

**GigaDevice Semiconductor Inc.**

**GD32W515P-EVAL**

**User Guide**

**V1.0**

# Tables of Contents

<b>TABLES OF CONTENTS .....</b>	<b>1</b>
<b>LIST OF FIGURES .....</b>	<b>4</b>
<b>LIST OF TABLES .....</b>	<b>5</b>
<b>1. SUMMARY .....</b>	<b>6</b>
<b>2. FUNCTION PIN ASSIGN .....</b>	<b>6</b>
<b>3. GETTING STARTED .....</b>	<b>7</b>
<b>4. HARDWARE LAYOUT OVERVIEW .....</b>	<b>7</b>
4.1. Power supply.....	7
4.2. Boot option.....	7
4.3. LED .....	8
4.4. KEY .....	8
4.5. ADC .....	8
4.6. USART .....	9
4.7. I2C.....	9
4.8. I2S.....	9
4.9. HPDF.....	10
4.10. IFRP.....	10
4.11. TSI .....	10
4.12. LCD.....	11
4.13. SDIO .....	11
4.14. USB .....	12
4.15. Extension .....	12
4.16. GD-Link .....	12
4.17. MCU.....	13
<b>5. ROUTINE USE GUIDE .....</b>	<b>14</b>
5.1. GPIO_Running_LED.....	14
5.1.1. DEMO purpose.....	14
5.1.2. DEMO running result.....	14
5.2. GPIO_Key_Polling_mode.....	14
5.2.1. DEMO purpose.....	14
5.2.2. DEMO running result.....	14
5.3. EXTI_Key_Interrupt_mode.....	15
5.3.1. DEMO purpose.....	15
5.3.2. DEMO running result.....	15
5.4. USART_Printf.....	15
5.4.1. DEMO purpose.....	15
5.4.2. DEMO running result.....	15

<b>5.5. USART_Echo_Interrupt_mode.....</b>	<b>16</b>
5.5.1. DEMO purpose.....	16
5.5.2. DEMO running result.....	16
<b>5.6. USART_DMA.....</b>	<b>16</b>
5.6.1. DEMO purpose.....	16
5.6.2. DEMO running result.....	17
<b>5.7. ADC_conversion_triggered_by_timer.....</b>	<b>17</b>
5.7.1. DEMO purpose.....	17
5.7.2. DEMO running result.....	17
<b>5.8. I2C_EEPROM.....</b>	<b>18</b>
5.8.1. DEMO purpose.....	18
5.8.2. DEMO running result.....	18
<b>5.9. HPDF_I2S_Audio.....</b>	<b>19</b>
5.9.1. DEMO purpose.....	19
5.9.2. DEMO running result.....	19
<b>5.10. SPI_LCD.....</b>	<b>20</b>
5.10.1. DEMO purpose.....	20
5.10.2. DEMO running result.....	20
<b>5.11. SDIO_SDCard.....</b>	<b>20</b>
5.11.1. DEMO purpose.....	20
5.11.2. DEMO running result.....	21
<b>5.12. TRNG_Get_Random.....</b>	<b>21</b>
5.12.1. DEMO purpose.....	21
5.12.2. DEMO running result.....	22
<b>5.13. CAU.....</b>	<b>22</b>
5.13.1. DEMO purpose.....	22
5.13.2. DEMO running result.....	22
<b>5.14. HAU.....</b>	<b>24</b>
5.14.1. DEMO purpose.....	24
5.14.2. DEMO running result.....	24
<b>5.15. PKCAU_Modular_Addition_Interrupt.....</b>	<b>25</b>
5.15.1. DEMO purpose.....	25
5.15.2. DEMO running result.....	25
<b>5.16. RCU_Clock_Out.....</b>	<b>25</b>
5.16.1. DEMO purpose.....	25
5.16.2. DEMO running result.....	26
<b>5.17. PMU_Sleep_Wakeup.....</b>	<b>26</b>
5.17.1. DEMO purpose.....	26
5.17.2. DEMO running result.....	26
<b>5.18. RTC_Calendar.....</b>	<b>26</b>
5.18.1. DEMO purpose.....	26
5.18.2. DEMO running result.....	26
<b>5.19. TSI.....</b>	<b>27</b>
5.19.1. DEMO running result.....	27

<b>5.20.</b>	<b>IFPR.....</b>	<b>27</b>
5.20.1.	DEMO purpose.....	27
5.20.2.	DEMO running result.....	28
<b>5.21.</b>	<b>TIMER_Breath_LED .....</b>	<b>28</b>
5.21.1.	DEMO purpose.....	28
5.21.2.	DEMO running result.....	28
<b>5.22.</b>	<b>USBFS_Device.....</b>	<b>28</b>
5.22.1.	HID_Keyboard.....	28
4.1.1.	CDC_ACM.....	29
<b>5.23.</b>	<b>USBFS_Host .....</b>	<b>30</b>
5.23.1.	HID_Host.....	30
5.23.2.	MSC .....	30
<b>5.24.</b>	<b>Trustzone .....</b>	<b>31</b>
5.24.1.	DEMO purpose.....	31
5.24.2.	DEMO running result.....	31
<b>6.</b>	<b>REVISION HISTORY .....</b>	<b>32</b>

## List of Figures

Figure 4-1. Schematic diagram of power supply.....	7
Figure 4-2. Schematic diagram of boot option.....	7
Figure 4-3. Schematic diagram of LED function.....	8
Figure 4-4. Schematic diagram of Key function.....	8
Figure 4-5. Schematic diagram of ADC .....	8
Figure 4-7. Schematic diagram of USART .....	9
Figure 4-8. Schematic diagram of I2C .....	9
Figure 4-8. Schematic diagram of I2S .....	9
Figure 4-9. Schematic diagram of HPDF.....	10
Figure 4-9. Schematic diagram of IFRP.....	10
Figure 4-9. Schematic diagram of TSI.....	10
Figure 4-13. Schematic diagram of LCD.....	11
Figure 4-14. Schematic diagram of SDIO.....	11
Figure 4-15. Schematic diagram of USB.....	12
Figure 4-16. Schematic diagram of Extension .....	12
Figure 4-17. Schematic diagram of GD-Link.....	12
Figure 4-18. Schematic diagram of MCU .....	13

# List of Tables

Table 2-1. Function pin assignment.....	6
Table 6-1. Revision history.....	32

## 1. Summary

GD32W515P-EVAL uses GD32W515PIQ6 as the main controller. It uses GD-Link Mini USB interface to supply 5V power. Reset, Boot, K1, K2, LED, I2S, HPDF, I2C-EEPROM, LCD, IFPR, TSI, SDIO, USB and USART to USB interface are also included. For more details please refer to GD32W515P-EVAL-Rev1.1 schematic.

## 2. Function Pin Assign

**Table 2-1. Function pin assignment**

Function	Pin	Description
LED	PB6	LED1
	PA15	LED2
	PA6	LED3
RESET		K1-Reset
KEY	PA2	K2-Tamper/Wakeup
ADC	PA0	ADC_IN0
USART	PB10	USART2_TX
	PB11	USART2_RX
I2C	PB15	I2C1_SCL
	PA8	I2C1_SDA
I2S	PA4	I2S1_SD
	PB8	I2S1_CK
	PA7	I2S1_WS
	PA15	I2S1_MCK
HPDF	PA3	HPDF_DATA1
	PC2	HPDF_CKOUT
IFPR	PB5	IR_OUT
	PB7	TIMER_CH1
LCD	PB12	SP1_NSS
	PC6	LCD_RESET
	PC7	LCD_D/C
	PB9	SP1_MOSI
	PB13	SP1_SCK
	PB14	SP1_MISO
TSI	PB0	G1_IO0
	PB1	G1_IO1
	PB2	G1_IO2
SDIO	PA9	SDIO_CMD
	PA11	SDIO_CLK
	PA10	SDIO_DAT0

Function	Pin	Description
	PA12	SDIO_DAT1
	PB3	SDIO_DAT2
	PB4	SDIO_DAT3
USB	PB11	USB_ID
	PB14	USB_VBUS
	PB13	USB_DM
	PB12	USB_DP

### 3. Getting started

The EVAL board uses GD-Link Mini USB connector to get power DC +5V, which is the hardware system normal work voltage. A GD-Link on board is necessary in order to download and debug programs. Select the correct boot mode and then power on, the LEDPWR will turn on, which indicates the power supply is OK.

