STM32F078CB/RB/VB Errata sheet

STM32F078CB/RB/VB device limitations

Applicability

This document applies to the STM32F078CB/RB/VB devices and their variants shown in *Table 1*.

Section 1 gives a summary and *Section 2* a description of device limitations, with respect to the device datasheet and reference manual RM0091.

Table 1. Device variants

Reference	Silicon revision codes	
	Device marking(1)	REV_ID(2)
STM32F078CB/RB/VB	Y or 1	0x2001

1. Refer to the device data sheet for how to identify this code on different types of package.
2. REV_ID[15:0] bit field of DBGMCU_IDCODE register. Refer to the reference manual.

Contents

1 Summary of device limitations 4

2 Description of device limitations 6

2.1 GPIO 6

2.1.1 GPIOx locking mechanism not working properly for GPIOx_OTYPER register 6

2.2 ADC 6

2.2.1 Overrun flag not set if EOC reset coincides with new conversion end . . 6

2.2.2 ADEN bit cannot be set immediately after the ADC calibration 6

2.3 COMP 7

2.3.1 Long V_{REFINT} scaler startup time after power on 7

2.4 TSC 7

2.4.1 Inhibited acquisition in short transfer phase configuration 7

2.5 IWDG 8

2.5.1 RVU, PVU and WVU flags are not reset in STOP mode 8

2.5.2 RVU, PVU and WVU flags are not reset with low-frequency APB 8

2.6 RTC 8

2.6.1 Spurious tamper detection when disabling the tamper channel 8

2.6.2 A tamper event preceding the tamper detect enable not detected 9

2.6.3 RTC calendar registers are not locked properly 9

2.7 I2C 9

2.7.1 Wrong data sampling when data setup time ($t_{SU;DAT}$) is shorter than one I2C kernel clock period 9

2.7.2 Spurious bus error detection in master mode 10

2.7.3 10-bit slave mode: wrong direction bit value upon Read header receipt 10

2.7.4 10-bit combined with 7-bit slave mode: ADDCODE may indicate wrong slave address detection 11

2.7.5 Wakeup frames may not wake up the MCU when Stop mode entry follows I2C enabling 12

2.7.6 Wakeup frame may not wake up the MCU from Stop mode if $t_{HD;STA}$ is close to I2C kernel clock startup time 12

2.7.7 Wrong behavior in Stop mode when wakeup from Stop mode is disabled in I2C 13

2.7.8 10-bit master mode: new transfer cannot be launched if first part of the address is not acknowledged by the slave 13

2.7.9 Last-received byte loss in reload mode 14
2.8 USART 14
2.8.1 Non-compliant sampling for NACK signal from smartcard 14
2.8.2 Break request preventing TC flag from being set 15
2.8.3 RTS is active while RE = 0 or UE = 0 15
2.8.4 Receiver timeout counter wrong start in two-stop-bit configuration 15
2.8.5 USART4 transmission does not work on PC11 15
2.8.6 Last byte written in TDR might not be transmitted if TE is cleared just after writing in TDR 16
2.9 SPI/I2S 16
2.9.1 BSY bit may stay high when SPI is disabled 16
2.9.2 BSY bit may stay high at the end of data transfer in slave mode 16
2.9.3 CRC error in SPI slave mode if internal NSS changes before CRC transfer 17
2.9.4 SPI CRC corruption upon DMA transaction completion by another peripheral 18
2.9.5 In I^2S slave mode, enabling I2S while WS is active causes desynchronization 18
2.10 USB 18
2.10.1 The USB BCD functionality limited below -20°C 18
2.10.2 DCD (data contact detect) function not compliant 19
2.11 HDMI-CEC 19
2.11.1 Transmission blocked when transmitted start bit is corrupted 19
2.11.2 Missed CEC messages in normal receiving mode 20

3 Revision history 21

1 Summary of device limitations

The following table gives a quick references to all documented device limitations of STM32F078CB/RB/VB and their status:

A = workaround available

N = no workaround available

P = partial workaround available

Applicability of a workaround may depend on specific conditions of target application. Adoption of a workaround may cause restrictions to target application. Workaround for a limitation is deemed partial if it only reduces the rate of occurrence and/or consequences of the limitation, or if it is fully effective for only a subset of instances on the device or in only a subset of operating modes, of the function concerned.

