STM32L100RC STM32L15xxC Errata sheet

STM32L100RC and STM32L15xxC ultra-low-power device limitations

Silicon identification

This errata sheet applies to revisions A, Y, X and V of STMicroelectronics STM32L100RC, and STM32L15xCC/RC/UC/VC(3) ultra-low-power products. This family features an ARM® 32-bit Cortex®-M3 core, for which an errata notice is also available (see *Section 1* for details).

The list of root part numbers and corresponding part numbers is given in *Table 1*.

The products can be identified (see *Table 2*) by:

- The revision code marked below the sales type on the device package
- The last three digits of the internal sales type printed on the box label

Table 1. Device summary

Reference	Part number
STM32L100RC	STM32L100RCT6
STM32L151CC STM32L151RC STM32L151UC STM32L151VC	STM32L151CCT6, STM32L151CCU6 STM32L151RCT6 STM32L151UCY6 STM32L151VCT6, STM32L151VCH6
STM32L152CC STM32L152RC STM32L152UC STM32L152VC	STM32L152CCT6, STM32L152CCU6 STM32L152RCT6 STM32L152UCY6 STM32L152VCT6, STM32L152VCH6

Table 2. Device identification(1)

Sales type	Revision code(2) marked on device
STM32L100RC STM32L15xCC/RC/UC/VC except products in WLCSP64 package (for example STM32L15xRCY) and except sales types ending with “A” (STM32L151RCT6A) (3)	"A", "Y", "X" or “V”

1. The REV_ID bits in the DBGMCU_IDCODE register show the revision code of the device (see the *STM32L151xx, STM32L152xx and STM32L162xx advanced ARM®-based 32-bit MCU* reference manual (RM0038) for details on how to find the revision code).
2. Refer to the device datasheets for details on how to identify the revision code on the different packages.
3. Device limitations for sales types ending with "A" and STM32L15xxC products in WLCSP64 package are described in STM32L15xxD/C/C-A errata sheet.

Contents

List of tables

1 ARM® 32-bit Cortex®-M3 limitations

An ARM errata notice of the STM32L1xxxx core is available searching for “Cortex®-M3 errata” at the following web address: www.arm.com.

All the described limitations are minor and relate to revision r2p0-00rel0 of the Cortex®-M3 core. *Table 3* summarizes these limitations and their implications on the behavior of the STM32L1xxxx ultra-low-power devices.

Table 3. Cortex®-M3 core limitations and impact on microcontroller behavior

ARM ID	ARM category	ARM summary of errata	Impact on STM32L1xxxx ultra-low-power devices
602117	Cat 2	LDRD with base in list may result in incorrect base register when interrupted or faulted	Minor
563915	Cat 2	Event register is not set by interrupts and debug	Minor
752419	Cat 2	Interrupted loads to SP can cause erroneous behavior	Minor
740455	Cat 2	SVC and BusFault/MemManage may occur out of order	Minor

1.1 Cortex®-M3 limitation description for the STM32L1xxxC ultra-low-power devices

Only the limitations described below have an impact, even though minor, on the implementation of STM32L1xxxx ultra-low-power devices.

All other limitations described in the ARM errata notice (and summarized in *Table 3* above) have no impact and are not related to the implementation of the STM32L1xxxx ultra-low-power devices (Cortex-M3 r2p0-00rel0).

1.1.1 Cortex®-M3 LDRD with base in list may result in incorrect base register when interrupted or faulted

Description

The Cortex®-M3 Core has a limitation when executing an LDRD instruction from the system-bus area, with the base register in a list of the form LDRD Ra, Rb, [Ra, #imm]. The execution may not complete after loading the first destination register due to an interrupt before the second loading completes or due to the second loading getting a bus fault.

Workaround

1. This limitation does not impact the STM32L1xxxx code execution when executing from the embedded Flash memory, which is the standard use of the microcontroller.
2. Use the latest compiler releases. As of today, the compilers no longer generate this particular sequence. Moreover, a scanning tool is provided to detect this sequence on previous releases (refer to your preferred compiler provider).

1.1.2 Cortex®-M3 event register is not set by interrupts and debug

Description

When interrupts related to a wake from event (WFE) occur before the WFE is executed, the event register used for WFE wakeup events is not set and the event is missed. Therefore, when the WFE is executed, the core does not wake up from a WFE if no other event or interrupt occurs.