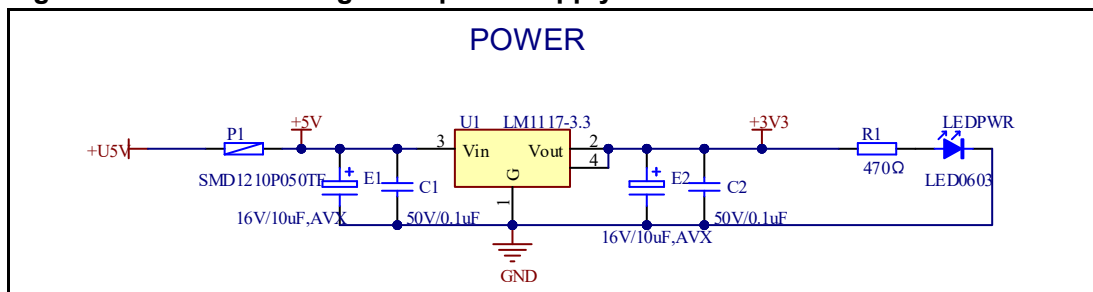
There are Keil version and IAR version of all projects. Keil version of the projects are created based on Keil MDK-ARM 5.27 uVision5. IAR version of the projects are created based on IAR Embedded Workbench for ARM 8.32.1. During use, the following points should be noted:

1. If you use Keil uVision5 to open the project. In order to solve the "Device Missing (s)" problem, you can install GigaDevice.GD32W51x\_DFP.1.0.0.pack.
2. If you use IAR to open the project, install IAR\_GD32W51x\_ADDON\_1.0.0.exe to load the associated files.

### 4. Hardware layout overview

#### 4.1. Power supply

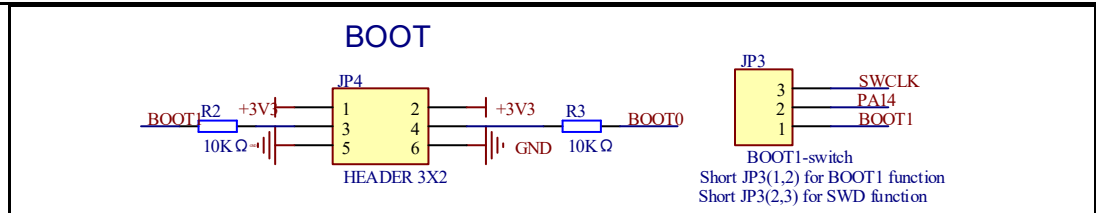
Figure 4-1. Schematic diagram of power supply



#### 4.2. Boot option

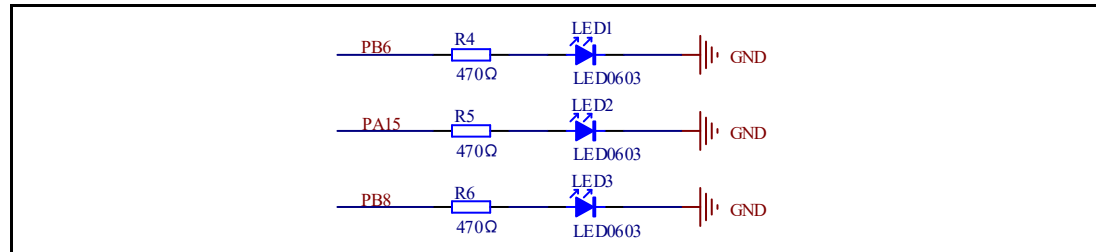
Figure 4-2. Schematic diagram of boot option





