

**GigaDevice Semiconductor Inc.**

**GD32H759I-EVAL**

**User Guide**

**Rev1.0**

# Tables of Contents

<b>TABLES OF CONTENTS .....</b>	<b>1</b>
<b>LIST OF FIGURES .....</b>	<b>4</b>
<b>LIST OF TABLES .....</b>	<b>5</b>
<b>1. SUMMARY .....</b>	<b>6</b>
<b>2. FUNCTION PIN ASSIGN .....</b>	<b>6</b>
<b>3. GETTING STARTED .....</b>	<b>10</b>
<b>4. HARDWARE LAYOUT OVERVIEW .....</b>	<b>10</b>
4.1. Power supply .....	10
4.2. Boot option .....	11
4.3. LED .....	11
4.4. KEY .....	11
4.5. ADC .....	12
4.6. DAC .....	12
4.7. CAN .....	12
4.8. DCI .....	13
4.9. ENET .....	13
4.10. HPDF .....	14
4.11. I2C .....	14
4.12. I2S .....	14
4.13. SAI .....	15
4.14. OSPI .....	15
4.15. SPI_LCD .....	16
4.16. SDIO .....	16
4.17. SDRAM .....	17
4.18. TLI_LCD .....	18
4.19. USART .....	18
4.20. USB .....	19
4.21. Extension .....	19
4.22. GD-Link .....	20
4.23. MCU .....	21
<b>5. ROUTINE USE GUIDE .....</b>	<b>22</b>
5.1. GPIO_Running_LED .....	22
5.1.1. DEMO purpose .....	22
5.1.2. DEMO running result .....	22
5.2. GPIO_Key_Polling_mode .....	22
5.2.1. DEMO purpose .....	22
5.2.2. DEMO running result .....	22

<b>5.3. EXTI_Key_Interrupt_mode .....</b>	<b>23</b>
5.3.1. DEMO purpose .....	23
5.3.2. DEMO running result .....	23
<b>5.4. USART_Printf.....</b>	<b>23</b>
5.4.1. DEMO purpose .....	23
5.4.2. DEMO running result .....	23
<b>5.5. USART_HyperTerminal_Interrupt.....</b>	<b>24</b>
5.5.1. DEMO purpose .....	24
5.5.2. DEMO running result .....	24
<b>5.6. USART_DMA.....</b>	<b>24</b>
5.6.1. DEMO purpose .....	24
5.6.2. DEMO running result .....	24
<b>5.7. ADC_Temperature_Vrefint.....</b>	<b>25</b>
5.7.1. DEMO purpose .....	25
5.7.2. DEMO running result .....	25
<b>5.8. ADC0_ADC1_Follow_up_mode .....</b>	<b>26</b>
5.8.1. DEMO purpose .....	26
5.8.2. DEMO running result .....	26
<b>5.9. ADC0_ADC1_Regular_Parallel_mode .....</b>	<b>27</b>
5.9.1. DEMO purpose .....	27
5.9.2. DEMO running result .....	27
<b>5.10. DAC_Output_Voltage_Value .....</b>	<b>28</b>
5.10.1. DEMO purpose .....	28
5.10.2. DEMO running result .....	28
<b>5.11. I2C_EEPROM.....</b>	<b>28</b>
5.11.1. DEMO purpose .....	28
5.11.2. DEMO running result .....	28
<b>5.12. HPDF_I2S_Audio.....</b>	<b>29</b>
5.12.1. DEMO purpose .....	29
5.12.2. DEMO running result .....	30
<b>5.13. HPDF_SAI_Audio .....</b>	<b>30</b>
5.13.1. DEMO purpose .....	30
5.13.2. DEMO running result .....	30
<b>5.14. SPI_Quad_LCD.....</b>	<b>30</b>
5.14.1. DEMO purpose .....	30
5.14.2. DEMO running result .....	31
<b>5.15. OSPI_Octal_Flash .....</b>	<b>31</b>
5.15.1. DEMO purpose .....	31
5.15.2. DEMO running result .....	31
<b>5.16. EXMC_SDRAM.....</b>	<b>32</b>
5.16.1. DEMO purpose .....	32
5.16.2. DEMO running result .....	33
<b>5.17. EXMC_SDRAM_DeepSleep.....</b>	<b>34</b>
5.17.1. DEMO purpose .....	34

5.17.2. DEMO running result .....	34
<b>5.18. SDIO_SDCard .....</b>	<b>36</b>
5.18.1. DEMO purpose .....	36
5.18.2. DEMO running result .....	36
<b>5.19. CAN_Network .....</b>	<b>37</b>
5.19.1. DEMO purpose .....	37
5.19.2. DEMO running result .....	37
<b>5.20. RCU_Clock_Out .....</b>	<b>37</b>
5.20.1. DEMO purpose .....	37
5.20.2. DEMO running result .....	37
<b>5.21. PMU_Sleep_Wakeup .....</b>	<b>38</b>
5.21.1. DEMO purpose .....	38
5.21.2. DEMO running result .....	38
<b>5.22. RTC_Calendar .....</b>	<b>38</b>
5.22.1. DEMO purpose .....	38
5.22.2. DEMO running result .....	38
<b>5.23. TIMER_Breath_LED .....</b>	<b>39</b>
5.23.1. DEMO purpose .....	39
5.23.2. DEMO running result .....	39
<b>5.24. TLI_IPA .....</b>	<b>39</b>
5.24.1. DEMO purpose .....	39
5.24.2. DEMO running result .....	39
<b>5.25. DCI_OV2640 .....</b>	<b>40</b>
5.25.1. DEMO purpose .....	40
5.25.2. DEMO running result .....	40
<b>5.26. ENET .....</b>	<b>41</b>
5.26.1. FreeRTOS_tcpudp .....	41
5.26.2. Raw_tcpudp .....	44
5.26.3. Raw_webserver .....	46
<b>5.27. USB_Device .....</b>	<b>48</b>
5.27.1. CDC_ACM .....	48
5.27.2. HID_Keyboard .....	49
<b>5.28. USB_Host .....</b>	<b>50</b>
5.28.1. USB HID Host .....	50
5.28.2. MSC_Host .....	51
<b>6. REVISION HISTORY .....</b>	<b>53</b>

## List of Figures

Figure 4-1. Schematic diagram of power supply.....	10
Figure 4-2. Schematic diagram of boot option .....	11
Figure 4-3. Schematic diagram of LED function .....	11
Figure 4-4. Schematic diagram of Key function .....	11
Figure 4-5. Schematic diagram of ADC .....	12
Figure 4-6. Schematic diagram of DAC .....	12
Figure 4-7. Schematic diagram of CAN .....	12
Figure 4-8. Schematic diagram of DCI.....	13
Figure 4-9. Schematic diagram of ENET .....	13
Figure 4-10. Schematic diagram of HPDF .....	14
Figure 4-11. Schematic diagram of I2C .....	14
Figure 4-12. Schematic diagram of I2S .....	14
Figure 4-13. Schematic diagram of SAI .....	15
Figure 4-14. Schematic diagram of OSPI .....	15
Figure 4-15. Schematic diagram of SPI_LCD.....	16
Figure 4-16. Schematic diagram of SDIO .....	16
Figure 4-17. Schematic diagram of SDRAM.....	17
Figure 4-18. Schematic diagram of TLI .....	18
Figure 4-19. Schematic diagram of USART .....	18
Figure 4-20. Schematic diagram of USB .....	19
Figure 4-21. Schematic diagram of Extension.....	19
Figure 4-22. Schematic diagram of GD-Link.....	20
Figure 4-23. Schematic diagram of MCU.....	21

# List of Tables

Table 2-1. Function pin assignment.....	6
Table 6-1. Revision history .....	53

## 1. Summary

GD32H759I-EVAL uses GD32H759IMK6 as the main controller. It uses GD-Link Mini USB interface or DC-005 connector to supply 5V power. Reset, Boot, Wakeup KEY, Tamper KEY, User KEY, LED, ADC, DAC, CAN, DCI, ETHNET, HPDF, SAI, I2S, I2C\_SMBus, OSPI, SPI\_LCD, SDIO, SDRAM, TLI\_LCD, USB and USART to USB interface are also included. For more details please refer to GD32H759I-EVAL-V1.1 schematic.

## 2. Function Pin Assign

Table 2-1. Function pin assignment

功能	引脚	描述
LED	PF10	LED1
	PA6	LED2
RESET		K1-Reset
KEY	PA0	K2-Wakeup
	PC13	K3-Tamper
	PF8	K4-User
ADC	PA0_C	ADC01_IN0
DAC	PA5	DAC0_OUT1
CAN	PB5	CAN1_RX
	PB6	CAN1_TX
	PD12	CAN2_RX
	PF7	CAN2_TX
DCI	PH4	DCI_I2C1_SCL
	PB11	DCI_I2C1_SDA
	PB7	DCI_VSYNC
	PA4	DCI_HSYNC
	PE3	DCI_PIXCLK
	PA8	DCI_XCLK
	PE6	DCI_D7
	PE5	DCI_D6
	PB6	DCI_D5
	PE4	DCI_D4
	PC9	DCI_D3
	PC8	DCI_D2
	PH10	DCI_D1
	PC6	DCI_D0
ENET	PG11	ETH0_RMII_TX_EN
	PB12	ETH0_RMII_TXD0
	PG12	ETH0_RMII_TXD1

功能	引脚	描述
	PC4	ETH0_RMII_RXD0
	PC5	ETH0_RMII_RXD1
	PA7	ETH0_RMII_CRS_DV
	PC1	RMII_MDC
	PA2	RMII_MDIO
	PA1	RMII_REF_CLK
HPDF	PB0	HPDF_CKOUT
	PB1	HPDF_DATA
I2C	PH4	I2C1_SCL
	PB11	I2C1_SDA
I2S	PC3	I2S1_SD
	PD3	I2S1_CK
	PB9	I2S1_WS
	PC6	I2S1_MCK
SAI	PG9	SAI1_FS
	PH2	SAI1_SCK
	PH3	SAI1_MCLK
	PG10	SAI1_SD
OSPI	PB10	OSPI0_NCS
	PA3	OSPI0_CLK
	PD11	OSPI0_IO0
	PC10	OSPI0_IO1
	PE2	OSPI0_IO2
	PD13	OSPI0_IO3
	PD4	OSPI0_IO4
	PD5	OSPI0_IO5
	PD6	OSPI0_IO6
	PD7	OSPI0_IO7
SPI_LCD	PH6	SPI4_SCK
	PG13	LCD_PWM
	PF6	SPI4_NSS
	PF9	SPI4_IO0
	PH7	SPI4_IO1
	PH8	SPI4_IO2
	PH9	SPI4_IO3
SDIO	PD2	SDIO_CMD
	PC12	SDIO_CLK
	PB13	SDIO_DAT0
	PC9	SDIO_DAT1
	PC10	SDIO_DAT2
	PC11	SDIO_DAT3



