GigaDevice Semiconductor Inc.

GD32L235R-EVAL
Arm® Cortex®-M23 32-bit MCU

User Guide

Revision 1.1

(Aug. 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 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. CMP 12
4.8. CAN 12
4.9. USART 12
4.10. I2C 13
4.11. I2S 13
4.12. SPI 14
4.13. SLCD 14
4.14. USB 15
4.15. Extension 15
4.16. GD-Link 16
4.17. MCU 16
5. Routine use guide 17
5.1. GPIO_Running_LED 17
5.1.1. DEMO purpose 17
5.1.2. DEMO running result 17
5.2. GPIO_Key_Polling_mode 17
5.2.1. DEMO purpose 17
5.2.2. DEMO running result 17
5.3. EXTI_Key_Interrupt_mode 18
5.3.1. DEMO purpose 18
5.3.2. DEMO running result 18
5.4. USART_Printf 18
5.4.1. DEMO purpose 18
5.4.2. DEMO running result 18

5.5. USART_HyperTerminal_Interrupt 19
5.5.1. DEMO purpose 19
5.5.2. DEMO running result 19
5.6. USART_DMA 19
5.6.1. DEMO purpose 19
5.6.2. DEMO running result 19
5.7. LPUART_Deepsleep_Wakeup 20
5.7.1. DEMO purpose 20
5.7.2. DEMO running result 20
5.8. ADC_Temperature_Vrefint 20
5.8.1. DEMO purpose 20
5.8.2. DEMO running result 20
5.9. DAC_Output_Voltage_Value 21
5.9.1. DEMO purpose 21
5.9.2. DEMO running result 21
5.10. Comparator_Obtain_Brightness 21
5.10.1. DEMO purpose 21
5.10.2. DEMO running result 22
5.11. I2C_EEPROM 22
5.11.1. DEMO purpose 22
5.11.2. DEMO running result 22
5.12. SPI_QUAD_FLASH 23
5.12.1. DEMO purpose 23
5.12.2. DEMO running result 23
5.13. I2S_Audio_Player 25
5.13.1. DEMO purpose 25
5.13.2. DEMO running result 25
5.14. TRNG_Get_Random 25
5.14.1. DEMO purpose 25
5.14.2. DEMO running result 25
5.15. CAU 26
5.15.1. DEMO purpose 26
5.15.2. DEMO running result 26
5.16. CAN_Network 28
5.16.1. DEMO purpose 28
5.16.2. DEMO running result 28
5.17. RCU_Clock_Out 28
5.17.1. DEMO purpose 28
5.17.2. DEMO running result 28
5.18. CTC_Calibration 29
5.18.1. DEMO purpose 29
5.18.2. DEMO running result 29
5.19. PMU_Sleep_Wakeup 29
5.19.1. DEMO purpose 29

5.19.2. DEMO running result 29
5.20. RTC_Calendar 30
5.20.1. DEMO purpose 30
5.20.2. DEMO running result 30
5.21. TIMER_Breath_LED 31
5.21.1. DEMO purpose 31
5.21.2. DEMO running result 31
5.22. LPTIMER_Deepsleep_Pwmout 32
5.22.1. DEMO purpose 32
5.22.2. DEMO running result 32
5.23. SLCD_Glass 32
5.23.1. DEMO purpose 32
5.23.2. DEMO running result 32
5.24. USBD_Keyboard 32
5.24.1. DEMO_purpose 32
5.24.2. DEMO running result 33
5.25. USBD_CDC_ACM 33
5.25.1. DEMO purpose 33
5.25.2. DEMO running result 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 CMP 12
Figure 4-8. Schematic diagram of CAN 12
Figure 4-9. Schematic diagram of USART 12
Figure 4-10. Schematic diagram of I2C 13
Figure 4-11. Schematic diagram of I2S 13
Figure 4-12. Schematic diagram of SPI 14
Figure 4-13. Schematic diagram of SLCD 14
Figure 4-14. Schematic diagram of USB 15
Figure 4-15. Schematic diagram of Extension 15
Figure 4-16. Schematic diagram of GD-Link 16
Figure 4-17. Schematic diagram of MCU 16

List of Tables

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

1. Summary

GD32L235R-EVAL uses GD32L235RBT6 as the main controller. It uses GD-Link Mini USB interface to supply 5V power. Reset, Boot, Button key, LED, I2S, I2C-EEPROM, SLCD, QSPI-Flash, USB and USART to USB interface are also included. For more details, please refer to GD32L235R-EVAL_Rev1.0 schematic.

2. Function Pin Assign

Table 2-1. Function pin assignment

Function	Pin	Description
LED	PA5	LED1
	PB12	LED2
RESET		Reset
KEY	PA0	K3(Wakeup)
	PC13	K2(Tamper)
ADC	PB0	ADC_P
	PB1	ADC_N
DAC	PA4	DAC_OUT
USART	PA2	USART1_TX(LPUART0_TX)
	PA3	USART1_RX(LPUART0_RX)
I2C	PB10	I2C1_SCL
	PB11	I2C1_SDA
I2S	PC12	I2S1_SD
	PC10	I2S1_CK
	PA15	I2S1_WS
	PC6	I2S1_MCK
SPI	PD9	SPI0_CS
	PD8	SPI0_SCK
	PA6	SPI0_MISO
	PA7	SPI0_MOSI
	PB6	SPI0_IO2
	PB7	SPI0_IO3
SLCD	PA8	COM0
	PA9	COM1
	PA10	COM2
	PB9	COM3
	PC10	COM4
	PC11	COM5
	PC12	COM6
	PD2	COM7
	PA1	SEG0
	PA2	SEG1
	PA3	SEG2
	PA6	SEG3
	PA7	SEG4
	PB0	SEG5
	PB1	SEG6

