

GigaDevice Semiconductor Inc.

GD32F5HCR-EVAL

Arm[®] Cortex[®]-M33 32-bit MCU

User Guide

Revision 1.0

(Apr. 2026)

Table of Contents

Table of Contents	1
List of Figures	4
List of Tables	5
1. Summary	6
2. Function Pin Assign	7
3. Getting started	8
4. Hardware layout overview.....	9
4.1. Power supply	9
4.2. Boot option	9
4.3. LED	9
4.4. KEY	10
4.5. ADC	10
4.6. IFRP	11
4.7. USART	11
4.8. I2C.....	12
4.9. SPI	12
4.10. I2S	13
4.11. Extension.....	13
4.12. GD-Link.....	13
4.13. USB	14
4.14. MCU.....	15
5. Routine use guide.....	16
5.1. GPIO_Runing_Led.....	16
5.1.1. DEMO Purpose.....	16
5.1.2. DEMO Running Result	16
5.2. GPIO_Key_Polling_mode.....	16
5.2.1. DEMO Purpose.....	16
5.2.2. DEMO Running Result	16
5.3. EXTI_Key_Interrupt_mode.....	17
5.3.1. DEMO Purpose.....	17
5.3.2. DEMO Running Result	17
5.4. USART_Printf.....	17
5.4.1. DEMO Purpose.....	17
5.4.2. DEMO Running Result	17
5.5. USART_Echo_Interrupt_mode.....	18
5.5.1. DEMO Purpose.....	18
5.5.2. DEMO Running Result	18

5.6. USART_DMA	18
5.6.1. DEMO Purpose	18
5.6.2. DEMO Running Result	19
5.7. ADC_conversion_triggered_by_timer	19
5.7.1. DEMO Purpose	19
5.7.2. DEMO Running Result	19
5.8. I2C_EEPROM	20
5.8.1. DEMO Purpose	20
5.8.2. DEMO Running Result	20
5.9. CTC_Calibration	21
5.9.1. DEMO Purpose	21
5.9.2. DEMO Running Result	21
5.10. SPI_LCD	21
5.10.1. DEMO Purpose	21
5.10.2. DEMO Running Result	21
5.11. TRNG_Get_Random	22
5.11.1. DEMO purpose	22
5.11.2. DEMO running result	22
5.12. CAU	23
5.12.1. DEMO Purpose	23
5.12.2. DEMO Running Result	23
5.13. HAU	24
5.13.1. DEMO purpose	24
5.13.2. DEMO running result	24
5.14. PKCAU_Modular_Addition_Interrupt	25
5.14.1. DEMO Purpose	25
5.14.2. DEMO Running Result	25
5.15. RCU_Clock_Out	26
5.15.1. DEMO Purpose	26
5.15.2. DEMO Running Result	26
5.16. PMU_sleep_wakeup	26
5.16.1. DEMO Purpose	26
5.16.2. DEMO Running Result	26
5.17. RTC_Calendar	27
5.17.1. DEMO Purpose	27
5.17.2. DEMO running result	27
5.18. IFRP	27
5.18.1. DEMO Purpose	27
5.18.2. DEMO Running Result	28
5.19. TIMER_Breath_LED	28
5.19.1. DEMO Purpose	28
5.19.2. DEMO Running Result	28
5.20. USB_Device	28

5.20.1.	HID_Keyboard	28
5.20.2.	CDC_ACM	29
5.21.	USB_Host	30
5.21.1.	HID_Host	30
5.21.2.	MSC_Host.....	30
5.22.	Trustzone	31
5.22.1.	DEMO Purpose.....	31
5.22.2.	DEMO Running Result	31
6.	Revision history	32

List of Figures

Figure 4-1. Schematic diagram of power supply.....	9
Figure 4-2. Schematic diagram of boot option	9
Figure 4-3. Schematic diagram of LED function	9
Figure 4-4. Schematic diagram of Key function	10
Figure 4-5. Schematic diagram of ADC	10
Figure 4-6. Schematic diagram of DAC	11
Figure 4-7. Schematic diagram of USART	11
Figure 4-8. Schematic diagram of I2C	12
Figure 4-9. Schematic diagram of SPI	12
Figure 4-10. Schematic diagram of I2S	13
Figure 4-11. Schematic diagram of Extension	13
Figure 4-12. Schematic diagram of GD-Link.....	13
Figure 4-13. Schematic diagram of USB	14
Figure 4-14. Schematic diagram of MCU.....	15

List of Tables

Table 2-1. Function pin assignment.....	7
Table 6-1. Revision history	32

1. Summary

GD32F5HCR-EVAL uses GD32F5HCRIH6 as the main controller. It uses GD-Link Type-C interface to supply 5V power. SWD, Reset, Boot, User button key, LED, I2C, I2S, USART, LCD, SPI, QSPI, ADC, USB, GD-Link and Extension Pins are also included. For more details, please refer to GD32F5HCR-EVAL-V1.0 schematic.

