life.augmented

STM32F070x6/xB
Errata sheet

STM32F070x6/xB device limitations

Silicon identification

This document applies to the part numbers of STM32F070x6/xB devices listed in *Table 1* and their silicon revisions shown in *Table 2*.

Section 1 gives a summary and *Section 2* a description of device limitations, with respect to the device datasheet and reference manual RM0360.

Table 1. Device summary

Reference	Part numbers
STM32F070x6	STM32F070C6, STM32F070F6
STM32F070xB	STM32F070CB, STM32F070RB

Table 2. Device identification(1)

Reference	Revision code marked on the device(2)
STM32F070x6	'A'
STM32F070xB	'Y', '1'

1. The REV_ID bits in the DBGMCU_IDCODE register indicate the revision code of the device (see the reference manual for details on the revision code).
2. Refer to datasheet for details on how to identify the silicon revision code on different types of package.

Contents

1 Summary of device limitations

The following table gives a quick references to all documented device limitations of STM32F070x6/xB and their status:

A = workaround available

N = no workaround available

P = partial workaround available

"-" grayed = limitation not existing / limitation fixed

Table 3. Summary of device limitations

Function	Section	Limitation	Status	
			Rev. 'A'	Rev. 'Y', '1'
System	2.1.1	*Wakeup sequence from Standby mode when using more than one wakeup source*	A	A
USART	2.2.1	*Break request can prevent the Transmission Complete flag (TC) from being set*	A	A
	2.2.2	*nRTS is active while RE or UE = 0*	A	A
	2.2.3	*Receiver timeout counter starting in case of 2 stops bit configuration*	A	A
	2.2.4	*USART4 transmission does not work on PC11*	-	A
	2.2.5	*Last byte written in TDR might not be transmitted if TE is cleared just after writing in TDR*	A	A
GPIO	2.3.1	*GPIOx locking mechanism not working properly for GPIOx_OTYPER register*	P	P
I2C	2.4.1	*Wrong data sampling when data set-up time (tSU;DAT) is shorter than one I2CCLK period*	P	P
	2.4.2	*Spurious bus error detection in master mode*	A	A
	2.4.3	*10-bit slave mode: wrong direction bit value after Read header reception*	-	A
	2.4.4	*10-bit combined with 7-bit slave mode: ADDCODE may indicate wrong slave address detection*	-	N
	2.4.5	*Wrong behavior in Stop mode*	A	A
	2.4.6	*10-bit master mode: new transfer cannot be launched if first part of the address has not been acknowledged by the slave*	A	A
SPI	2.5.1	*BSY bit may stay high when SPI is disabled*	A	A
	2.5.2	*BSY bit may stay high at the end of a data transfer in slave mode*	A	A
	2.5.3	*Wrong CRC transmitted in master mode with delayed SCK feedback*	A	A
	2.5.4	*CRC error in SPI slave mode if internal NSS changes before CRC transfer*	A	A
	2.5.5	*SPI CRC corrupted upon DMA transaction completion by another peripheral*	P	P

Table 3. Summary of device limitations (continued)

Function	Section	Limitation	Status	
			Rev. 'A'	Rev. 'Y', '1'
USB	*2.6.1*	*The USB BCD functionality limited below -20°C*	N	N
	2.6.2	*DCD (data contact detect) function not compliant*	N	N
RTC	*2.7.1*	*Spurious tamper detection when disabling the tamper channel*	N	N
	2.7.2	*A tamper event preceding the tamper detect enable not detected*	A	A
	2.7.3	*RTC calendar registers are not locked properly*	A	A
ADC	*2.8.1*	*Overrun flag not set if EOC reset coincides with new conversion end*	A	A
	2.8.2	*ADEN bit cannot be set immediately after the ADC calibration*	A	A
IWDG	*2.9.1*	*RVU, PVU and WVU flags are not reset in STOP mode*	A	A
	2.9.2	*RVU, PVU and WVU flags are not reset with low-frequency APB*	N	N

2 Description of device limitations

The following sections describe device limitations and provide workarounds if available. They are grouped by device functions.

2.1 System

2.1.1 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.