功能	引脚	描述
SDRAM	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
	PC0	EXMC_D12
	PD8	EXMC_D13
	PD9	EXMC_D14
	PD10	EXMC_D15
	PE0	EXMC_NBL0
	PE1	EXMC_NBL1
	PC3	EXMC_SDCKE0
	PG4	EXMC_BA0
	PG5	EXMC_BA1
	PG8	EXMC_SDCLK
	PG15	EXMC_SDNCAS
	PF11	EXMC_SDNRAS
	PC2	EXMC_SDNE0
	PH5	EXMC_SDNWE
	PF0	EXMC_A0
	PF1	EXMC_A1
	PF2	EXMC_A2
	PF3	EXMC_A3
	PF4	EXMC_A4
	PF5	EXMC_A5
	PF12	EXMC_A6
	PF13	EXMC_A7
	PF14	EXMC_A8
	PF15	EXMC_A9
	PG0	EXMC_A10
	PG1	EXMC_A11
	PG2	EXMC_A12
TLI_LCD	PG7	LCD_CLK
	PE15	LCD_HSYNC

功能	引脚	描述
	PA7	LCD_VSYNC
	PF10	LCD_DE
	PG3	LCD_Touch_PENIRQ
	PF9	LCD_SPI4_MOSI
	PH7	LCD_SPI4_MISO
	PH6	LCD_SPI4_SCK
	PF6	LCD_SPI4_NSS
	PG13	LCD_PWM_BackLight
	PF8	LCD_Touch_Busy
	PH2	LCD_R0
	PH3	LCD_R1
	PH8	LCD_R2
	PH9	LCD_R3
	PA5	LCD_R4
	PH11	LCD_R5
	PH12	LCD_R6
	PG6	LCD_R7
	PB1	LCD_G0
	PB0	LCD_G1
	PH13	LCD_G2
	PH14	LCD_G3
	PH15	LCD_G4
	PC1	LCD_G5
	PC7	LCD_G6
	PD3	LCD_G7
	PG14	LCD_B0
	PG12	LCD_B1
	PG10	LCD_B2
	PG11	LCD_B3
	PC11	LCD_B4
	PB5	LCD_B5
	PB8	LCD_B6
	PB9	LCD_B7
USART	USART0_TX	PA9
	USART0_RX	PA10
USB	PA9	USBHS0_VBUS
	PA10	USBHS0_ID
	USBHS0_DM	USBHS0_DM
	USBHS0_DP	USBHS0_DP
	PB12	USBHS1_VBUS
	PB13	USBHS1_ID

功能	引脚	描述
	USBHS1_DM	USBHS1_DM
	USBHS0_DP	USBHS1_DM

### 3. Getting started

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

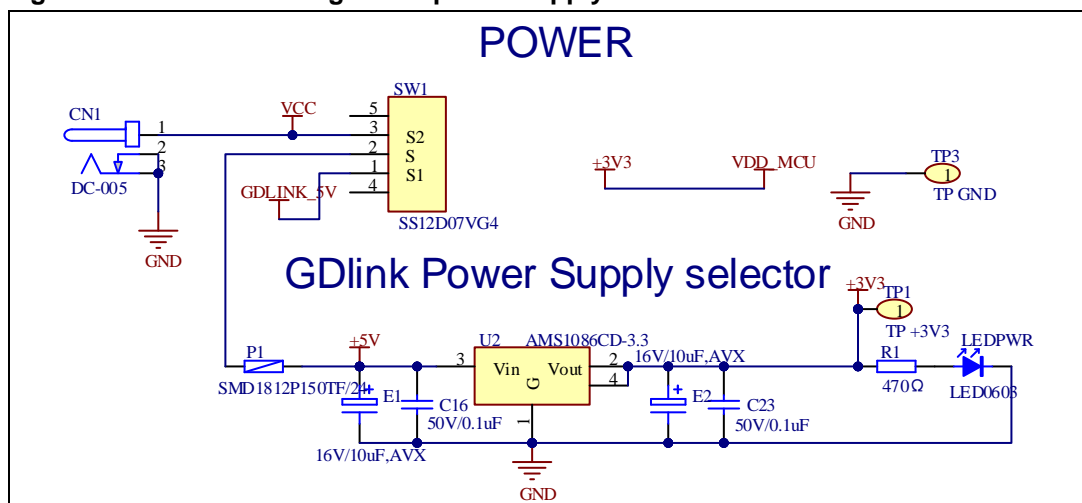
There are Keil version and IAR version of all projects. Keil version of the projects are created based on Keil MDK-ARM 5.29 uVision5. IAR version of the projects are created based on IAR Embedded Workbench for ARM 8.32.1. During use, the following points should be noted:

1. If you use Keil uVision5 to open the project. In order to solve the "Device Missing (s)" problem, you can install GigaDevice.GD32H7xx\_DFP.1.0.0.pack.
2. If you use IAR to open the project, install IAR\_GD32H7xx\_ADDON\_1.0.0.exe to load the associated files.

