

GigaDevice Semiconductor Inc.

GD32F503V-EVAL

Arm[®] Cortex[®]-M33 32-bit MCU

User Guide

Revision 1.1

(Nov. 2025)

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	9
4. Hardware layout overview.....	10
4.1. Power supply	10
4.2. Boot option	10
4.3. LED	10
4.4. KEY	11
4.5. ADC	11
4.6. DAC	11
4.7. CAN	11
4.8. CMP	12
4.9. USART	12
4.10. I2C	13
4.11. SPI	13
4.12. I2S	14
4.13. EXMC	14
4.14. Extension.....	15
4.15. GD-Link.....	16
4.16. USB	16
4.17. MCU.....	17
5. Routine use guide.....	18
5.1. GPIO_Runing_Led.....	18
5.1.1. DEMO Purpose.....	18
5.1.2. DEMO Running Result	18
5.2. GPIO_Key_Polling_mode.....	18
5.2.1. DEMO Purpose.....	18
5.2.2. DEMO Running Result	18
5.3. EXTI_Key_Interrupt_mode.....	19
5.3.1. DEMO Purpose.....	19
5.3.2. DEMO Running Result	19
5.4. USART_Printf.....	19
5.4.1. DEMO Purpose.....	19
5.4.2. DEMO Running Result	19

5.5. USART_Echo_Interrupt_mode.....	20
5.5.1. DEMO Purpose.....	20
5.5.2. DEMO Running Result	20
5.6. USART_DMA.....	20
5.6.1. DEMO Purpose.....	20
5.6.2. DEMO Running Result	20
5.7. ADC_Temperature_Vrefint	21
5.7.1. DEMO Purpose.....	21
5.7.2. DEMO Running Result	21
5.8. ADC0_ADC1_Follow_up_mode	22
5.8.1. DEMO Purpose.....	22
5.8.2. DEMO Running Result	22
5.9. ADC0_ADC1_Routine_Parallel_mode	22
5.9.1. DEMO Purpose.....	22
5.9.2. DEMO Running Result	23
5.10. DAC_Output_Voltage_Value	23
5.10.1. DEMO Purpose.....	23
5.10.2. DEMO Running Result	23
5.11. Comparator_Obtain_Brightness	24
5.11.1. DEMO Purpose.....	24
5.11.2. DEMO Running Result	24
5.12. I2C_EEPROM.....	24
5.12.1. DEMO Purpose.....	24
5.12.2. DEMO Running Result	24
5.13. SPI_Quad_Flash.....	25
5.13.1. DEMO Purpose.....	25
5.13.2. DEMO Running Result	25
5.14. I2S_Audio_Player.....	26
5.14.1. DEMO Purpose.....	26
5.14.2. DEMO Running Result	26
5.15. EXMC_NandFlash.....	27
5.15.1. DEMO Purpose.....	27
5.15.2. DEMO Running Result	27
5.16. EXMC_TouchScreen	27
5.16.1. DEMO Purpose.....	27
5.16.2. DEMO Running Result	27
5.17. CAN_Network	28
5.17.1. DEMO Purpose.....	28
5.17.2. DEMO Running Result	28
5.18. RCU_Clock_Out	29
5.18.1. DEMO Purpose.....	29
5.18.2. DEMO Running Result	29
5.19. CTC_Calibration	29

5.19.1.	DEMO Purpose	29
5.19.2.	DEMO Running Result	30
5.20.	PMU_sleep_wakeup	30
5.20.1.	DEMO Purpose	30
5.20.2.	DEMO Running Result	30
5.21.	RTC_Calendar	30
5.21.1.	DEMO Purpose	30
5.21.2.	DEMO Running Result	30
5.22.	TIMER_Breath_LED.....	31
5.22.1.	DEMO Purpose	31
5.22.2.	DEMO Running Result	31
5.23.	USB_Device	31
5.23.1.	HID_Keyboard	31
5.23.2.	CDC_ACM	32
5.24.	USB_Host	33
5.24.1.	HID_Host	33
5.24.2.	MSC_Host.....	34
6.	Revision history	35

List of Figures

Figure 4-1. Schematic diagram of power supply.....	10
Figure 4-2. Schematic diagram of boot option	10
Figure 4-3. Schematic diagram of LED function	10
Figure 4-4. Schematic diagram of Key function	11
Figure 4-5. Schematic diagram of ADC	11
Figure 4-6. Schematic diagram of DAC	11
Figure 4-7. Schematic diagram of CAN	11
Figure 4-8. Schematic diagram of CMP	12
Figure 4-9. Schematic diagram of USART	12
Figure 4-10. Schematic diagram of I2C	13
Figure 4-11. Schematic diagram of SPI	13
Figure 4-12. Schematic diagram of I2S	14
Figure 4-13. Schematic diagram of EXMC	14
Figure 4-14. Schematic diagram of Extension.....	15
Figure 4-15. Schematic diagram of GD-Link.....	16
Figure 4-16. Schematic diagram of USB	16
Figure 4-17. Schematic diagram of MCU.....	17

List of Tables

Table 2-1. Function pin assignment.....	7
Table 6-1. Revision history	35

1. Summary

GD32F503V-EVAL uses GD32F503VGT6 as the main controller. It uses GD-Link Type-C interface or DC-005 connector to supply 5V power. SWD, Reset, Boot, User button key, LED, CAN, I2C, I2S, USART, RTC, LCD, SPI, ADC, DAC, EXMC, USB, GD-Link and Extension Pins are also included. For more details, please refer to GD32F503V-EVAL-V1.1 schematic.

2. Function Pin Assign

Table 2-1. Function pin assignment

Function	Pin	Description
LED	PC7	LED1
	PC8	LED2
	PC9	LED3
RESET	NRST	K1-Reset
KEY	PA0	K2-Wakeup
	PC13	K3-Tamper
	PA1	K4-User key
USART0	PA9	USART0_TX
	PA10	USART0_RX
ADC	PC0	ADC012_IN13
DAC	PA4	DAC_OUT0
CMP	PA7	CMP0_IP
I2C	PB6	I2C0_SCL
	PB7	I2C0_SDA
SPI	PA2	SPI0_IO2
	PA3	SPI0_IO3
	PA4	SPI0_NSS
	PA5	SPI0_SCK
	PA6	SPI0_MISO
	PA7	SPI0_MOSI
I2S	PB12	I2S1_WS
	PB13	I2S1_CK
	PB15	I2S1_SD
	PC6	I2S1_MCK
CAN	PB8	CAN0_RX
	PB9	CAN0_TX
NAND Flash	PD14	EXMC_D0
	PD15	EXMC_D1
	PD0	EXMC_D2
	PD1	EXMC_D3
	PE7	EXMC_D4
	PE8	EXMC_D5
	PE9	EXMC_D6
	PE10	EXMC_D7
	PD11	EXMC_A16
	PD12	EXMC_A17

