

## STM32L476xx/STM32L486xx device limitations

### Silicon identification

This errata sheet applies to revisions '3' and '4' of the STMicroelectronics STM32L476xx and STM32L486xx products. The STM32L4x6xx family features an ARM® 32-bit Cortex®-M4 core.

[Section 2](#) gives a detailed description of the product silicon limitations.

The full list of part numbers is shown in [Table 2](#). The products are identifiable by the revision code marked below the order code on the device package, as shown in [Table 1](#).

**Table 1. Device identification<sup>(1)</sup>**

Sales type	Revision code marked on the device <sup>(2)</sup>
STM32L4x6xx	'3' '4'

1. The REV\_ID bits in the DBGMCU\_IDCODE register show the revision code of the device (see the reference manual RM0351 for details on how to find the revision code).
2. Refer to datasheet for details on how to identify the revision code according to the packages.

**Table 2. Device summary**

Reference	Part numbers
STM32L476xx	STM32L476RC, STM32L476VC STM32L476JE, STM32L476ME, STM32L476QE, STM32L476RE, STM32L476VE, STM32L476ZE STM32L476JG, STM32L476MG, STM32L476QG, STM32L476RG, STM32L476VG, STM32L476ZG
STM32L486xx	STM32L486JG, STM32L486QG, STM32L486RG, STM32L486VG, STM32L486ZG

# Contents

<b>1</b>	<b>ARM® 32-bit Cortex®-M4 FPU core limitations</b>	<b>8</b>
1.1	Cortex®-M4 FPU core interrupted loads to stack pointer can cause erroneous behavior	8
<b>2</b>	<b>STM32L4x6xx silicon limitations</b>	<b>9</b>
2.1	System limitations	13
2.1.1	Write operation in the Flash memory while it is not ready (Flash memory in power-down) is not correctly handled	13
2.1.2	The configuration of the I/Os not available in WLCSP package can be modified by software	13
2.1.3	Some I/Os must not be used as output when $VDDA > VDD + 0.6\text{ V}$ or $VDDUSB > VDD + 0.6\text{ V}$ or $VLCD > VDD + 0.6\text{ V}$	13
2.1.4	Bootloader: SPI and CAN interfaces are not supported	14
2.1.5	SRAM2 read access while the SRAM2 hardware erase is ongoing is not correctly supported	14
2.1.6	Extra consumption when $VDDA$ is below $0.7\text{ V}$	14
2.1.7	System clock is stopped during few $\mu\text{s}$ when switching between Low-power run and Run modes	15
2.1.8	PH0/PH1 is controlled by the GPIOH registers when HSE is enabled	15
2.1.9	PWR_CR4 register write access may not be completed if a Low-power mode is entered just after the write operation	15
2.1.10	HSI user trim is limited on some samples	15
2.1.11	Option byte loading can fail if MSI frequency is greater than $8\text{ MHz}$	16
2.1.12	MSI at high frequency ranges cannot be used as wakeup from Stop 0 clock source	17
2.1.13	PLL may not lock if VCO is below $96\text{ MHz}$ and temperature is below $0^{\circ}\text{C}$	17
2.1.14	Full JTAG configuration without NJTRST pin cannot be used	17
2.1.15	MSIRDY flag issue preventing entry in low power mode	17
2.2	FMC peripheral limitations	19
2.2.1	Dummy read cycles inserted when reading synchronous memories	19
2.2.2	Data corruption during burst read from FMC synchronous memory	19
2.2.3	FMC bank switching to asynchronous bank for write	19
2.2.4	Read burst access of 9 words or more is not supported by FMC	20
2.3	QUADSPI peripheral limitations	20
2.3.1	QUADSPI_BK1_IO1 is always an input when the command is sent in dual or quad SPI mode	20

2.3.2	Hard fault is not generated in case of out-of-range memory-mapped access . . . . .	21
2.3.3	Extra data written in the FIFO at the end of a read transfer . . . . .	21
2.4	ADC peripheral limitations . . . . .	21
2.4.1	Injected queue of context is not available in case of JQM=0 . . . . .	21
2.4.2	ADC triggers from EXTI require external interrupt management . . . . .	22
2.4.3	DMA2 channels 2 and 3 (respectively) cannot be used when the ADC2 and ADC3 requests are selected on DMA2 channels 4 and 5 . . . . .	22
2.4.4	Wrong ADC conversion results when delay between calibration and first conversion or between 2 consecutive conversions is too long . . . . .	22
2.4.5	Burst read or write accesses are not supported by the ADC . . . . .	23
2.5	DFSDM peripheral limitations . . . . .	23
2.5.1	RDATACH[2:0] status bits are not implemented . . . . .	23
2.5.2	New regular channel selection taken into account at the end of an injected conversion when a regular conversion is pending . . . . .	23
2.5.3	DFSDM triggers from timers can be missed in specific conditions . . . . .	23
2.5.4	DFSDM triggers from EXTI require external interrupt management . . . . .	24
2.6	I2C peripheral limitations . . . . .	24
2.6.1	I2C Fast-mode Plus drive is not available on all SDA/SCL I/Os . . . . .	24
2.6.2	Wrong behavior related with MCU Stop mode when wakeup from Stop mode by I2C peripheral is disabled . . . . .	24
2.6.3	Wrong data sampling when data set-up time (tSU;DAT) is smaller than one I2CCLK period . . . . .	25
2.6.4	Spurious Bus Error detection in master mode . . . . .	25
2.6.5	I2C3 analog filters requires that both PC0/PC1 or both PG7/PG8 are configured as I2C3 alternate function . . . . .	25
2.7	SDMMC peripheral limitations . . . . .	26
2.7.1	Wrong CCRCFAIL status after a response without CRC is received . . . . .	26
2.7.2	Clock division per 255 is not possible . . . . .	26
2.7.3	Wait for response bits “10” configuration does not work correctly . . . . .	26
2.7.4	MMC stream write of less than 8 bytes does not work correctly . . . . .	26
2.8	bxCAN peripheral limitations . . . . .	27
2.8.1	bxCAN time triggered mode not supported . . . . .	27
2.9	OTG_FS peripheral limitations . . . . .	27
2.9.1	Suspend mode robustness marginality after receiving corrupted packet from Host . . . . .	27
2.9.2	Bits OTG_FS_GLPMPCFG[6:2] are not write-protected in Device mode . . . . .	28
2.9.3	L1 exit with simultaneous Device initiated and Host initiated Resume results in minimum Device wake-up time of 3 ms . . . . .	28

2.10	LCD peripheral limitations	28
2.10.1	LCD segments 23 and 24 alternate function cannot be configured independently	28
2.11	SPI limitations	29
2.11.1	BSY bit may stay high at the end of a data transfer at slave mode	29
2.12	SWPMI peripheral limitations	29
2.12.1	SUSPENDED mode entry delayed	29
2.12.2	SUSPENDED mode never entered	29
2.12.3	SRF flag not set	30
2.12.4	SWP SUSPENDED mode not supported when STM32L4 is in Stop 0 or Stop 1 mode	30
2.12.5	SWPMI_IO transceiver bypass mode is not functional	30
2.13	RTC peripheral limitations	30
2.13.1	Spurious tamper detection when disabling the tamper channel	30
2.14	USART limitations	31
2.14.1	Start bit detected too soon when sampling for NACK signal from the smartcard	31
2.14.2	Break request can prevent the Transmission Complete flag (TC) from being set	31
2.14.3	nRTS is active while RE or UE = 0	31
2.15	DAC peripheral limitations	32
2.15.1	PA4 and PA5 must be configured in analog mode when DAC1_OUT1 or DAC1_OUT2 respectively is connected to OPAMP input	32
2.15.2	PA4 and PA5 can't be used in output mode when DAC1_OUT1 or DAC1_OUT2 respectively is connected to on-chip peripherals	32
2.15.3	DAC cannot be used at low VDDA if APB1 frequency is higher than 40 MHz	32
2.15.4	DAC characteristics can be degraded depending on the current sunk on DAC output	33
2.16	OPAMP limitations	33
2.16.1	OPAMP characteristics can be degraded depending on the current sunk on OPAMP output	33
2.16.2	Calibration procedure does not work in PGA mode	34
2.17	COMP peripheral limitations	34
2.17.1	Comparators propagation delay is longer than expected for input steps higher than 200 mV	34
2.17.2	Comparators output cannot be configured in open-drain	34
2.17.3	COMP1 and COMP2 configuration lost with software reset	35
2.18	LPTIM peripheral limitations	35

---

2.18.1	Low power timer 1 (LPTIM1) outputs cannot be configured in open-drain .....	35
2.19	LPUART peripheral limitations .....	35
2.19.1	Low power UART1 (LPUART1) outputs cannot be configured in open-drain .....	35
2.20	AES peripheral limitations .....	36
2.20.1	Burst read or write accesses are not supported by the AES .....	36
2.20.2	AES does not support Load multiple .....	36
<b>3</b>	<b>Revision history .....</b>	<b>37</b>

List of tables

Table 1. Device identification . . . . . 1

Table 2. Device summary . . . . . 1

Table 3. Cortex®-M4 FPU core limitations and impact on microcontroller behavior . . . . . 8

Table 4. Summary of silicon limitation . . . . . 9

Table 5. COMP characteristics . . . . . 34

Table 6. Document revision history . . . . . 37



List of figures

Figure 1. HSI oscillator trimming characteristics ..... 16

Figure 2. Parasitic resistor on DAC output..... 33

Figure 3. Parasitic resistor on OPAMP output ..... 33

# 1 ARM® 32-bit Cortex®-M4 FPU core limitations

An errata notice of the STM32L4x6xx core is available from the following web address:  
<http://infocenter.arm.com>.

