GigaDevice Semiconductor Inc.

Device limitations of GD32W51x

Errata Sheet

Table of Contents

List of Figures

List of Tables

1. Introduction

This document applies to GD32W51x product series, as shown in ***Table 1-1. Applicable products***. It provides the technical details that need to be paid attention to in the process of using GD32 MCU, as well as solutions to related problems.

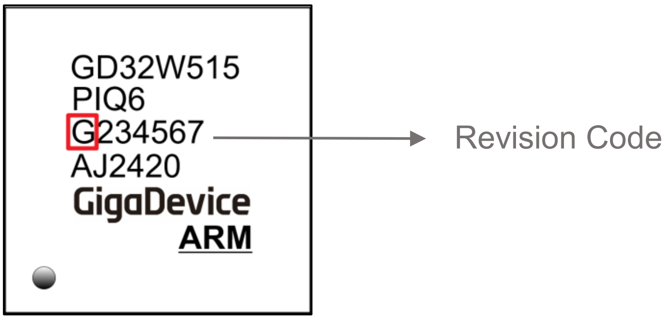
Table 1-1. Applicable products

Type	Part Numbers
MCU	GD32W515xx series

1.1. Revision identification

The device revision can be determined by the mark on the top of the package. The 1st code on the line 3 of the mark represents product revision code. As the picture shown in ***Figure 1-1. Device revision code of GD32W51x***.

Figure 1-1. Device revision code of GD32W51x



1.2. Summary of device limitations

The device limitations of GD32W51x 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 G	Rev. Code H
ICACHE	***When using ICACHE for SQPI access, if a cross-page boundary access occurs, there is a probability of data read errors***	Y	Y
PMU	***Standby mode cannot be waked up due to frequent wakeup signals before or after entering standby mode***	N	N
	IRC32K clock frequency is inaccurate in standby mode	N	N
	The power consumption state may be incorrect after	Y	Y

	MCU reset		
RCU	*MCU can not be waked up after entering deep-sleep mode when DSLP_HOLD bit is set*	Y	Y
TIMER	*Data lost when using timer capture / compare event to trigger DMA transfer and enabling the output compare shadow function*	Y	Y
USART	*When the USART is configured in half-duplex single-wire mode and the TX / RX pin functions are swapped, the transmission and reception data polarity is inconsistent*	Y	Y
	In mute mode, when noise errors, frame errors, and parity errors occur with the wake-up address, the corresponding error flag bits will be set to 1	N	N
	In deep-sleep mode, the parity error caused by wakeup frames will set PERR bit	N	N
	Frame errors are only detected on the first stop bit when the stop bit is configured as 2 bits	N	N
I2C	*When SDA line interference causes garbled data on the I2C bus, it can lead to a stuck in the I2C slave device*	N	N
	After the I2C0 address match wakes up the MCU, it will cause the I2C bus to run incorrectly	Y	Y
SPI	*When the SPI CRC function is enabled, switching from receive-only mode to transmit-only mode will cause the CRCERR to be set*	N	N
QSPI	*When the QSPI transfer timeout event occurs simultaneously with a bus read memory operation, it will cause the bus to stuck*	Y	Y
USBFS	*When USBFS-OTG is communicating with low-speed devices through the HUB, the data transmission is abnormal USBFS-OTG*	N	N
	The USBFS-OTG suspension state entry time is not 3ms	N	N
TSI	*The starting function of TSI charge transfer sequence is abnormal in the rising edge triggering mode*	Y	Y
PKCAU	*When the PKCAU module is performing calculations, the MCU cannot enter deep sleep mode*	Y	Y
Core	*Access permission faults are prioritized over unaligned Device memory faults*	N	N

Note:

Y = Limitation present, workaround available

N = Limitation present, no workaround available

'--' = Limitation fixed

2. Descriptions of device limitations

2.1. ICACHE

2.1.1. When using ICACHE for SQPI access, if a cross-page boundary access occurs, there is a probability of data read errors

Description & impact

When using ICACHE for SQPI access, if a cross-page boundary access occurs, there is a probability of data read errors.

Note: The page structure can be obtained by the datasheet of the corresponding external storage device.

Workarounds

When performing SQPI reads, set the SQPI access address to uncacheable access mode.

2.2. PMU

2.2.1. Standby mode cannot be waked up due to frequent wakeup signals before or after entering standby mode

Description & impact

When reset the internal signal STBY_CTL to enter to standby mode, if the Tglitch is smaller than 100ns, which will cause the mcu cannot be waked up. The narrow glitch will result in incorrect Vcore voltage.

