GigaDevice Semiconductor Inc.

GD32107C-EVAL User Guide V2.2

Table of Contents

List of Tables

1 Summary

GD32107C-EVAL uses GD32F107VCT6 as the main controller. It uses Mini USB 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, ENET, USBFS and Extension Pins are also included. For more details please refer to GD32107C-EVAL-V1.3 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	ADC012_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	SPI0_CS
I2S	PA4	MSEL
	PA5	MCLK
	PA7	MDIN
	PB12	I2S_WS
	PB13	I2S_CK
	PB15	I2S_DIN
	PC6	I2S_MCK
CAN0	PD0	CAN0_RX
	PD1	CAN0_TX

CAN1	PB5	CAN1_RX
	PB6	CAN1_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_NCE1
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
	PD7	EXMC_NE0
ENET	PB11	RMII_TX_EN
	PB12	RMII_TXD0
	PB13	RMII_TXD1
	PC4	RMII_RXD0
	PC5	RMII_RXD1
	PA7	RMII_CRS_DV

	PC1	RMII_MDC
	PA2	RMII_MDIO
	PB0	RMII_INT
	PA1	RMII_REF_CLK
USBFS	PA9	USB_VBUS
	PA11	USB_DM
	PA12	USB_DP
	PA10	USB_ID

3 Getting started

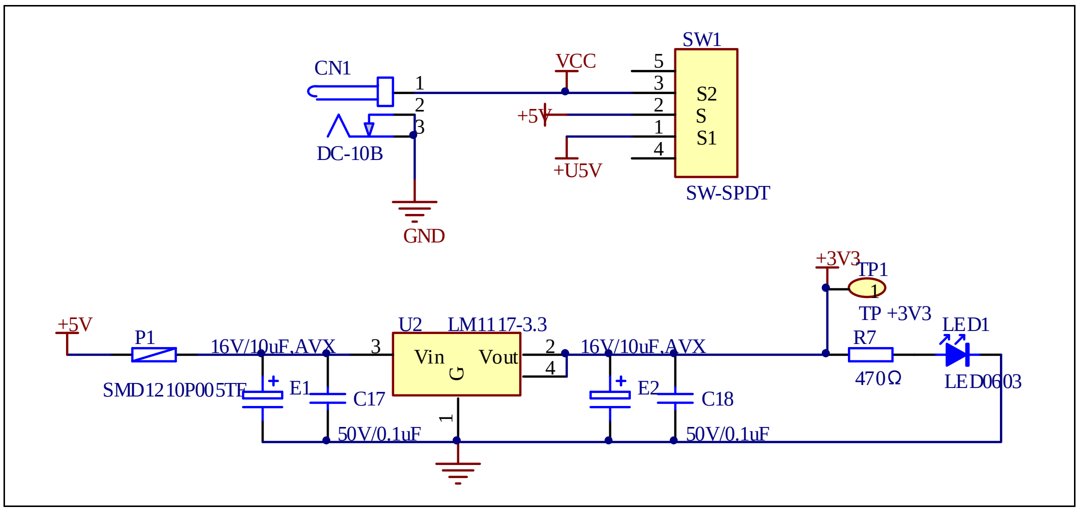
The EVAL board uses Mini USB connecter or DC-005 connector to get power DC +5V, which is the hardware system normal work voltage. A J-Link tool is necessary in order to download and debug programs. Select the correct boot mode and then power on, the LED1 will turn on, which indicates that the power supply is OK.

There are Keil version and IAR version 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:

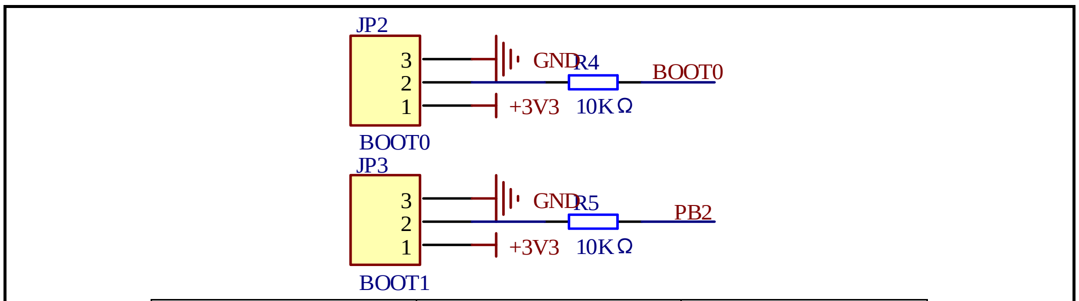
1. If you use Keil uVision4 to open the project, install the GD32F10x_Addon.2.0.0.exe to load the associated files.
2. If you use Keil uVision5 to open the project, there are two ways to solve the "Device Missing (s)" problem. One is to install GigaDevice.GD32F10x_DFP.2.0.1.pack. In Project menu, select 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_GD32F10x_Addon.2.0.0.exe to load the associated files.

4 Hardware layout overview

4.1 Power

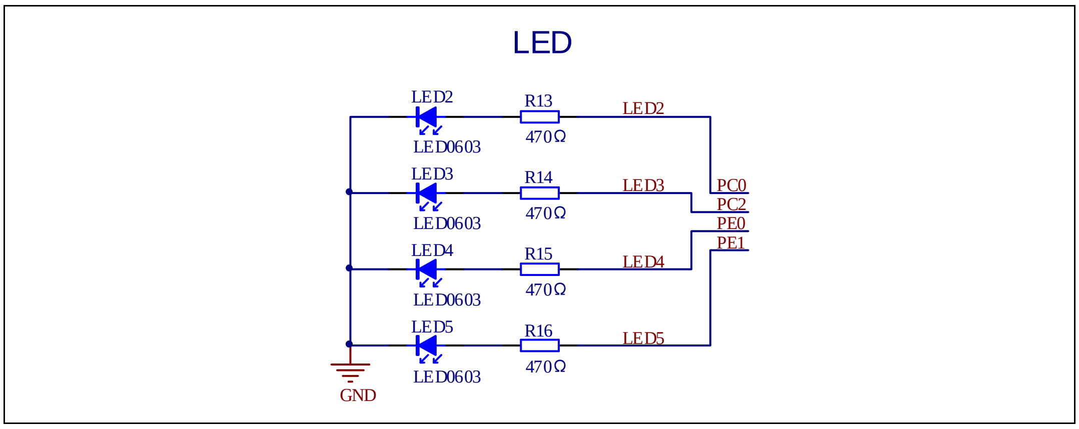


4.2 Boot

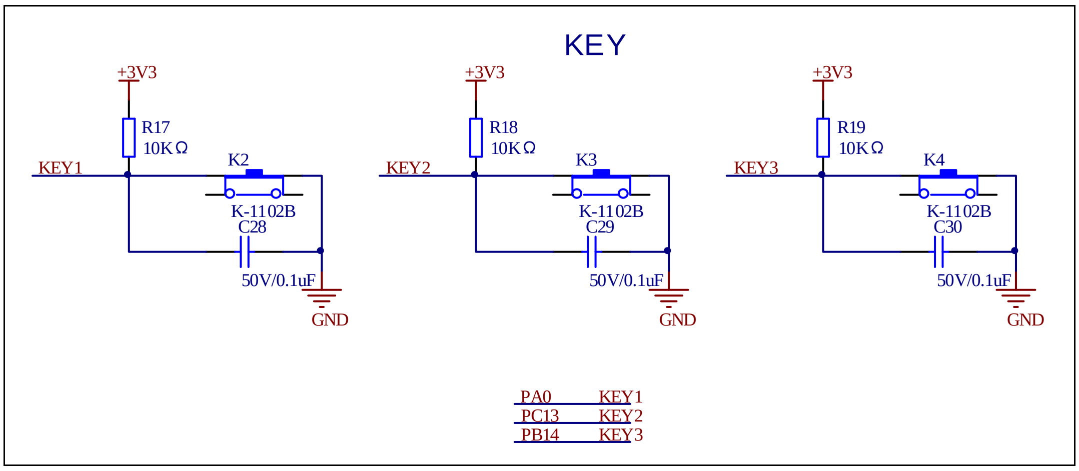


BOOT1	BOOT0	Boot Mode
Any	2-3	User memory
2-3	1-2	System memory
1-2	1-2	SRAM memory