All the described limitations are minor and related to the revision r0p1-v1 of the Cortex®-M4 FPU core. [Table 3](#) summarizes these limitations and their implications on the behavior of STM32L4x6xx devices.

**Table 3. Cortex®-M4 FPU core limitations and impact on microcontroller behavior**

ARM ID	ARM category	ARM summary of errata	Impact
752419	Cat 2	Interrupted loads to SP can cause erroneous behavior	Minor

## 1.1 Cortex®-M4 FPU core interrupted loads to stack pointer can cause erroneous behavior

### Description

An interrupt occurring during the data-phase of a single word load to the stack pointer (SP/R13) can cause an erroneous behavior of the device. In addition, returning from the interrupt results in the load instruction being executed with an additional time.

For all the instructions performing an update of the base register, the base register is erroneously updated on each execution, resulting in the stack pointer being loaded from an incorrect memory location.

The instructions affected by this limitation are the following:

- LDR SP, [Rn],#imm
- LDR SP, [Rn,#imm]!
- LDR SP, [Rn,#imm]
- LDR SP, [Rn]
- LDR SP, [Rn,Rm]

### Workaround

As of today, no compiler generates these particular instructions. This limitation can only occur with hand-written assembly code.

Both issues can be solved by replacing the direct load to the stack pointer by an intermediate load to a general-purpose register followed by a move to the stack pointer.

Example:

Replace LDR SP, [R0] by

LDR R2,[R0]

MOV SP,R2



## 2 STM32L4x6xx silicon limitations

[Table 4](#) gives quick references to all documented limitations.

Legend for [Table 4](#): A = workaround available; N = no workaround available; P = partial workaround available, '-' and grayed = fixed.

**Table 4. Summary of silicon limitation**

Section	Limitation	Rev 3	Rev 4
Section 2.1: System limitations	<a href="#">Section 2.1.1: Write operation in the Flash memory while it is not ready (Flash memory in power-down) is not correctly handled</a>	A	A
	<a href="#">Section 2.1.2: The configuration of the I/Os not available in WLCSP package can be modified by software</a>	A	A
	<a href="#">Section 2.1.3: Some I/Os must not be used as output when <math>VDDA &gt; VDD + 0.6\text{ V}</math> or <math>VDDUSB &gt; VDD + 0.6\text{ V}</math> or <math>VLCD &gt; VDD + 0.6\text{ V}</math></a>	N	-
	<a href="#">Section 2.1.4: Bootloader: SPI and CAN interfaces are not supported</a>	N	-
	<a href="#">Section 2.1.5: SRAM2 read access while the SRAM2 hardware erase is ongoing is not correctly supported</a>	N	N
	<a href="#">Section 2.1.6: Extra consumption when <math>VDDA</math> is below <math>0.7\text{ V}</math></a>	A	-
	<a href="#">Section 2.1.7: System clock is stopped during few <math>\mu\text{s}</math> when switching between Low-power run and Run modes</a>	N	-
	<a href="#">Section 2.1.8: PH0/PH1 is controlled by the GPIOH registers when HSE is enabled</a>	P	P
	<a href="#">Section 2.1.9: PWR_CR4 register write access may not be completed if a Low-power mode is entered just after the write operation</a>	A	A
	<a href="#">Section 2.1.10: HSI user trim is limited on some samples</a>	N	N
	<a href="#">Section 2.1.11: Option byte loading can fail if MSI frequency is greater than <math>8\text{ MHz}</math></a>	A	A
	<a href="#">Section 2.1.12: MSI at high frequency ranges cannot be used as wakeup from Stop 0 clock source</a>	P	-
	<a href="#">Section 2.1.13: PLL may not lock if VCO is below <math>96\text{ MHz}</math> and temperature is below <math>0^\circ\text{C}</math></a>	A	A
	<a href="#">Section 2.1.14: Full JTAG configuration without NJTRST pin cannot be used</a>	A	A
	<a href="#">Section 2.1.14: Full JTAG configuration without NJTRST pin cannot be used</a>	A	A
	<a href="#">Section 2.1.15: MSIRDY flag issue preventing entry in low power mode</a>	A	A
Section 2.2: FMC peripheral limitations	<a href="#">Section 2.2.1: Dummy read cycles inserted when reading synchronous memories</a>	N	N
	<a href="#">Section 2.2.2: Data corruption during burst read from FMC synchronous memory</a>	A	A
	<a href="#">Section 2.2.3: FMC bank switching to asynchronous bank for write</a>	A	A
	<a href="#">Section 2.2.4: Read burst access of 9 words or more is not supported by FMC</a>	P	P

Table 4. Summary of silicon limitation (continued)

Section	Limitation	Rev 3	Rev 4
Section 2.3: QUADSPI peripheral limitations	Section 2.3.1: QUADSPI_BK1_IO1 is always an input when the command is sent in dual or quad SPI mode	P	P
	Section 2.3.2: Hard fault is not generated in case of out-of-range memory-mapped access	N	N
	Section 2.3.3: Extra data written in the FIFO at the end of a read transfer	A	A
Section 2.4: ADC peripheral limitations	Section 2.4.1: Injected queue of context is not available in case of JQM=0	N	N
	Section 2.4.2: ADC triggers from EXTI require external interrupt management	P	-
	Section 2.4.3: DMA2 channels 2 and 3 (respectively) cannot be used when the ADC2 and ADC3 requests are selected on DMA2 channels 4 and 5	P	P
	Section 2.4.4: Wrong ADC conversion results when delay between calibration and first conversion or between 2 consecutive conversions is too long	N	N
	Section 2.4.5: Burst read or write accesses are not supported by the ADC	A	A
Section 2.5: DFSDM peripheral limitations	Section 2.5.1: RDATACH[2:0] status bits are not implemented	N	N
	Section 2.5.2: New regular channel selection taken into account at the end of an injected conversion when a regular conversion is pending	N	N
	Section 2.5.3: DFSDM triggers from timers can be missed in specific conditions	A	A
	Section 2.5.4: DFSDM triggers from EXTI require external interrupt management	P	-
Section 2.6: I2C peripheral limitations	Section 2.6.1: I2C Fast-mode Plus drive is not available on all SDA/SCL I/Os	N	N
	Section 2.6.2: Wrong behavior related with MCU Stop mode when wakeup from Stop mode by I2C peripheral is disabled	A	A
	Section 2.6.3: Wrong data sampling when data set-up time (tSU;DAT) is smaller than one I2CCLK period	P	P
	Section 2.6.4: Spurious Bus Error detection in master mode	A	A
	Section 2.6.5: I2C3 analog filters requires that both PC0/PC1 or both PG7/PG8 are configured as I2C3 alternate function	A	A
Section 2.7: SDMMC peripheral limitations	Section 2.7.1: Wrong CCRCFAIL status after a response without CRC is received	A	A
	Section 2.7.2: Clock division per 255 is not possible	N	N
	Section 2.7.3: Wait for response bits "10" configuration does not work correctly	A	A
	Section 2.7.4: MMC stream write of less than 8 bytes does not work correctly	A	A
Section 2.8: bxCAN peripheral limitations	Section 2.8.1: bxCAN time triggered mode not supported	N	N

Table 4. Summary of silicon limitation (continued)