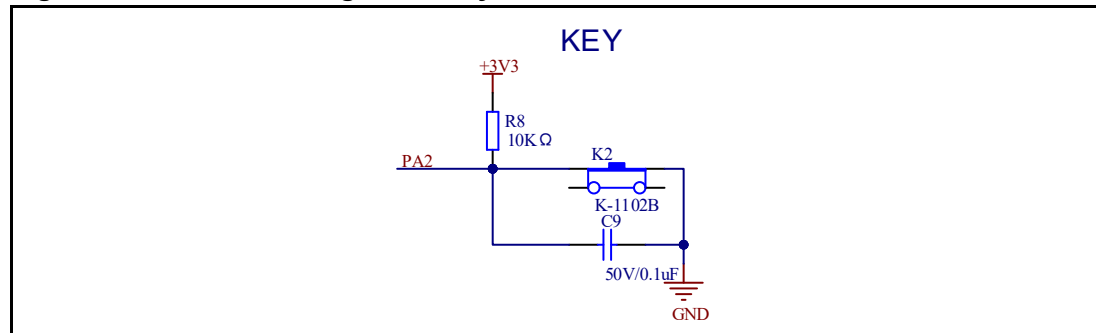
### 4.3. LED

Figure 4-3. Schematic diagram of LED function



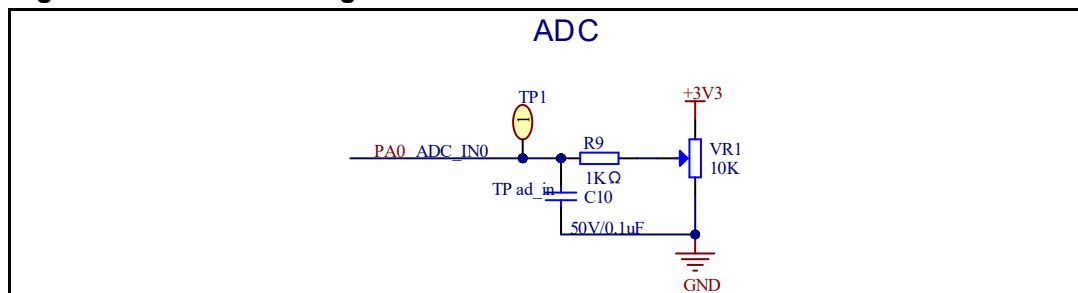
### 4.4. KEY

Figure 4-4. Schematic diagram of Key function



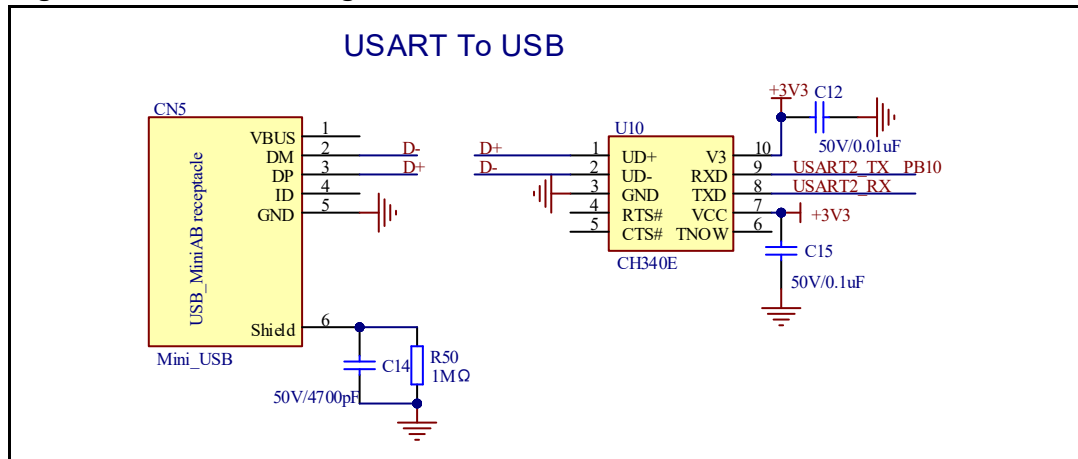
### 4.5. ADC

Figure 4-5. Schematic diagram of ADC



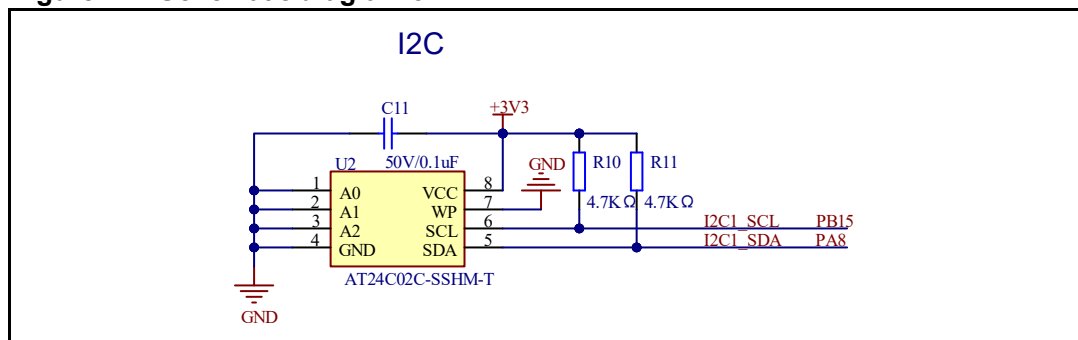
## 4.6. USART

Figure 4-6. Schematic diagram of USART



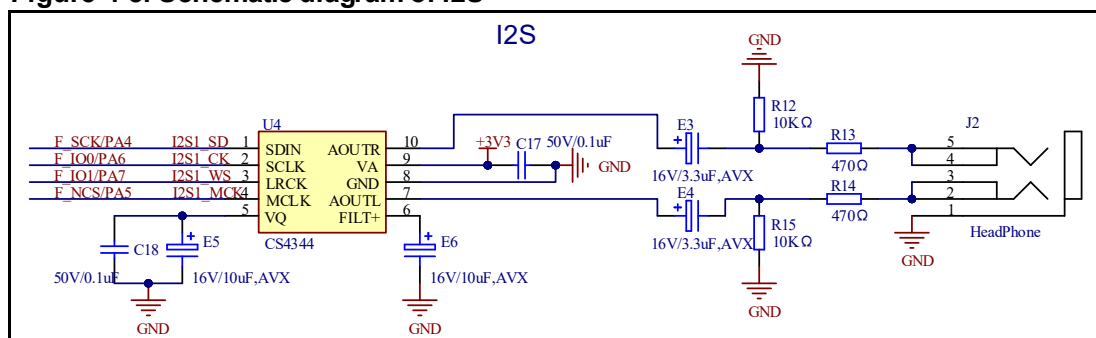
## 4.7. I2C

Figure 4-7. Schematic diagram of I2C



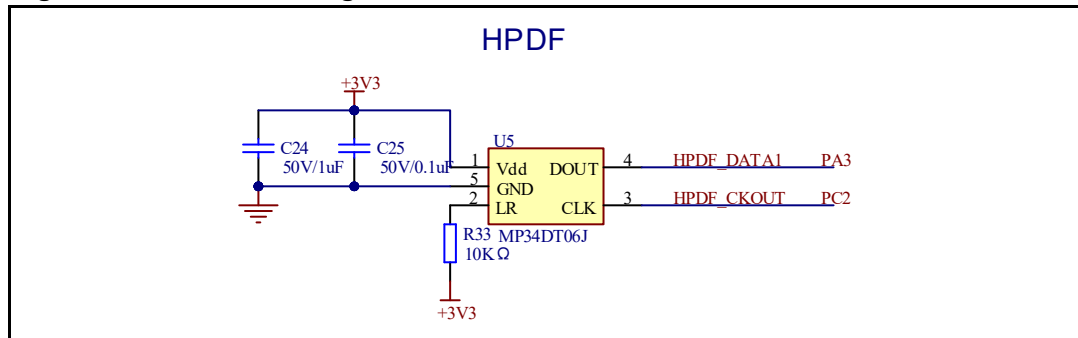
## 4.8. I2S

Figure 4-8. Schematic diagram of I2S



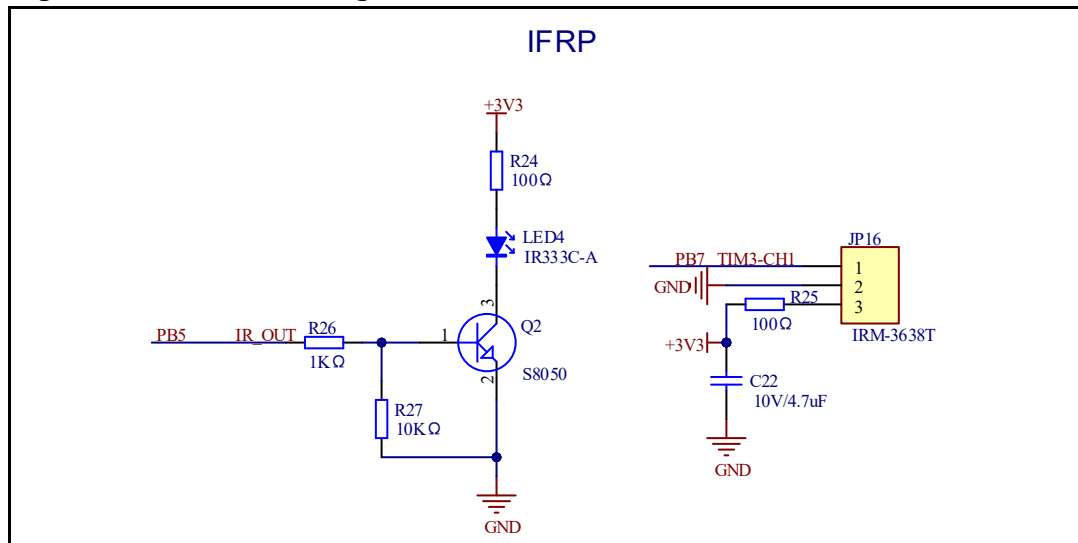
## 4.9. HPDF

Figure 4-9. Schematic diagram of HPDF



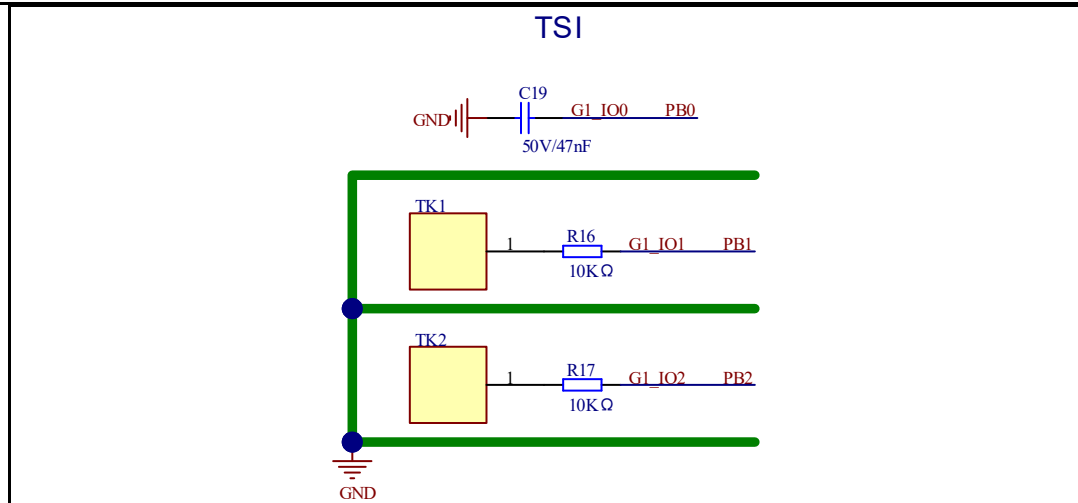
## 4.10. IFRP

Figure 4-10. Schematic diagram of IFRP



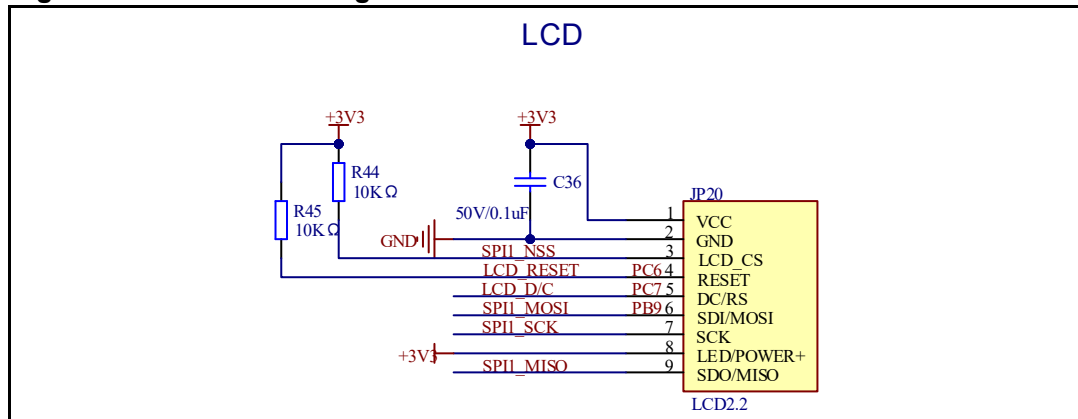
## 4.11. TSI

Figure 4-11. Schematic diagram of TSI



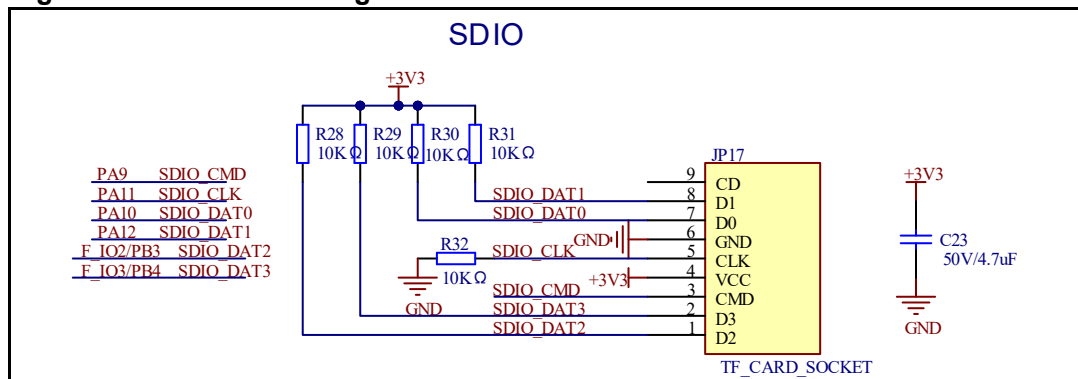
## 4.12. LCD

Figure 4-12. Schematic diagram of LCD



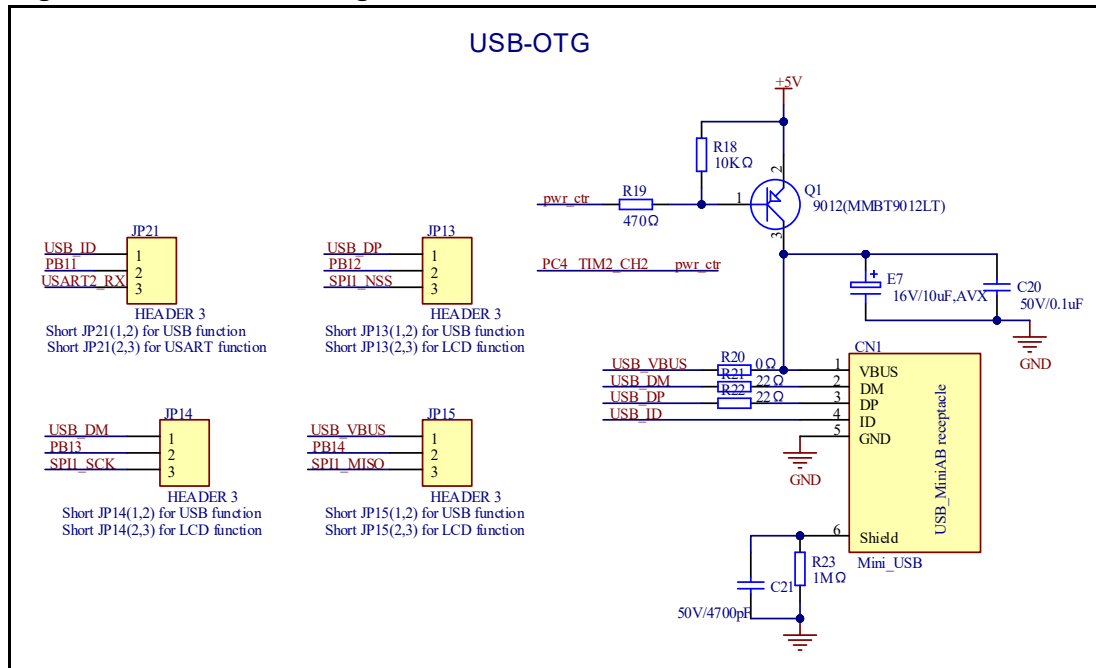
## 4.13. SDIO

Figure 4-13. Schematic diagram of SDIO



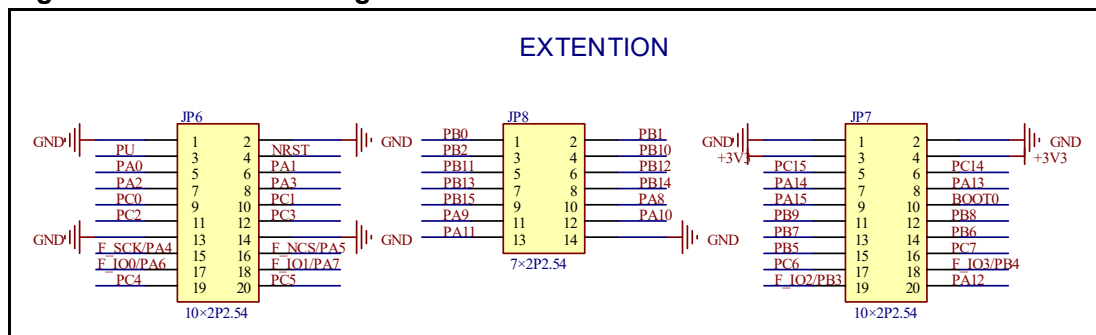