2. Function Pin Assign

Table 2-1. Function pin assignment

Function	Pin	Description
LED	PB6	LED1
	PA15	LED2
	PA6	LED3
	PA1	LED4
RESET	NRST	K1-Reset
KEY	PA2	K2-Wakeup & Tamper
	PA3	K3- User key2
	PD2	K4-User key1
USART2	PB10	USART2_TX
	PB11	USART2_RX
ADC	PA0	ADC_IN0
	PC1	ADC_IN5
I2C	PB15	I2C1_SCL
	PA8	I2C1_SDA
SPI_LCD	PB1	LCD_RESET
	PB2	LCD_D/C
	PC3	SPILCD_CS
	PC11	SPI_SCK
	PC13	SPI_MISO
	PD0	SPI_MOSI
I2S	PA7	I2S_WS
	PB8	I2S_CK
	PA4	I2S_SD
	PA5	I2S_MCK
QSPI	PA10	QSPI_CSN
	PA9	QSPI_SCK
	PA11	QSPI_IO0
	PA12	QSPI_IO1
	PB3	QSPI_IO2
	PB4	QSPI_IO3
IFRP	PB5	IR_OUT
USB	PB13	USBFS_DM
	PB12	USBFS_DP

3. Getting started

The EVAL board uses GD-Link Type-C interface to get power DC +5V, which is the hardware system normal work voltage. A J-Link tool or 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 that the power supply is OK.

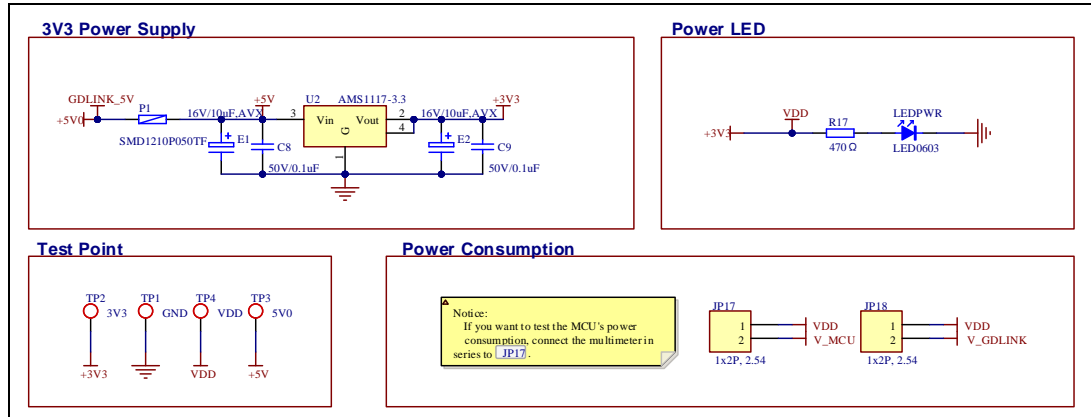
There are Keil version, IAR version and GD32EBuilder version of all projects. Keil version of the projects are created based on Keil MDK-ARM 5.29 uVision5. IAR version of the projects are created based on IAR Embedded Workbench for ARM 8.32.1 and GD32EBuilder version of the projects are created based on GD32EmbeddedBuilder_v1.5.5_Rel. 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, the latest version of GigaDevice.GD32W51x_F5HC_DFP (URL: <https://www.gd32mcu.com>) should be installed to load related files.
2. If you use IAR to open the project, the latest version of IAR_GD32W51x_F5HC _ADDON (URL: <https://www.gd32mcu.com>) should be installed to load related files.