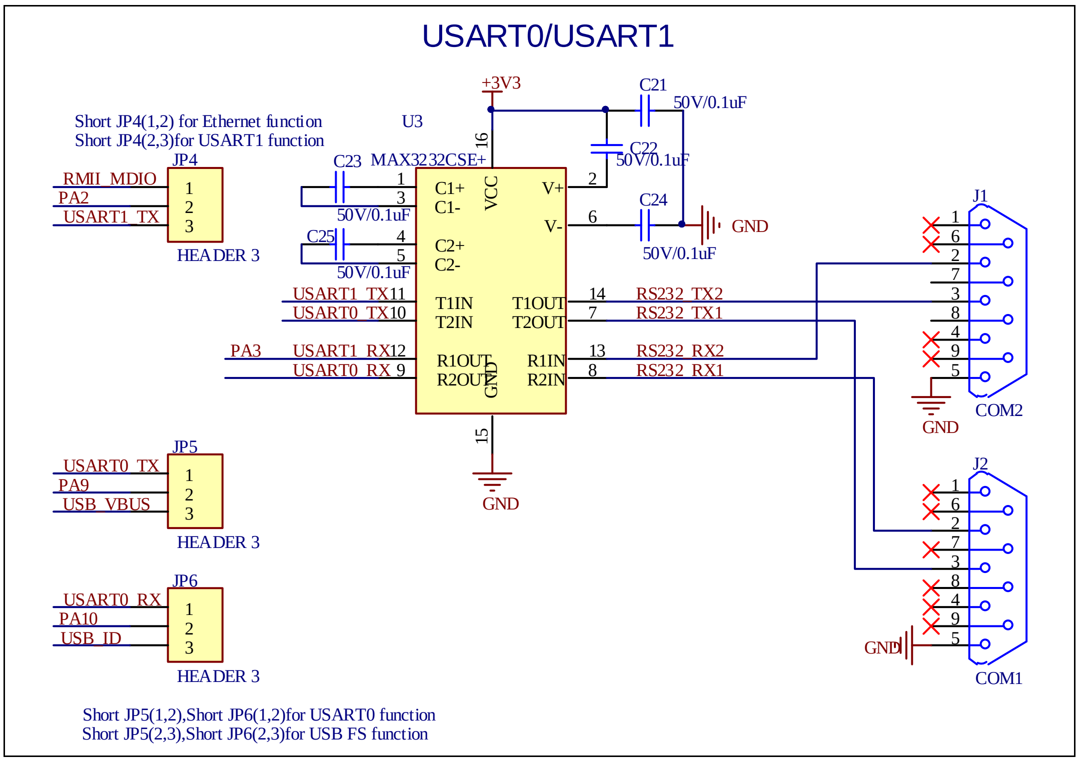
4.3 LED


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

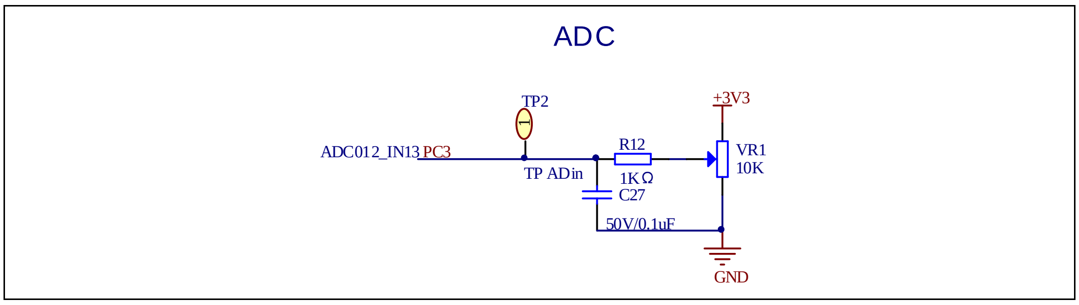
4.4 KEY


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

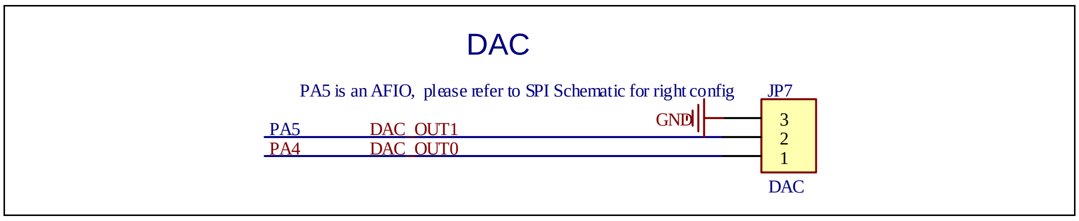
4.5 USART



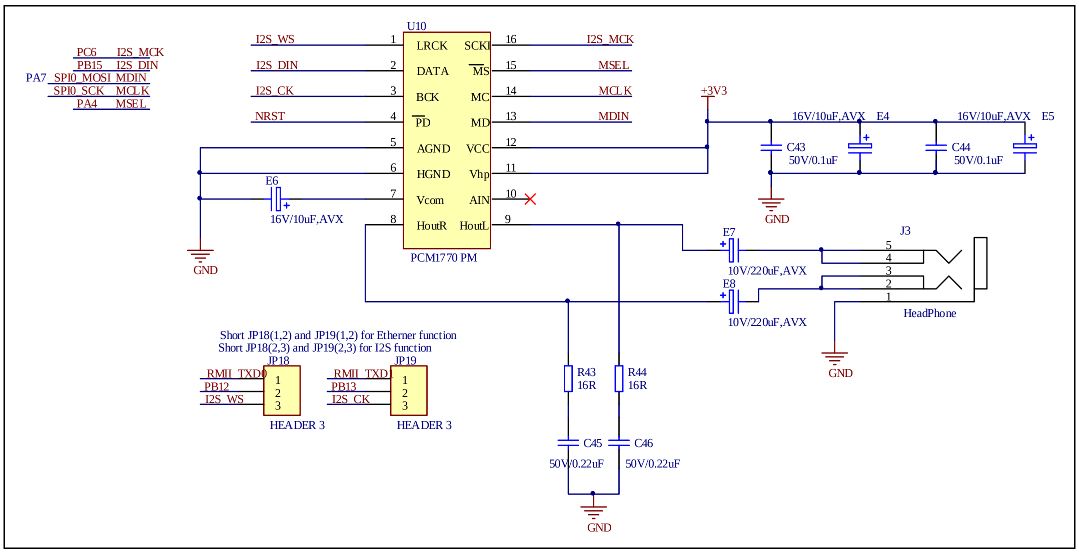
4.6 ADC



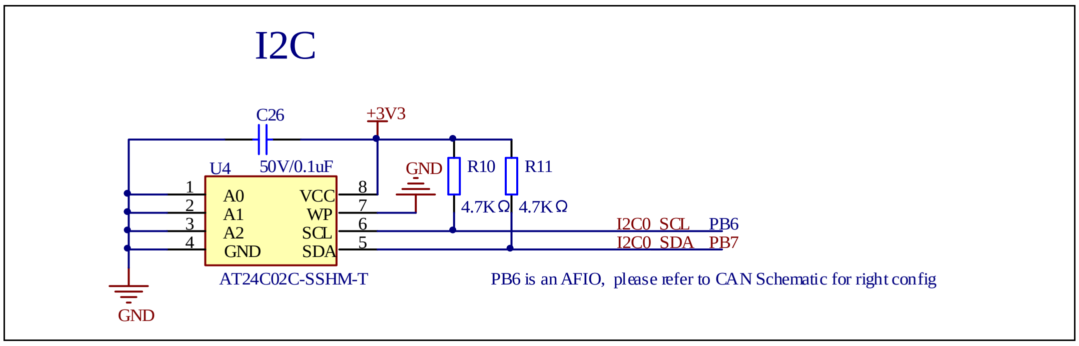
4.7 DAC


DAC
PA5 is an AFIO, please refer to SPI Schematic for right config
JP7
GND
PA5 DAC_OUT1
PA4 DAC_OUT0
DAC

