GigaDevice Semiconductor Inc.

Device limitations of GD32L235

Errata Sheet

Revision 1.0

(May. 2026)

Table of Contents

List of Figures

List of Tables

1. Introduction

This document applies to GD32L235 product series, as shown in ***Table 1-1. Applicable products***. It offers technical guidance for using GD32MCU and provides workaround to current device limitations.

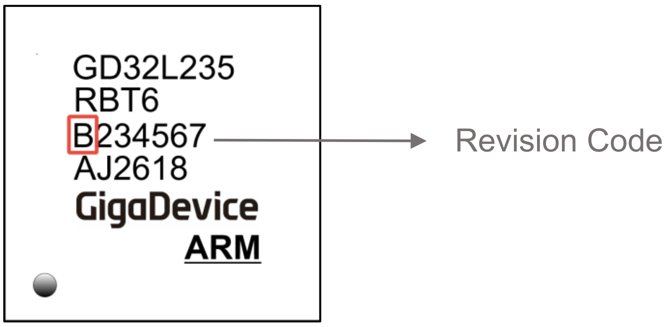
Table 1-1. Applicable products

Type	Part Numbers
MCU	GD32L235xx series

1.1. Revision identification

The device revision can be identified according to the mark on the top of the package. The 1st code on Line 3 of the mark is the product revision code, as shown in ***Figure 1-1. Device revision code of GD32L235***.

Figure 1-1. Device revision code of GD32L235



1.2. Summary of device limitations

The device limitations of GD32L235 are shown in ***Table 1-2. Device limitations***, please refer to Section 2 for more details.

Table 1-2. Device limitations

Module	Limitations	Workaround
		Rev. Code B
PMU	*FWDGTRSTF flag cannot be set in Deep-sleep mode*	Y
RCU	*The LXTALSTB bit cannot be cleared by disabling LXTAL when LXTAL stops unexpectedly*	Y
USART	*When USART is woken up from mute mode by an idle frame, it will not be woken up when it enters mute mode again*	Y
	When waking up the USART from mute mode via an idle frame, IDLEF is set	Y
	The high baud rate of USART will cause data loss when using	Y

Module	Limitations	Workaround Rev. Code B
	hardware flow control mode	
	In smartcard mode during data reception, a parity error detected during TX transmission is considered a retransmission, and FERR and RBNE cannot be set	N
	When DENR = 1, DDRE = 0, and HCM = 1, RTS remains asserted high	N
	In deep-sleep mode, the parity error caused by wakeup frames will set PERR and EPERR bit	N
	In smartcard mode, the PERR flag is set abnormally	N
	In synchronous mode, the PERR flag is set abnormally	N
	In deep-sleep mode, the parity error caused by wakeup frames will set PERR bit but not EPERR	N
I2C	*When I2C works in 7/10 address slave mode, receiving an abnormal timing will cause the SDA line to be stuck*	Y
	When the I2C slave is configured in 10-bit address mode, if the external master does not send a STOP signal after transmitting a frame of data, the I2C slave will be unable to match the slave address in subsequent operations	N
	When the I2C is configured as a master in 10-bit address mode and fails to send a STOP signal after transmitting a data frame, subsequent data frame transmissions will encounter anomalies	Y
	Transmission timing abnormality when the I2C master is configured in 10-bit address reception mode with HEAD10R=1	Y
	When the I2C clock is configured to an IRC16M division factor other than 1, an address mismatch in low-power wake-up scenarios will cause subsequent wake-up failure	Y
	When I2C is operating as a master transmitter, if the slave responds with NACK to the last byte, a START condition cannot be correctly issued in the transfer complete interrupt	Y
	When I2C is operating as a master, the occurrence of a START signal from another master will probabilistically cause subsequent transmission failure	N
CAN	*When TFO is set to 1, after sequentially enabling 3 mailboxes for transmission, aborting the lowest priority transmit mailbox will also cause the second priority transmit mailbox to be aborted*	Y
	When TFO is cleared and the identifiers of transmission mailbox 1 or 2 are configured as 0x1FFFFFFF, the data in transmission mailbox 1 or 2 cannot be sent	Y
USBD	*The USBD SUSPEND time count is not stable*	N