4. Hardware layout overview

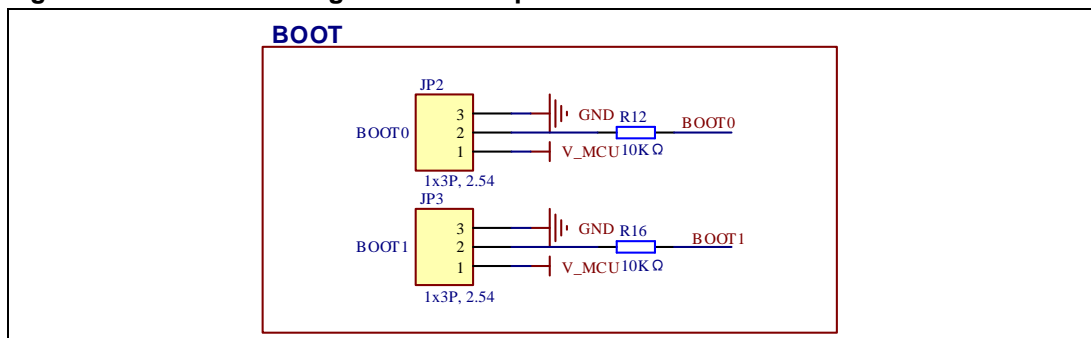
4.1. Power supply

Figure 4-1. Schematic diagram of power supply



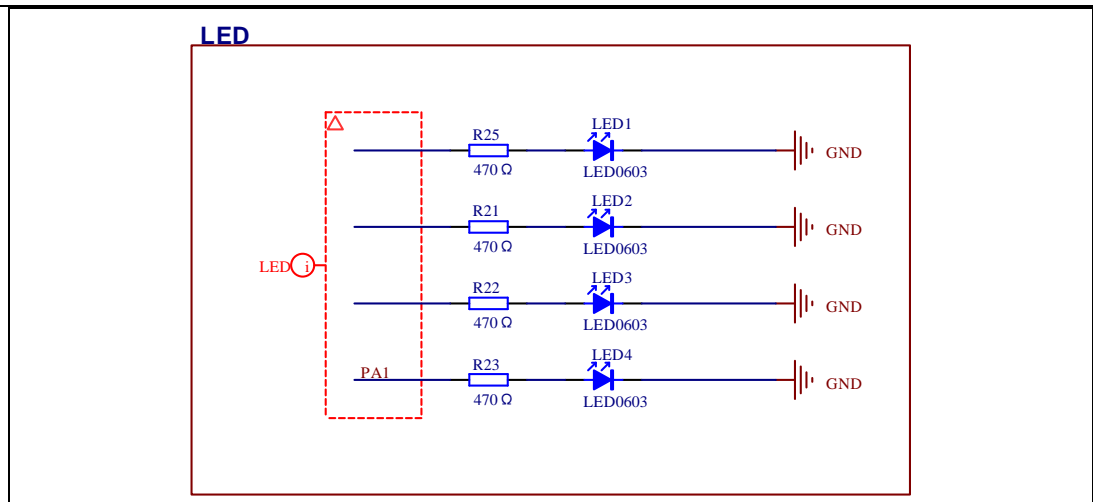
4.2. Boot option

Figure 4-2. Schematic diagram of boot option



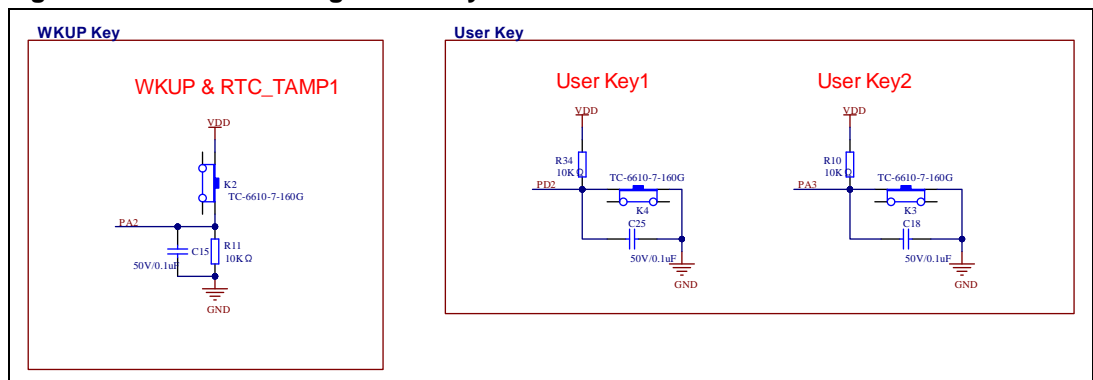
4.3. LED

Figure 4-3. Schematic diagram of LED function



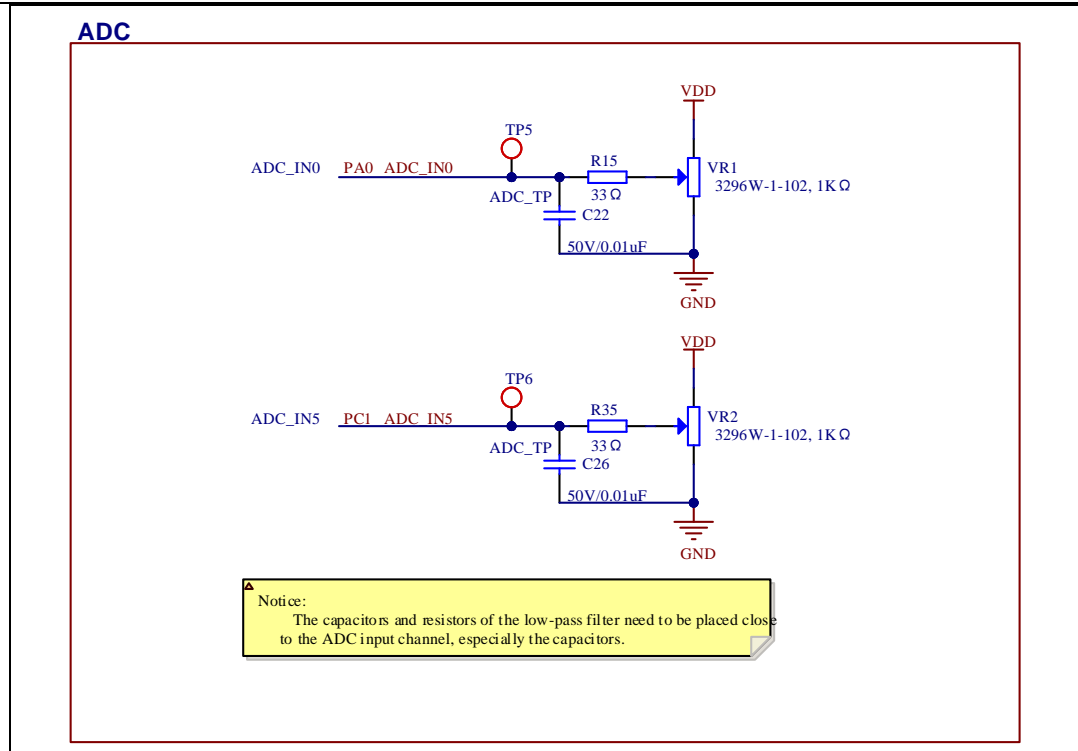
4.4. KEY

Figure 4-4. Schematic diagram of Key function