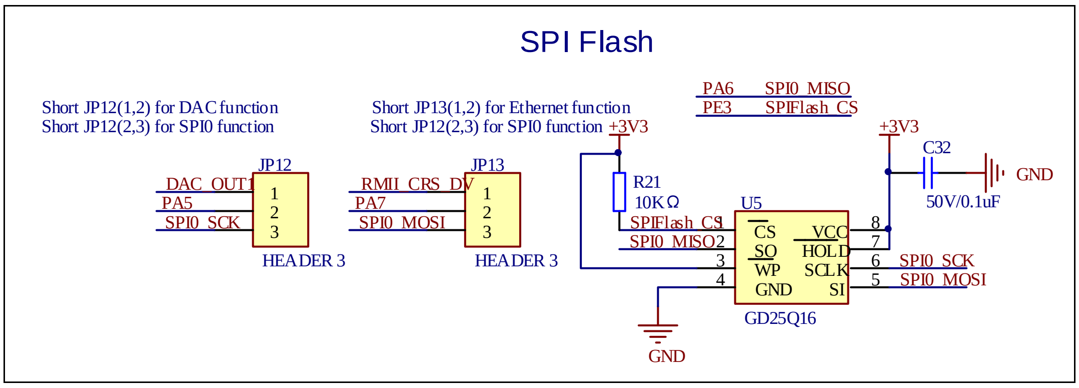
4.8 I2S



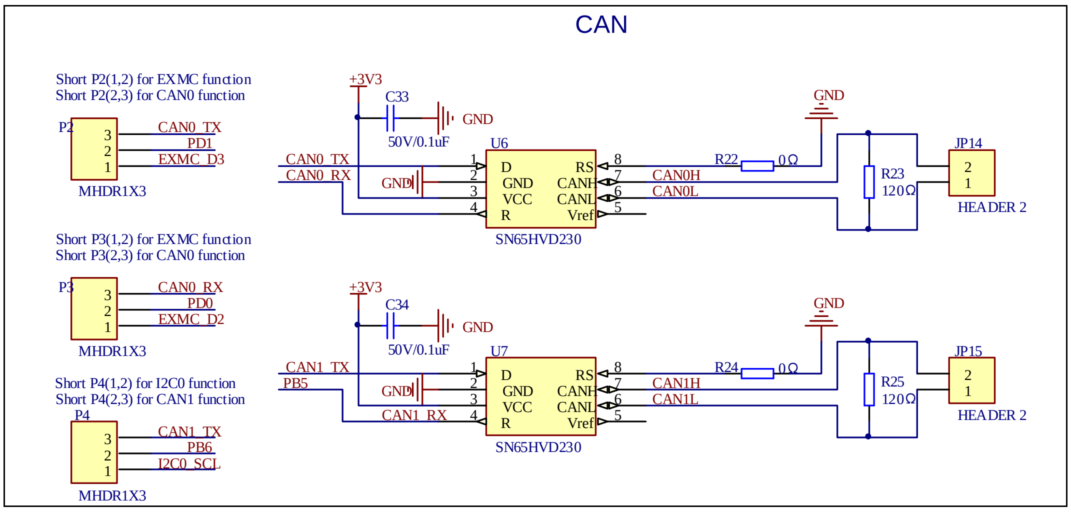
4.9 I2C



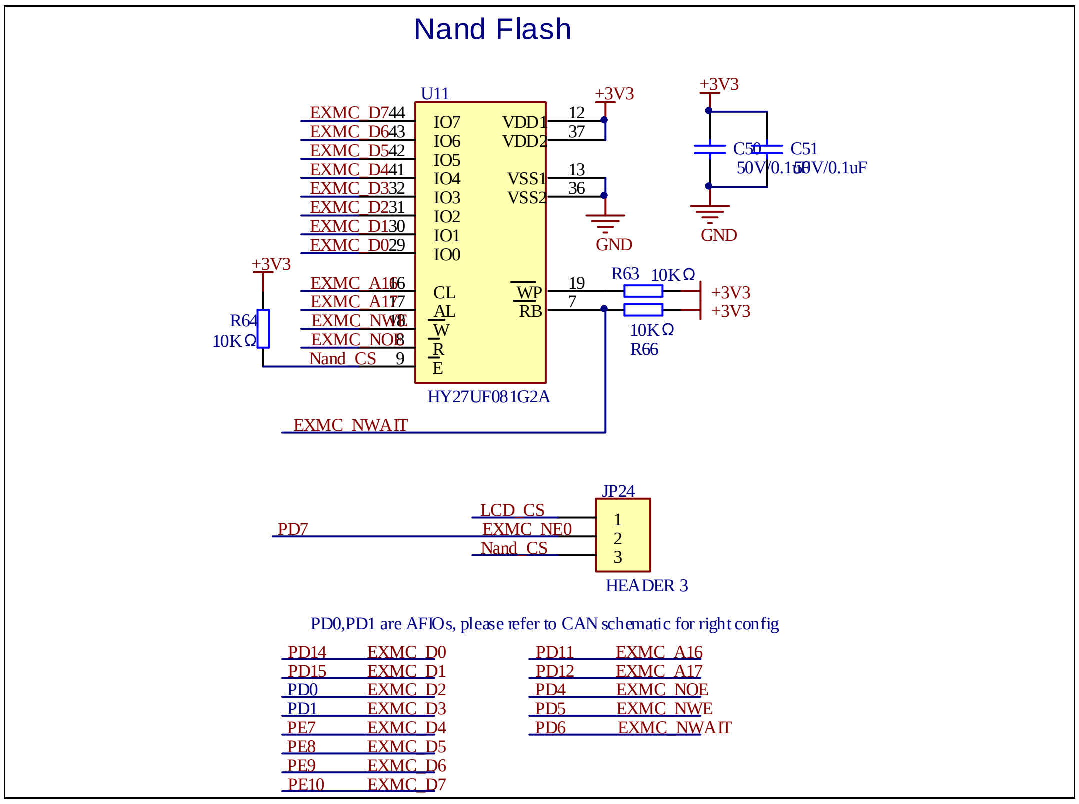
4.10 SPI



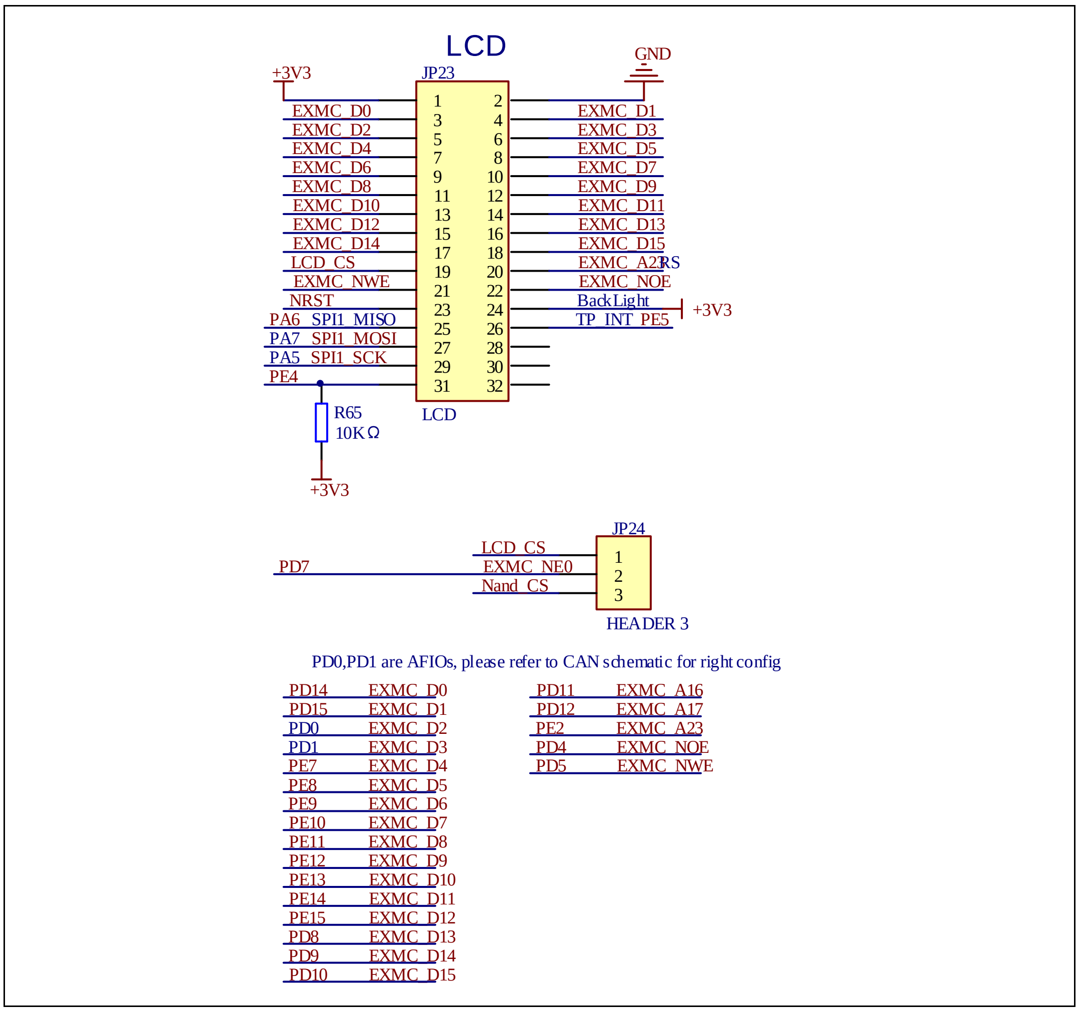
4.11 CAN



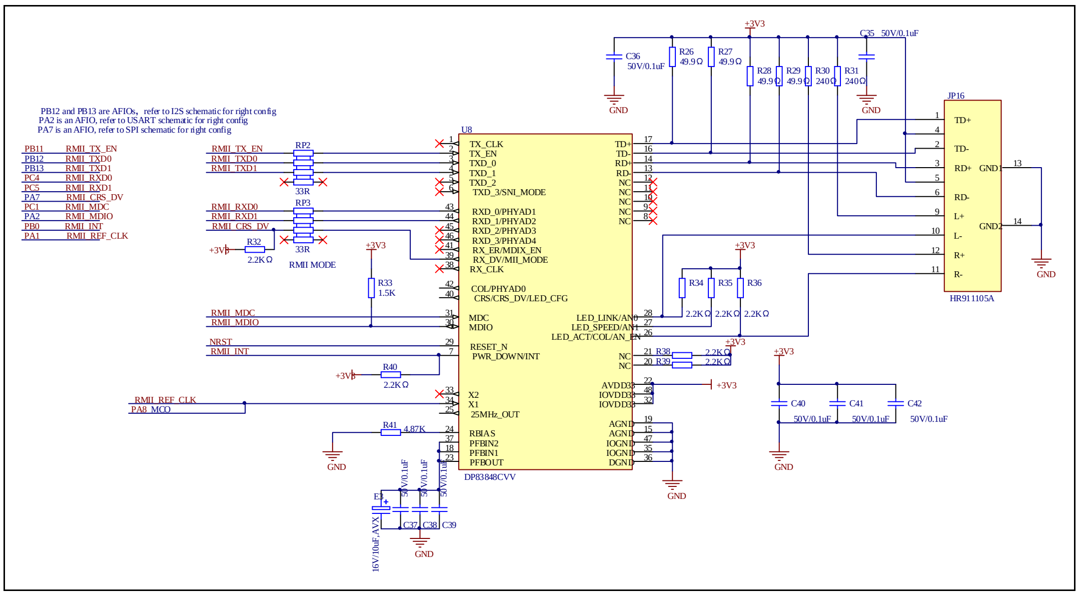
4.12 NAND



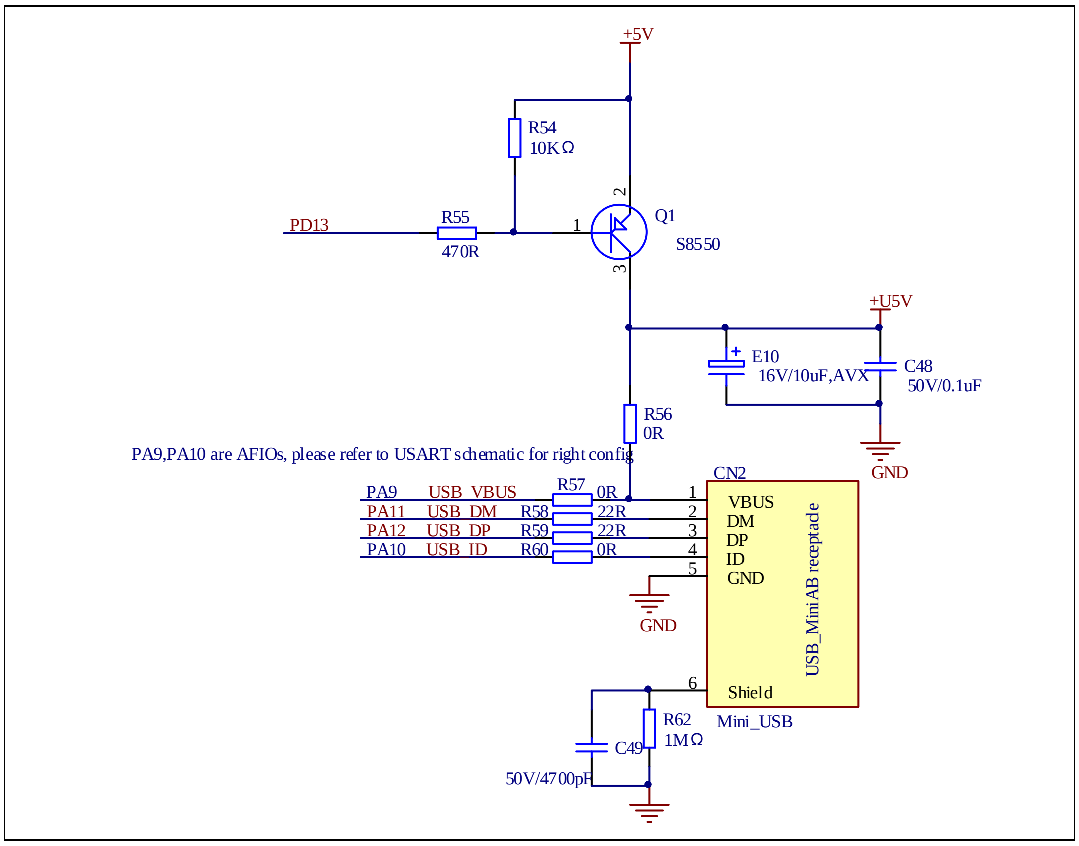
4.13 LCD


LCD
JP23
+3V3
GND
EXMC_D0
EXMC_D1
EXMC_D2
EXMC_D3
EXMC_D4
EXMC_D5
EXMC_D6
EXMC_D7
EXMC_D8
EXMC_D9
EXMC_D10
EXMC_D11
EXMC_D12
EXMC_D13
EXMC_D14
EXMC_D15
LCD_CS
EXMC_A23RS
EXMC_NWE
EXMC_NOE
NRST
BackLight
+3V3
PA6_SPI1_MISO
TP_INT_PE5
PA7_SPI1_MOSI
PA5_SPI1_SCK
PE4
R65
10KΩ
LCD
+3V3
JP24
LCD_CS
PD7
EXMC_NE0
Nand_CS
HEADER 3
PD0,PD1 are AFIOs, please refer to CAN schematic for right config
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
PD11 EXMC_A16
PD12 EXMC_A17
PE2 EXMC_A23
PD4 EXMC_NOE
PD5 EXMC_NWE

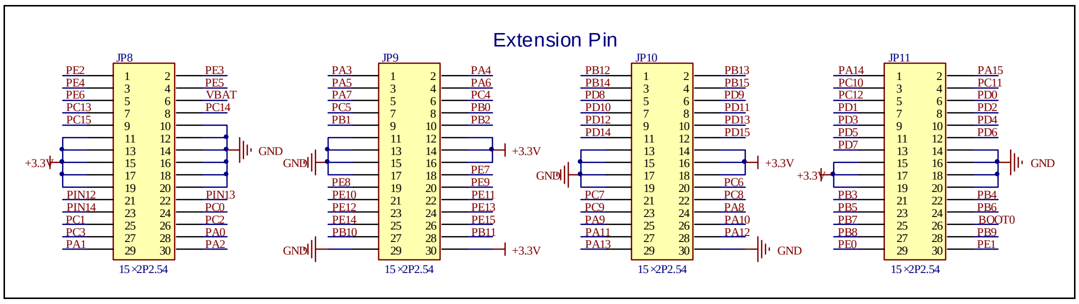
4.14 ENET



4.15 USBFS



4.16 Extension



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

GD32107C-EVAL 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, LED5 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

GD32107C-EVAL board has four keys and four LEDs. The five keys are Reset key, Tamper key, Wakeup key, 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 LED3. 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 LED3.

5.2.2 DEMO Running Result

Download the program <02_GPIO_Key_Polling_mode> to the EVAL board, press down the Tamper Key, LED3 will be turned on. Press down the Tamper Key again, LED3 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

GD32107C-EVAL board has four keys and four LEDs. The five keys are Reset key, Tamper key, Wakeup key, 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 LED3.When press down the Tamper Key, it will produce an interrupt. In the interrupt service function, the demo will toggle LED3.

5.3.2 DEMO Running Result

Download the program <03_EXTI_Key_Interrupt_mode> to the EVAL board, Press down the Tamper Key, LED3 will be turned on. Press down the Tamper Key again, LED3 will be turned off.

5.4 USART_Printf

5.4.1 DEMO Purpose

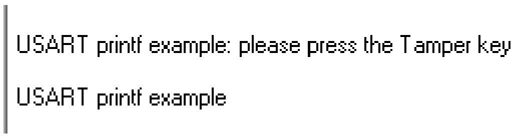
This demo includes the following functions of GD32 MCU:

- 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, jump the JP5 to USART with the jumper cap and connect serial cable to EVAL_COM0. 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_HyperTerminal_Interrupt

5.5.1 DEMO Purpose

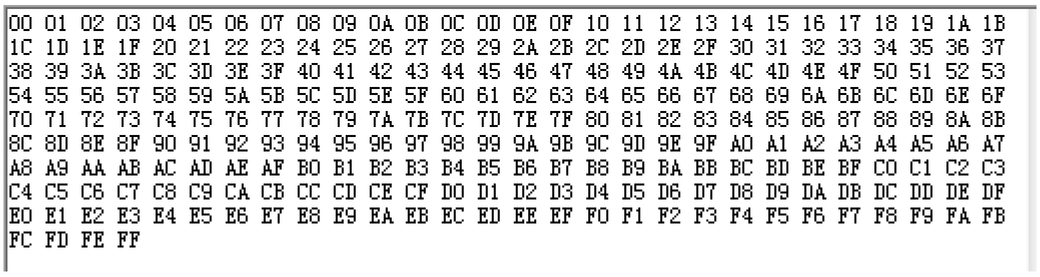
This demo includes the following functions of GD32 MCU:

- Learn to use the USART transmit and receive interrupts to communicate with the serial terminal tool

5.5.2 DEMO Running Result

Download the program < 05_USART_HyperTerminal_Interrupt > to the EVAL board, jump the JP5 to USART with the jumper cap and connect serial cable to EVAL_COM0. 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 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.