	PD4	EXMC_NOE
	PD5	EXMC_NWE
	PD6	EXMC_NWAIT
	PD7	EXMC_NCE
LCD	PD14	EXMC_D0
	PD15	EXMC_D1
	PD0	EXMC_D2
	PD1	EXMC_D3
	PE7	EXMC_D4
	PE8	EXMC_D5
	PE9	EXMC_D6
	PE10	EXMC_D7
	PE11	EXMC_D8
	PE12	EXMC_D9
	PE13	EXMC_D10
	PE14	EXMC_D11
	PE15	EXMC_D12
	PD8	EXMC_D13
	PD9	EXMC_D14
	PD10	EXMC_D15
	PE2	EXMC_A23
	PD4	EXMC_NOE
	PD5	EXMC_NWE
	PG9	EXMC_NE
	PC11	SPI2_MISO
	PC12	SPI2_MOSI
	PC10	SPI2_SCK
	PE5	TP_INT
USB	PA11	USBFS_DM
	PA12	USBFS_DP

3. Getting started

The EVAL board uses GD-Link Type-C interface or DC-005 connector 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.

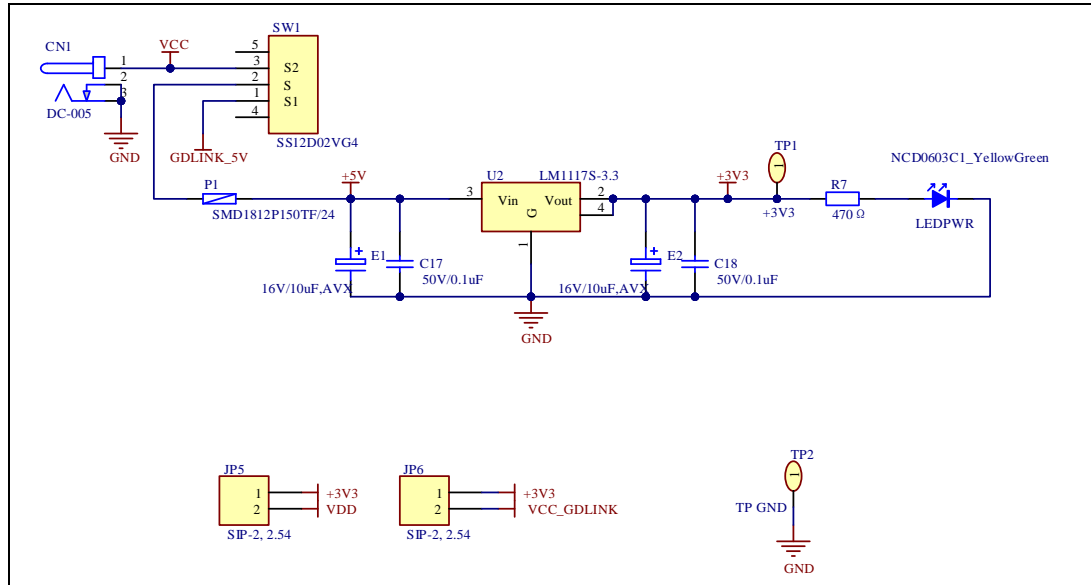
There are Keil version, IAR version and GD32EBuilder version of all projects. Keil version of the projects are created based on Keil MDK-ARM 5.29 uVision5. IAR version of the projects are created based on IAR Embedded Workbench for ARM 8.32.1 and GD32EBuilder version of the projects are created based 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.GD32F50x_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_GD32F50x_ADDON (URL: <https://www.gd32mcu.com>) should be installed to load related files.

