIM17 are unduly clocked by SYSCLK instead of being clocked by the timer clock TIMPCLK. As a consequence, they do not reflect AHB and APB prescaler settings.

The TIM16 and TIM17 are fully functional as long as the SYSCLK-to-PCLK frequency ratio remains smaller than or equal to four.

Workaround

None.

2.9 LPTIM

2.9.1 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.

2.9.2 Device may remain stuck in LPTIM interrupt when clearing event flag

Description

This limitation occurs when the LPTIM is configured in interrupt mode (at least one interrupt is enabled) and the software clears any flag in LPTIM_ISR register by writing its corresponding bit in LPTIM_ICR register. If the interrupt status flag corresponding to a disabled interrupt is cleared simultaneously with a new event detection, the set and clear commands might reach the APB domain at the same time, leading to an asynchronous interrupt signal permanently stuck high.

This issue can occur either during an interrupt subroutine execution (where the flag clearing is usually done), or outside an interrupt subroutine.

Consequently, the firmware remains stuck in the LPTIM interrupt routine, and the device cannot enter Stop mode.

Workaround

To avoid this issue, it is strongly advised to follow the recommendations listed below:

- Clear the flag only when its corresponding interrupt is enabled in the interrupt enable register.
- If for specific reasons, it is required to clear some flags that have corresponding interrupt lines disabled in the interrupt enable register, it is recommended to clear them during the current subroutine prior to those which have corresponding interrupt line enabled in the interrupt enable register.
- Flags must not be cleared outside the interrupt subroutine.

Note: The standard clear sequence implemented in the HAL_LPTIM_IRQHandler in the STM32Cube is considered as the proper clear sequence.

2.10 RTC and TAMP

2.10.1 Calendar initialization may fail in case of consecutive INIT mode entry

Description

If the INIT bit of the RTC_ICSR register is set between one and two RTCCLK cycles after being cleared, the INITF flag is set immediately instead of waiting for synchronization delay (which should be between one and two RTCCLK cycles), and the initialization of registers may fail. Depending on the INIT bit clearing and setting instants versus the RTCCLK edges, it can happen that, after being immediately set, the INITF flag is cleared during one RTCCLK period then set again. As writes to calendar registers are ignored when INITF is low, a write occurring during this critical period might result in the corruption of one or more calendar registers.

Workaround

After existing the initialization mode, clear the BYPSHAD bit (if set) then wait for RSF to rise, before entering the initialization mode again.

Note: It is recommended to write all registers in a single initialization session to avoid accumulating synchronization delays.

2.10.2 Tamper flag not set on LSE failure detection

Description

With the timestamp on tamper event enabled (the TAMPTS bit of the RTC_CR register set), the LSE failure detection (LSE clock stopped) event connected to the internal tamper 3 fails to raise the ITAMP3F and ITAMP3MF flags, although it duly erases or blocks (depending on the internal tamper 3 configuration) the backup registers and other device secrets, and the RTC and TAMP peripherals resume normally upon the LSE restart.

Note: As expected in this particular case, the TSF and TSMF flags remain low as long as LSE is stopped as they require running RTCCLK clock to operate.

Workaround

None.

2.11 I2C

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

Description

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

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

The device does not correctly sample the I^2C-bus SDA line when $t_{SU;DAT}$ is smaller than one I2C kernel clock (I^2C-bus peripheral clock) period: the previous SDA value is sampled instead of the current one. This can result in a wrong receipt of 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^2C-bus standard, the minimum I2CCLK frequencies are as follows:

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

2.11.2 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.

2.11.3 Spurious master transfer upon own slave address match

Description

When the device is configured to operate at the same time as master and slave (in a multi- master I^2C-bus application), a spurious master transfer may occur under the following condition:

- Another master on the bus is in process of sending the slave address of the device (the bus is busy).
- The device initiates a master transfer by bit set before the slave address match event (the ADDR flag set in the I2C_ISR register) occurs.
- After the ADDR flag is set:
  - the device does not write I2C_CR2 before clearing the ADDR flag, or
  - the device writes I2C_CR2 earlier than three I2C kernel clock cycles before clearing the ADDR flag

In these circumstances, even though the START bit is automatically cleared by the circuitry handling the ADDR flag, the device spuriously proceeds to the master transfer as soon as the bus becomes free. The transfer configuration depends on the content of the I2C_CR2 register when the master transfer starts. Moreover, if the I2C_CR2 is written less than three kernel clocks before the ADDR flag is cleared, the I2C peripheral may fall into an unpredictable state.