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

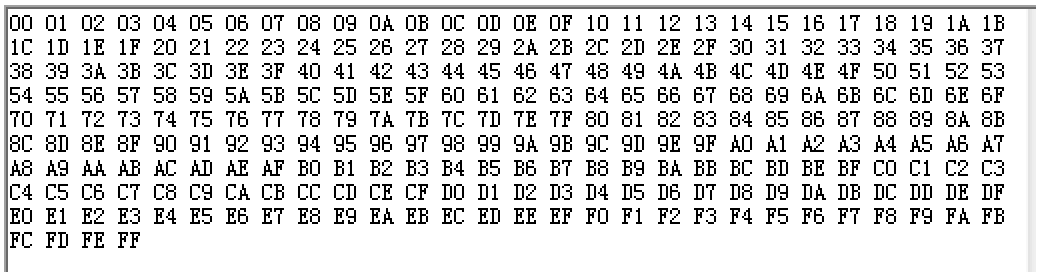
5.6.1 DEMO Purpose

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, jump the JP5 to USART with the jumper cap and connect serial cable to EVAL_COM0. 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:

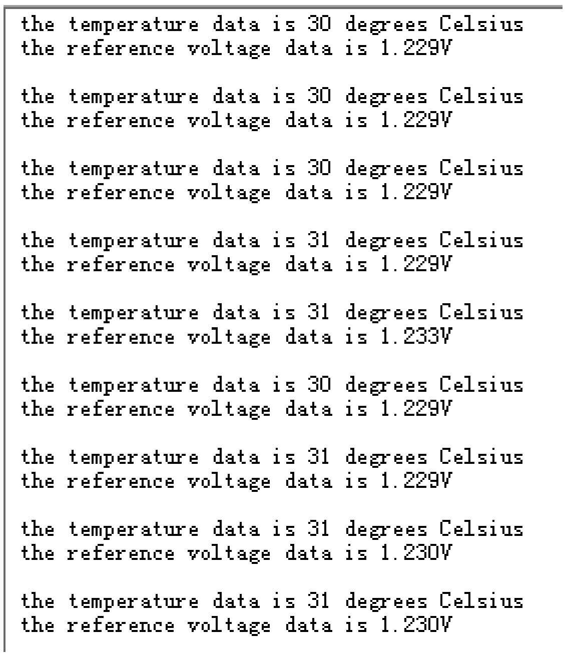
- Learn to use the ADC to convert analog signal to digital data
- Learn to get the value of inner channel 16(temperature sensor channel) and channel 17 (VREFINT channel)

5.7.2 DEMO Running Result

Jump the JP5 to USART with the jumper cap, and then download the program <07_ADC_Temperature_Vrefint> to the GD32107C-EVAL 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 (VREFINT).

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 30 degrees Celsius
the reference voltage data is 1.229V

the temperature data is 30 degrees Celsius
the reference voltage data is 1.229V

the temperature data is 30 degrees Celsius
the reference voltage data is 1.229V

the temperature data is 31 degrees Celsius
the reference voltage data is 1.229V

the temperature data is 31 degrees Celsius
the reference voltage data is 1.233V

the temperature data is 30 degrees Celsius
the reference voltage data is 1.229V

the temperature data is 31 degrees Celsius
the reference voltage data is 1.229V

the temperature data is 31 degrees Celsius
the reference voltage data is 1.230V

the temperature data is 31 degrees Celsius
the reference voltage data is 1.230V
```

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

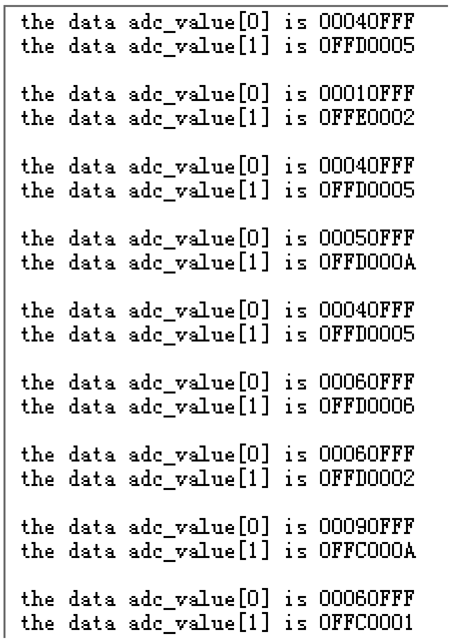
5.8.2 DEMO Running Result

Jump the JP5 to USART0 with the jumper cap, and then download the program <08_ADC0_ADC1_Follow_up_mode> to the GD32107C-EVAL 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, ADC0 starts immediately and ADC1 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 the first rising edge of TIMER0_CH0 coming, the value of the ADC0 conversion of PC3 pin is stored into the low half word of adc_value[0], and after a delay of several ADC clock cycles the value of the ADC1 conversion of PC5 pin is stored into the high half word of adc_value[0]. When the second rising edge of TIMER0_CH0 coming, the value of the ADC0 conversion of PC5 pin is stored into the low half word of adc_value[1], and after a delay of several ADC clock cycles 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 display the regular value of ADC0 and ADC1 by adc_value[0] and adc_value[1].



```
the data adc_value[0] is 00040FFF
the data adc_value[1] is 0FFD0005

the data adc_value[0] is 00010FFF
the data adc_value[1] is 0FFE0002

the data adc_value[0] is 00040FFF
the data adc_value[1] is 0FFD0005

the data adc_value[0] is 00050FFF
the data adc_value[1] is 0FFD000A

the data adc_value[0] is 00040FFF
the data adc_value[1] is 0FFD0005

the data adc_value[0] is 00060FFF
the data adc_value[1] is 0FFD0006

the data adc_value[0] is 00060FFF
the data adc_value[1] is 0FFD0002

the data adc_value[0] is 00090FFF
the data adc_value[1] is 0FFC000A

the data adc_value[0] is 00060FFF
the data adc_value[1] is 0FFC0001
```

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

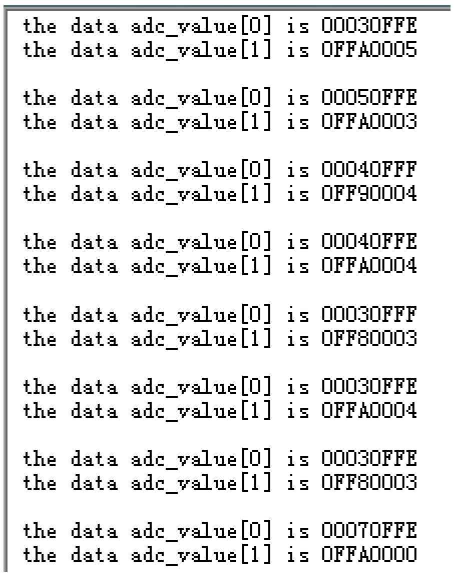
5.9.2 DEMO Running Result

Jump the JP5 to USART0 with the jumper cap, and then download the program <09_ADC0_ADC1_Regular_Parallel_mode> to the GD32107C-EVAL 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 the first rising edge of TIMER0_CH0 coming, 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 the second rising edge of TIMER0_CH0 coming, 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 00030FFE
the data adc_value[1] is 0FFA0005

the data adc_value[0] is 00050FFE
the data adc_value[1] is 0FFA0003

the data adc_value[0] is 00040FFF
the data adc_value[1] is 0FF90004

the data adc_value[0] is 00040FFE
the data adc_value[1] is 0FFA0004

the data adc_value[0] is 00030FFF
the data adc_value[1] is 0FF80003

the data adc_value[0] is 00030FFE
the data adc_value[1] is 0FFA0004

the data adc_value[0] is 00030FFE
the data adc_value[1] is 0FF80003

the data adc_value[0] is 00070FFE
the data adc_value[1] is 0FFA0000
```

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 output

5.10.2 DEMO Running Result

Download the program <10_DAC_Output_Voltage_Value> to the EVAL board and run, all the LEDs will turn on and turn off for test. The digital value is 0x7FF0, its converted analog voltage should be 1.65V (VREF/2), using the voltmeter to measure PA4 or DA0 on JP7, its value is 1.65V.

5.11 I2C_EEPROM

5.11.1 DEMO Purpose

This demo includes the following functions of GD32 MCU:

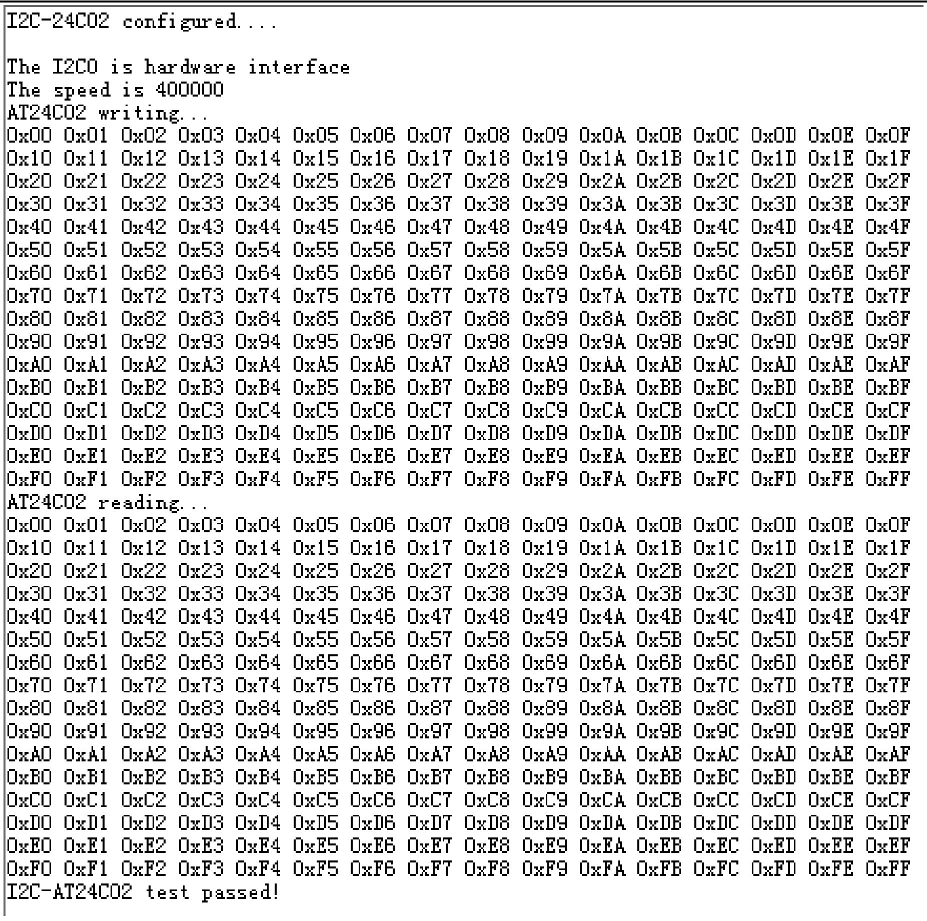
- Learn to use the master transmitting mode of I2C module
- Learn to use the master receiving mode of I2C module
- Learn to read and write the EEPROM with I2C interface

5.11.2 DEMO Running Result

Jump the JP5 to USART with the jumper cap and the P4 to I2C, then download the program <11_I2C_EEPROM> to the EVAL board and run. Connect serial cable to COM0, and open the HyperTerminal to show the print message.