4. Hardware layout overview

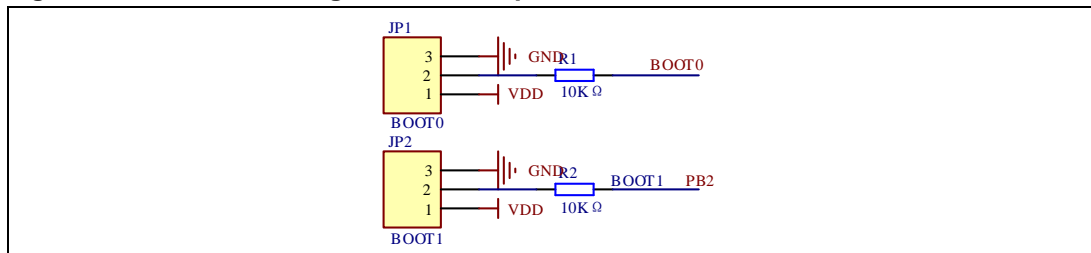
4.1. Power supply

Figure 4-1. Schematic diagram of power supply



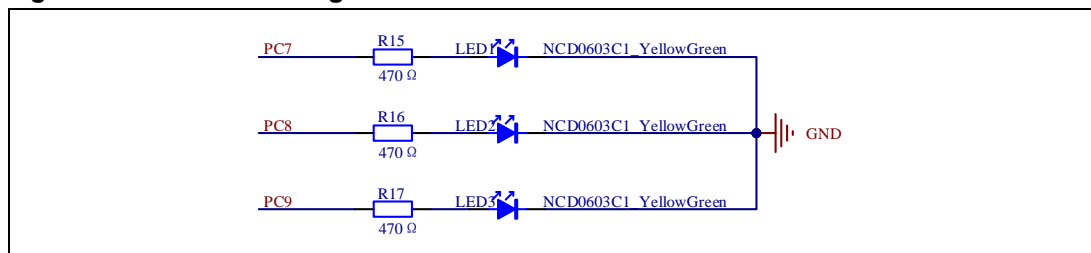
4.2. Boot option

Figure 4-2. Schematic diagram of boot option



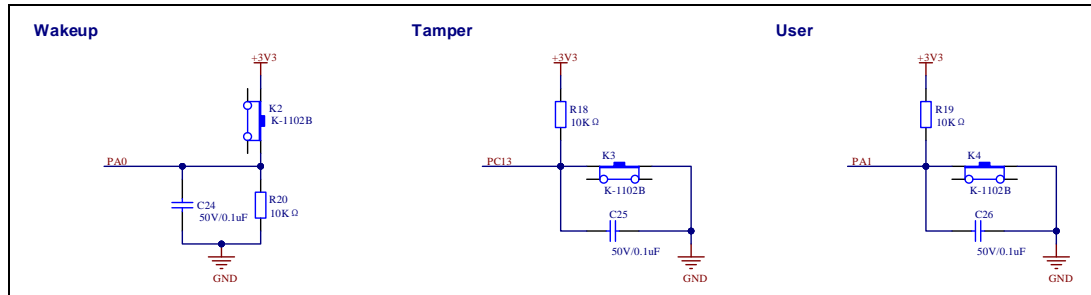
4.3. LED

Figure 4-3. Schematic diagram of LED function



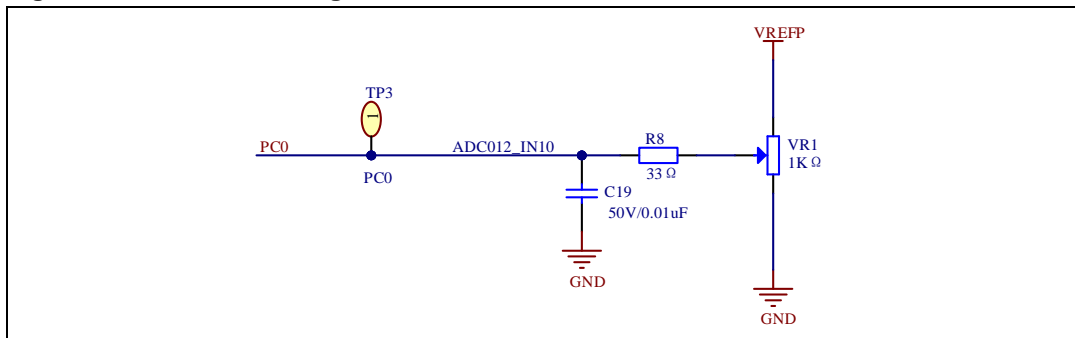
4.4. KEY

Figure 4-4. Schematic diagram of Key function



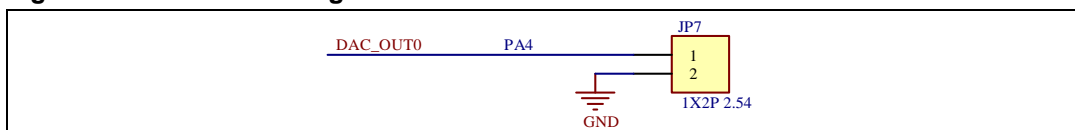
4.5. ADC

Figure 4-5. Schematic diagram of ADC



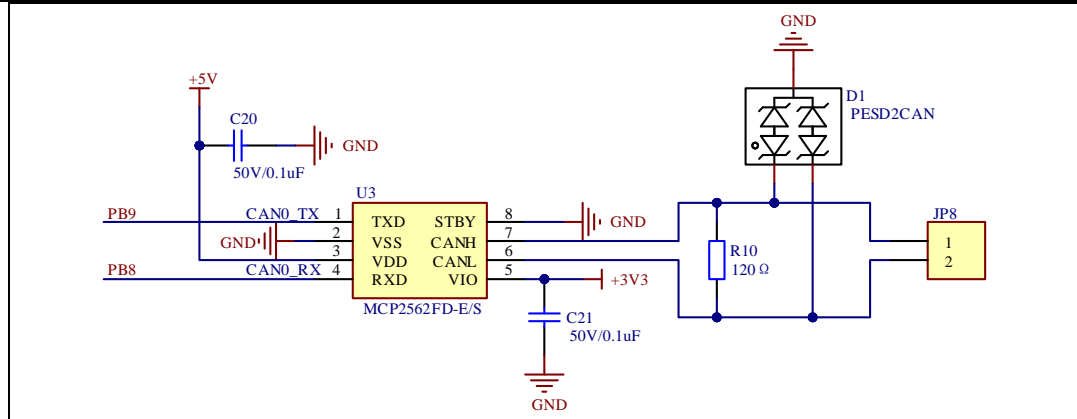
4.6. DAC

