STM32C5A3xG, STM32C593xx, and STM32C591xx device errata

Applicability

This document applies to the part numbers of STM32C5A3xG, STM32C593xx, and STM32C591xx 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 RM0522.

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
STM32C5A3xG	STM32C5A3ZG, STM32C5A3VG, STM32C5A3RG, STM32C5A3MG, STM32C5A3KG, STM32C5A3CG
STM32C593xx	STM32C593ZG, STM32C593VG, STM32C593RG, STM32C593MG, STM32C593KG, STM32C593CG, STM32C593ZE, STM32C593VE, STM32C593RE, STM32C593ME, STM32C593KE, STM32C593CE
STM32C591xx	STM32C591ZG, STM32C591VG, STM32C591RG, STM32C591MG, STM32C591KG, STM32C591CG, STM32C591CE, STM32C591ZE, STM32C591VE, STM32C591RE, STM32C591ME, STM32C591KE

Table 2. Device variants

Reference	Silicon revision codes	
	Device marking(1)	DIE_ID
STM32C5A3xG/593xG/591xx	Z	0x45A

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

1 Summary of device errata

The following table gives a quick reference to the STM32C5A3xG, STM32C593xx, and STM32C591xx 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	Access permission faults are prioritized over unaligned device memory faults	N
System	2.2.1	Low-speed external clock in analog bypass mode might not work properly	A
	2.2.2	LSE crystal oscillator may be disturbed by transitions on PC13	N
	2.2.3	LSE low drive mode is not functional	N
XSPI	2.3.1	Possible deadlock when a request arrives during the disabling process	A
	2.3.2	A read mismatch occurs at the last memory address if a new request arrives before data is read from the I/O pins	A
	2.3.3	A read mismatch occurs after disabling the controller during a read transaction	A
ADC	2.4.1	In certain dual modes, the fixed trigger latency for the injected conversions may not be respected	A
	2.4.2	In simultaneous regular mode, stopping an injected conversion may shift the next regular conversion master and slave timing by one clock cycle	A
	2.4.3	Injected conversions do not work when SMPTRIG bit is set, regular conversions start then stop, and SMPTRIG bit is cleared	A
RNG	2.5.1	RNG may report continuous seed errors in specific environmental conditions	N
LPTIM	2.7.1	Device may remain stuck in LPTIM interrupt when entering Stop mode	A
	2.7.2	ARRM and CMPM flags are not set when APB clock is slower than kernel clock	A
	2.7.3	Interrupt status flag is cleared by hardware upon writing its corresponding bit in LPTIM_DIER register	N
RTC	2.8.1	Alarm flag may be repeatedly set when the core is stopped in debug	N
	2.8.2	RTC wrong calendar read value through shadow registers	A
I2C	2.9.1	Wrong data sampling when data setup time ($t_{SU;DAT}$) is shorter than one I2C kernel clock period	P
	2.9.2	Spurious bus error detection in controller mode	A
USART	2.10.1	Received data may be corrupted upon clearing the ABREN bit	A
	2.10.2	Noise error flag set while ONEBIT is set	N
LPUART	2.11.1	Possible LPUART transmitter issue when using low BRR[15:0] value	P
SPI	2.12.1	RDY output failure at high serial clock frequency	N
FDCAN	2.13.1	Desynchronization under specific condition with edge filtering enabled	A

Function	Section	Limitation	Status
			Rev. Z
FDCAN	2.13.2	Tx FIFO messages inverted under specific buffer usage and priority setting	A
USB	2.14.1	Buffer description table update completes after CTR interrupt triggers	A

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

Table 4. Summary of device documentation errata

Function	Section	Documentation erratum
SAES	2.6.1	Data transfer from TAMP_BKPxR to key registers must be done only in ascending order when KEYSEL[2:0] is set to 010 or 100

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®-M33 core revision r0p1 is available from http://infocenter.arm.com.

2.1.1 Access permission faults are prioritized over unaligned device memory faults

Description

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

The failure occurs when the MPU is enabled and:

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

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

Workaround

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

2.2 System

2.2.1 Low-speed external clock in analog bypass mode might not work properly

Description

While the LSE bypass in analog mode is selected (LSEBYP = 1 and LSEEXT = 0 in the RCC_BDCR register), the LSE clock might not work properly.

Workaround

Do not use the LSE bypass in analog mode. Instead, use the digital LSE bypass mode (LSEBYP = 1 and LSEEXT = 1 in the RCC_BDCR register).