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

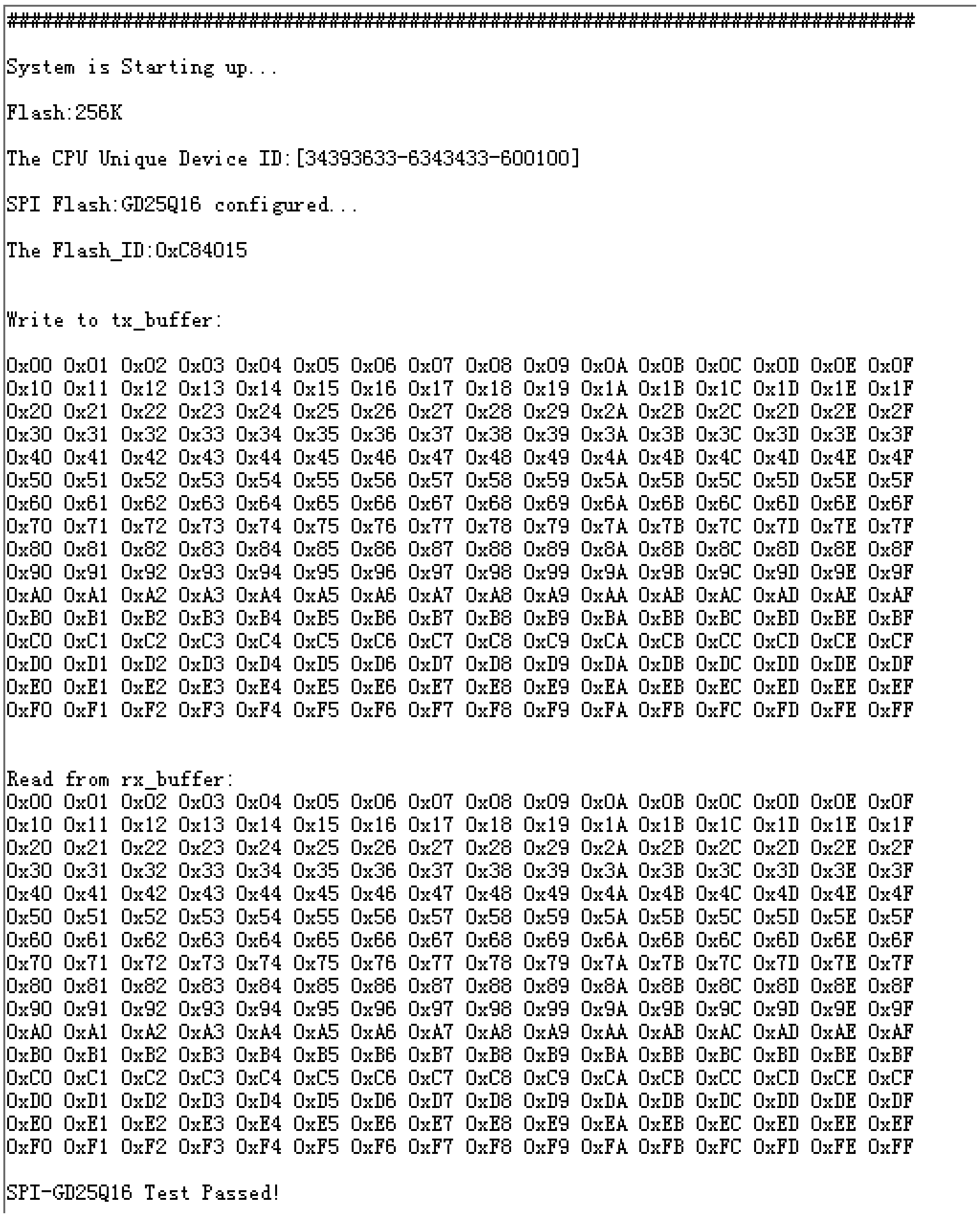
- Learn to use the master mode of SPI unit to read and write NOR Flash with the SPI interface

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 JP5 to USART, and jump the JP12 and JP13 to SPI.

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.



```
################################################################################

System is Starting up...

Flash:256K

The CPU Unique Device ID:[34393633-6343433-600100]

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

GD32107C-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

Jump JP18 and JP19 to the I2S.

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_NandFlash

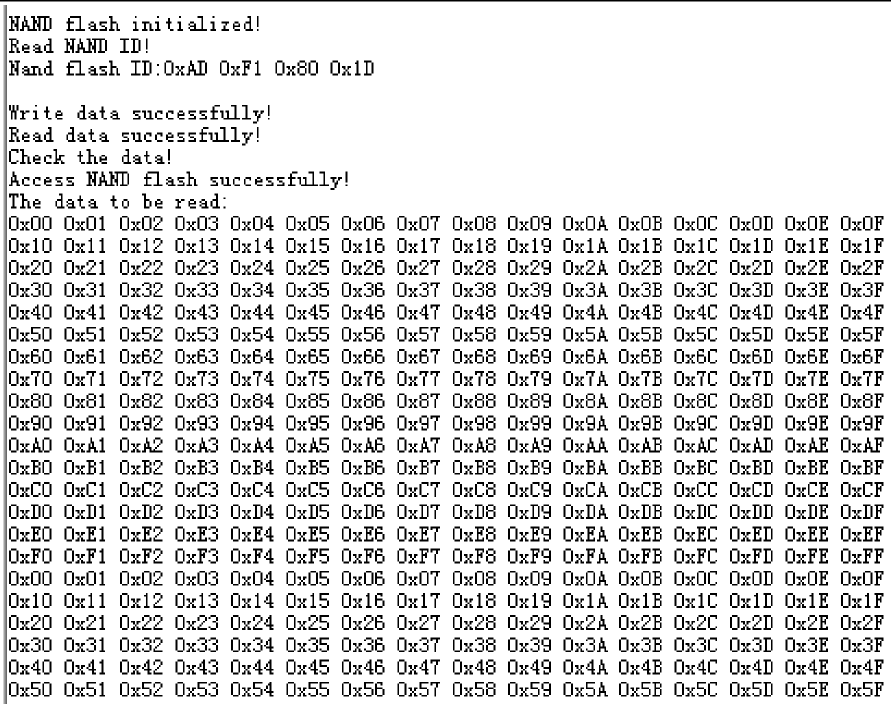
5.14.1 DEMO Purpose

This demo includes the following functions of GD32 MCU:

- Learn to use EXMC control the NAND flash

5.14.2 DEMO Running Result

GD32107C-EVAL board has EXMC module to control NAND flash. Before running the demo, JP5 must be fitted to USART, P2 and P3 must be fitted to the EXMC port, JP24 must be fitted to the Nand port. Download the program <14_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, LED2 will be turned on. Otherwise, turn on the LED4. Information via a HyperTerminal output as following:



```
NAND flash initialized!
Read NAND ID!
Nand flash ID:0xAD 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
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
```

5.15 EXMC_TouchScreen

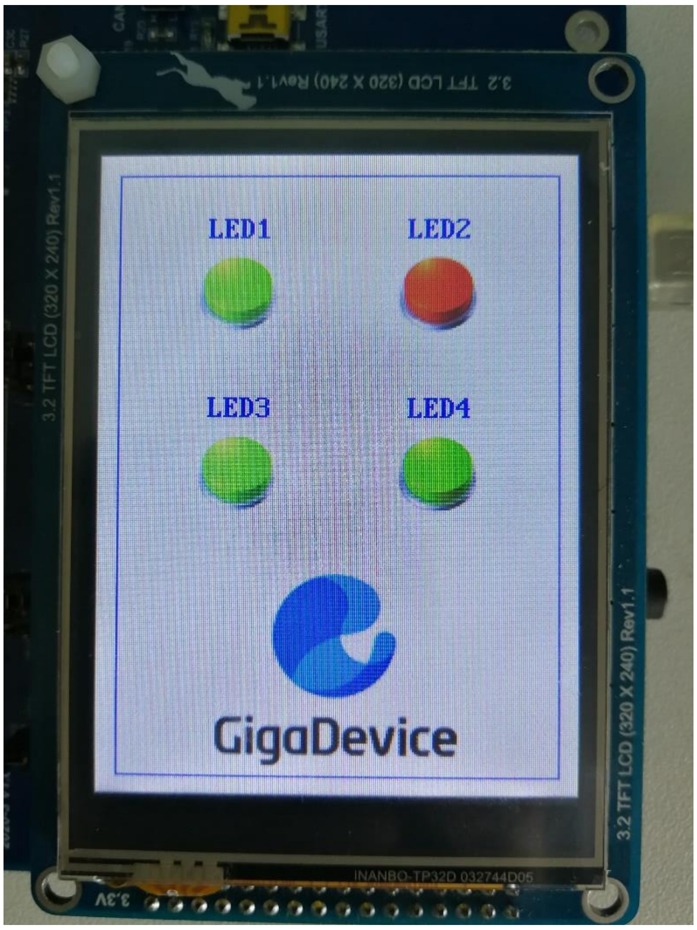
5.15.1 DEMO Purpose

This demo includes the following functions of GD32 MCU:

- Learn to use EXMC control LCD

5.15.2 DEMO Running Result

GD32107C-EVAL board has EXMC module to control LCD. Before running the demo, JP12 and JP13 must be fitted to the SPI port, P2 and P3 must be fitted to the EXMC port. Download the program <15_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.



5.16 CAN_Network

5.16.1 DEMO Purpose

This demo includes the following functions of GD32 MCU:

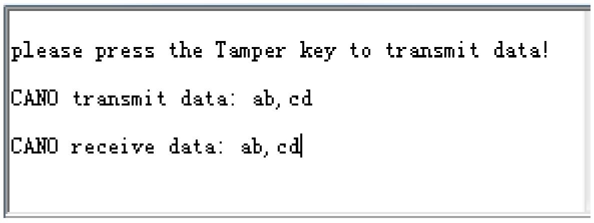
- Learn to use the CAN0 communication between two boards

GD32107C-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 CAN0.

5.16.2 DEMO Running Result

This example is tested with two GD32F107C-EVAL boards. Jump the JP5 to USART and P2, P3 to CAN with the jumper cap. Connect L pin to L pin and H pin to H pin of JP14 on the boards for sending and receiving frames. Download the program <16_CAN_Network> to the two EVAL boards, and connect serial cable to COM0. Firstly, the COM0 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: ab,cd
CAN0 receive data: ab,cd
```

5.17 RCU_Clock_Out

5.17.1 DEMO Purpose

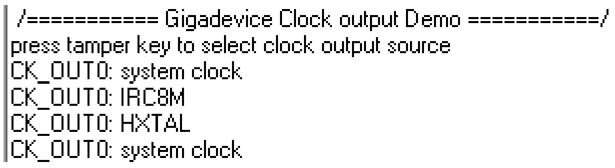
This demo includes the following functions of GD32 MCU:

- 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

Jump the JP5 to USART with the jumper cap, and download the program <17_RCU_Clock_Out> to the EVAL board and run. Connect serial cable to EVAL_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
```

5.18 PMU_sleep_wakeup

5.18.1 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

5.18.2 DEMO Running Result

Download the program < 18_PMU_sleep_wakeup > to the EVAL board, jump the JP5 to USART with the jumper cap and connect serial cable to EVAL_COM0. After power-on, all the LEDs are off. The mcu will enter sleep mode and the software stop 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.

5.19 RTC_Calendar

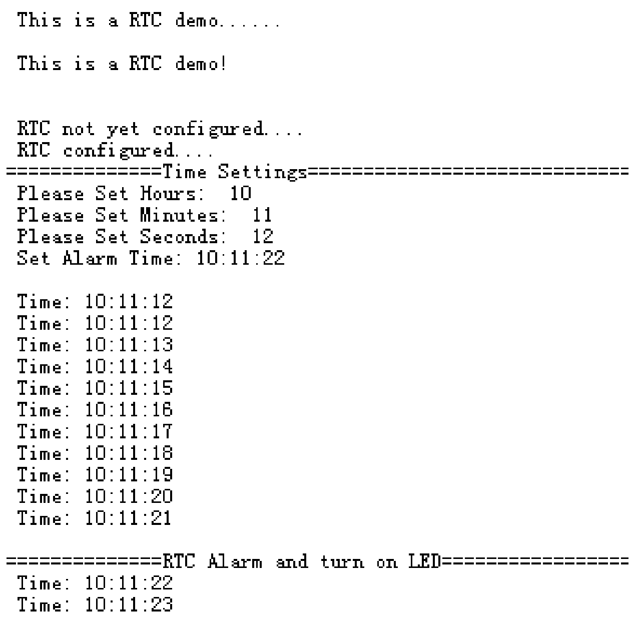
5.19.1 DEMO Purpose

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

5.19.2 DEMO Running Result

Jump the JP5 to USART with the jumper cap, and download the program <19_RTC_Calendar> to the EVAL board and run. Connect serial cable to EVAL_COM0, 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. At the same time, set current time add 10 second as alarm time. After 10 second, the alarm note will displayed on the HyperTerminal and turn on LEDs.



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

 This is a RTC demo!


 RTC not yet configured....
 RTC configured....
==============Time Settings=============================
 Please Set Hours:  10
 Please Set Minutes:  11
 Please Set Seconds:  12
 Set Alarm Time: 10:11:22

 Time: 10:11:12
 Time: 10:11:12
 Time: 10:11:13
 Time: 10:11:14
 Time: 10:11:15
 Time: 10:11:16
 Time: 10:11:17
 Time: 10:11:18
 Time: 10:11:19
 Time: 10:11:20
 Time: 10:11:21

==============RTC Alarm and turn on LED=================
 Time: 10:11:22
 Time: 10:11:23
```

5.20 TIMER_Breath_LED

5.20.1 DEMO Purpose

This demo includes the following functions of GD32 MCU:

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

5.20.2 DEMO Running Result

Use the DuPont line to connect the TIMER0_CH0 (PA8) and LED2 (PC0), and then download the program <20_TIMER_Breath_LED> to the GD32107C-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.