Function	Pin	Description
	PB3	SEG7
	PB4	SEG8
	PB5	SEG9
	PB10	SEG10
	PB11	SEG11
	PB12	SEG12
	PB13	SEG13
	PB14	SEG14
	PB15	SEG15
	PB8	SEG16
	PA15	SEG17
	PC0	SEG18
	PC1	SEG19
	PC2	SEG20
	PC3	SEG21
	PC4	SEG22
	PC5	SEG23
	PC6	SEG24
	PC7	SEG25
	PC8	SEG26
	PC9	SEG27
	PD4	SEG28
	PD5	SEG29
	PD8	SEG30
	PD9	SEG31
CMP	PA1	CMP0_IP
CAN	PD0	CAN_RX
	PD1	CAN_TX
USB	PA11	USB_DM
	PA12	USB_DP

3. Getting started

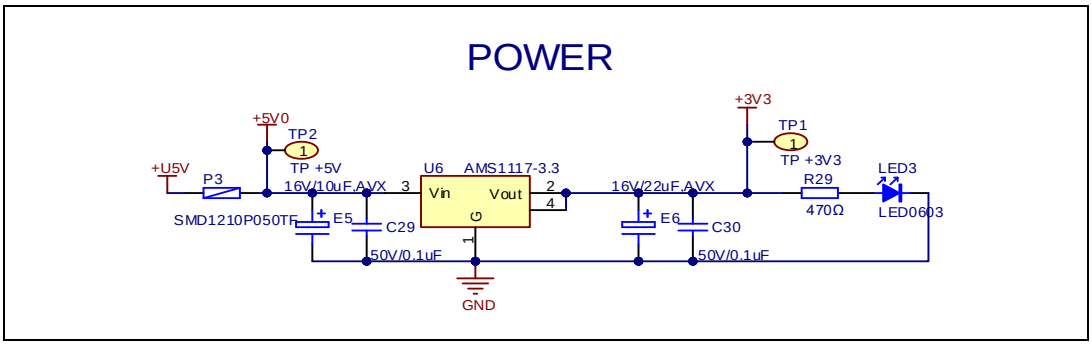
The EVAL board uses GD-Link Mini USB connecter to get power DC +5V, which is the hardware system normal work voltage. A GD-Link on board is necessary in order to download and debug programs. Select the correct boot mode and then power on, the LED3 will turn on, which indicates the power supply is OK.

There are Keil version and IAR version of all projects. Keil version of the projects are created based on Keil MDK-ARM 5.29 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.GD32L23x_DFP.1.1.6.
2. If you use IAR to open the project, install IAR_GD32L23x_ADDON.1.1.1.exe to load the associated files.