Section	Limitation	Rev 3	Rev 4
Section 2.9: OTG_FS peripheral limitations	Section 2.9.1: Suspend mode robustness marginality after receiving corrupted packet from Host	A	A
	Section 2.9.2: Bits OTG_FS_GLPMCFG[6:2] are not write-protected in Device mode	P	P
	Section 2.9.3: L1 exit with simultaneous Device initiated and Host initiated Resume results in minimum Device wake-up time of 3 ms	P	P
Section 2.10: LCD peripheral limitations	Section 2.10.1: LCD segments 23 and 24 alternate function cannot be configured independently	P	P
Section 2.11: SPI limitations	Section 2.11.1: BSY bit may stay high at the end of a data transfer at slave mode	A	A
Section 2.12: SWPMI peripheral limitations	Section 2.12.1: SUSPENDED mode entry delayed	N	N
	Section 2.12.2: SUSPENDED mode never entered	P	P
	Section 2.12.3: SRF flag not set	N	N
	Section 2.12.4: SWP SUSPENDED mode not supported when STM32L4 is in Stop 0 or Stop 1 mode	N	N
	Section 2.12.5: SWPMI_IO transceiver bypass mode is not functional	N	N
Section 2.13: RTC peripheral limitations	Section 2.13.1: Spurious tamper detection when disabling the tamper channel	P	P
Section 2.14: USART limitations	Section 2.14.1: Start bit detected too soon when sampling for NACK signal from the smartcard	N	N
	Section 2.14.2: Break request can prevent the Transmission Complete flag (TC) from being set	A	A
	Section 2.14.3: nRTS is active while RE or UE = 0	A	A
Section 2.15: DAC peripheral limitations	Section 2.15.1: PA4 and PA5 must be configured in analog mode when DAC1_OUT1 or DAC1_OUT2 respectively is connected to OPAMP input	N	-
	Section 2.15.2: PA4 and PA5 can't be used in output mode when DAC1_OUT1 or DAC1_OUT2 respectively is connected to on-chip peripherals	N	-
	Section 2.15.3: DAC cannot be used at low VDDA if APB1 frequency is higher than 40 MHz	P	-
	Section 2.15.4: DAC characteristics can be degraded depending on the current sunk on DAC output	N	-
Section 2.16: OPAMP limitations	Section 2.16.1: OPAMP characteristics can be degraded depending on the current sunk on OPAMP output	N	-
	Section 2.16.2: Calibration procedure does not work in PGA mode	A	A
Section 2.17: COMP peripheral limitations	Section 2.17.1: Comparators propagation delay is longer than expected for input steps higher than 200 mV	N	N
	Section 2.17.2: Comparators output cannot be configured in open-drain	N	N
	Section 2.17.3: COMP1 and COMP2 configuration lost with software reset	N	N

Table 4. Summary of silicon limitation (continued)

Section	Limitation	Rev 3	Rev 4
<i>Section 2.18: LPTIM peripheral limitations</i>	<i>Section 2.18.1: Low power timer 1 (LPTIM1) outputs cannot be configured in open-drain</i>	N	N
<i>Section 2.19: LPUART peripheral limitations</i>	<i>Section 2.19.1: Low power UART1 (LPUART1) outputs cannot be configured in open-drain</i>	N	N
<i>Section 2.20: AES peripheral limitations</i>	<i>Section 2.20.1: Burst read or write accesses are not supported by the AES</i>	A	A
	<i>Section 2.20.2: AES does not support Load multiple</i>	A	A

## 2.1 System limitations

### 2.1.1 Write operation in the Flash memory while it is not ready (Flash memory in power-down) is not correctly handled

#### Description

If the Flash memory is read while it is not ready after a power-down state (i.e after Stop mode, or when it was put in power-down by means of SLEEP\_PD or RUN\_PD bits in the FLASH\_ACR register), the AHB bus is normally stalled until the Flash memory is ready and can provide the correct data.

However, this mechanism is not supported in case of a write operation in normal programming mode: the Flash memory read which is automatically performed before the programming in order to check that the address is virgin is incorrect. Consequently the write operation is skipped.

#### Workaround

The user must wait until the Flash memory is ready before performing a write operation: this is done by reading first an address of the Flash memory not present in the cache.

### 2.1.2 The configuration of the I/Os not available in WLCSP package can be modified by software

#### Description

Unbonded I/Os in WLCSP package can be configured by software. This can lead to an extra consumption if they are floating with Schmitt trigger ON.

#### Workaround

Configure all unbonded I/Os as analog input or as output push-pull 0 state.

### 2.1.3 Some I/Os must not be used as output when $V_{DDA} > V_{DD} + 0.6\text{ V}$ or $V_{DDUSB} > V_{DD} + 0.6\text{ V}$ or $V_{LCD} > V_{DD} + 0.6\text{ V}$

#### Description

When  $V_{DDA} > V_{DD} + 0.6\text{ V}$ , or when  $V_{DDUSB} > V_{DD} + 0.6\text{ V}$ , or when  $V_{LCD} > V_{DD} + 0.6\text{ V}$ , the I/Os listed below must not be used as:

- general purpose output, push-pull or open-drain
- alternate function output, push-pull or open-drain.

An unexpected leakage occurs between the respective supply voltage ( $V_{DDA}$  or  $V_{DDUSB}$  or  $V_{LCD}$ ) and  $V_{DD}$ , and the output state can be lost.

These I/Os can be used in:

- input mode;
- analog mode;
- alternate function input or analog mode.

When  $V_{DDA} > V_{DD} + 0.6$  V, the concerned I/Os are:

- PA0, PA1, PA2, PA3, PA4, PA5, PA6, PA7
- PB0, PB1, PB2, PB3, PB4, PB5, PB6, PB7
- PC0, PC1, PC2, PC3, PC4, PC5
- PF3, PF4, PF5, PF6, PF7, PF8, PF9, PF10.

When  $V_{DDUSB} > V_{DD} + 0.6$  V, the concerned I/Os are:

- PA9, PA10, PA11, PA12.

When  $V_{LCD} > V_{DD} + 0.6$  V, the concerned I/Os are:

- PA1, PA2, PA3, PA6, PA7, PA8, PA9, PA10, PA15
- PB0, PB1, PB3, PB4, PB5, PB7, PB8, PB9, PB10, PB11, PB12, PB13, PB14, PB15
- PC0, PC1, PC2, PC4, PC5, PC6, PC7, PC8, PC9, PC10, PC11, PC12
- PD2, PD8, PD9, PD10, PD11, PD12, PD13, PD14, PD15
- PE0, PE1, PE2, PE3.

#### Workaround

None.

### 2.1.4 Bootloader: SPI and CAN interfaces are not supported

#### Description

The bootloader does not support yet SPI and CAN interfaces.

#### Workaround

None.

### 2.1.5 SRAM2 read access while the SRAM2 hardware erase is ongoing is not correctly supported

#### Description

If a read access is done in the SRAM2, while a SRAM2 hardware erase operation is ongoing: the read access is stalled as long as the erase operation is not completed, and the read value is not 0x0, but the latest value previously read from SRAM2.

#### Workaround

None.

### 2.1.6 Extra consumption when $V_{DDA}$ is below 0.7 V

#### Description

If  $V_{DDA}$  is tied to ground or is below 0.7 V, an unexpected leakage from I/O to ground is present on PA3 and PB0.

**Workaround**

None.

**2.1.7 System clock is stopped during few  $\mu$ s when switching between Low-power run and Run modes****Description**

When switching from Low-power run to Run mode, or when switching from Run mode to Low-power run mode, the System clock (SYSCLK) is stopped as long as the regulator is not ready (few  $\mu$ s).

**Workaround**

None.

**2.1.8 PH0/PH1 is controlled by the GPIOH registers when HSE is enabled****Description**

The PH0 and PH1 GPIO configuration is not bypassed when the HSE is enabled. The oscillator can stop if the I/Os are not configured in analog mode.

**Workaround**

PH0 and PH1 must be configured in analog mode (reset value) when HSE is enabled.

**2.1.9 PWR\_CR4 register write access may not be completed if a Low-power mode is entered just after the write operation****Description**

PWR\_CR4 write access should insert 3 APB1 wait states, but it does not. Consequently, if a low-power mode is entered just after writing into this register, the PWR\_CR4 register may not be updated before the low-power mode is entered. This can occur in particular when the APB1 clock is prescaled.

**Workaround**

PWR\_CR4 register must be read after the write operation to ensure that the write is done before entering the low-power mode.