## 4.14. USB

Figure 4-14. Schematic diagram of USB



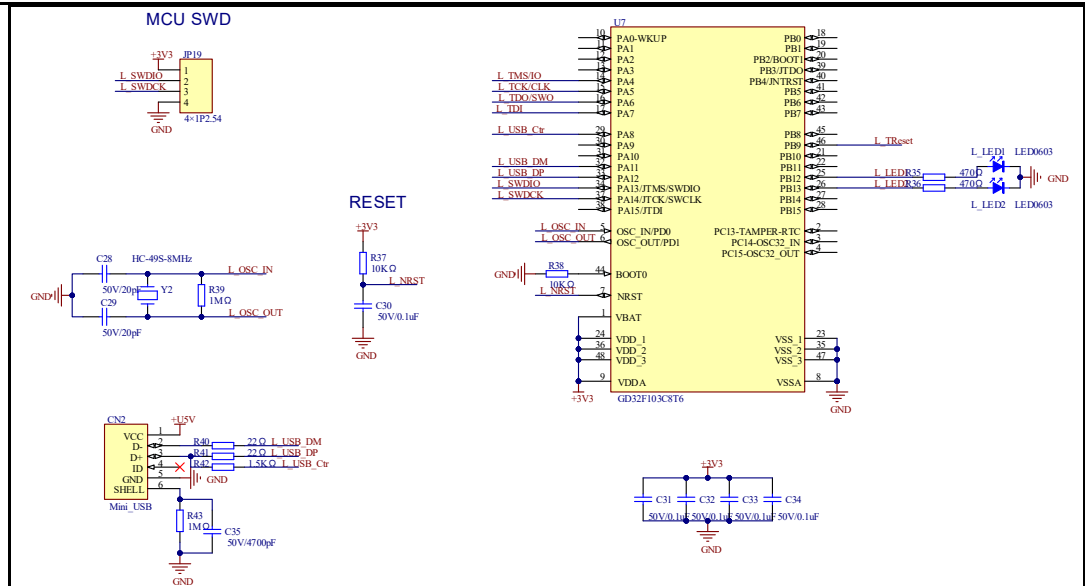
## 4.15. Extension

Figure 4-15. Schematic diagram of Extension



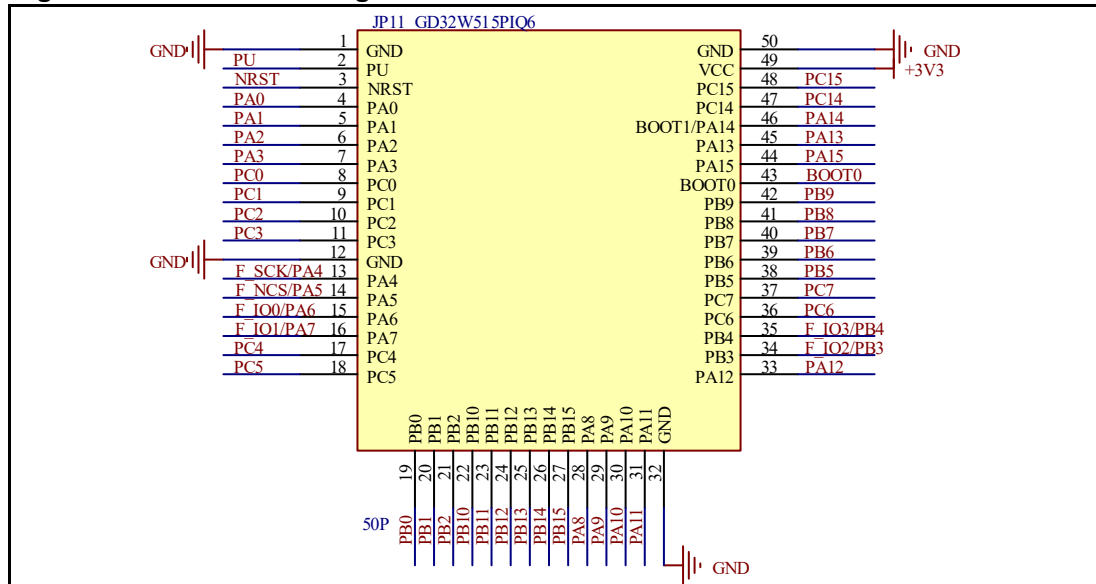
## 4.16. GD-Link

Figure 4-16. Schematic diagram of GD-Link



## 4.17. MCU

Figure 4-17. Schematic diagram of MCU



## 5. Routine use guide

### 5.1. GPIO\_Running\_LED

#### 5.1.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

- Learn to use GPIO control the LED
- Learn to use SysTick to generate 1ms delay

GD32W515P-EVAL-V1.1 board has one user key and three LEDs, the LEDs are controlled by GPIO.

This demo will show how to light the LEDs.

#### 5.1.2. DEMO running result

Download the program < 01\_GPIO\_Running\_LED > to the EVAL board, three LEDs can light cycles.

### 5.2. GPIO\_Key\_Polling\_mode

#### 5.2.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

- Learn to use GPIO control the LED and the KEY
- Learn to use SysTick to generate 1ms delay

GD32W515P-EVAL-V1.1 board has one user key and three LEDs. The key is K2-Tamper/Wakeup. The LEDs are controlled by GPIO.

