GigaDevice Semiconductor Inc.

GD32E103V-EVAL
Arm® Cortex®-M4 32-bit MCU

User Guide

Revision 1.3

(Aug. 2026)

Table of Contents

Table of Contents **1**
List of Tables **3**
1 Summary **4**
2 Function Pin Assign **4**
3 Getting started **5**
4 Hardware layout overview **6**
4.1 Power 6
4.2 Boot 6
4.3 LED 7
4.4 KEY 7
4.5 USART 8
4.6 ADC 8
4.7 DAC 8
4.8 I2S 9
4.9 I2C 9
4.10 SPI 9
4.11 LCD 10
4.12 USBFS 10
4.13 Extension 11
4.14 GD-Link 11
5 Routine use guide **11**
5.1 GPIO_Runing_Led 11
5.2 GPIO_Key_Polling_mode 12
5.3 EXTI_Key_Interrupt_mode 12
5.4 USART_Printf 13
5.5 USART_Echo_Interrupt_mode 14
5.6 USART_DMA 14
5.7 ADC_Temperature_Vrefint 15
5.8 ADC0_ADC1_Follow_up_mode 16
5.9 ADC0_ADC1_Regular_Parallel_mode 17
5.10 DAC_Output_Voltage_Value 18
5.11 I2C_EEPROM 18
5.12 SPI_SPI_Flash 19
5.13 I2S_Audio_Player 21
5.14 EXMC_TouchScreen 21
5.15 RCU_Clock_Out 22
5.16 CTC_Calibration 23
5.17 PMU_sleep_wakeup 23
5.18 RTC_Calendar 24
5.19 TIMER_Breath_LED 25
5.20 USBFS 25

5.21 LiteOS 28
6 Revision history 30

List of Tables

Table 1. Function pin assign 4
Table 2. Revision history 30

1 Summary

GD32E103V-EVAL uses GD32E103VBT6 as the main controller. It uses Mini USB interface or DC-005 connector to supply 5V power. SWD, Reset, Boot, User button key, LED, I2C, I2S, USART, RTC, LCD, SPI, ADC, DAC, EXMC, CTC, USB, GD-Link and Extension Pins are also included. For more details please refer to GD32E103V-EVAL-Rev2.0 schematic.

2 Function Pin Assign

Table 1. Function pin assign

Function	Pin	Description
LED	PC0	LED2
	PC2	LED3
	PE0	LED4
	PE1	LED5
RESET		K1-Reset
KEY	PA0	K2-Wakeup
	PC13	K3-Tamper
	PB14	K4-User key
USART0	PA9	USART0_TX
	PA10	USART0_RX
USART1	PA2	USART1_TX
	PA3	USART1_RX
ADC	PC3	ADC01_IN13
DAC	PA4	DAC_OUT0
	PA5	DAC_OUT1
I2C	PB6	I2C0_SCL
	PB7	I2C0_SDA
SPI	PA5	SPI0_SCK
	PA6	SPI0_MISO
	PA7	SPI0_MOSI
	PE3	SPIFlash_CS
I2S	PB12	I2S1_WS
	PB13	I2S1_CK
	PB15	I2S1_SD
	PC6	I2S1_MCK
LCD	PD14	EXMC_D0
	PD15	EXMC_D1
	PD0	EXMC_D2
	PD1	EXMC_D3

Function	Pin	Description
	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
	PD7	EXMC_NE
USBFS	PA9	USB_VBUS
	PA11	USB_DM
	PA12	USB_DP
	PA10	USB_ID

3 Getting started

The EVAL board uses Mini USB connecter 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 LED1 will turn on, which indicates that the power supply is OK.

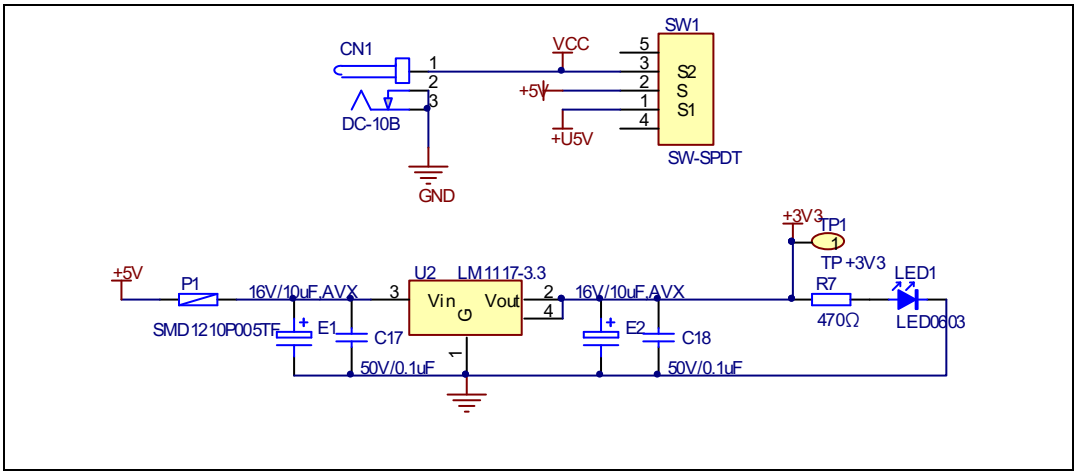
There are Keil version, IAR version and GD32EmbeddedBuilder of all projects. Keil version of the projects are created based on Keil MDK-ARM 4.74 uVision4. IAR version of the projects are created based on IAR Embedded Workbench for ARM 7.40.2. During use, the following points should be noted:

