STM32L151xx and STM32L152xx Errata sheet

STM32L151xx and STM32L152xx ultralow power limitations

Silicon identification

This errata sheet applies to the revision W of the STMicroelectronics STM32L151xx and STM32L152xx ultralow 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).

A full list of root part numbers is shown 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
STM32L151xx	STM32L151CB, STM32L151RB, STM32L151VB, STM32L151C8 STM32L151R8, STM32L151V8
STM32L152xx	STM32L152CB, STM32L152RB, STM32L152VB, STM32L152C8 STM32L152R8, STM32L152V8

Table 2. Device identification(1)

Sales type	Revision code(2) marked on device
STM32L151xx	“W”
STM32L152xx	“W”

1. The REV_ID bits in the DBGMCU_IDCODE register show the revision code of the device (see the STM32L15xxx reference manual for details on how to find the revision code).
2. Refer to *Appendix A: Revision and date codes on device marking* for details on how to identify the revision code on the different packages.

Contents

List of tables

List of figures

1 ARM™ 32-bit Cortex®-M3 limitations

An ARM errata notice of the STM32L15xxx core is available from the following web address: *http://infocenter.arm.com/help/index.jsp?topic=/com.arm.doc.eat0420c/index.html.*

The direct link to the errata notice pdf is:
http://infocenter.arm.com/help/topic/com.arm.doc.eat0420c/Cortex-M3-Errata-r2p0-v2.pdf

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 STM32L151xx / STM32L152xx 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 STM32L151xx / STM32L152xx ultralow 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

1.1 Cortex-M3 limitation description for the STM32L151xx / STM32L152xx ultra low power devices

Only the limitations described below have an impact, even though minor, on the implementation of STM32L151xx / STM32L152xx ultralow 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 STM32L151xx / STM32L152xx ultralow 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.

Workarounds

1. This limitation does not impact the STM32L15xxx 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.

Workarounds:

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)

   use STM32L15x 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.

2 STM32L15xxx silicon limitations

Table 4 gives a summary of the fix status.

Legend for *Table 4*: A = workaround available; N = no workaround available; P = partial workaround available, '-' and grayed = fixed.

Table 4. Summary of silicon limitations in revision W devices

Links to silicon limitations		Rev Y (cut 1.1)	Rev X (cut 1.2)	Rev W (cut 1.3)
Section 2.1: System limitations	*Section 2.1.1: Unexpected Flash/EEPROM behavior on system reset during programming/erasing*	N	-	-
	Section 2.1.2: Bootloader unavailability on STM32L151xx and STM32L152xx devices	N	-	-
	Section 2.1.3: Undefined instruction exception during IAP	A	-	-
	Section 2.1.4: Factory trimming values not available	A	-	-
	Section 2.1.5: Debug support for low power modes with entry through the WFE instruction	A	A	A
	Section 2.1.6: Operating temperature range limited to "-10°C to +85°C"	N	N	-
	Section 2.1.7: MCU may not restart after a reset when using HSE bypass as main clock source	N	N	-
Section 2.2: I2C peripheral limitations	*Section 2.2.1: SMBus standard not fully supported*	A	A	A
	Section 2.2.2: Wrong behavior of I2C peripheral in Master mode after misplaced STOP	A	A	A
	Section 2.2.3: Violation of I2C "setup time for repeated START condition" parameter	A	A	A
	Section 2.2.4: In I2C slave "NOSTRETCH" mode, underrun errors may not be detected and may generate bus errors	A	A	A
Section 2.3: SPI peripheral limitations	*CRC still sensitive to communication clock when SPI is in slave mode even with NSS high*	A	A	A
Section 2.4: ADC peripheral limitations	*Section 2.4.1: ADC converter partially tested on parts with date code before 047*	A	-	-

2.1 System limitations

2.1.1 Unexpected Flash/EEPROM behavior on system reset during programming/erasing

Description

Part of the internal EEPROM state machine is not correctly reset if the reset occurs during a programming or reset phase. This can cause the MCU either to take several seconds to exit from the reset phase or need a power cycling to resume operation.

Workaround

None.l

2.1.2 Bootloader unavailability on STM32L151xx and STM32L152xx devices

Description

The boot loader cannot be used due the issue described in *Section 2.1.3: Undefined instruction exception during IAP*.