Workaround

1. For the following interrupt sources:
   - all external interrupts/events lines (EXTI)
   - PVD output on EXTI line 16 (if VREFINT is enabled only)
   - RTC Alarm on EXTI line 17
   - USB Wake-up on EXTI line 18
   - RTC tamper and timestamp on EXTI line 19
   - RTC Wake-up on EXTI line 20
   - Comparator 1 wake-up on EXTI line 21 (if VREFINT is enabled only)
   - Comparator 2 wake-up on EXTI line 22 (if VREFINT is enabled only)
   - Channel acquisition on EXTI line 23

   Use STM32L1xxxx external events instead of interrupts to wake up the core from a WFE by configuring an external or internal EXTI line in event mode.
2. For all other interrupt sources, a timer must be programmed to provide a timeout event and wake-up the core if the event is likely to arrive before the WFE instruction is executed.

1.1.3 Cortex®-M3 Interrupted loads to the stack-pointer can cause erroneous behaviour

Description

If an interrupt occurs during the data-phase of a single word load to the stack-pointer (SP/R13), erroneous behavior can occur. In all cases, returning from the interrupt will result in the load instruction being executed an additional time. For all instructions performing an update to the base register, the base register will be 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

1.1.4 SVC and BusFault/MemManage may occur out of order

Description

If an SVC exception is generated by executing the SVC instruction while the following instruction fetch is faulted, then the MemManage or BusFault handler may be entered even though the faulted instruction which followed the SVC should not have been executed.

Workaround

A workaround is only required if the SVC handler will not return to the return address that has been stacked for the SVC exception and the instruction access after the SVC will fault. If this is the case then padding can be inserted between the SVC and the faulting area of code, for example, by inserting NOP instructions.

2 STM32L100RC and STM32L15xxC silicon limitations

Table 4 summarizes the fix status for the products listed in *Table 2: Device identification*.

The legend for *Table 4* is as follows:

A = workaround available,

N = no workaround available,

P = partial workaround available,

'-' and grayed = fixed.

Table 4. Summary of silicon limitations

Links to silicon limitations		Rev A	Rev Y	Rev X and V
Section 2.1: System limitations	*Section 2.1.1: If Debugger is connected in JTAG mode and JNTRST (PB4) pin configuration is changed, the connection is lost*	A	A	A
	Section 2.1.2: Read protection: a mass erase occurs if the RDP register is written with Level0 when Level0 is already set	A	A	A
	Section 2.1.3: Debugging Stop mode with WFE entry	A	A	A
	Section 2.1.4: Range 3 of dynamic voltage scaling cannot be used	A	-	-
	Section 2.1.5: The operational amplifier factory trimming value cannot be selected	A	A	A
	Section 2.1.6: Electrostatic discharge limits are 1kV (HBM) and 250 V (CDM) instead of 2 kV and 500 V respectively	N	-	-
	Section 2.1.7: Data EEPROM cycling limited to 100 kcycles	N	N	N
	Section 2.1.8: Range 1 of dynamic voltage scaling has a lower limit of 2.0 V	A	N	N
	Section 2.1.9: Wakeup sequence from Standby mode when using more than one wakeup source	A	A	A
	Section 2.1.10: BOOT0 pin sensitive to Electrostatic discharge	P	P	-
Section 2.2: IWDG peripheral limitation	*Section 2.2.1: RVU and PVU flags are not reset in STOP mode*	A	A	A
Section 2.3: I^2C peripheral limitations	*Section 2.3.1: SMBus standard not fully supported*	A	A	A
	Section 2.3.2: Wrong behavior of I^2C peripheral in Master mode after misplaced STOP	A	A	A
	Section 2.3.3: Violation of I^2C "setup time for repeated START condition" parameter	A	A	A
	Section 2.3.4: In I^2C slave "NOSTRETCH" mode, underrun errors may not be detected and may generate bus errors	A	A	A

Table 4. Summary of silicon limitations (continued)