1. If you use Keil uVision4 to open the project, install the GD32E10x_AddOn.1.2.0.exe to load the associated files.
2. If you use Keil uVision5 to open the project, there are two ways to solve the "Device Missing (s)" problem. One is to install GigaDevice.GD32E10x_DFP.1.2.0.pack. In Project menu, select the Manage sub menu, click on the "Version Migrate 5 Format..." menu, the Keil uVision4 project will be converted to Keil uVision5 project. Then add "C:\Keil_v5\ARM\Pack\ARM\CMSIS\4.2.0\CMSIS\Include" to C/C++ in Option for Target. The other is to install Addon directly. Select the installation directory of Keil uVision5 software, such as C:\Keil_v5, in Destination Folder of Folder Selection. Select the

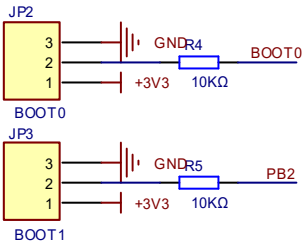
corresponding device in Device of Option for Target and add "C:\Keil_v5\ARM\Pack\ARM\CMSIS\4.2.0\CMSIS\Include" to C/C++ in Option for Target.
3. If you use IAR to open the project, install IAR_GD32E10x_ADDON.1.2.0.exe to load the associated files.

4 Hardware layout overview

4.1 Power

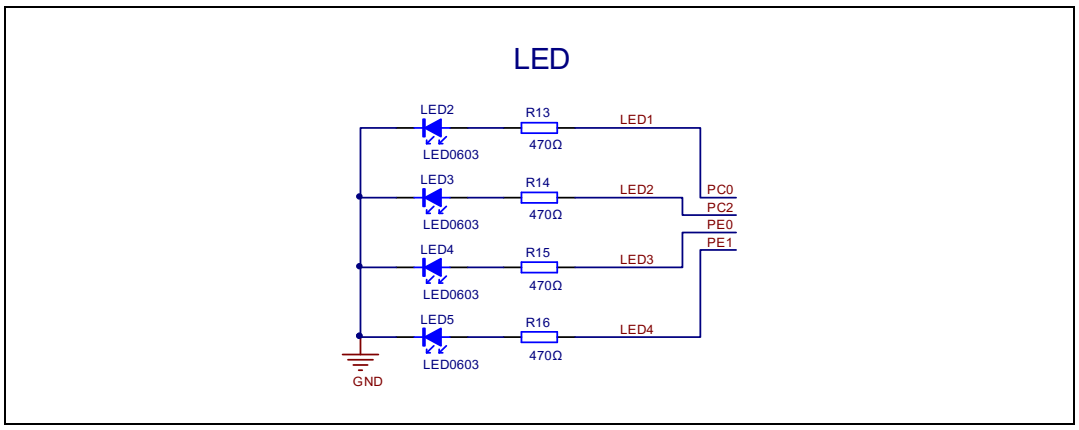


4.2 Boot

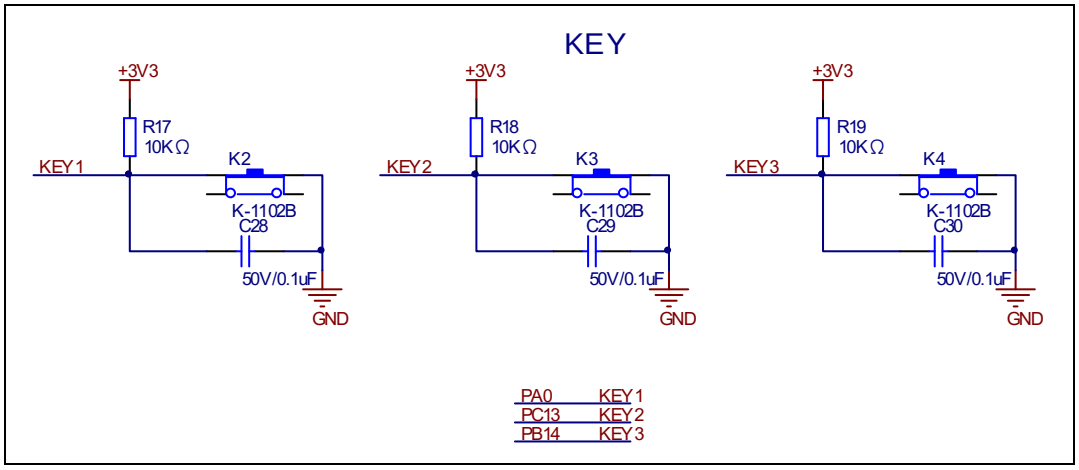


BOOT1	BOOT0	Boot Mode
Any	2-3	User memory
2-3	1-2	System memory
1-2	1-2	SRAM memory