Table 2. Summary of device limitations

Function	Section	Limitation	Status
			Rev. Y or 1
GPIO	*2.1.1*	*GPIOx locking mechanism not working properly for GPIOx_OTYPER register*	P
ADC	*2.2.1*	*Overrun flag not set if EOC reset coincides with new conversion end*	A
	2.2.2	*ADEN bit cannot be set immediately after the ADC calibration*	A
COMP	*2.3.1*	*Long V_{REFINT} scaler startup time after power on*	N
TSC	*2.4.1*	*Inhibited acquisition in short transfer phase configuration*	P
IWDG	*2.5.1*	*RVU, PVU and WVU flags are not reset in STOP mode*	A
	2.5.2	*RVU, PVU and WVU flags are not reset with low-frequency APB*	N
RTC	*2.6.1*	*Spurious tamper detection when disabling the tamper channel*	P
	2.6.2	*A tamper event preceding the tamper detect enable not detected*	A
	2.6.3	*RTC calendar registers are not locked properly*	A
I2C	*2.7.1*	*Wrong data sampling when data setup time ($t_{SU;DAT}$) is shorter than one I2C kernel clock period*	P
	2.7.2	*Spurious bus error detection in master mode*	A
	2.7.3	*10-bit slave mode: wrong direction bit value upon Read header receipt*	A
	2.7.4	*10-bit combined with 7-bit slave mode: ADDCODE may indicate wrong slave address detection*	N
	2.7.5	*Wakeup frames may not wake up the MCU when Stop mode entry follows I2C enabling*	A
	2.7.6	*Wakeup frame may not wake up the MCU from Stop mode if $t_{HD;STA}$ is close to I2C kernel clock startup time*	P

Table 2. Summary of device limitations (continued)

Function	Section	Limitation	Status Rev. Y or 1
I2C	2.7.7	Wrong behavior in Stop mode when wakeup from Stop mode is disabled in I2C	A
	2.7.8	10-bit master mode: new transfer cannot be launched if first part of the address is not acknowledged by the slave	A
	2.7.9	Last-received byte loss in reload mode	A
USART	2.8.1	Non-compliant sampling for NACK signal from smartcard	N
	2.8.2	Break request preventing TC flag from being set	A
	2.8.3	RTS is active while RE = 0 or UE = 0	A
	2.8.4	Receiver timeout counter wrong start in two-stop-bit configuration	A
	2.8.5	USART4 transmission does not work on PC11	A
	2.8.6	Last byte written in TDR might not be transmitted if TE is cleared just after writing in TDR	A
SPI/I2S	2.9.1	BSY bit may stay high when SPI is disabled	A
	2.9.2	BSY bit may stay high at the end of data transfer in slave mode	A
	2.9.3	CRC error in SPI slave mode if internal NSS changes before CRC transfer	A
	2.9.4	SPI CRC corruption upon DMA transaction completion by another peripheral	P
	2.9.5	In I^2S slave mode, enabling I2S while WS is active causes desynchronization	P
USB	2.10.1	The USB BCD functionality limited below -20°C	N
	2.10.2	DCD (data contact detect) function not compliant	N
HDMI-CEC	2.11.1	Transmission blocked when transmitted start bit is corrupted	P
	2.11.2	Missed CEC messages in normal receiving mode	A

2 Description of device limitations

The following sections describe limitations of the applicable devices with Arm®(a) core and provide workarounds if available. They are grouped by device functions.

2.1 GPIO

2.1.1 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].

2.2 ADC

2.2.1 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.

2.2.2 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

a. Arm is a registered trademark of Arm Limited (or its subsidiaries) in the US and/or elsewhere.

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.

2.3 COMP

2.3.1 Long V_{REFINT} scaler startup time after power on

Description

The V_{REFINT} scaler is an embedded voltage follower providing the V_{REFINT} or its fractions (1/2, 1/4 or 3/4) to the comparator input.

The maximum V_{REFINT} scaler startup time t_{S_SC}(max), specified to 0.2 ms, is not respected for the first activation of the V_{REFINT} scaler after powering on the device. In worst-case conditions, it can be as much as 1 s. The startup time depends mainly on the voltage and temperature. See the device datasheet for more details.