5.21 ENET

5.21.1 FreeRTOS_tcpudp

DEMO Purpose

This demo includes the following functions of GD32 MCU:

- Learn to use Lwip stack
- Learn to use FreeRTOS operation system
- Learn to use netconn and socket API to handle with a task
- Learn how to realize a tcp server
- Learn how to realize a tcp client
- Learn how to realize a udp server/client
- Learn how to use DHCP to allocate ip address automatically

This demo is based on the GD32107C-EVAL evaluation board, it shows how to configure the enet peripherals to send and receive frames in normal mode and use lwip tcp/ip stack to realize ping, telnet and server/client functions.

JP4, JP13, JP18, JP19 must be fitted.

It is configured in RMII mode, and 25MHz oscillator is used, the system clock is configured to 108MHz.

This demo realizes three applications:

1) Telnet application, the eval board acts as tcp server. Users can link the client with the eval board server, using 8000 port. Users can see the reply from the server, and can send the name(should input enter key) to server.

2) tcp client application, the eval board acts as tcp client. Users can link the eval board client with the server, using 10260 port. Users can send information from server to client, then the client will send back the information.

3) udp application. Users can link the eval board with other station, using 1025 port. Users can send information to board from station, then the board will send back the information.

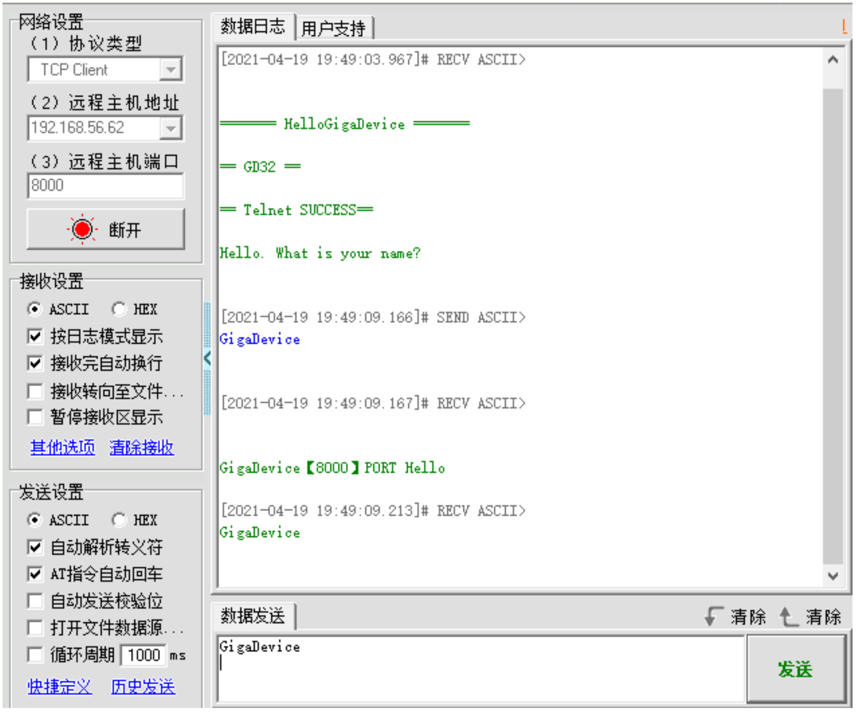
If users need dhcp function, it can be configured from the private defines in main.h. This function is closed by default.

Note: Users should configure ip address, mask and gw of GD32107C-EVAL evaluation board or served according to the actual net situation from the private defines in main.h.

DEMO Running Result

Download the program <FreeRTOS_tcpudp> to the EVAL board, LED3 will light every 500ms.

Using Network assistant software, configure the pc side to tcp client, using 8000 port, and when send something through the assistant, users can see the reply from the server:



网络设置
（1）协议类型 TCP Client
（2）远程主机地址 192.168.56.62
（3）远程主机端口 8000
断开

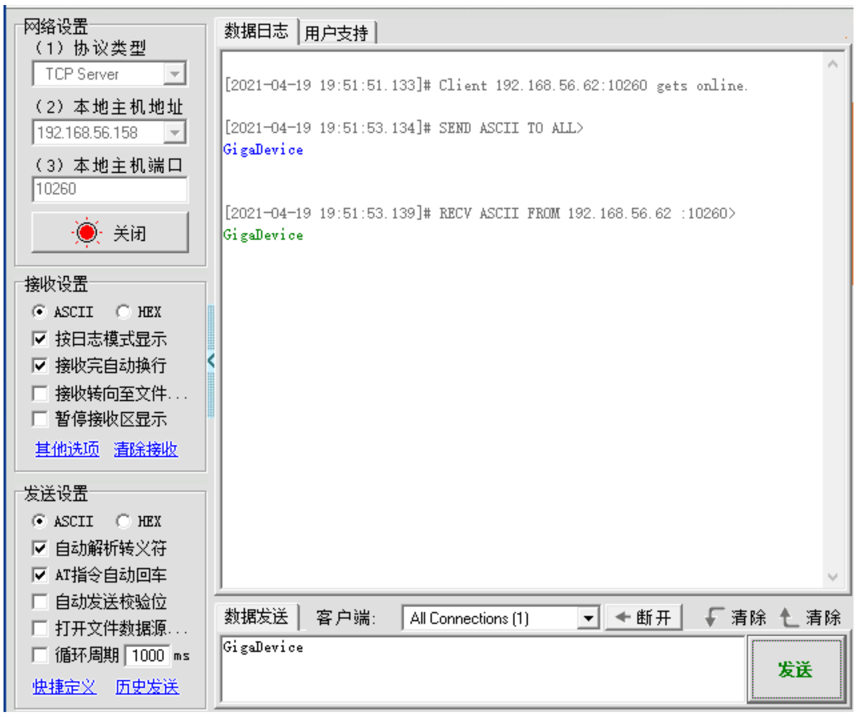
接收设置
[✓ ASCII] HEX
[✓ 按日志模式显示]
[✓ 接收完自动换行]
接收转向至文件...
暂停接收区显示

发送设置
[✓ ASCII] HEX
[✓ 自动解析转义符]
[✓ AT指令自动回车]
自动发送校验位
打开文件数据源...
循环周期 1000 ms

```
[2021-04-19 19:49:03.967]# RECV ASCII>


====== HelloGigaDevice ======

== GD32 ==

== Telnet SUCCESS==

Hello. What is your name?


[2021-04-19 19:49:09.166]# SEND ASCII>
GigaDevice


[2021-04-19 19:49:09.167]# RECV ASCII>


GigaDevice【8000】PORT Hello

[2021-04-19 19:49:09.213]# RECV ASCII>
GigaDevice
```

数据发送: GigaDevice

Using Network assistant software, configure the pc side to tcp server, using 10260 port (ensure that the port number is not occupied by other processes), and when send something through the assistant, users can see the echo reply from the client:



网络设置
（1）协议类型 TCP Server
（2）本地主机地址 192.168.56.158
（3）本地主机端口 10260
关闭

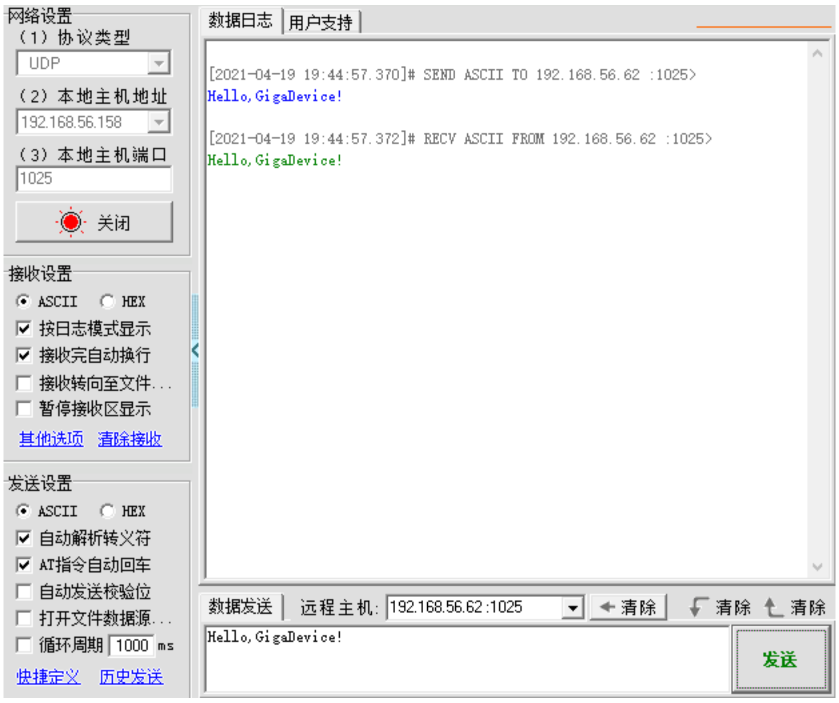
接收设置
[✓ ASCII] HEX
[✓ 按日志模式显示]
[✓ 接收完自动换行]
接收转向至文件...
暂停接收区显示

发送设置
[✓ ASCII] HEX
[✓ 自动解析转义符]
[✓ AT指令自动回车]
自动发送校验位
打开文件数据源...
循环周期 1000 ms

```
[2021-04-19 19:51:51.133]# Client 192.168.56.62:10260 gets online.

[2021-04-19 19:51:53.134]# SEND ASCII TO ALL>
GigaDevice


[2021-04-19 19:51:53.139]# RECV ASCII FROM 192.168.56.62 :10260>
GigaDevice
```

数据发送 客户端: All Connections (1)
GigaDevice

Using Network assistant software, configure to use udp protocol, using 1025 port (ensure that the port number is not occupied by other processes), and when send something through the assistant, users can see the echo reply from the board:



网络设置

（1）协议类型: UDP

（2）本地主机地址: 192.168.56.158

（3）本地主机端口: 1025

接收设置: [✓ ASCII] HEX

☑ 按日志模式显示

☑ 接收完自动换行

☐ 接收转向至文件...

☐ 暂停接收区显示

发送设置: [✓ ASCII] HEX

☑ 自动解析转义符

☑ AT指令自动回车

☐ 自动发送校验位

☐ 打开文件数据源...

☐ 循环周期 1000 ms

远程主机: 192.168.56.62 :1025

```
[2021-04-19 19:44:57.370]# SEND ASCII TO 192.168.56.62 :1025>
Hello,GigaDevice!

[2021-04-19 19:44:57.372]# RECV ASCII FROM 192.168.56.62 :1025>
Hello,GigaDevice!
```

数据发送: Hello,GigaDevice!

Open the DHCP function in main.h, using a router to connect the board with the pc, users can see the automatic allocated ip address of the board from the HyperTerminal.