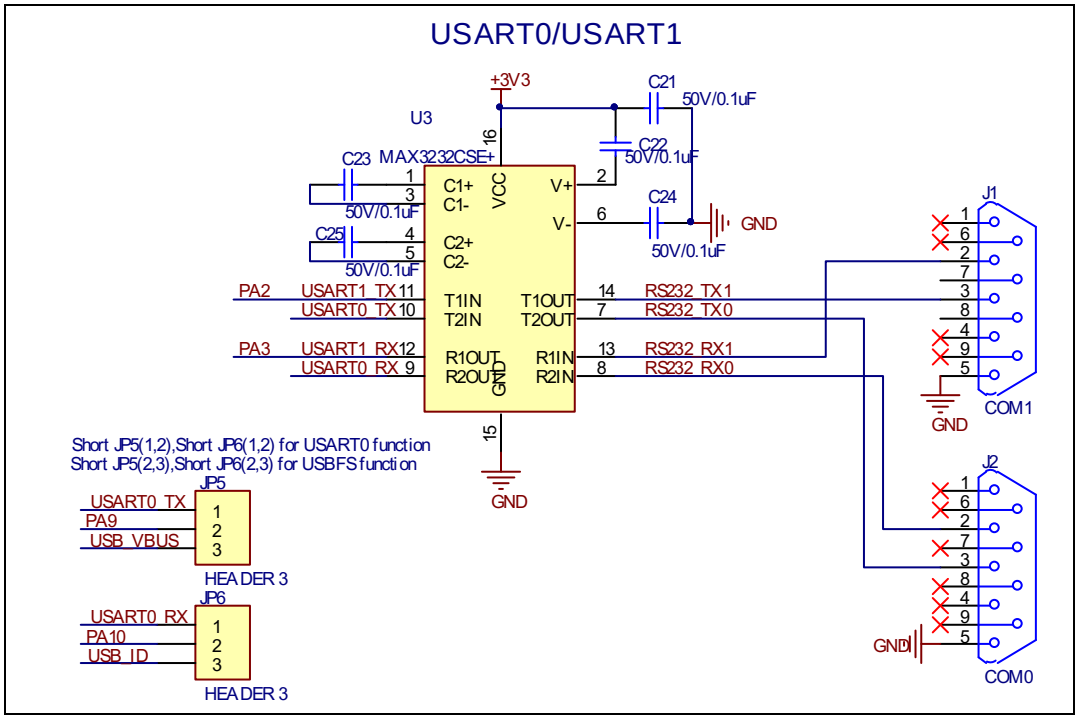
4.3 LED


LED
LED2
R13
LED1
LED0603
470Ω
LED3
R14
LED2
PC0
PC2
LED0603
470Ω
PE0
LED4
R15
PE1
LED3
LED0603
470Ω
LED5
R16
LED4
LED0603
470Ω
GND

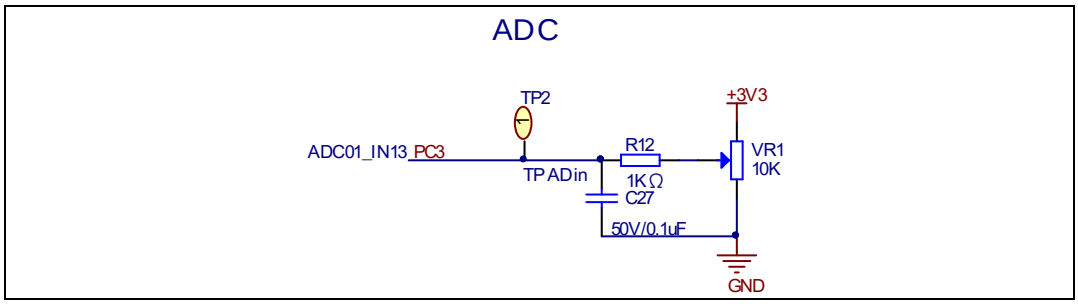
4.4 KEY


KEY
+3V3
R17
10KΩ
KEY1
K2
K-1102B
C28
50V/0.1uF
GND
+3V3
R18
10KΩ
KEY2
K3
K-1102B
C29
50V/0.1uF
GND
+3V3
R19
10KΩ
KEY3
K4
K-1102B
C30
50V/0.1uF
GND
PA0
KEY1
PC13
KEY2
PB14
KEY3

