STM32F030x4/6/8/C Errata sheet

STM32F030x4/6/8/C Rev A and Rev B device limitations

Silicon identification

This errata sheet applies to revisions A and B of the STMicroelectronics STM32F030x4/6/8/C products. STM32F030x4/6/8/C devices feature an ARM® 32-bit Cortex®-M0 core.

Section 1 gives a detailed description of the product silicon limitations.

Table 2 shows the full list of part numbers concerned by these limitations.

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(1)

Sales type	Revision code marked on the device(2)
STM32F030x4	A
STM32F030x6	A
STM32F030x8	B
STM32F030xC	A

1. The REV_ID bits in the DBGMCU_IDCODE register show the revision code of the device (see the STM32F0x0xx reference manual (RM0360) for details on how to find the revision code).
2. Refer to the device marking sections in STM32F030x4/6/8/C datasheet for details on how to identify the revision code according to the packages.

Table 2. Device summary

Reference	Part number
STM32F030x4	STM32F030F4
STM32F030x6	STM32F030C6, STM32F030K6
STM32F030x8	STM32F030C8, STM32F030R8
STM32F030xC	STM32F030CC, STM32F030RC

Contents

1 STM32F030x4/6/8/C silicon limitations

Table 3 gives quick references to all documented limitations.

Legend for *Table 3*:

A = workaround available,

N = no workaround available,

P = partial workaround available,

'-' grayed = fixed.

Table 3. Summary of silicon limitations

Section	Limitation	Rev A, B
Section 1.1: System limitations	*Section 1.1.1: Wakeup sequence from Standby mode when using more than one wakeup source*	A
Section 1.2: USART peripheral limitation	*Section 1.2.2: Last byte written in TDR might not be transmitted if TE is cleared just after writing in TDR*	A
Section 1.3: GPIO peripheral limitations	*Section 1.3.1: Extra consumption on GPIOs PC0..5 on 48-pin and PB0 on 20-pin devices*	A
	Section 1.3.2: GPIOx locking mechanism not working properly for GPIOx_OTYPER register	P
Section 1.4: I^2C peripheral limitations	*Section 1.4.1: 10-bit slave mode: wrong direction bit value after Read header reception*	A
	Section 1.4.2: 10-bit combined with 7-bit slave mode: ADDCODE may indicate wrong slave address detection	N
	Section 1.4.3: Wrong behaviors in Stop mode	A
Section 1.5: SPI peripheral limitations	*Section 1.5.1: Packing mode limitation at reception*	N

1.1 System limitations

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

Description

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

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

Workaround

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

- Disable all used wakeup sources,
- Clear all related wakeup flags,
- Re-enable all used wakeup sources,
- Enter Standby mode

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

1.2 USART peripheral limitation

1.2.1 Communication parameters reprogramming after ATR in Smartcard mode when SCLK is used to clock the card

Description

If the USART is used in Smartcard mode and the card cannot use the default communication parameters after Answer To Reset and doesn't support clock stop, it is not possible to use SCLK to clock the card. This is due to the fact that the USART and its clock output must be disabled while reprogramming some of the parameters.

Workaround

Use another clock source to clock the card (e.g. a timer output programmed to the desired clock frequency).

1.2.2 Last byte written in TDR might not be transmitted if TE is cleared just after writing in TDR

Description

If the USART clock source is slow (for example LSE) and TE bit is cleared immediately after the last write to TDR, the last byte will probably not be transmitted.

Workarounds

1. Wait until TXE flag is set before clearing TE bit
2. Wait until TC flag is set before clearing TE bit

1.3 GPIO peripheral limitations

1.3.1 Extra consumption on GPIOs PC0..5 on 48-pin and PB0 on 20-pin devices

Description

For lower pin count devices, some GPIOs are not available on the package. The hardware forces them to safe configuration.

Software reconfiguration of PB0 on STM32F030F4 and PC0..5 on STM32F030C8 to analog mode opens a path between VDDA and VDDIO. Additional current consumption in the range of tens of µA per pin can be observed if VDDA is higher than VDDIO.

Workaround

Do not reconfigure listed pins to the analog mode on 20/48 pin packages.

1.3.2 GPIOx locking mechanism not working properly for GPIOx_OTYPER register

Description

Locking of GPIOx_OTYPER[i] with i = 15..8 depends from setting of GPIOx_LCKR[i-8] and not from GPIOx_LCKR[i]. GPIOx_LCKR[i-8] is locking GPIOx_OTYPER[i] together with GPIOx_OTYPER[i-8]. It is not possible to lock GPIOx_OTYPER[i] with i = 15...8, without locking also GPIOx_OTYPER[i-8].

Workaround

The only way to lock GPIOx_OTYPER[i] with i=15..8 is to lock also GPIOx_OTYPER[i-8].

1.4 I²C peripheral limitations

1.4.1 10-bit slave mode: wrong direction bit value after Read header reception

Description

Under specific conditions, the transfer direction bit DIR (bit 16 of status register I2C_ISR) is low instead of high after reception of the 10-bit addressing Read header. Nevertheless, the I²C operates correctly in slave transmission mode, and data can be sent using the TXIS flag.

To see the limitation, all the following conditions have to be fulfilled:

- I²C has to be configured in 10-bit addressing mode (OA1MODE is set in the I2C_OAR1 register).
- The high LSBs of the I²C slave address are equal to the 10-bit addressing Read header value (i.e. OA1[7:3] = 11110, OA1[2] = OA1[9], OA1[1] = OA1[8] and OA1[0] = 1 in the I2C_OAR1 register).
- The I²C receives the 10-bit addressing Read header (0X 1111 0XX1) after the repeated start condition to enter slave transmission mode.