4. Hardware layout overview

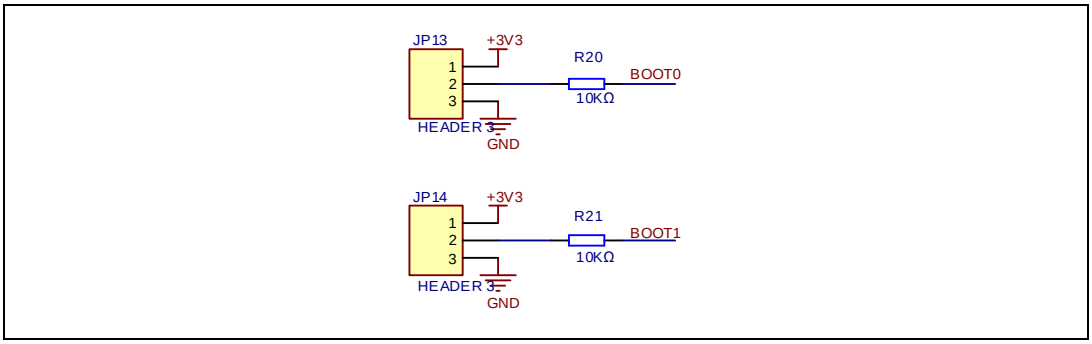
4.1. Power supply

Figure 4-1. Schematic diagram of power supply



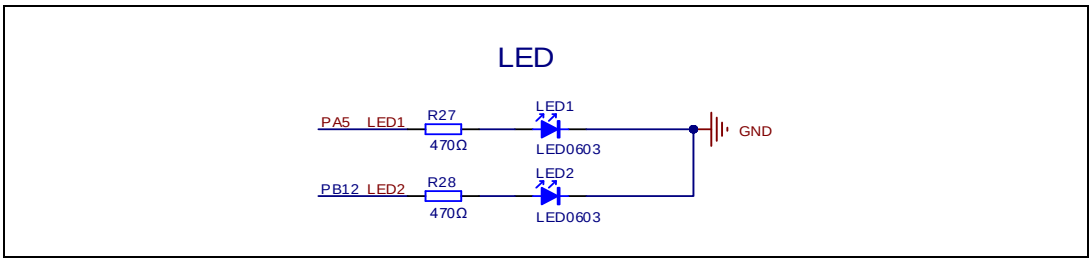
4.2. Boot option

Figure 4-2. Schematic diagram of boot option



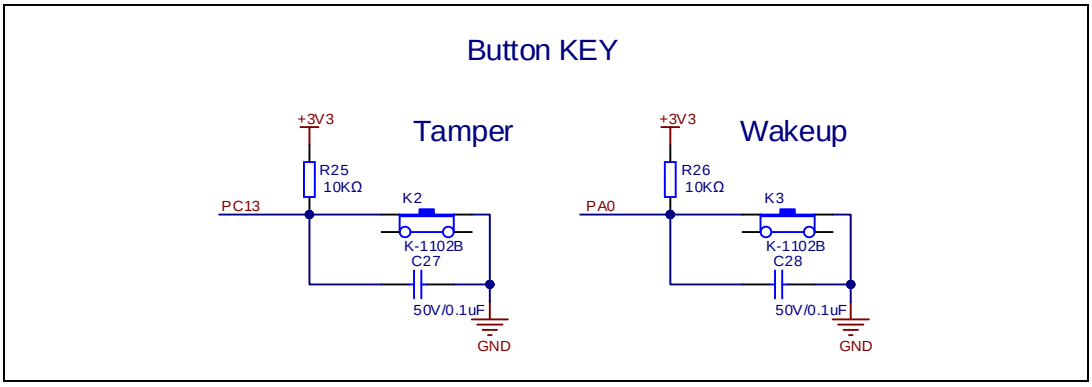
4.3. LED

Figure 4-3. Schematic diagram of LED function



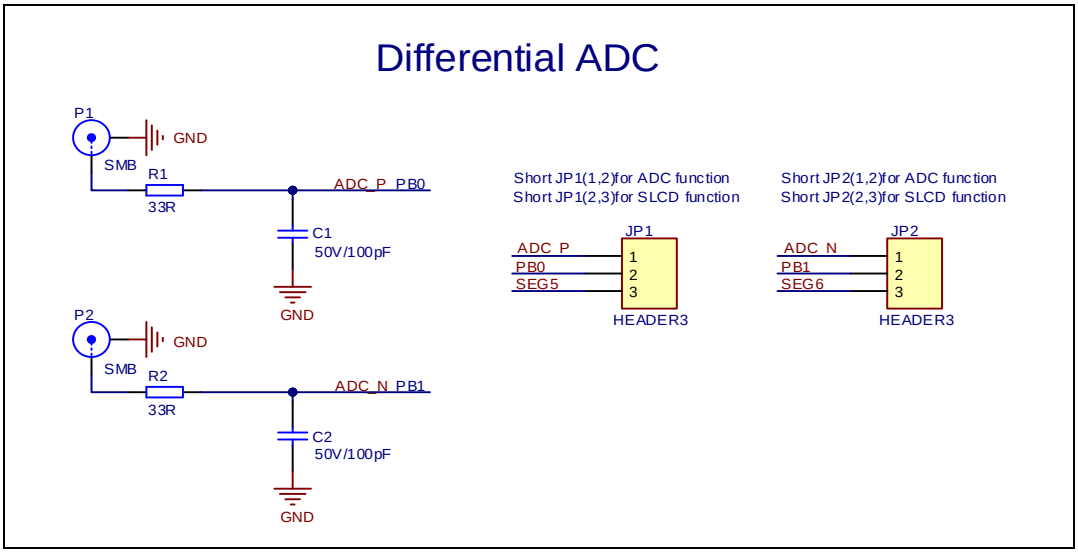
4.4. KEY

Figure 4-4. Schematic diagram of Key function



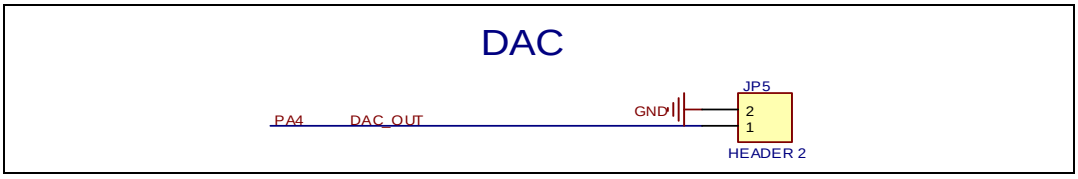