Workaround

No workaround is available.

2.1.3 Undefined instruction exception during IAP

Description

The Flash memory can return a corrupted data following a programming of the Flash itself or the EEPROM area and possibly cause a hard fault exception due to undefined instruction.

The behavior depends on the location of the routine which is executing the programming. It occurs only when IAP is executed from the upper part of memory (0x08000000 to 0x0800 FFFF)

The issue can occur when the bootloader is used.

Workaround

There are two possible workarounds

1. 1. The IAP routines must be located in the bottom part of the memory (0x08010000 to 0x0801 FFFF)

or

2. The IAP routine must be relocated in RAM, and programming followed by a dummy read in Flash before resuming execution from the Flash memory

Before branching to the RAM, the following procedure must be followed:

The vector table and all exception and interrupt handlers that might be triggered must be copied into the RAM memory. The vector table offset must be updated in the SCB_VTOR register of the Cortex-M3.

The amount of RAM memory needed can be minimized by disabling all interrupts during the IAP execution. This can be done with the void __disable_irq(void) function from the CMSIS library. Similarly, the Clock Security System (CSS) can be disabled to prevent any NMI from occurring. However, it is mandatory to remap at least the Cortex M3's fault handlers.

2.1.4 Factory trimming values not available

Description

The following parameters are not available in the non-volatile memory factory trimming area:

- V_{REFINT} value
- Temperature sensor value at 90 °C

Workaround

Parameters can be measured on the customer production line and stored in the data EEPROM.

2.1.5 Debug support for low power modes with entry through the WFE instruction

Description

The DBG_STOP and DBG_SLEEP bits in the DBGMCU_CR register can be used to debug an application low power mode without loosing the JTAG connection.

The application may not resume correctly in the following cases:

- the DBG_STOP bit is set and the WFE instruction is used
- the DBG_SLEEP bit is set, either the SRAMLPEN or the FLTFEN bits are set (in the RCC_AHBLPENR register) and the WFE instruction is used.

This affects only the debug of Stop mode and Sleep modes with an entry into WFE mode. Low power modes are not affected when the MCU is not in debug configuration.

Workaround

The WFE instruction must be executed in a dedicated function with 1 instruction (NOP) between the execution of the WFE and the BX LR.

```
__asm void _WFE(void)
{
WFE
NOP
BX lr
}
```

2.1.6 Operating temperature range limited to “-10°C to +85°C”

Description

The microcontroller has an operating temperature range of -10°C to +85°C.

Workaround

None.

2.1.7 MCU may not restart after a reset when using HSE bypass as main clock source

Description

When the system clock source is the HSE bypass (for instance, external oscillator), the MCU may not restart after a reset event.

Workaround

None.

2.2 I²C peripheral limitations

2.2.1 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.

Workarounds

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

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

Description

The I²C peripheral does not enter Master mode properly if a misplaced STOP is generated on the bus. This can happen in the following conditions:

- If a void message is received (START condition immediately followed by a STOP): the BERR (bus error) flag is not set, and the I²C peripheral is not able to send a START condition on the bus after writing to the START bit in the I2C_CR2 register.
- In the other cases of a misplaced STOP, the BERR flag is set in the IC2_CR2 register. If the START bit is already set in I2C_CR2, 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 implement a timeout to ensure that the SB (start bit) flag is set after the START control bit is set. In case the timeout has elapsed, the peripheral must be reset by setting the SWRST bit in the I2C_CR2 control register. The I²C peripheral should be reset in the same way if a BERR is detected while the START bit is set in I2C_CR2.

No fix is planned for this limitation.

2.2.3 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 I²C 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 I²C Fast mode if it is supported by the slave.

2.2.4 In I²C 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 I²C 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 I²C 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 I²C 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.

2.3 SPI peripheral limitations

2.3.1 CRC still sensitive to communication clock when SPI is in slave mode even with NSS high

Description

When the SPI is configured in slave mode with the CRC feature enabled, the CRC is calculated even if the NSS pin deselects the SPI (high level applied on the NSS pin).

Workaround

The CRC has to be cleared on both Master and Slave sides between the slave deselection (high level on NSS) and the slave selection (low level on NSS), in order to resynchronize the Master and Slave for their respective CRC calculation.