**2.1.10 HSI user trim is limited on some samples****Description**

The HSI user trimming step is typically:

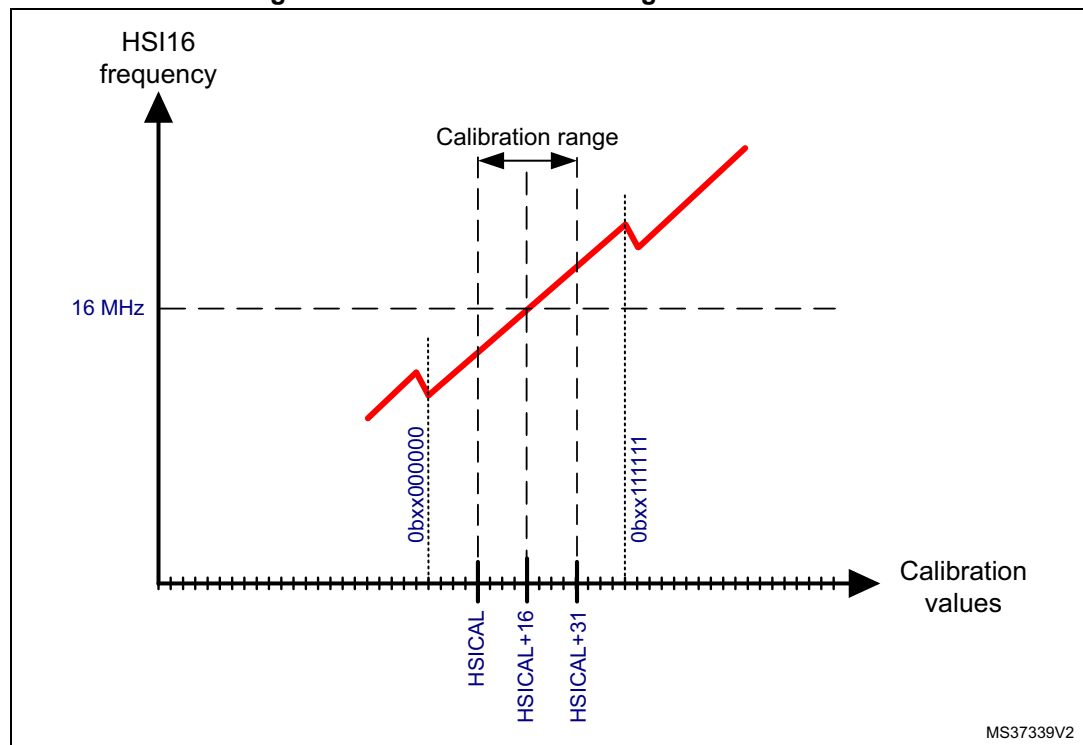
- +0.3% of frequency when the trimming code is not a multiple of 64;
- -7% of frequency when the trimming code is a multiple of 64.

As shown in [Figure 1](#), the HSI user trimming allows to add or to subtract up to 16 steps compared to the factory trim value.

If the HSI factory trim value is close to a multiple of 64, with only 16 steps for positive or for negative correction, it is not possible to compensate the 7% drop when the code is a multiple of 64.

Consequently, the user trim correction must not jump over the codes multiple of 64, which can limit the user correction either in positive or in negative direction.

**Figure 1. HSI oscillator trimming characteristics**



#### Workaround

None.

### 2.1.11 Option byte loading can fail if MSI frequency is greater than 8 MHz

#### Description

The option byte loading operation can fail if the MSI clock is ON with a frequency equal or greater than 8 MHz before performing an option byte loading by setting OBL\_LAUNCH bit in the FLASH\_CR register. Some options and engineering bytes values can be corrupted.

#### Workaround

If MSI is ON at a frequency equal or higher than 8MHz, It is mandatory to reduce MSI frequency to 4 MHz or less before launching the Option Byte loading operation by setting the OBL\_LAUNCH bit in the FLASH\_CR register.



### 2.1.12 MSI at high frequency ranges cannot be used as wakeup from Stop 0 clock source

#### Description

When  $V_{DD} > 3\text{ V}$ , the MSI startup frequency overshoot is higher than expected and can be higher than the maximum allowed product frequency. Therefore it is not possible to select the MSI at 48 or at 32 MHz as wakeup clock source when exiting Stop 0 in Range 1, because the frequency reached during the overshoot is too high compared to the flash wait states configuration. For the same reason it is not possible to select the MSI at 24, 16 or 8 MHz as wakeup clock source when exiting Stop 0 in Range 2.

The MSI in all ranges can be used to exit Stop 1 or Stop 2 modes, because the main regulator wakeup delay is used to avoid the clock propagation and the regulator wakeup time is greater than the MSI overshoot duration.

#### Workaround

Wait for 6  $\mu\text{s}$  after exiting Stop 0 before switching the MSI frequency to the higher frequency, or use Stop 1 or Stop 2 instead of Stop 0.

### 2.1.13 PLL may not lock if VCO is below 96 MHz and temperature is below 0°C

#### Description

The VCO minimum value should be 64 MHz. When the VCO is below 96 MHz and the temperature is below 0°C the PLL may never lock.

#### Workaround

Program the PLL with VCO at 96 MHz or above.

### 2.1.14 Full JTAG configuration without NJTRST pin cannot be used

#### Description

When using the JTAG debug port in debug mode, the connection with the debugger is lost if the NJTRST pin (PB4) is used as a GPIO. Only the 4-wire JTAG port configuration is impacted.

#### Workaround

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

### 2.1.15 MSIRDY flag issue preventing entry in low power mode

#### Description

If the MSI clock is stopped when it is running at high frequency (24 MHz or above), the MSIRDY (MSI ready) flag can stay at value 1 instead of 0.

Once this flag remains set while the MSI clock is OFF, entry in Stop 0, Stop 1, Stop 2, Standby and Shutdown modes is no longer possible (the product is blocked in the low power mode entry phase).

The following factors increase the probability to have this issue:

- a high MSI frequency
- a low  $V_{DD}$  external supply
- a high temperature.

### Workaround

- The MSI clock can run at any frequency if both conditions below are met:
  - a) the system clock is MSI or PLL fed by MSI when requesting entry in low power mode;
  - b) the wakeup clock is MSI.
- If the system clock is any other clock than MSI, lower the MSI frequency to 16 MHz (or less) and add a short delay loop (200 ns minimum) before stopping MSI by software. This will ensure that the MSIRDY flag be really 0 before requesting entry in low power mode.
- If the system clock is MSI and the wakeup clock is not MSI, lower the MSI frequency to 16 MHz (or less) and add a short delay loop (200 ns minimum) before requesting entry in low power mode.
- If the system clock is PLL fed by MSI and the wakeup clock is not MSI, the MSI frequency used as PLL input should not exceed 16 MHz. In other words, do not use the PLL fed by MSI as system clock with MSI at high frequency (24 MHz or above) divided by PLLM (ensuring the PLL input is between 4 and 16 MHz).

## 2.2 FMC peripheral limitations

### 2.2.1 Dummy read cycles inserted when reading synchronous memories

#### Description

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

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

#### Workaround

None.

### 2.2.2 Data corruption during burst read from FMC synchronous memory

#### Description

A burst read from static memory can be corrupted if all the following conditions are met:

- One FMC bank is configured in synchronous mode with WAITEN bit enabled while another FMC bank is used with WAITEN bit disabled.
- A read burst transaction is ongoing from static synchronous memory with wait feature enabled.
- The synchronous memory asserts the wait signal during the ongoing burst read.
- The read burst transaction is followed by an access to an FMC banks for which the WAITEN bit is disabled in the FMC\_BCRx register.

#### Workaround

1. Set the WAITEN bit on all FMC static banks even if it is not used by the memory.
2. Set the same WAIT polarity on all static banks.
3. Enable the internal pull-up on PD6 in order to set to ready the FMC\_NWAIT input when the synchronous memory is de-selected and the other FMC bank without wait feature is selected.

### 2.2.3 FMC bank switching to asynchronous bank for write

#### Description

When switching from one of the FMC banks in read transaction to another asynchronous bank for write, the FMC could hang in one of the following conditions:

- one FMC bank is enabled with BUSTURN timing > 0;
- a second FMC bank is enabled in asynchronous (multiplexed or not multiplexed) mode with BUSTURN = 0;
- a read from the first bank followed by a write transaction on the second bank.

#### Workaround

Use BUSTURN = 0, or different from 0 for used banks.

## 2.2.4 Read burst access of 9 words or more is not supported by FMC

### Description

CPU read burst access equal to or more than 9 registers to FMC returns corrupted data starting from the 9<sup>th</sup> read word. These bursts can only be generated by Cortex<sup>®</sup>-M4 CPU and not by the other masters (i.e not by DMA).