Links to silicon limitations		Rev A	Rev Y	Rev X and V
Section 2.4: SPI/I2S peripheral limitations	*Section 2.4.1: In I2S slave mode, WS level must to be set by the external master when enabling the I2S*	A	A	A
	Section 2.4.2: BSY bit may stay high at the end of a data transfer at slave mode	A	A	A
	Section 2.4.3: Wrong CRC calculation when the polynomial is even	A	A	A
Section 2.5: USART peripheral limitations	*Section 2.5.1: Idle frame is not detected if receiver clock speed is deviated*	N	N	N
	Section 2.5.2: In full duplex mode, the Parity Error (PE) flag can be cleared by writing the data register	A	A	A
	Section 2.5.3: Parity Error (PE) flag is not set when receiving in Mute mode using address mark detection	N	N	N
	Section 2.5.4: Break frame is transmitted regardless of nCTS input line status	N	N	N
	Section 2.5.5: nRTS signal abnormally driven low after a protocol violation	A	A	A
	Section 2.5.6: Start bit detected too soon when sampling for NACK signal from the smartcard	N	N	N
	Section 2.5.7: Break request can prevent the Transmission Complete flag (TC) from being set	A	A	A
	Section 2.5.8: Guard time is not respected when data are sent on TXE events	A	A	A
	Section 2.5.9: nRTS is active while RE or UE = 0	A	A	A
Section 2.6: ADC peripheral limitation	*Section 2.6.1: ADC accuracy lowered*	N	-	-
Section 2.7: RCC peripheral limitation	*Section 2.7.1: Delay after an RCC peripheral clock enabling*	A	A	A
Section 2.8: RTC limitation	*Section 2.8.1: Spurious tamper detection when disabling the tamper channel*	N	N	N
Section 2.9: Package limitation	*Section 2.9.1: Thermal characteristics for UFQFPN48 package*	A	A	A

2.1 System limitations

2.1.1 If Debugger is connected in JTAG mode and JNTRST (PB4) pin configuration is changed, the connection is lost

Description

PB4 is configured by default in Alternate function mode after Reset.

When the configuration bit changes from Alternate function to Input, Analog or GPIO, the Reset signal connected to the CPU is tied to '0', and forces a Reset on the CPU TAP that stops the Debugger connection, even if the pin itself is pulled up.

Only JTAG mode with 4 wires is impacted. Serial Wire Debug (SWD) mode is not impacted.

Workaround

During the debug phase, when the debugger is connected in JTAG mode is used, I/O port PB4 should not be used by the application. If the application needs to use PB4 even during debug phase, the Debugger should use Serial Wire Debug (SWD) mode to connect.

2.1.2 Read protection: a mass erase occurs if the RDP register is written with Level0 when Level0 is already set

Description

The read protection ranges from the lowest level 0 (no read protection) to level 2 (disable debug/chip read protection). It is always possible to increase the read protection level without any side effects. It is not possible to decrease the protection level from level 2 to a lower level. It is possible to decrease the protection level from level 1 to level 0, but to avoid allowing access to data that were previously protected, a Mass Erase of the data EEPROM, program Flash and backup registers is performed.

The limitation appears when level 0 is requested (writing 0xAA in the Option byte RDP) while the protection Level is already at 0, in this case an unwanted mass erase is performed.

Workaround

Before setting Level0 in the RDP register, check that the current level is not equal to Level0.

- If the current level is not equal to Level0, Level0 can be activated.
- If the current level is Level0 then the RDP register must not be written again with Level0.

2.1.3 Debugging Stop mode with WFE entry

Description

When the Stop debug mode is enabled (DBG_STOP bit set in the DBGMCU_CR register) this allows software debugging during Stop mode. However, if the application software uses the WFE instruction to enter Stop mode, after wakeup some instructions could be missed if the WFE is followed by sequential instructions. This affects only Stop debug mode with WFE entry.

Workaround

To debug Stop mode with WFE entry, the WFE instruction must be inside a dedicated function with 1 instruction (NOP) between the execution of the WFE and the Bx LR.

Example: __asm void _WFE(void) {

WFE

NOP

BX lr }

2.1.4 Range 3 of dynamic voltage scaling cannot be used

Description

Dynamic voltage scaling is a power management technique which consists in increasing or decreasing the voltage used for the digital peripherals (VCORE) according on the circumstances.