Note:

Y = Limitation present, workaround available

N = Limitation present, no workaround available

'--' = Limitation fixed

2. Descriptions of device limitations

2.1. PMU

2.1.1. FWDGTRSTF flag cannot be set in Deep-sleep mode

Description & impact

FWDGTRSTF bit cannot be set by hardware when MCU is in Deep-sleep / Deep-sleep 1 / Deep-sleep 2 mode and FWDGT reset is occurred.

Workarounds

The application programme can determine whether a FWDGT reset has occurred. For example, by marking whether the system has experienced a reset, and then excluding the cause of the reset, it can be determined if it was due to a FWDGT reset.

2.2. RCU

2.2.1. The LXTALSTB bit cannot be cleared by disabling LXTAL when LXTAL stops unexpectedly

Description & impact

When LXTAL stops unexpectedly, the LXTALSTB bit cannot be cleared by disabling the LXTAL, which prevents the LXTAL from restarting.

Workarounds

By repeatedly setting and resetting the LXTALBPS more than ten times to clear the LXTALSTB bit, and then reconfiguring the LXTAL. The reference code for clearing LXTALSTB bits is as follows:

```
void lxtal_stb_clear(void)
{
    volatile    uint32_t i = 0U;
    /* close LXTAL clock */
    rcu_osci_off(RCU_LXTAL);
    /* set PC14 */
    rcu_periph_clock_enable(RCU_GPIOC);
    gpio_mode_set(GPIOC,      GPIO_MODE_OUTPUT,      GPIO_PUPD_NONE,
GPIO_PIN_14);
    gpio_output_options_set(GPIOC, GPIO_OTYPE_PP, GPIO_OSPEED_50MHZ,
```

```
GPIO_PIN_14);
        GPIO_BOP(GPIOC) = GPIO_PIN_14;
        for(i = 0; i < 10; i++) {
            /* enable the LXTAL bypass mode */
            rcu_osci_bypass_mode_enable(RCU_LXTAL);
            /* disable the LXTAL bypass mode */
            rcu_osci_bypass_mode_disable(RCU_LXTAL);
        }
}
```

2.3. USART

2.3.1. When USART is woken up from mute mode by an idle frame, it will not be woken up when it enters mute mode again

Description & impact

When USART works in multiprocessor communication mode and the USART is woken from mute mode by an idle frame, it will cause the USART to not be woken up when the bus is in idle mode and the USART enters mute mode.

Workarounds

When an idle frame is used to wake the USART mute mode, it is not allowed to enter mute mode while the bus is idle.

2.3.2. When waking up the USART from mute mode via an idle frame, IDLEF is set

Description & impact

When waking up the USART from mute mode via an idle frame, IDLEF is set. If the IDLE interrupt is enabled at this time, the IDLE interrupt handler will be executed after the idle frame wake-up.

Workarounds

Disable the IDLE interrupt before entering mute mode, and enable the IDLE interrupt when needed.

2.3.3. The high baud rate of USART will cause data loss when using hardware flow control mode

Description & impact

When using hardware flow control, during high baud rate communication of the USART, data loss may occur due to CTS not being pulled low in time (flow control delay).

Workarounds

Avoid using high baud rates, or use 2 stop bits at high baud rates. In flow control mode with 1 stop bit, limit the baud rate to within 0.94 MHz (APB2 = 64 MHz) and 0.47 MHz (APB1 = 32 MHz).

2.3.4. In smartcard mode during data reception, a parity error detected during TX transmission is considered a retransmission, and FERR and RBNE cannot be set

Description & impact

In smartcard mode during data reception, a parity error detected during TX transmission is considered a retransmission, but the TX pin does not detect a NACK signal. The read data buffer not empty flag (RBNE) and the framing error flag (FERR) cannot be set.