This issue occurs when the stack is remapped on the external memory on the FMC and POP operations are performed with 9 or more registers.

This also occurs when LDM/VLDM operations are used with 9 or more registers.

### Workaround

Stack must be located in the internal SRAM.

IAR-EWARM: EWARM compiler does not generate LDM/VLDM operations with more than 8 registers except when the *setjmp* or *longjmp* functions of C Library are explicitly used in the source code.

Keil<sup>®</sup> MDK-ARM<sup>™</sup>: Starting from version 5.06, ARM<sup>®</sup> Compiler implements a patch which limits the number of registers with LDM/VLDM operations. Starting from version 5.16, MDK-ARM<sup>™</sup> includes ARM<sup>®</sup> compiler with this patch, addressing the hardware limitations of the STM32L4 FMC burst reads.

Tasking: TASKING compiler starting from version v5.2r1 does not generate LDM instructions with more than 8 registers except when the *setjmp* or *longjmp* functions of C Library are explicitly used in the source code. It generates POP instructions with 9 registers.

In assembly written code, LDM/VLDM operations must be limited to 8 registers.

## 2.3 QUADSPI peripheral limitations

### 2.3.1 QUADSPI\_BK1\_IO1 is always an input when the command is sent in dual or quad SPI mode

#### Description

The IO1 is staying as an input instead of changing to output to drive the command when this last is on 2 or 4 lines and there is no address, no data and no alternate byte in the frame, or they exist on 1 line.

Therefore it is not possible to use dual-/quad-mode for the command phase.

This bug affects Numonyx's dual-/quad-instruction mode.

#### Workaround

The dual-/quad-instruction mode of Micron or Numonyx flashes is working fine when user does not send only a command in the frame. For example, the «write enable» command can be avoided by writing directly the status register of the flash instead. Otherwise the application can reset the flash through the reset pin (if it exists) to disable the dual-/quad instruction mode before each time it needs to send a frame which contains only the command phase.

### 2.3.2 Hard fault is not generated in case of out-of-range memory-mapped access

#### Description

A hard fault is not generated when the Cortex CPU executes an instruction which reads from an out-of-range memory-mapped address. Consequently the CPU effectively skips the instruction (the destination register is not modified) and continues executing without giving a hard fault.

#### Workaround

None.

### 2.3.3 Extra data written in the FIFO at the end of a read transfer

#### Description

When all the conditions listed below are gathered:

- QUADSPI is used in indirect mode
- QUADSPI clock is AHB/2 (PRESCALER = 0x01 in the QUADSPI\_CR)
- QUADSPI is in quad mode (DMODE = 0b11 in the QUADSPI\_CCR)
- QUADSPI is in DDR mode (DDRM = 0b1 in the QUADSPI\_CCR)

An extra data is incorrectly written in the FIFO when a data is read at the same time that the FIFO gets full at the end of a read transfer.

#### Workaround

One of the two workarounds listed below can be done:

- Read out the extra data until the BUSY flag goes low and discard it.
- Request an abort after reading out all the correct received data from FIFO in order to flush FIFO and have the busy low. Abort will keep the last register configuration (set the ABORT bit in the QUADSPI\_CR).

## 2.4 ADC peripheral limitations

### 2.4.1 Injected queue of context is not available in case of JQM=0

#### Description

The queue mechanism is not functional when JQM = 0. The effective queue length is equal to 1 stage: a new context written before the previous context's consumption will lead to a queue overflow and will be ignored. Consequently, the ADC must be stopped before programming the JSQR register.

#### Workaround

None.

## 2.4.2 ADC triggers from EXTI require external interrupt management

### Description

When the ADC external trigger for regular or injected channel selects the EXTI line 11 or EXTI line 15, it is required to configure and enable the line as an interrupt source in the EXTI and NVIC peripherals.

### Workaround

Configure the EXTI line as an interrupt source in the EXTI and enable the interrupt line in the NVIC. The EXTI interrupt flag must be cleared in the interrupt subroutine in order to allow new trigger event detection.

## 2.4.3 DMA2 channels 2 and 3 (respectively) cannot be used when the ADC2 and ADC3 requests are selected on DMA2 channels 4 and 5

### Description

When the DMA2 channel 4 is used for the ADC2 requests (C4S=0000 in the DMA2\_CSELR register) it is also needed to select request 0 for DMA2 channel 2 (C2S=0000 in the DMA2\_CSELR register). The consequence is that the DMA2 channel 2 cannot be used by another peripheral.

When the DMA2 channel 5 is used for the ADC3 requests (C5S=0000 in the DMA2\_CSELR register) it is also needed to select request 0 for DMA2 channel 3 (C3S=0000 in the DMA2\_CSELR register). The consequence is that the DMA2 channel 3 cannot be used by another peripheral.

### Workaround

Select the DMA1 channel 2 for the ADC2 DMA requests or use other available mapping for the peripherals mapped on DMA2 channel 2.

Select the DMA1 channel 3 for the ADC3 DMA requests or use other available mapping for the peripherals mapped on DMA2 channel 3.

## 2.4.4 Wrong ADC conversion results when delay between calibration and first conversion or between 2 consecutive conversions is too long

### Description

When the delay between two consecutive ADC conversions is higher than 1 ms the result of the second conversion might be incorrect. The same issue occurs when the delay between the calibration and the first conversion is higher than 1 ms.

### Workaround

When the delay between two ADC conversions is higher than the above limit, perform two ADC consecutive conversions in single, scan or continuous mode: the first is a dummy conversion of any ADC channel. This conversion should not be taken into account by the application.

## 2.4.5 Burst read or write accesses are not supported by the ADC

### Description

The ADC does not support LDM, STM, LDRD and STRD instructions for successive multiple-data read and write accesses to a contiguous address block.

### Workaround

Prevent compilers from generating LDM, STM, LDRD and STRD instructions. In general, this can be achieved organizing the source code to avoid consecutive read or write accesses to neighboring addresses in lower-to-higher order. In cases where consecutive read or write accesses to neighboring addresses cannot be avoided, order the source code so that it accesses higher address first.

## 2.5 DFSDM peripheral limitations

### 2.5.1 RDATACH[2:0] status bits are not implemented

#### Description

RDATACH[2:0] "Regular channel most recently converted" status bits are not implemented in the DFSDM data register for the regular channel (DFSDMx\_RDATAR).

#### Workaround

None. These bits will be implemented in next silicon release.

### 2.5.2 New regular channel selection taken into account at the end of an injected conversion when a regular conversion is pending

#### Description

When a regular channel conversion is on going and is interrupted by an injected channel conversion: if the regular channel is changed on the fly during the injected channel conversion, the new regular channel selection is taken into account at the end of the injected conversion.

#### Workaround

None: do not change the regular channel selection on the fly when regular continuous conversions are requested.

### 2.5.3 DFSDM triggers from timers can be missed in specific conditions

#### Description

Triggers from timers to DFSDM can be missed when all the conditions listed below occur:

- the DFSDM clock is the APB2 clock PCLK2 (DFSDMSEL=0 in the RCC\_CCIPR register)
- the DFSDM is triggered by TIM3\_TRGO, TIM4\_TRGO, TIM6\_TRGO or TIM7\_TRGO
- the APB2 frequency is smaller than the APB1 frequency

**Workaround**

Select the System clock as DFSDM clock (DFSDMSEL=1 in the RCC\_CCIPR register).

**2.5.4 DFSDM triggers from EXTI require external interrupt management****Description**

When the DFSDM external trigger for regular or injected channel selects the EXTI line 11 or EXTI line 15, it is required to configure and enable the line as an interrupt source in the EXTI and NVIC peripherals.

**Workaround**

Configure the EXTI line as an interrupt source in the EXTI and enable the interrupt line in the NVIC. The EXTI interrupt flag must be cleared in the interrupt subroutine in order to allow new trigger event detection.

**2.6 I2C peripheral limitations****2.6.1 I2C Fast-mode Plus drive is not available on all SDA/SCL I/Os****Description**

Only PB6,7,8,9,13,14, PC0,1 and PG7,8 can effectively be configured in I2C Fm+ driving mode. Setting I2C1\_FMP, I2C2\_FMP and I2C3\_FMP in the SYSCFG\_CFGR1 register has no effect on PB10, PB11, PF0, PF1, PG13, PG14.

*Note:* When using the I/O without 20 mA Fm+ drive, it is still possible to reach 1 MHz Fm+ speed, with limited bus load.

**Workaround**

None.