Three voltage ranges can be configured. From those three voltage ranges, only Range 1 and Range 2 are functional. The low power/low performance Range 3 must not be selected.

Workaround

Use voltage scaling Range 2 instead of Range 3.

2.1.5 The operational amplifier factory trimming value cannot be selected

Description

The factory trimming values of the operational amplifiers are present but cannot be selected as trimming values.

Workaround

When the operational amplifiers are enabled, the trimming values should be initialized with the preset factory trimming value.

If you want to keep this behavior, you need to copy the factory trimming value to the user trimming values:

Copy content of address 0x1FF80048[15:0] to OPAMP_OTR[15:0].

Copy content of address 0x1FF8004C[3:0] to OPAMP_OTR[19:16].

Copy content of address 0x1FF80050[15:0] to OPAMP_LPOTR[15:0].

Copy content of address 0x1FF80054[3:0] to OPAMP_LPOTR[19:16].

2.1.6 Electrostatic discharge limits are 1kV (HBM) and 250 V (CDM) instead of 2 kV and 500 V respectively

Description

The ESD absolute maximum ratings are limited as follows:

Table 5. ESD absolute maximum ratings

Symbol	Ratings	Conditions	Class	Maximum value(1)	Unit
$V_{ESD(HBM)}$	Electrostatic discharge voltage (human body model)	T_A = +25 °C, conforming to JESD22-A114	2	1000	V
$V_{ESD(CDM)}$	Electrostatic discharge voltage (charge device model)	T_A = +25 °C, conforming to JESD22-C101	II	250	

1. Based on characterization results, not tested in production.

Workaround

None.

2.1.7 Data EEPROM cycling limited to 100 kcycles

The Data EEPROM, which usually supports 300 kcycles, supports only 100 kcycles. All the retention parameter are the same (replacing in the electrical characteristics 300 kcycles with 100 kcycles).

Workaround

None.

2.1.8 Range 1 of dynamic voltage scaling has a lower limit of 2.0 V

In newest revisions of documents, the range 1 of dynamic voltage scaling is available from VDD = 1.71 V and above. This does not apply to all revisions.

Workaround

The revisions impacted by this limitation can use range 1 when VDD is equal to or above 2.0 V.

2.1.9 Wakeup sequence from Standby mode when using more than one wakeup source

The various wakeup sources are logically OR-ed in front of the rising-edge detector which generates the wakeup flag (WUF). The WUF flag needs to be cleared prior to the Standby mode entry, otherwise the MCU wakes up immediately.

If one of the configured wakeup sources is kept high during the WUF flag clearing (by setting the CWUF bit), it may mask further wakeup events on the input of the edge detector. As a consequence, the MCU could not wake up from the Standby mode.

Workaround

To avoid this problem, the following sequence should be applied before entering the Standby mode:

1. Disable all used wakeup sources,
2. Clear all related wakeup flags,
3. Re-enable all used wakeup sources,
4. Enter the Standby mode.

Note: When applying this workaround, if one of the wakeup sources is still kept high, be aware that the MCU will enter the Standby mode but then wakes up immediately, generating the power reset.

2.1.10 BOOT0 pin sensitive to Electrostatic discharge

On some parts the Boot0 pin can be destroyed by electrostatic discharge.

Workaround

When different boot modes are not used, soldering the Boot0 pin to steady voltage removes the risk on final application.

2.2 IWDG peripheral limitation

2.2.1 RVU and PVU flags are not reset in STOP mode

Description

The RVU and PVU flags of the IWDG_SR register are set by hardware after a write access to the IWDG_RLR and the IWDG_PR registers, respectively. If the Stop mode is entered immediately after the write access, the RVU and PVU flags are not reset by hardware.

Before performing a second write operation to the IWDG_RLR or the IWDG_PR register, the application software must wait for the RVU or PVU flag to be reset. However, since the

RVU/PVU bit is not reset after exiting Stop mode, the software goes into an infinite loop and the independent watchdog (IWDG) generates a reset after the programmed timeout period.

Workaround

Wait until the RVU or PVU flag of the IWDG_SR register are reset before entering Stop mode.