For correct operation of the V_{REFINT} scaler, the comparator analog supply voltage, V_{DDA}, must not be below 2 V.

Workaround

None.

2.4 TSC

2.4.1 Inhibited acquisition in short transfer phase configuration

Description

The GPIO input buffer is masked outside the transfer window time and then sampled twice before being checked for the acquisition. This check is performed on the last touch sensing clock cycle of the charge transfer phase. When the charge transfer duration is less than three clock cycles, the acquisition is inhibited.

Workaround

Do not use the following TSC control register configurations:

- PGPSC[2:0] bits set to 000 and CTPL[3:0] bits set to 0000 or 0001 in TSC_CR register
- PGPSC[2:0] bits set to 001 and bits CTPL[3:0] set to 0000 in TSC_CR register

2.5 IWDG

2.5.1 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.

2.5.2 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

2.6 RTC

2.6.1 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, which may result in the erasure of backup registers.

Workaround

The false detection of tamper event cannot be avoided. The erasure of the backup registerscan be avoided by setting the TAMPxNOERASE bit before clearing the TAMPxE bit, in two separate RTC_TAMPCR write accesses.

2.6.2 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 and before writing sensitive values in the backup registers, in order to ensure that no active edge occurred before enabling the tamper event detection.

2.6.3 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.

2.7 I2C

2.7.1 Wrong data sampling when data setup time ($t_{SU;DAT}$) is shorter than one I2C kernel clock period

Description

The I^2C-bus specification and user manual specify a minimum data setup time ($t_{SU;DAT}$) as:

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

The MCU does not correctly sample the I^2C-bus SDA line when $t_{SU;DAT}$ is smaller than one I2C kernel clock (I^2C-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 I2C kernel clock frequency to get I2C kernel clock period within the transmitter minimum data setup time. Alternatively, increase transmitter's minimum data setup time. If the transmitter setup time minimum value corresponds to the minimum value provided in the I^2C-bus standard, the minimum I2CCLK frequencies are as follows:

- In Standard mode, if the transmitter minimum setup time is 250 ns, the I2CCLK frequency must be at least 4 MHz.
- In Fast mode, if the transmitter minimum setup time is 100 ns, the I2CCLK frequency must be at least 10 MHz.
- In Fast-mode Plus, if the transmitter minimum setup time is 50 ns, the I2CCLK frequency must be at least 20 MHz.

2.7.2 Spurious bus error detection in master mode

Description

In master mode, a bus error can be detected spuriously, with the consequence of setting the BERR flag of the I2C_SR register and generating bus error interrupt if such interrupt is enabled. Detection of bus error has no effect on the I^2C-bus transfer in master mode and any such transfer continues 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 ongoing transfer can be handled normally.

2.7.3 10-bit slave mode: wrong direction bit value upon Read header receipt

Description

Under specific conditions, the transfer direction bit DIR (bit 16 of status register I2C_ISR) remains low upon receipt of 10-bit addressing Read header, while normally it should be set high. Nevertheless, I2C operates correctly in slave transmission mode, and data can be sent using the TXIS flag.

The failure described occurs when the following conditions are all met:

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

Workaround

Avoid using the following 10-bit slave addresses:

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

If the use of one of these slave addresses cannot be avoided, do not use the DIR bit in the firmware.

2.7.4 10-bit combined with 7-bit slave mode: ADDCODE may indicate wrong slave address detection

Description

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

The failure occurs when the following conditions are all met:

- A 10-bit slave address OA1 is enabled (OA1EN = 1 and OA1MODE = 1)
- A 7-bit slave address OA2 is enabled (OA2EN = 1) and it matches the non-masked bits of OA1[7:1], that is, one of the following configurations 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] = 0000000
  - ALERTEN = 1 and OA1[7:1] = 0001100
  - SMBDEN = 1 and OA1[7:1] = 1100001
  - SMBHEN = 1 and OA1[7:1] = 0001000
- The MCU is addressed by a bus master with its 10-bit slave address OA1.

Upon the address receipt, 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 OA1 [7:1] part of the 10-bit slave address must be different than the 7-bit slave address.