**2.6.2 Wrong behavior related with MCU Stop mode when wakeup from Stop mode by I2C peripheral is disabled****Description**

When wakeup from Stop mode by I2C peripheral is disabled (WUPEN = 0) and the MCU enters Stop mode while a transaction is on-going on the I<sup>2</sup>C bus, the following wrong operations may occur:

1. BUSY flag may be wrongly set when the MCU exits Stop mode. This prevents from initiating a transfer in master mode, as the START condition cannot be sent when BUSY is set. This failure may occur in master mode of the I2C peripheral used in multi-master I<sup>2</sup>C-bus environment.
2. If I<sup>2</sup>C-bus clock stretching is enabled in I2C peripheral (NOSTRETCH = 0), the I2C peripheral may pull SCL low as long as the MCU remains in Stop mode, suspending all I<sup>2</sup>C-bus activity during that time. This may occur when the MCU enters Stop mode during the address phase of an I<sup>2</sup>C-bus transaction, in low period of SCL. This failure may occur in slave mode of the I2C peripheral or, in master mode of the I2C peripheral used in multi-master I<sup>2</sup>C-bus environment. Its probability depends on the timing.



**Workaround**

Disable the I2C peripheral (PE=0) before entering Stop mode and re-enable it in Run mode.

**2.6.3 Wrong data sampling when data set-up time ( $t_{\text{SU;DAT}}$ ) is smaller than one I2CCLK period****Description**

The I2C bus specification and user manual specify a minimum data set-up time ( $t_{\text{SU;DAT}}$ ) at:

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

The I2C SDA line is not correctly sampled when  $t_{\text{SU;DAT}}$  is smaller than one I2CCLK (I2C clock) period: the previous SDA value is sampled instead of the current one. This can result in a wrong slave address reception, a wrong received data byte, or a wrong received acknowledge bit.

**Workaround**

Increase the I2CCLK frequency to get I2CCLK period smaller than the transmitter minimum data set-up time. Or, if it is possible, increase the transmitter minimum data set-up time.

**2.6.4 Spurious Bus Error detection in master mode****Description**

In master mode, a bus error can be detected by mistake, so the BERR flag can be wrongly raised in the status register. This will generate a spurious Bus Error interrupt if the interrupt is enabled. A bus error detection has no effect on the transfer in master mode, therefore the I2C transfer can continue 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 on-going transfer can be handled normally.

**2.6.5 I2C3 analog filters requires that both PC0/PC1 or both PG7/PG8 are configured as I2C3 alternate function****Description**

I2C3 analog filters can be enabled for PC0 and/or PC1 only if both IOs are configured in I2C3 Alternate Function mode. For example if you use PC0 for clock and PB4 for data, PC0 filter can be enabled only if PC1 is configured in I2C3 Alternate Function mode too.

I2C3 filter can be enabled for PG7 and/or PG8 only if both IOs are configured in I2C3 Alternate Function mode. For example if you use PG7 for clock and PC1 for data, PG7 filter can be enabled only if PG8 is configured in I2C3 Alternate Function mode too.

**Workaround**

Use both PC0/PC1 as I2C3 alternate functions or use both PG7/PG8 as I2C3 alternate functions if the analog filter is needed.

## **2.7 SDMMC peripheral limitations**

### **2.7.1 Wrong CCRCFAIL status after a response without CRC is received**

**Description**

The CRC is calculated even if the response to a command does not contain any CRC field. As a consequence, after the SDIO command IO\_SEND\_OP\_COND (CMD5) is sent, the CCRCFAIL bit of the SDMMC\_STA register is set.

**Workaround**

The CCRCFAIL bit in the SDMMC\_STA register shall be ignored by the software. CCRCFAIL must be cleared by setting CCRCFAILC bit of the SDMMC\_ICR register after reception of the response to the CMD5 command.

### **2.7.2 Clock division per 255 is not possible**

**Description**

When CLKDIV divide factor in SDMMC\_CLKCR register is equal to 255, the SDMMC\_CK clock output is not provided.

**Workaround**

Do not use CLKDIV value equals to 255: use clock divider factors from 0 to 254.

### **2.7.3 Wait for response bits “10” configuration does not work correctly**

**Description**

The Wait for response bits configuration “10” (WAITRESP in the SDMMC\_CMD register) does not work correctly. When WAITRESP bits are programmed to '10' Command Path State Machine (CPSM) waits for a non-existing response.

**Workaround**

Do not use WAITRESP value equal to “10” when sending a command without a response.

Use WAITRESP value equal to “00” to indicate to SDMMC CPSM that no response is expected.

### **2.7.4 MMC stream write of less than 8 bytes does not work correctly**

When SDMMC host starts a stream write (WRITE\_DAT\_UNTIL\_STOP CMD20), the number of bytes to transfer is not known by the card.

The card will write data from the host until a STOP\_TRANSMISSION (CMD12) command is received.

The WAITPEND bit 9 of SDMMC\_CMD register is set to synchronize the sending of the STOP\_TRANSMISSION (CMD12) command with the data flow.

When WAITPEND is set, the transmission of this command stays pending until 50 data bits including STOP bit remain to transmit.

For a stream write of less than 8 bytes, the STOP\_TRANSMISSION (CMD12) command should be started before the data transfer starts. Instead of this, the data write and the command sending are started simultaneously.

It implies that when less than 8 bytes have to be transmitted, (8 - DATALENGTH) bytes are programmed to 0xFF in the card after the last byte programmed (where DATALENGTH is the number of data bytes to be transferred).

### **Workaround**

Do not use stream write WRITE\_DAT\_UNTIL\_STOP (CMD20) with a DATALENGTH less than 8 bytes.

Use set block length (SET\_BLOCKLEN: CMD16) followed by single block write command (WRITE\_BLOCK: CMD24) instead of stream write (CMD20) with desired block length.

## **2.8 bxCAN peripheral limitations**

### **2.8.1 bxCAN time triggered mode not supported**

#### **Description**

The time triggered communication mode described in the reference manual is not supported, and so time stamp values are not available. TTCM bit must be kept cleared in the CAN\_MCR register (time triggered communication mode disabled).

#### **Workaround**

None.

## **2.9 OTG\_FS peripheral limitations**

### **2.9.1 Suspend mode robustness marginality after receiving corrupted packet from Host**

#### **Description**

Once the early suspend interrupt is set after 3 ms of USB idleness, the corrupted packet with missing EOP unexpectedly sent by the Host stops the suspend timer and when the bus becomes idle again, suspend/early suspend interrupts are no more generated. No error flag is set and it is not possible to issue remote wake-up to the Host. Device restarts operate normally only upon reception of next EOP from Host.

#### **Workaround**

If such scenario is believed possible, implement a timeout check started once early suspend interrupt is detected. If within next 4 ms there is no suspend, resume, early suspend or SOF

detected, it means that the suspend timer is blocked and need to be recovered by reinitializing the USB peripheral by a soft reset (CSRST bit in OTG\_FS\_GRSTCTL). After that, when the USB system is eventually woken up from the Suspend mode (either by Host or by Device generated remote wake-up), it is necessary to generate a soft disconnect & re-connect, as the enumeration status is lost by the soft reset.

## 2.9.2 Bits OTG\_FS\_GLPMCFG[6:2] are not write-protected in Device mode

### Description

Bits OTG\_FS\_GLPMCFG[6:2] are not write-protected in Device mode as expected, but could be modified by application software. In Device mode, the core updates the fields with the values received in the LPM Token. When some read-modify-write access is performed to OTG\_FS\_GLPMCFG register during the time of LPM token exchange, the application software could over-write the values written by OTG\_FS core with the old value.

### Workaround

Program the LPMACK bit in OTG\_FS\_GLPMCFG register to NYET response before updating OTG\_FS\_GLPMCFG register while USB transfers are in progress. Set the LPMACK bit back to ACK response when updating is finished.

## 2.9.3 L1 exit with simultaneous Device initiated and Host initiated Resume results in minimum Device wake-up time of 3 ms

### Description

When the device application initiates the remote wakeup by setting the RWUSIG bit in OTG\_FS\_DCTL register while the host is initiating resume at the same time, further LPM tokens sent by the Host within next 3 ms are ignored (the device does not enter L1 on any LPM token acknowledgment). If the host has sent an LPM token during these 3 ms and is acknowledged by the device core, then host enters L1 whereas device does not enter L1. Instead, the device enters normal Suspend mode due to 3 ms of bus inactivity.

### Workaround

Program the LPMACK bit in OTG\_FS\_GLPMCFG register to NYET response before initiating remote wakeup and set it back to ACK only after 3 ms.