2.2 USART

2.2.1 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.

2.2.2 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.

2.2.3 Receiver timeout counter starting in case of 2 stops bit configuration

Description

In the case of 2 stop bits configuration, the receiver timeout counter starts counting from the end of the second stop bit of the last character instead of the end of the first stop bit.

Workaround

Change the RTO value in the USARTx_RTOR register by subtracting 1 bit duration.

2.2.4 USART4 transmission does not work on PC11

Description

USART4_RX does not work as output on PC11.

As a consequence, single wire half duplex mode is not supported with pin PC11.

Workaround

Use USART4_RX mapped on PA0 instead on PC11.

2.2.5 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

2.3 GPIO

2.3.1 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].

2.4 I2C

2.4.1 Wrong data sampling when data set-up time ($t_{SU;DAT}$) is shorter than one I2CCLK period

Description

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

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

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

Workaround

Increase the I2CCLK frequency to get I2CCLK period within the transmitter minimum data set-up time. Alternatively, increase transmitter's minimum data set-up time.

2.4.2 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.

2.4.3 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 I^2C 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:

- I^2C has to be configured in 10-bit addressing mode (OA1MODE is set in the I2C_OAR1 register).
- The high LSBs of the I^2C 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 I^2C 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 I^2C slave address, the DIR bit must not be used in the FW.

2.4.4 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.

2.4.5 Wrong behavior 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.

2.4.6 10-bit master mode: new transfer cannot be launched if first part of the address has not been acknowledged by the slave

Description

In master mode, the master automatically sends a STOP bit when the slave has not acknowledged a byte during the address transmission.

In 10-bit addressing mode, if the first byte of the 10-bit address (5-bit header + 2 MSBs of the address + direction bit) has not been acknowledged by the slave, the STOP bit is sent but the START bit is not cleared and the master cannot launch a new transfer.

Workaround

When the I2C is configured in 10-bit addressing master mode and the NACKF status flag is set in the I2C_ISR register while the START bit is still set in I2C_CR2 register, then proceed as follows:

1. Wait for the STOP condition detection (STOPF = 1 in I2C_ISR register).
2. Disable the I2C peripheral.
3. Wait for a minimum of 3 APB cycles.
4. Enable the I2C peripheral again.

2.5 SPI

2.5.1 BSY bit may stay high when SPI is disabled

Description

The BSY flag may remain high upon disabling the SPI while operating in:

- a master transmit mode and the TXE flag is low (data register full).
- a master receive only mode (simplex receive or half-duplex bidirectional receive phase) and an SCK strobing edge has not occurred since the transition of the RXNE flag from low to high.
- slave mode and NSS signal is removed during the communication.

Workaround

When the SPI operates in:

- a master transmit mode, disable the SPI when TXE=1 and BSY=0.
- a master receive only mode, ignore the BSY flag.
- slave mode, do not remove the NSS signal during the communication.

2.5.2 BSY bit may stay high at the end of a data transfer in slave mode

Description

In slave mode, The 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 the SPI 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 the BSY bit before entering low-power mode or modifying the SPI configuration (e.g. direction of the bidirectional mode).

Workaround

- When the SPI interface is in receive mode, the end of a transaction with the master can be detected by the corresponding RXNE event when this flag is set after the last bit of that transaction is sampled and the received data are stored.
- When the following sequence is used, the synchronization issue does not occur. The BSY bit works correctly and can be used to recognize the end of any transmission transaction (including when RXNE is not raised in bidirectional mode):
  a) Write the last data into data register.
  b) Poll the TXE flag till it becomes high to make sure the data transfer has started.
  c) Disable the SPI interface by clearing the SPE bit while the last data transfer is on going.
  d) Poll the BSY bit till it becomes low.

Note: The second workaround can be used only when the CPU is fast enough to disable the SPI interface after a TXE event is detected while the data frame transfer is ongoing. It cannot be implemented when the ratio between CPU and SPI clock is low and the data frame is particularly short. At this specific case, the timeout can be measured from the TXE event instead by calculating a fixed number of CPU clock cycles corresponding to the time necessary to complete the data frame transaction.