2.3 I²C peripheral limitations

2.3.1 SMBus standard not fully supported

Description

The I²C peripheral is not fully compliant with the SMBus v2.0 standard since it does not support the capability to NACK an invalid byte/command.

Workaround

The following higher-level mechanisms should be used to verify that a write operation is being performed correctly at the target device:

1. The SMBA pin if supported by the host
2. The alert response address (ARA) protocol
3. The host notify protocol

2.3.2 Wrong behavior of I²C peripheral in Master mode after misplaced STOP

The I2C peripheral does not enter Master mode properly if a misplaced STOP is generated on the bus and the START bit is already set in I2C_CR2. In this case the START condition is not correctly generated on the bus and can create bus errors.

Workaround

In the I2C standard, it is not allowed to send a STOP before the full byte is transmitted (8 bits + acknowledge). Other derived protocols like CBUS allow it, but they are not supported by the I²C peripheral.

In case of noisy environment in which unwanted bus errors can occur, it is recommended to reset the I2C peripheral by setting the SWRST bit in the I2C_CR2 control register if a BERR is detected while the START bit is set in I2C_CR2.

No fix is planned for this limitation.

2.3.3 Violation of I²C “setup time for repeated START condition” parameter

Description

In case of a repeated Start, the “setup time for repeated START condition” parameter (named $t_{SU(STA)}$ in the datasheet and Tsu:sta in the I²C specifications) may be slightly violated when the I²C operates in Master Standard mode at a frequency ranging from 88 to 100 kHz. $t_{SU(STA)}$ minimum value may be 4 µs instead of 4.7 µs.

The issue occurs under the following conditions:

1. The I^{2}C peripheral operates in Master Standard mode at a frequency ranging from 88 to 100 kHz (no issue in Fast mode)
2. and the SCL rise time meets one of the following conditions:
   – The slave does not stretch the clock and the SCL rise time is more than 300 ns (the issue cannot occur when the SCL rise time is less than 300 ns).
   – or the slave stretches the clock.

Workaround

Reduce the frequency down to 88 kHz or use the I^{2}C Fast mode if it is supported by the slave.

2.3.4 In I^{2}C slave “NOSTRETCH” mode, underrun errors may not be detected and may generate bus errors

Description

The data valid time ($t_{VD;DAT}$, $t_{VD;ACK}$) described by the I^{2}C specifications may be violated as well as the maximum current data hold time ($t_{HD;DAT}$) under the conditions described below. In addition, if the data register is written too late and close to the SCL rising edge, an error may be generated on the bus: SDA toggles while SCL is high. These violations cannot be detected because the OVR flag is not set (no transmit buffer underrun is detected).

This issue occurs under the following conditions:

1. The I^{2}C peripheral operates In Slave transmit mode with clock stretching disabled (NOSTRETCH=1)
2. and the application is late to write the DR data register, but not late enough to set the OVR flag (the data register is written before the SCL rising edge).

Workaround

If the master device supports it, use the clock stretching mechanism by programming the bit NOSTRETCH=0 in the I2C_CR1 register.

If the master device does not support it, ensure that the write operation to the data register is performed just after TXE or ADDR events. You can use an interrupt on the TXE or ADDR flag and boost its priority to the higher level or use DMA.

Using the “NOSTRETCH” mode with a slow I^{2}C bus speed can prevent the application from being late to write the DR register (second condition).

Note: The first data to be transmitted must be written into the data register after the ADDR flag is cleared, and before the next SCL rising edge, so that the time window to write the first data into the data register is less than t_{LOW}.

If this is not possible, a possible workaround can be the following:

1. Clear the ADDR flag
2. Wait for the OVR flag to be set
3. Clear OVR and write the first data.

The time window for writing the next data is then the time to transfer one byte. In that case, the master must discard the first received data.

2.4 SPI/I2S peripheral limitations

2.4.1 In I2S slave mode, WS level must to be set by the external master when enabling the I2S

Description

In slave mode the WS signal level is used only to start the communication. If the I2S (in slave mode) is enabled while the master is already sending the clock and the WS signal level is low (for I2S protocol) or is high (for the LSB or MSB-justified mode), the slave starts communicating data immediately. In this case the master and slave will be desynchronized throughout the whole communication.