## 2.10 LCD peripheral limitations

### 2.10.1 LCD segments 23 and 24 alternate function cannot be configured independently

#### Description

When PC5 is configured as alternate function LCD to output LCD\_SEG23, PC6 is also configured as alternate function LCD and output LCD\_SEG24. As a consequence, when PC5 is used for LCD, PC6 cannot be used for another purpose than LCD. In addition, in order to use PC6 as LCD alternate function, it is required to configure PC5 as LCD alternate function.

**Workaround**

When the LCD is used, use both PC5 and PC6 as LCD segments, or use none of them.

## 2.11 SPI limitations

### 2.11.1 BSY bit may stay high at the end of a data transfer at slave mode

**Description**

In slave mode, BSY bit is not reliable to handle the end of data frame transaction due to some bad synchronization between the CPU clock and external SCK clock provided by master. Sporadically, the BSY bit is not cleared at the end of a data frame transfer. As a consequence, it is not recommended to rely on BSY bit before entering low power mode or modifying the SPI configuration.

**Workaround**

To ensure BSY bit flag is always reset at the end of a frame communication, disable the SPI by software before the communication is finished by following sequence:

- write last data to data register to transmit the data,
- poll TXE till it becomes high to ensure the data transfer has started,
- disable SPI by clearing SPE while the last data transfer is still on going,
- poll the BSY bit till it becomes low.

## 2.12 SWPMI peripheral limitations

### 2.12.1 SUSPENDED mode entry delayed

**Description**

When activating the SWPMI by setting the SWPEN bit in SWPMI\_CR register, the SWPMI will rise the SWPMI\_IO to 1.8V, send a short transition sequence and 14 idle bits before switching to SUSPENDED mode. As a consequence, the SRF flag is set and the SUSP flag is cleared in SWPMI\_ISR register.

**Workaround**

None.

### 2.12.2 SUSPENDED mode never entered

**Description**

When activating the SWPMI by setting the SWPEN bit in SWPMI\_CR register, the SWPMI will generate the following sequence in loop: rise the SWPIO, send a short transition sequence and 14 idle bits. As a consequence, SWP stays in ACTIVATED state, and never switch to SUSPENDED state.

**Workaround**

Keep SWPMI1SEL bit cleared in RCC\_CCIPR register to select PCLK1 as SWPMI clock source, and configure the PCLK1 prescaler to feed the SWPMI with a clock frequency below or equal to 8 MHz.

**2.12.3 SRF flag not set****Description**

If the SWPMI1SEL bit is set in RCC\_CCIPR register to select HSI as SWPMI clock source, when receiving a resume by slave, the SRF flag may not be set. Nevertheless, the SWPMI is switching correctly from SUSPENDED to ACTIVATED when receiving a RESUME by slave. Therefore frame reception is not impacted.

**Workaround**

None.

**2.12.4 SWP SUSPENDED mode not supported when STM32L4 is in Stop 0 or Stop 1 mode****Description**

STM32L4 cannot enter Stop 0 or Stop 1 mode if SWPMI is activated and is in SUSPENDED state.

**Workaround**

Deactivate the SWP bus before requesting entry in Stop 0 or Stop 1 mode. Refer to the SWPMI section in the product reference manual for the deactivation procedure.

**2.12.5 SWPMI\_IO transceiver bypass mode is not functional****Description**

When the internal SWPMI transceiver is bypassed by setting the SWP\_TBYP bit in the SWPMI\_OR register, SWPMI1\_RX mapped on PB14 is forced in output mode instead of input mode.

**Workaround**

None. The internal transceiver must be used.

**2.13 RTC peripheral limitations****2.13.1 Spurious tamper detection when disabling the tamper channel****Description**

If the tamper detection is configured for detection on falling edge event (TAMPFLT=00 and TAMPxTRG=1) and if the tamper event detection is disabled when the tamper pin is at high level, a false tamper event is detected.

**Workaround**

The false tamper event detection cannot be avoided, but the backup registers erase can be avoided by setting TAMPxNOERASE bit before clearing TAMPxE bit. The two bits must be written in two separate RTC\_TAMPCR write accesses.

**2.14 USART limitations****2.14.1 Start bit detected too soon when sampling for NACK signal from the smartcard****Description**

In the ISO7816, when a character parity error is incorrect, the Smartcard receiver should transmit a NACK error signal at (10.5 +/- 0.2) etu after the character START bit falling edge. In this case, the USART transmitter should be able to detect correctly the NACK signal by sampling at (11.0 +/- 0.2) etu after the character START bit falling edge.

The USART peripheral used in Smartcard mode doesn't respect the (11 +/- 0.2) etu timing, and when the NACK falling edge arrives at 10.68 etu or later, the USART might misinterpret this transition as a START bit even if the NACK is correctly detected.

**Workaround**

None.

**2.14.2 Break request can prevent the Transmission Complete flag (TC) from being set****Description**

After the end of transmission of a data (D1), the Transmission Complete (TC) flag will not be set in the following conditions:

- CTS hardware flow control is enabled;
- D1 is being transmitted;
- abreak transfer is requested before the end of D1 transfer;
- nCTS is de-asserted before the end of transfer of D1.

**Workaround**

If the application needs to detect the end of transfer of the data, the break request should be done after making sure that the TC flag is set.

**2.14.3 nRTS is active while RE or UE = 0****Description**

The nRTS line is driven low as soon as the RTSE bit is set and even if the USART is disabled (UE = 0) or if the receiver is disabled (RE=0) i.e. not ready to receive data.

**Workaround**

Configure the I/O used for nRTS as an alternate function after setting the UE and RE bits.

## 2.15 DAC peripheral limitations

### 2.15.1 PA4 and PA5 must be configured in analog mode when DAC1\_OUT1 or DAC1\_OUT2 respectively is connected to OPAMP input

**Description**

When the DAC outputs are connected only to on-chip peripherals, the DAC1\_OUTx I/Os (PA4 or PA5) should be available for any other functions.

In case the DAC1\_OUT1 is used as OPAMP1 input, PA4 must be configured in analog mode to allow the connection. As a consequence, PA4 cannot be used for GPIO or alternate functions, but can still be used as ADC input.

**Workaround**

None.

### 2.15.2 PA4 and PA5 can't be used in output mode when DAC1\_OUT1 or DAC1\_OUT2 respectively is connected to on-chip peripherals

**Description**

When the DAC outputs are connected only to on-chip peripherals, the DAC1\_OUTx I/Os (PA4 or PA5) should be available for any other functions.

In case the DAC1\_OUT1 is configured for on-chip peripherals connection only, PA4 output buffer remains disable. As a consequence PA4 cannot be used in output mode (GPIO or alternate function), but can still be used in input or analog mode.

In case the DAC1\_OUT2 is configured for on-chip peripherals connection only, PA5 output buffer remains disable. As a consequence PA5 cannot be used in output mode (GPIO or alternate function), but can still be used in input or analog mode.

**Workaround**

None.

### 2.15.3 DAC cannot be used at low $V_{DDA}$ if APB1 frequency is higher than 40 MHz

**Description**

The DAC is not functional when  $V_{DDA}$  is lower than 2.7 V and APB1 frequency is higher than 40 MHz.

**Workaround**

When  $V_{DDA} < 2.7$  V, the APB1 frequency must be reduced below 40 MHz thanks to RCC prescalers.

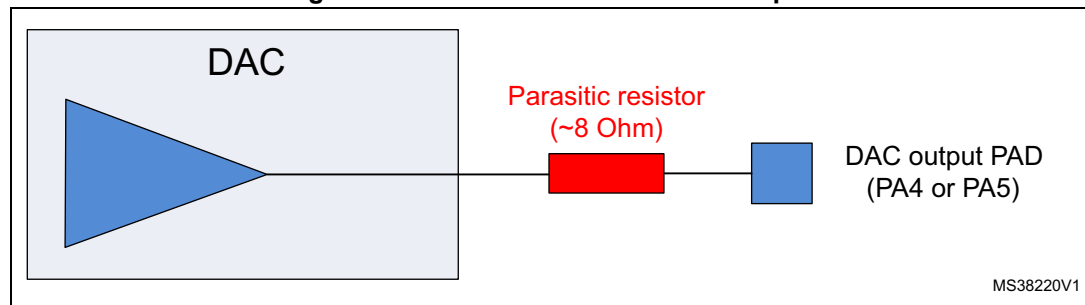


## 2.15.4 DAC characteristics can be degraded depending on the current sunk on DAC output

### Description

Resistor value between DAC outputs and I/O ( PA4 or PA5 ) is greater than expected: 8ohm typical, as shown in [Figure 2](#). As a consequence the DAC characteristics can be impacted depending on the current sunk on DAC output.