Workarounds

Not available.

2.3.5. When DENR = 1, DDRE = 0, and HCM = 1, RTS remains asserted high

Description & impact

When DENR = 1, DDRE = 0, and HCM = 1, the RTS signal remains asserted high, causing hardware flow control to fail.

Workarounds

Not available. Ensure that the above three conditions are not all true at the same time during operation.

2.3.6. In deep-sleep mode, the parity error caused by wakeup frames will set PERR and EPERR bit

Description & impact

In deep sleep mode, parity errors caused by wake-up frames will set the PERR bit. For example, when using USART address-match to wake up from deep sleep mode, if a frame with a parity error and a non-matching address is received first, followed by a frame with no parity error and a matching address, the PERR and EPERR bit will be set after wakeup.

Workarounds

Not available. The software ignores the parity error flag generated in this case.

2.3.7. In smartcard mode, the PERR flag is set abnormally

Description & impact

In smartcard mode, when NACK is disabled (NKEN = 0) and SCRTNUM is configured to a non-zero value, the PERR bit fails to be set after the USART receives a frame with a parity error.

Workarounds

Not available.

2.3.8. In synchronous mode, the PERR flag is set abnormally

Description & impact

In synchronous mode, when the data bit inversion function is enabled (DINV = 1), the PERR bit will still be set even if the USART receives a frame with no parity error.

Workarounds

Not available. Do not enable the data bit inversion function in synchronous mode.

2.3.9. In deep-sleep mode, the parity error caused by wakeup frames will set PERR bit but not EPERR

Description & impact

In deep sleep mode, parity errors caused by wake-up frames will set the PERR bit but not EPERR bit. For example, when using USART address-match to wake up from deep sleep mode, if a frame with a parity error and a non-matching address is received first, followed by a frame with parity error and a matching address, the PERR bit will be set after wakeup while the EPERR bit remains reset.

Workarounds

Not available. The software ignores the parity error flag generated in this case.

2.4. I2C

2.4.1. When I2C works in 7/10 address slave mode, receiving an abnormal timing will cause the SDA line to be stuck

Description & impact

When the I2C is operating as a slave device in 7-bit address mode and the I2C master simulates I2C communication via IO. If the master sends the following sequence, the I2C slave will enter an error state, causing it to malfunction and the SDA line to remain low:

Start + 10-bit Match Head Address + Start + 7-bit Address Read + Wait ACK + Start

When the I2C is operating as a slave device in 10-bit address mode and the I2C master simulates I2C communication via IO. If the master sends the following sequence, the I2C slave will enter an error state, causing it to malfunction and the SDA line to remain low:

Start + 10-bit Mismatch Head Address + Start
or
Start + 10-bit Match Head Address + Wait ACK + 10-bit Mismatch 8-bit Address + Start

Workarounds

Software periodically checks the status of the SDA line. If SDA is detected to be stuck low, reinitialize the I2C module.

2.4.2. When the I2C slave is configured in 10-bit address mode, if the external master does not send a STOP signal after transmitting a frame of data, the I2C slave will be unable to match the slave address in subsequent operations

Description & impact

When the I2C slave is configured in 10-bit address mode, if the external master does not send a STOP signal after transmitting a frame of data and instead sends a START signal to initiate the transmission of a second frame, the I2C slave will misinterpret the second byte of the slave address (the lower 8 bits of the 10-bit address) as data, and the address match flag (ADDSEND) will not be set. For example, if the slave is in address polling mode, it will continuously wait for an address match and remain stuck in a loop. Similarly, if the slave is in interrupt or DMA mode, it will fail to process subsequent data due to the inability to match the slave address.

Workarounds

Not available. When the I2C slave is operating in 10-bit address mode, the external I2C master must send the corresponding STOP signal at the end of each frame transmission.