The procedure to clear the CRC is as follows:

1. Disable the SPI (SPE = 0)
2. Clear the CRCEN bit
3. Set the CRCEN bit
4. Enable the SPI (SPE = 1)

2.4 ADC peripheral limitations

2.4.1 ADC converter partially tested on parts with date code before 047

Description

The ADC converter production test is partially implemented for parts with date code before 047. As a result, it must be anticipated that 0.15% of parts will have ADC failures.

Appendix A Revision and date codes on device marking

Figure 1 and *Figure 2* show the marking compositions for the LQFP100 and LQFP64 packages, respectively. The only fields shown are the “additional” field, containing the revision code, and the “year” and “week” fields making up the date code.

Figure 1. LQFP100 top package view

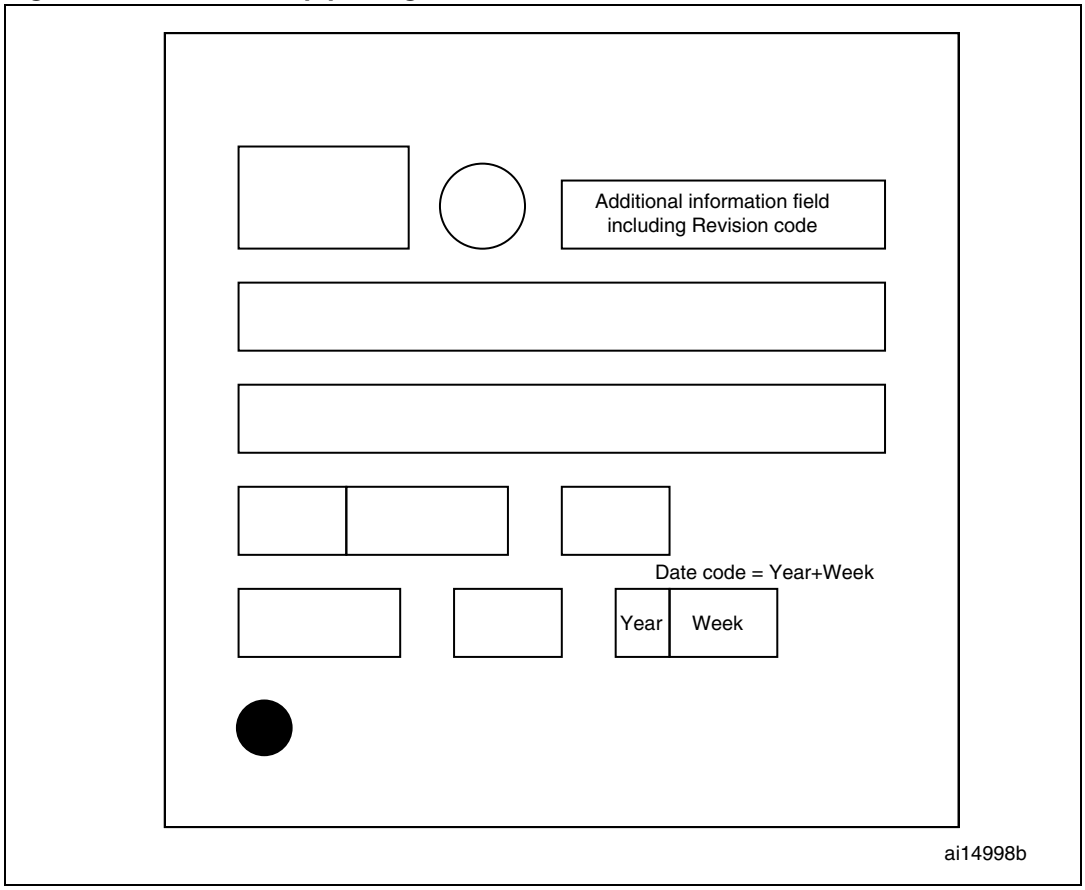
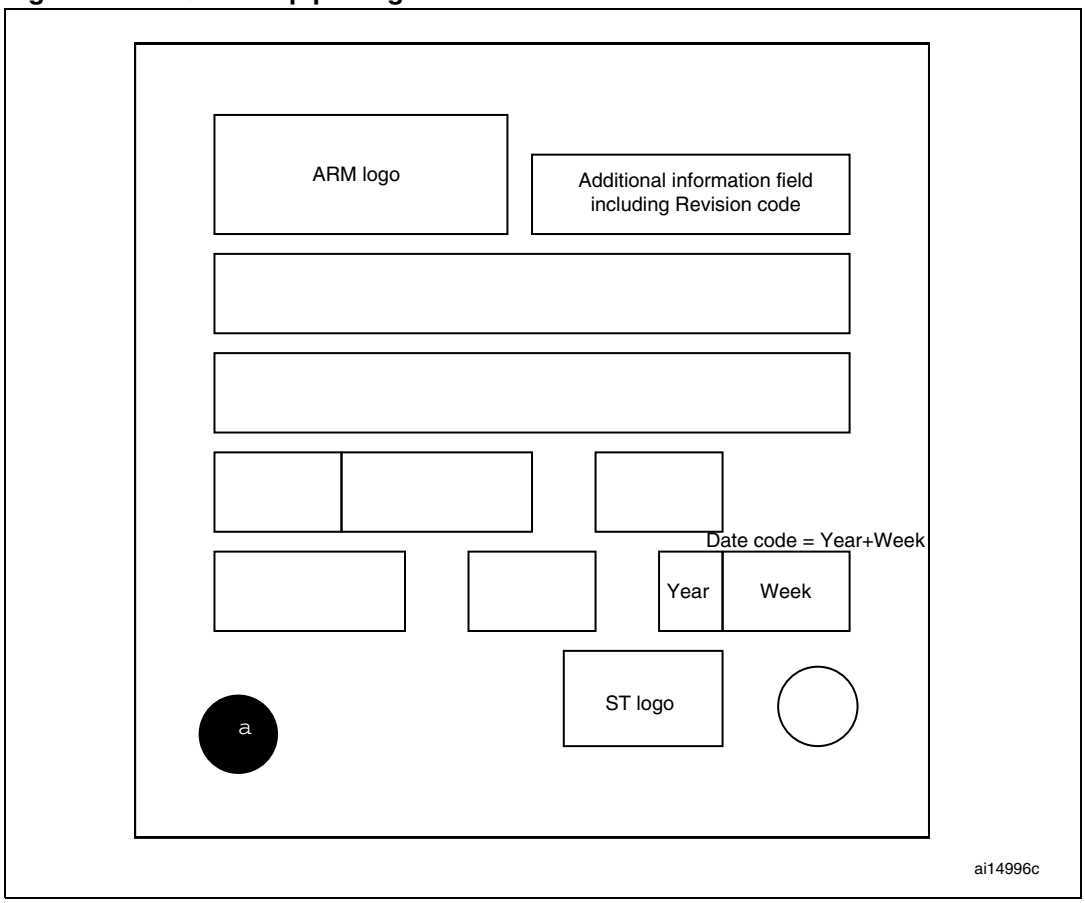