Figure 4-6. Schematic diagram of DAC



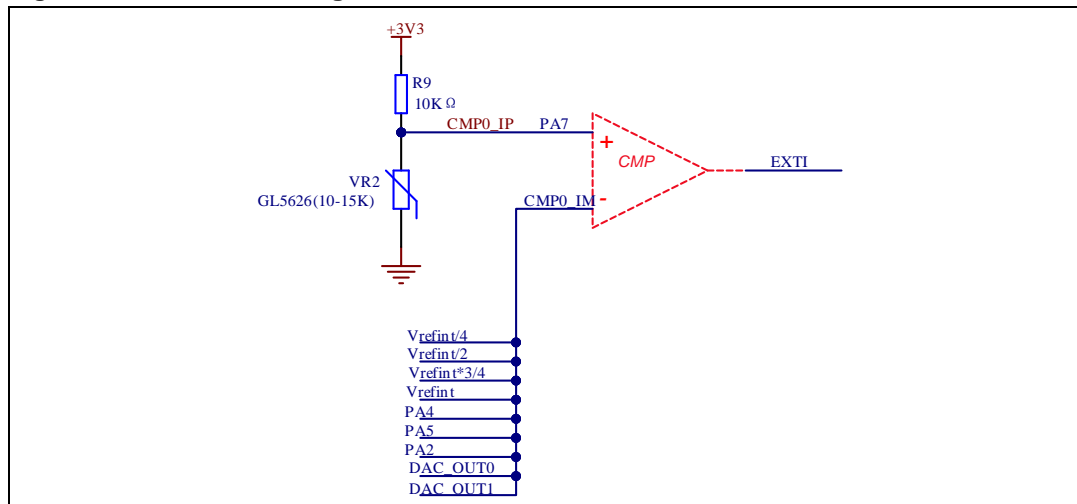
4.7. CAN

Figure 4-7. Schematic diagram of CAN



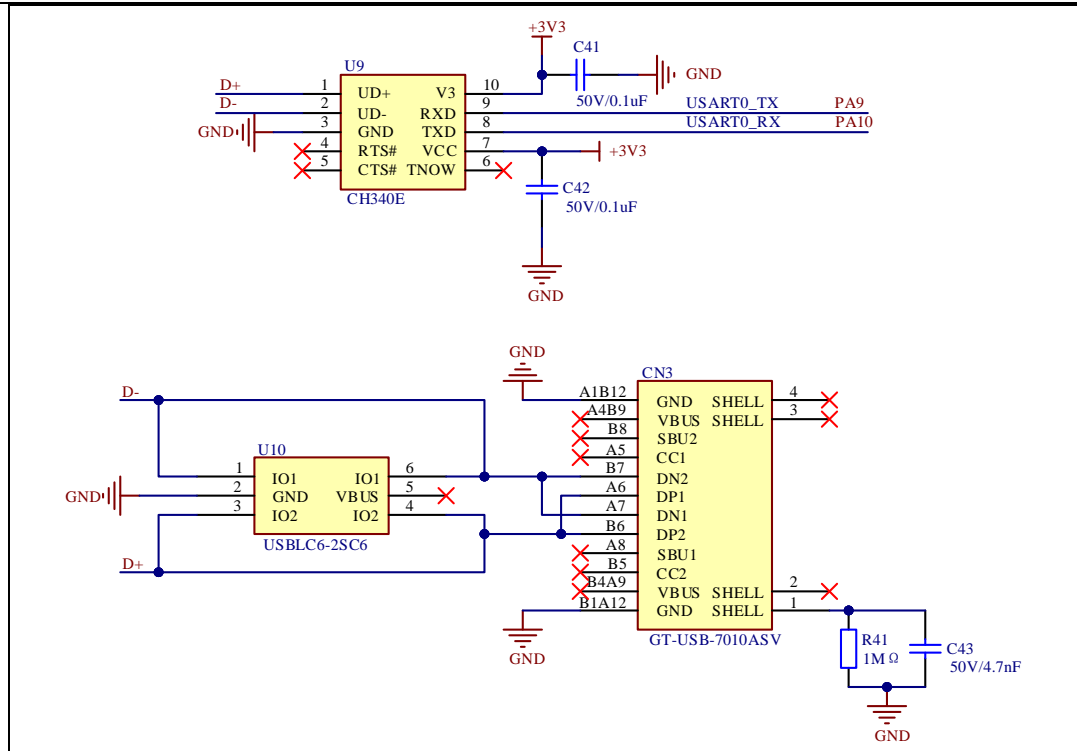
4.8. CMP

Figure 4-8. Schematic diagram of CMP



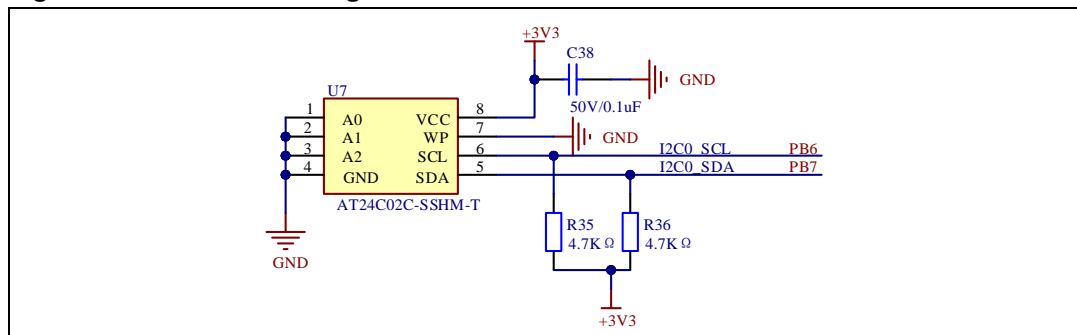
4.9. USART

Figure 4-9. Schematic diagram of USART



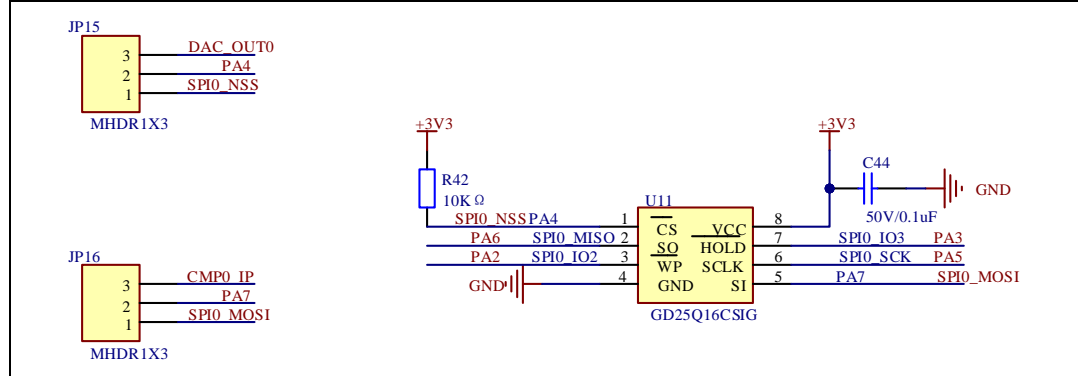
4.10. I2C

Figure 4-10. Schematic diagram of I2C



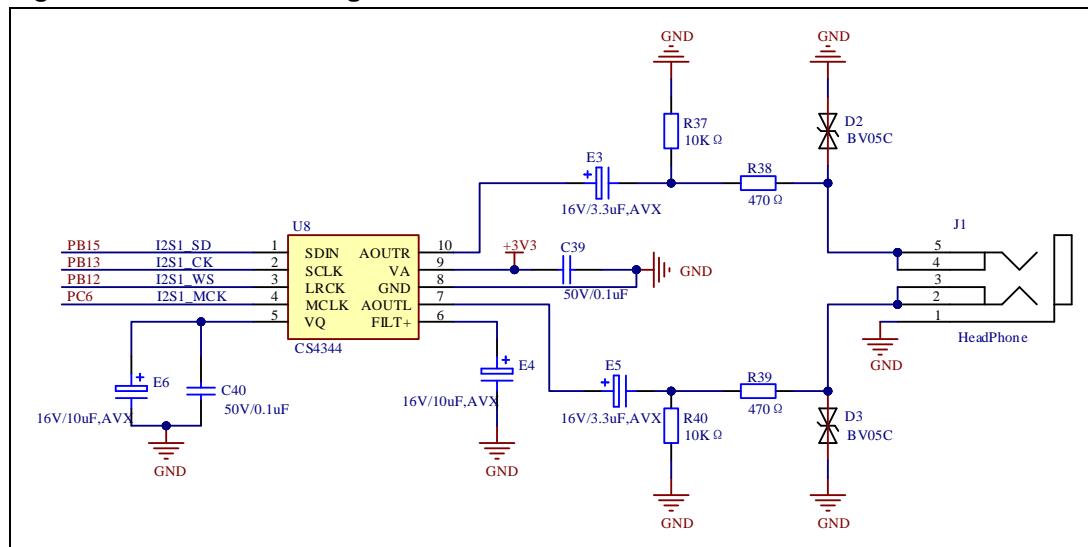
4.11. SPI

Figure 4-11. Schematic diagram of SPI



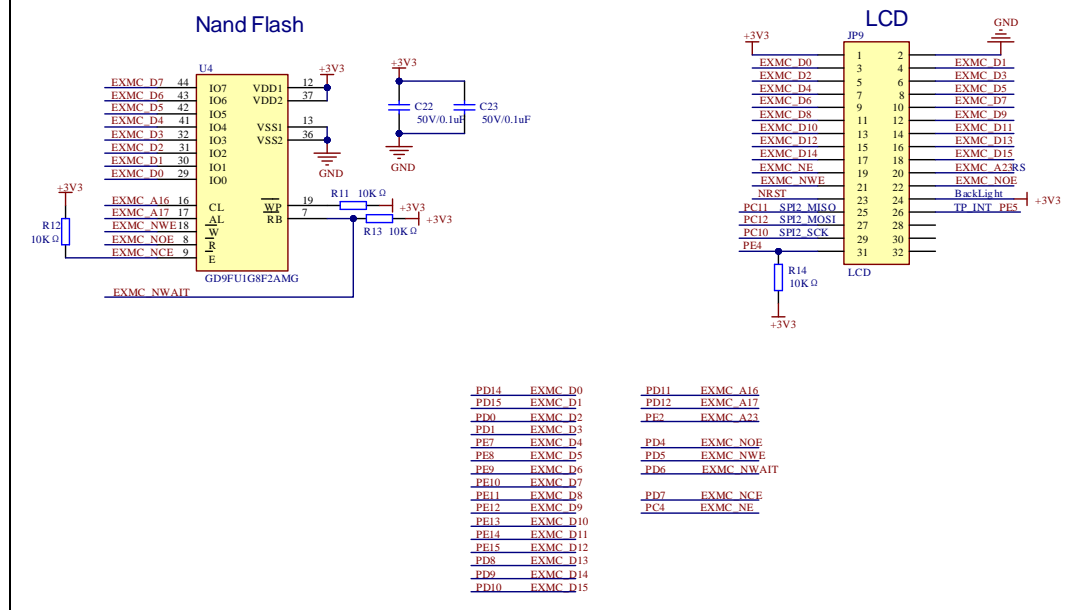
4.12. I2S

Figure 4-12. Schematic diagram of I2S