2.4.3. When the I2C is configured as a master in 10-bit address mode and fails to send a STOP signal after transmitting a data frame, subsequent data frame transmissions will encounter anomalies

Description & impact

When the I2C master fails to send a STOP signal after transmitting a data frame and the software subsequently configures it to master receive mode, regardless of whether HEAD10R is set to 0 or 1, the waveform of the master receive part will always be RESTART + 10-bit address head + Master Receive. This means the HEAD10R configuration becomes ineffective.

If the I2C master fails to send a STOP signal after transmitting a data frame, when HEAD10R = 0, the subsequent RESTART will directly enter the waveform of master receive mode. when HEAD10R = 1, the subsequent RESTART will repeatedly send the first part of the address sequence in a loop (RESTART+10bit address head).

Workarounds

When the I2C master is configured in 10-bit address mode, a corresponding STOP signal must be sent at the end of each frame transmission.

2.4.4. Transmission timing abnormality when the I2C master is configured in 10-bit address reception mode with HEAD10R=1

Description & impact

When the I2C master is configured in 10-bit address reception mode with HEAD10R=1, the I2C master timing sequence is START + 10-bit address head + Master Receive, which causes the slave device to not ACK and fails to address the slave device. Under this configuration, a normal master transmission sequence should be START + 10-bit address head (write) + second address byte + RESTART + 10-bit address head (read).

Workarounds

When the master needs to send the sequence START + 10-bit address head (write) + second address byte + RESTART + 10-bit address head (read), configure HEAD10R to 0.

2.4.5. When the I2C clock is configured to an IRC16M division factor other than 1, an address mismatch in low-power wake-up scenarios will cause subsequent wake-up failure

Description & impact

When the I2C clock is configured to an IRC16M division factor other than 1, an address mismatch in low-power wake-up scenarios may cause the I2C clock to be lost, resulting in subsequent wake-up failure.

Workarounds

For low-power wake-up scenarios, configure IRC16MDIV to divide by 1.

2.4.6. When I2C is operating as a master transmitter, if the slave responds with NACK to the last byte, a START condition cannot be correctly issued in the transfer complete interrupt

Description & impact

When I2C is operating as a master and has finished transmitting the last byte of data, if the slave responds with a NACK signal, the master cannot correctly issue a START condition within the transfer complete (TC) interrupt, meaning the next transfer cannot be initiated.

Workarounds

Send a STOP condition in the NACK interrupt handler first, then initiate the next transfer.

2.4.7. When I2C is operating as a master, the occurrence of a START signal from another master will probabilistically cause subsequent transmission failure

Description & impact

During I2C master transmission, if a START signal sent by another master appears on the bus, subsequent data transmission will fail probabilistically, regardless of whether the master wins the arbitration.

Workarounds

Not available.

2.5. CAN

2.5.1. When TFO is set to 1, after sequentially enabling 3 mailboxes for transmission, aborting the lowest priority transmit mailbox will also cause the second priority transmit mailbox to be aborted

Description & impact

When all pending transmit mailboxes are sent in first-in-first-out order (TFO=1), such as the transmission sequence is 0->1->2. When aborting mailbox 2 transmission, if mailbox 1 is still in pending status, mailbox 1 will also be aborted, meaning the data in transmit mailbox 1 will not be sent out.

Workarounds

Use one of the following solutions:

1) Before aborting the lowest priority transmit mailbox, ensure that the second priority transmit mailbox is not in pending status.
2) After aborting the lowest priority transmit mailbox, reconfigure and transmit the second priority and lowest priority transmit mailbox.

2.5.2. When TFO is cleared and the identifiers of transmission mailbox 1 or 2 are configured as 0x1FFFFFFF, the data in transmission mailbox 1 or 2 cannot be sent

Description & impact

When TFO is cleared and the identifiers of transmission mailbox 1 or 2 are configured as 0x1FFFFFFF, the data in transmission mailbox 1 or 2 cannot be sent. For example, when TFO is set to 0 and the ID of transmission mailbox 1 is configured as 0x1FFFFFFF, the data in transmission mailbox 1 cannot be sent.