5.21.2 Raw_tcpudp

DEMO Purpose

This demo includes the following functions of GD32 MCU:

- Learn to use Lwip stack
- Learn to use raw API to handle with a task
- Learn how to realize a tcp server
- Learn how to realize a tcp client
- Learn how to realize a udp server/client
- Learn how to use DHCP to allocate ip address automatically
- Learn to handle with received packet in polling mode and in interrupt mode

This demo is based on the GD32107C-EVAL evaluation board, it shows how to configure the enet peripherals to send and receive frames in normal mode and use lwip tcp/ip stack to realize ping, telnet and server/client functions.

JP4, JP13, JP18, JP19 must be fitted. JP5 jump to USART0.

It is configured in RMII mode, and 25MHz oscillator is used, the system clock is configured to

108MHz.

This demo realizes three applications:

1) Telnet application, the eval board acts as tcp server. Users can link the client with the eval board server, using 8000 port. Users can see the reply from the server, and can send the name(should input enter key) to server.

2) tcp client application, the eval board acts as tcp client. Users can link the eval board client with the server, using 10260 port. Users can send information to client from server, then the client will send back the information. If the server is not online at first, or is break during process, when the server is ready again, users can press tamper key to reconnect with server, and communicate.

3) udp application, Users can link the eval board with other station, using 1025 port. Users can send information from station to board, then the board will send back the information.

By default, the packet reception is polled in while(1). If users want to receive packet in interrupt service, uncomment the macro defined USE_ENET_INTERRUPT in main.h.

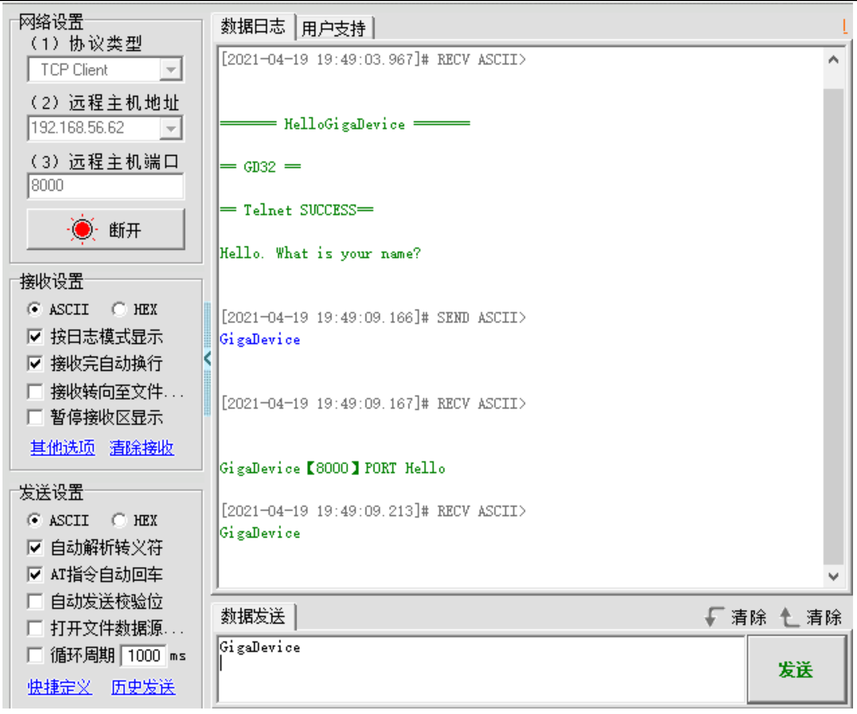
If users need dhcp function, it can be configured from the private defines in main.h. This function is closed in default.

Note: Users should configure ip address, mask and gw of GD32107C-EVAL evaluation board, or server according to the actual net situation from the private defines in main.h.

DEMO Running Result

Download the program <Raw_tcpudp> to the EVAL board.

Using Network assistant software, configure the pc side to tcp client, using 8000 port, and when send something through the assistant, users can see the reply from the server:



网络设置
(1) 协议类型: TCP Client
(2) 远程主机地址: 192.168.56.62
(3) 远程主机端口: 8000
断开

接收设置
[✓ ASCII] HEX
[✓] 按日志模式显示
[✓] 接收完自动换行
[ ] 接收转向至文件...
[ ] 暂停接收区显示
其他选项 清除接收

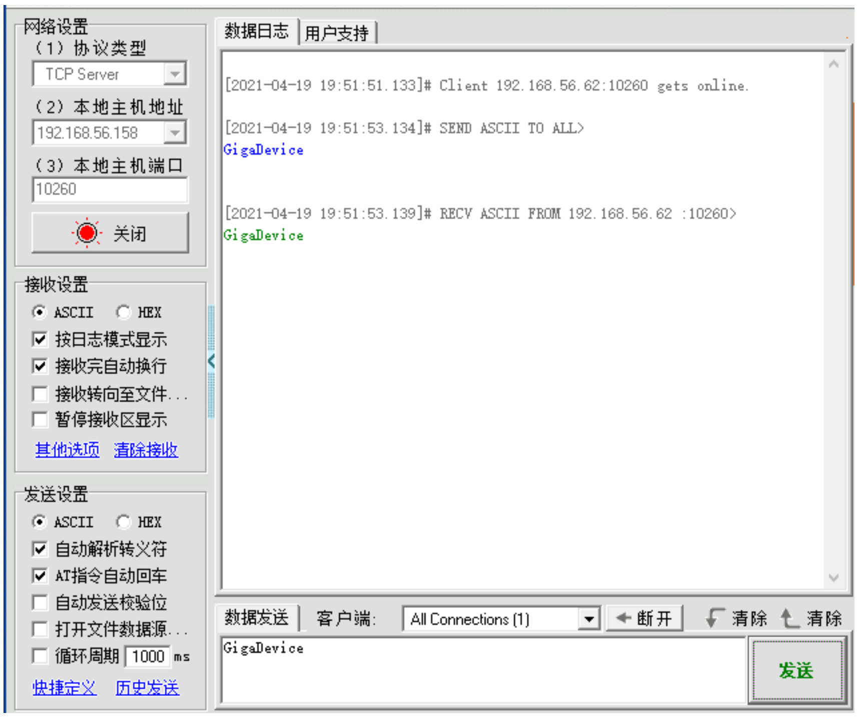
发送设置
[✓ ASCII] HEX
[✓] 自动解析转义符
[✓] AT指令自动回车
[ ] 自动发送校验位
[ ] 打开文件数据源...
[ ] 循环周期 1000 ms
快捷定义 历史发送

数据日志 | 用户支持

```
[2021-04-19 19:49:03.967]# RECV ASCII>

====== HelloGigaDevice ======

== GD32 ==

== Telnet SUCCESS==

Hello. What is your name?


[2021-04-19 19:49:09.166]# SEND ASCII>
GigaDevice


[2021-04-19 19:49:09.167]# RECV ASCII>


GigaDevice【8000】PORT Hello

[2021-04-19 19:49:09.213]# RECV ASCII>
GigaDevice
```

数据发送 清除 清除
GigaDevice
发送

Using Network assistant software, configure the pc side to tcp server, using 10260 port, press the Tamper key, and when send something through the assistant, users can see the echo reply from the client:



网络设置
(1) 协议类型: TCP Server
(2) 本地主机地址: 192.168.56.158
(3) 本地主机端口: 10260
关闭

接收设置
[✓ ASCII] HEX
[✓] 按日志模式显示
[✓] 接收完自动换行
[ ] 接收转向至文件...
[ ] 暂停接收区显示
其他选项 清除接收

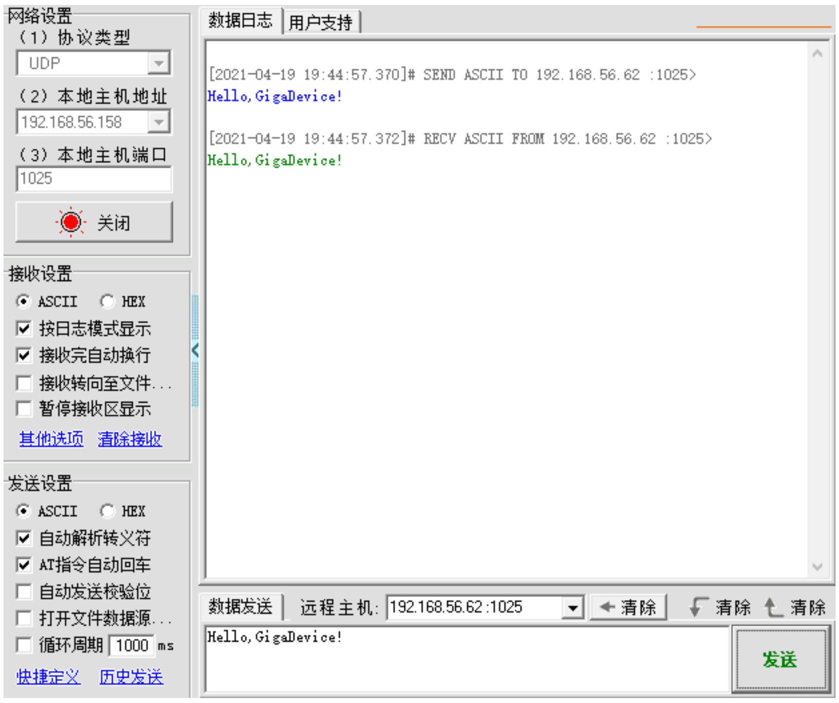
发送设置
[✓ ASCII] HEX
[✓] 自动解析转义符
[✓] AT指令自动回车
[ ] 自动发送校验位
[ ] 打开文件数据源...
[ ] 循环周期 1000 ms
快捷定义 历史发送

数据日志 | 用户支持

```
[2021-04-19 19:51:51.133]# Client 192.168.56.62:10260 gets online.

[2021-04-19 19:51:53.134]# SEND ASCII TO ALL>
GigaDevice


[2021-04-19 19:51:53.139]# RECV ASCII FROM 192.168.56.62 :10260>
GigaDevice
```

数据发送 | 客户端: All Connections (1) 断开 清除 清除
GigaDevice
发送

Using Network assistant software, configure to use udp protocol, using 1025 port, and when send something through the assistant, users can see the echo reply from the board:



网络设置
（1）协议类型 UDP
（2）本地主机地址 192.168.56.158
（3）本地主机端口 1025

接收设置
ASCII [✓] HEX
[✓] 按日志模式显示
[✓] 接收完自动换行
[ ] 接收转向至文件...
[ ] 暂停接收区显示

发送设置
ASCII [✓] HEX
[✓] 自动解析转义符
[✓] AT指令自动回车
[ ] 自动发送校验位
[ ] 打开文件数据源...
[ ] 循环周期 1000 ms

```
[2021-04-19 19:44:57.370]# SEND ASCII TO 192.168.56.62 :1025>
Hello,GigaDevice!

[2021-04-19 19:44:57.372]# RECV ASCII FROM 192.168.56.62 :1025>
Hello,GigaDevice!
```

远程主机: 192.168.56.62 :1025

Hello,GigaDevice!

Open the DHCP function in main.h, using a router to connect the board with the pc, users can see the automatic allocated ip address of the board from the HyperTerminal.

5.21.3 Raw_webserver

DEMO Purpose

This demo includes the following functions of GD32 MCU:

- Learn to use Lwip stack
- Learn to use raw API to handle with a task
- Learn how to realize a web server
- Learn how to use a web server to control LEDs
- Learn how to use a web server to monitor the board VREFINT voltage
- Learn how to use DHCP to allocate ip address automatically
- Learn to handle with received packet in polling mode and in interrupt mode

This demo is based on the GD32107C-EVAL evaluation board, it shows how to configure the enet peripherals to send and receive frames in normal mode and use lwip tcp/ip stack to realize webserver application.

JP4, JP13, JP18, JP19 must be fitted. JP5 jump to USART0.

It is configured in RMII mode, and 25MHz oscillator is used, the system clock is configured to 108MHz.

This demo realizes webserver application:

Users can visit the eval board through Internet Explorer, the eval board acts as a webserver, and the url is the local ip address of the eval board. There are two experiments realized, one is the LEDs control, the other one is the ADC monitoring VREFINT voltage in real-time.