4.5. ADC

Figure 4-5. Schematic diagram of ADC



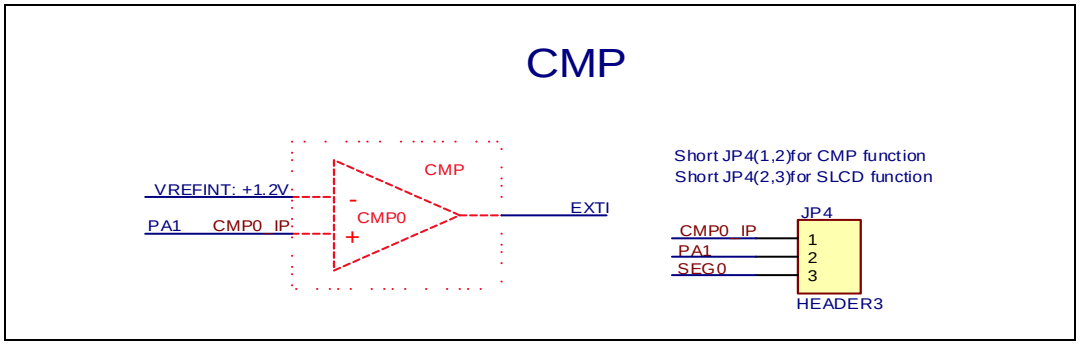
4.6. DAC

Figure 4-6. Schematic diagram of DAC



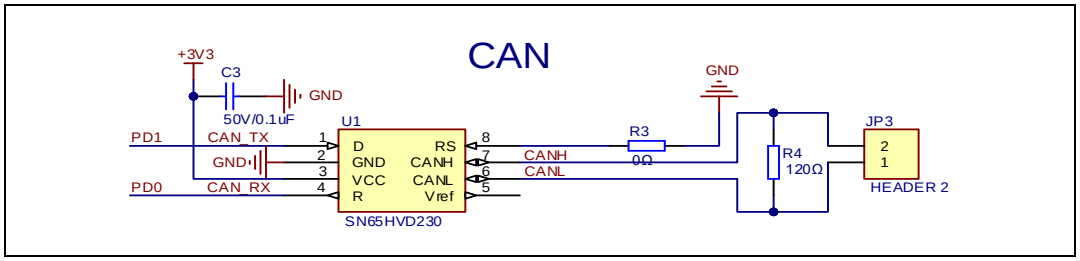
4.7. CMP

Figure 4-7. Schematic diagram of CMP



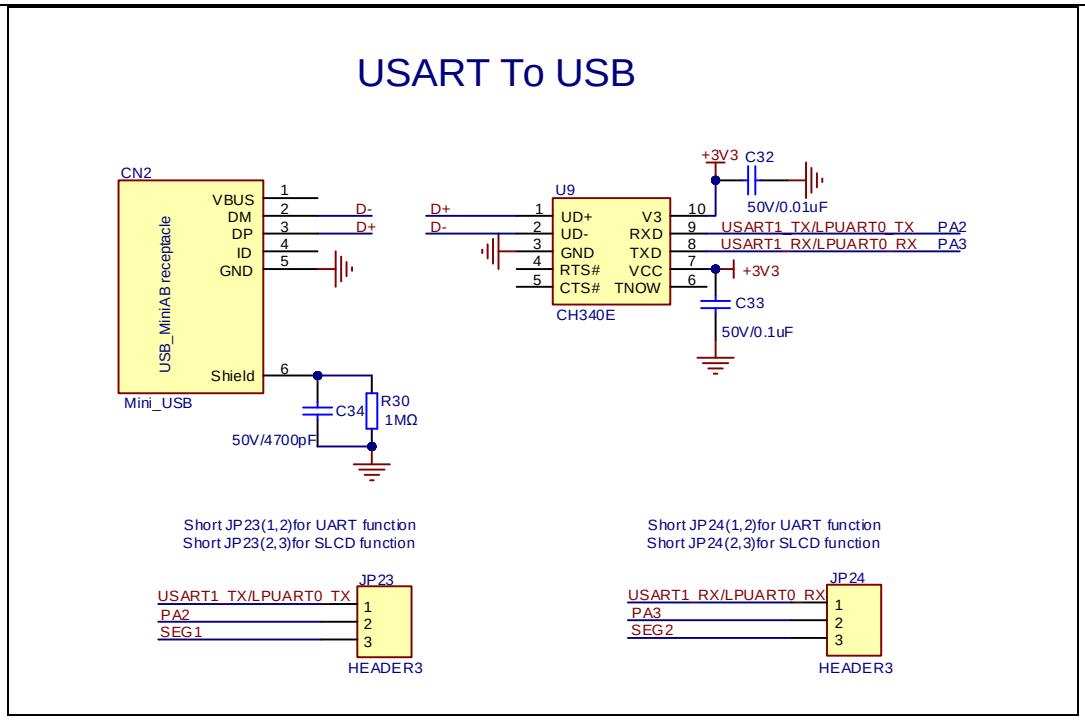
4.8. CAN

Figure 4-8. Schematic diagram of CAN



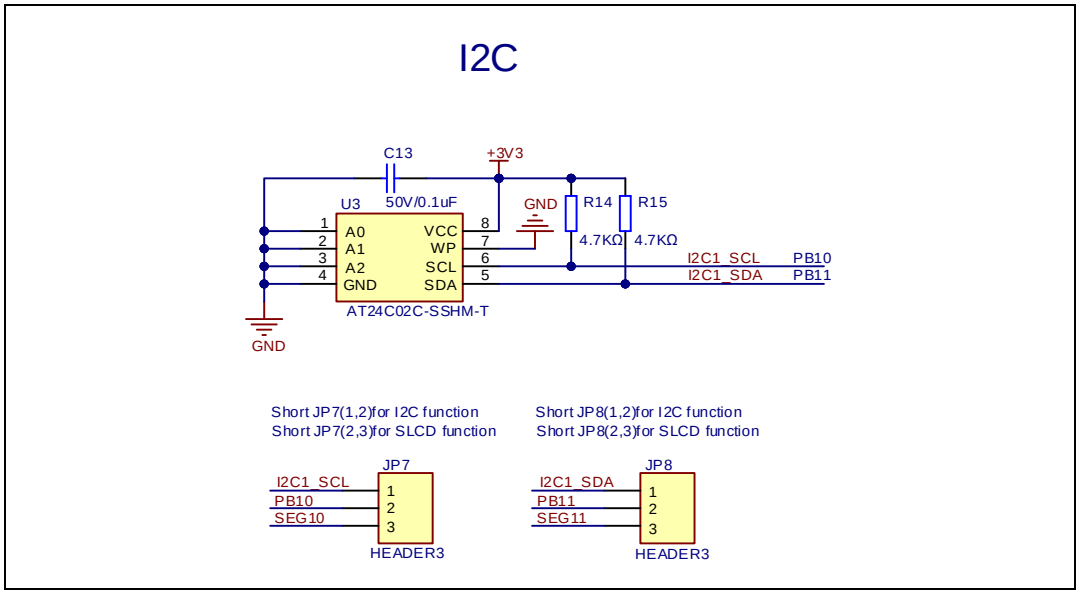
4.9. USART

Figure 4-9. Schematic diagram of USART



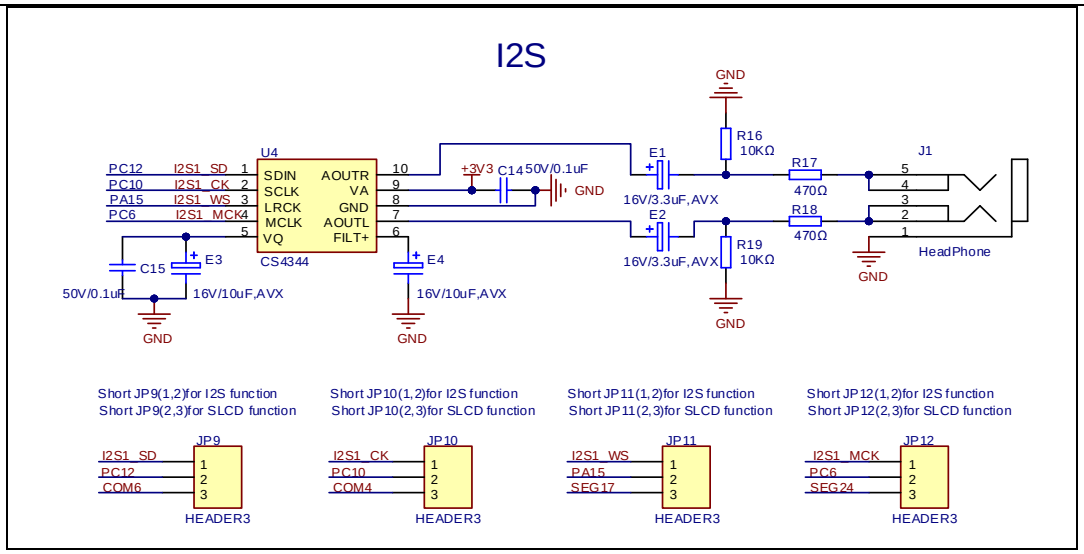
4.10. I2C