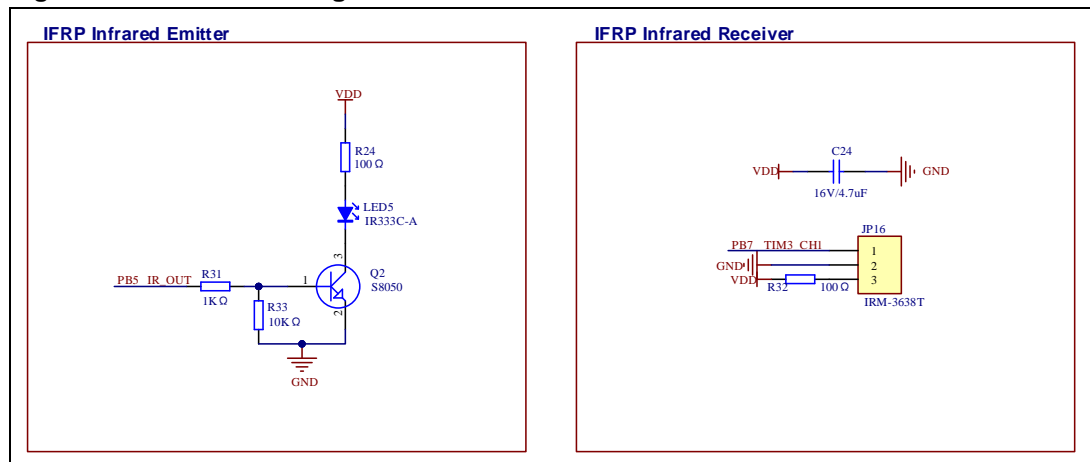
4.5. ADC

Figure 4-5. Schematic diagram of ADC



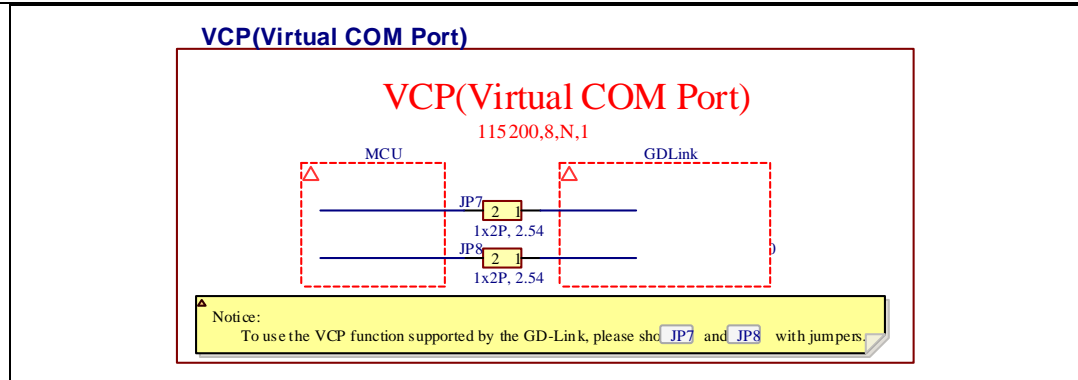
4.6. IFRP

Figure 4-6. Schematic diagram of DAC



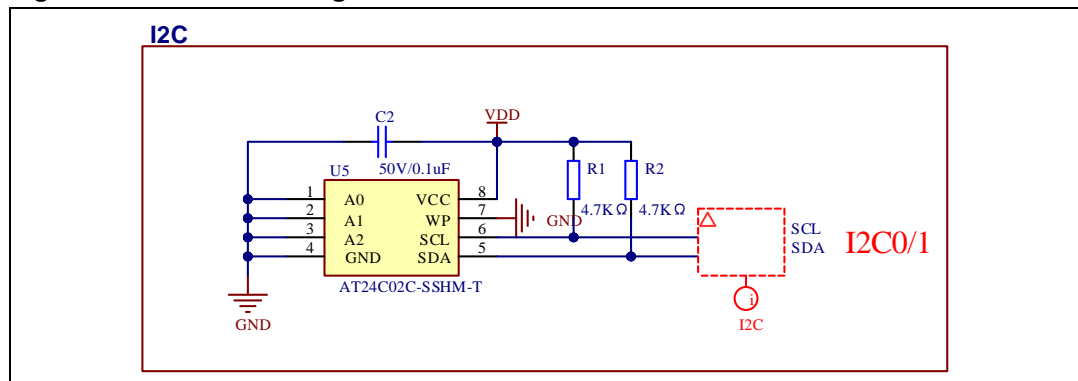
4.7. USART

Figure 4-7. Schematic diagram of USART



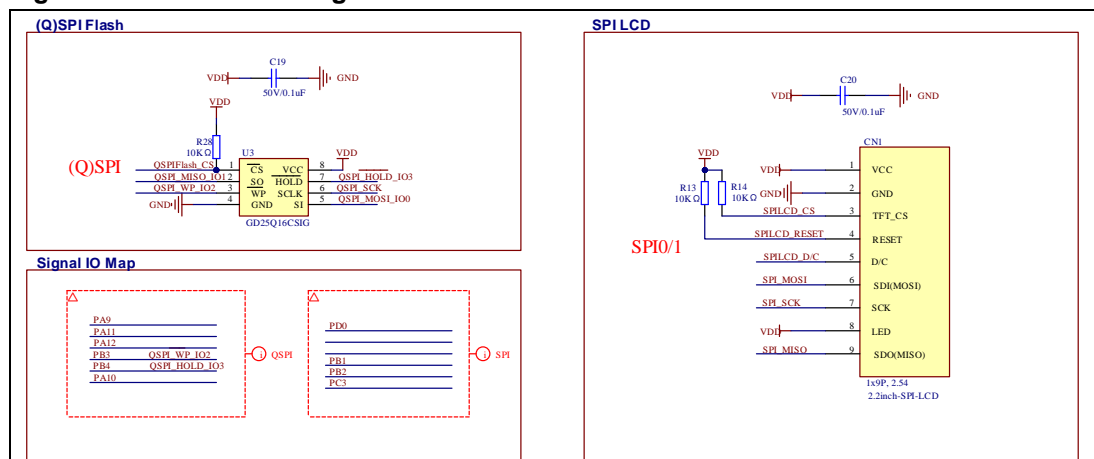
4.8. I2C

Figure 4-8. Schematic diagram of I2C



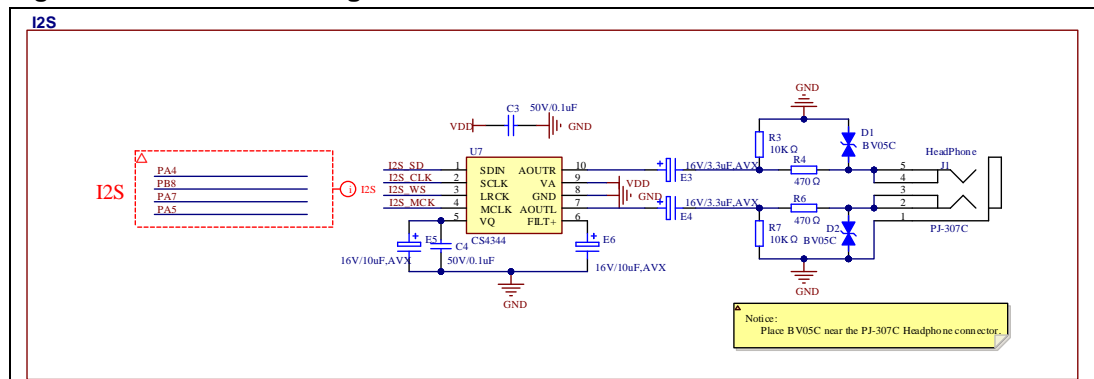
