

GigaDevice Semiconductor Inc.

GD32F527I-EVAL

Arm[®] Cortex[®]-M33 32-bit MCU

User Guide

Revision 1.2

(Aug. 2025)

Table of Contents

TABLE OF CONTENTS.....	1
LIST OF FIGURES	5
LIST OF TABLES	6
1. SUMMARY	7
2. FUNCTION PIN ASSIGNMENT	7
3. GETTING STARTED	11
4. HARDWARE LAYOUT OVERVIEW	12
4.1. Power supply	12
4.2. Boot option	12
4.3. LED	13
4.4. KEY	13
4.5. USART	14
4.6. ADC	14
4.7. DAC	14
4.8. I2S.....	15
4.9. I2C.....	15
4.10. SPI	16
4.11. CAN	16
4.12. ENET	17
4.13. SDIO	17
4.14. NAND flash	18
4.15. SDRAM	18
4.16. DCI.....	19
4.17. LCD	19
4.18. USBFS.....	20
4.19. USBHS	20
4.20. Extension.....	21
4.21. GD-Link.....	21
4.22. MCU.....	22
5. ROUTINE USE GUIDE	23
5.1. GPIO_Running_LED.....	23
5.1.1. DEMO purpose	23
5.1.2. DEMO running result	23
5.2. GPIO_Key_Polling_mode.....	23
5.2.1. DEMO purpose	23
5.2.2. DEMO running result	23
5.3. EXTI_Key_Interrupt_mode.....	24

5.3.1.	DEMO purpose	24
5.3.2.	DEMO running result	24
5.4.	USART_Printf.....	24
5.4.1.	DEMO purpose	24
5.4.2.	DEMO running result	24
5.5.	USART_Echo_Interrupt_mode.....	25
5.5.1.	DEMO purpose	25
5.5.2.	DEMO running result	25
5.6.	USART_DMA.....	25
5.6.1.	DEMO purpose	25
5.6.2.	DEMO running result	25
5.7.	ADC_Temperature_Vrefint.....	26
5.7.1.	DEMO purpose	26
5.7.2.	DEMO running result	26
5.8.	ADC0_ADC1_Follow_up_mode	27
5.8.1.	DEMO purpose	27
5.8.2.	DEMO running result	27
5.9.	ADC0_ADC1_Regular_Parallel_mode.....	28
5.9.1.	DEMO purpose	28
5.9.2.	DEMO running result	28
5.10.	DAC_Output_Voltage_Value	29
5.10.1.	DEMO purpose	29
5.10.2.	DEMO running result	29
5.11.	I2C_EEPROM.....	29
5.11.1.	DEMO purpose	29
5.11.2.	DEMO running result	29
5.12.	I2S_Audio_Player.....	30
5.12.1.	DEMO purpose	30
5.12.2.	DEMO running result	31
5.13.	SPI_Quad_Flash.....	31
5.13.1.	DEMO purpose	31
5.13.2.	DEMO running result	31
5.14.	EXMC_SDRAM.....	32
5.14.1.	DEMO purpose	32
5.14.2.	DEMO running result	32
5.15.	EXMC_SDRAM_DeepSleep.....	34
5.15.1.	DEMO purpose	34
5.15.2.	DEMO running result	34
5.16.	EXMC_NandFlash.....	35
5.16.1.	DEMO purpose	35
5.16.2.	DEMO running result	36
5.17.	SDIO_SDCardTest.....	36
5.17.1.	DEMO purpose	36
5.17.2.	DEMO running result	36

5.18. CAN_Network	37
5.18.1. DEMO purpose	37
5.18.2. DEMO running result	37
5.19. RCU_Clock_Out	38
5.19.1. DEMO purpose	38
5.19.2. DEMO running result	38
5.20. CTC_Calibration	38
5.20.1. DEMO purpose	38
5.20.2. DEMO running result	39
5.21. PMU_Sleep_Wakeup	39
5.21.1. DEMO purpose	39
5.21.2. DEMO running result	39
5.22. RTC_Calendar	39
5.22.1. DEMO purpose	39
5.22.2. DEMO running result	39
5.23. TIMER_Breath_LED	40
5.23.1. DEMO purpose	40
5.23.2. DEMO running result	40
5.24. TLI_IPA	40
5.24.1. DEMO purpose	40
5.24.2. DEMO running result	40
5.25. DCI_OV2640	41
5.25.1. DEMO purpose	41
5.25.2. DEMO running result	41
5.26. CAU	42
5.26.1. DEMO purpose	42
5.26.2. DEMO running result	42
5.27. HAU	44
5.27.1. DEMO purpose	44
5.27.2. DEMO running result	44
5.28. PKCAU	45
5.28.1. DEMO purpose	45
5.28.2. DEMO running result	46
5.29. TRNG_Get_Random	46
5.29.1. DEMO purpose	46
5.29.2. DEMO running result	46
5.30. ENET	46
5.30.1. FreeRTOS_tcpudp	46
5.30.2. Raw_tcpudp	49
5.30.3. Raw_webserver	51
5.31. USB_Device	53
5.31.1. HID_Keyboard	53
5.31.2. MSC_Udisk	54
5.32. USB_Host	56

5.32.1.	HID_Host	56
5.32.2.	MSC_Host.....	58
6.	REVISION HISTORY	60

List of Figures

Figure 4-1 Schematic diagram of power supply.....	12
Figure 4-2 Schematic diagram of boot option	12
Figure 4-3 Schematic diagram of LED function	13
Figure 4-4 Schematic diagram of Key function	13
Figure 4-5 Schematic diagram of USART0 function	14
Figure 4-6 Schematic diagram of ADC function.....	14
Figure 4-7 Schematic diagram of DAC function.....	14
Figure 4-8 Schematic diagram of I2S function	15
Figure 4-9 Schematic diagram of I2C function	15
Figure 4-10 Schematic diagram of SPI function.....	16
Figure 4-11 Schematic diagram of CAN function.....	16
Figure 4-12 Schematic diagram of Ethernet function	17
Figure 4-13 Schematic diagram of SDIO function.....	17
Figure 4-14 Schematic diagram of NAND flash function.....	18
Figure 4-15 Schematic diagram of SDRAM function	18
Figure 4-16 Schematic diagram of DCI function	19
Figure 4-17 Schematic diagram of LCD function	19
Figure 4-18 Schematic diagram of USBFS function	20
Figure 4-19 Schematic diagram of USBHS function	20
Figure 4-20 Schematic diagram of Extension Pin	21
Figure 4-21 Schematic diagram of GD-Link.....	21
Figure 4-22 Schematic diagram of MCU.....	22

List of Tables

Table 2-1 Function pin assignment	7
Table 6-1 Revision history	60

1. Summary

GD32F527I-EVAL uses GD32F527IST7 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, CTC, SDIO, DCI, ENET, USBFS, USBHS, GD-Link and Extension Pins are also included. For more details please refer to GD32F527I-EVAL-V1.0 schematic.

2. Function pin assignment

Table 2-1 Function pin assignment

Function	Pin	Description
LED	PF7	LED1
	PF8	LED2
	PE3	LED3
	PE2	LED4
RESET		K1-Reset
KEY	PA0	K2-Wakeup
	PC13	K3-Tamper
	PB14	K4-User key
USART0	PA9	USART0_TX
	PA10	USART0_RX
ADC	PC3	ADC012_IN13
DAC	PA4	DAC_OUT0
I2C	PB6	I2C0_SCL
	PB7	I2C0_SDA
SPI	PG10	SPI5_IO2
	PH4	SPI5_IO3
	PG13	SPI5_SCK
	PG14	SPI5_MOSI
	PG12	SPI5_MISO
	PI8	SPIFlash_CS
I2S	PC6	I2S_MCK
	PI1	I2S_CK
	PI0	I2S_WS
	PC1	I2S_SD
CAN	PI9	CAN0_RX
	PH13	CAN0_TX
NAND Flash	PD4	EXMC_NOE
	PD5	EXMC_NWE

		PD6	EXMC_NWAIT
		PD7	EXMC_NCE1
		PD11	EXMC_A16
		PD12	EXMC_A17
		PD14	EXMC_D0
		PD15	EXMC_D1
		PD0	EXMC_D2
		PD1	EXMC_D3
		PE7	EXMC_D4
		PE8	EXMC_D5
		PE9	EXMC_D6
		PE10	EXMC_D7
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
		PE15	EXMC_D12
		PD8	EXMC_D13
		PD9	EXMC_D14
		PD10	EXMC_D15
		PE0	EXMC_NBL0
		PE1	EXMC_NBL1
		PC5	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
SDIO		PD2	SDIO_CMD
		PC12	SDIO_CLK
		PC8	SDIO_DAT0
		PC9	SDIO_DAT1
		PC10	SDIO_DAT2
		PC11	SDIO_DAT3
DCI		PB6	DCI_I2C0_SCL
		PB7	DCI_I2C0_SDA
		PA4	DCI_HSYNC
		PG9	DCI_VSYNC
		PA6	DCI_PIXCLK
		PA8	DCI_XCLK
		PB9	DCI_D7
		PB8	DCI_D6
		PD3	DCI_D5
		PC11	DCI_D4
		PC9	DCI_D3
		PC8	DCI_D2
		PC7	DCI_D1
		PC6	DCI_D0
LCD		PI3	LCD_Touch_PENIRQ
		PF9	LCD_SPI4_MOSI
		PH7	LCD_SPI4_MISO
		PH6	LCD_SPI4_SCK
		PF6	LCD_SPI4_NSS
		PB15	LCD_PWM_BackLight
		PG3	LCD_Touch_Busy
		PF10	LCD_DE
		PH2	LCD_R0
		PH3	LCD_R1
		PH8	LCD_R2
		PH9	LCD_R3
		PH10	LCD_R4
		PH11	LCD_R5
		PH12	LCD_R6