This demo will show how to use the K2-Tamper/Wakeup to control the LED2. When press down the K2-Tamper/Wakeup, it will check the input value of the IO port. If the value is 0 and will wait for 100ms. Check the input value of the IO port again. If the value still is 0, it indicates that the button is pressed successfully and toggle LED2.

#### 5.2.2. DEMO running result

Download the program < 02\_GPIO\_Key\_Polling\_mode > to the EVAL board, press down the K2-Tamper/Wakeup, LED2 will be turned on. Press down the K2-Tamper/Wakeup again, LED2 will be turned off.

## 5.3. EXTI\_Key\_Interrupt\_mode

### 5.3.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

- Learn to use GPIO control the LED and the KEY
- Learn to use EXTI to generate external interrupt

GD32W515P-EVAL-V1.1 board has one user key and three LEDs. The key is K2-Tamper/Wakeup. The LEDs are controlled by GPIO.

This demo will show how to use the EXTI interrupt line to control the LED2. When press down the K2-Tamper/Wakeup, it will produce an interrupt. In the interrupt service function, the demo will toggle LED2.

### 5.3.2. DEMO running result

Download the program < 03\_EXTI\_Key\_Interrupt\_mode > to the EVAL board, LED2 is turned on and off for test. When press down the K2-Tamper/Wakeup, LED2 will be turned on. Press down the K2-Tamper/Wakeup again, LED2 will be turned off.

## 5.4. USART\_Printf

### 5.4.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

- Learn to use GPIO control the LED
- Learn to retarget the C library printf function to the USART

### 5.4.2. DEMO running result

Download the program < 04\_USART\_Printf > to the EVAL board, connect serial cable to USART. Firstly, all the LEDs are turned on and off for test. Then, this implementation outputs "USART printf example: please press the Tamper/Wakeup Key" on the HyperTerminal using USART. Press the Tamper/Wakeup Key, the LED1 will be turned on and serial port will output "USART printf example".

The output information via the HyperTerminal is as following:



```
please press the Tamper/Wakeup Key

USART printf example
```

## 5.5. USART\_Echo\_Interrupt\_mode

### 5.5.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

- Learn to use the USART transmit and receive interrupts to communicate with the HyperTerminal.

### 5.5.2. DEMO running result

Download the program <05\_USART\_Echo\_Interrupt\_mode> to the EVAL board, connect serial cable to USART. Firstly, all the LEDs are turned on and off for test. Then, the USART0 sends the tx\_buffer array (from 0x00 to 0xFF) to the hyperterminal and waits for receiving data from the hyperterminal that you must send. The string that you have sent is stored in the rx\_buffer array. The receive buffer have a BUFFER\_SIZE bytes as maximum. After that, compare tx\_buffer with rx\_buffer. If tx\_buffer is same with rx\_buffer, LED1, LED2 are tum on, and LED3 is turn off. Otherwise, LED1 is turn on and LED2, LED3 are turn off.

The output information via the HyperTerminal is as following:

```
00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F 10 11 12 13 14 15 16 17
18 19 1A 1B 1C 1D 1E 1F 20 21 22 23 24 25 26 27 28 29 2A 2B 2C 2D 2E 2F
30 31 32 33 34 35 36 37 38 39 3A 3B 3C 3D 3E 3F 40 41 42 43 44 45 46 47
48 49 4A 4B 4C 4D 4E 4F 50 51 52 53 54 55 56 57 58 59 5A 5B 5C 5D 5E 5F
60 61 62 63 64 65 66 67 68 69 6A 6B 6C 6D 6E 6F 70 71 72 73 74 75 76 77
78 79 7A 7B 7C 7D 7E 7F 80 81 82 83 84 85 86 87 88 89 8A 8B 8C 8D 8E 8F
90 91 92 93 94 95 96 97 98 99 9A 9B 9C 9D 9E 9F A0 A1 A2 A3 A4 A5 A6 A7
A8 A9 AA AB AC AD AE AF B0 B1 B2 B3 B4 B5 B6 B7 B8 B9 BA BB BC BD BE BF
C0 C1 C2 C3 C4 C5 C6 C7 C8 C9 CA CB CC CD CE CF D0 D1 D2 D3 D4 D5 D6 D7
D8 D9 DA DB DC DD DE DF E0 E1 E2 E3 E4 E5 E6 E7 E8 E9 EA EB EC ED EE EF
F0 F1 F2 F3 F4 F5 F6 F7 F8 F9 FA FB FC FD FE FF
```

## 5.6. USART\_DMA

### 5.6.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

- Learn to use the USART transmit and receive data using DMA.

### 5.6.2. DEMO running result

Download the program <06\_USART\_DMA> to the EVAL board, connect serial cable to USART. Firstly, all the LEDs are flash 3 times for test. Then, the USART sends the tx\_buffer array to the hyperterminal and waits for receiving data from the hyperterminal that you must send. The string that you have sent is stored in the rx\_buffer array. The receive buffer have a BUFFER\_SIZE bytes as maximum. After that, compare tx\_buffer with rx\_buffer. If tx\_buffer is same with rx\_buffer, LED1, LED2, LED3 are turn on. Otherwise, LED1, LED2, LED3 are turn off.

The output information via the HyperTerminal is as following:

```
00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F 10 11 12 13 14 15 16 17
18 19 1A 1B 1C 1D 1E 1F 20 21 22 23 24 25 26 27 28 29 2A 2B 2C 2D 2E 2F
30 31 32 33 34 35 36 37 38 39 3A 3B 3C 3D 3E 3F 40 41 42 43 44 45 46 47
48 49 4A 4B 4C 4D 4E 4F 50 51 52 53 54 55 56 57 58 59 5A 5B 5C 5D 5E 5F
60 61 62 63 64 65 66 67 68 69 6A 6B 6C 6D 6E 6F 70 71 72 73 74 75 76 77
78 79 7A 7B 7C 7D 7E 7F 80 81 82 83 84 85 86 87 88 89 8A 8B 8C 8D 8E 8F
90 91 92 93 94 95 96 97 98 99 9A 9B 9C 9D 9E 9F A0 A1 A2 A3 A4 A5 A6 A7
A8 A9 AA AB AC AD AE AF B0 B1 B2 B3 B4 B5 B6 B7 B8 B9 BA BB BC BD BE BF
C0 C1 C2 C3 C4 C5 C6 C7 C8 C9 CA CB CC CD CE CF D0 D1 D2 D3 D4 D5 D6 D7
D8 D9 DA DB DC DD DE DF E0 E1 E2 E3 E4 E5 E6 E7 E8 E9 EA EB EC ED EE EF
F0 F1 F2 F3 F4 F5 F6 F7 F8 F9 FA FB FC FD FE FF
```

## 5.7. ADC\_conversion\_triggered\_by\_timer

### 5.7.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

- Learn to use the ADC to convert analog signal to digital data
- Learn to use TIMER to generate a channel compare event

TIMER0 CH0 event triggers ADC conversion, the value corresponds to the ADC analog input, and changes with it. The converted data are moved to SRAM through DMA continuously, and printed by USART.

### 5.7.2. DEMO running result

Download the program <07\_ADC\_conversion\_triggered\_by\_timer> to the GD32W515P-EVAL-V1.1 board and connect serial cable to USART, adjust the adjustable potentiometer knob to change the analog input. The ADC, which is triggered by TIMER0 CH0 event, will convert the analog input, and you will see the result by USART.

```
//*****//  
the ADC conversion result is 0x0AFE  
  
//*****//  
the ADC conversion result is 0x0AFE  
  
//*****//  
the ADC conversion result is 0x0AFE
```

## 5.8. I2C\_EEPROM

### 5.8.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

- Learn to use the master transmitting mode of I2C module
- Learn to use the master receiving mode of I2C module
- Learn to read and write the EEPROM with I2C interface

### 5.8.2. DEMO running result

Download the program <8\_I2C\_EEPROM> to the EVAL board and run. Connect serial cable to USART, jump JP21 to USART, then open the HyperTerminal to show the print message.

Firstly, the data of 256 bytes will be written to the EEPROM from the address 0x00 and printed by the serial port. Then, reading the EEPROM from address 0x00 for 256 bytes and the result will be printed. Finally, compare the data that were written to the EEPROM and the data that were read from the EEPROM. If they are the same, the serial port will output "I2C-AT24C02 test passed!" and the three LEDs flashing, otherwise the serial port will output "Err:data read and write aren't matching." and all the three LEDs light.