2.5.3 Wrong CRC transmitted in master mode with delayed SCK feedback

Description

In transmit transaction of the SPI/I^{2}S interface in SPI master mode with CRC enabled, the CRC data transmission may be corrupted if the delay of an internal feedback signal derived from the SCK output (further feedback clock) is greater than one APB clock period. While data and CRC bit shifting and transfer is based on an internal clock, the CRC progressive calculation uses the feedback clock. If the delay of the feedback clock is greater than one APB period, the transmitted CRC value may get wrong.

The main factors contributing to the delay increase are low V_{DD} level, high temperature, high SCK pin capacitive load and low SCK IO output speed. The SPI communication speed has no impact.

Workaround

Set the application such as to speed up the SCK edges and / or slow down the APB clock, through:

- configuring the SCK output GPIO so as to reach lower output impedance
- minimizing the capacitive load on the SCK output line
- configuring the APB clock speed

2.5.4 CRC error in SPI slave mode if internal NSS changes before CRC transfer

Description

When the device is configured as SPI slave, the transition of the internal NSS after the CRCNEXT flag is set may result in wrong CRC value computed by the device and, as a consequence, a CRC error. As a consequence, the NSS pulse mode cannot be used along with the CRC function.

Workaround

Prevent the internal NSS signal from changing in the critical period, by configuring the device to software NSS control if the SPI master pulses the NSS (for example in NSS pulse mode).

2.5.5 SPI CRC corrupted upon DMA transaction completion by another peripheral

Description

When the following conditions are all met:

- CRC function for the SPI is enabled,
- SPI transaction managed by software (as opposed to DMA) is ongoing and CRCNEXT flag set,
- another peripheral using the same DMA channel on which the SPI is mapped completes a DMA transfer,

the CRCNEXT bit is unexpectedly cleared and the SPI CRC calculation may be corrupted, setting the CRC error flag.

Workaround

If possible, do not use the DMA channel, on which the SPI is mapped, by any other peripheral.

If possible, remap SPI2 to a DMA channel not used by another peripheral.

2.6 USB

2.6.1 The USB BCD functionality limited below -20°C

Description

Primary and secondary detection can return an incorrectly detected port type.

This limitation may be observed on a small number of devices when the temperature is below -20°C.

Workaround

None.

2.6.2 DCD (data contact detect) function not compliant

Description

The DCD function on the device is not compliant with the "USB Battery Charging 1.2 Compliance Plan rev 1.0" specification.

Workaround

Do not use the DCD function. Instead, upon attaching a USB device, wait for at least "TDCD_TIMEOUT" amount of time before starting Primary Detection. This is in line with the "Battery Charging Specification rev1.2" recommendation for portable devices that do not support the DCD function.

2.7 RTC

2.7.1 Spurious tamper detection when disabling the tamper channel

Description

If the tamper detection is configured for detecting on falling-edge event (TAMPFLT[1:0]=00 and TAMPxTRG=1) and if the tamper event detection is disabled when the tamper pin is at high level, a false detection of a tamper event occurs.

Workaround

The false detection of tamper event cannot be avoided.

2.7.2 A tamper event preceding the tamper detect enable not detected

Description

When the tamper detect is enabled, set in edge detection mode (TAMPFLT[1:0]=00), and

- set to active rising edge (TAMPxTRG=0): if the tamper input is already high (tamper event already occurred) at the moment of enabling the tamper detection, the tamper event may not be detected. The probability of detection increases with the APB frequency.
- set to active falling edge (TAMPxTRG=1): if the tamper input is already low (tamper event already occurred) at the moment of enabling the tamper detection, the tamper event is not detected.

Workaround

The I/O state should be checked by software in the GPIO registers after enabling the tamper detection, in order to ensure that no active edge occurred before enabling the tamper event detection.

2.7.3 RTC calendar registers are not locked properly

Description

When reading the calendar registers with BYPSHAD=0, the RTC_TR and RTC_DR registers may not be locked after the read of RTC_SSR register. This happens if the read of RTC_SSR is initiated one APB clock period before the shadow registers are updated. This can result in a non-consistency of the 3 registers. Similarly, RTC_DR register can be updated after the read of the RTC_TR register instead of being locked.