		PG6	LCD_R7
		PE5	LCD_G0
		PE6	LCD_G1
		PH13	LCD_G2
		PH14	LCD_G3
		PH15	LCD_G4
		PI0	LCD_G5
		PI1	LCD_G6
		PI2	LCD_G7
		PE4	LCD_B0
		PG12	LCD_B1
		PG10	LCD_B2
		PG11	LCD_B3
		PI4	LCD_B4
		PI5	LCD_B5
		PI6	LCD_B6
		PI7	LCD_B7
		PG7	LCD_CLK
		PI10	LCD_HSYNC
		PI9	LCD_VSYNC
Ethernet		PA1	ETH_RMII_REF_CLK
		PA2	ETH_MDIO
		PA7	ETH_RMII_CRS_DV
		PG11	ETH_RMII_TX_EN
		PG13	ETH_RMII_TXD0
		PG14	ETH_RMII_TXD1
		PC1	ETH_MDC
		PC4	ETH_RMII_RXD0
		PC5	ETH_RMII_RXD1
USB_FS		PA9	USBFS_VBUS
		PA11	USBFS_DM
		PA12	USBFS_DP
USB_HS		PC3	USB_HS_ULPI_NXT
		PI11	USB_HS_ULPI_DIR
		PC0	USB_HS_ULPI_STP
		PA5	USB_HS_ULPI_CK
		PB5	USB_HS_ULPI_D7
		PB13	USB_HS_ULPI_D6
		PB12	USB_HS_ULPI_D5
		PB11	USB_HS_ULPI_D4
		PB10	USB_HS_ULPI_D3
		PB1	USB_HS_ULPI_D2
		PB0	USB_HS_ULPI_D1

	PA3	USB_HS_ULPI_D0
--	-----	----------------

3. Getting started

The EVAL board uses Mini USB connector or DC-005 connector to get power DC +5V, which is the hardware system normal work voltage. Three kinds of different USB power supply which are USB_FS, USB_HS_ULPI, GD-Link can be chosen through JP4. A J-Link tool or GD-Link on board is necessary in order to download and debug programs. Select the correct boot mode and then power on, the LEDPWR will turn on, which indicates that the power supply is OK.

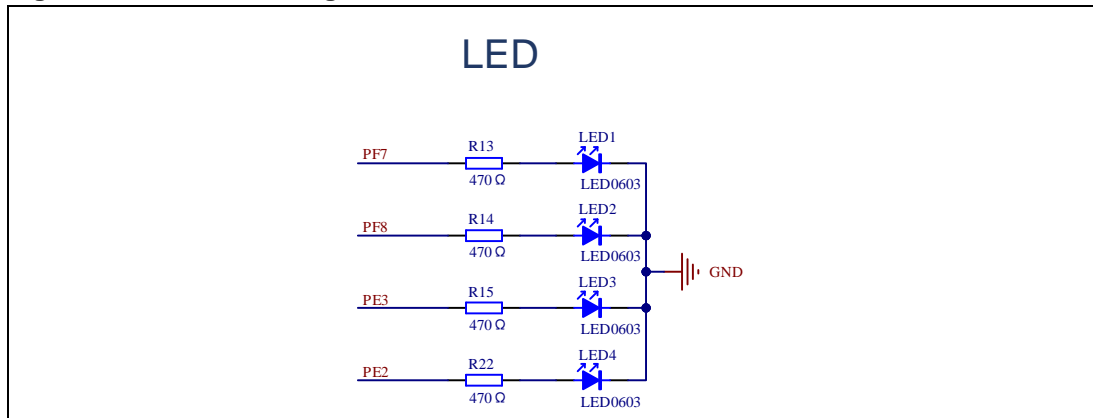
There are Keil version and IAR version of all projects. Keil version of the projects are created based on Keil MDK-ARM 5.28 uVision5. IAR version of the projects are created based on IAR Embedded Workbench for ARM 8.32.1. During use, the following points should be noted:

1. If you use Keil uVision5 to open the project. you need to install the latest version of GigaDevice.GigaDevice.GD32F527_DFP to load the related files.
2. If you use IAR to open the project, you need to install the latest version of IAR_GD32F527_ADDON to load the related files.