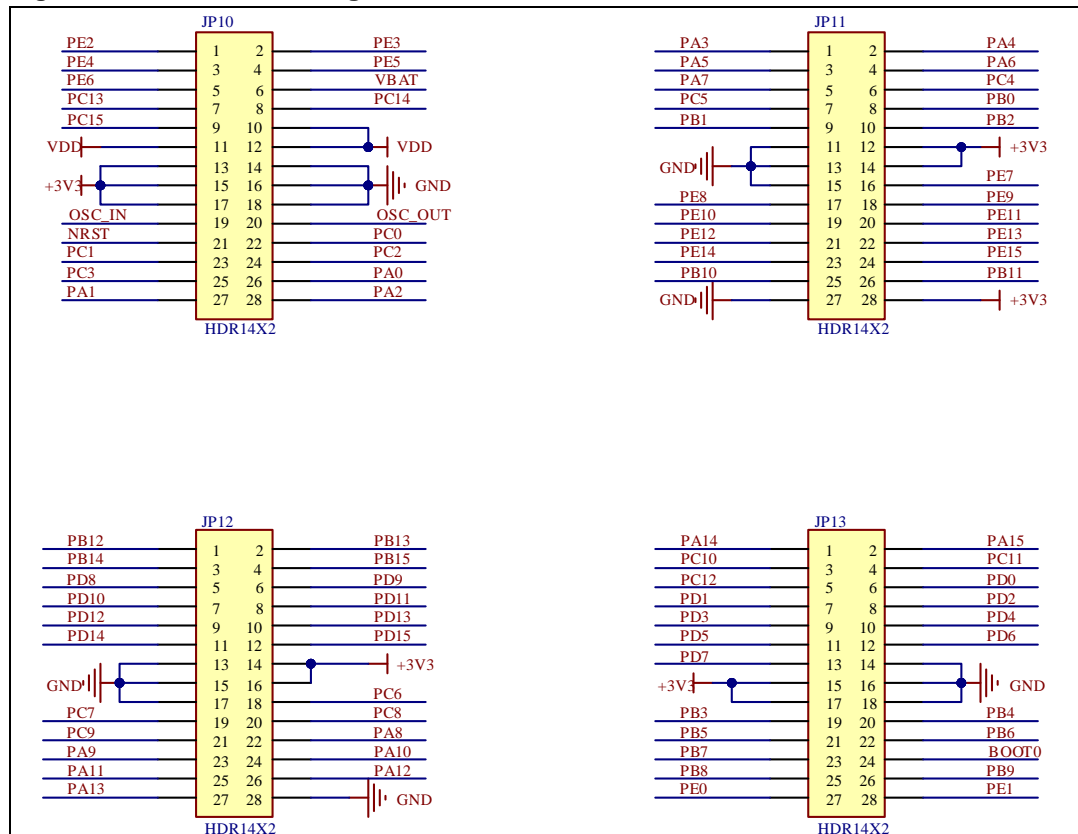
4.13. EXMC

Figure 4-13. Schematic diagram of EXMC



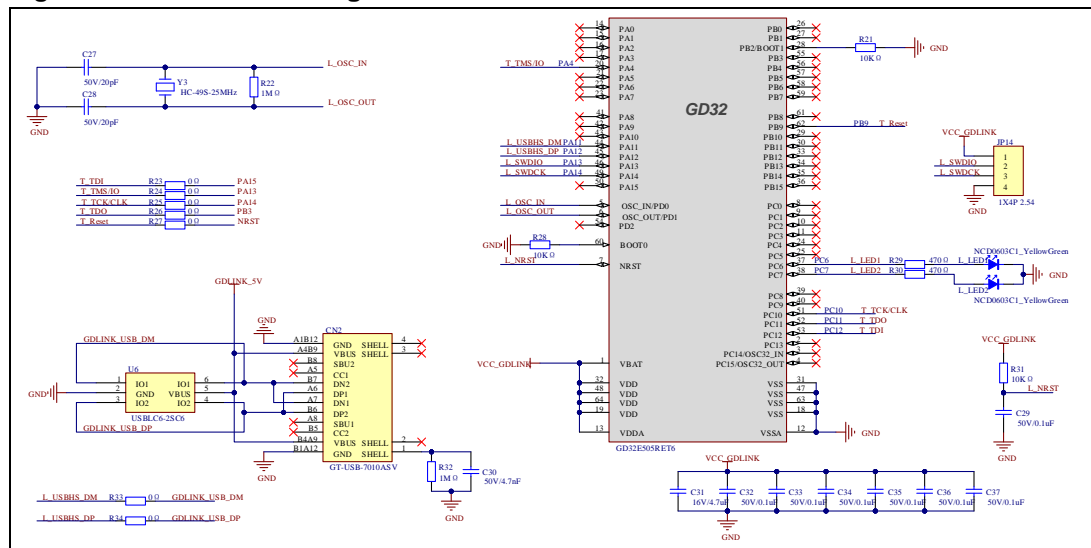
4.14. Extension

Figure 4-14. Schematic diagram of Extension



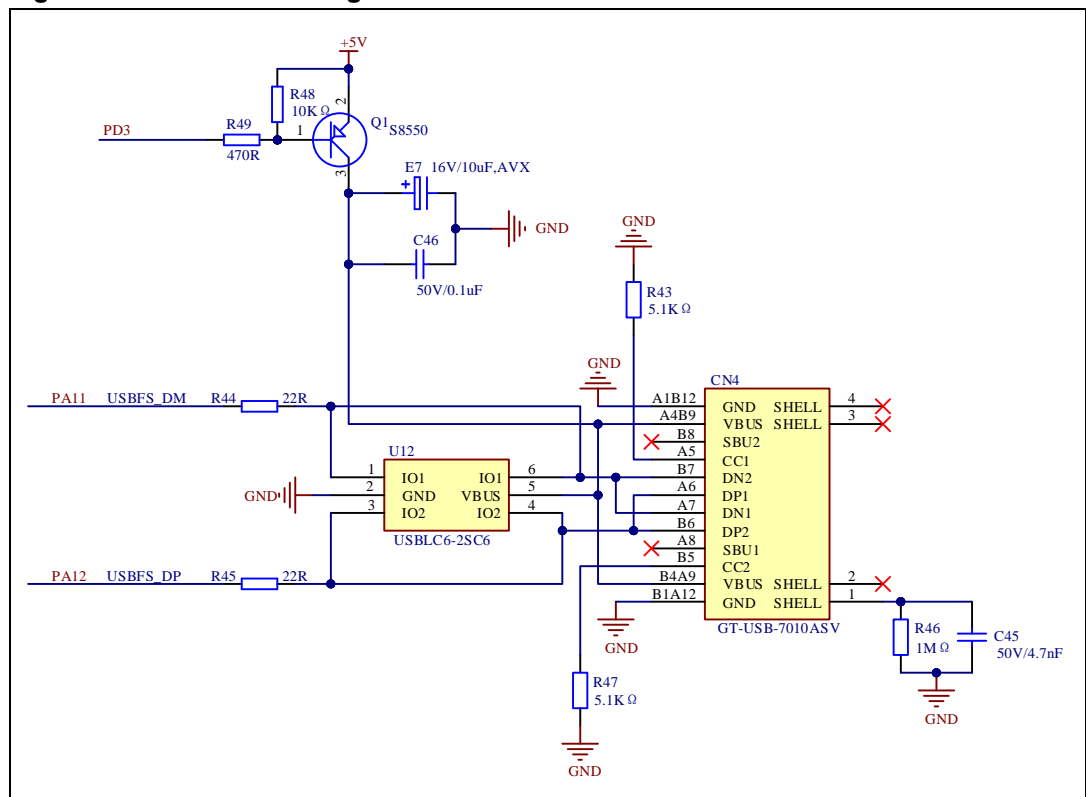
4.15. GD-Link

Figure 4-15. Schematic diagram of GD-Link



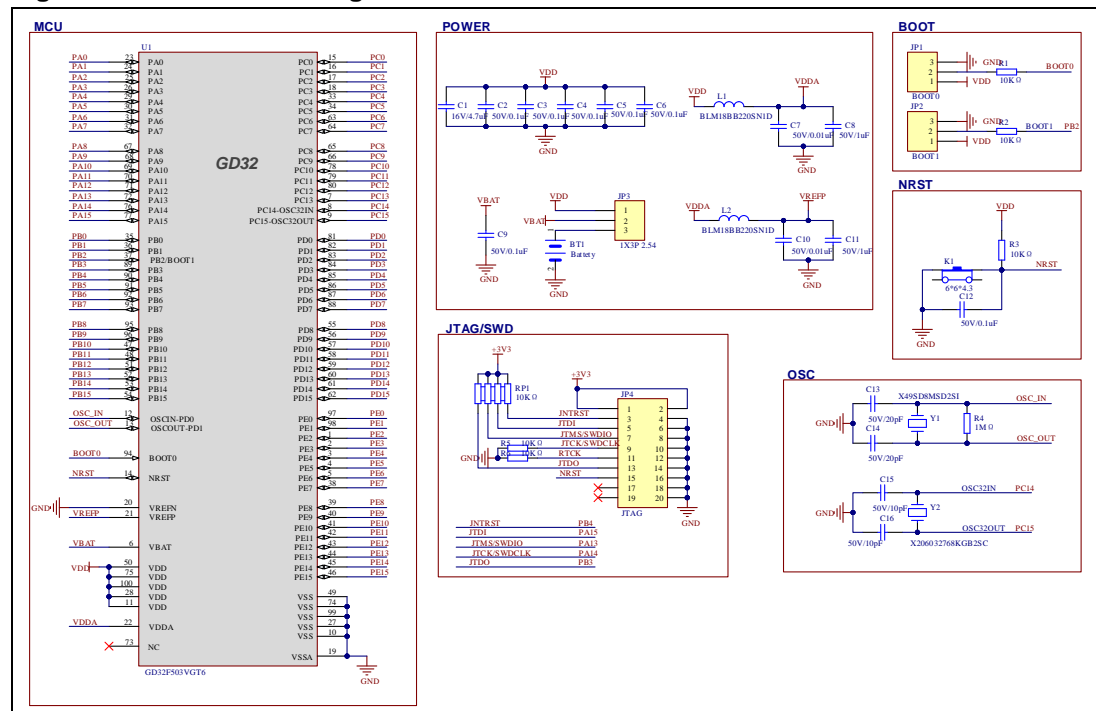
4.16. USB

Figure 4-16. Schematic diagram of USB



4.17. MCU

Figure 4-17. 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

GD32F503V-EVAL board has three LEDs. The LED1, LED2, LED3 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 will turn on in sequence with interval of 500ms, and turn off together, 500ms later, repeat the process.

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

GD32F503V-EVAL board has four keys and three LEDs. The four keys are Reset key, Tamper key, Wakeup key, User key. The LED1, LED2, LED3 are controlled by GPIO.

This demo will show how to use the Tamper key to control the LED1. When press down the Tamper Key, it will check the input value of the IO port. If the value is 0 and will wait for 50ms. 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 LED1.