2.7.5 Wakeup frames may not wake up the MCU when Stop mode entry follows I2C enabling

Description

If I2C is enabled (PE = 1) and wakeup from Stop mode is enabled in I2C (WUPEN = 1) while a transfer occurs on the I^2C-bus and Stop mode is entered during the same transfer while SCL = 0, I2C is not able to detect the following START condition. As a consequence, the MCU does not wake up from Stop mode when it is addressed on the I^2C-bus and it does not acknowledge the receipt of the address.

Workaround

After enabling I2C (by setting PE to 1), do not enter Stop mode until any I^2C-bus transaction in progress ends.

2.7.6 Wakeup frame may not wake up the MCU from Stop mode if $t_{HD;STA}$ is close to I2C kernel clock startup time

Description

Under specific conditions and if the START condition hold time $t_{HD;STA}$ is very close to the startup time of the internal oscillator selected for I2C kernel clock, I2C is not able to detect the address match and, as a consequence, to wake up the MCU from Stop mode.

The failure described occurs when one of the following conditions is met:

1. Timeout detection is enabled (TIMOUTEN = 1 or TEXTEN = 1) and the frame before the wakeup frame is finished abnormally due to I2C timeout detection (TIMOUT = 1).
2. Slave arbitration is lost during the frame preceding the wakeup frame (ARLO = 1).
3. The MCU enters Stop mode while another slave is addressed, after the address phase and before STOP condition (BUSY = 1).
4. The MCU is in Stop mode and another slave is addressed before the MCU itself is addressed.

Note: The conditions 2, 3 and 4 can only occur in a multi-slave network.

In Stop mode, the internal oscillator selected for I2C kernel clock is switched on by I2C when START condition is detected. The I2C kernel clock is then used to receive the address. The internal oscillator is switched off upon the address receipt if the address received does not match the own slave address. If one of the conditions listed is met and if the SCL falling edge following the START condition occurs within the first cycle of the I2C kernel clock, the address is received incorrectly and the address match wakeup interrupt is not generated.

Workaround

None at MCU level.

Upon non-acknowledge by the MCU of a wakeup frame, the I^2C-bus master with programmable START condition hold time can set that hold time such that it exceeds one MCU internal oscillator period, then resend the wakeup frame.

2.7.7 Wrong behavior in Stop mode when wakeup from Stop mode is disabled in I2C

Description

With the MCU operating as slave or as master in multi-master topology, when wakeup from Stop mode is disabled in I2C (WUPEN = 0) and the MCU enters Stop mode while a transfer is ongoing on the bus, the following may occur:

1. BUSY flag is 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 SCL line is pulled low by I2C and the transfer stalled as long as the MCU remains in Stop mode.
   The occurrence of such condition depends on the timing configuration, peripheral clock frequency, and I^2C-bus frequency.

Workaround

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

2.7.8 10-bit master mode: new transfer cannot be launched if first part of the address is not acknowledged by the slave

Description

An I^2C-bus master generates STOP condition upon non-acknowledge of I^2C address that it sends. This applies to 7-bit address as well as to each byte of 10-bit address.

When the MCU set as I^2C-bus master transmits a 10-bit address of which the first byte (5-bit header + 2 MSBs of the address + direction bit) is not acknowledged, the MCU duly generates STOP condition but it then cannot start any new I^2C-bus transfer. In this spurious state, the NACKF flag of the I2C_ISR register and the START bit of the I2C_CR2 register are both set, while the START bit should normally be cleared.

Workaround

In 10-bit-address master mode, if both NACKF flag and START bit get simultaneously set, 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 three APB cycles.
4. Enable the I2C peripheral again.

2.7.9 Last-received byte loss in reload mode

Description

If in master receiver mode or slave receive mode with SBC = 1 the following conditions are all met:

- I^2C-bus stretching is enabled (NOSTRETCH = 0)
- RELOAD bit of the I2C_CR2 register is set
- NBYTES bitfield of the I2C_CR2 register is set to N greater than 1
- byte N is received on the I^2C-bus, raising the TCR flag
- N - 1 byte is not yet read out from the data register at the instant TCR is raised,

then the SCL line is pulled low (I^2C-bus clock stretching) and the transfer of the byte N from the shift register to the data register inhibited until the byte N-1 is read and NBYTES bitfield reloaded with a new value, the latter of which also clears the TCR flag. As a consequence, the software cannot get the byte N and use its content before setting the new value into the NBYTES field.