	Any	2-3	User memory	
	2-3	1-2	System memory	
	1-2	1-2	SRAM memory	

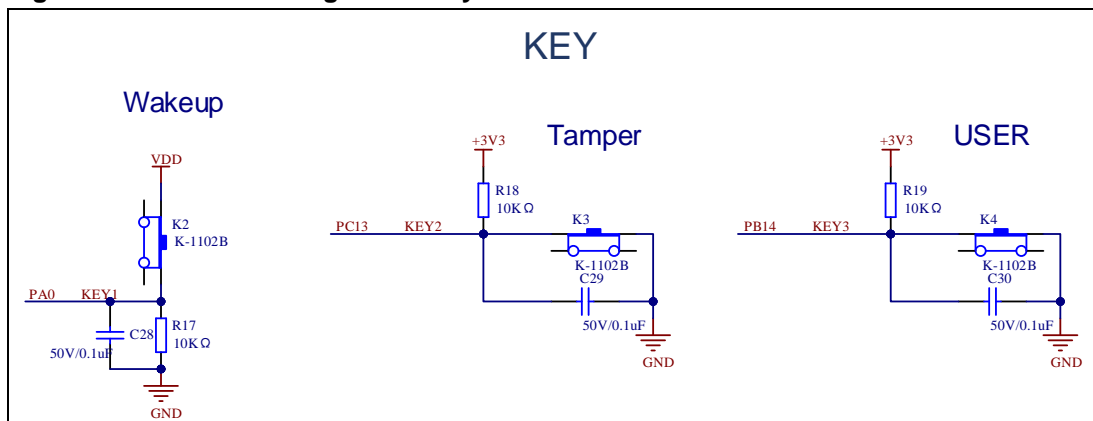
4.3. LED

Figure 4-3 Schematic diagram of LED function



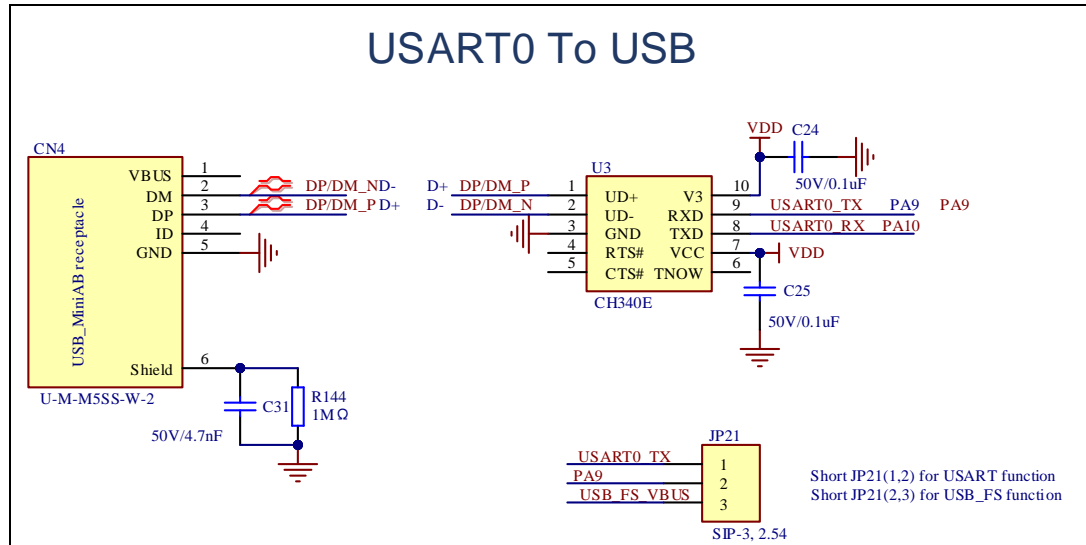
4.4. KEY

Figure 4-4 Schematic diagram of Key function



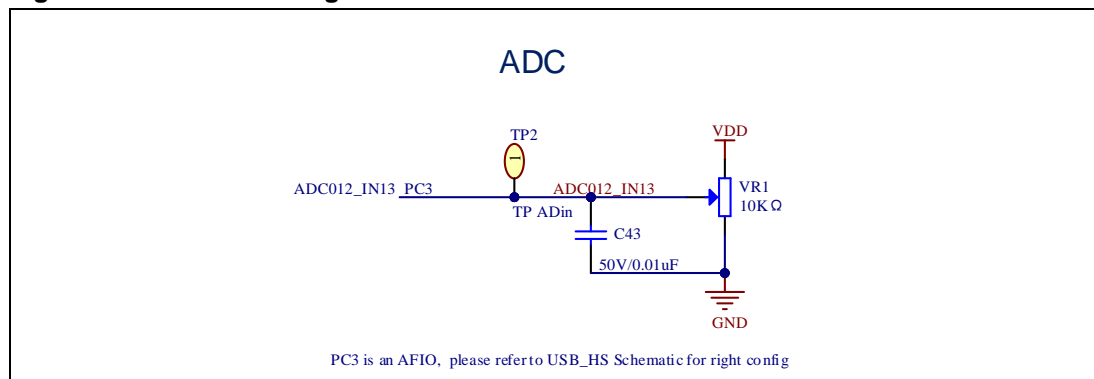
4.5. USART

Figure 4-5 Schematic diagram of USART0 function



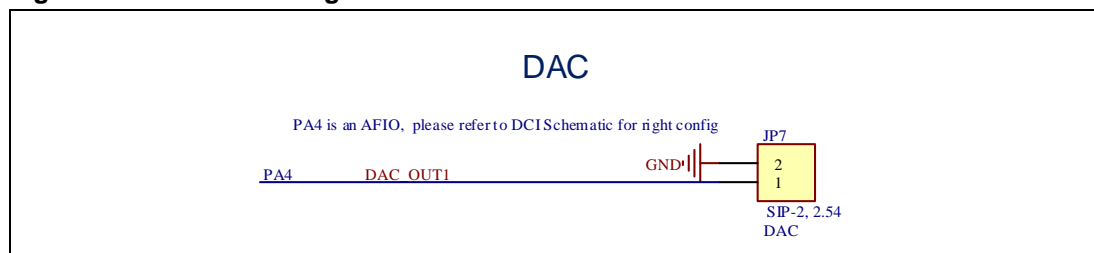
4.6. ADC

Figure 4-6 Schematic diagram of ADC function



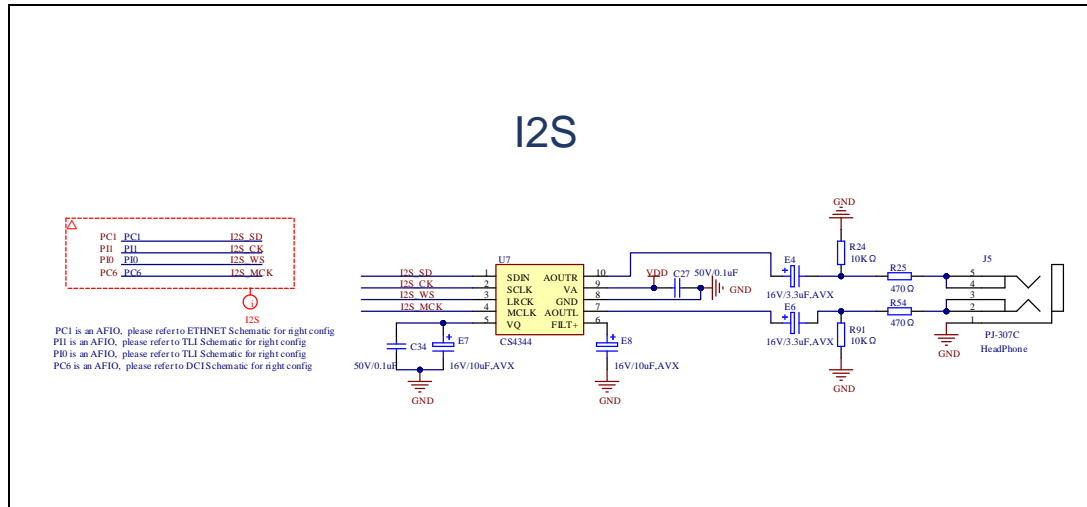
4.7. DAC

Figure 4-7 Schematic diagram of DAC function



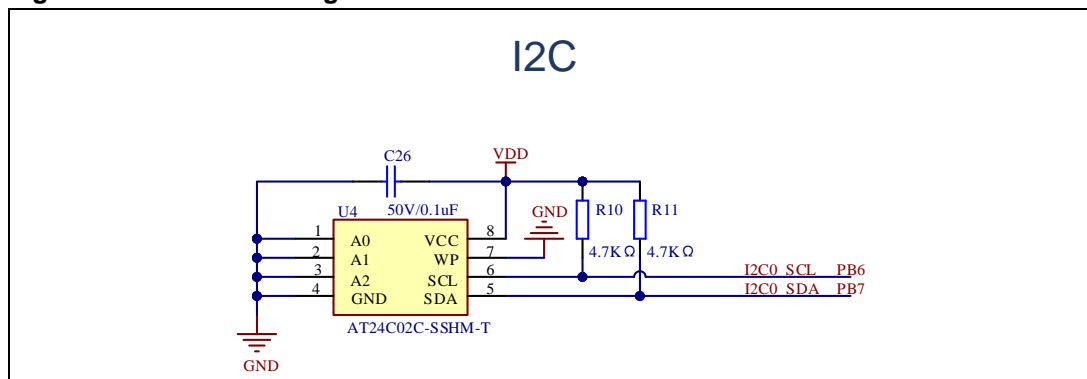
4.8. I2S

Figure 4-8 Schematic diagram of I2S function



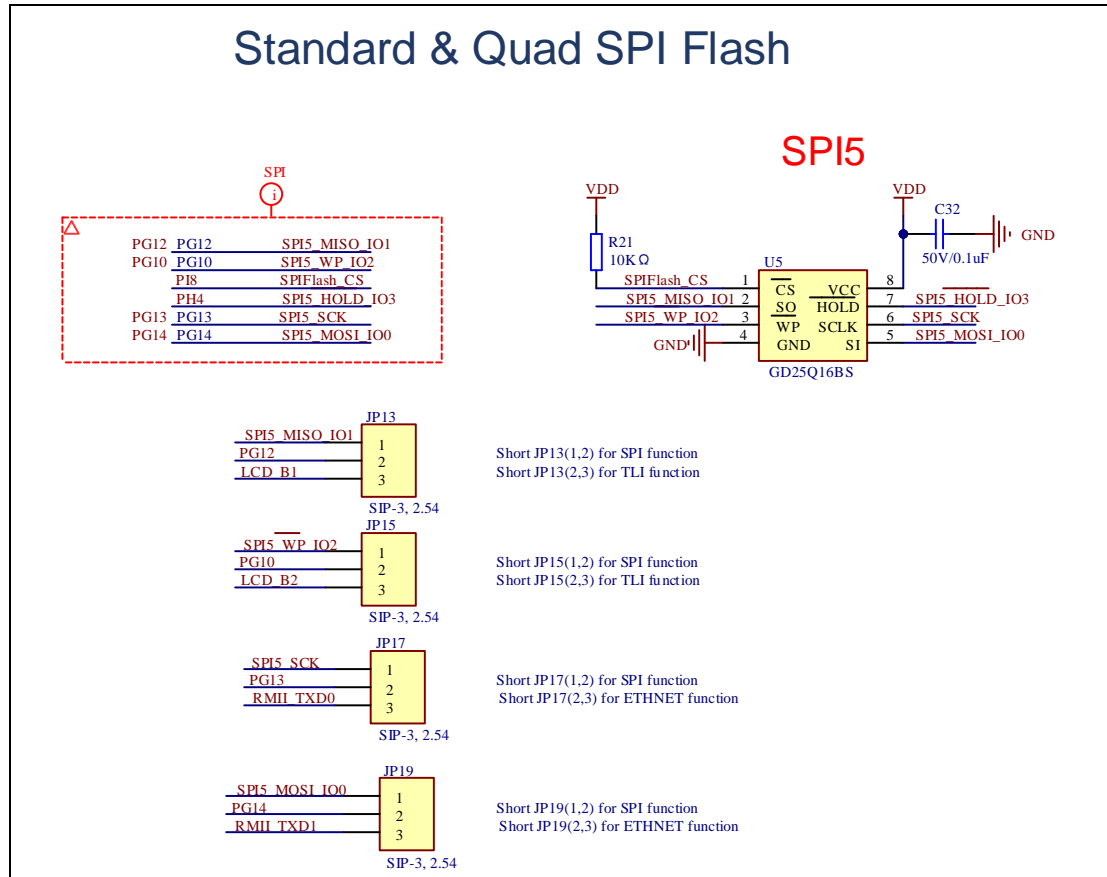
4.9. I2C

Figure 4-9 Schematic diagram of I2C function



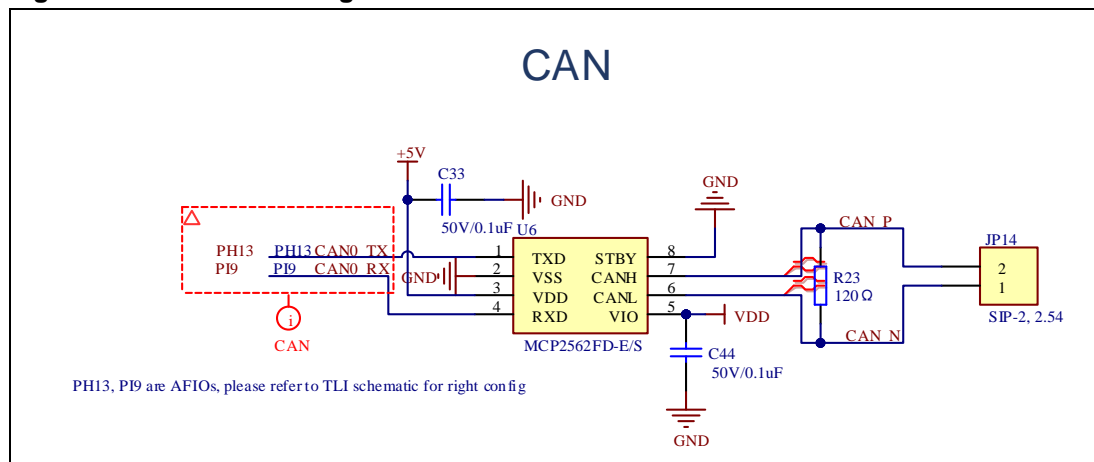
4.10. SPI

Figure 4-10 Schematic diagram of SPI function



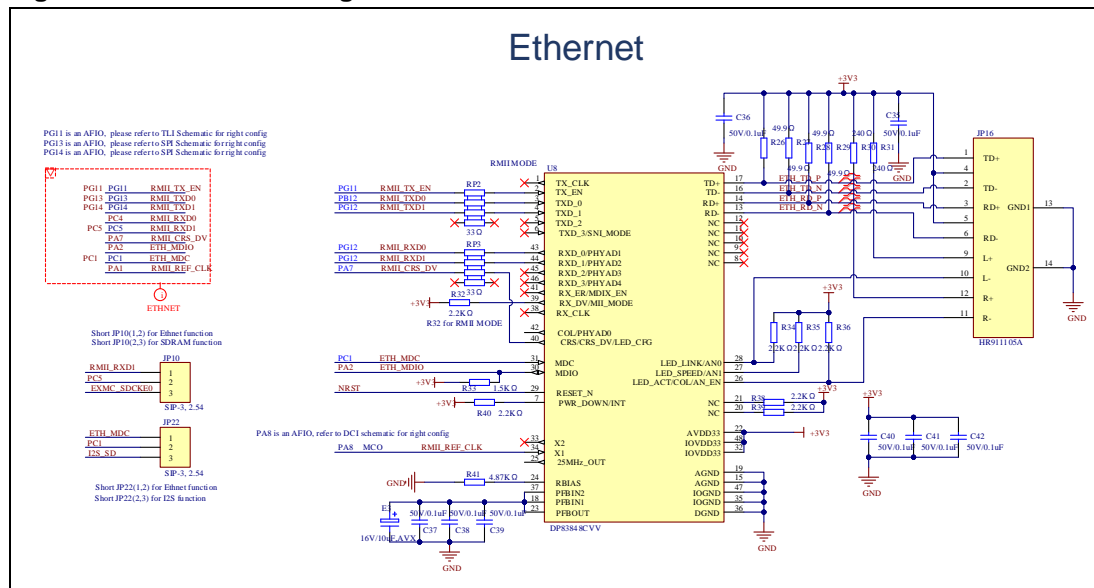
4.11. CAN

Figure 4-11 Schematic diagram of CAN function



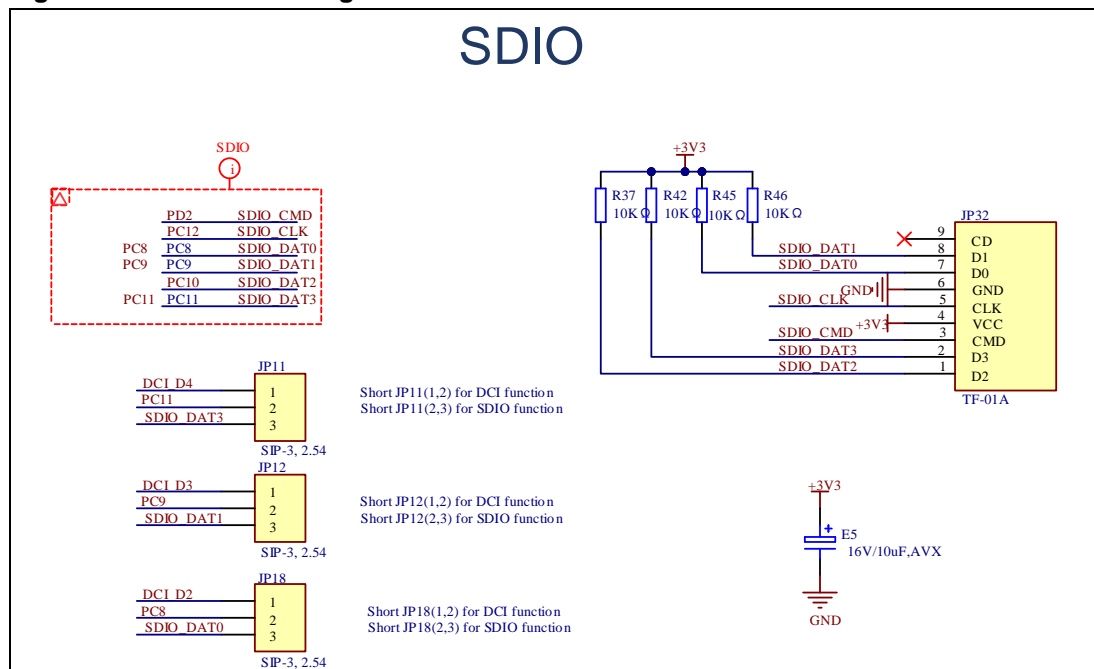
4.12. ENET

Figure 4-12 Schematic diagram of Ethernet function



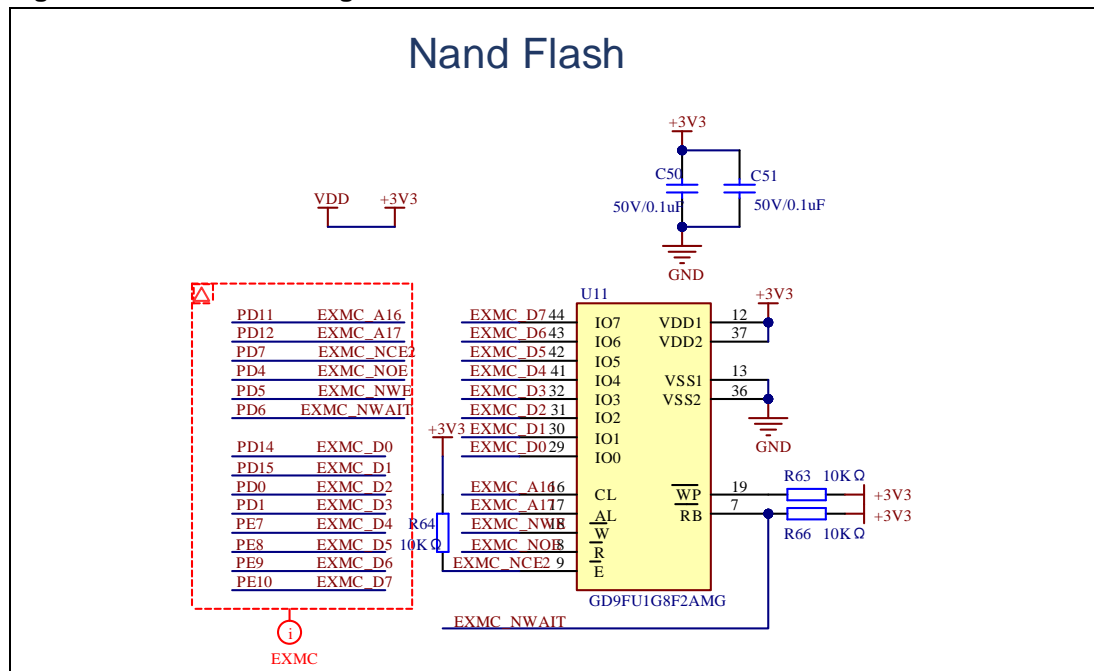
4.13. SDIO

Figure 4-13 Schematic diagram of SDIO function



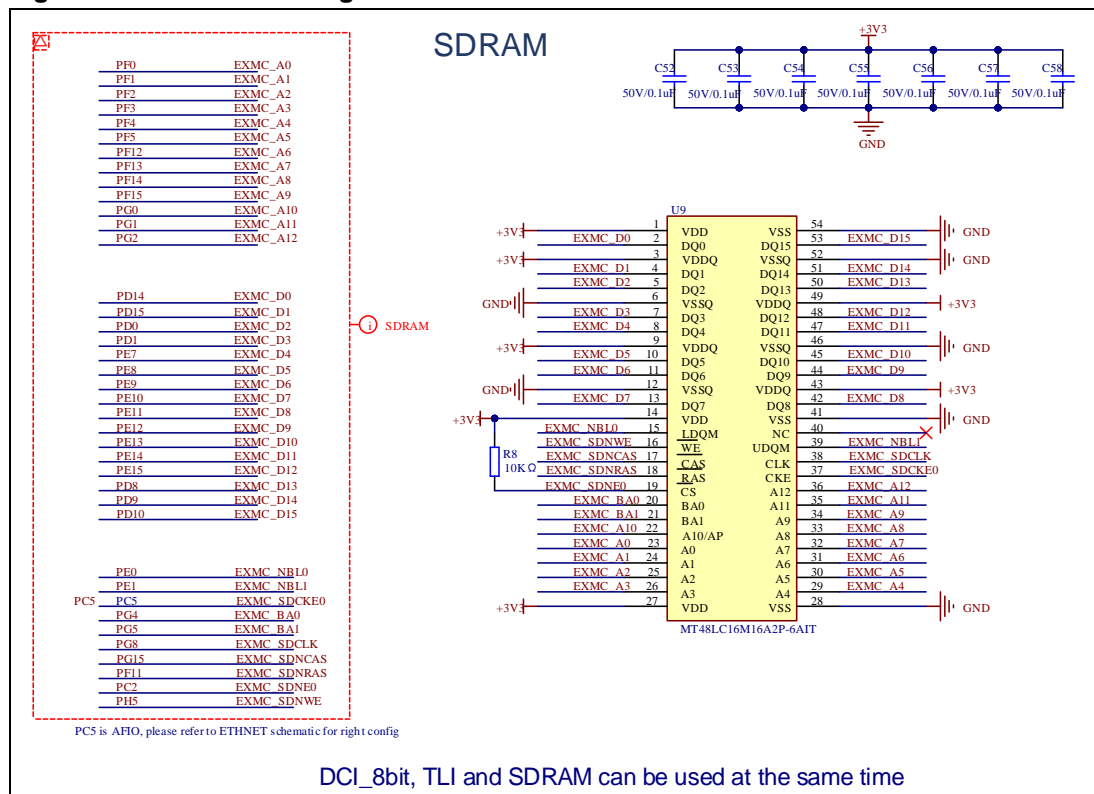
4.14. NAND flash

Figure 4-14 Schematic diagram of NAND flash function



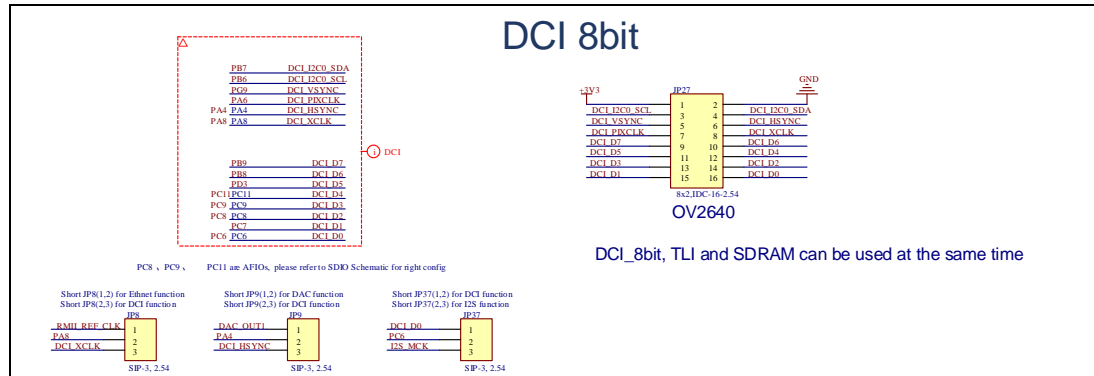
4.15. SDRAM

Figure 4-15 Schematic diagram of SDRAM function



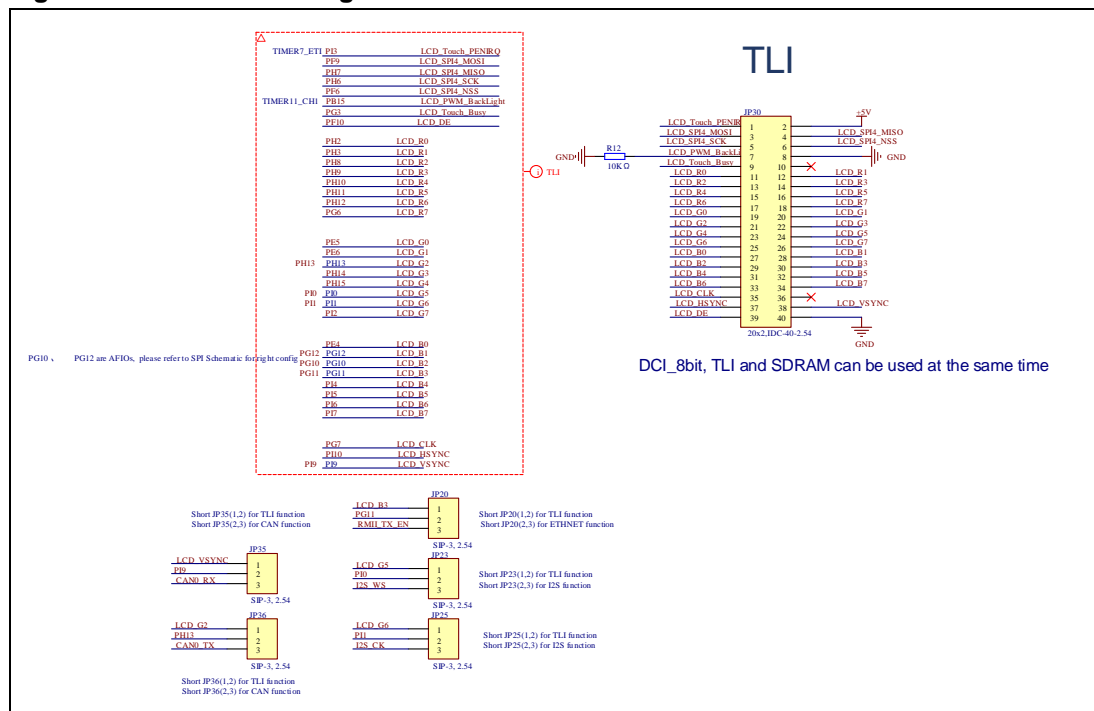
4.16. DCI

Figure 4-16 Schematic diagram of DCI function



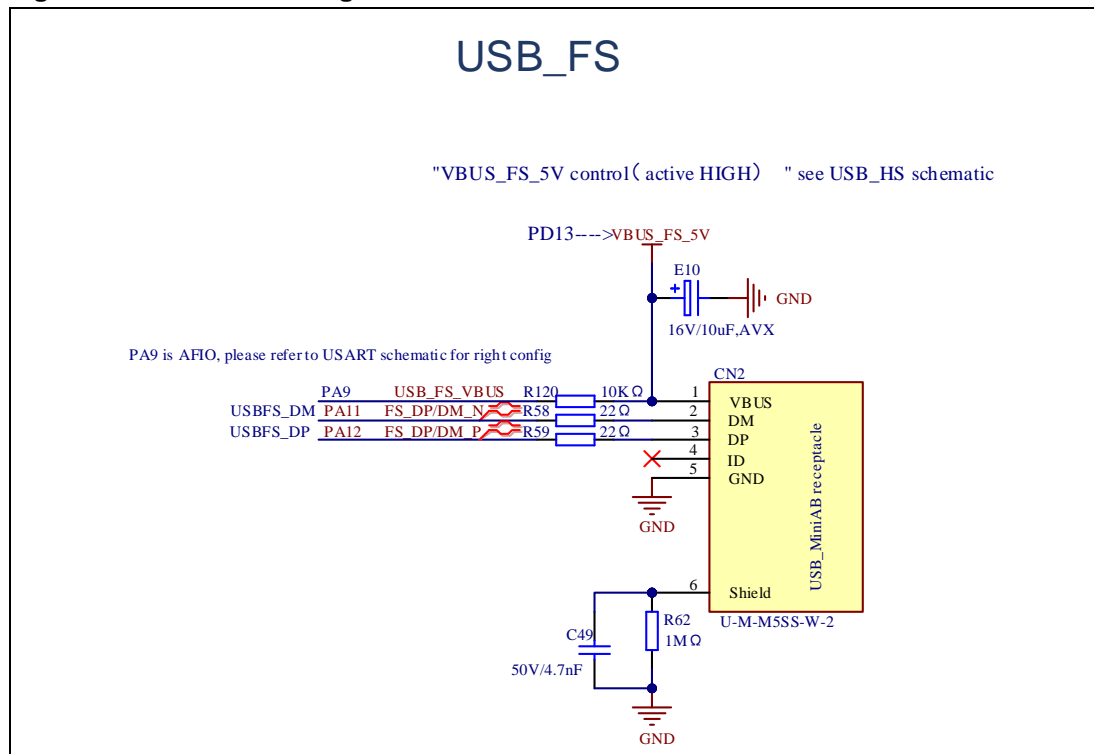
4.17. LCD

Figure 4-17 Schematic diagram of LCD function



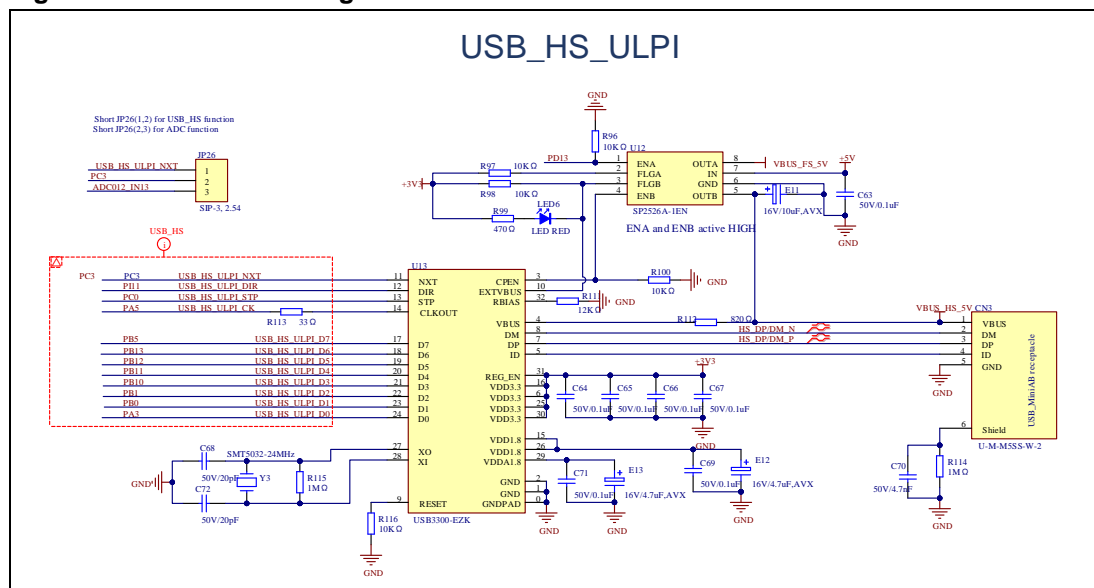
USBFS

Figure 4-18 Schematic diagram of USBFS function



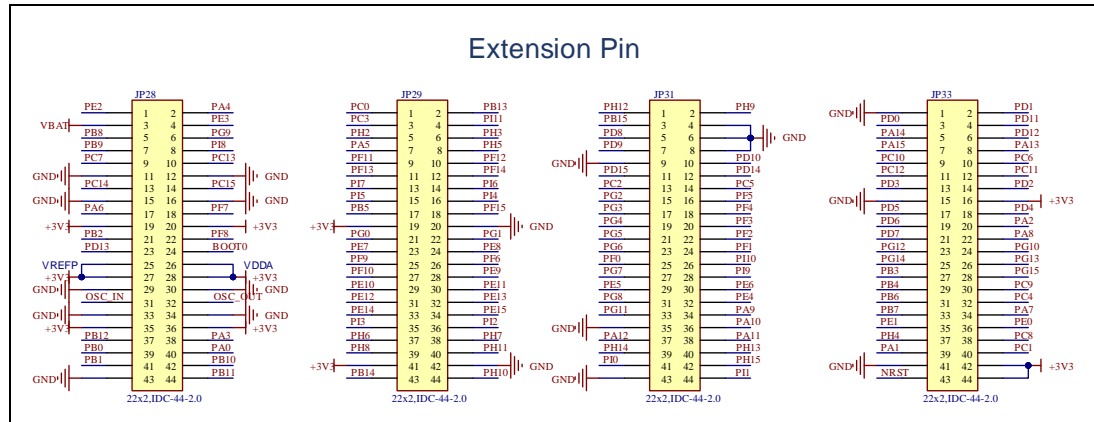
USBHS

Figure 4-19 Schematic diagram of USBHS function



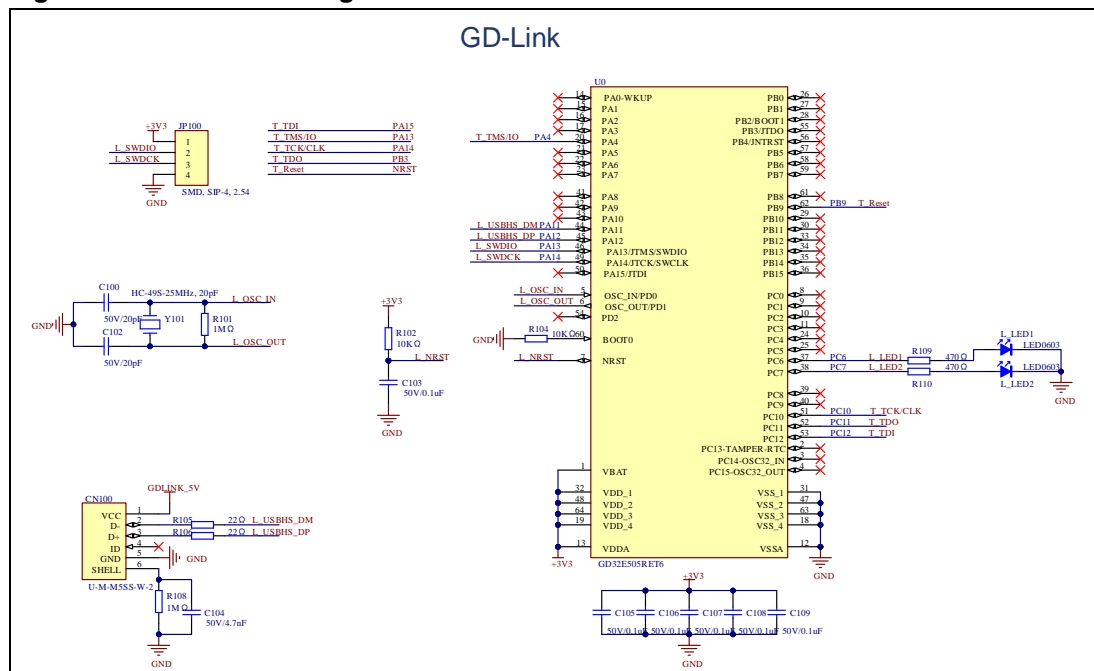
4.20. Extension

Figure 4-20 Schematic diagram of Extension Pin