### 4. Hardware layout overview

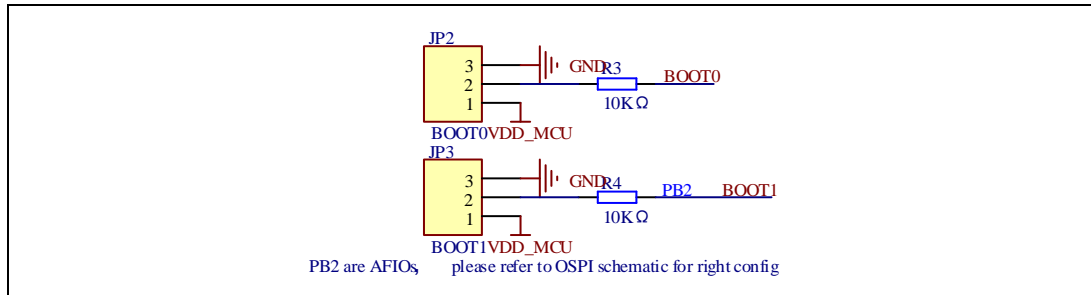
#### 4.1. Power supply

Figure 4-1. Schematic diagram of power supply



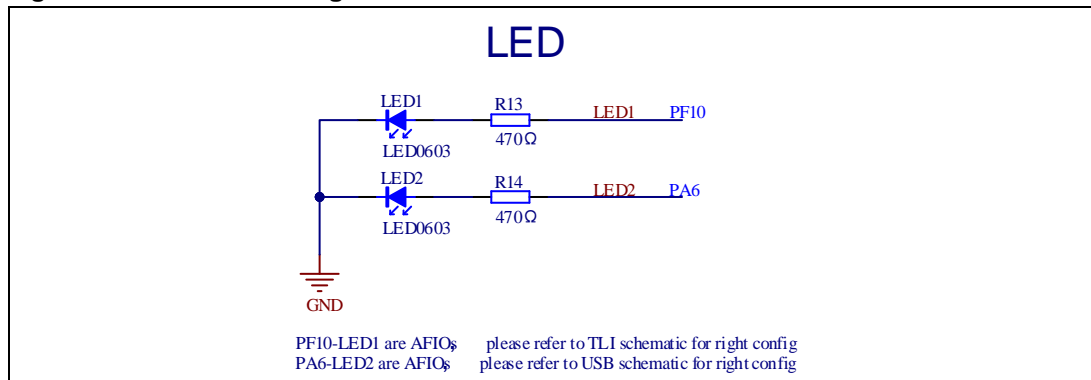
## 4.2. Boot option

Figure 4-2. Schematic diagram of boot option



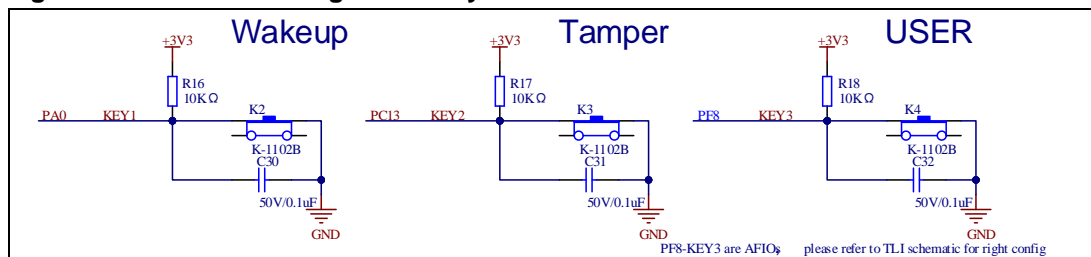
## 4.3. LED

Figure 4-3. Schematic diagram of LED function



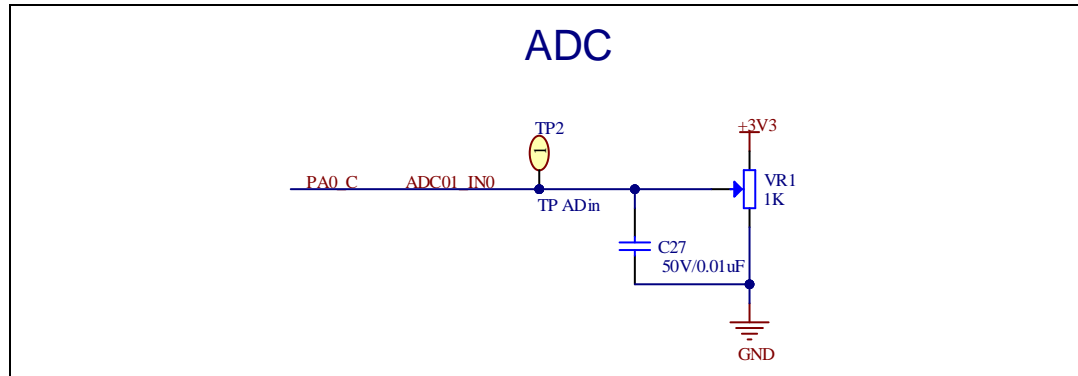
## 4.4. KEY

Figure 4-4. Schematic diagram of Key function



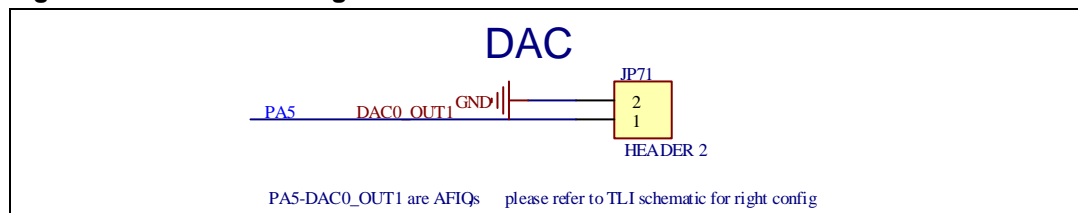
## 4.5. ADC

Figure 4-5. Schematic diagram of ADC



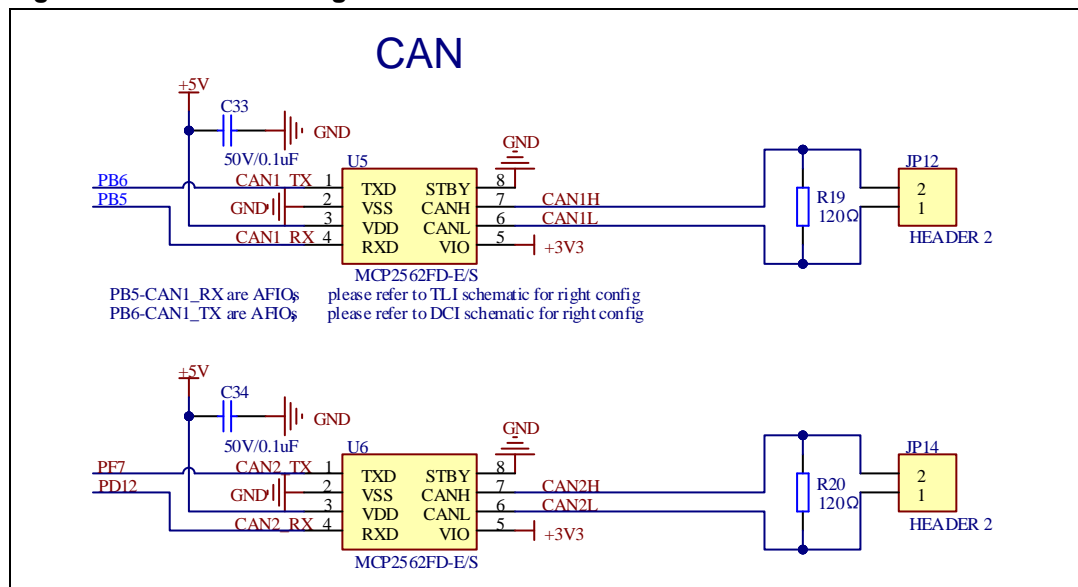
## 4.6. DAC

Figure 4-6. Schematic diagram of DAC



## 4.7. CAN

Figure 4-7. Schematic diagram of CAN



# DCI

## DCI 8bit

PH4 DCI\_I2C1\_SCL  
PB11 DCI\_I2C1\_SDA  
PB7 DCI\_VSYNC  
PA4 DCI\_HSYNC  
PE3 DCI\_PIXCLK  
PA8 DCI\_XCLK  
  
PE6 DCI\_D7  
PE5 DCI\_D6  
PB6 DCI\_D5  
PE4 DCI\_D4  
PC9 DCI\_D3  
PC8 DCI\_D2  
PH10 DCI\_D1  
PC6 DCI\_D0

DCI\_I2C1\_SCL  
DCI\_VSYNC  
DCI\_PIXCLK  
DCI\_D7  
DCI\_D5  
DCI\_D3  
DCI\_D1

DCI\_I2C1\_SDA  
DCI\_HSYNC  
DCI\_XCLK  
DCI\_D6  
DCI\_D4  
DCI\_D2  
DCI\_D0

8x2P2.54

**OV2640**

Short JP60(1,2)for DCI function  
Short JP60(2,3)for ETH function

HEADER3

Short JP61(1,2)for DCI function  
Short JP61(2,3)for CAN function

HEADER3

Short JP62(1,2)for DCI function  
Short JP62(2,3)for SDIO function

HEADER3

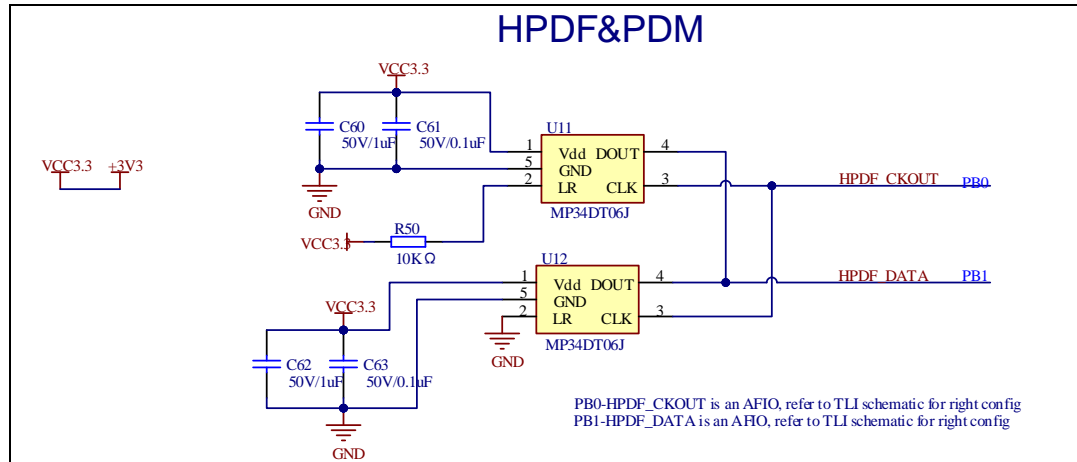
Short JP63(1,2)for DCI function  
Short JP63(2,3)for I2S function