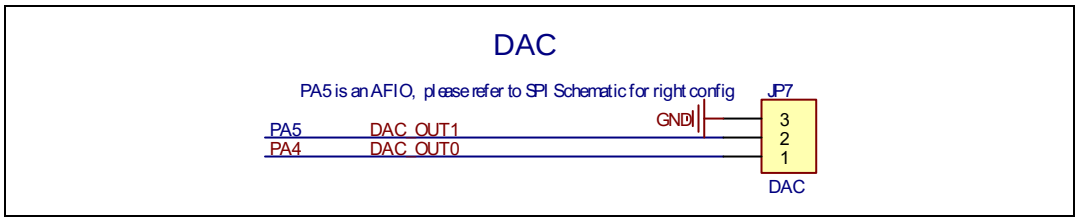
4.5 USART



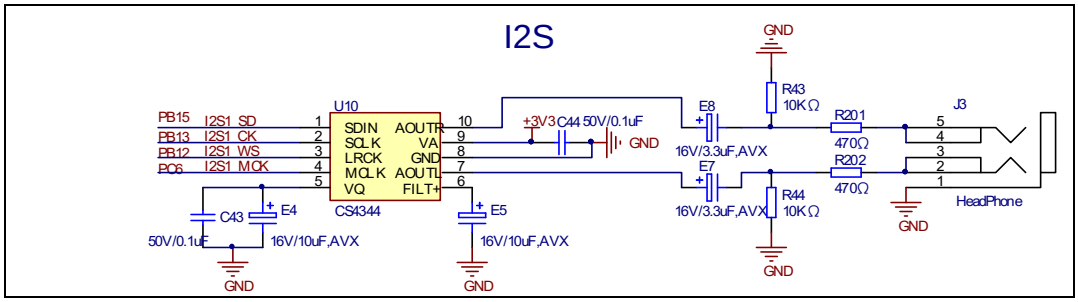
4.6 ADC



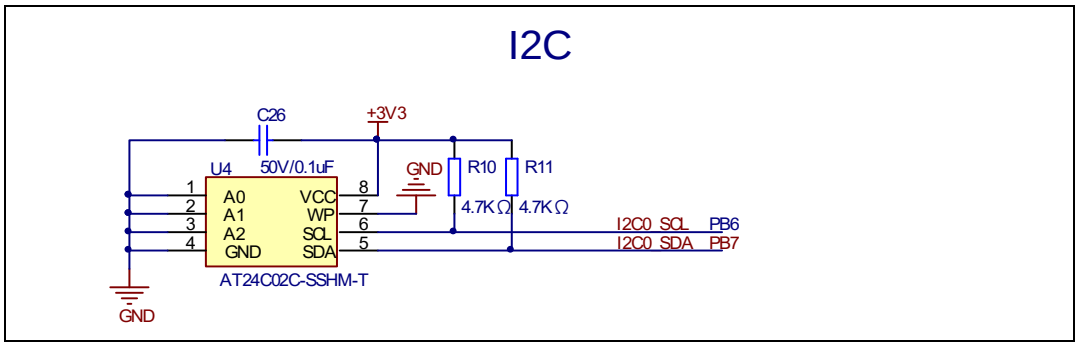
4.7 DAC



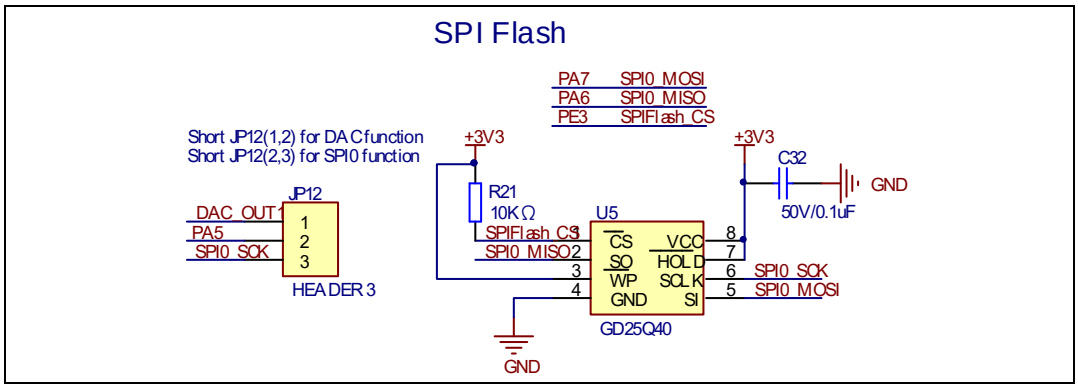
4.8 I2S



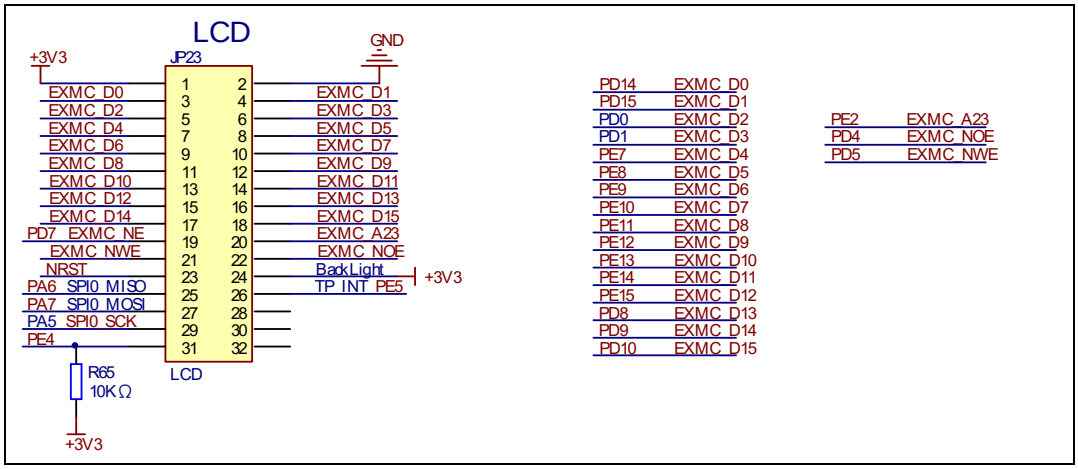
4.9 I2C



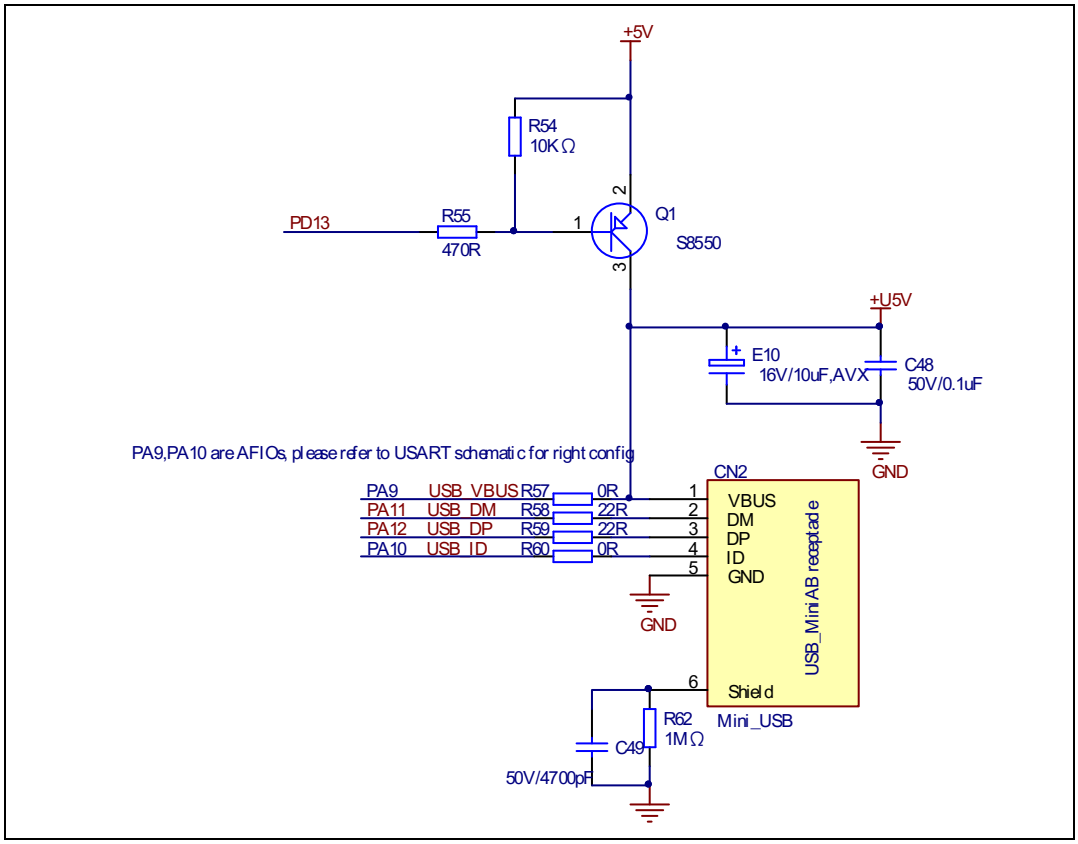
4.10 SPI



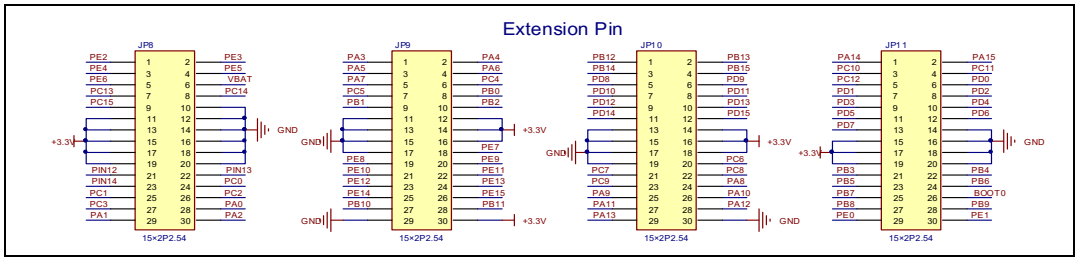
4.11 LCD



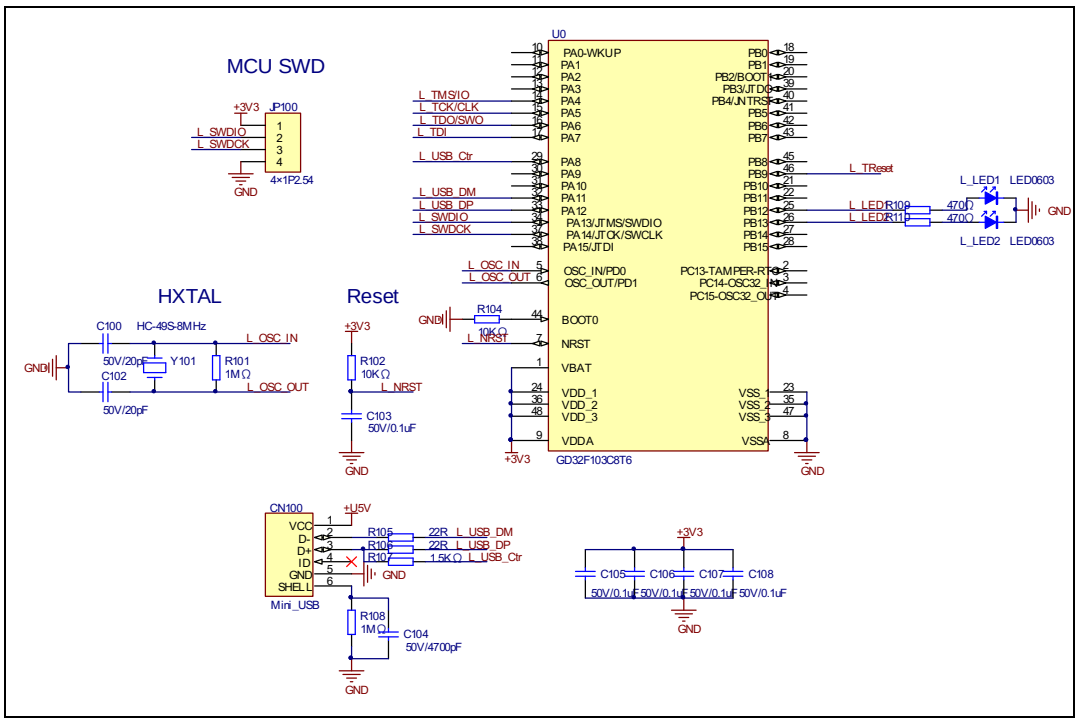
4.12 USBFS



4.13 Extension



4.14 GD-Link



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

GD32E103V-EVAL-V2.0 board has four LEDs. The LED2, LED3, LED4 and LED5 are controlled by GPIO. This demo will show how to light the LEDs.

5.1.2 DEMO Running Result

Download the program < 01_GPIO_Runing_Led > to the EVAL board, LED2, LED3, LED4 will turn on in sequence with interval of 200ms, and turn off together, 200ms 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

GD32E103V-EVAL-V2.0 board has four keys and four LEDs. The four keys are Reset key, Tamper key, Wakeup key and User key. The LED2, LED3, LED4 and LED5 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 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 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

GD32E103V-EVAL board has four keys and four LEDs. The four keys are Reset key, Tamper key, Wakeup key and User key. The LED2, LED3, LED4 and LED5 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. After startup, the LED2 flash once, 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 EVAL_COM0 and jump JP5 and JP6 to USART0. This implementation outputs "USART printf example: please press the Tamper key" on the HyperTerminal using EVAL_COM0. Press the Tamper key, serial port will output "USART printf example".