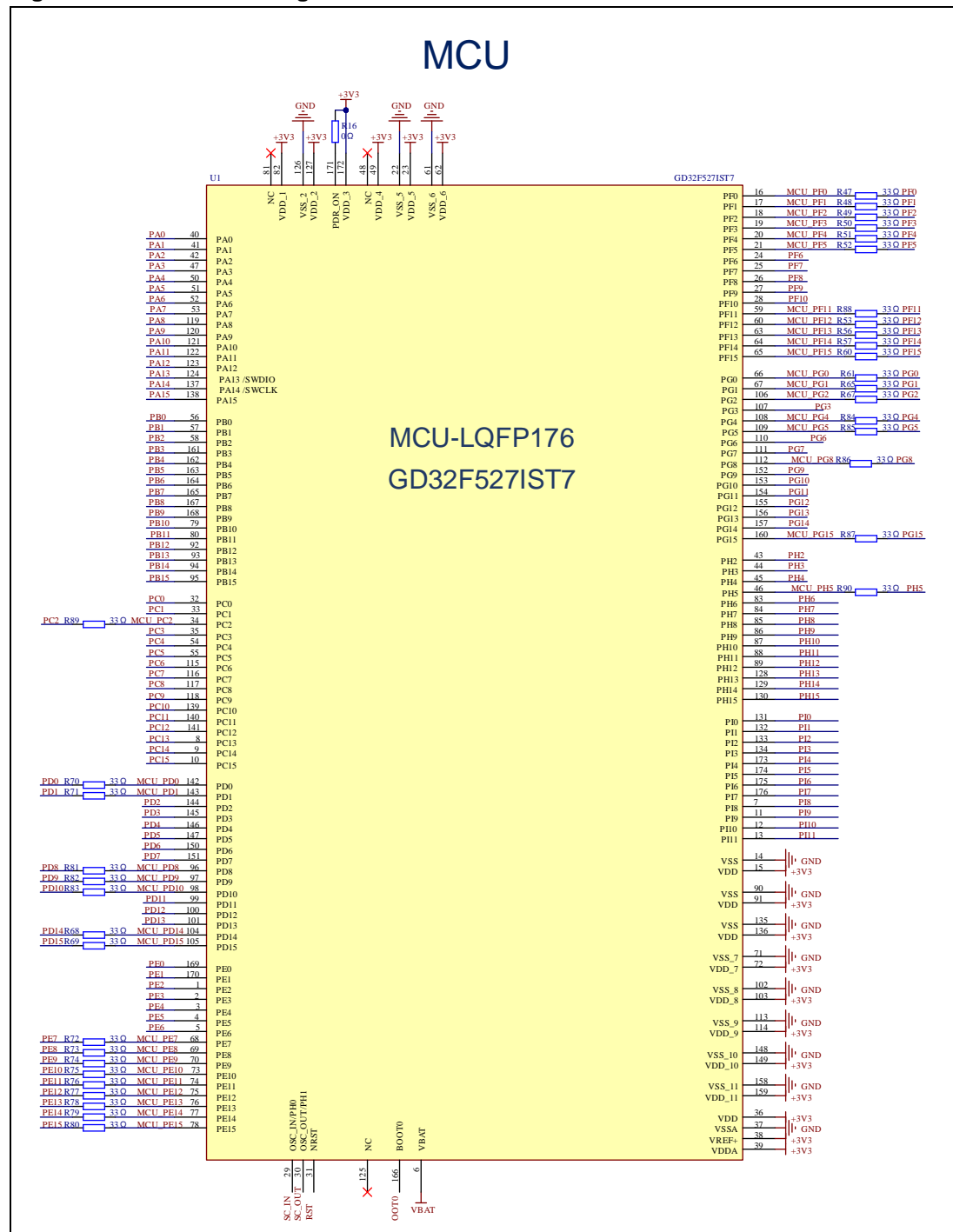
4.21. GD-Link

Figure 4-21 Schematic diagram of GD-Link



MCU

Figure 4-22 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.

GD32F527I-EVAL board has four LEDs. The LED1, LED2, LED3 and LED4 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 , LED3 and LED4 will turn on in sequence with interval of 1000ms, and 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.

GD32F527I-EVAL board has four keys and four LEDs. The four keys are Reset key, Tamper key, Wakeup key and User key. The LED1, LED2, LED3 and LED4 are controlled by GPIO.

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

5.2.2. DEMO running result

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

5.3. EXTI_Key_Interrupt_mode

5.3.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

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

GD32F527I-EVAL board has four keys and four LEDs. The four keys are Reset key, Tamper key, Wakeup key and User key. The LED1, LED2, LED3 and LED4 are controlled by GPIO.

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

5.3.2. DEMO running result

Download the program <03_EXTI_Key_Interrupt_mode> to the EVAL board, LED2 is turned on and off for test. 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 COM0 and jump JP21 to USART. This implementation outputs "USART printf example: please press the Tamper key" on the HyperTerminal using COM0. Press the Tamper key, the LED1 will be turned on and serial port will output "USART printf example".

The output information via the serial port is as following.

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

5.5. USART_Echo_Interrupt_mode

5.5.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

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

5.5.2. DEMO running result