2.2.2 LSE crystal oscillator may be disturbed by transitions on PC13

Description

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

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

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

Workaround

Avoid toggling PC13 when LSE is used on LQFP packages.

2.2.3 LSE low drive mode is not functional

Description

The LSE oscillator may not start or may stop in low drive mode (LSEDRV = 00). Using this mode is forbidden.

Workaround

None.

2.3 XSPI

2.3.1 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.

2.3.2 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).

2.3.3 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.

2.4 ADC

2.4.1 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.

2.4.2 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.

2.4.3 Injected conversions do not work when SMPTRIG bit is set, regular conversions start then stop, and SMPTRIG bit is cleared

Description

Injected conversions do not work after the following sequence of events:

1. The SMPTRIG bit of the ADC_CFGR2 register is set (sampling time control trigger mode enabled).
2. Regular conversions are started then stopped.
3. The SMPTRIG is cleared.

Workaround

If SMPTRIG is set, disable the ADC, clear the SMPTRIG, and enable again the ADC before starting injected conversions.

2.5 RNG

2.5.1 RNG may report continuous seed errors in specific environmental conditions

Description

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

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

Workaround

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

2.6 SAES

2.6.1 Data transfer from TAMP_BKPxR to key registers must be done only in ascending order when KEYSEL[2:0] is set to 010 or 100

Description

The KEYSEL[2:0] bitfield of the SAES_CR register defines the source of the key information to use in the SAES cryptographic core:

- When KEYSEL[2:0] is set to 010, the boot hardware key (BHK), stored in tamper-resistant secure backup registers, is entirely transferred into the key registers upon a secure application performing a single read of all TAMP_BKPxR registers (x = 0 to 3 for KEYSIZE = 0, x = 0 to 7 for KEYSIZE = 1).
- When KEYSEL[2:0] is set to 100, the XOR combination of DHUK and BHK is entirely transferred into the key registers upon a secure application performing a single read of all TAMP_BKPxR registers (x = 0 to 3 for KEYSIZE = 0, x = 0 to 7 for KEYSIZE = 1).

Some revisions of the reference manual may wrongly specify that the read operation can be performed either in ascending or descending order, while it must be performed always in **ascending** order.

This is a documentation issue rather than a product limitation.

Workaround

No application workaround is required, provided that the read operation to the TAMP_BKPxR registers is always done in ascending order.

2.7 LPTIM

2.7.1 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.

2.7.2 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.

2.7.3 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.

2.8 RTC

2.8.1 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.

2.8.2 RTC wrong calendar read value through shadow registers

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 eliminate the risk of failure, set the BYPSHAD control bit to 1 (bypassing the shadow registers) and read the registers twice. Then, compare the two readings. If the values differ, repeat the process until matching results are obtained. Refer to the reference manual section *Reading the calendar - When the BYPSHAD control bit is set in the RTC_CR register (bypass shadow registers)* for more details.

2.9 I2C

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

Description

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

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

The device does not correctly sample the I^2C-bus SDA line when $t_{SU;DAT}$ is smaller than one I2C kernel clock (I^2C-bus peripheral clock) period: the previous SDA value is sampled instead of the current one. This can result in a wrong receipt of target address, data byte, or acknowledge bit.

Workaround

Increase the I2C kernel clock frequency to get I2C kernel clock period within the transmitter minimum data setup time. Alternatively, increase transmitter's minimum data setup time. If the transmitter setup time minimum value corresponds to the minimum value provided in the I^2C-bus standard, the minimum I2CCLK frequencies are as follows:

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

2.9.2 Spurious bus error detection in controller mode

Description

In controller mode, a bus error can be detected spuriously, with the consequence of setting the BERR flag of the I2C_SR register and generating bus error interrupt if such interrupt is enabled. Detection of bus error has no effect on the I^2C-bus transfer in controller mode and any such transfer continues normally.

Workaround

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

2.10 USART

2.10.1 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.

2.10.2 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.

2.11 LPUART

2.11.1 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.

2.12 SPI

2.12.1 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.

2.13 FDCAN

2.13.1 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.

2.13.2 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
```

2.14 USB

2.14.1 Buffer description table update completes after CTR interrupt triggers

Description

During OUT transfers, the correct transfer interrupt (CTR) is triggered a little before the last USB SRAM accesses have completed. If the software responds quickly to the interrupt, the full buffer contents may not be correct.

Workaround

Software should ensure that a small delay is included before accessing the SRAM contents. This delay should be 800 ns in Full Speed mode and 6.4 µs in Low Speed mode.

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
10-Feb-2026	1	Initial release.

Contents