HEADER3

# ENET

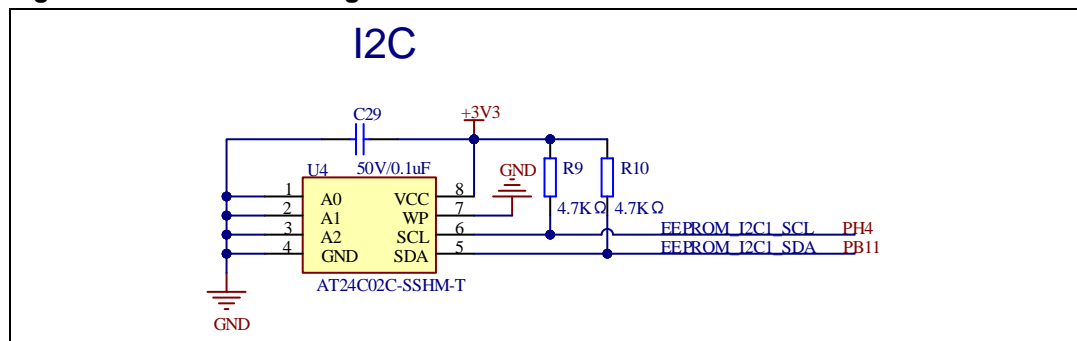
## 4.10. HPDF

Figure 4-10. Schematic diagram of HPDF



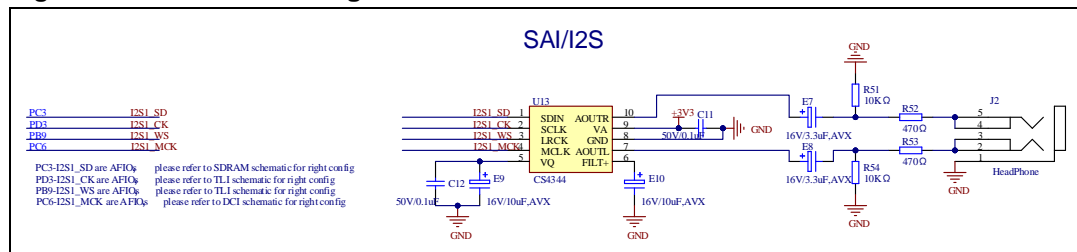
## 4.11. I2C

Figure 4-11. Schematic diagram of I2C



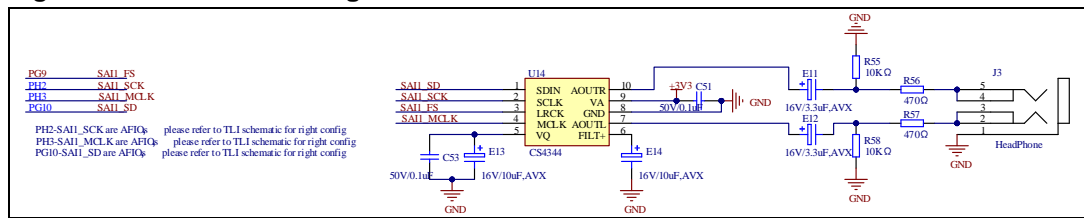
## 4.12. I2S

Figure 4-12. Schematic diagram of I2S



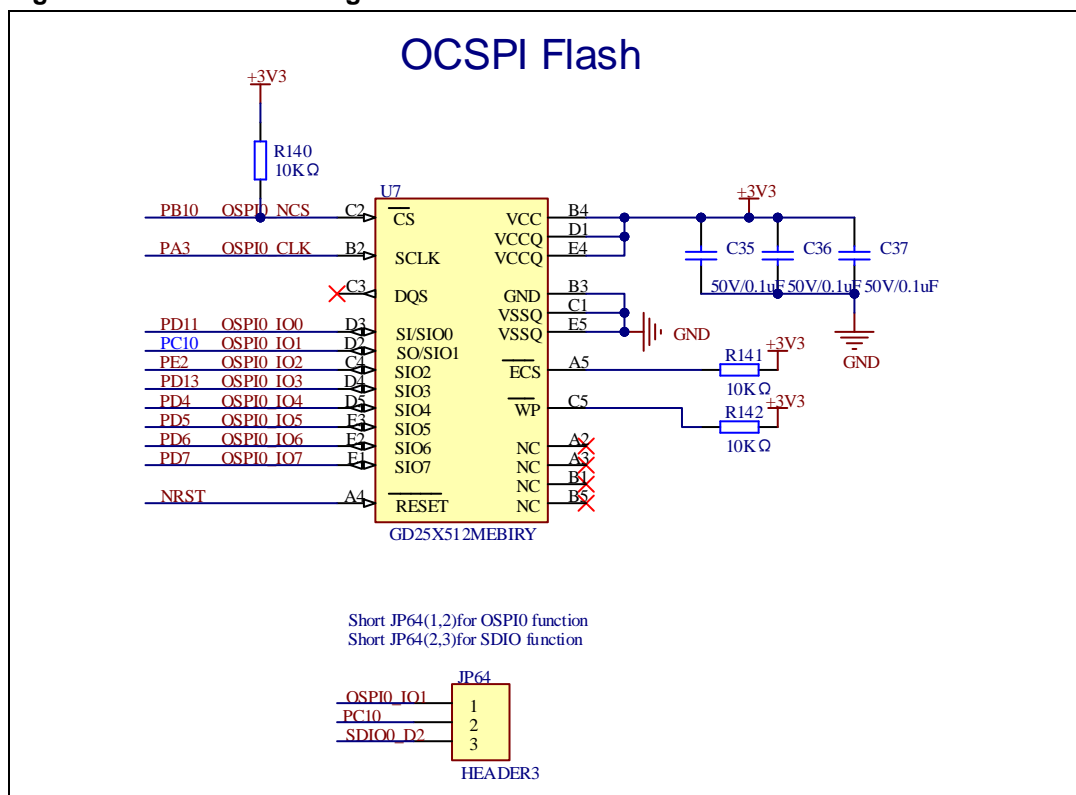
#### 4.13. SAI

**Figure 4-13. Schematic diagram of SAI**



#### 4.14. OSPI

**Figure 4-14. Schematic diagram of OSPI**

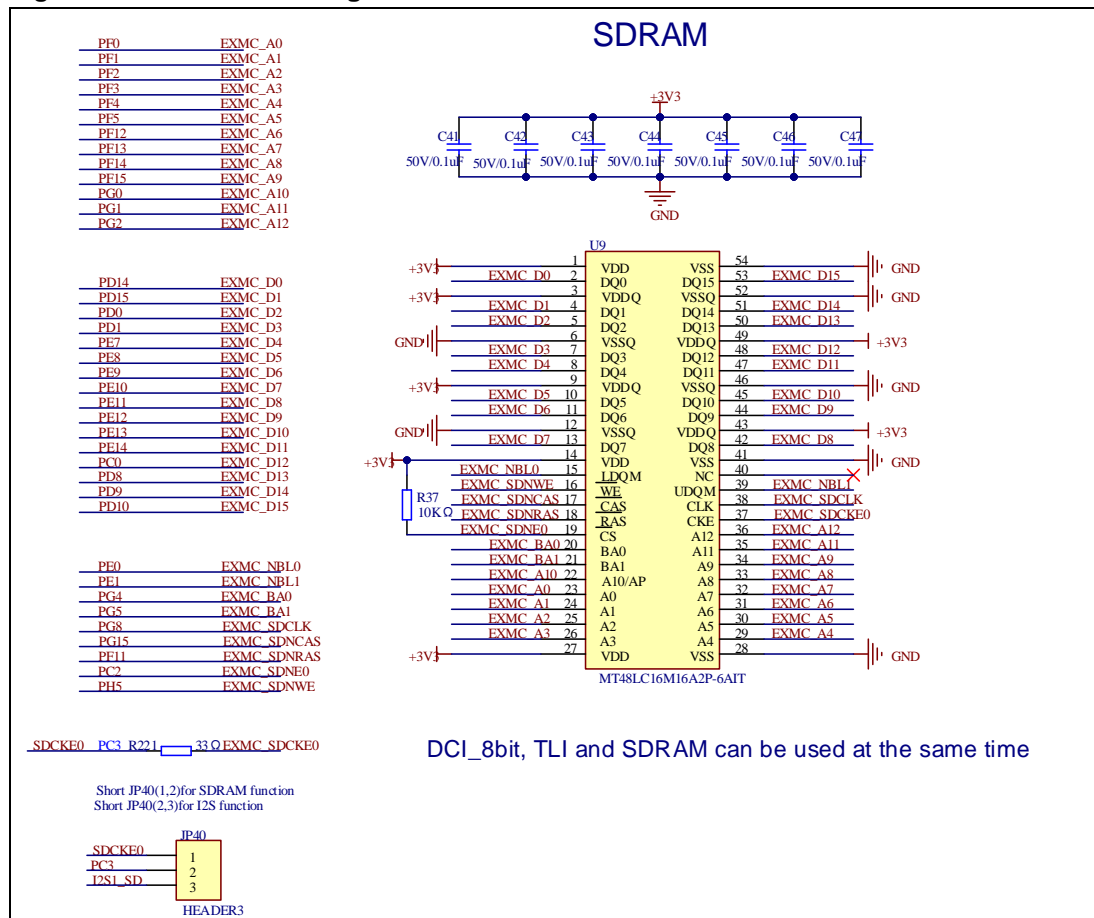






## 4.17. SDRAM

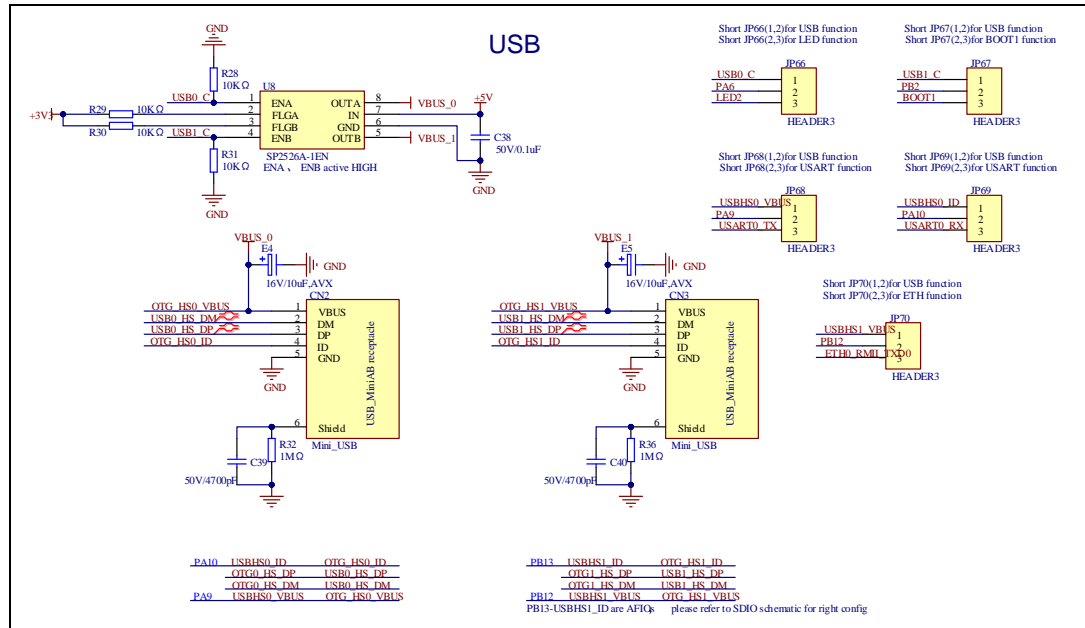
Figure 4-17. Schematic diagram of SDRAM





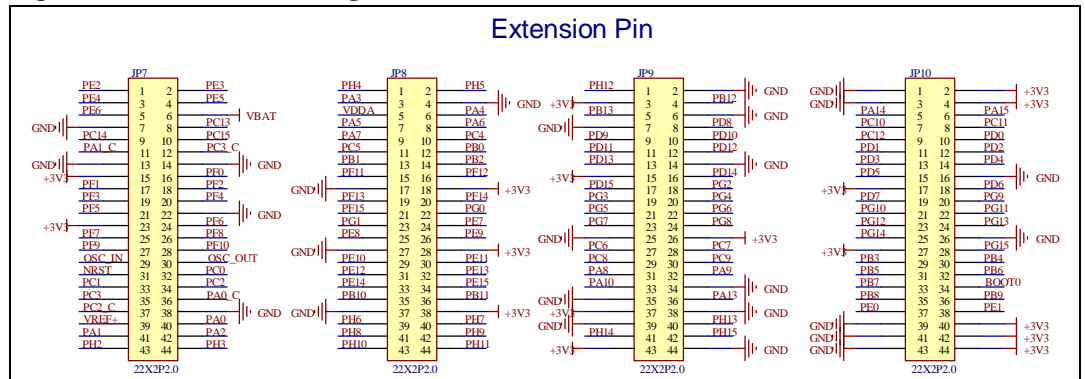
## 4.20. USB

Figure 4-20. Schematic diagram of USB



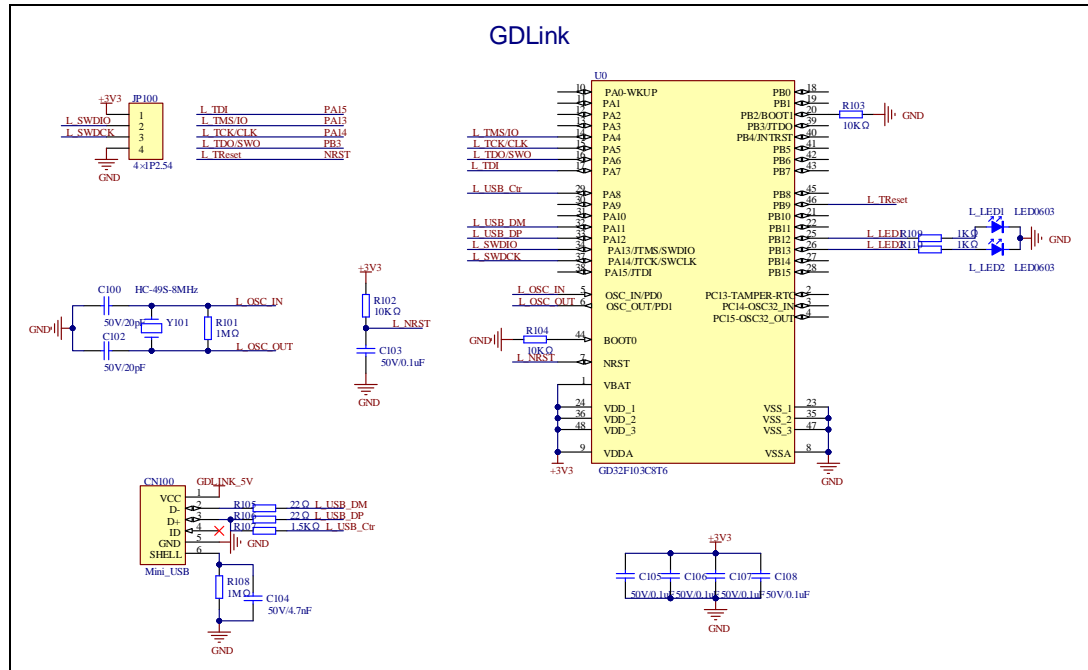
## 4.21. Extension

Figure 4-21. Schematic diagram of Extension



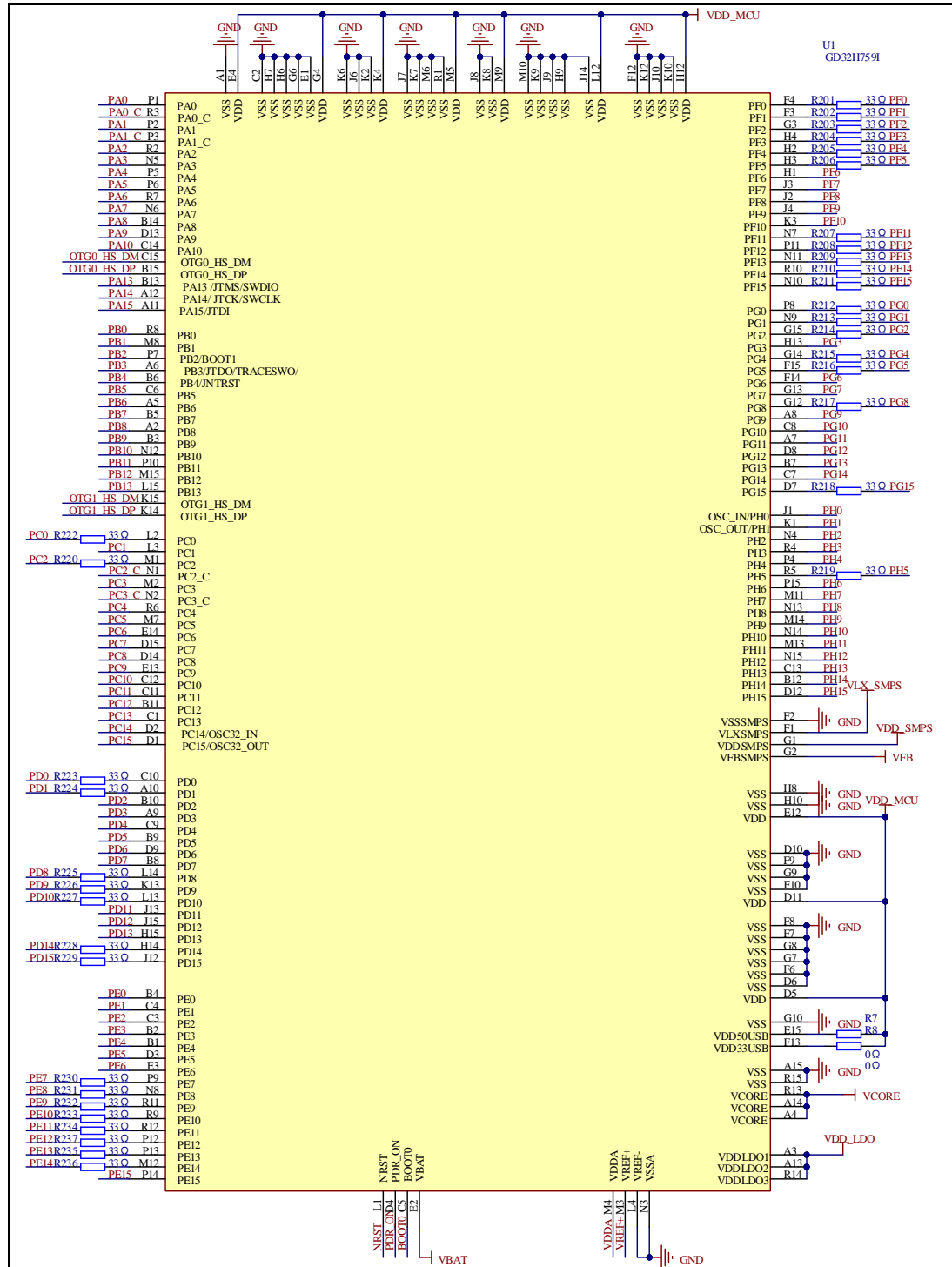
## 4.22. GD-Link

Figure 4-22. Schematic diagram of GD-Link



## 4.23. MCU

Figure 4-23. Schematic diagram of MCU



## 5. Routine use guide

### 5.1. GPIO\_Running\_LED

#### 5.1.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

- Learn to use GPIO control the LED.
- Learn to use SysTick to generate 1ms delay.

GD32H759I-EVAL-V1.1 board has two LEDs. The LED1 and LED2 are controlled by GPIO. This demo will show how to light the LEDs.

#### 5.1.2. DEMO running result

Download the program <01\_GPIO\_Running\_LED> to the EVAL board, LED1, LED2 will change the state like running water and then repeat the whole process over and over again.

### 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.

GD32H759I-EVAL-V1.1 board has four keys and two LEDs. The four keys are Reset key, Tamper key, Wakeup key and User key. The LED1 and LED2 are controlled by GPIO.

This demo will show how to use the Tamper key to control the LED2. When press down the Tamper Key, it will check the input value of the IO port. If the value is 0 and will wait for 100ms. Check the input value of the IO port again. If the value still is 0, it indicates that the button is pressed successfully and toggle LED2.

#### 5.2.2. DEMO running result

Download the program <02\_GPIO\_Key\_Polling\_mode> to the EVAL board. Press down the Tamper Key, LED2 will be turned on. Press down the Tamper Key again, LED2 will be turned off.

## 5.3. EXTI\_Key\_Interrupt\_mode

### 5.3.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

- Learn to use GPIO control the LED and the KEY
- Learn to use EXTI to generate external interrupt

GD32H759I-EVAL board has four keys and two LEDs. The four keys are Reset key, Tamper key, Wakeup key, User key. The LED1 and LED2 are controlled by GPIO.

This demo will show how to use the EXTI interrupt line to control the LED2. When press down the Tamper key, it will produce an interrupt. In the interrupt service function, the demo will toggle LED2.

### 5.3.2. DEMO running result

Download the program <03\_EXTI\_Key\_Interrupt\_mode> to the EVAL board. 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

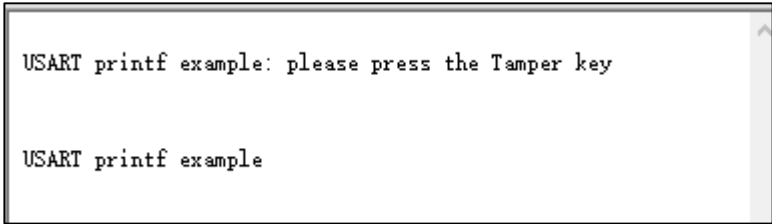
This demo includes the following functions of GD32 MCU:

- 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 USART. Firstly, all the LEDs flash 2 times for test. Then, this implementation outputs "USART printf example: please press the Tamper key" on the HyperTerminal using USART. Press the Tamper key, the serial port will output "USART printf example".

The output information via the HyperTerminal is as following:



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

USART printf example
```



## 5.5. USART\_HyperTerminal\_Interrupt

### 5.5.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

- Learn to use the USART transmit and receive interrupts to communicate with the HyperTerminal

### 5.5.2. DEMO running result

Download the program <05\_USART\_HyperTerminal\_Interrupt> to the EVAL board and connect serial cable to USART. Firstly, all the LEDs are turned on and off for test. Then, the USART sends the tx\_buffer array (from 0x00 to 0xFF) to the hyperterminal and waits for receiving data from the hyperterminal that you must send. The string that you have sent is stored in the rx\_buffer array. The receive buffer have a BUFFER\_SIZE bytes as maximum. After that, compare tx\_buffer with rx\_buffer. If tx\_buffer is same with rx\_buffer, LED1, LED2 flash by turns. Otherwise, LED1, LED2 toggle together.

The output information via the HyperTerminal is as following:

```
00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F 10 11 12 13 14 15 16 17 18 19 1A
1B 1C 1D 1E 1F 20 21 22 23 24 25 26 27 28 29 2A 2B 2C 2D 2E 2F 30 31 32 33 34 35
36 37 38 39 3A 3B 3C 3D 3E 3F 40 41 42 43 44 45 46 47 48 49 4A 4B 4C 4D 4E 4F 50
51 52 53 54 55 56 57 58 59 5A 5B 5C 5D 5E 5F 60 61 62 63 64 65 66 67 68 69 6A 6B
6C 6D 6E 6F 70 71 72 73 74 75 76 77 78 79 7A 7B 7C 7D 7E 7F 80 81 82 83 84 85 86
87 88 89 8A 8B 8C 8D 8E 8F 90 91 92 93 94 95 96 97 98 99 9A 9B 9C 9D 9E 9F A0 A1