Figure 4-10. Schematic diagram of I2C



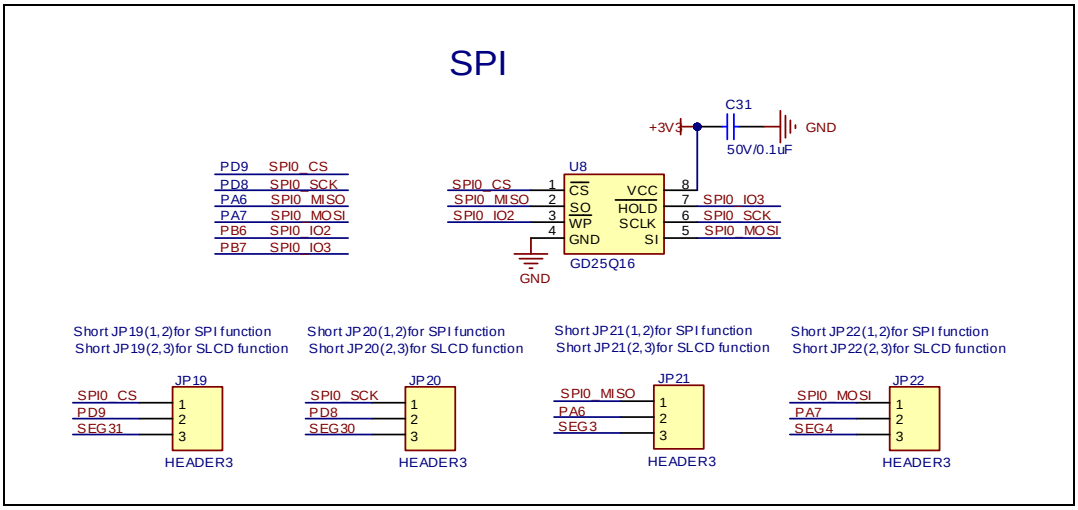
4.11. I2S

Figure 4-11. Schematic diagram of I2S



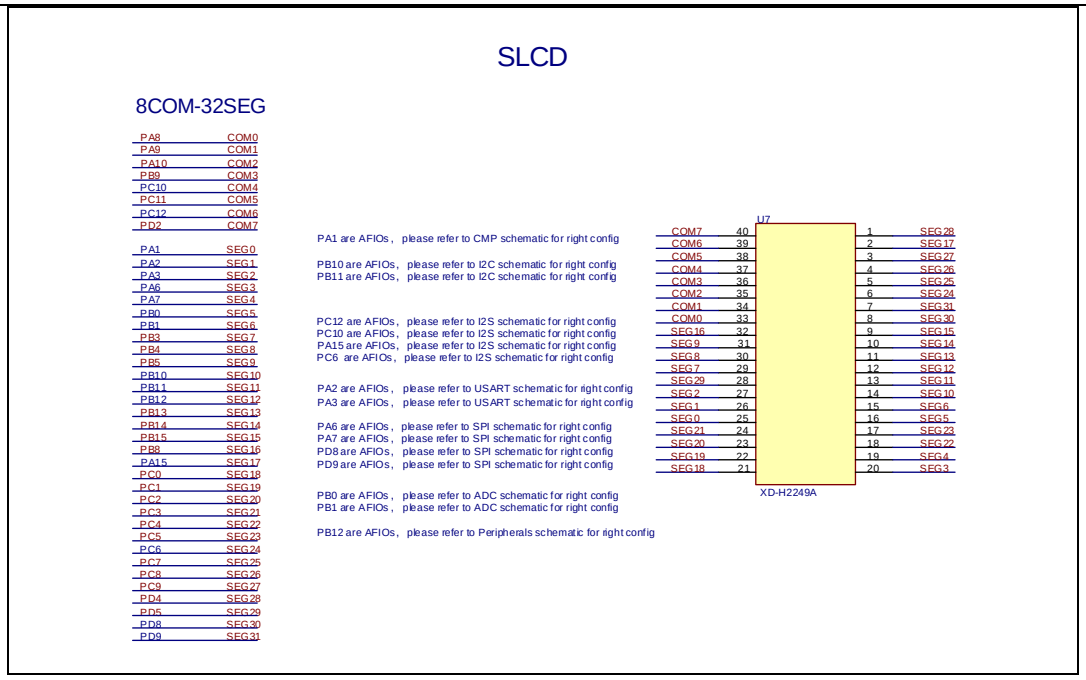
4.12. SPI

Figure 4-12. Schematic diagram of SPI



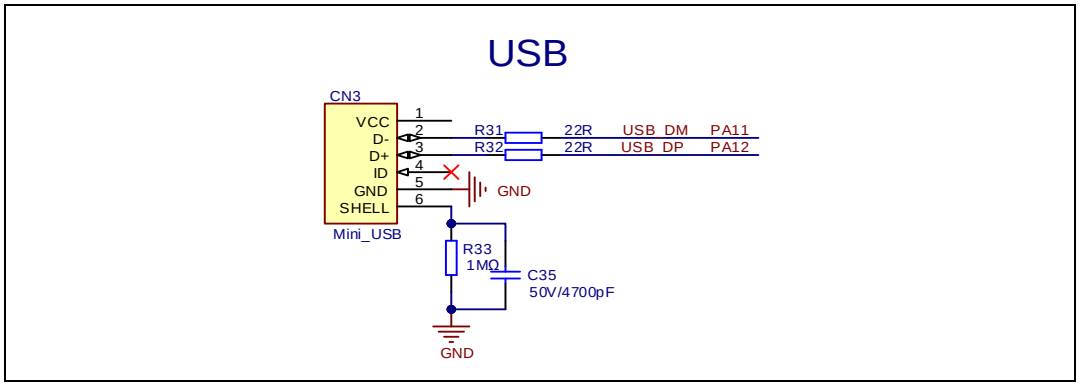
4.13. SLCD

Figure 4-13. Schematic diagram of SLCD



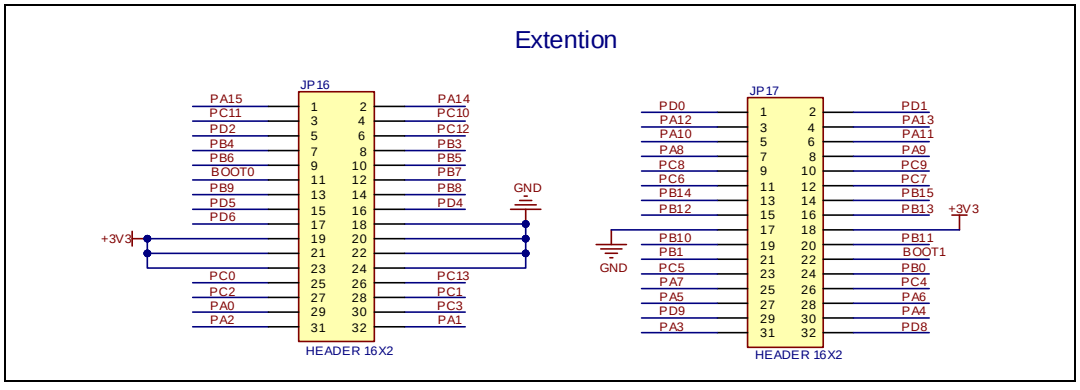
4.14. USB

Figure 4-14. Schematic diagram of USB



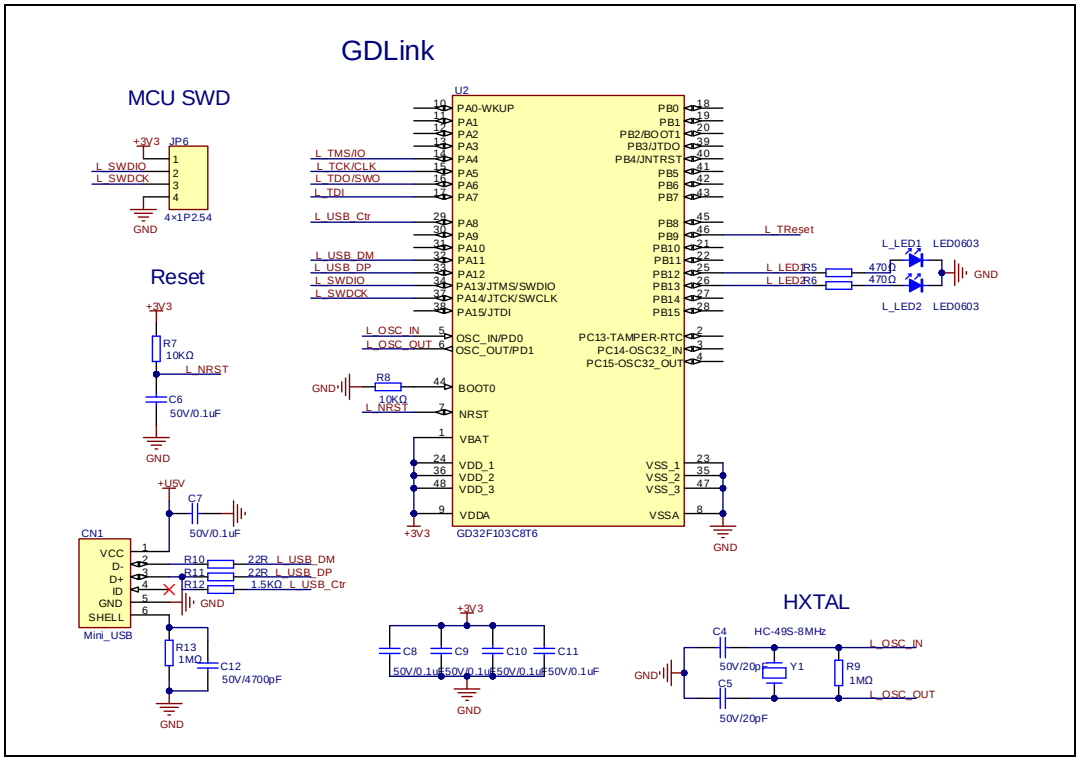
4.15. Extension