5.2.2. DEMO Running Result

Download the program <02_GPIO_Key_Polling_mode> to the EVAL board, press down the Tamper Key, LED1 will be turned on. Press down the Tamper Key again, LED1 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

GD32F503V-EVAL board has four keys and three LEDs. The four keys are Reset key, Tamper key, Wakeup key, User key. The LED1, LED2, LED3 are controlled by GPIO.

This demo will show how to use the EXTI interrupt line to control the LED1. When press down the Tamper Key, it will produce an interrupt. In the interrupt service function, the demo will toggle LED1.

5.3.2. DEMO Running Result

Download the program <03_EXTI_Key_Interrupt_mode> to the EVAL board, LED1 is turned on and off for test. When press down the Tamper Key, LED1 will be turned on. Press down the Tamper Key again, LED1 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 LED2 is turned on, otherwise, LED2 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 flash by turns. Otherwise, LED1, LED2, LED3 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 flash by turns. Otherwise, LED1, LED2, LED3 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_Temperature_Vrefint

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 get the value of inner channel 16(temperature sensor channel) and channel 17 (V_{REFINT} channel)

5.7.2. DEMO Running Result

Download the program <07_ADC_Temperature_Vrefint> to the GD32F503V-EVAL board. Connect serial cable to USART, open the HyperTerminal.

When the program is running, HyperTerminal display the value of temperature and internal voltage reference (V_{REFINT}).

Notice: because there is an offset, when inner temperature sensor is used to detect accurate temperature, an external temperature sensor part should be used to calibrate the offset error.

```
the temperature data is 29 degrees Celsius
the reference voltage data is 1.192V
```

```
the temperature data is 28 degrees Celsius
the reference voltage data is 1.194V
```

```
the temperature data is 29 degrees Celsius
the reference voltage data is 1.192V
```

```
the temperature data is 29 degrees Celsius
the reference voltage data is 1.193V
```

```
the temperature data is 29 degrees Celsius
the reference voltage data is 1.193V
```

5.8. ADC0_ADC1_Follow_up_mode

5.8.1. 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 ADC0 and ADC1 follow-up mode

5.8.2. DEMO Running Result

Download the program <08_ADC0_ADC1_Follow_up_mode> to the GD32F503V-EVAL board. Connect serial cable to USART, open the HyperTerminal. PC0 and PC1 pin voltage access by external voltage.

TIMER1_CH0 is the trigger source of ADC0 and ADC1. When the rising edge of TIMER1_CH0 coming, ADC1 starts immediately and ADC0 starts after a delay of several ADC clock cycles. The values of ADC0 and ADC1 are transmitted to array `adc_value[0]` and `adc_value[1]` by DMA.

When sampling the first channel of ADCx (x=0,1), the value of the ADC1 conversion of PC1 pin is stored into the high half word of `adc_value[0]`, and after a delay of several ADC clock cycles the value of the ADC0 conversion of PC0 pin is stored into the low half word of `adc_value[0]`. When sampling the second channel of ADCx (x=0,1), the value of the ADC1 conversion of PC0 pin is stored into the high half word of `adc_value[1]`, and after a delay of several ADC clock cycles the value of the ADC0 conversion of PC1 pin is stored into the low half word of `adc_value[1]`.

When the program is running, HyperTerminal display the regular value of ADC0 and ADC1 by `adc_value[0]` and `adc_value[1]`.

```
*****  
  
adc_value[0] = 000006DA  
  
adc_value[1] = 06DC0000  
  
*****  
  
adc_value[0] = 000006DC  
  
adc_value[1] = 06DC0001  
  
*****
```

5.9. ADC0_ADC1_Routine_Parallel_mode

5.9.1. 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 ADC0 and ADC1 routine parallel mode

5.9.2. DEMO Running Result

Download the program <09_ADC0_ADC1_Routine_Parallel_mode> to the GD32F503V-EVAL board. Connect serial cable to USART, open the HyperTerminal. PC0 and PC1 pin connect to external voltage input.

TIMER1_CH0 is the trigger source of ADC0 and ADC1. When the rising edge of TIMER1_CH0 coming, ADC0 and ADC1 convert the routine sequence parallelly. The values of ADC0 and ADC1 are transmitted to array `adc_value[0]` and `adc_value[1]` by DMA.

When the first rising edge of TIMER1_CH0 coming, the value of the ADC0 conversion of PC0 pin is stored into the low half word of `adc_value[0]`, the value of the ADC1 conversion of PC1 pin is stored into the high half word of `adc_value[0]`. When the second rising edge of TIMER1_CH0 coming, the value of the ADC0 conversion of PC1 pin is stored into the low half word of `adc_value[1]`, the value of the ADC1 conversion of PC0 pin is stored into the high half word of `adc_value[1]`.

When the program is running, HyperTerminal displays the regular value of ADC0 and ADC1 stored in `adc_value[0]` and `adc_value[1]`.

```
*****  
  
adc_value[0] = 000006DA  
adc_value[1] = 06DC0000  
  
*****  
  
adc_value[0] = 000006DC  
adc_value[1] = 06DC0001  
  
*****
```

5.10. DAC_Output_Voltage_Value

5.10.1. DEMO Purpose

This demo includes the following functions of GD32 MCU:

- Learn to use DAC to output voltage on DAC0_OUT0 output

5.10.2. DEMO Running Result

Download the program <10_DAC_Output_Voltage_Value> to the EVAL board and run.

Firstly, all the LEDs will turn on and turn off for test. And then the digital value 0x7FF0, which should be 1.65V ($V_{REF}/2$), would be output on PA4.

The voltage on PA4 can be observed through the oscilloscope.

5.11. Comparator_Obtain_Brightness

5.11.1. DEMO Purpose

This demo includes the following functions of GD32 MCU:

- Learn to use comparator output compare result

The comparator has two inputs, in this demo, one input is PA7, and the other one is the reference voltage. Compare the two input voltages, the output is a high or low level, and the LED2 will perform the corresponding action.

5.11.2. DEMO Running Result

Download the program <11_Comparator_Obtain_Brightness> to the EVAL board, comparing two input voltage, if output level is high, LED2 is on, otherwise LED2 is off.