**Figure 2. Parasitic resistor on DAC output**



### Workaround

None.

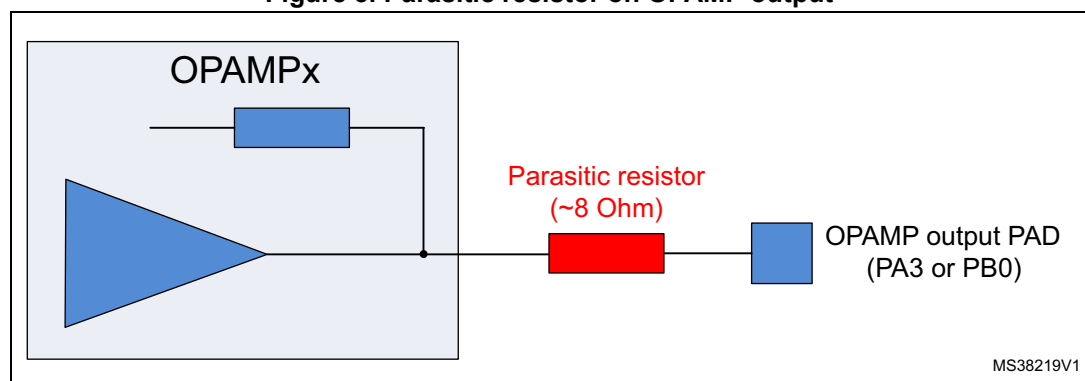
## 2.16 OPAMP limitations

### 2.16.1 OPAMP characteristics can be degraded depending on the current sunk on OPAMP output

#### Description

Resistor value between OPAMP outputs and I/O ( PA3 or PB0 ) is greater than expected: 8 ohm typical, as shown in [Figure 3](#). As a consequence the OPAMP characteristics can be impacted depending on the current sunk on the output.

**Figure 3. Parasitic resistor on OPAMP output**



### Workaround

None.

## 2.16.2 Calibration procedure does not work in PGA mode

### Description

During the offset calibration procedure, when the PGA mode is selected by setting OPAMODE=10, CALOUT signal is not reliable, resulting trim bit found in procedure is wrong.

### Workaround

Use the PGA disable or internal follower mode (OPAMODE=00, 01, or 11) during the calibration procedure.

## 2.17 COMP peripheral limitations

### 2.17.1 Comparators propagation delay is longer than expected for input steps higher than 200 mV

#### Description

[Table 5](#) summarizes the comparators propagation delay values. The propagation delay for steps higher than 200 mV is out of targeted specification.

**Table 5. COMP characteristics<sup>(1)</sup>**

Symbol	Parameter	Conditions		Min	Typ	Max	Unit
t <sub>D</sub>	Propagation delay for 200 mV step with 100 mV overdrive	High-speed mode	V <sub>DDA</sub> ≥ 2.7 V	-	55	80	ns
			V <sub>DDA</sub> < 2.7 V	-	65	100	
		Medium mode	V <sub>DDA</sub> ≥ 2.7 V	-	0.55	0.9	μs
			V <sub>DDA</sub> < 2.7 V	-	0.65	1.0	
		Ultralow-power mode	V <sub>DDA</sub> ≥ 2.7 V	-	5	12	
			V <sub>DDA</sub> < 2.7 V	-	5	12	
	Propagation delay for step > 200 mV with 100 mV overdrive on positive inputs	High-speed mode	-	-	0.1	1.0	μs
		Medium mode	-	-	2	3	
		Ultralow-power mode	-	-	8	24	

1. Data guaranteed by design, not tested in production, unless otherwise specified.

### Workaround

None.

### 2.17.2 Comparators output cannot be configured in open-drain

#### Description

Comparators output are always forced in push-pull mode whatever the GPIO output type configuration bit value.

**Workaround**

None.

**2.17.3 COMP1 and COMP2 configuration lost with software reset****Description**

The Comparators 1 and 2 control and status registers (COMP1\_CSR and COMP2\_CSR) should be protected from software changes when the LOCK bit is set. Therefore these registers should be reset only by a system reset. However, it is possible to reset these registers by setting SYSCFGRST in the RCC\_APB2RSTR.

**Workaround**

None.

**2.18 LPTIM peripheral limitations****2.18.1 Low power timer 1 (LPTIM1) outputs cannot be configured in open-drain****Description**

LPTIM1 outputs are always forced in push-pull mode whatever the GPIO output type configuration bit value.

**Workaround**

None.

**2.19 LPUART peripheral limitations****2.19.1 Low power UART1 (LPUART1) outputs cannot be configured in open-drain****Description**

LPUART1 outputs are always forced in push-pull mode whatever the GPIO output type configuration bit value.

**Workaround**

None.

## 2.20 AES peripheral limitations

### 2.20.1 Burst read or write accesses are not supported by the AES

#### Description

The AES only supports non-sequential read/write accesses.

AES registers read and write accesses using Load/Store multiple instruction LDM/STM are not supported.

LDRD, STRD are not supported neither.

#### Workaround

If LDM, STM, LDRD and STRD instructions are generated by the compiler to access AES registers, by reordering the software code by accessing higher memory address register to lower memory address register, compiler will not create multiple data instructions.

### 2.20.2 AES does not support Load multiple

#### Description

The AES does not support LDM, STM, LDRD and STRD instructions for successive multiple-data read and write accesses to a contiguous address block.

#### Workaround

The workaround consists in preventing compilers from generating LDM, STM, LDRD and STRD instructions.

In general, this can be achieved organizing the source code in a way that avoids consecutive read or write accesses to neighboring addresses in lower-to-higher order. In cases where consecutive read or write accesses to neighboring addresses cannot be avoided, the source code must be ordered to access higher address first.

### 3 Revision history

**Table 6. Document revision history**

Date	Revision	Changes
29-May-2015	1	Initial release.
25-Sep-2015	2	<p>Updated <a href="#">Table 2: Device summary</a> and <a href="#">Table 4: Summary of silicon limitation</a>.</p> <p>Updated <a href="#">Section 2.1.2: The configuration of the I/Os not available in WLCSP package can be modified by software</a>, <a href="#">Section 2.2.4: Read burst access of 9 words or more is not supported by FMC</a>, <a href="#">Section 2.4.5: Burst read or write accesses are not supported by the ADC</a>, <a href="#">Section 2.6.1: I2C Fast-mode Plus drive is not available on all SDA/SCL I/Os</a> and <a href="#">Section 2.6.2: Wrong behavior related with MCU Stop mode when wakeup from Stop mode by I2C peripheral is disabled</a>.</p> <p>Added <a href="#">Section 2.1.14: Full JTAG configuration without NJTRST pin cannot be used</a>, <a href="#">Section 2.6.4: Spurious Bus Error detection in master mode</a>, <a href="#">Section 2.6.5: I2C3 analog filters requires that both PC0/PC1 or both PG7/PG8 are configured as I2C3 alternate function</a> and <a href="#">Section 2.20: AES peripheral limitations</a>.</p>
03-Dec-2015	3	<p>Updated <a href="#">Table 4: Summary of silicon limitation</a>.</p> <p>Added <a href="#">Section 2.1.15: MSIRDY flag issue preventing entry in low power mode</a>, <a href="#">Section 2.16.2: Calibration procedure does not work in PGA mode</a> and <a href="#">Section 2.17.3: COMP1 and COMP2 configuration lost with software reset</a>.</p> <p>Updated <a href="#">Section 2.1.1: Write operation in the Flash memory while it is not ready (Flash memory in power-down) is not correctly handled</a>, <a href="#">Section 2.1.12: MSI at high frequency ranges cannot be used as wakeup from Stop 0 clock source</a>, <a href="#">Section 2.2.4: Read burst access of 9 words or more is not supported by FMC</a> and <a href="#">Section 2.12.4: SWP SUSPENDED mode not supported when STM32L4 is in Stop 0 or Stop 1 mode</a>.</p>

**IMPORTANT NOTICE – PLEASE READ CAREFULLY**

STMicroelectronics NV and its subsidiaries ("ST") reserve the right to make changes, corrections, enhancements, modifications, and improvements to ST products and/or to this document at any time without notice. Purchasers should obtain the latest relevant information on ST products before placing orders. ST products are sold pursuant to ST's terms and conditions of sale in place at the time of order acknowledgement.

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

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

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

ST and the ST logo are trademarks of ST. 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.

© 2015 STMicroelectronics – All rights reserved