Download the program < 05_USART_Echo_Interrupt_mode > to the EVAL board, connect serial cable to COM0 and jump JP21 to USART. Firstly, all the LEDs are turned on and off for test. Then, the COM0 sends the tx_buffer array (from 0x00 to 0xFF) to the serial terminal tool supporting hex format communication and waits for receiving data of BUFFER_SIZE bytes from the serial terminal. The data MCU has received is stored in the rx_buffer array. After that, compare tx_buffer with rx_buffer. If tx_buffer is same with rx_buffer, LED1, LED2, LED3, LED4 flash by turns. Otherwise, LED1, LED2, LED3, LED4 toggle together.

The output information via the serial port is as following.

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

5.6. USART_DMA

5.6.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

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

5.6.2. DEMO running result

Download the program < 06_USART_DMA > to the EVAL board, connect serial cable to COM0 and jump JP21 to USART. Firstly, all the LEDs are turned on and off for test. Then, the 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, LED1, LED2, LED3, LED4 flash by turns. Otherwise, LED1, LED2, LED3, LED4 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	AO	A1	A2	A3	A4	A5	A6	A7
A8	A9	AA	AB	AC	AD	AE	AF	BO	B1	B2	B3	B4	B5	B6	B7	B8	B9	BA	BB	BC	BD	BE	BF	CO	C1	C2	C3
C4	C5	C6	C7	C8	C9	CA	CB	CC	CD	CE	CF	DO	D1	D2	D3	D4	D5	D6	D7	D8	D9	DA	DB	DC	DD	DE	DF
EO	E1	E2	E3	E4	E5	E6	E7	E8	E9	EA	EB	EC	ED	EE	EF	FO	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), channel 17 (VREFINT channel) and channel 18(VBAT/4 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_Vbat> to the board. Connect serial cable to COM0, open the HyperTerminal.

When the program is running, HyperTerminal display the value of temperature, internal voltage reference (VREFINT) and external battery voltage VBAT.

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 24 degrees Celsius
the reference voltage data is 1.198V
the battery voltage is 3.213V