A2 A3 A4 A5 A6 A7 A8 A9 AA AB AC AD AE AF B0 B1 B2 B3 B4 B5 B6 B7 B8 B9 BA BB BC
BD BE BF C0 C1 C2 C3 C4 C5 C6 C7 C8 C9 CA CB CC CD CE CF D0 D1 D2 D3 D4 D5 D6 D7
D8 D9 DA DB DC DD DE DF E0 E1 E2 E3 E4 E5 E6 E7 E8 E9 EA EB EC ED EE EF F0 F1 F2
F3 F4 F5 F6 F7 F8 F9 FA FB FC FD FE FF
```

## 5.6. USART\_DMA

### 5.6.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

- Learn to use the USART transmit and receive data using DMA

### 5.6.2. DEMO running result

Download the program <06\_USART\_DMA> to the EVAL board and connect serial cable to USART. Firstly, the USART sends "USART DMA interrupt receive and transmit example, please input 32 bytes:" to hyperterminal and waits for receiving 32 bytes data from the hyperterminal that you must send. After MCU receives the data, the USART will continue to

output the received data to the hyper terminal.

The output information via the HyperTerminal is as following:



## 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 18(temperature sensor channel), channel 19 (V<sub>REFINT</sub> channel)

### 5.7.2. DEMO running result

Download the program <07\_ADC\_Temperature\_Vrefint> to the EVAL board and connect serial cable to USART, open the HyperTerminal.

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

**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 31 degrees Celsius  
the reference voltage data is 1.184V  
  
the temperature data is 31 degrees Celsius  
the reference voltage data is 1.184V  
  
the temperature data is 32 degrees Celsius  
the reference voltage data is 1.184V  
  
the temperature data is 31 degrees Celsius  
the reference voltage data is 1.183V  
  
the temperature data is 30 degrees Celsius  
the reference voltage data is 1.184V
```

## 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 EVAL board and connect serial cable to USART, open the HyperTerminal. PA4 pin connect to the external voltage input. PA0\_C is the output voltage of the slide rheostat VR1 on board.

TIMER1\_CH1 is the trigger source of ADC0 and ADC1. When the rising edge of TIMER1\_CH1 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 TIMER1\_CH1 coming, the value of the ADC0 conversion of PA0\_C 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 PA4 pin is stored into the high half word of `adc_value[0]`. When the second rising edge of TIMER1\_CH1 coming, the value of the ADC0 conversion of PA4 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 PA0\_C 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 0x1F4520EA
the data adc_value[1] is 0x20DD1EE9

the data adc_value[0] is 0x1F4720E9
the data adc_value[1] is 0x20E91EEC

the data adc_value[0] is 0x1F5920EA
the data adc_value[1] is 0x20F01EFD

the data adc_value[0] is 0x1F6620E6
the data adc_value[1] is 0x20E91FOB

the data adc_value[0] is 0x1F7720E6
the data adc_value[1] is 0x20E91F1C
```

## 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 EVAL board and connect serial cable to USART, open the HyperTerminal. PA4 pin connect to the external voltage input. PA0\_C is the output voltage of the slide rheostat VR1 on board.

TIMER1\_CH1 is the trigger source of ADC0 and ADC1. When the rising edge of TIMER1\_CH1 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 TIMER1\_CH1 coming, the value of the ADC0 conversion of PA0\_C pin is stored into the low half word of adc\_value[0], the value of the ADC1 conversion of PA4 pin is stored into the high half word of adc\_value[0]. When the second rising edge of TIMER1\_CH1 coming, the value of the ADC0 conversion of PA4 pin is stored into the low half word of adc\_value[1], the value of the ADC1 conversion of PA0\_C 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 0x1FA523D9
the data adc_value[1] is 0x23C51FB4

the data adc_value[0] is 0x1FA923DA
the data adc_value[1] is 0x23D91FBA

the data adc_value[0] is 0x1FBB23D0
the data adc_value[1] is 0x23D91FC5

the data adc_value[0] is 0x1FC423D9
the data adc_value[1] is 0x23D81FCD

the data adc_value[0] is 0x1FD523D9
the data adc_value[1] is 0x23D91FDB
```

## 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, The digital value is 0x7FF0, its converted analog voltage should be 1.65V (VREF/2), using the voltmeter to measure PA5, its value is 1.65V. And the signal also 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 USART, open the HyperTerminal to show the print message.

Firstly, the data of 256 bytes will be written to the EEPROM from the address 0x00 and printed by the serial port. Then, reading the EEPROM from address 0x00 for 256 bytes and the result will be printed. Finally, compare the data that were written to the EEPROM and the data that

were read from the EEPROM. If they are the same, the serial port will output "I2C-AT24C02 test passed!" and the two LEDs lights flashing, otherwise the serial port will output "Err:data read and write aren't matching." and all the two LEDs light.