The output information via the serial port is as following.

```
I2C-24C02 configured...

The I2C is hardware interface
The speed is 400K
AT24C02 writing...
0x00 0x01 0x02 0x03 0x04 0x05 0x06 0x07 0x08 0x09 0x0A 0x0B 0x0C 0x0D 0x0E 0x0F
0x10 0x11 0x12 0x13 0x14 0x15 0x16 0x17 0x18 0x19 0x1A 0x1B 0x1C 0x1D 0x1E 0x1F
0x20 0x21 0x22 0x23 0x24 0x25 0x26 0x27 0x28 0x29 0x2A 0x2B 0x2C 0x2D 0x2E 0x2F
0x30 0x31 0x32 0x33 0x34 0x35 0x36 0x37 0x38 0x39 0x3A 0x3B 0x3C 0x3D 0x3E 0x3F
0x40 0x41 0x42 0x43 0x44 0x45 0x46 0x47 0x48 0x49 0x4A 0x4B 0x4C 0x4D 0x4E 0x4F
0x50 0x51 0x52 0x53 0x54 0x55 0x56 0x57 0x58 0x59 0x5A 0x5B 0x5C 0x5D 0x5E 0x5F
0x60 0x61 0x62 0x63 0x64 0x65 0x66 0x67 0x68 0x69 0x6A 0x6B 0x6C 0x6D 0x6E 0x6F
0x70 0x71 0x72 0x73 0x74 0x75 0x76 0x77 0x78 0x79 0x7A 0x7B 0x7C 0x7D 0x7E 0x7F
0x80 0x81 0x82 0x83 0x84 0x85 0x86 0x87 0x88 0x89 0x8A 0x8B 0x8C 0x8D 0x8E 0x8F
0x90 0x91 0x92 0x93 0x94 0x95 0x96 0x97 0x98 0x99 0x9A 0x9B 0x9C 0x9D 0x9E 0x9F
0xA0 0xA1 0xA2 0xA3 0xA4 0xA5 0xA6 0xA7 0xA8 0xA9 0xAA 0xAB 0xAC 0xAD 0xAE 0xAF
0xB0 0xB1 0xB2 0xB3 0xB4 0xB5 0xB6 0xB7 0xB8 0xB9 0xBA 0xBB 0xBC 0xBD 0xBE 0xBF
0xC0 0xC1 0xC2 0xC3 0xC4 0xC5 0xC6 0xC7 0xC8 0xC9 0xCA 0xCB 0xCC 0xCD 0xCE 0xCF
0xD0 0xD1 0xD2 0xD3 0xD4 0xD5 0xD6 0xD7 0xD8 0xD9 0xDA 0xDB 0xDC 0xDD 0xDE 0xDF
0xE0 0xE1 0xE2 0xE3 0xE4 0xE5 0xE6 0xE7 0xE8 0xE9 0xEA 0xEB 0xEC 0xED 0xEE 0xEF
0xF0 0xF1 0xF2 0xF3 0xF4 0xF5 0xF6 0xF7 0xF8 0xF9 0xFA 0xFB 0xFC 0xFD 0xFE 0xFF
AT24C02 reading...
0x00 0x01 0x02 0x03 0x04 0x05 0x06 0x07 0x08 0x09 0x0A 0x0B 0x0C 0x0D 0x0E 0x0F
0x10 0x11 0x12 0x13 0x14 0x15 0x16 0x17 0x18 0x19 0x1A 0x1B 0x1C 0x1D 0x1E 0x1F
0x20 0x21 0x22 0x23 0x24 0x25 0x26 0x27 0x28 0x29 0x2A 0x2B 0x2C 0x2D 0x2E 0x2F
0x30 0x31 0x32 0x33 0x34 0x35 0x36 0x37 0x38 0x39 0x3A 0x3B 0x3C 0x3D 0x3E 0x3F
0x40 0x41 0x42 0x43 0x44 0x45 0x46 0x47 0x48 0x49 0x4A 0x4B 0x4C 0x4D 0x4E 0x4F
0x50 0x51 0x52 0x53 0x54 0x55 0x56 0x57 0x58 0x59 0x5A 0x5B 0x5C 0x5D 0x5E 0x5F
0x60 0x61 0x62 0x63 0x64 0x65 0x66 0x67 0x68 0x69 0x6A 0x6B 0x6C 0x6D 0x6E 0x6F
0x70 0x71 0x72 0x73 0x74 0x75 0x76 0x77 0x78 0x79 0x7A 0x7B 0x7C 0x7D 0x7E 0x7F
0x80 0x81 0x82 0x83 0x84 0x85 0x86 0x87 0x88 0x89 0x8A 0x8B 0x8C 0x8D 0x8E 0x8F
0x90 0x91 0x92 0x93 0x94 0x95 0x96 0x97 0x98 0x99 0x9A 0x9B 0x9C 0x9D 0x9E 0x9F
0xA0 0xA1 0xA2 0xA3 0xA4 0xA5 0xA6 0xA7 0xA8 0xA9 0xAA 0xAB 0xAC 0xAD 0xAE 0xAF
0xB0 0xB1 0xB2 0xB3 0xB4 0xB5 0xB6 0xB7 0xB8 0xB9 0xBA 0xBB 0xBC 0xBD 0xBE 0xBF
0xC0 0xC1 0xC2 0xC3 0xC4 0xC5 0xC6 0xC7 0xC8 0xC9 0xCA 0xCB 0xCC 0xCD 0xCE 0xCF
0xD0 0xD1 0xD2 0xD3 0xD4 0xD5 0xD6 0xD7 0xD8 0xD9 0xDA 0xDB 0xDC 0xDD 0xDE 0xDF
0xE0 0xE1 0xE2 0xE3 0xE4 0xE5 0xE6 0xE7 0xE8 0xE9 0xEA 0xEB 0xEC 0xED 0xEE 0xEF
0xF0 0xF1 0xF2 0xF3 0xF4 0xF5 0xF6 0xF7 0xF8 0xF9 0xFA 0xFB 0xFC 0xFD 0xFE 0xFF
I2C-AT24C02 test passed!
```

## 5.9. HPDF\_I2S\_Audio

### 5.9.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

- Learn to use I2S module to output audio data
- Learn to use HPDF interface to process PDM data

The GD32W515P-EVAL-V1.1 board integrates HPDF and I2S modules. The two modules can cooperate with each other to play the audio signal of the microphone. This example demonstrates the process of collecting single-channel audio data through the HPDF of the EVAL board and playing dual-channel audio using the I2S interface.

### 5.9.2. DEMO running result

Download the program <09\_HPDI2S\_Audio> to the development board and run, plug in the headset to hear the sound from the microphone.

## 5.10. SPI\_LCD

### 5.10.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

- Learn how to use SPI to drive TFT LCD screen and display

GD32W515P-EVAL-V1.1 board has a TFT LCD screen which supports SPI interface. In this demo, tests of font, number, draw and color are displayed on the LCD screen respectively.

### 5.10.2. DEMO running result

LCD is controlled by SPI module on GD32W515P-EVAL-V1.1 board. Firstly, JP13 to JP15 must be fitted to LCD port and then download the program <10\_SPI\_LCD > to the EVAL board. All the LEDs are turned on and then turned off for test. After that, the LCD screen on the board will display the GUI tests in infinite loop.



## 5.11. SDIO\_SDCard

### 5.11.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

- Learn to use SDIO to single block or multiple block write and read
- Learn to use SDIO to erase, lock and unlock a SD card

GD32W515P-EVAL-V1.1 board has a secure digital input/output interface (SDIO) which

defines the SD/SD I/O /MMC CE-ATA card host interface. This demo will show how to use SDIO to operate on SD card.

### 5.11.2. DEMO running result

Download the program <11\_SDIO\_SDCardTest> to the EVAL board and run. Connect serial cable to USART, open the HyperTerminal. Firstly, all the LEDs are turned on and off for test. Then initialize the card and print out the information of the card. After that, test the function of single block operation, lock and unlock operation, erase operation and multiple blocks operation. If any error occurs, print the error message and turn on LED1, LED3 and turn off LED2. Otherwise, turn on all the LEDs.

Uncomment the macro DATA\_PRINT to print out the data and display them through HyperTerminal. Set bus mode(1-bit or 4-bit) and data transfer mode(polling mode or DMA mode) by comment and uncomment the related statements.