4.9. SPI

Figure 4-9. Schematic diagram of SPI



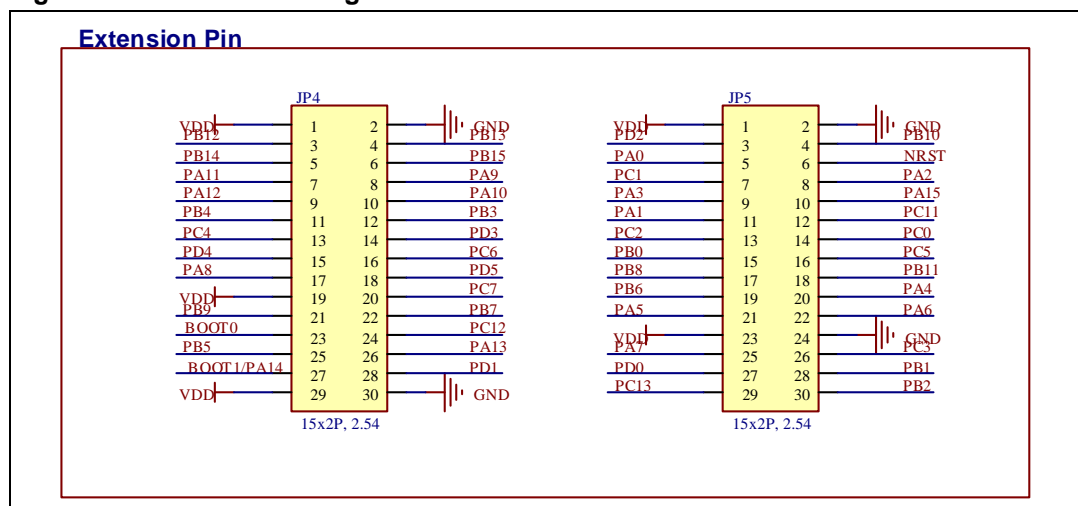
4.10. I2S

Figure 4-10. Schematic diagram of I2S



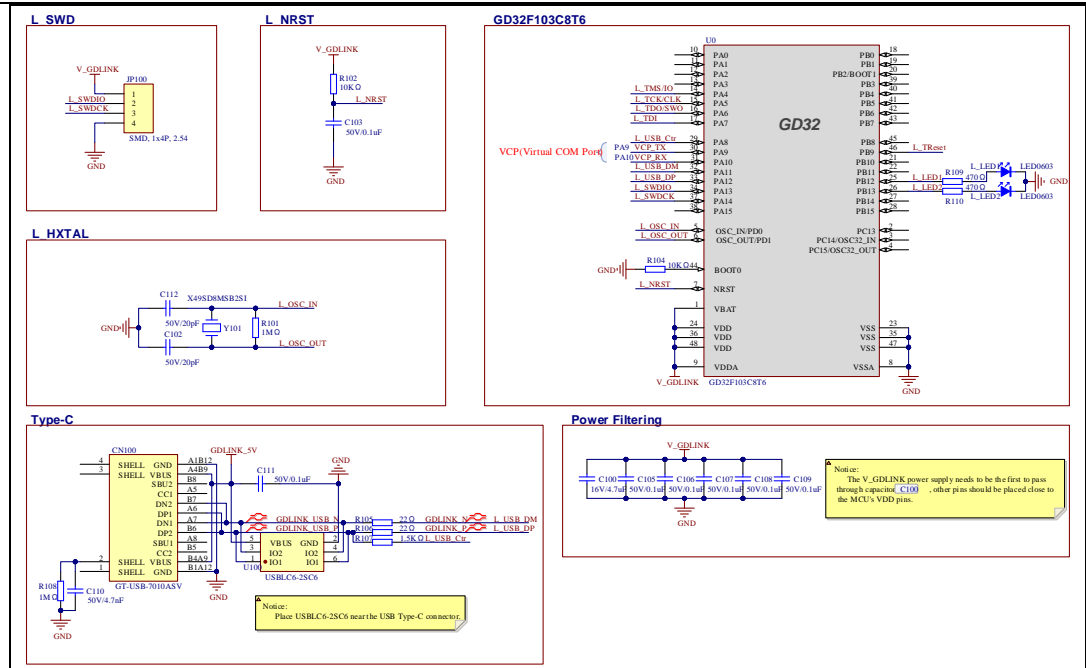
4.11. Extension

Figure 4-11. Schematic diagram of Extension



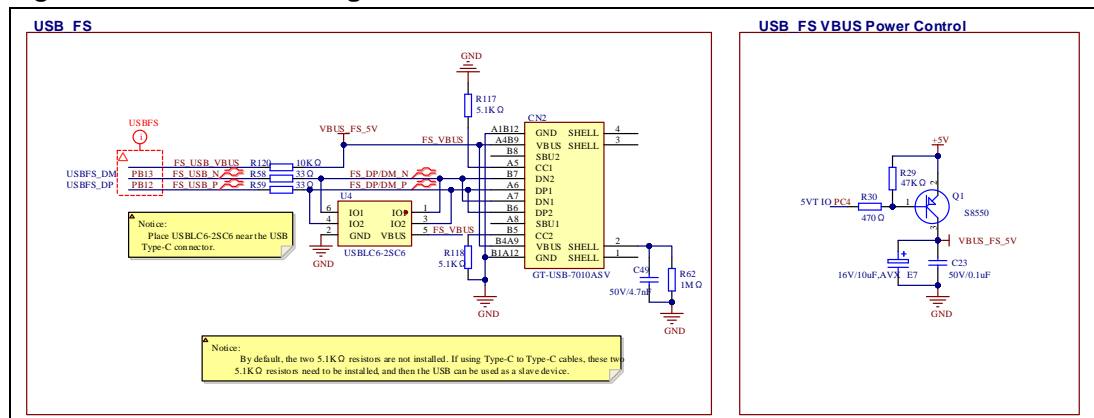
4.12. GD-Link

Figure 4-12. Schematic diagram of GD-Link



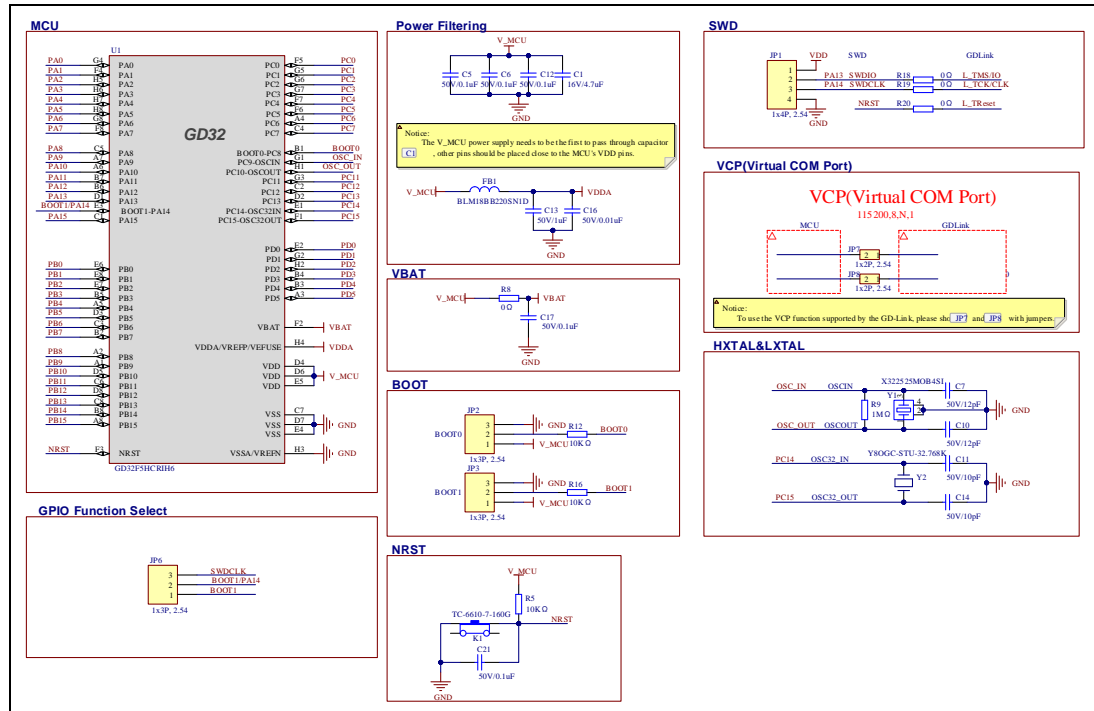
4.13. USB

Figure 4-13. Schematic diagram of USB



4.14. MCU

Figure 4-14. Schematic diagram of MCU



5. Routine use guide

5.1. GPIO_Runing_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

GD32F5HCR-EVAL-V1.0 board has four LEDs. The LED1, LED2, LED3 and LED4 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 START board, LED1, LED2, LED3 and LED4 turn on in sequence for 500ms and then turn off, after which the sequence repeats.