The output information via the serial port is as following.

```
I2C-24C02 configured...

The I2C is hardware interface
The speed is 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. HPDF\_I2S\_Audio

### 5.12.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

- Learn to use I2S module to output audio data

- Learn to use HPDF interface to process PDM data

The GD32H759I-EVAL-V1.1 board integrates HPDF and I2S modules. Jump JP55 and JP56 to HPDF, jump JP40/Jp54/Jp58 and JP63 to I2S. The two modules can cooperate with each other to play the audio signal of the microphone. This example demonstrates the process of collecting dual-channel audio data through the HPDF of the EVAL board and playing dual-channel audio using the I2S interface.

### 5.12.2. DEMO running result

Download the program <12\_HPDI\_I2S\_Audio> to the development board and run, plug in the headset to hear the sound from the microphone.

## 5.13. HPDI\_SAI\_Audio

### 5.13.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

- Learn to use SAI module to output audio data
- Learn to use HPDI interface to process PDM data

The GD32H759I-EVAL-V1.1 board integrates HPDI and SAI modules. Jump JP55 and JP56 to HPDI, jump JP34/Jp43/Jp49 to SAI. The two modules can cooperate with each other to play the audio signal of the microphone. This example demonstrates the process of collecting dual-channel audio data through the HPDI of the EVAL board and playing dual-channel audio using the SAI interface.

### 5.13.2. DEMO running result

Download the program <13\_HPDI\_SAI\_Audio> to the development board and run, plug in the headset to hear the sound from the microphone

## 5.14. SPI\_Quad\_LCD

### 5.14.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

- Learn to use SPI unit with single / four wire function to control GC9B71 LCD, to transmit instruction and pixel data.

GD32H7xxI-EVAL-V1.0 board integrates SPI mode can communicate with external SPI LCD devices. GC9B71 is SPI interface LCD, with 320x386 resolution ratio, RGB565 character, support single wire instructions / address / parameters transmission, four wire pixel data

transmission, the maximum transfer rate is 50 MHz.

### 5.14.2. DEMO running result

Jump the JP41, JP45, JP46 to QSPI, and connect the LCD to the JP13. Download the program <14\_SPI\_Quad\_LCD> to the development board. By observing the LCD screen can observe operation. The following is the experimental results.



## 5.15. OSPI\_Octal\_Flash

### 5.15.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

- Learn to use OSPI unit to read and write Nor Flash with the Octal-SPI interface

GD32H759I-EVAL-V1.1 board integrates OSPI mode can communicate with external NOR Flash devices. The SPI Nor Flash is a Flash memory chip GD25X512ME which size is 512Mbit, the chip supports standard SPI and Octal-SPI operation instructions.

### 5.15.2. DEMO running result

The computer serial port line connected to the USART 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 JP68, JP69 to USART, jump the JP50, JP66 to LED, and jump the JP64 to OSPI.

Download the program <15\_OSPI\_Octal\_Flash> to the EVAL board, the HyperTerminal



software can observe the operation condition and will display the ID of the flash, the data of txbuffer1, txbuffer2 and txbuffer3 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 success message, otherwise, the serial port will output failure message. At last, turn on and off the leds one by one. The following is the experimental results.

```
#####
```

```
OSPI read flash ID and read/write with 108 lines in indirect/memory mapped mode test!
```

```
The device ID is 0xC8481AFF
```

```
The data written with 1 line in indirect mode to flash is:
```

```
GD32H759I_EVAL octal-flash SPI mode with 1 line in indirect mode read&write test!
```

```
The data read with 1 line in indirect mode from flash is:
```

```
GD32H759I_EVAL octal-flash SPI mode with 1 line in indirect mode read&write test!
```

```
OSPI read/write w
```

```
ith 1 line in indirect test success!
```

```
The data written with 8 lines in indirect mode to flash is:
```

```
GD32H759I_EVAL octal-flash OSPI mode with 8 lines in indirect mode read&write test!
```

```
The data read with 8 lines in indirect mode from flash is:
```

```
GD32H759I_EVAL octal-flash OSPI mode with 8 lines in indirect mode read&write test!
```

```
OSPI read/write with 8 lines in indirect test success!
```

```
The data written in indirect mode to flash is:
```

```
GD32H759I_EVAL octal-flash memory mapped read test!
```

```
The data read
```

```
in memory mapped mode from flash is:
```

```
GD32H759I_EVAL octal-flash memory mapped read test!
```

```
OSPI read in memory mapped mode test success!
```

## 5.16. EXMC\_SDRAM

### 5.16.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

■ Learn to EXMC control the SDRAM

## 5.16.2. DEMO running result

GD32H759I-EVAL board has EXMC module to control SDRAM. Download the program <16\_EXMC\_SDRAM> to the EVAL board. This demo shows the write and read operation process of SDRAM memory by EXMC module. If the test succeed, LED1 will be turned on. Otherwise, turn on the LED2. Information via a HyperTerminal output as following:

```
SDRAM initialized!
SDRAM write data completed!
SDRAM read data completed!
Check the data!
SDRAM test succeeded!
The data is:
0 1 2 3 4 5 6 7 8 9 a b c d e f
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

0 1 2 3 4 5 6 7 8 9 a b c d e f
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

0 1 2 3 4 5 6 7 8 9 a b c d e f
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
```

## **5.17. EXMC\_SDRAM\_DeepSleep**

### **5.17.1. DEMO purpose**

This demo includes the following functions of GD32 MCU:

- Learn to use EXMC control the SDRAM
- Learn to use deepsleep mode

### **5.17.2. DEMO running result**

GD32H759I-EVAL board has EXMC module to control SDRAM. Download the program <17\_EXMC\_SDRAM\_DeepSleep> to the EVAL board. This demo shows how to use SDRAM in the deepsleep mode. Firstly, MCU works in the normal mode, SDRAM auto-refresh cycles are performed by MCU, we write the specified data to the SDRAM. Secondly, we make the MCU to deepsleep mode, at the time, SDRAM auto-refresh cycles are performed by itself and LED1 will light on. Thirdly, press the wakeup key to wake up MCU, compare the data which read from SDRAM with the write data, if the test pass, LED2 will be turned on. Otherwise, turn off the LED2. Information via a HyperTerminal output as following:

```

SDRAM initialized!
SDRAM write data completed!
Enter deepsleep mode!
Press the Wakeup key to wakeup the MCU!

Wakeup key has been pressed!
SDRAM read data completed!
Check the data!
SDRAM test succeeded!
The data is:
  0    1    2    3    4    5    6    7    8    9    a    b    c    d    e    f
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

  0    1    2    3    4    5    6    7    8    9    a    b    c    d    e    f
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

  0    1    2    3    4    5    6    7    8    9    a    b    c    d    e    f
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

```

## 5.18. SDIO\_SDCard

### 5.18.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

- Learn to use SDIO to single block or multiple block write and read
- Learn to use SDIO to erase, lock and unlock a SD card

GD32H759I-EVAL board has a secure digital input/output interface (SDIO) which defines the SD/SD I/O / e•MMC card host interface. This demo will show how to use SDIO to operate on SD card.

### 5.18.2. DEMO running result

Jump the JP59/JP60/JP62 to SDIO. Jump the JP68/69 to USART. Download the program <18\_SDIO\_SDCardTest> to the EVAL board and run. Connect serial cable to USART, open the HyperTerminal. Firstly, all the LEDs are turned on and off for test. Then initialize the card and print out the information of the card. After that, test the function of single block operation, lock and unlock operation, erase operation and multiple blocks operation. If any error occurs, print the error message and turn on LED1, LED2. Otherwise, turn off all the LEDs.

Uncomment the macro DATA\_PRINT to print out the data and display them through HyperTerminal. Set bus mode(1-bit or 4-bit), bus speed mode(default speed mode or high speed mode) and data transfer mode(polling mode or DMA mode) by select different macros.

The output information via the HyperTerminal is as following:

```
Card init success!

Card information:
## Card version 3.0x ##
## SDHC card ##
## Device size is 7782400KB ##
## Block size is 512B ##
## Block count is 15564800 ##
## CardCommandClasses is: 5b5 ##
## Block operation supported ##
## Erase supported ##
## Lock unlock supported ##
## Application specific supported ##
## Switch function supported ##

Card test:
Block write success!
Block read success!
The card is locked!
Erase failed!
The card is unlocked!
Erase success!
Block read success!
Multiple block write success!
Multiple block read success!
```

## 5.19. CAN\_Network

### 5.19.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

- Learn to use CAN to realize communication between two CAN nodes.
- Learn to use USART module to communicate with the HyperTerminal.

### 5.19.2. DEMO running result

Download the program <19\_CAN\_Network> to the EVAL board. Connect L pin to L pin, and H pin to H pin of JP12 and JP14. Jump JP61 and JP53 to CAN, jump JP66 to LED, jump JP68 and JP69 to USART, and connect serial cable to USART. When user press the WAKEUP key, the frames are sent and the data are printed. When the frames are received, the data of receiving will be printed and the LED2 will toggle once. The output information via the serial port is as following.

```
communication test CAN1 and CAN2, please press WAKEUP key to start!

CAN2 transmit data:
a1 a2 a3 a4 a5 a6 a7 a8
CAN1 receive data:
a1 a2 a3 a4 a5 a6 a7 a8
```

## 5.20. RCU\_Clock\_Out

### 5.20.1. 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

### 5.20.2. DEMO running result

Download the program <20\_RCU\_Clock\_Out> to the EVAL board and run. Connect serial cable to USART, open the HyperTerminal. When the program is running, HyperTerminal will display the initial information. Then user can choose the type of the output clock by pressing the TAMPER key. After pressing, the corresponding LED will be turned on and HyperTerminal will display which mode be selected. The frequency of the output clock can be observed through the oscilloscope by PA8 and PC9 pin.

Information via a serial port output as following:

```
/===== Gigadevice Clock Output Demo =====/  
press tamper key to select clock output source  
CK_OUT1: system clock, DIV: 8  
CK_OUT0: IRC64M, DIV: 1  
CK_OUT0: IRC48M, DIV: 1  
CK_OUT1: IRC32K, DIV: 1  
CK_OUT0: LXTAL, DIV: 1  
CK_OUT0: HXTAL, DIV: 1  
CK_OUT1: PLL1R, DIV: 8  
CK_OUT1: PLL2R, DIV: 8
```

## 5.21. PMU\_Sleep\_Wakeup

### 5.21.1. 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

### 5.21.2. DEMO running result

Download the program < 21\_PMU\_sleep\_wakeup > to the EVAL board, connect serial cable to USART. After power-on, all the LEDs are off. The MCU will enter sleep mode and the software stop running. When the USART receives a byte of data from the HyperTerminal, the MCU will wake up from a receive interrupt. And all the LEDs will flash together.

## 5.22. RTC\_Calendar

### 5.22.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

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

### 5.22.2. DEMO running result

Download the program <19\_RTC\_Calendar> to the EVAL board and run. Connect serial cable to USART, open the HyperTerminal. After start-up, the program will ask to set the time on the HyperTerminal. The calendar will be displayed on the HyperTerminal.

## 5.23. TIMER\_Breath\_LED

### 5.23.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

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

GD32H759I-EVAL-V1.1 board has two LEDs. The LED1 and LED2 are controlled by GPIO.

### 5.23.2. DEMO running result

Use the DuPont line to connect the TIMER0\_CH3 (PC7) and LED1 (PF10), and then download the program <23\_TIMER\_Breath\_LED> to the board and run. PC7 should not be reused by other peripherals.

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

## 5.24. TLI\_IPA

### 5.24.1. DEMO purpose

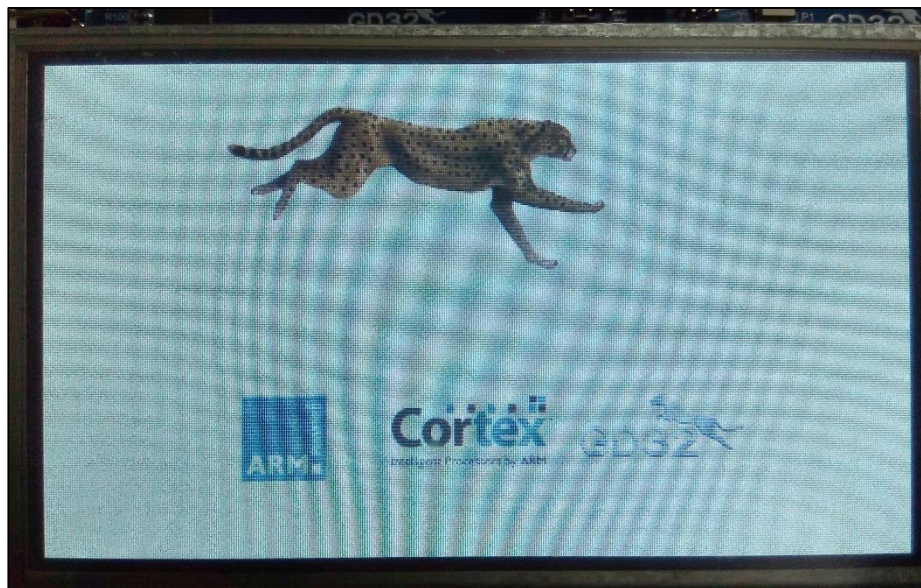
This demo includes the following functions of GD32 MCU:

- Learn to use TLI to control LCD for displaying different images
- Learn to use IPA to process image data

### 5.24.2. DEMO running result

Connect jumper JP60 / JP61 / JP62 / JP63 to DCI, jumper JP41 / JP43 / JP44 / JP45 / JP46 / JP47 / JP48 / JP49 / JP50 / JP51 / JP52 / JP53 / JP54 / JP55 / JP56 / JP57 / JP58 / JP59 to LCD, and download the program <24\_TLI\_IPA> to the EVAL board and run. After downloading program to board, a running cheetah on the background of GD logo is appeared on the LCD, which outputs as following. DC-5V power supply is recommended due to the large current consumption caused by LCD screen.





## 5.25. DCI\_OV2640

### 5.25.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

- Learn to use DCI interface capture image from OV2640 camera
- Learn to use TLI interface display the captured image

### 5.25.2. DEMO running result

Connect jumper JP60 / JP61 / JP62 / JP63 to DCI, jumper JP41 / JP43 / JP44 / JP45 / JP46 / JP47 / JP48 / JP49 / JP50 / JP51 / JP52 / JP53 / JP54 / JP55 / JP56 / JP57 / JP58 / JP59 to LCD, jumper JP42 to User key, jumper JP40 to SDRAM. Download the program<25\_DCI\_OV2640>to the GD32H759I-EVAL-V1.1 board, the correct installation of LCD display and OV2640 camera to the development board. After power on, you can observe the capture image of camera displayed on the LCD screen, you can press the user key to take photo and press tamper key to display photo. You can also return to the camera capture state when press the wakeup key on the development board.



## 5.26. ENET

### 5.26.1. 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 GD32H759I-EVAL-V1.1 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.

JP48, JP51, JP57, JP59, JP60, JP70 must be fitted. JP68, JP69 jump to Usart.

It is configured in RMII mode, and 25MHz oscillator is used, the system clock is configured to 600MHz.

This demo realizes three applications:

- 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.
- 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.
- udp application. Users can link the eval board with another station, using 1025 port. Users can send information from station to board, 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.

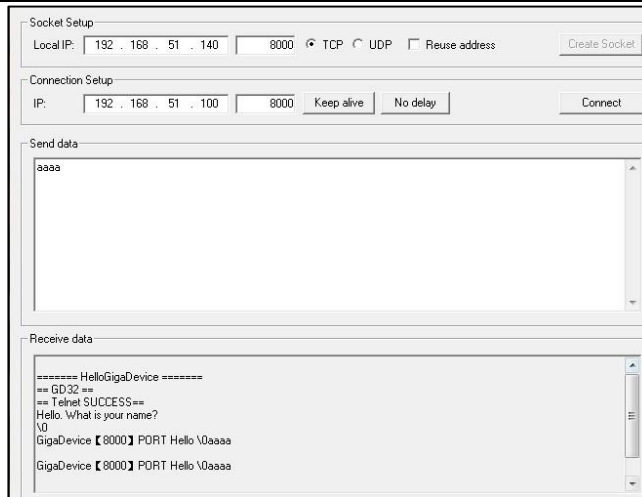
Note:

- Users should configure ip address, mask and gw of GD32H759I-EVAL-V1.1 board or served according to the actual net situation from the private defines in main.h.
- The USE\_ENET0 and USE\_ENET1 macros cannot be opened at the same time during use.

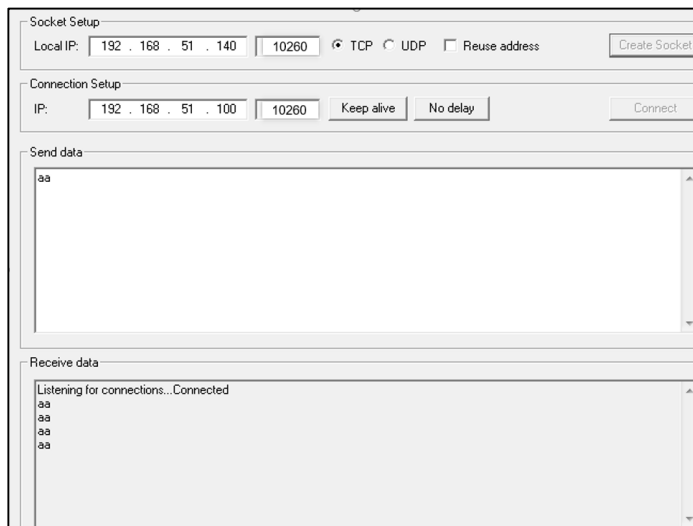
## DEMO running result

Download the program <FreeRTOS\_tcpudp> to the EVAL board, LED1 will light every 250ms.

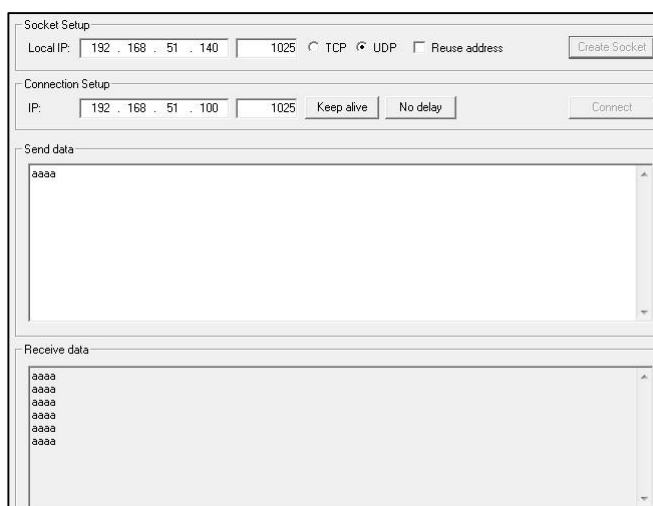
Using Network assistant software, configure the pc side to tcp client, using 8000 port, and when send something through the assistant, users can see the reply from the server:



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



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:



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.

### 5.26.2. 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 GD32H759I-EVAL-V1.1 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.

JP48, JP51, JP57, JP59, JP60, JP70 must be fitted. JP68, JP69 jump to Usart.

It is configured in RMII mode, and 25MHz oscillator is used, the system clock is configured to 600MHz.

This demo realizes three applications:

- 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.
- 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. 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.
- udp application. Users can link the eval board with another 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.

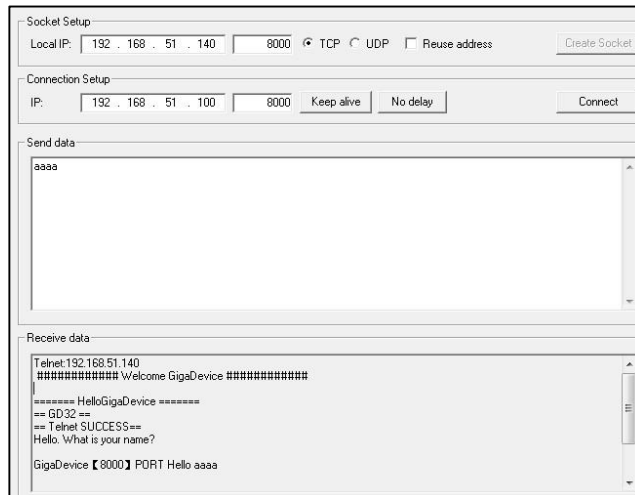
Note:

- Users should configure ip address, mask and gw of GD32H759I-EVAL-V1.1 board or served according to the actual net situation from the private defines in main.h.
- The USE\_ENET0 and USE\_ENET1 macros cannot be opened at the same time during use.

## DEMO running result

Download the program <Raw\_tcpudp> to the EVAL board.

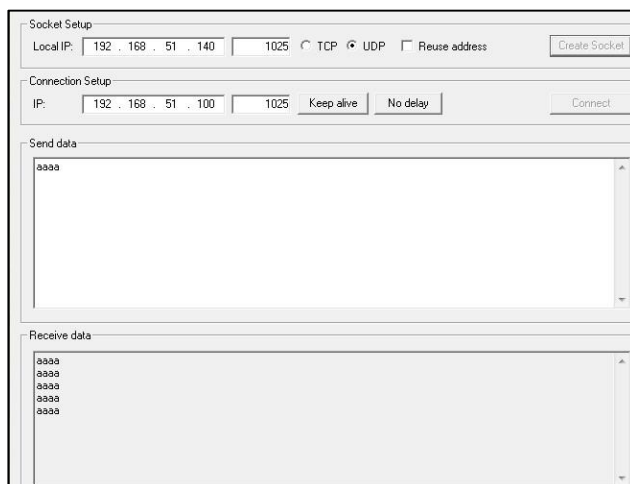
Using Network assistant software, configure the pc side to tcp client, using 8000 port, and when send something through the assistant, users can see the reply from the server:



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:



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:



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.

### 5.26.3. 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  $V_{REFINT}$  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 GD32H759I-EVAL-V1.1 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.

JP48, JP51, JP57, JP59, JP60, JP70 must be fitted. JP68, JP69 jump to Usart.

It is configured in RMII mode, and 25MHz oscillator is used, the system clock is configured to 600MHz.

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  $V_{REFINT}$  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.

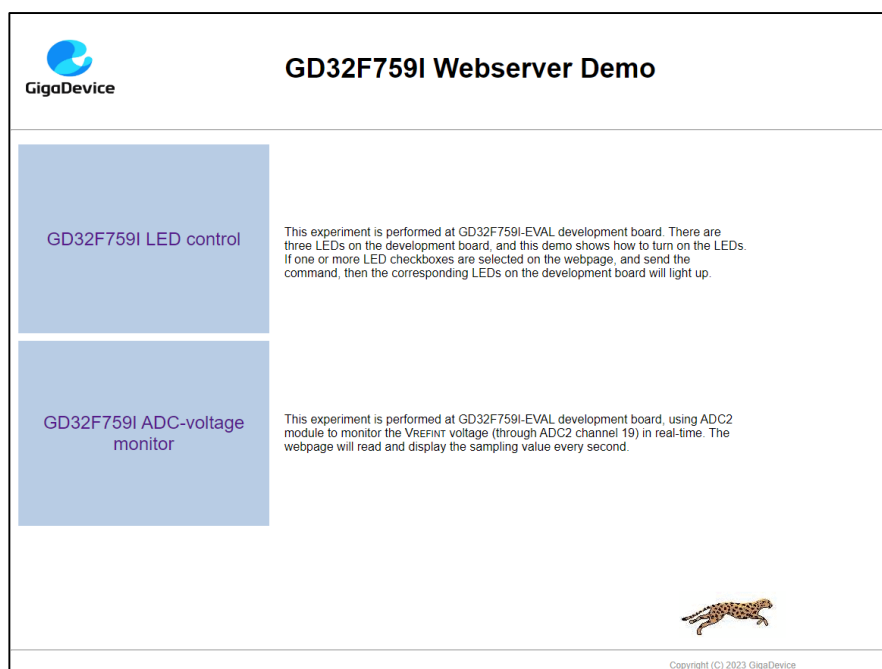
Note:

- Users should configure ip address, mask and gw of GD32H759I-EVAL-V1.1 board or served according to the actual net situation from the private defines in main.h.
- The USE\_ENET0 and USE\_ENET1 macros cannot be opened at the same time during use.

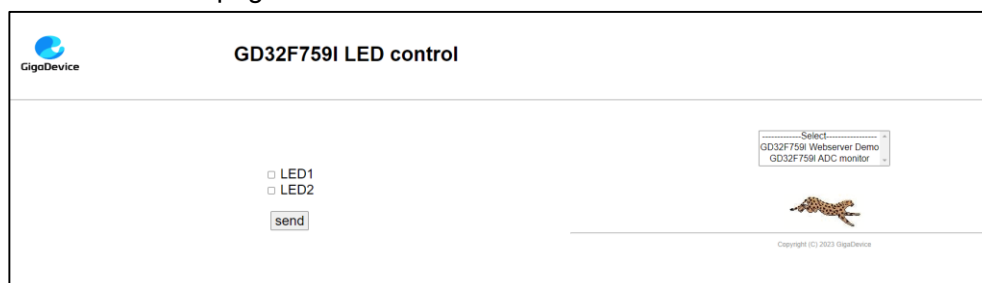
## DEMO running result

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  $V_{REFINT}$  voltage is showed on the webpage, and the data refreshes every second automatically.

The web home page shows as below:

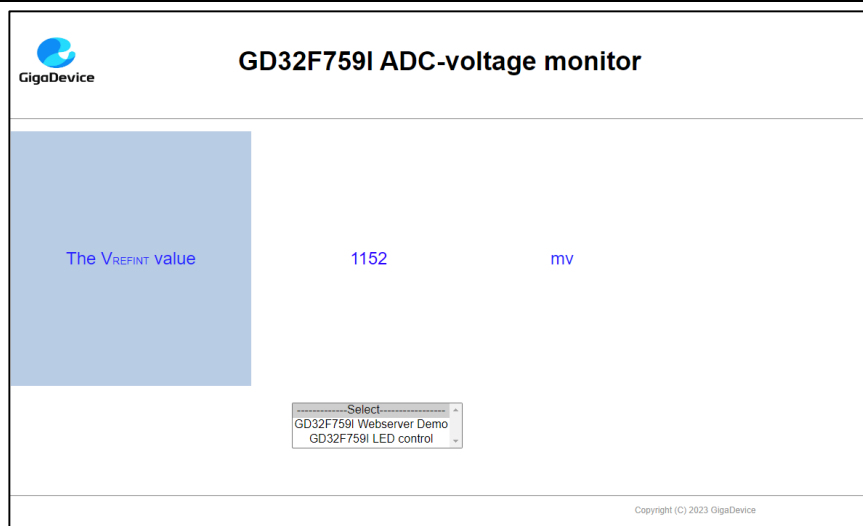


The LED control page shows as below:



The ADC monitor page shows as below:





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.

## 5.27. USB\_Device

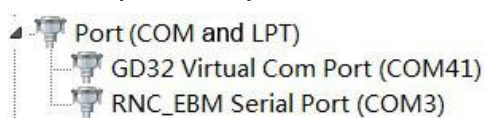
### 5.27.1. CDC\_ACM

#### DEMO purpose

This demo includes the following functions of GD32 MCU:

- Learn how to use the USBFS/HS peripheral
- Learn how to implement USB CDC device

GD32H759I-EVAL-V1.1 board has two USBFS/HS interface. In this demo, the GD32H759I-EVAL-V1.1 board USBHS0 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 < 27\_USB\_Device\CDC\_ACM > to the EVAL board and run. When you input message through computer keyboard, the HyperTerminal will receive and shown the message. For example, when input "GigaDevice MCU", the HyperTerminal will get and show it as below.

GigaDevice MCU

## 5.27.2. HID\_Keyboard

### DEMO Purpose

This demo includes the following functions of GD32 MCU:

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

GD32H759I-EVAL-V1.1 board has four keys, and two USBFS/HS interface. The four keys are Reset key, Wakeup key, User key and Tamper key. In this demo, the USBHS0 of GD32H759I-EVAL-V1.1 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

After doing this, download the program < 27\_USB\_Device\HID\_Keyboard > to the EVAL board and run. If you press the Wakeup key, will output 'a'. If you press the User key, will output 'c'. If you press the Tamper key, will output 'b'.

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

## 5.28. USB\_Host

### 5.28.1. USB HID Host

#### DEMO Purpose

This demo includes the following functions of GD32 MCU:

- Learn to use the USBHS 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

GD32H759I-EVAL-V1.1 board integrates USBFS module and the USBHS 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 USBHS1 as a USB HID host to communicate with external USB HID device.

#### DEMO Running Result

Download the program < 28\_USB\_Host\Host\_HID > to the 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 in the HyperTerminal.