Figure 4-15. Schematic diagram of Extention



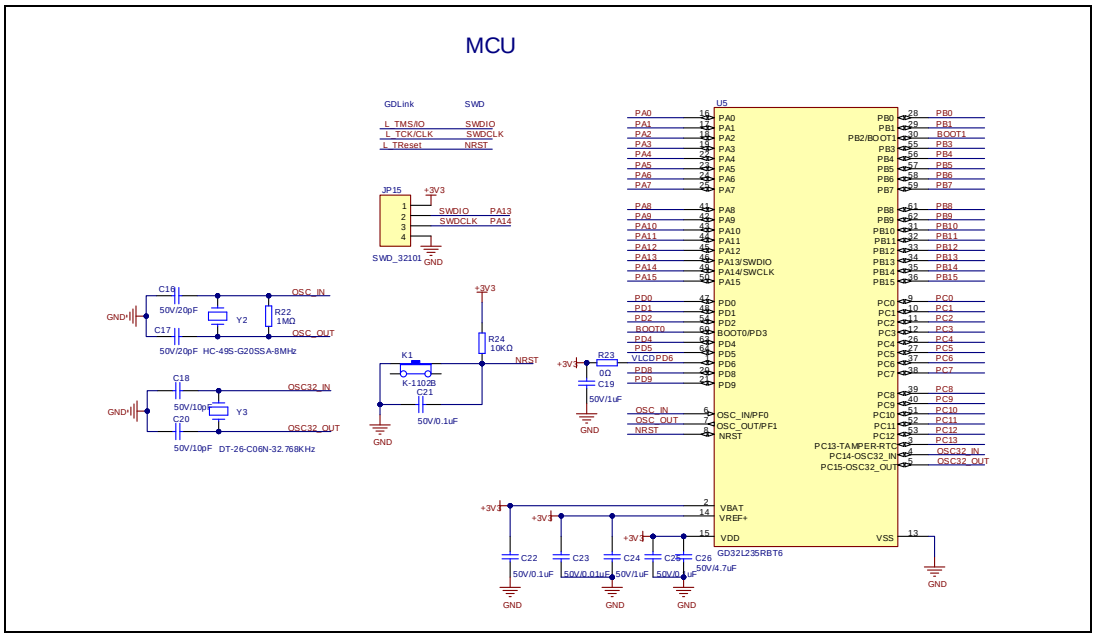
4.16. GD-Link

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

GD32L235R-EVAL board has two user keys and two LEDs. The keys are Tamper key, and Wakeup key. 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, two 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

GD32L235R-EVAL board has two user keys and two LEDs. The keys are Tamper key, and Wakeup key. The LEDs are controlled by GPIO.