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 EVAL_COM0 and jump JP5 and JP6 to USART0. Firstly, all the LEDs are turned on and off for test. Then, the EVAL_COM0 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, LED2, LED3, LED4, LED5 flash by turns. Otherwise, LED2, LED3, LED4, LED5 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 EVAL_COM0 and jump JP5 and JP6 to USART0. Firstly, all the LEDs are turned on and

off for test. Then, the EVAL_COM0 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, LED2, LED3, LED4, LED5 flash by turns. Otherwise, LED2, LED3, LED4, LED5 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 (VREFINT channel)

5.7.2 DEMO Running Result

Download the program <07_ADC_Temperature_Vrefint> to the GD32E103V-EVAL-V2.0 board. Connect serial cable to EVAL_COM0, 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 1.200V

the temperature data is 30 degrees Celsius
the reference voltage data is 1.203V

the temperature data is 29 degrees Celsius
the reference voltage data is 1.201V

the temperature data is 29 degrees Celsius
the reference voltage data is 1.202V

the temperature data is 29 degrees Celsius
the reference voltage data is 1.202V

the temperature data is 29 degrees Celsius
the reference voltage data is 1.202V
```

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 GD32E103V-EVAL-V2.0 board. Connect serial cable to EVAL_COM0, open the HyperTerminal. PC3 and PC5 pin voltage access by external voltage.

TIMER0_CH0 is the trigger source of ADC0 and ADC1. When the rising edge of TIMER0_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 PC5 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 PC3 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 PC3 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 PC5 pin is stored into the low 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].