Workaround

Upon the address match event (ADDR flag set), apply the following sequence.

Normal mode (SBC = 0):

1. Set the ADDRCF bit.
2. Before Stop condition occurs on the bus, write I2C_CR2 with the START bit low.

Slave byte control mode (SBC = 1):

1. Write I2C_CR2 with the slave transfer configuration and the START bit low.
2. Wait for longer than three I2C kernel clock cycles.
3. Set the ADDRCF bit.
4. Before Stop condition occurs on the bus, write I2C_CR2 again with its current value.

The time for the software application to write the I2C_CR2 register before the Stop condition is limited, as the clock stretching (if enabled), is aborted when clearing the ADDR flag.

Polling the BUSY flag before requesting the master transfer is not a reliable workaround as the bus may become busy between the BUSY flag check and the write into the I2C_CR2 register with the START bit set.

2.11.4 START bit is cleared upon setting ADDRCF, not upon address match

Description

Some reference manual revisions may state that the START bit of the I2C_CR2 register is cleared upon slave address match event.

Instead, the START bit is cleared upon setting, by software, the ADDRCF bit of the I2C_ICR register, which does not guarantee the abort of master transfer request when the device is being addressed as slave. This product limitation and its workaround are the subject of a separate erratum.

Workaround

No application workaround is required for this description inaccuracy issue.

2.11.5 OVR flag not set in underrun condition

Description

In slave transmission with clock stretching disabled (NOSTRETCH = 1 in the I2C_CR1 register), an underrun condition occurs if the current byte transmission is completed on the I^2C bus, and the next data is not yet written in the TXDATA[7:0] bitfield. In this condition, the device is expected to set the OVR flag of the I2C_ISR register and send 0xFF on the bus.

However, if the I2C_TXDR is written within the interval between two I2C kernel clock cycles before and three APB clock cycles after the start of the next data transmission, the OVR flag is not set, although the transmitted value is 0xFF.

Workaround

None.

2.11.6 Transmission stalled after first byte transfer

Description

When the first byte to transmit is not prepared in the TXDATA register, two bytes are required successively, through TXIS status flag setting or through a DMA request. If the first of the two bytes is written in the I2C_TXDR register in less than two I2C kernel clock cycles after the TXIS/DMA request, and the ratio between APB clock and I2C kernel clock frequencies is between 1.5 and 3, the second byte written in the I2C_TXDR is not internally detected. This causes a state in which the I2C peripheral is stalled in master mode or in slave mode, with clock stretching enabled (NOSTRETCH = 0). This state can only be released by disabling the peripheral (PE = 0) or by resetting it.

Workaround

Apply one of the following measures:

- Write the first data in I2C_TXDR before the transmission starts.
- Set the APB clock frequency so that its ratio with respect to the I2C kernel clock frequency is lower than 1.5 or higher than 3.

2.11.7 SDA held low upon SMBus timeout expiry in slave mode

Description

For the slave mode, the SMBus specification defines $t_{TIMEOUT}$ (detect clock low timeout) and $t_{LOW:SEXT}$ (cumulative clock low extend time) timeouts. When one of them expires while the I2C peripheral in slave mode drives SDA low to acknowledge either its address or a data transmitted by the master, the device is expected to report such an expiry and release the SDA line.

However, although the device duly reports the timeout expiry, it fails to release SDA. This stalls the I^2C bus and prevents the master from generating RESTART or STOP condition.

Workaround

When a timeout is reported in slave mode (TIMEOUT bit of the I2C_ISR register is set), apply this sequence:

1. Wait until the frame is expected to end.
2. Read the STOPF bit of the I2C_ISR register. If it is low, reset the I2C kernel by clearing the PE bit of the I2C_CR1 register.
3. Wait for at least three APB clock cycles before enabling again the I2C peripheral.