Information via a serial port output as following.

```
Card init success!

Card information:
## Card version 3.0x ##
## Device size is 15122432KB ##
## Block size is 512B ##
## Block count is 30244864 ##
## CardCommandClasses is: 5b5 ##
## Block operation supported ##
## Erase supported ##
## Lock unlock supported ##
## Application specific supported ##
## Switch function supported ##

Card test:
Block write success!
Block read success!
The card is locked!
Erase failed!
The card is unlocked!
Erase success!
Block read success!

Multiple block write success!
Multiple block read success!
```

## 5.12. TRNG\_Get\_Random

### 5.12.1. DEMO purpose

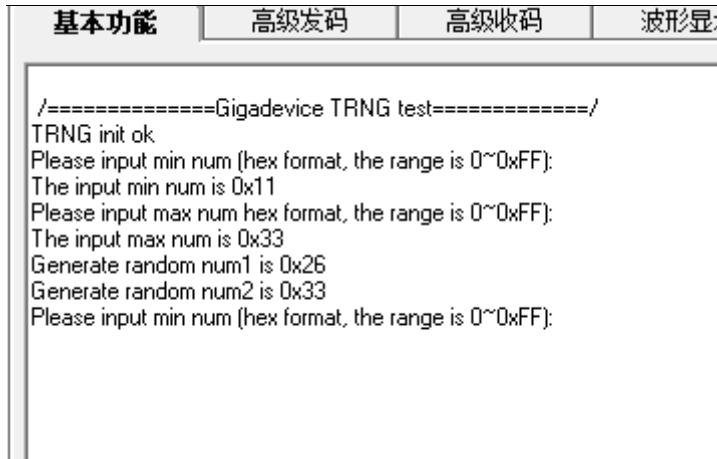
This demo includes the following functions of GD32 MCU:

- Learn to use TRNG generate the random number
- Learn to communicate with PC by USART

### 5.12.2. DEMO running result

Download the program <12\_TRNG\_Get\_Random> to the EVAL board and run. Connect serial cable to EVAL\_COM, open the serial terminal tool supporting hex format communication. When the program is running, the serial terminal tool will display the initial information. User can use the serial terminal tool to input the minimum and maximum values (for example, the minimum value is 0x011, the maximum value is 0x33), then application will generate random number in the input range and display it by the serial terminal tool.

Information via a serial port output as following:



```

=====Gigadevice TRNG test=====
TRNG init ok
Please input min num (hex format, the range is 0~0xFF):
The input min num is 0x11
Please input max num hex format, the range is 0~0xFF):
The input max num is 0x33
Generate random num1 is 0x26
Generate random num2 is 0x33
Please input min num (hex format, the range is 0~0xFF):

```

## 5.13. CAU

### 5.13.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

- Learn DES, Triple-DES and AES algorithm
- Learn Electronic codebook (ECB) mode, Cipher block chaining (CBC) mode, Counter (CTR) mode, Galois/counter (GCM) mode, combined cipher machine (CCM) mode, Cipher Feedback (CFB) mode, and Output Feedback (OFB) mode
- Learn to use CAU to encrypt and decrypt
- Learn to communicate with PC by USART

### 5.13.2. DEMO running result

Download the program <13\_CAU> to the EVAL board and run. Connect USB cable to CN5. JP21 must be jumped to USART. When the program is running, the serial terminal tool will display the information, as shown in the following figure. Plaintext data value, the encryption algorithm, and the mode can be selected are shown. After the user setting the algorithm and mode according to the serial output information indicating, serial port will print out selected algorithm and mode, as shown below.

```

Plain data :
0x6B 0xC1 0xBE 0xE2 0x2E 0x40 0x9F 0x96 0xE9 0x3D 0x7E 0x11 0x73 0x93 0x17 0x2A [Block 0]
0xAE 0x2D 0x8A 0x57 0x1E 0x03 0xAC 0x9C 0x9E 0xB7 0x6F 0xAC 0x45 0xAF 0x8E 0x51 [Block 1]
0x30 0xC8 0x1C 0x46 0xA3 0x5C 0xE4 0x11 0xE5 0xFB 0xC1 0x19 0x1A 0x0A 0x52 0xEF [Block 2]
0xF6 0x9F 0x24 0x45 0xDF 0x4F 0x9B 0x17 0xAD 0x2B 0x41 0x7B 0xE6 0x6C 0x37 0x10 [Block 3]
=====Choose CAU algorithm=====
1: DES algorithm
2: TDES algorithm
3: AES algorithm

You choose to use DES algorithm
=====Choose CAU mode=====
1: ECB mode
2: CBC mode
3: CTR mode only when choose AES algorithm
4: GCM mode only when choose AES algorithm
5: CCM mode only when choose AES algorithm
6: CFB mode only when choose AES algorithm
7: OFB mode only when choose AES algorithm

```

You choose to use ECB mode

After selection, the program starts encryption and decryption operations, the results are printed through the serial port.

Encrypted data with DES Mode ECB :

```

0x6E 0xDF 0xD1 0xB7 0xA0 0x01 0xCD 0x17 0xCD 0xC5 0x7F 0xF7 0x9C 0xF8 0x72 0xD0 [Block 0]
0x11 0x97 0xA6 0xD2 0x13 0x59 0x4F 0x7A 0x3D 0x7C 0x7C 0xEC 0xBC 0xDD 0xD2 0x20 [Block 1]
0x3A 0x75 0x8B 0x06 0x75 0x2E 0x18 0x0D 0x55 0x0F 0xDD 0x57 0x5A 0xF1 0x3B 0x94 [Block 2]
0x18 0x3D 0x4D 0xA1 0x1E 0x14 0x75 0x6B 0x0F 0xD9 0xD9 0x64 0x16 0xA0 0x60 0x14 [Block 3]

```

Decrypted data with DES Mode ECB :

```

0x6B 0xC1 0xBE 0xE2 0x2E 0x40 0x9F 0x96 0xE9 0x3D 0x7E 0x11 0x73 0x93 0x17 0x2A [Block 0]
0xAE 0x2D 0x8A 0x57 0x1E 0x03 0xAC 0x9C 0x9E 0xB7 0x6F 0xAC 0x45 0xAF 0x8E 0x51 [Block 1]
0x30 0xC8 0x1C 0x46 0xA3 0x5C 0xE4 0x11 0xE5 0xFB 0xC1 0x19 0x1A 0x0A 0x52 0xEF [Block 2]
0xF6 0x9F 0x24 0x45 0xDF 0x4F 0x9B 0x17 0xAD 0x2B 0x41 0x7B 0xE6 0x6C 0x37 0x10 [Block 3]

```

Example restarted...

And then restart for users to select a different algorithm and mode to repeat demo, as shown below.

```

Plain data :
0x6B 0xC1 0xBE 0xE2 0x2E 0x40 0x9F 0x96 0xE9 0x3D 0x7E 0x11 0x73 0x93 0x17 0x2A [Block 0]
0xAE 0x2D 0x8A 0x57 0x1E 0x03 0xAC 0x9C 0x9E 0xB7 0x6F 0xAC 0x45 0xAF 0x8E 0x51 [Block 1]
0x30 0xC8 0x1C 0x46 0xA3 0x5C 0xE4 0x11 0xE5 0xFB 0xC1 0x19 0x1A 0x0A 0x52 0xEF [Block 2]
0xF6 0x9F 0x24 0x45 0xDF 0x4F 0x9B 0x17 0xAD 0x2B 0x41 0x7B 0xE6 0x6C 0x37 0x10 [Block 3]
=====Choose CAU algorithm=====
1: DES algorithm
2: TDES algorithm
3: AES algorithm

```



## 5.14. HAU

### 5.14.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

- Learn SHA-1, SHA-224, SHA-256 and MD5 algorithm
- Learn HASH mode and HMAC (keyed-hash message authentication code) mode
- Learn to use HAU to calculate digest for the input message
- Learn to communicate with PC by USART

### 5.14.2. DEMO running result

Download the program <14\_HAU> to the EVAL board and run. Connect USB cable to CN5. JP21 must be jumped to USART. When the program is running, the serial terminal tool will display the information, as shown in the following figure. After the user setting the algorithm and mode according to the serial output information indicating, serial port will print out selected algorithm and mode, as shown below.

```
message to be hashed:
```

```
The hash processor is a fully compliant implementation of the secure
hash algorithm (SHA-1), the MD5 (message-digest algorithm 5) hash
algorithm and the HMAC (keyed-hash message authentication code)
algorithm suitable for a variety of applications. =====Choose HAU
algorithm=====
1: SHA1 algorithm
2: SHA224 algorithm
3: SHA256 algorithm
4: MD5 algorithm
```

```
You choose to use SHA1 algorithm
=====Choose HAU mode=====
1: HASH mode
2: HMAC mode
```

```
You choose to use HASH mode
```

After selection, the program starts digest calculation, the results are printed through the serial port. And then restart for users to select a different algorithm and mode to repeat demo, as shown below.

```
message digest with SHA-1 Mode HASH (160 bits):
```

```
0x08 0x54 0x77 0x5E
0xC2 0xA1 0x0C 0x7F
0x4B 0x80 0x37 0xD9
0xE7 0x7C 0xA7 0x30
0xF0 0x5D 0xFA 0x2E
```

```
Example restarted...
```

```
message to be hashed:
```

```
The hash processor is a fully compliant implementation of the secure
hash algorithm (SHA-1), the MD5 (message-digest algorithm 5) hash
algorithm and the HMAC (keyed-hash message authentication code)
algorithm suitable for a variety of applications. =====Choose HAU
algorithm=====
1: SHA1 algorithm
2: SHA224 algorithm
3: SHA256 algorithm
4: MD5 algorithm
```

## 5.15. PKCAU\_Modular\_Addition\_Interrupt

### 5.15.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

- Learn modular addition algorithm by interrupt

### 5.15.2. DEMO running result

Download the program < 15\_PKCAU\_Modular\_Addition\_Interrupt > to the EVAL board. After system start-up, firstly, wait for the PKCAU busy flag to be reset, then, perform modular addition computation, when the computation is completed, an interrupt will be generated, finally, read results from specific PKCAU RAM address and compare with the expected result, if the result is the same with the expected one, LED1 will on, or else, LED2 is on.