```
the data adc_value[0] is 00040711
the data adc_value[1] is 070C0009

the data adc_value[0] is 00000713
the data adc_value[1] is 070A0000

the data adc_value[0] is 00060713
the data adc_value[1] is 070A0000

the data adc_value[0] is 00030715
the data adc_value[1] is 070C0000

the data adc_value[0] is 00030710
the data adc_value[1] is 070D0000

the data adc_value[0] is 00000711
the data adc_value[1] is 070C0006
```

5.9 ADC0_ADC1_Regular_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 regular parallel mode

5.9.2 DEMO Running Result

Download the program <09_ADC0_ADC1_Regular_Parallel_mode> to the GD32E103V-EVAL-V2.0 board. Connect serial cable to EVAL_COM0, open the HyperTerminal. PC3 and PC5 pin connect to external voltage input.

TIMER0_CH0 is the trigger source of ADC0 and ADC1. When the rising edge of TIMER0_CH0 coming, ADC0 and ADC1 convert the regular channel group parallelly. 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 ADC0 conversion of PC3 pin is stored into the low half word of adc_value[0], the value of the ADC1 conversion of PC5 pin is stored into the high half word of adc_value[0]. When sampling the second channel of ADCx (x=0,1), the value of the ADC0 conversion of PC5 pin is stored into the low half word of adc_value[1], the value of the ADC1 conversion of PC3 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].

```
the data adc_value[0] is 00000714
the data adc_value[1] is 07140000

the data adc_value[0] is 00050714
the data adc_value[1] is 07160000

the data adc_value[0] is 00040711
the data adc_value[1] is 07130000

the data adc_value[0] is 00000715
the data adc_value[1] is 07130001

the data adc_value[0] is 00000715
the data adc_value[1] is 07130002

the data adc_value[0] is 00060713
the data adc_value[1] is 07130000
```

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 (VREF/2), would be output on PA4.

The voltage on PA4 can be observed through the oscilloscope.

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 COM0, jump the P4 I2C and jump JP5 to USART, then open the HyperTerminal to show the print message.

Firstly, the data of 256 bytes will be written to the EEPROM from the address 0x00 and printed by the serial port. Then, reading the EEPROM from address 0x00 for 256 bytes and the result will be printed. Finally, compare the data that were written to the EEPROM and the data that were read from the EEPROM. If they are the same, the serial port will output "I2C-AT24C02 test passed!" and the four LEDs lights flashing, 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 I2C0 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!
```

5.12 SPI_SPI_Flash

5.12.1 DEMO Purpose

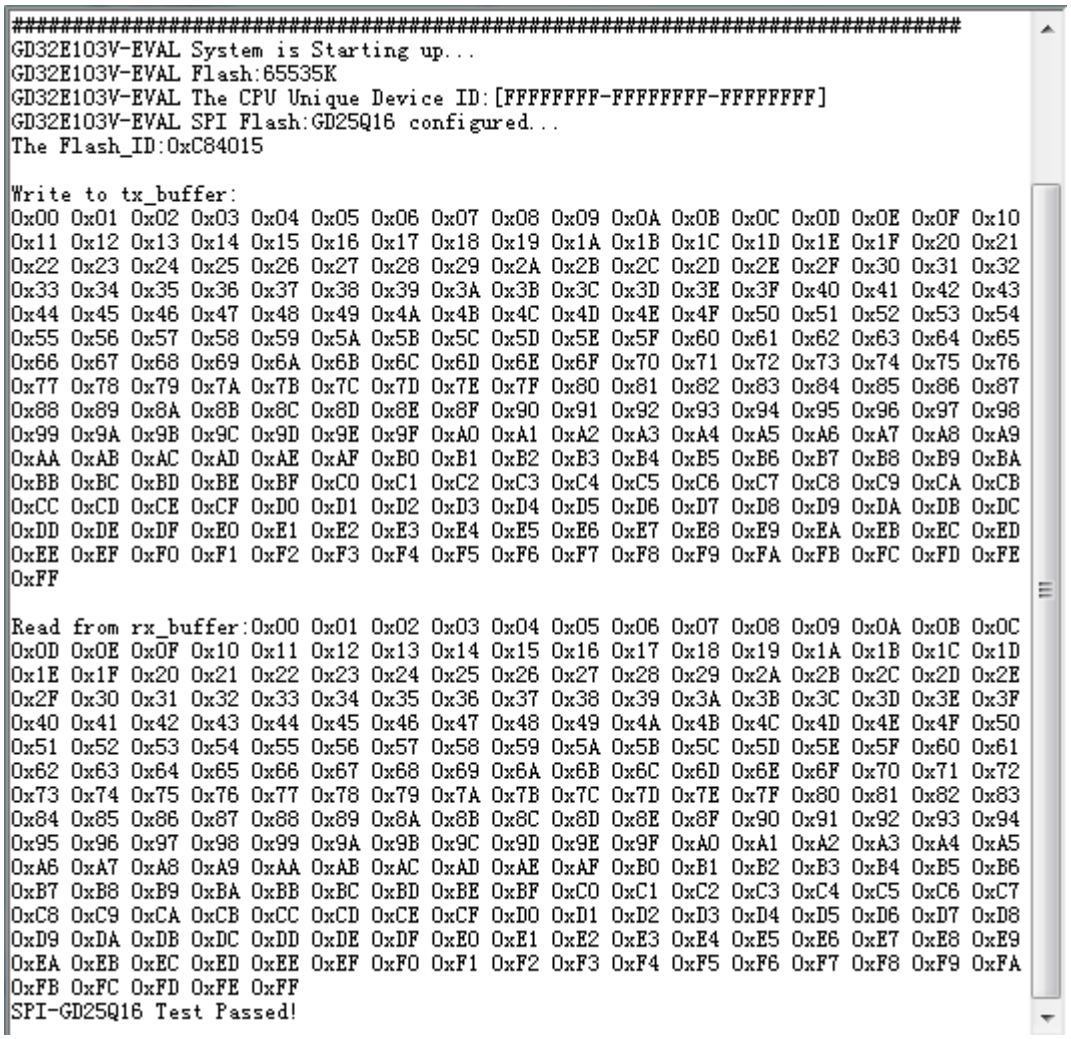
This demo includes the following functions of GD32 MCU:

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

5.12.2 DEMO Running Result

The computer serial port line connected to the COM0 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 JP12 to SPI, jump JP5 to USART.

Download the program <12_SPI_SPI_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.



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

GD32E103V-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 EXMC_TouchScreen

5.14.1 DEMO Purpose

This demo includes the following functions of GD32 MCU:

- Learn to use EXMC to control LCD

5.14.2 DEMO Running Result

GD32E103V-EVAL board has EXMC module to control LCD. Before running the demo, JP12 must be fitted to the SPI port, P2 and P3 must be fitted to the EXMC port, JP23 must be fitted to the Lcd port. Download the program <14_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.



5.15 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 COM0, 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
```

5.16 CTC_Calibration

5.16.1 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.

5.16.2 DEMO Running Result

Download the program <16_CTC_Calibration> to the GD32E103V-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.

5.17 PMU_sleep_wakeup

5.17.1 DEMO Purpose

This demo includes the following functions of GD32 MCU:

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

5.17.2 DEMO Running Result

Download the program < 17_PMU_sleep_wakeup > to the EVAL board, jump JP5 and JP6 to USART and connect serial cable to COM0. 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.

5.18 RTC_Calendar

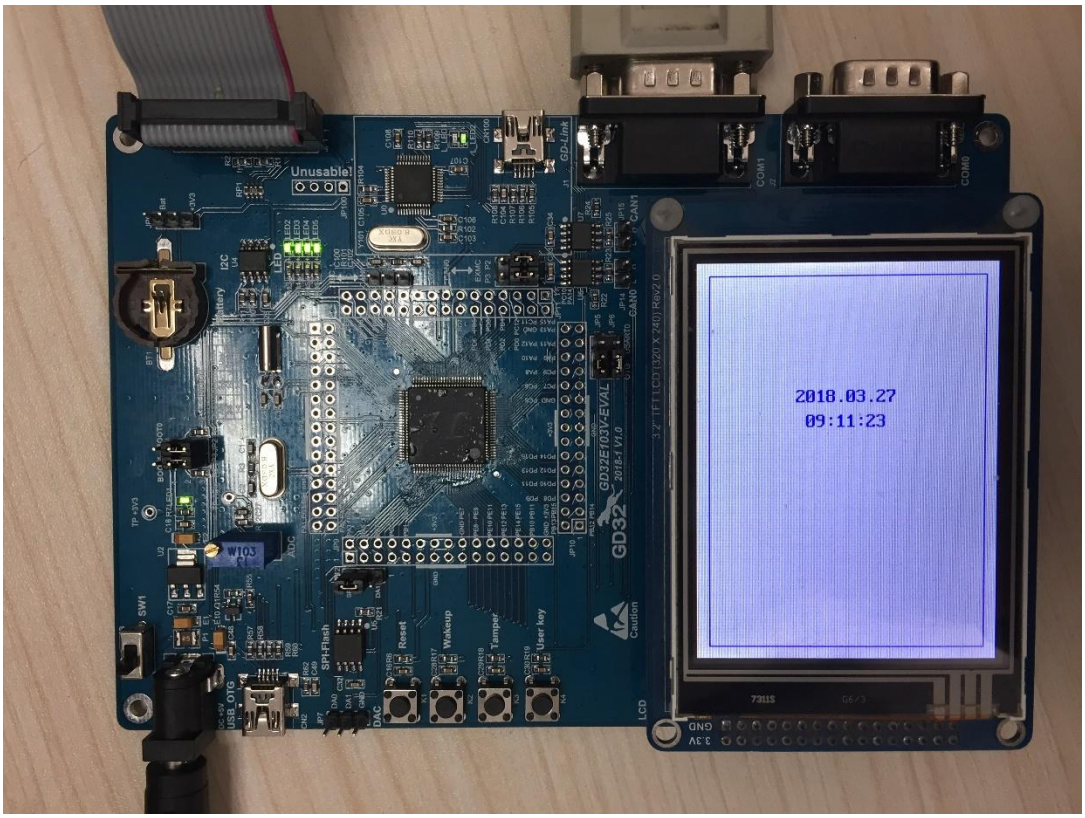
5.18.1 DEMO Purpose

This demo includes the following functions of GD32 MCU:

- Learn to use RTC module to implement calendar and alarm function
- Learn to use EXMC module to implement time display on LCD

5.18.2 DEMO Running Result

Download the program <18_RTC_Calendar> to the EVAL board and run. Jump P2 and P3 to EXMC. Connect serial cable to COM1, open the HyperTerminal. After start-up, the program will ask to set the time on the HyperTerminal. The calendar will be displayed on the LCD after the time has been set. At the same time, the program realizes the alarm function. Every time when the USER key is pressed, an alarm is generated after 10 seconds and all of the LEDs will be toggled at the same time.



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 LED2 (PC0), and then download the program <19_TIMER_Breath_LED> to the GD32E103V-EVAL board and run. PA8 should not be reused by other peripherals.
When the program is running, you can see LED2 lighting from dark to bright gradually and then gradually darken, ad infinitum, just like breathing as rhythm.

5.20 USBFS

5.20.1 HID_Keyboard (Device)

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

GD32E103V-EVAL evaluation board has five keys and one USB_FS interface. The four keys are Reset key, Wakeup key, Tamper key, User key. In this demo, the GD32E103V-EVAL evaluation board is enumerated as an 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

Before running the demo, please ensure that jumper JP5/JP6 jump to USB. After doing this, download the program <20_USBFS\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.

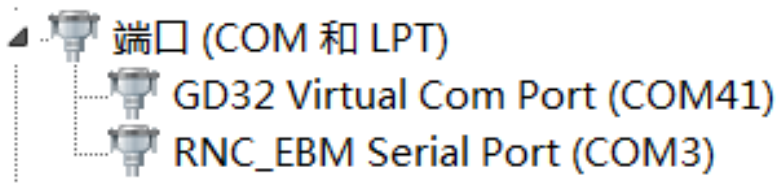
5.20.2 CDC_ACM (Device)

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

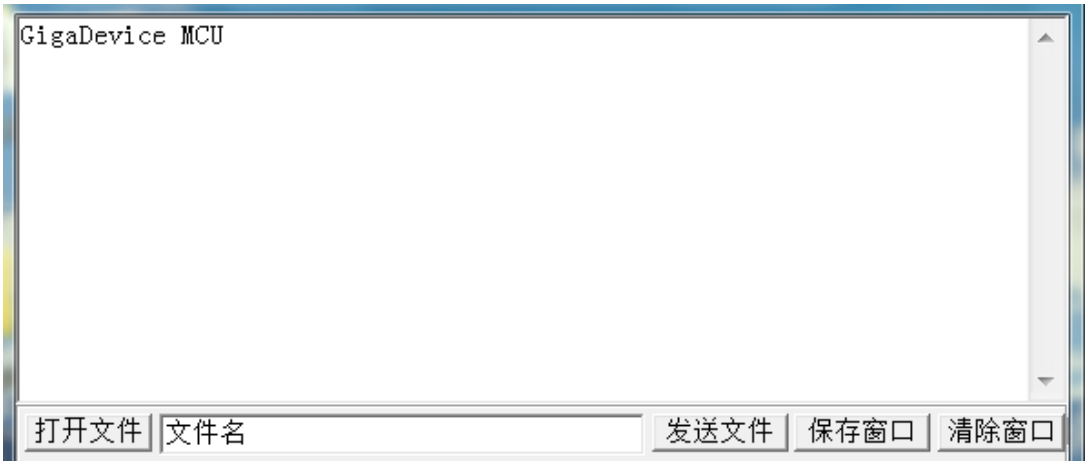