This demo will show how to use the Tamper key to control the LED2. 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 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 Tamper 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.

GD32L235R-EVAL board has two user keys and two LEDs. The keys are Tamper key and Wakeup key. The LED1 and LED2 are controlled by GPIO.

This demo will show how to use the EXTI interrupt line to control the LED2. When press down the Tamper 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 Tamper key, LED2 will be turned on. Press down the Tamper 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 flash 2 times for test. Then, this implementation outputs "USART printf example: please press the Tamper key" on the HyperTerminal using USART. Press the Tamper key, the serial port will output "USART printf example".

The output information via the HyperTerminal is as following:

```
USART printf example: please press the Tamper key

USART printf example
```

5.5. USART_HyperTerminal_Interrupt

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_HyperTerminal_Interrupt> to the EVAL board and 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 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 flash by turns. Otherwise, LED1, LED2 toggle together.

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 and connect serial cable to USART. Firstly, the USART sends "USART DMA interrupt receive and transmit example, please input 10 bytes:" to hyperterminal and waits for receiving 10 bytes data from the hyperterminal that you must send. After MCU receives the data, the USART will continue to

output the received data to the hyper terminal.

The output information via the HyperTerminal is as following:

```
USART DMA interrupt receive and transmit example, please
input 10 bytes:



abcdefghij
```

5.7. LPUART_Deepsleep_Wakeup

5.7.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

- Learn to use the LPUART wakeup the MCU from DEEPSLEEP mode.

5.7.2. DEMO running result

Download the program <07_LPUART_Deepsleep_Wakeup> to the EVAL board and connect serial cable to LPUART(USART). Firstly, the MCU enters DEEPSLEEP mode after the LED2 flashs 2 times, and the LED1 stop flashing. The MCU is waken up after HyperTerminal send any character, and LED1 keeps flashing.

5.8. ADC_Temperature_Vrefint

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 get the value of inner channel 16 (temperature sensor channel) and channel 17 (Vrefint channel)

5.8.2. DEMO running result

Download the program <08_ADC_Temperature_Vrefint> to the GD32L235R-EVAL board. Connect serial cable to USART1, open the HyperTerminal.

When the program is running, HyperTerminal display the value of temperature and internal voltage reference.

```
the Temperature data is 31 degrees Celsius
the Reference voltage data is 1.171V

the Temperature data is 31 degrees Celsius
the Reference voltage data is 1.170V

the Temperature data is 31 degrees Celsius
the Reference voltage data is 1.170V

the Temperature data is 31 degrees Celsius
the Reference voltage data is 1.171V

the Temperature data is 31 degrees Celsius
the Reference voltage data is 1.171V

the Temperature data is 31 degrees Celsius
the Reference voltage data is 1.171V
```

5.9. DAC_Output_Voltage_Value

5.9.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

- Learn to use DAC to output voltage on DAC0_OUT0 output

5.9.2. DEMO running result

Download the program <09_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 (VREF/2), would be output on PA4.

The voltage on PA4 can be observed through the oscilloscope.

5.10. Comparator_Obtain_Brightness

5.10.1. 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 PA1, 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 performs the corresponding action.

5.10.2. DEMO running result

Download the program <10_Comparator_Obtain_Brightness> to the EVAL board, comparing two input voltage, if output level is high, LED2 is on, otherwise LED2 is off.

5.11. I2C_EEPROM

5.11.1. 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

5.11.2. DEMO running result

Download the program <11_I2C_EEPROM> to the EVAL board and run. Connect serial cable to USART, 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 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!
```

5.12. SPI_QUAD_FLASH

5.12.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

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

5.12.2. 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.

Download the program <12_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.

```
################################################################################
GD32L235R-EVAL System is Starting up...
GD32L235R-EVAL Flash:128K
GD32L235R-EVAL The CPU Unique Device ID:[FFFFFFFF-FFFFFFFF-FFFFFFFF]
GD32L235R-EVAL 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!
```

5.13. I2S_Audio_Player

5.13.1. 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

GD32L235R-EVAL board integrates the I2S (Inter-IC Sound) module, and the 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.

5.13.2. DEMO running result

Download the program<13_I2S_Audio_Player>to the EVAL board, insert the headphone into the audio port, and then listen to the audio file.

5.14. TRNG_Get_Random

5.14.1. 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

5.14.2. DEMO running result

Download the program <14_TRNG_Get_Random> to the EVAL board and run. Connect serial cable to USART, 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 0x15
Generate random num2 is 0x27
Please input min num (hex format, the range is 0~0xFF):
```

5.15. CAU

5.15.1. 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.