If users need dhcp function, it can be configured from the private defines in main.h. This function is closed by default. Users can use a router to connect the eval board, and use the COM port to print the automatic allocated ip address, then connect your mobile phone to the wifi which the router send. Users can visit the eval board and control it on your mobile phone.

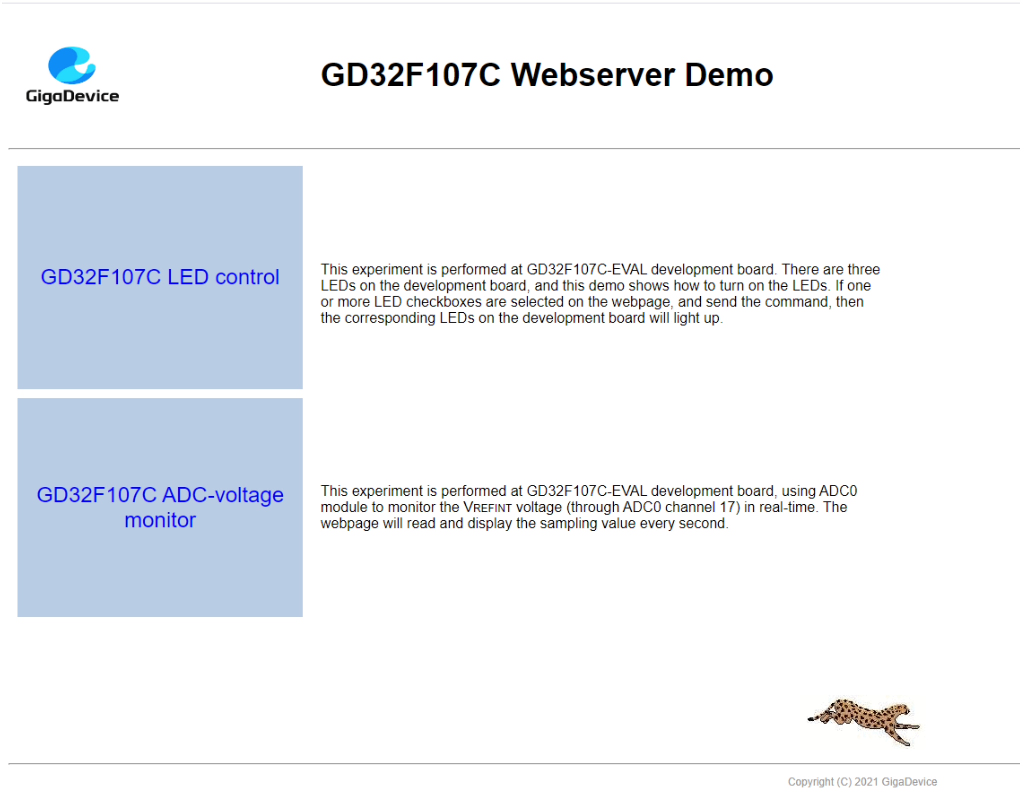
By default, the packet reception is polled in while(1). If users want to receive packet in interrupt service, uncomment the macro define USE_ENET_INTERRUPT in main.h.

Note: Users should configure ip address, mask and gw of GD32107C-EVAL evaluation board according to the actual net situation from the private defines in main.h.

DEMO Running Result

Download the program <Raw_webserver> to the EVAL board, using Internet Explorer software, enter in the ip address of the board, click on the LED control linker, choose the LED checkboxes users want to light, and “send”, the corresponding LEDs will light. Click on the ADC monitor linker, the real-time VREFINT voltage is showed on the webpage, and the data refreshes every second automatically.

The web home page shows as below:



GD32F107C Webserver Demo

GD32F107C LED control

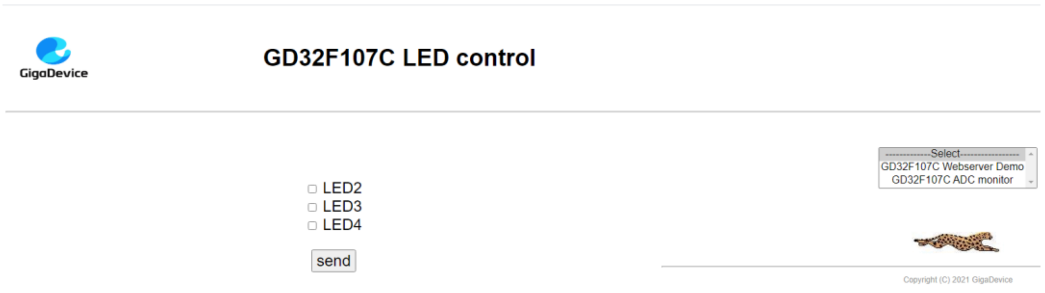
This experiment is performed at GD32F107C-EVAL development board. There are three LEDs on the development board, and this demo shows how to turn on the LEDs. If one or more LED checkboxes are selected on the webpage, and send the command, then the corresponding LEDs on the development board will light up.

GD32F107C ADC-voltage monitor

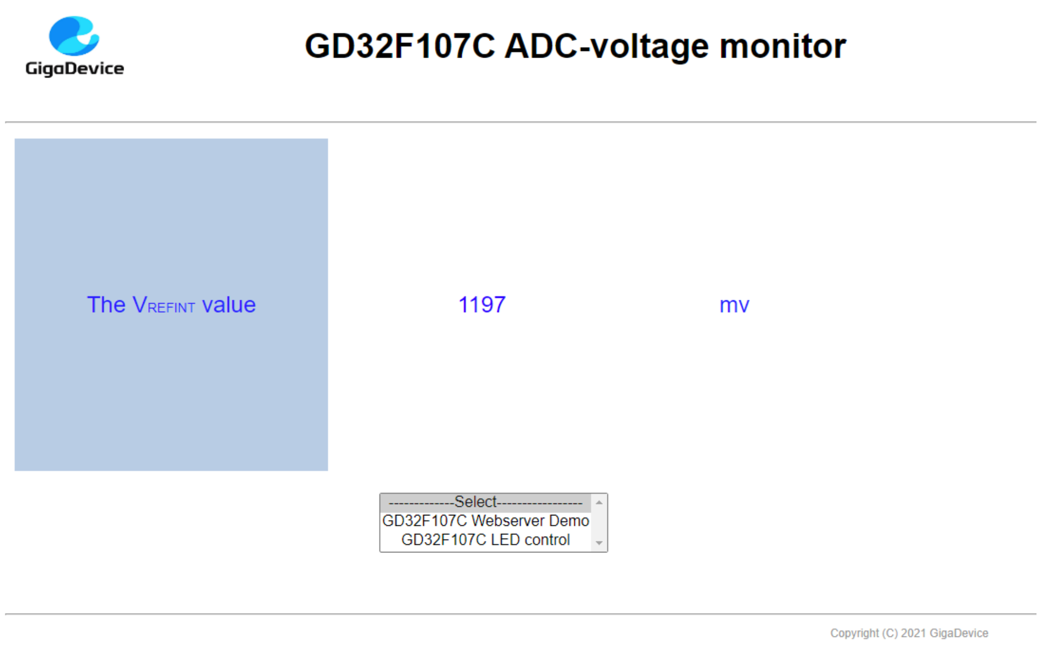
This experiment is performed at GD32F107C-EVAL development board, using ADC0 module to monitor the VREFINT voltage (through ADC0 channel 17) in real-time. The webpage will read and display the sampling value every second.

Copyright (C) 2021 GigaDevice

The LED control page shows as below:



The ADC monitor page shows as below:


GigaDevice
GD32F107C ADC-voltage monitor
The VREFINT value
1197
mv
------------Select----------------
GD32F107C Webserver Demo
GD32F107C LED control
Copyright (C) 2021 GigaDevice

Open the DHCP function in main.h, using a router to connect the board, and use the HyperTerminal to print the automatic allocated ip address, then connect your mobile phone to the wifi which the router send. Users can visit the eval board and control it on your mobile phone.

5.22 USB_Device

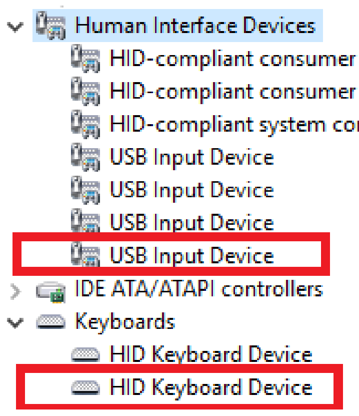
5.22.1 HID_Keyboard

DEMO Purpose

This demo includes the following functions of GD32 MCU:

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

The GD32107C-EVAL board is enumerated as an USB Keyboard, which uses the native PC Host HID driver, as shown below. The USB Keyboard use three keys 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.


Human Interface Devices
HID-compliant consumer control device
HID-compliant consumer control device
HID-compliant system controller
USB Input Device
USB Input Device
USB Input Device
USB Input Device
IDE ATA/ATAPI controllers
Keyboards
HID Keyboard Device
HID Keyboard Device

DEMO Running Result

Jump the JP5 and JP6 to USB, download the program <22_USBFS\USB_Device\HID_Keyboard> to the EVAL board and run. The USB Keyboard use three keys to output three characters (‘b’, ‘a’ and ‘c’).

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.

5.22.2 MSC_Udisk

DEMO Purpose

This demo includes the following functions of GD32 MCU:

- Learn how to use the USBFS
- Learn how to implement USB MSC(mass storage) device

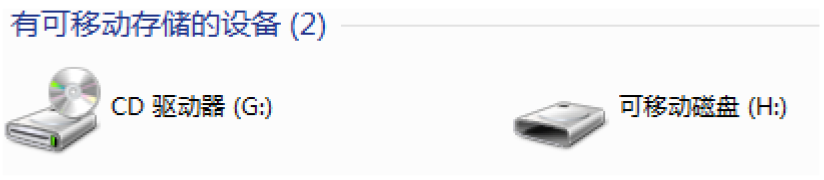
This demo mainly implements a U disk. U disk is currently very widely used removable MSC devices. MSC, the Mass Storage device Class, is a transport protocol between a computer and mobile devices, which allow a universal serial bus (USB) equipment to access a host computing device, file transfer between them, mainly including mobile hard disk, mobile U disk drive, etc. The MSC device must have a storage medium, and this demo uses the MCU's internal flash as the storage medium. For more details of the MSC protocol please refer to the MSC protocol standard.

MSC device will use a variety of transport protocols and command formats for communication, so it need to choose the appropriate protocol and command format in the realization of the application. This demo selects the BOT (bulk only transport) protocol and the required SCSI (small computer interface) command, and is compatible with a wide variety of Window operating systems. Specific BOT protocol and SCSI command specification please refer to the standard of their agreement.

DEMO Running Result

Jump the JP5 and JP6 to USB, download the program <22_USBFS\USB_Device\MSC_Udisk> to the EVAL board and run. When the EVAL-board connect to the PC, user will find a USB large capacity storage device is in the universal serial bus controller, and there is 1 more disk drives in the equipment manager of PC.

Then, after opening the resource manager, you will see more of the 1 disk, as shown in the following diagram:



At this point, the write/read/formatting operation can be performed as the other mobile devices.

5.23 USB_Host

5.23.1 Host_HID

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

GD32107C-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 and JP6 to USB, download the program <22_USBFS\USB_Host\Host_HID> to the EVAL board and run. If a mouse has been attached, the user will see the information of mouse enumeration. Firstly 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. Firstly 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.

5.23.2 Host_MSC

DEMO Purpose

This demo includes the following functions of GD32 MCU:

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

GD32107C-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 and JP6 to USB, download the program <22_USBFS\USB_Host\Host_MSC> to the EVAL board and run. If an Udisk has been attached, the user will see the information of Udisk enumeration. First pressing the 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 2 Revision history

Revision No.	Description	Date
1.0	Initial Release	Dec. 26, 2014
2.0	Firmware Update	Jun. 30, 2017
2.1	Firmware Update, Consistency Update	Jul. 31, 2018
2.2	Firmware Update. Routine name, LCD routine logo, SD card driver update	Apr. 30. 2021

Important Notice