For I2C instances with independent clock, the last-received data is definitively lost (never transferred from the shift register to the data register) if the data N - 1 is read within four APB clock cycles preceding the receipt of the last data bit of byte N and thus the TCR flag raising. Refer to the product reference manual or datasheet for the I2C implementation table.

Workaround

- In slave mode with SBC = 1, use the reload mode with NBYTES = 1.
- In master receiver mode, if the number of bytes to transfer is greater than 255 bytes, do not use the reload mode. Instead, split the transfer into sections not exceeding 255 bytes and separate them with repeated START conditions.
- Make sure, for example through the use of DMA, that the byte N - 1 is always read before the TCR flag is raised. Specifically for I2C instances with independent clock, make sure that it is always read earlier than four APB clock cycles before the receipt of the last data bit of byte N and thus the TCR flag raising.

The last workaround in the list must be evaluated carefully for each application as the timing depends on factors such as the bus speed, interrupt management, software processing latencies, and DMA channel priority.

2.8 USART

2.8.1 Non-compliant sampling for NACK signal from smartcard

Description

According to ISO/IEC 7816-3 standard, when a character parity error is detected, the receiver must assert a NACK signal, by pulling the transmit line low for one ETU period, at 10.3 to 10.7 ETU after the character START bit falling edge. The transmitter is expected to sample the line for NACK (for low level) from 10.8 to 11.2 ETU after the character START bit falling edge.

Instead, the USART peripheral in Smartcard mode samples the transmit line for NACK from 10.3 to 10.7 ETU after the character START bit falling edge. This is unlikely to cause issues with receivers (smartcards) that respect the ISO/IEC 7816-3 standard. However, it may cause issues with respect to certification.

Workaround

None.

2.8.2 Break request preventing TC flag from being set

Description

After the end of transmission of data (D1), the transmission complete (TC) flag is not set when the following condition is met:

- CTS hardware flow control is enabled
- D1 transmission is in progress
- a break transfer is requested before the end of D1 transfer
- nCTS is de-asserted before the end of D1 transfer

As a consequence, an application relying on the TC flag fails to detect the end of data transfer.

Workaround

In the application, only allow break request after the TC flag is set.

2.8.3 RTS is active while RE = 0 or UE = 0

Description

The RTS 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), that is, not ready to receive data.

Workaround

Upon setting the UE and RE bits, configure the I/O used for RTS into alternate function.

2.8.4 Receiver timeout counter wrong start in two-stop-bit configuration

Description

In two-stop-bit configuration, the receiver timeout counter starts counting from the end of the second stop bit of the last character instead of starting from the end of the first stop bit.

Workaround

Subtract one bit duration from the value in the RTO bitfield of the USARTx_RTOR register.

2.8.5 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.

2.8.6 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

2.9 SPI/I2S

2.9.1 BSY bit may stay high when SPI is disabled

Description

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

- master transmit mode and the TXE flag is low (data register full).
- 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:

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

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

Description

BSY flag may sporadically remain high at the end of a data transfer in slave mode. This occurs upon coincidence of internal CPU clock and external SCK clock provided by master.

In such an event, if the software only relies on BSY flag to detect the end of SPI slave data transaction (for example to enter low-power mode or to change data line direction in half-duplex bidirectional mode), the detection fails.

As a conclusion, the BSY flag is unreliable for detecting the end of data transactions.

Workaround

Depending on SPI operating mode, use the following means for detecting the end of transaction:

- When NSS hardware management is applied and NSS signal is provided by master, use NSS flag.
- In SPI receiving mode, use the corresponding RXNE event flag.
- In SPI transmit-only mode, use the BSY flag in conjunction with a timeout expiry event. Set the timeout such as to exceed the expected duration of the last data frame and start it upon TXE event that occurs with the second bit of the last data frame. The end of the transaction corresponds to either the BSY flag becoming low or the timeout expiry, whichever happens first.