2.12 USART

2.12.1 Anticipated end-of-transmission signaling in SPI slave mode

Description

In SPI slave mode, at low USART baud rate with respect to the USART kernel and APB clock frequencies, the *transmission complete* flag TC of the USARTx_ISR register may unduly be set before the last bit is shifted on the transmit line.

This leads to data corruption if, based on this anticipated end-of-transmission signaling, the application disables the peripheral before the last bit is transmitted.

Workaround

Upon the TC flag rise, wait until the clock line remains idle for more than the half of the communication clock cycle. Then only consider the transmission as ended.

2.12.2 Data corruption due to noisy receive line

Description

In all modes, except synchronous slave mode, 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

Apply one of the following measures:

- Either use a noiseless receive line, or
- add a filter to remove the glitches if the receive line is noisy.

2.12.3 USART prescaler feature missing in USART implementation section

Description

Some reference manual revisions may omit the information that the USART prescaler is not present in all USART instances. This information is provided in the USART implementation section of the corresponding reference manual.

This is a documentation issue rather than a product limitation.

Workaround

No application workaround is required or applicable.

2.12.4 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.

2.12.5 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.

2.13 LPUART

2.13.1 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.

2.14 SPI

2.14.1 BSY bit may stay high when SPI is disabled

Description

The BSY flag may remain high upon disabling the SPI while operating in:

- master transmit mode and the TXE flag is low (data register full).
- master receive-only mode (simplex receive or half-duplex bidirectional receive phase) and an SCK strobing edge has not occurred since the transition of the RXNE flag from low to high.
- slave mode and NSS signal is removed during the communication.

Workaround

When the SPI operates in:

- master transmit mode, disable the SPI when TXE = 1 and BSY = 0.
- master receive-only mode, ignore the BSY flag.
- slave mode, do not remove the NSS signal during the communication.

2.14.2 BSY bit may stay high at the end of data transfer in slave mode

Description

BSY flag may sporadically remain high at the end of a data transfer in slave mode. This occurs upon coincidence of internal CPU clock and external SCK clock provided by master.

In such an event, if the software only relies on BSY flag to detect the end of SPI slave data transaction (for example to enter low-power mode or to change data line direction in half-duplex bidirectional mode), the detection fails.

As a conclusion, the BSY flag is unreliable for detecting the end of data transactions.

Workaround

Depending on SPI operating mode, use the following means for detecting the end of transaction:

- When NSS hardware management is applied and NSS signal is provided by master, use NSS flag.
- In SPI receiving mode, use the corresponding RXNE event flag.
- In SPI transmit-only mode, use the BSY flag in conjunction with a timeout expiry event. Set the timeout such as to exceed the expected duration of the last data frame and start it upon TXE event that occurs with the second bit of the last data frame. The end of the transaction corresponds to either the BSY flag becoming low or the timeout expiry, whichever happens first.

Prefer one of the first two measures to the third as they are simpler and less constraining.

Alternatively, apply the following sequence to ensure reliable operation of the BSY flag in SPI transmit mode:

1. Write last data to data register.
2. Poll the TXE flag until it becomes high, which occurs with the second bit of the data frame transfer.
3. Disable SPI by clearing the SPE bit mandatorily before the end of the frame transfer.
4. Poll the BSY bit until it becomes low, which signals the end of transfer.

Note: The alternative method can only be used with relatively fast CPU speeds versus relatively slow SPI clocks or/and long last data frames. The faster is the software execution, the shorter can be the duration of the last data frame.

2.15 UCPD

2.15.1 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.

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

Description

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

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

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

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

Workaround

None.

2.15.3 UCPD wrong R_{pdb} default trimming value after power-on

Description

When the device is used in application supporting dead battery mode of USB Type-C®, the R_{pdb} pull-down resistor is expected to be 5.1 kΩ ±20% for the device powered down or in the process of booting and 5.1 kΩ ±10% when it is powered and running, as per the USB Type-C® specification.