GD32E103V-EVAL board has one USBFS interface. In this demo, the GD32E103V-EVAL board is enumerated as an USB virtual COM port, which was shown in device manager of PC as below. This demo makes the USB device look like a serial port, and loops back the contents of a text file over USB port. To run the demo, input a message using the PC's keyboard. Any data that shows in HyperTerminal is received from the device.



DEMO Running Result

Download the program < 20_USBFS\ USB_Device \CDC_ACM > to the EVAL board and

run. When you input message through computer keyboard, the HyperTerminal will receive and shown the message. For example, when you input "GigaDevice MCU", the HyperTerminal will get and show it as below.



5.20.3 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

GD32E103V-EVAL board integrates the USBFS module, and the module can be used as a USB device, a USB host or an OTG device. This demo mainly shows how to use the USBFS as a USB HID host to communicate with external USB HID device.

DEMO Running Result

Jump the JP5 to OTG. Then download the program <20_USBFS\USB_Host\HID_Host> 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.

5.20.4 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

GD32E103V-EVAL board integrates the USBFS module, and the module can be used as a USB device, a USB host or an OTG device. This demo mainly shows how to use the USBFS as a USB MSC host to communicate with external Udisk.

DEMO Running Result

Jump the JP5 to USB. Then insert the OTG cable to the USB port, download the program <20_USBFS\USB_Host\MSC_Host > 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.

5.21 LiteOS

5.21.1 DEMO Purpose

This demo includes the following functions of GD32 MCU:

- Learn to use the HUAWEI LiteOS RTOS
- Learn to communicate with PC by USART

5.21.2 DEMO Running Result

Download the program <21_LiteOS> to the EVAL board and run. Connect serial cable to COM0, open the HyperTerminal. When the program is running, the LED3 will blink per 1s and print information by COM every 5s. When user press the User key down, LED4 will blink once.
The output information via the serial port is as following:

```
 /============== Gigadevice LiteOS Demo ==============/
LED Task running
Key Task start
LED Task running
LED Task running
press the User key, LED will be respond
LED Task running
LED Task running
press the User key, LED will be respond
LED Task running
LED Task running
LED Task running
the User key has been pressed
```

6 Revision history

Table 2. Revision history

Revision No.	Description	Date
1.0	Initial Release	Dec.26, 2017
1.1	Version update	Sep.27, 2020
1.2	Delete CAN, update the name of routines	Dec.31, 2020
1.3	Update the Important Notice.	Aug.8, 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, aeronautic or aerospace applications, pollution control or hazardous substance management; (ii) life-support systems, other medical equipment or systems (including life support equipment and surgical implants); and/or (iii) 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 Products. 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 Products.

While the Company has implemented advanced security features, the Product may be subject to unidentified vulnerabilities. Customers are responsible for the design and operation of their applications and products to reduce the effect of these vulnerabilities on Customer's applications and products, and to the maximum extent permitted by applicable law, the Company accepts no liability for any vulnerability that is discovered. Customers should implement appropriate design and operating safeguards to minimize the risks associated with their applications and products.

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 - All rights reserved