Figure 2. LQFP64 top package view



Revision history

Table 5. Document revision history

Date	Revision	Changes
05-Jul-2010	1	Initial release.
01-Oct-2010	2	Added workarounds under *Section 1.1.2: Cortex-M3 event register is not set by interrupts and debug*. Added *Section 2.1: System limitations*. Changed workaround under *Section 2.4: Extra consumption in STOP / STANDBY mode*. Added *Section 2.2: I2C peripheral limitations*. Added *Section 2.6: POR start-up delay at cold temperature*.
06-Dec-2010	3	Removed sections describing Rev A issues fixed in Rev B. Added *Section 2.1.3: Undefined instruction exception during IAP*. Updated *Section 2.1.4: Factory trimming values not available*.
19-Jan-2011	4	Replaced Rev B by Rev Y. Modified descriptions of *Section 2.2.2: Wrong behavior of I2C peripheral in Master mode after misplaced STOP*, *Section 2.2.3: Violation of I2C "setup time for repeated START condition" parameter* and *Section 2.2.4: In I2C slave "NOSTRETCH" mode, underrun errors may not be detected and may generate bus errors*. Updated *Table 4*.
25-Feb-2011	5	Added description of Rev W. Added *Section 2.1.6: Operating temperature range limited to "-10°C to +85°C"* and *Section 2.1.7: MCU may not restart after a reset when using HSE bypass as main clock source*.