As a result, the DIR bit is incorrect in slave mode under specific conditions.

Workaround

If possible, do not use these four values as 10-bit addresses in slave mode:

- OA1[9:0] = 0011110001
- OA1[9:0] = 0111110011
- OA1[9:0] = 1011110101
- OA1[9:0] = 1111110111

If one of these addresses is the I²C slave address, the DIR bit must not be used in the FW.

1.4.2 10-bit combined with 7-bit slave mode: ADDCODE may indicate wrong slave address detection

Description

Under specific conditions, the ADDCODE (Address match code) in the I2C_ISR register indicates a wrong slave address.

To see the limitation, all the following conditions have to be fulfilled:

- The I^2C slave address OA1 is enabled and configured in 10-bit mode (OA1EN=1 and OA1MODE=1)
- Another 7-bit slave address is enabled and the bits 1 to 7 of the 10-bit slave address OA1 are equal to the 7-bit slave address, i.e., one of the configurations below is set:
  - OA2EN=1 and OA2MSK = 0 and OA1[7:1] = OA2[7:1]
  - OA2EN=1 and OA2MSK = 1 and OA1[7:2] = OA2[7:2]
  - OA2EN=1 and OA2MSK = 2 and OA1[7:3] = OA2[7:3]
  - OA2EN=1 and OA2MSK = 3 and OA1[7:4] = OA2[7:4]
  - OA2EN=1 and OA2MSK = 4 and OA1[7:5] = OA2[7:5]
  - OA2EN=1 and OA2MSK = 5 and OA1[7:6] = OA2[7:6]
  - OA2EN=1 and OA2MSK = 6 and OA1[7] = OA2[7]
  - OA2EN=1 and OA2MSK = 7
  - GCEN=1 and OA1[7:1] = 0b0000000
  - ALERTEN=1 and OA1[7:1] = 0b0001100
  - SMBDEN=1 and OA1[7:1] = 0b1100001
  - SMBHEN=1 and OA1[7:1] = 0b0001000
- The master starts a transfer addressed to the 10-bit slave address OA1.

As a result, after the address reception, the ADDCODE value is OA1[7:1] equal to the 7-bit slave address, instead of 0b11110 & OA1[9:8].

Workaround

None. If several slave addresses are enabled, mixing 10-bit and 7-bit addresses, the 10-bit Slave address OA1 [7:1] must not be equal to the 7-bit slave address.

1.4.3 Wrong behaviors in Stop mode

Description

When the MCU enters Stop mode while a transfer is on going on the bus, some wrong behaviors may happen:

1. BUSY flag can 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.
2. If clock stretching is enabled (NOSTRETCH = 0), the I2C clock SCL may be stretched low by the I2C as long as the MCU is in Stop mode. This limitation may occur when the Stop mode is entered during the address phase of a transfer on the I2C bus while SCL = 0. Therefore the transfer may be stalled as long as the MCU is in Stop mode. The probability of the occurrence depends also on the timings configuration, the peripheral clock frequency and the I2C bus frequency.

These behaviors can occur in Slave mode and in Master mode in a multi-master topology.

Workaround

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

1.5 SPI peripheral limitations

1.5.1 Packing mode limitation at reception

Description

When the SPI is configured in the short data frame mode, the packing mode on the reception side may not be usable. Using this feature may generate a wrong RXNE event to an Interrupt or DMA request and so the software may read back inconsistent data with FIFO pointers misalignment on the reception FIFO.

The worst case is the slave mode if the external master is running in continuous mode without clock interruption between two data transfers.

In full duplex master mode, it runs correctly if the SPI is working in non-continuous mode, meaning that the SPI is transferring two data, then stopping the data transmission until the two data received are read back before sending the next two data.

Conditions to see this limitation:

- Packing mode is used
- SPI master (in continuous mode) or slave (worst case)
- Full duplex or receiver mode

If the packing mode is used in reception mode, the FIFO reception threshold has to be set to 16 bit. Under those setting and conditions, when a read operation (half-word to read two data in one APB access) takes place while the FIFO level is equal to 3/4 (new data came before the two first ones are read), the 16-bit read decreases the FIFO level to 1/4. The RXNE flag is not de-asserted (clear condition on FIFO empty event) and a new request is present to read back two data although the FIFO contains only one data. Read and write pointers in the FIFO become misaligned and the data is corrupted.

The packing mode in reception has to be discarded when the conditions described above are met. It means that the reception FIFO requests that the data is read back until the FIFO content is empty. It also means that for short data frame (the worst case being the 4-bit data size), if the software or the DMA is not able to manage the high data rate when the SPI is running full speed, an Overrun condition may occur at regular intervals.

Workaround

There is no workaround.

The only way to avoid this overrun condition would be to slow down the SPI communication clock frequency in order to let time to the DMA (best case) to read back data without any FIFO full condition.

2 Revision history

Table 4. Document revision history

Date	Revision	Changes
19-Sep-2013	1	Initial release.
15-Jan-2015	2	Extended the applicability to STM32F030xC devices. Removed the Appendix: Revision code on device marking as the device marking information is now included in the product datasheet. Removed the USART limitation “Communication parameters reprogramming after ATR in Smartcard mode when SCLK is used to clock the card” Removed the I2C limitation “Wakeup frames may not wakeup the MCU mode when STOP mode entry follows I2C enabling”. Removed the I2C limitation “Wakeup frame may not wakeup from STOP if $t_{HD;STA}$ is close to HSI startup time”. Updated *Section 1.4.3: Wrong behaviors in Stop mode*. Added *Section 1.3.2: GPIOx locking mechanism not working properly for GPIOx_OTYPER register*.