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

GD32F5HCR-EVAL-V1.0 board has four keys and four LEDs. The four keys are Reset key, Wakeup key, UserKey1 key and UserKey2 key. The LED1, LED2, LED3, LED4 are controlled by GPIO.

This demo will show how to use the UserKey2 key to control the LED2. When press down the UserKey2 key, 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 UserKey2 Key, LED2 will be turned on. Press down the Tamper Key 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

GD32F5HCR-EVAL-V1.0 board has four keys and four LEDs. The four keys are Reset key, Wakeup key, UserKey1 key and UserKey2 key. The LED1, LED2, LED3, LED4 are controlled by GPIO.

This demo will show how to use the EXTI interrupt line to control the LED2. When press down the UserKey2 Key, 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 UserKey2 Key, LED2 will be turned on. Press down the UserKey2 Key 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, HyperTerminal outputs "USART printf example: please press the Tamper key" on the HyperTerminal using USART. Press the Tamper key, serial port will output "USART printf example" and LED1 is turned on, otherwise, LED1 turn off.

The output information via the serial port is as following.

USART printf example: please press the Tamper 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 serial terminal tool

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 USART sends the tx_buffer array (from 0x00 to 0xFF) to the serial terminal tool supporting hex format communication and waits for receiving data of BUFFER_SIZE bytes from the serial terminal. The data MCU has received is stored in the rx_buffer array. After that, compare tx_buffer with rx_buffer. If tx_buffer is same with rx_buffer, LED1, LED2, LED3, LED4 turns on. Otherwise, LED1, LED2, LED3, LED4 toggle together.

The output information via the serial port 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 turned on and off for test. Then, the USART sends the tx_buffer array (from 0x00 to 0xFF) to the serial terminal tool supporting hex format communication and waits for receiving data of same bytes as tx_buffer from the serial terminal. The data MCU have received is stored in the rx_buffer array. After that, compare tx_buffer with rx_buffer. If tx_buffer is same with rx_buffer, LED1, LED2, LED3, LED4 turns on. Otherwise, LED1, LED2, LED3, LED4 toggle together.

```
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 GD32F5HCR-EVAL-V1.0 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 <12_I2C_EEPROM> to the EVAL board and run. Connect serial cable to USART, and 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 four LEDs lights by turns, otherwise the serial port will output "Err: data read and write aren't matching." and all the four LEDs light.

The output information via the serial port is as following.

```
I2C-24C02 configured...

The I2C is hardware interface
The speed is 400000
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. CTC_Calibration

5.9.1. DEMO Purpose

This demo includes the following functions of GD32 MCU:

- Learn to use external low speed crystal oscillator (LXTAL) to implement the CTC calibration function
- Learn to use clock trim controller (CTC) to trim internal 48MHz RC oscillator (IRC48M) clock

The CTC unit trim the frequency of the IRC48M based on an external accurate reference signal source. It can automatically adjust the trim value to provide a precise IRC48M clock.

5.9.2. DEMO Running Result

Download the program <09_CTC_Calibration> to the EVAL board and run. Firstly, all the LEDs flash once for test. Then if the clock trim is OK, LED2 will be on. Otherwise, all the LEDs are turned off.

5.10. SPI_LCD

5.10.1. DEMO Purpose

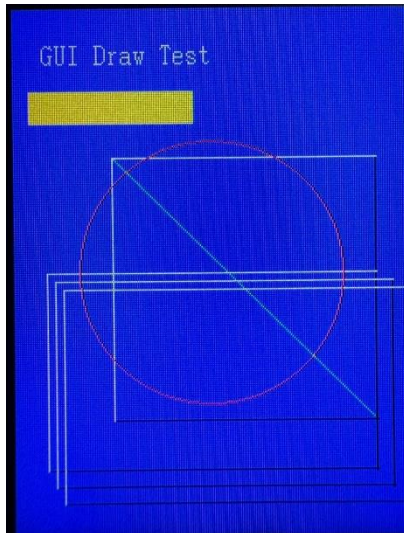
This demo includes the following functions of GD32 MCU:

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

GD32F5HCR-EVAL 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 GD32F5HCR-EVAL board. 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. TRNG_Get_Random

5.11.1. DEMO purpose

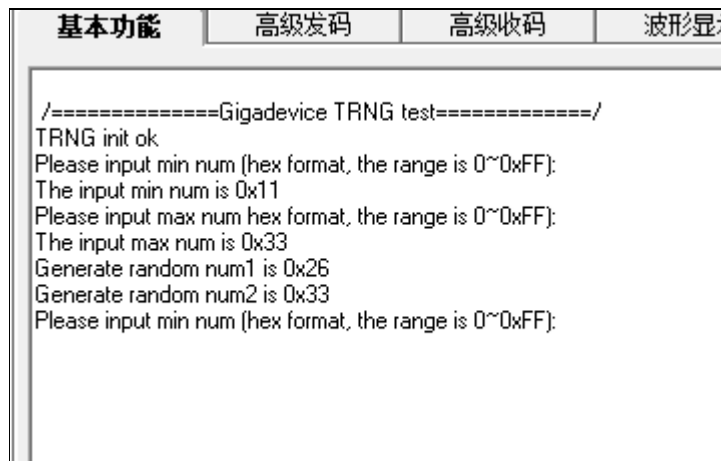
This demo includes the following functions of GD32 MCU:

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

5.11.2. DEMO running result

Download the program <11_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:



5.12. CAU

5.12.1. 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

5.12.2. DEMO Running Result

Download the program <12_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

5.13. HAU

5.13.1. 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

5.13.2. DEMO running result

Download the program <13_HAU> to the EVAL board and run. Connect serial cable to USART and open the serial terminal tool. 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:

CHN GigaDevice Semiconductor IncCHN GigaDevice Semiconductor IncCHN
GigaDevice Semiconductor IncCHN GigaDevice Semiconductor IncCHN
GigaDevice Semiconductor IncCHN GigaDevice Semiconductor IncCHN
GigaDevice Semiconductor IncCHN GigaDevice Semiconductor
Inc=====Choose HAU algorithm=====
1: SHA1 algorithm
2: SHA224 algorithm
3: SHA256 algorithm
4: MD5 algorithm

```

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):

0x06 0x9B 0x69 0x4C
0x14 0x07 0xDE 0x79
0xB7 0x2F 0x31 0xD9
0xB6 0xB4 0x97 0x84
0x6C 0x24 0x5A 0xB0

Example restarted...

message to be hashed:

CHN GigaDevice Semiconductor IncCHN GigaDevice Semiconductor IncCHN
GigaDevice Semiconductor IncCHN GigaDevice Semiconductor IncCHN
GigaDevice Semiconductor IncCHN GigaDevice Semiconductor IncCHN
GigaDevice Semiconductor IncCHN GigaDevice Semiconductor
Inc=====Choose HAU algorithm=====
1: SHA1 algorithm
2: SHA224 algorithm
3: SHA256 algorithm
4: MD5 algorithm

```

5.14. PKCAU_Modular_Addition_Interrupt

5.14.1. DEMO Purpose

This demo includes the following functions of GD32 MCU:

- Learn modular addition algorithm by interrupt

5.14.2. 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.

5.15. RCU_Clock_Out

5.15.1. 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

5.15.2. DEMO Running Result

Download the program <15_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 button. 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 and PC5 pin.

Information via a serial port output as following:

```
/===== GigaDevice Clock Output Demo =====/  
press tamper/wakeup key to select clock output source  
CK_OUT0: PLLP clock, DIV:5  
CK_OUT0: IRC16M, DIV:1  
CK_OUT0: HXTAL, DIV:2  
CK_OUT0: LXTAL
```

5.16. PMU_sleep_wakeup

5.16.1. DEMO Purpose

This demo includes the following functions of GD32 MCU:

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

5.16.2. DEMO Running Result

Download the program <16_PMU_sleep_wakeup> to the EVAL board, connect serial cable

to USART. After power-on, all the LEDs are off. The mcu will enter sleep mode and the software stops running. When the USART receives a byte of data from the HyperTerminal, the mcu will wake up from a receive interrupt. And all the LEDs will flash together.

5.17. RTC_Calendar

5.17.1. 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

5.17.2. DEMO running result

Download the program <17_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
```

5.18. IFRP

5.18.1. 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

5.18.2. DEMO Running Result

Download the program <18_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.

5.19. TIMER_Breath_LED

5.19.1. DEMO Purpose

This demo includes the following functions of GD32 MCU:

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

5.19.2. DEMO Running Result

Use the DuPont line to connect the TIMER0_CH0 (PA8) and LED (PB6). Then download the program <19_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.

5.20. USB_Device

5.20.1. 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)

GD32FHC-EVAL-V1.0 board has four keys and one USB_FS interface. The four keys are Reset key, Wakeup key, Tamper key and User key. In this demo, the GD32FHC-EVAL-V1.0 board is enumerated as a USB Keyboard, which uses the native PC Host HID driver, as shown below. The USB Keyboard uses three keys (wakeup key, tamper key and user key) to output three characters ('b', 'a' and 'c'). 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 wakeup key is used as the remote wakeup source.



DEMO running result

Download the program < 23_USB_FS\USB_Device_HID_Keyboard > to the EVAL board and run. If you press the Wakeup key, will output 'b'. If you press the User key, will output 'c'. If you press the Tamper key, 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 Wakeup key
- If PC is ON, remote wakeup is OK, else failed.

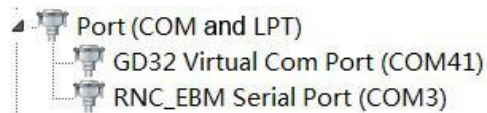
5.20.2. CDC_ACM

DEMO purpose

This demo includes the following functions of GD32 MCU:


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

GD32FHC-EVAL-V1.0 board has one USBFS interface. In this demo, the GD32FHC-EVAL-V1.0 board is enumerated as a 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 < 23_USB_FS\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

5.21. USB_Host

5.21.1. 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

GD32FHC-EVAL-V1.0 board integrates the USBFS module, and the module can be used as a USB device or a USB host. 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 <23_USB_FS\USB_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 user key will see the inserted device is mouse, and then moving the mouse will show the position of mouse and the state of button in the screen.

If a keyboard has been attached, the user will see the information of keyboard enumeration. First pressing the user key will see the inserted device is keyboard, and then pressing the keyboard will show the state of the button in the screen.

5.21.2. MSC_Host

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

GD32FHC-EVAL-V1.0 board integrates the USBFS module, and the module can be used as a USB device or a USB host. 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 <23_USB_FS\USB_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 user key will see the Udisk information, next pressing the tamper key will see the root content of the Udisk, then press the wakeup key will write file to the Udisk, finally the user will see information that the msc host demo is end.

5.22. Trustzone

5.22.1. 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 Trustzone
- 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

5.22.2. 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	Apr.15, 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.