5.12. I2C_EEPROM

5.12.1. 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

5.12.2. 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 two LEDs lights flashing, otherwise the serial port will output "Err: data read and write aren't matching." and all the two 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!
```

5.13. SPI_Quad_Flash

5.13.1. DEMO Purpose

This demo includes the following functions of GD32 MCU:

- Learn to use the master mode of SPI unit to read and write NOR Flash with the SPI interface

5.13.2. DEMO Running Result

The computer serial port line connected to the COM port of development board, set the baud rate of HyperTerminal software to 115200, 8 bits data bit, 1 bit stop bit. At the same time you should jump the JP15 and JP16 to SPI.

Download the program <13_SPI_Quad_Flash> to the EVAL board, the HyperTerminal software can observe the operation condition and will display the ID of the flash, 256 bytes data which are written to and read from flash. Compare the data that were written to the flash and the data that were read from the flash. If they are the same, the serial port will output "SPI-GD25Q16 Test Passed!", otherwise, the serial port will output "Err: Data Read and Write aren't Matching.". At last, turn on and off the leds one by one. The following is the experimental

results.

```

SPI Flash:GD25Q16 configured...
The Flash_ID:0xC84015

Write to tx_buffer:
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

Read from rx_buffer:
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
SPI-GD25Q16 Test Passed!

```

5.14. I2S_Audio_Player

5.14.1. DEMO Purpose

This Demo includes the following functions of GD32 MCU:

- Learn to use I2S module to output audio file
- Parsing audio files of wav format

The I2S (Inter-IC Sound) module can communicate with external devices using the I2S audio protocol. This Demo mainly shows how to use the I2S interface of the board for audio output.

5.14.2. DEMO Running Result

Download the program<14_I2S_Audio_Player>to the EVAL board, insert the headphone into the audio port, and then listen to the audio file.

5.15. EXMC_NandFlash

5.15.1. DEMO Purpose

This demo includes the following functions of GD32 MCU:

- Learn to use EXMC control the NAND flash

5.15.2. DEMO Running Result

GD32F503V-EVAL board has EXMC module to control NAND flash. Download the program <15_EXMC_NandFlash> to the EVAL board. This demo shows the write and read operation process of NAND flash memory by EXMC module. If the test pass, LED1 will be turned on. Otherwise, turn on the LED3. Information via a HyperTerminal output as following:

```
NAND flash initialized!
Read NAND ID!
Nand flash ID:0xC8 0xF1 0x80 0x1D

Write data successfully!
Read data successfully!
Check the data!
Access NAND flash successfully!
The data to be read:
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
0x00 0x01 0x02 0x03 0x04 0x05 0x06 0x07 0x08 0x09 0x0A 0x0B 0x0C 0x0D 0x0E 0x0F
```

5.16. EXMC_TouchScreen

5.16.1. DEMO Purpose

This demo includes the following functions of GD32 MCU:

- Learn to use EXMC control LCD

5.16.2. DEMO Running Result

GD32F503V-EVAL board has EXMC module to control LCD. Download the program

<16_EXMC_TouchScreen> to the EVAL board. This demo displays GigaDevice logo and four green buttons on the LCD screen by EXMC module. Users can touch the green button to turn on the corresponding LED on board, and then the color of button you had touched will change to red.



5.17. CAN_Network

5.17.1. DEMO Purpose

This demo includes the following functions of GD32 MCU:

- Learn to use the CAN0 communication between two boards

GD32F503V-EVAL board integrates CAN (controller area network) bus controller. It is a common industrial control bus. The CAN bus controller supports the CAN protocols version 2.0A, 2.0B, ISO11891-1:2015 and BOSCH CAN FD specification. This routine demonstrates the communication between two boards through CAN0.

5.17.2. DEMO Running Result

This example is tested with two GD32F503V-EVAL boards. Connect L pin to L pin and H pin to H pin of JP8 on the boards for sending and receiving frames. Download the program <17_CAN_Network> to the two EVAL boards, and connect serial cable to USART. Firstly, the USART sends “please press the Tamper key to transmit data!” to the HyperTerminal. The frames are sent and the transmit data are printed by pressing Tamper Key push button. When the frames are received, the receive data will be printed and the LED1 will toggle one time.

The output information via the serial port is as following.

```

please press the Tamper key to transmit data!

can0 transmit data: a0 a1 a2 a3 a4 a5 a6 a7

can0 receive data: aa aa aa aa aa aa aa aa

```

5.18. RCU_Clock_Out

5.18.1. 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

5.18.2. DEMO Running Result

Download the program <18_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 pin.

Information via a serial port output as following:

```

/===== Gigadevice Clock output Demo =====/
press tamper key to select clock output source
CK_OUT0: system clock
CK_OUT0: IRC8M
CK_OUT0: HXTAL
CK_OUT0: system clock

```

5.19. CTC_Calibration

5.19.1. 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.

5.19.2. DEMO Running Result

Download the program <19_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.

5.20. PMU_sleep_wakeup

5.20.1. 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

5.20.2. DEMO Running Result

Download the program <20_PMU_sleep_wakeup> to the EVAL board, connect serial cable to COM0. After power-on, all the LEDs are off. The mcu will enter sleep mode and the software stops running. When the USART0 receives a byte of data from the HyperTerminal, the mcu will wake up from a receive interrupt. And all the LEDs will flash together.

5.21. RTC_Calendar

5.21.1. DEMO Purpose

This demo includes the following functions of GD32 MCU:

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

5.21.2. DEMO Running Result

Download the program <21_RTC_Calendar> 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.

```
This is a RTC demo.....

This is a RTC demo!

RTC not yet configured...
RTC configured...

=====Time Settings=====
Please Set Hours
: 0
Please Set Minutes
: 0
Please Set Seconds
: 0

Time: 00:00:00

Time: 00:00:00

Time: 00:00:01

Time: 00:00:02
```

5.22. TIMER_Breath_LED

5.22.1. DEMO Purpose

This demo includes the following functions of GD32 MCU:

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

5.22.2. DEMO Running Result

Download the program <22_TIMER_Breath_LED> to the GD32F503V-EVAL board and run. PC7 should not be reused by other peripherals.

When the program is running, you can see LED1 lighting from dark to bright gradually and then gradually darken, ad infinitum, just like breathing as rhythm.

5.23. USB_Device

5.23.1. 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)

GD32F503V-EVAL-V1.1 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 GD32F503V-EVAL-V1.1 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.

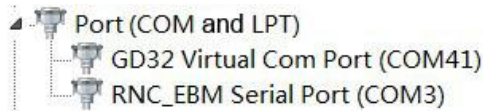
5.23.2. CDC_ACM

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

GD32F503V-EVAL-V1.1 board has one USBFS interface. In this demo, the GD32F503V-EVAL-V1.1 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.



5.24. USB_Host

5.24.1. 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

GD32F503V-EVAL-V1.1 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.

5.24.2. 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

GD32F503V-EVAL-V1.1 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.

6. Revision history

Table 6-1. Revision history

Revision No.	Description	Date
1.0	Initial Release	Nov.3, 2025
1.1	Modify the description in the “Getting started”	Nov.10, 2025

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.