Workarounds

Use one of the following solutions:

1) When the identifier is 0x1FFFFFFF, use transmission mailbox 0 for data frame transmission.
2) Use first-in-first-out (FIFO) transmission order, i.e., set TFO = 1.

2.6. USBD

2.6.1. The USBD SUSPEND time count is not stable

Description & impact

According to the USB protocol standard, the SUSPEND time should be 3ms after detecting the IDLE state to start executing the suspend operation. However, the actual USBD SUSPEND time is not always 3ms, which is not consistent with the USB protocol standard, but does not affect the normal entry of SUSPEND.

Workarounds

Not available.

2.7. Core

About Cortex-M23 (r1p0) limitations, please refer to "Cortex-M23 Software Developer Errata Notice". This document can be downloaded on ***ARM official website***.

3. Revision history

Table 3-1. Revision history

Revision No.	Description	Date
1.0	Initial Release	May.31 2026

Important Notice

This document is the property of GigaDevice Semiconductor Inc. and its subsidiaries (the "Company"). This document, including any product of the Company described in this document (the “Product”), is owned by the Company according to the laws of the People’s Republic of China and other applicable laws. The Company reserves all rights under such laws and no Intellectual Property Rights are transferred (either wholly or partially) or licensed by the Company (either expressly or impliedly) herein. The names and brands of third party referred thereto (if any) are the property of their respective owner and referred to for identification purposes only.

To the maximum extent permitted by applicable law, the Company makes no representations or warranties of any kind, express or implied, with regard to the merchantability and the fitness for a particular purpose of the Product, nor does the Company assume any liability arising out of the application or use of any Product. Any information provided in this document is provided only for reference purposes. It is the sole responsibility of the user of this document to determine whether the Product is suitable and fit for its applications and products planned, and properly design, program, and test the functionality and safety of its applications and products planned using the Product. The Product is designed, developed, and/or manufactured for ordinary business, industrial, personal, and/or household applications only, and the Product is not designed or intended for use in (i) safety critical applications such as weapons systems, nuclear facilities, atomic energy controller, combustion controller, aeronautic or aerospace applications, traffic signal instruments, pollution control or hazardous substance management; (ii) life-support systems, other medical equipment or systems (including life support equipment and surgical implants); (iii) automotive applications or environments, including but not limited to applications for active and passive safety of automobiles (regardless of front market or aftermarket), for example, EPS, braking, ADAS (camera/fusion), EMS, TCU, BMS, BSG, TPMS, Airbag, Suspension, DMS, ICMS, Domain, ESC, DCDC, e-clutch, advanced-lighting, etc.. Automobile herein means a vehicle propelled by a self-contained motor, engine or the like, such as, without limitation, cars, trucks, motorcycles, electric cars, and other transportation devices; and/or (iv) other uses where the failure of the device or the Product can reasonably be expected to result in personal injury, death, or severe property or environmental damage (collectively "Unintended Uses"). Customers shall take any and all actions to ensure the Product meets the applicable laws and regulations. The Company is not liable for, in whole or in part, and customers shall hereby release the Company as well as its suppliers and/or distributors from, any claim, damage, or other liability arising from or related to all Unintended Uses of the Product. Customers shall indemnify and hold the Company, and its officers, employees, subsidiaries, affiliates as well as its suppliers and/or distributors harmless from and against all claims, costs, damages, and other liabilities, including claims for personal injury or death, arising from or related to any Unintended Uses of the Product.

Information in this document is provided solely in connection with the Product. The Company reserves the right to make changes, corrections, modifications or improvements to this document and the Product described herein at any time without notice. The Company shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them. Information in this document supersedes and replaces information previously supplied in any prior versions of this document.