However, upon powering the device up, the R_{pdb} values spuriously change from 5.1 kΩ ±20% to 3.9 kΩ ±20% in the instant when V_{DD} crosses the V_{POR} threshold and keep that value until the software validates the configuration through the UCPDx_STROBE bits of the SYSCFG_CFGR1 register.

This behavior remains compliant with the USB Type-C® specification revision 1.3 as long as the duration of this state is shorter than 100 ms ($t_{CCDebounce}$ parameter).

Workaround

In applications supporting the dead battery mode of USB Type-C®, validate the configuration of the pull-down resistors, through UCPDx_STROBE bits, within the $t_{CCDebounce}$ time after power-on, to stay compliant with the USB Type-C® specification.

2.16 CEC

2.16.1 Missed CEC messages in normal receiving mode

Description

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

The listen mode operates correctly.

Workaround

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

2.16.2 Unexpected TXERR flag during a message transmission

Description

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

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

Workaround

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

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

Important security notice

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

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

Revision history

Table 5. **Document revision history**

Date	Version	Changes
14-Nov-2018	1	Initial release.
16-Jan-2020	2	Added errata: • DMAMUX cannot be synchronized or triggered by EXTI • Overwriting with all zeros a flash memory location previously programmed with all ones fails • GPIO assigned to DAC cannot be used in output mode when the DAC output is connected to on-chip peripheral • Overrun flag is not set if EOC reset coincides with new conversion end • Writing ADC_CFGR1 register while ADEN bit is set resets RES[1:0] bitfield • Out-of-threshold value is not detected in AWD1 Single mode • TIM1 and TIM15 synchronization trigger might be missed • TIM16 and TIM17 are unduly clocked by SYSCLK • Data corruption due to noisy receive line • USART prescaler feature missing in USART implementation section Added .
01-Feb-2021	3	Added errata: • Wakeup from Stop not effective under certain conditions • Flash memory PCROP area weakness • PC13 signal transitions disturb LSE • ADC sampling time might be one cycle longer • ADC trigger latency parameter • Consecutive compare event missed in specific conditions • Output compare clear not working with external counter reset Modified errata: • Device may remain stuck in LPTIM interrupt when entering Stop mode • Device may remain stuck in LPTIM interrupt when clearing event flag
17-Oct-2022	4	Added Section Important security notice. Added errata: • Boot select after debug interface connection • Corrupted content of the RTC domain due to a missed power-on reset after this domain supply voltage drop Updated errata: • Overwriting with all zeros a flash memory location previously programmed with all ones fails • Writing ADC_CFGR1 register while ADEN bit is set resets RES[1:0] bitfield • ADC offset may be out of specification • Device may remain stuck in LPTIM interrupt when clearing event flag
23-Nov-2023	5	Added errata: • **System:** Device lock upon mismatch of option bytes • **ADC:** Writing ADC_CFGR2 register while ADEN bit is set resets CKMODE[1:0] bitfield • **DAC:** Invalid DAC channel analog output if the DAC channel MODE bitfield is programmed before DAC initialization • DMA underrun flag not set when an internal trigger is detected on the clock cycle of the DMA request acknowledge • **TIM:** Bidirectional break mode not working with short pulses • **RTC:** Tamper flag not set on LSE failure detection • **I2C:** START bit is cleared upon setting ADDRCF, not upon address match • OVR flag not set in underrun condition • Transmission stalled after first byte transfer

Date	Version	Changes
		• SDA held low upon SMBus timeout expiry in slave mode • **USART:** Anticipated end-of-transmission signaling in SPI slave mode • Received data may be corrupted upon clearing the ABREN bit • Noise error flag set while ONEBIT is set • **LPUART:** Possible LPUART transmitter issue when using low BRR[15:0] value • **CEC:** Missed CEC messages in normal receiving mode • Unexpected TXERR flag during a message transmission • **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 Modified document title. Modified errata: • **System:** Unstable LSI when it clocks RTC or CSS on LSE (PWRRST flag name) • **ADC:** ADC offset may be out of specification (workaround: added possibility of using DAC output to provide V_{REF+} internally to the ADC) • **LPTIM:** Device may remain stuck in LPTIM interrupt when entering Stop mode • **USART:** Data corruption due to noisy receive line (workaround now available)

Contents