Prefer one of the first two measures to the third as they are simpler and less constraining.

Alternatively, apply the following sequence to ensure reliable operation of the BSY flag in SPI transmit mode:

1. Write last data to data register
2. Poll the TXE flag until it becomes high, which occurs with the second bit of the data frame transfer
3. Disable SPI by clearing the SPE bit mandatorily before the end of the frame transfer
4. Poll the BSY bit until it becomes low, which signals the end of transfer

Note: The alternative method can only be used with relatively fast CPU speeds versus relatively slow SPI clocks or/and long last data frames. The faster is the software execution, the shorter can be the duration of the last data frame.

2.9.3 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 signal after the CRCNEXT flag is set may result in wrong CRC value computed by the device and, as a consequence, in 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).

2.9.4 SPI CRC corruption 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 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

Ensure that the DMA channel on which the SPI is mapped is not concurrently in use by another peripheral.

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

2.9.5 In I^2S slave mode, enabling I2S while WS is active causes desynchronization

Description

In I^2S slave mode, the WS signal level is used to start the communication. If the I2S peripheral is enabled while the WS line is active (low for I^2S protocol, high for LSB- or MSB-justified mode), and if the master is already sending the clock, the I2S peripheral (slave) starts communicating data from the instant of its enable, which causes desynchronization between the master and the slave throughout the whole communication.

Workaround

Enable I2S peripheral while the WS line is at:

- high level, for I^2S protocol.
- low level, for LSB- or MSB-justified mode.

2.10 USB

2.10.1 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.

2.10.2 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 "$T_{DCD\_TIMEOUT}$" 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.

2.11 HDMI-CEC

2.11.1 Transmission blocked when transmitted start bit is corrupted

Description

When the HDMI-CEC communication start bit transmitted by the device is corrupted by another device on the CEC line, the CEC transmission is stalled.

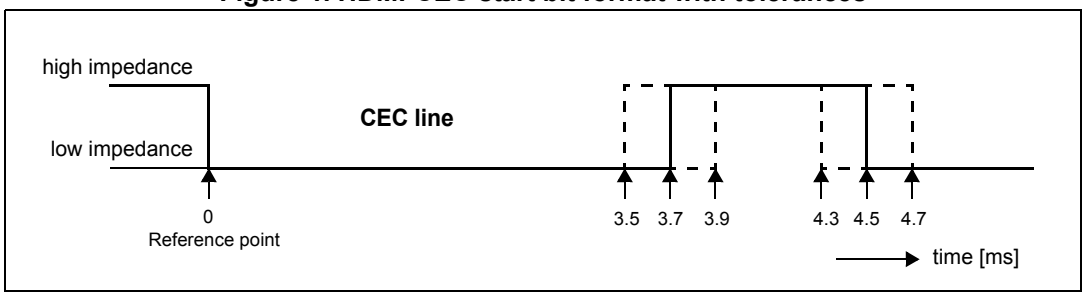
This failure is unlikely to happen as the CEC start bit corruption by another device can only occur if that device does not respect the CEC communication protocol.

The start bit timing standard tolerances are shown in *Figure 1*. The start bit is initiated by the device by driving the CEC line low (reference point). After 3.7 ms, the device releases the CEC line and starts checking its level. The following conditions must be met for the start bit to be valid:

- the CEC line goes high no later than 3.9 ms (4.05 ms with extended tolerance) from the reference point
- a falling edge on the CEC line does not occur earlier than 4.3 ms (4.15 ms with extended tolerance) from the reference point

If one of these conditions is not met, the transmission is aborted and never automatically retried. No error flag is set and the TXSOM (Tx Start Of Message) bit is not cleared.

Figure 1. HDMI-CEC start bit format with tolerances



Workaround

The only way to detect this error is for the application software to start a timeout when setting the TXSOM bit, restart it upon ARBLST or any RX event (as the transmission can be

delayed by interleaved reception), and stop it upon TXBR (proof that the start bit was transmitted successfully) or TXEND event, or upon any TX error (which clears TXSOM). If the timeout expires (because none of those events occurred), the application software must restart the HDMI-CEC peripheral and retransmit the message.