the temperature data is 25 degrees Celsius
the reference voltage data is 1.201V
the battery voltage is 3.213V

the temperature data is 25 degrees Celsius
the reference voltage data is 1.199V
the battery voltage is 3.203V

the temperature data is 25 degrees Celsius
the reference voltage data is 1.198V
the battery voltage is 3.213V
```

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 USART with the jumper cap, and then download the program <08_ADC0_ADC1_Follow_up_mode> to the board. Connect serial cable to COM0, open the HyperTerminal. PC5 pin connect to the external voltage input. PC3 is the output voltage of the slide rheostat VR1 on board. Keep PC5 pin should not be reused by other peripherals. JP17 should not be connected.

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 sampling the first channel of ADCx (x=0,1), the value of the ADC0 conversion of PC3 pin is stored into the low half word of `adc_value [0]`, 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 sampling the second channel of ADCx (x=0,1), the value of the ADC0 conversion of PC5 pin is stored into the low half word of `adc_value [1]`, 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 00DE0FF3
the data adc_value[1] is 0FFF00A3

the data adc_value[0] is 00E30FFE
the data adc_value[1] is 0FFF00A4

the data adc_value[0] is 00EA0FF9
the data adc_value[1] is 0FF400B2

the data adc_value[0] is 00DE0FFF
the data adc_value[1] is 0FFE00A9

the data adc_value[0] is 00E00FF1
the data adc_value[1] is 0FF500A6
```

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 USART with the jumper cap, and then download the program <09_ADC0_ADC1_Regular_Parallel_mode> to the board. Connect serial cable to COM0, open the HyperTerminal. PC5 pin connect to the external voltage input. PC3 is the output voltage of the slide rheostat VR1 on board. Keep PC5 pin should not be reused by other peripherals. JP17 should not be connected.