Workaround

The I2S peripheral must be enabled when the external master sets the WS line at:

- High level when the I2S protocol is selected.
- Low level when the LSB or MSB-justified mode is selected.

2.4.2 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.

2.4.3 Wrong CRC calculation when the polynomial is even

Description

When the CRC is enabled, the CRC calculation will be wrong if the polynomial is even.

Workaround

Use odd polynomial.

2.5 USART peripheral limitations

2.5.1 Idle frame is not detected if receiver clock speed is deviated

Description

If the USART receives an idle frame followed by a character, and the clock of the transmitter device is faster than the USART receiver clock, the USART receive signal falls too early when receiving the character start bit, with the result that the idle frame is not detected (IDLE flag is not set).

Workaround

None.

2.5.2 In full duplex mode, the Parity Error (PE) flag can be cleared by writing the data register

Description

In full duplex mode, when the Parity Error flag is set by the receiver at the end of a reception, it may be cleared while transmitting by reading the USART_SR register to check the TXE or TC flags and writing data in the data register.

Consequently, the software receiver can read the PE flag as '0' even if a parity error occurred.

Workaround

The Parity Error flag should be checked after the end of reception and before transmission.

2.5.3 Parity Error (PE) flag is not set when receiving in Mute mode using address mark detection

Description

The USART receiver is in Mute mode and is configured to exit the Mute mode using the address mark detection. When the USART receiver recognizes a valid address with a parity error, it exits the Mute mode without setting the Parity Error flag.

Workaround

None.

2.5.4 Break frame is transmitted regardless of nCTS input line status

Description

When CTS hardware flow control is enabled (CTSE = 1) and the Send Break bit (SBK) is set, the transmitter sends a break frame at the end of current transmission regardless of nCTS input line status.

Consequently, if an external receiver device is not ready to accept a frame, the transmitted break frame is lost.

Workaround

None.

2.5.5 nRTS signal abnormally driven low after a protocol violation

Description

When RTS hardware flow control is enabled, the nRTS signal goes high when a data is received. If this data was not read and a new data is sent to the USART (protocol violation), the nRTS signal goes back to low level at the end of this new data.

Consequently, the sender gets the wrong information that the USART is ready to receive further data.

On USART side, an overrun is detected which indicates that some data has been lost.

Workaround

Workarounds are required only if the other USART device violates the communication protocol which is not the case in most applications.

Two workarounds can be used:

- After data reception and before reading in the data in the data register, the software takes over the control of the nRTS signal as a GPIO and holds it high as long as needed. If the USART device is not ready, the software holds the nRTS pin high, and releases it when the device is ready to receive new data.
- The time required by the software to read the received data must always be lower than the duration of the second data reception. For example, this can be ensured by treating all the receptions by DMA mode.

2.5.6 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 shall 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.

2.5.7 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.
- A break 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.

2.5.8 Guard time is not respected when data are sent on TXE events

Description

In smartcard mode, when sending a data on TXE event, the programmed guard time is not respected i.e. the data written in the data register is transferred on the bus without waiting for the completion of the guardtime duration corresponding to the previous transmitted data.

Workaround

Write the data after TC is set because in smartcard mode, the TC flag is set at the end of the guard time duration.

2.5.9 nRTS is active while RE or UE = 0

Description

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

Workaround

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

2.6 ADC peripheral limitation

2.6.1 ADC accuracy lowered

Workaround

Some ADC accuracy characteristics have maximum errors increased. See updated table below.

Table 6. ADC accuracy(1)(2)