## 5.16. RCU\_Clock\_Out

### 5.16.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

- Learn to use GPIO control the LED
- Learn to use the clock output function of RCU
- Learn to communicate with PC by USART

### 5.16.2. DEMO running result

Download the program <16\_RCU\_Clock\_Out> to the EVAL board and run. Connect serial cable to USART, open the HyperTerminal. When the program is running, HyperTerminal will display the initial information. Then user can choose the type of the output clock by pressing the TAMPER/WAKEUP key. After pressing, the corresponding LED will be turned on and HyperTerminal will display which mode be selected. The frequency of the output clock can be observed through the oscilloscope by PA8 pin.

Information via a serial port output as following:

```
/===== Gigadevice Clock output Demo =====/  
press tamper/wakeup key to select clock output source  
CK_OUTO: PLLP clock  
CK_OUTO: IRC8M  
CK_OUTO: HXTAL
```

## 5.17. PMU\_Sleep\_Wakeup

### 5.17.1. DEMO purpose

This Demo includes the following functions of GD32 MCU:

- Learn to use the USART receive interrupt to wake up the PMU from sleep mode

### 5.17.2. DEMO running result

Download the program < 17\_PMU\_sleep\_wakeup > to the EVAL board. Connect USB cable to CN5. JP21 must be jumped to USART. After power-on, all the LEDs are off. The MCU will enter sleep mode and the software stop running. When the USART2 receives a byte of data from the HyperTerminal, the MCU will wake up from a receive interrupt. And all the LEDs will flash together.

## 5.18. RTC\_Calendar

### 5.18.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

- Learn to use RTC module to implement calendar function
- Learn to use USART module to implement time display

### 5.18.2. DEMO running result

Download the program <18\_RTC\_Calender> to the EVAL board and run. Connect serial

cable to USART, open the HyperTerminal. After start-up, the program will ask to set the time on the HyperTerminal. The calendar will be displayed on the HyperTerminal.

```
***** RTC calendar demo *****  
  
=====Configure RTC Time=====  
  
please input hour:  
  
10  
  
please input minute:  
  
12  
  
please input second:  
  
14  
  
** RTC time configuration success! **  
  
Current time: 10:12:14
```

## 5.19. TSI

This Demo includes the following functions of GD32 MCU:

- Learn to use TSI module implement Touch Key function

### 5.19.1. DEMO running result

Download the program <19\_TSI> to the EVAL board and run. When the program is running, you can use a finger touch the TSI Sensor on the EVAL board, and then the associated LED1 and LED2 will light.

## 5.20. IFRP

### 5.20.1. DEMO purpose

This Demo includes the following functions of GD32 MCU:

- Learn to use general timer output PWM wave
- Learn to use general timer generated update interrupt
- Learn to use general timer capture interrupt
- Learn to use general timer TIMER15 and TIMER16 implement Infrared function

### 5.20.2. DEMO running result

Download the program <20\_IFRP> to the EVAL board and run. When the program is running, if the infrared receiver received data is correct, LED1, LED2, LED3 light in turn, otherwise LED1, LED2, LED3 toggle together.

## 5.21. TIMER\_Breath\_LED

### 5.21.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

- Learn to use TIMER output PWM wave
- Learn to update channel value

### 5.21.2. DEMO running result

Use the DuPont line to connect the TIMER0\_CH0 (PA8) and LED (PB6). Then download the program <21\_TIMER\_Breath\_LED> to the EVAL board and run. PA8 should not be reused by other peripherals.

When the program is running, you can see LED lighting from dark to bright gradually and then gradually darken, ad infinitum, just like breathing as rhythm.

## 5.22. USBFS\_Device

### 5.22.1. HID\_Keyboard

#### DEMO Purpose

This demo includes the following functions of GD32 MCU:

- Learn how to use the USBFS peripheral mode
- Learn how to implement USB HID(human interface) device

The GD32W515P-EVAL board has two keys and one USB\_FS interface. The two keys are Reset key and Tamper/Wakeup key. The GD32W515P-EVAL board is enumerated as an USB Keyboard, which uses the native PC Host HID driver, as shown below. The USB Keyboard uses the Tamper/wakeup key to output the characters 'a'. In addition, the demo also supports remote wakeup which is the ability of a USB device to bring a suspended bus back to the active condition, and the Tamper/Wakeup key is used as the remote wakeup source.



## DEMO Running Result

Download the program <22\_USBFS\USB\_Device\HID\_Keyboard> to the EVAL board and run. If you press the Tamper/Wakeup key, The USB Keyboard will output 'a'.

If you want to test USB remote wakeup function, you can do as follows:

- Manually switch PC to standby mode
- Wait for PC to fully enter the standby mode
- Push the Tamper/Wakeup key
- If PC is ON, remote wakeup is OK, else failed.

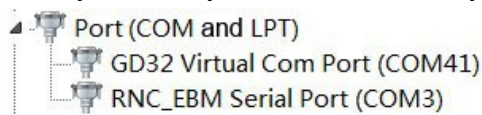
### 5.22.2. CDC\_ACM

## DEMO Purpose

This demo includes the following functions of GD32 MCU:


- Learn how to use the USBFS
- Learn how to implement USB CDC device

The GD32W515P-EVAL board has one USBFS interface. In this demo, the GD32W515P-EVAL-V1.1 board is enumerated as an USB virtual COM port, which was shown in device manager of PC as below. This demo makes the USB device look like a serial port, and loops back the contents of a text file over USB port. To run the demo, input a message using the PC's keyboard. Any data that shows in HyperTerminal is received from the device.



## DEMO Running Result

Download the program <22\_USBFS\USB\_Device\CDC\_ACM> to the EVAL board and run. When you input message through computer keyboard, the HyperTerminal will receive and shown the message. For example, when input "GigaDevice MCU", the HyperTerminal will get and show it as below.



GigaDevice MCU

## 5.23. USBFS\_Host

### 5.23.1. HID\_Host

#### DEMO Purpose

This demo includes the following functions of GD32 MCU:

- Learn to use the USBFS as a HID host
- Learn the operation between the HID host and the mouse device
- Learn the operation between the HID host and the keyboard device

The GD32W515P-EVAL board integrates the USBFS module, and the module can be used as a USB device, a USB host or an OTG device. This demo mainly shows how to use the USBFS as a USB HID host to communicate with external USB HID device.

#### DEMO Running Result

Download the program <22\_USBFS\USB\_Host\Host\_HID> to the EVAL board and run.

If a mouse has been attached, the user will see the information of mouse enumeration. First pressing the Tamper/Wakeup key will see the inserted device is mouse, and then moving the mouse will print the coordinates on the serial port assistant.

If a keyboard has been attached, the user will see the information of keyboard enumeration. First pressing the Tamper/Wakeup key will see the inserted device is keyboard, and then pressing the keyboard will show the state of the button in the serial port assistant.

### 5.23.2. MSC

#### DEMO Purpose

This demo includes the following functions of GD32 MCU:

- Learn to use the USBFS as a MSC host
- Learn the operation between the MSC host and the Udisk

GD32W515P-EVAL-V1.1 board integrates the USBFS module, and the module can be used as a USB device, a USB host or an OTG device. This demo mainly shows how to use the USBFS as a USB MSC host to communicate with external Udisk.

## DEMO Running Result

Download the program <22\_USBFS\USB\_Host\Host\_MSC> to the EVAL board and run.

If an Udisk has been attached, the user will see the information of Udisk enumeration. First pressing the Tamper/Wakeup key will see the Udisk information, next pressing the Tamper/Wakeup key will see the root content of the Udisk, then press the Tamper/Wakeup key will write file to the Udisk, finally the user will see information that the MSC host demo is end.

## 5.24. Trustzone

### 5.24.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

- Learn to use MCU when TZEN = 1
- Learn to use SAU/IDAU to configure NSC and NS address area
- Learn to use option bytes to configure secure mark pages
- Learn to use code to enable Ttrustzone
- Learn to use TZPCU to configure non-secure SRAM area
- Learn to configure GPIO to non-secure
- Learn to use TZPCU to configure USART to secure
- Learn how secure code jump to non-secure code
- Learn how secure code call non-secure code function
- Learn how non-secure code call secure code function by non-secure callable function

### 5.24.2. DEMO running result

Download the program < 23\_Trustzone> to the EVAL board and run. LED1 and LED2 can light cycles. And HyperTerminal will print “secure code print: secure code toggle LED1.” and “non-secure code print: non-secure code toggle LED2.” cycles.



## 6. Revision history

**Table 6-1. Revision history**

Revision No.	Description	Date
1.0	Initial Release	Jan.21, 2021

## 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 its 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 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 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.