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 sampling the first channel of ADCx (x=0,1), the value of the ADC0 conversion of PC3 pin is stored into the low half word of `adc_value [0]`, the value of the ADC1 conversion of PC5 pin is stored into the high half word of `adc_value [0]`. When sampling the second channel of ADCx (x=0,1), the value of the ADC0 conversion of PC5 pin is stored into the low half word of `adc_value [1]`, the value of the ADC1 conversion of PC3 pin is stored into the high half word of `adc_value [1]`.

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

```
the data adc_value[0] is 06210000
the data adc_value[1] is 00000627

the data adc_value[0] is 06290829
the data adc_value[1] is 0B40061F

the data adc_value[0] is 06250B49
the data adc_value[1] is 0B590629

the data adc_value[0] is 06280B3F
the data adc_value[1] is 0B320628

the data adc_value[0] is 06230B30
the data adc_value[1] is 0B430622
```

5.10. DAC_Output_Voltage_Value

5.10.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

- Learn to use DAC to output voltage on DAC0_OUT0 output.

5.10.2. DEMO running result

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

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

The voltage on PA4 can be observed through the oscilloscope.

5.11. 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 USART0, jump JP21 to USART0, then open the HyperTerminal to show the print message.

Firstly, the data of 256 bytes will be written to the EEPROM from the address 0x00 and printed by the serial port. Then, reading the EEPROM from address 0x00 for 256 bytes and the result will be printed. Finally, compare the data that were written to the EEPROM and the data that were read from the EEPROM. If they are the same, the serial port will output "I2C-AT24C02 test passed!" and the three LEDs lights flashing, otherwise the serial port will output "Err: data read and write aren't matching." And all the three LEDs light.

The output information via the serial port is as following.

```
I2C-24C02 configured...

The I2C is hardware interface
The speed is 400K
AT24C02 writing...
0x00 0x01 0x02 0x03 0x04 0x05 0x06 0x07 0x08 0x09 0x0A 0x0B 0x0C 0x0D 0x0E 0x0F
0x10 0x11 0x12 0x13 0x14 0x15 0x16 0x17 0x18 0x19 0x1A 0x1B 0x1C 0x1D 0x1E 0x1F
0x20 0x21 0x22 0x23 0x24 0x25 0x26 0x27 0x28 0x29 0x2A 0x2B 0x2C 0x2D 0x2E 0x2F
0x30 0x31 0x32 0x33 0x34 0x35 0x36 0x37 0x38 0x39 0x3A 0x3B 0x3C 0x3D 0x3E 0x3F
0x40 0x41 0x42 0x43 0x44 0x45 0x46 0x47 0x48 0x49 0x4A 0x4B 0x4C 0x4D 0x4E 0x4F
0x50 0x51 0x52 0x53 0x54 0x55 0x56 0x57 0x58 0x59 0x5A 0x5B 0x5C 0x5D 0x5E 0x5F
0x60 0x61 0x62 0x63 0x64 0x65 0x66 0x67 0x68 0x69 0x6A 0x6B 0x6C 0x6D 0x6E 0x6F
0x70 0x71 0x72 0x73 0x74 0x75 0x76 0x77 0x78 0x79 0x7A 0x7B 0x7C 0x7D 0x7E 0x7F
0x80 0x81 0x82 0x83 0x84 0x85 0x86 0x87 0x88 0x89 0x8A 0x8B 0x8C 0x8D 0x8E 0x8F
0x90 0x91 0x92 0x93 0x94 0x95 0x96 0x97 0x98 0x99 0x9A 0x9B 0x9C 0x9D 0x9E 0x9F
0xA0 0xA1 0xA2 0xA3 0xA4 0xA5 0xA6 0xA7 0xA8 0xA9 0xAA 0xAB 0xAC 0xAD 0xAE 0xAF
0xB0 0xB1 0xB2 0xB3 0xB4 0xB5 0xB6 0xB7 0xB8 0xB9 0xBA 0xBB 0xBC 0xBD 0xBE 0xBF
0xC0 0xC1 0xC2 0xC3 0xC4 0xC5 0xC6 0xC7 0xC8 0xC9 0xCA 0xCB 0xCC 0xCD 0xCE 0xCF
0xD0 0xD1 0xD2 0xD3 0xD4 0xD5 0xD6 0xD7 0xD8 0xD9 0xDA 0xDB 0xDC 0xDD 0xDE 0xDF
0xE0 0xE1 0xE2 0xE3 0xE4 0xE5 0xE6 0xE7 0xE8 0xE9 0xEA 0xEB 0xEC 0xED 0xEE 0xEF
0xF0 0xF1 0xF2 0xF3 0xF4 0xF5 0xF6 0xF7 0xF8 0xF9 0xFA 0xFB 0xFC 0xFD 0xFE 0xFF
AT24C02 reading...
0x00 0x01 0x02 0x03 0x04 0x05 0x06 0x07 0x08 0x09 0x0A 0x0B 0x0C 0x0D 0x0E 0x0F
0x10 0x11 0x12 0x13 0x14 0x15 0x16 0x17 0x18 0x19 0x1A 0x1B 0x1C 0x1D 0x1E 0x1F
0x20 0x21 0x22 0x23 0x24 0x25 0x26 0x27 0x28 0x29 0x2A 0x2B 0x2C 0x2D 0x2E 0x2F
0x30 0x31 0x32 0x33 0x34 0x35 0x36 0x37 0x38 0x39 0x3A 0x3B 0x3C 0x3D 0x3E 0x3F
0x40 0x41 0x42 0x43 0x44 0x45 0x46 0x47 0x48 0x49 0x4A 0x4B 0x4C 0x4D 0x4E 0x4F
0x50 0x51 0x52 0x53 0x54 0x55 0x56 0x57 0x58 0x59 0x5A 0x5B 0x5C 0x5D 0x5E 0x5F
0x60 0x61 0x62 0x63 0x64 0x65 0x66 0x67 0x68 0x69 0x6A 0x6B 0x6C 0x6D 0x6E 0x6F
0x70 0x71 0x72 0x73 0x74 0x75 0x76 0x77 0x78 0x79 0x7A 0x7B 0x7C 0x7D 0x7E 0x7F
0x80 0x81 0x82 0x83 0x84 0x85 0x86 0x87 0x88 0x89 0x8A 0x8B 0x8C 0x8D 0x8E 0x8F
0x90 0x91 0x92 0x93 0x94 0x95 0x96 0x97 0x98 0x99 0x9A 0x9B 0x9C 0x9D 0x9E 0x9F
0xA0 0xA1 0xA2 0xA3 0xA4 0xA5 0xA6 0xA7 0xA8 0xA9 0xAA 0xAB 0xAC 0xAD 0xAE 0xAF
0xB0 0xB1 0xB2 0xB3 0xB4 0xB5 0xB6 0xB7 0xB8 0xB9 0xBA 0xBB 0xBC 0xBD 0xBE 0xBF
0xC0 0xC1 0xC2 0xC3 0xC4 0xC5 0xC6 0xC7 0xC8 0xC9 0xCA 0xCB 0xCC 0xCD 0xCE 0xCF
0xD0 0xD1 0xD2 0xD3 0xD4 0xD5 0xD6 0xD7 0xD8 0xD9 0xDA 0xDB 0xDC 0xDD 0xDE 0xDF
0xE0 0xE1 0xE2 0xE3 0xE4 0xE5 0xE6 0xE7 0xE8 0xE9 0xEA 0xEB 0xEC 0xED 0xEE 0xEF
0xF0 0xF1 0xF2 0xF3 0xF4 0xF5 0xF6 0xF7 0xF8 0xF9 0xFA 0xFB 0xFC 0xFD 0xFE 0xFF
I2C-AT24C02 test passed!
```

5.12. I2S_Audio_Player

5.12.1. 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

GD32F527I-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.

5.12.2. DEMO running result

Download the program<12_I2S_Audio_Player>to the EVAL board, jump the JP22, JP23, JP25, JP37 to I2S with the jumper cap, insert the headphone into the audio port, and then listen to the audio file.

5.13. SPI_Quad_Flash

5.13.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

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

GD32F527I-EVAL board integrates SPI5 module with Quad-SPI mode and the mode can communicate with external NOR Flash devices. The SPI NOR FLASH is a serial FLASH memory chip GD25Q16B which size is 16Mbit, the chip supports standard SPI and quad SPI operation instructions.

5.13.2. 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 JP21 to USART, and jump the JP13, JP15, JP17, JP19 to SPI.