Note: The T_{glitch} is the time between STBY_CTL low level and the wakeup signal (PA0 high level).

Workarounds

Not available.

2.2.2. IRC32K clock frequency is inaccurate in standby mode

Description & impact

When the MCU is in standby mode, the IRC32K clock frequency is inaccurate due to the calibration value of IRC32K not being properly latched.

Workarounds

Not available.

2.2.3. The power consumption state may be incorrect after MCU reset

Description & impact

After the MCU reset, the power consumption state may be incorrect due to the PMU clock not being enabled.

Workarounds

Application programme needs to enable PMU clock (PMUEN = 1)

2.3. RCU

2.3.1. MCU can not be waked up after entering deep-sleep mode when DSLP_HOLD bit is set

Description & impact

When DSLP_HOLD bit is set and debug the mcu in deep-sleep mode, the mcu will not be waked up.

Workarounds

When the DSLP_HOLD bit is set to enable low power debugging, the application programme need switch the system clock to IRC16M before entering the deep-sleep mode.

2.4. TIMER

2.4.1. Data lost when using timer capture / compare event to trigger DMA transfer and enabling the output compare shadow function

Description & impact

When using timer capture / compare event to trigger DMA transfer and enabling the output compare shadow function, DMA transfers data 0x00 to TIMERx_CHyCV register which will result in the second data lost after data 0x00.

Workarounds

Use one of the following methods:

1) Do not use data 0x00 in DMA transfer buffer.

2) Transfer the second data after the 0x00 twice.

3) Use the timer update event to trigger DMA transfer.

2.5. USART

2.5.1. When the USART is configured in half-duplex single-wire mode and the TX / RX pin functions are swapped, the transmission and reception data polarity is inconsistent

Description & impact

When the USART is configured in half-duplex single-wire mode and the functions of the TX/RX pins are swapped, if the TINV and RINV configurations are inconsistent, which will cause the polarity of the transmitted and received data to be inconsistent.

Workarounds

Configure TINV and RINV with the same settings.

2.5.2. In mute mode, when noise errors, frame errors, and parity errors occur with the wake-up address, the corresponding error flag bits will be set to 1

Description & impact

In mute mode, when noise errors, frame errors, and parity errors occur with the wake-up address, the corresponding error flag bits will be set to 1.

Workarounds

The software ignores the error flag generated in this case.

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

Description & impact

In deep-sleep mode, a parity error caused by a wakeup frame will result in a parity error (the PERR bit in the USART_STAT register is set).

Workarounds

The software ignores the parity error flag generated in this case.

2.5.4. Frame errors are only detected on the first stop bit when the stop bit is configured as 2 bits

Description & impact

Frame errors are only detected on the first stop bit when the stop bit is configured as 2 bits.

Workarounds

Do not set the stop bit length to 2 bits, or use software methods to verify the frame data.

2.6. I2C

2.6.1. When SDA line interference causes garbled data on the I2C bus, it can lead to a stuck in the I2C slave device

Description & impact

When I2C operates as a slave and is configured in 7-bit addressing mode, and if there is a mismatch in the 10-bit address header during the I2C slave addressing phase, interference on the SCL / SDA line that causes the next RESTART/STOP signal to be sent early can result in the I2C slave pulling the SDA line low, ultimately leading to the I2C slave stuck.

When I2C operates as a slave and is configured in 10-bit addressing mode, and if there is a mismatch in the 10-bit address header or the lower 8 bits of the 10-bit address during the I2C slave addressing phase, interference on the SCL / SDA line that causes the next RESTART/STOP signal to be sent early can result in the I2C slave pulling the SDA line low, ultimately leading to the I2C slave stuck.

Workarounds

Not available.

2.6.2. After the I2C0 address match wakes up the MCU, it will cause the I2C bus to run incorrectly

Description & impact

When using the I2C0 address match to wake up the MCU from deep sleep, the I2C bus may run incorrectly due to the address match signal not being cleared in time during the interrupt handling, which leads to the extra issuance of a start signal.

Workarounds

Clear the WUEN bit before clearing ADDSEND by setting ADDSENDC.

2.7. SPI

2.7.1. When the SPI CRC function is enabled, switching from receive-only mode to transmit-only mode will cause the CRCERR to be set

Description & impact

When the SPI CRC function is enabled and the SPI firstly operates in receive-only mode, the CRC calculates continuously. Then, when the operating mode is switched to transmit-only mode, the internal Rx CRC value used for comparison is cleared to zero, while the RCRC value in the SPI_RCRC register remains. This ultimately causes the CRCERR flag to be set.