```
> Low speed device detected.
> Device Attached.
VID: 046Dh
PID: C077h
> HID device connected.
Manufacturer: Logitech
Product: USB Optical Mouse
> Enumeration completed.
>To start the HID class operations:
>Press User Key...
> HID Demo Device : Mouse.

MoveLeft 7f units—*—MoveUp 34 units—*—No button is pressed.
MoveLeft 52 units—*—MoveDown 0 units—*—No button is pressed.
MoveLeft 1 units—*—MoveDown 0 units—*—No button is pressed.
MoveLeft 1 units—*—MoveDown 0 units—*—No button is pressed.
MoveLeft 1 units—*—MoveUp 2 units—*—No button is pressed.
```

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 HyperTerminal.

```

> Low speed device detected.
> Device Attached.
VID: 413Ch
PID: 2113h
> HID device connected.
Product: Dell KB216 Wired Keyboard
> Enumeration completed.
>To start the HID class operations:
>Press User Key...
> HID Demo Device : Keyboard.
> Use Keyboard to type characters:

The pressed button is o
The pressed button is o
The pressed button is p
The pressed button is =
The pressed button is 9> Device Disconnected.

```

## 5.28.2. MSC\_Host

### DEMO Purpose

This demo includes the following functions of GD32 MCU:

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

GD32H759I-EVAL-V1.1 board integrates USBFS module and the USBHS module, and the modules can be used as USB device, USB host or OTG device. This demo mainly shows how to use the USBHS1 as a USB MSC host to communicate with external Udisk.

### DEMO Running Result

Jump the JP68/69 to USART and JP70 to USB. Insert the OTG cable to USB port. Then, download the program < 28\_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 on the serial Assistant.

First the user 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.

```
++++USB host library started++++
> Reset the USB device.
> High speed device detected.
> Device Attached.
VID: FFFFh
PID: 5678h
> Mass storage device connected.
Manufacturer: USB
Product: Disk 2.0
Serial Number: 9207302211445624486
> Enumeration completed.
>To see the disk information:
> File System initialized.
> Disk capacity: 4026531328d Bytes.
> Exploring disk flash ...
>>> To see the root content of disk
>>> Press Tamper Key...
|__System Volume Information
|__GD32.TXT
|__RECYCLER
|__PORT.JPG
>>> Press Wakeup Key to write file
> Writing File to disk flash ...
> GD32.TXT created in the disk.
> The MSC host demo is end.
```

## 6. Revision history

Table 6-1. Revision history

Revision No.	Description	Date
1.0	Initial Release	Mar.31, 2023

## 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 under the intellectual property laws and treaties of the People's Republic of China and other jurisdictions worldwide. The Company reserves all rights under such laws and treaties and does not grant any license under its patents, copyrights, trademarks, or other intellectual property rights. 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.

The Company makes no warranty of any kind, express or implied, with regard to this document or any Product, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. The Company does not assume any liability arising out of the application or use of any Product described in this document. Any information provided in this document is provided only for reference purposes. It is the responsibility of the user of this document to properly design, program, and test the functionality and safety of any application made of this information and any resulting product. Except for customized products which has been expressly identified in the applicable agreement, the Products are designed, developed, and/or manufactured for ordinary business, industrial, personal, and/or household applications only. The Products are not designed, intended, or authorized for use as components in systems designed or intended for the operation of weapons, weapons systems, nuclear installations, atomic energy control instruments, combustion control instruments, airplane or spaceship instruments, transportation instruments, traffic signal instruments, life-support devices or systems, other medical devices or systems (including resuscitation equipment and surgical implants), pollution control or hazardous substances management, or other uses where the failure of the device or Product could cause personal injury, death, property or environmental damage ("Unintended Uses"). Customers shall take any and all actions to ensure using and selling the Products in accordance with the applicable laws and regulations. The Company is not liable, in whole or in part, and customers shall and hereby do 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 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.

Information in this document is provided solely in connection with the Products. The Company reserves the right to make changes, corrections, modifications or improvements to this document and Products and services described herein at any time, without notice.