Symbol	Parameter	Test conditions	Min(3)	Typ	Max(3)	Unit
ET	Total unadjusted error	2.4 V ≤V_{DDA} ≤ 3.6 V 2.4 V ≤V_{REF+} ≤ 3.6 V f_{ADC} = 8 MHz, R_{AIN} = 50 Ω T_A = -40 to 105 °C	-	2	5	LSB
EO	Offset error		-	1	3	
EG	Gain error		-	1.5	3.5	
ED	Differential linearity error		-	1	2	
EL	Integral linearity error		-	1.7	3	
ENOB	Effective number of bits	2.4 V ≤V_{DDA} ≤ 3.6 V V_{DDA} = V_{REF+} f_{ADC} = 16 MHz, R_{AIN} = 50 Ω T_A = -40 to 105 °C 1 kHz ≤F_{input} ≤ 100 kHz	9.2	10	-	bit
SINAD	Signal-to-noise and distortion ratio		57.5	62	-	dB
SNR	Signal-to-noise ratio		57.5	62	-	
THD	Total harmonic distortion		-74	-75	-	
ET	Total unadjusted error	2.4 V ≤V_{DDA} ≤ 3.6 V 1.8 V ≤V_{REF+} ≤ 2.4 V f_{ADC} = 4 MHz, R_{AIN} = 50 Ω T_A = -40 to 105 °C	-	4	7	LSB
EO	Offset error		-	2	5	
EG	Gain error		-	4	6	
ED	Differential linearity error		-	1	2	
EL	Integral linearity error		-	1.5	4	
ET	Total unadjusted error	1.8 V ≤V_{DDA} ≤ 2.4 V 1.8 V ≤V_{REF+} ≤ 2.4 V f_{ADC} = 4 MHz, R_{AIN} = 50 Ω T_A = -40 to 105 °C	-	2	5	LSB
EO	Offset error		-	1	3	
EG	Gain error		-	1.5	2	
ED	Differential linearity error		-	1	2	
EL	Integral linearity error		-	1	3	

1. ADC DC accuracy values are measured after internal calibration.
2. ADC accuracy vs. negative injection current: Injecting a negative current on any analog input pins should be avoided as this significantly reduces the accuracy of the conversion being performed on another analog input. It is recommended to add a Schottky diode (pin to ground) to analog pins which may potentially inject negative currents.
   Any positive injection current within the limits specified for $I_{INJ(PIN)}$ and $\Sigma I_{INJ(PIN)}$ in *Section "I/O current injection characteristics" of relevant datasheet* does not affect the ADC accuracy.
3. Based on characterization, not tested in production.

Workaround

None.

2.7 RCC peripheral limitation

2.7.1 Delay after an RCC peripheral clock enabling

Description

A delay between an RCC peripheral clock enable and the effective peripheral enabling should be taken into account in order to manage the peripheral read/write to registers.

This delay depends on the peripheral's mapping:

- If the peripheral is mapped on AHB: the delay should be equal to 2 AHB cycles.
- If the peripheral is mapped on APB: the delay should be equal to 1 + (AHB/APB prescaler) cycles.

Workarounds

1. Use the DSB instruction to stall the Cortex-M CPU pipeline until the instruction is completed.
2. Insert "n" NOPs between the RCC enable bit write and the peripheral register writes (n = 2 for AHB peripherals, n = 1 + AHB/APB prescaler in case of APB peripherals).

2.8 RTC limitation

2.8.1 Spurious tamper detection when disabling the tamper channel

Description

If the tamper detection is configured for detection on the 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

None.

2.9 Package limitation

2.9.1 Thermal characteristics for UFQFPN48 package

Description

The thermal resistance junction-ambient for UFQFPN48 package was not correctly specified in STM32L15xxC datasheet prior revision 7.

Workarounds

Limit the power dissipation of the device to not exceed the maximum chip-junction temperature for the maximum ambient temperature at which the device is operated. For guidance and correct thermal resistance value, see the latest datasheet available on the company website.

Revision history

Table 7. Document revision history

Date	Revision	Changes
13-Feb-2014	1	Initial release.
15-Oct-2014	2	Added *Section 2.4.2: BSY bit may stay high at the end of a data transfer at slave mode*.
21-Apr-2015	3	Removed appendix A section: all package markings put in the corresponding datasheets. Added USART limitations: - *Section 2.5.6: Start bit detected too soon when sampling for NACK signal from the smartcard*. - *Section 2.5.7: Break request can prevent the Transmission Complete flag (TC) from being set*. - *Section 2.5.8: Guard time is not respected when data are sent on TXE events*. - *Section 2.5.9: nRTS is active while RE or UE = 0*. Added SPI limitation: - *Section 2.4.3: Wrong CRC calculation when the polynomial is even*. Added RTC limitation: - *Section 2.8.1: Spurious tamper detection when disabling the tamper channel*.
03-Aug-2015	4	Added revision V with same limitations as revision X.