5.15.2. DEMO running result

Download the program <15_CAU> to the EVAL board and run. 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
```

5.16. CAN_Network

5.16.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

- Learn to use the CAN communication between two boards

The EVAL development board integrates the CAN(Controller Area Network) bus controller, which is a common industrial control bus. CAN bus controller follows the CAN bus protocol of 2.0 A and 2.0 B. This demo mainly shows how to communicate two EVAL boards through CAN.

5.16.2. DEMO running result

This example is tested with two EVAL boards. Jump the JP23, JP24 to USART with the jumper cap. Connect L pin to L pin and H pin to H pin on the boards for sending and receiving frames. Download the program <16_CAN_Network> to the two EVAL boards, and connect serial cable to EVAL_COM. Firstly, the EVAL_COM 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 LED2 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: a0 a1 a2 a3 a4 a5 a6 a7
```

5.17. RCU_Clock_Out

5.17.1. 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

5.17.2. DEMO running result

Download the program <17_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 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 key to select clock output source
CK_OUT: system clock, DIV: 4
CK_OUT: IRC16M, DIV: 1
CK_OUT: IRC48M, DIV: 1
CK_OUT: IRC32K, DIV: 1
CK_OUT: LXTAL, DIV: 1
CK_OUT: HXTAL, DIV: 1
CK_OUT: PLL/2, DIV: 1
```

5.18. CTC_Calibration

5.18.1. 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.

5.18.2. DEMO running result

Download the program <18_CTC_Calibration> to the EVAL board and run. If the clock trim is OK, LED2 will be on. Otherwise, LED2 will be turned off.

5.19. PMU_Sleep_Wakeup

5.19.1. 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

5.19.2. DEMO running result

Download the program < 19_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 stop 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.

5.20. RTC_Calendar

5.20.1. 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

5.20.2. DEMO running result

Download the program <20_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.

```
  ****************** RTC calendar demo ******************

=======Configure RTC Time========

  please set the last two digits of current year:

  2021

  please input month:


  08

  please input day:


  12

  please input hour:


  12

  please input minute:


  12

  please input second:


  12


** RTC time configuration success! **

Current time: 2021-08-12 : 12:12:12

Current time: 2021-08-12 : 12:12:12
```

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 TIMER channel value

5.21.2. DEMO running result

Download the program <21_TIMER_Breath_LED> to the GD32L235R-EVAL board and run.

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.

5.22. LPTIMER_Deepsleep_Pwmout

5.22.1. DEMO purpose

This Demo includes the following functions of GD32 MCU:

- Learn to use LPTIMER output PWM wave
- Learn to use the LPTIMER interrupts to wake up the PMU from sleep mode

5.22.2. DEMO running result

Download the program <22_LPTIMER_Deepsleep_Pwmout> to the GD32L235R-EVAL board and run. When the program is running, you can see LED1 sparks and the LPTIMER0_O (PA4) outputs the PWM signal.

Press key Wakeup (K3) to enter deepsleep mode, LED1 stops in a certain status (on or off). When the LPTIMER0 count value matches the value of compare register or auto reload register, MCU will be wakeup from deepsleep mode by LPTIMER0, the transfer goes on normally and LED1 sparks again. During this period, LPTIMER0_O (PA4) always outputs the PWM signal.

5.23. SLCD_Glass

5.23.1. DEMO purpose

This Demo includes the following functions of GD32 MCU:

- Learn to use SLCD module to display number

5.23.2. DEMO running result

Download the program <23_SLCD_Glass> to the EVAL board and run. When the program is running, you can see the SLCD displaying the number which adds 1 per second.

5.24. USBD_Keyboard

5.24.1. DEMO_purpose

This demo includes the following functions of GD32 MCU:

- Learn how to use the USBD peripheral mode

■ Learn how to implement USB HID (human interface) device

The GD32L235R-EVAL board is enumerated as an USB Keyboard, which uses the native PC Host HID driver, as shown below. The USB Keyboard uses Tamper key and Wakeup key to output two characters ('a', 'b'). 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 'b' key is used as the remote wakeup source.



5.24.2. DEMO running result

Download the program <24_USBD_Keyboard> to the EVAL board and run. If you press the Tamper key, will output 'a'. If you press the Wakeup key, will output 'b'.

If user want to test USB remote wakeup function, user 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.

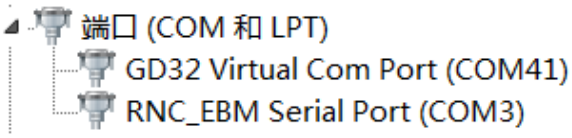
5.25. USBD_CDC_ACM

5.25.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

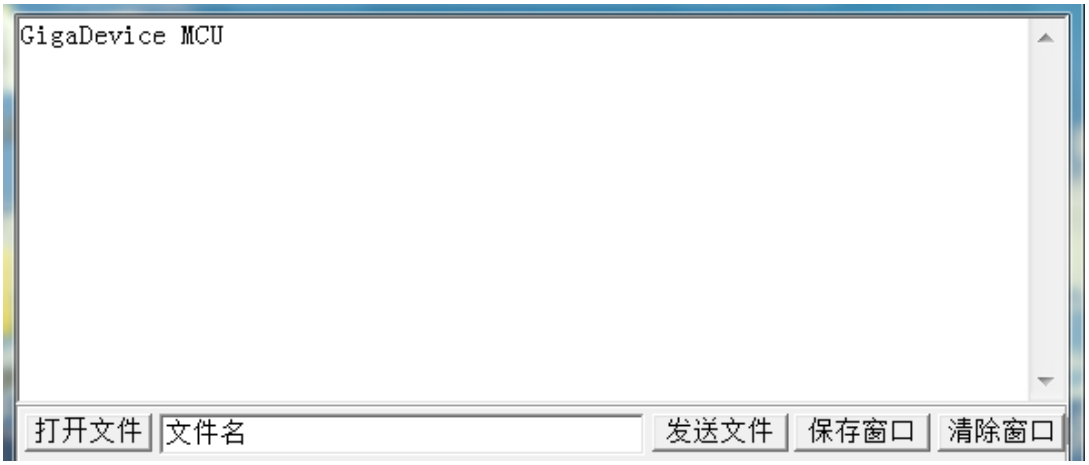
■ Learn how to use the USBD peripheral
■ Learn how to implement USB CDC device

GD32L235R-EVAL board has one USBD interface. In this demo, the GD32L235R-EVAL 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.



5.25.2. DEMO running result

Download the program <25_USBD_CDC_ACM> to the EVAL board and run. When user input message through computer keyboard, the HyperTerminal will receive and shown the message. For example, when you input "GigaDevice MCU", the HyperTerminal will get and show it as below.



6. Revision history

Table 6-1. Revision history

Revision No.	Description	Date
1.0	Initial Release	Jul.18, 2023
1.1	Update the font displayed in the document	Aug.21,2026

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.

© 2026 GigaDevice Semiconductor Inc. - All rights reserved