Workaround

1. Use BYPSHAD = 1 mode (Bypass shadow registers), or
2. In case BYPSHAD = 0: read SSR again after reading SSR/TR/DR to confirm that SSR is still the same, otherwise read the values again.

2.8 ADC

2.8.1 Overrun flag not set if EOC reset coincides with new conversion end

Description

If the EOC flag is cleared by ADC_DR register read operation or by software during the same APB cycle in which the data from a new conversion are written in the ADC_DR register, the overrun event duly occurs (which results in the loss of either current or new data) but the overrun flag (OVR) may stay low.

Workaround

Clear the EOC flag through ADC_DR register read operation or by software within less than one ADC conversion cycle period from the last conversion cycle end, so as to avoid the coincidence with the new conversion cycle end.

2.8.2 ADEN bit cannot be set immediately after the ADC calibration

Description

At the end of the ADC calibration, an internal reset of ADEN bit occurs four ADC clock cycles after the ADCAL bit is cleared by hardware. As a consequence, if the ADEN bit is set within those four ADC clock cycles, it is reset shortly after by the calibration logic and the ADC remains disabled.

Workaround

1. Keep setting the ADEN bit until the ADRDY flag goes high.
2. After the ADCAL is cleared, wait for a minimum of four ADC clock cycles before setting the ADEN bit.

2.9 IWDG

2.9.1 RVU, PVU and WVU flags are not reset in STOP mode

Description

The RVU,PVU and WVU 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,PVU and WVU 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, PVU and WVU flags to be reset. However, since the RVU/PVU/WPU bit is not reset after exiting the 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, PVU and WVU flags of the IWDG_SR register are reset, before entering the Stop mode.

2.9.2 RVU, PVU and WVU flags are not reset with low-frequency APB

Description

The RVU, PVU and WVU 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 APB clock frequency is two times slower than the IWDG clock frequency, the RVU, PVU and WVU flags will never be reset by hardware.

Workaround

None

3 Revision history

Table 4. Document revision history

Date	Revision	Changes
15-Jan-2015	1	Initial release.
12-Oct-2016	2	Added: *USART*: – *Section 2.2.1: Break request can prevent the Transmission Complete flag (TC) from being set* – *Section 2.2.2: nRTS is active while RE or UE = 0* – *Section 2.2.3: Receiver timeout counter starting in case of 2 stops bit configuration* *I2C*: – *Section 2.4.1: Wrong data sampling when data set-up time (tSU;DAT) is shorter than one I2CCLK period* – *Section 2.4.2: Spurious bus error detection in master mode* – *Section 2.4.6: 10-bit master mode: new transfer cannot be launched if first part of the address has not been acknowledged by the slave* *SPI*: – *Section 2.5.1: BSY bit may stay high when SPI is disabled* – *Section 2.5.2: BSY bit may stay high at the end of a data transfer in slave mode* – *Section 2.5.3: Wrong CRC transmitted in master mode with delayed SCK feedback* – *Section 2.5.4: CRC error in SPI slave mode if internal NSS changes before CRC transfer* – *Section 2.5.5: SPI CRC corrupted upon DMA transaction completion by another peripheral* *USB*: – *Section 2.6.2: DCD (data contact detect) function not compliant* *RTC*: – *Section 2.7.1: Spurious tamper detection when disabling the tamper channel* – *Section 2.7.2: A tamper event preceding the tamper detect enable not detected* – *Section 2.7.3: RTC calendar registers are not locked properly* *ADC*: – *Section 2.8.1: Overrun flag not set if EOC reset coincides with new conversion end* – *Section 2.8.2: ADEN bit cannot be set immediately after the ADC calibration*

Table 4. Document revision history (continued)

Date	Revision	Changes
12-Oct-2016	2	*IWDG*: – *Section 2.9.1: RVU, PVU and WVU flags are not reset in STOP mode* – *Section 2.9.2: RVU, PVU and WVU flags are not reset with low-frequency APB* Modified: – Document structure – Cover page and *Table 3* organization