Workarounds

Ignore this CRC error or avoid this usage.

2.8. QSPI

2.8.1. When the QSPI transfer timeout event occurs simultaneously with a bus read memory operation, it will cause the bus to stuck

Description & impact

When the QSPI transfer timeout event occurs simultaneously with a bus read memory operation, it will cause the bus to stuck.

Workarounds

Disable QSPI timeout function by clearring TMOUTEN or set the TMOUT value to 0xFFFF.

2.9. USBFS

2.9.1. When USBFS-OTG is communicating with low-speed devices through the HUB, the data transmission is abnormal USBFS-OTG

Description & impact

When USBFS-OTG is communicating with low-speed devices through the HUB, the data transmission is abnormal.

Workarounds

Not available.

2.9.2. The USBFS-OTG suspension state entry time is not 3ms

Description & impact

The USBFS-OTG suspension state entry time is not 3ms.

Workarounds

Not available. But it does not affect the normal use of USBFS-OTG.

2.10. TSI

2.10.1. The starting function of TSI charge transfer sequence is abnormal in the rising edge triggering mode

Description & impact

When the TSI charge transfer sequence start mode is set to hardware trigger mode and the trigger edge is a rising edge, the trigger pin can start the conversion sequence even if there is no rising edge.

Workarounds

In hardware trigger mode, do not use the rising edge trigger mode.

2.11. PKCAU

2.11.1. When the PKCAU module is performing calculations, the MCU cannot enter deep sleep mode

Description & impact

When the PKCAU module is performing calculations, the MCU cannot enter deep sleep mode.

Workarounds

Wait for the PKCAU computation to be completed, then the MCU enters deep sleep mode.

2.12. Core

2.12.1. Access permission faults are prioritized over unaligned Device memory faults

This limitation refers to Arm ID number 1080541 in “Cortex-M33 AT623 and Cortex-M33 with FPU AT624 Software Developers Errata Notice”.

Description & impact

A load or store which causes an unaligned access to Device memory will result in an UNALIGNED UsageFault exception. However, if the region is not accessible because of the MPU access permissions (as specified in MPU_RBAR.AP), then the resulting MemManage fault will be prioritized over the UsageFault.

This erratum affects all configurations of the Cortex-M33 processor with the MPU enabled.

The conditions is that the MPU is enabled and:

- A load/store access occurs to an address which is not aligned to the data type specified in the instruction.
- The memory access hits one region only.
- The region attributes (specified in the MAIR register) mark the location as Device memory.
- The region access permissions prevent the access (that is, unprivileged or write not allowed).

The implications of this limitation is that the MemManage fault caused by the access permission violation will be prioritized over the UNALIGNED UsageFault exception because of the memory attributes.

Workarounds

There is no workaround. However, it is expected that no existing software is relying on this behavior since it was permitted in Armv7-M.

3. Revision history

Table 3-1. Revision history

Revision No.	Description	Date
1.0	Initial Release	Aug.29 2024

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 under the intellectual property laws and treaties of the People’s Republic of China and other jurisdictions worldwide. The Company reserves all rights under such laws and treaties and does not grant any license under its patents, copyrights, trademarks, or other intellectual property rights. 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.

The Company makes no warranty of any kind, express or implied, with regard to this document or any Product, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. The Company does not assume any liability arising out of the application or use of any Product described in this document. Any information provided in this document is provided only for reference purposes. It is the responsibility of the user of this document to properly design, program, and test the functionality and safety of any application made of this information and any resulting product. Except for customized products which has been expressly identified in the applicable agreement, the Products are designed, developed, and/or manufactured for ordinary business, industrial, personal, and/or household applications only. The Products are not designed, intended, or authorized for use as components in systems designed or intended for the operation of weapons, weapons systems, nuclear installations, atomic energy control instruments, combustion control instruments, airplane or spaceship instruments, transportation instruments, traffic signal instruments, life-support devices or systems, other medical devices or systems (including resuscitation equipment and surgical implants), pollution control or hazardous substances management, or other uses where the failure of the device or Product could cause personal injury, death, property or environmental damage ("Unintended Uses"). Customers shall take any and all actions to ensure using and selling the Products in accordance with the applicable laws and regulations. The Company is not liable, in whole or in part, and customers shall and hereby do release the Company as well as it’s suppliers and/or distributors from any claim, damage, or other liability arising from or related to all Unintended Uses of the Products. Customers shall indemnify and hold the Company as well as it’s 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 Products.

Information in this document is provided solely in connection with the Products. The Company reserves the right to make changes, corrections, modifications or improvements to this document and Products and services described herein at any time, without notice.