2.11.2 Missed CEC messages in normal receiving mode

Description

In normal receiving mode, any CEC message with destination address different from the own address should normally be ignored and have no effect to the CEC peripheral. Instead, such a message is unduly written into the reception buffer and sets the CEC peripheral to a state in which any subsequent message with the destination address equal to the own address is rejected (NACK), although it sets RXOVR flag (because the reception buffer is considered full) and generates (if enabled) an interrupt. This failure can only occur in a multi-node CEC framework where messages with addresses other than own address can appear on the CEC line.

The listen mode operates correctly.

Workaround

Use listen mode (set LSTEN bit) instead of normal receiving mode. Discard messages to single listeners with destination address different from the own address of the HDMI-CEC peripheral.

3 Revision history

Table 3. Document revision history

Date	Revision	Changes
12-Jun-2014	1	Initial release.
12-Oct-2016	2	Added: *USART*: – *Section 2.10.1: Start bit detected too soon when sampling for NACK signal from the smartcard* – *Section 2.10.2: Break request can prevent the Transmission Complete flag (TC) from being set* – *Section 2.10.3: RTS is active while RE or UE = 0* – *Section 2.10.4: Receiver timeout counter starting in case of 2 stops bit configuration* *I2C*: – *Section 2.9.2: Spurious bus error detection in master mode* – *Section 2.9.9: 10-bit master mode: new transfer cannot be launched if first part of the address is not acknowledged by the slave* *SPI*: – *Section 2.8.1: BSY bit may stay high when SPI is disabled* – *Section 2.8.2: BSY bit may stay high at the end of a data transfer in slave mode* – *Section 2.8.3: Wrong CRC transmitted in master mode with delayed SCK feedback* – *Section 2.8.4: CRC error in SPI slave mode if internal NSS changes before CRC transfer* – *Section 2.4.5: SPI CRC corrupted upon DMA transaction completion by another peripheral* *USB*: – *Section 2.13.3: DCD (data contact detect) function not compliant* *RTC*: – *Section 2.8.1: Spurious tamper detection when disabling the tamper channel* – *Section 2.8.3: A tamper event preceding the tamper detect enable not detected* *Section 2.8.5: RTC calendar registers are not locked properly* *ADC*: – *Section 2.4.1: Overrun flag not set if EOC reset coincides with new conversion end* – *Section 2.4.2: ADEN bit cannot be set immediately after the ADC calibration*

Table 3. Document revision history (continued)

Date	Revision	Changes
12-Oct-2016	2	*HDMI-CEC*: – *Section 2.14.1: Transmission blocked when transmitted start bit is corrupted* *TSC*: – *Section 2.6.1: Inhibited acquisition in short transfer phase configuration* *IWDG*: – *Section 2.7.1: RVU, PVU and WVU flags are not reset in STOP mode* – *Section 2.7.2: RVU, PVU and WVU flags are not reset with low-frequency APB* Modified: – Document structure – Cover page and *Table 2* organization Removed: – *GPIO*: *Extra consumption on GPIOs PC0..5 on 48/49 pin packages* (This limitation does not exist on the product. It was kept in the previous revisions of the document for historical reasons) – Appendix A (package marking drawings are now available in the data sheet)
04-May-2018	3	**Added**: – *Section 2.7.9: Last-received byte loss in reload mode* – REV_ID bitfield information on the cover page – information on workaround qualifiers in *Section 1: Summary of device limitations* – *Section 2.11.2: Missed CEC messages in normal receiving mode* **Modified**: – order of functions and their names - alignment with the reference manual – minor modifications in titles and/or text of existing limitation descriptors in *I2C*, *SPI/I2S* and *USART* sections – workaround of the limitation in *Section 2.9.5: In I^2S slave mode, enabling I2S while WS is active causes desynchronization* re-qualified to “P” – workaround description in *Section 2.7.1: Wrong data sampling when data setup time ($t_{SU;DAT}$) is shorter than one I2C kernel clock period* – document ID in the footer of all pages to ES0262 – renaming of introductory section on the cover page **Removed**: – limitation “Wrong CRC transmitted in master mode with delay on SCK feedback” in *SPI/I2S* section, kept in previous versions for historical reasons.

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.

© 2018 STMicroelectronics – All rights reserved