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


```
#####
GD32F527I-EVAL System is Starting up...
GD32F527I-EVAL SystemCoreClock:200000000Hz
GD32F527I-EVAL The CPU Unique Device ID:[32303345-2384418-2301429]
GD32F527I-EVAL SPI Flash:GD25Q16 configured...
The Flash_ID:0xC84015
Write to tx_buffer:
0x00 0x01 0x02 0x03 0x04 0x05 0x06 0x07 0x08 0x09 0x0A 0x0B 0x0C 0x0D 0x0E 0x0F
0x10 0x11 0x12 0x13 0x14 0x15 0x16 0x17 0x18 0x19 0x1A 0x1B 0x1C 0x1D 0x1E 0x1F
0x20 0x21 0x22 0x23 0x24 0x25 0x26 0x27 0x28 0x29 0x2A 0x2B 0x2C 0x2D 0x2E 0x2F
0x30 0x31 0x32 0x33 0x34 0x35 0x36 0x37 0x38 0x39 0x3A 0x3B 0x3C 0x3D 0x3E 0x3F
0x40 0x41 0x42 0x43 0x44 0x45 0x46 0x47 0x48 0x49 0x4A 0x4B 0x4C 0x4D 0x4E 0x4F
0x50 0x51 0x52 0x53 0x54 0x55 0x56 0x57 0x58 0x59 0x5A 0x5B 0x5C 0x5D 0x5E 0x5F
0x60 0x61 0x62 0x63 0x64 0x65 0x66 0x67 0x68 0x69 0x6A 0x6B 0x6C 0x6D 0x6E 0x6F
0x70 0x71 0x72 0x73 0x74 0x75 0x76 0x77 0x78 0x79 0x7A 0x7B 0x7C 0x7D 0x7E 0x7F
0x80 0x81 0x82 0x83 0x84 0x85 0x86 0x87 0x88 0x89 0x8A 0x8B 0x8C 0x8D 0x8E 0x8F
0x90 0x91 0x92 0x93 0x94 0x95 0x96 0x97 0x98 0x99 0x9A 0x9B 0x9C 0x9D 0x9E 0x9F
0xA0 0xA1 0xA2 0xA3 0xA4 0xA5 0xA6 0xA7 0xA8 0xA9 0xAA 0xAB 0xAC 0xAD 0xAE 0xAF
0xB0 0xB1 0xB2 0xB3 0xB4 0xB5 0xB6 0xB7 0xB8 0xB9 0xBA 0xBB 0xBC 0xBD 0xBE 0xBF
0xC0 0xC1 0xC2 0xC3 0xC4 0xC5 0xC6 0xC7 0xC8 0xC9 0xCA 0xCB 0xCC 0xCD 0xCE 0xCF
0xD0 0xD1 0xD2 0xD3 0xD4 0xD5 0xD6 0xD7 0xD8 0xD9 0xDA 0xDB 0xDC 0xDD 0xDE 0xDF
0xE0 0xE1 0xE2 0xE3 0xE4 0xE5 0xE6 0xE7 0xE8 0xE9 0xEA 0xEB 0xEC 0xED 0xEE 0xEF
0xF0 0xF1 0xF2 0xF3 0xF4 0xF5 0xF6 0xF7 0xF8 0xF9 0xFA 0xFB 0xFC 0xFD 0xFE 0xFF
Read from rx_buffer:
0x00 0x01 0x02 0x03 0x04 0x05 0x06 0x07 0x08 0x09 0x0A 0x0B 0x0C 0x0D 0x0E 0x0F
0x10 0x11 0x12 0x13 0x14 0x15 0x16 0x17 0x18 0x19 0x1A 0x1B 0x1C 0x1D 0x1E 0x1F
0x20 0x21 0x22 0x23 0x24 0x25 0x26 0x27 0x28 0x29 0x2A 0x2B 0x2C 0x2D 0x2E 0x2F
0x30 0x31 0x32 0x33 0x34 0x35 0x36 0x37 0x38 0x39 0x3A 0x3B 0x3C 0x3D 0x3E 0x3F
0x40 0x41 0x42 0x43 0x44 0x45 0x46 0x47 0x48 0x49 0x4A 0x4B 0x4C 0x4D 0x4E 0x4F
0x50 0x51 0x52 0x53 0x54 0x55 0x56 0x57 0x58 0x59 0x5A 0x5B 0x5C 0x5D 0x5E 0x5F
0x60 0x61 0x62 0x63 0x64 0x65 0x66 0x67 0x68 0x69 0x6A 0x6B 0x6C 0x6D 0x6E 0x6F
0x70 0x71 0x72 0x73 0x74 0x75 0x76 0x77 0x78 0x79 0x7A 0x7B 0x7C 0x7D 0x7E 0x7F
0x80 0x81 0x82 0x83 0x84 0x85 0x86 0x87 0x88 0x89 0x8A 0x8B 0x8C 0x8D 0x8E 0x8F
0x90 0x91 0x92 0x93 0x94 0x95 0x96 0x97 0x98 0x99 0x9A 0x9B 0x9C 0x9D 0x9E 0x9F
0xA0 0xA1 0xA2 0xA3 0xA4 0xA5 0xA6 0xA7 0xA8 0xA9 0xAA 0xAB 0xAC 0xAD 0xAE 0xAF
0xB0 0xB1 0xB2 0xB3 0xB4 0xB5 0xB6 0xB7 0xB8 0xB9 0xBA 0xBB 0xBC 0xBD 0xBE 0xBF
0xC0 0xC1 0xC2 0xC3 0xC4 0xC5 0xC6 0xC7 0xC8 0xC9 0xCA 0xCB 0xCC 0xCD 0xCE 0xCF
0xD0 0xD1 0xD2 0xD3 0xD4 0xD5 0xD6 0xD7 0xD8 0xD9 0xDA 0xDB 0xDC 0xDD 0xDE 0xDF
0xE0 0xE1 0xE2 0xE3 0xE4 0xE5 0xE6 0xE7 0xE8 0xE9 0xEA 0xEB 0xEC 0xED 0xEE 0xEF
0xF0 0xF1 0xF2 0xF3 0xF4 0xF5 0xF6 0xF7 0xF8 0xF9 0xFA 0xFB 0xFC 0xFD 0xFE 0xFF
SPI-GD25Q16 Test Passed!
```

5.14. EXMC_SDRAM

5.14.1. DEMO purpose

This demo includes the following function of GD32 MCU:

- Learn to use EXMC control the SDRAM.

5.14.2. DEMO running result

GD32F527I-EVAL board has EXMC module to control SDRAM. Before running the demo,

JP10 must be fitted to SDRAM, JP21 must be fitted to USART. Download the program <14_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 LED3. 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

```

5.15. EXMC_SDRAM_DeepSleep

5.15.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

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

5.15.2. DEMO running result

GD32F527I-EVAL board has EXMC module to control SDRAM. Before running the demo, JP10 must be fitted to SDRAM, JP21 must be fitted to USART. Download the program <15_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 LED2 will light on. Thirdly, press the user key to wake up MCU, compare the data which read from SDRAM with the write data, if the test pass, LED1 will be turned on. Otherwise, turn on the LED3. Information via a HyperTerminal output as following:

```

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

User 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
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

5.16. EXMC_NandFlash

5.16.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

- Learn to use EXMC control the NAND flash.

5.16.2. DEMO running result

GD32F527I-EVAL board has EXMC module to control NAND flash. Before running the demo, JP21 must be fitted to USART. Download the program <16_EXMC_NandFlash> to the EVAL board. This demo shows the write and read operation process of NAND flash memory by EXMC module. If the test pass, LED1 will be turned on. Otherwise, turn on the LED3. Information via a HyperTerminal output as following:

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

Write data successfully!
Read data successfully!
Check the data!
Access NAND flash successfully!
The data to be read:
  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. SDIO_SDCardTest

5.17.1. 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

GD32F527I-EVAL board has a secure digital input/output interface (SDIO) which defines the SD/SD I/O /MMC CE-ATA card host interface. This demo will show how to use SDIO to operate on SD card.

5.17.2. DEMO running result

Jump the JP21 to USART to show the print message through HyperTerminal, and jump the JP11/JP12/JP18 to SDIO. Download the program <17_SDIO_SDCardTest> to the EVAL board and run. Connect serial cable to COM, open the HyperTerminal. Firstly, LED1~LED3 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, LED3 and turn off LED2. Otherwise, turn on LED1~LED3.

Uncomment the macro DATA_PRINT to print out the data and display them through HyperTerminal. Set bus mode(1-bit or 4-bit) and data transfer mode(polling mode or DMA mode) by comment and uncomment the related statements.

Information via a serial port output 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!
```

5.18. CAN_Network

5.18.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

- Learn to use the CAN0 communication between two boards

GD32F527I-EVAL board integrates CAN (controller area network) bus controller. It is a common industrial control bus. The CAN bus controller supports the CAN protocols version 2.0A, 2.0B, ISO11891-1:2015 and BOSCH CAN FD specification. This routine demonstrates the communication between two boards through CAN0.

5.18.2. DEMO running result

This example is tested with two GD32F527I-EVAL boards. Jump the JP21 to USART and JP35, JP36 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 <18_CAN_Network> to the two EVAL boards, and connect serial cable to USART. 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 LED1 will toggle one time.

The output information via the serial port is as following.

```

please press the Tamper key to transmit data!

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

```

5.19. RCU_Clock_Out

5.19.1. 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.

5.19.2. DEMO running result

Jump the JP21 to USART with the jumper cap, and download the program <19_RCU_Clock_Out> to the EVAL board and run. Connect serial cable to USART, open the HyperTerminal. When the program is running, HyperTerminal will display the initial information. Then user can choose the type of the output clock by pressing the TAMPER button. After pressing, the corresponding LED will be turned on and HyperTerminal will display which mode be selected. The frequency of the output clock can be observed through the oscilloscope by PA8 and PC9 pin.

Information via a serial port output as following:

```

/===== GigaDevice Clock output Demo =====/
press tamper key to select clock output source
CK_OUT0: IRC16M, CK_OUT1: system clock/5
CK_OUT0: LXTAL, CK_OUT1: PLLI2SR/5

```

5.20. CTC_Calibration

5.20.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

- Learn to use external low speed crystal oscillator (LXTAL) to implement the CTC calibration function.
- Learn to use clock trim controller (CTC) to trim internal 48MHz RC oscillator (IRC48M)

clock.

The CTC unit trim the frequency of the IRC48M based on an external accurate reference signal source. It can automatically adjust the trim value to provide a precise IRC48M clock.

5.20.2. DEMO running result

Download the program <20_CTC_Calibration > to the GD32F527I-EVAL board and run. The LED1 will turn on if the internal 48MHz RC oscillator (IRC48M) clock trim is OK.

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, jump the JP21 to USART with the jumper cap and connect serial cable to USART. After power-on, all the LEDs are off. The mcu will enter sleep mode and the software stops running. When the USART0 receives a byte of data from the HyperTerminal, the mcu will wake up from a receive interrupt. And all the LEDs will flash together.

5.22. 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

Jump the JP21 to USART with the jumper cap, and download the program <22_RTC_Calendar> to the EVAL board and run. Connect serial cable to USART, open the HyperTerminal. After start-up, the program will ask to set the time on the HyperTerminal. The calendar will be displayed on the HyperTerminal.


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

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

please input hour:

12

please input minute:

12

please input second:

12

** RTC time configuration success! **

Current time: 12:12:12
```

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

5.23.2. DEMO running result

Use the DuPont line to connect the TIMER1_CH2 (PB10) and LED1 (PF7), and then download the program <23_TIMER_Breath_LED> to the board and run. When the program is running, you can see LED1 lighting from dark to bright gradually and then gradually darken, just like breathing as rhythm.

5.24. TLI_IPA

5.24.1. DEMO purpose

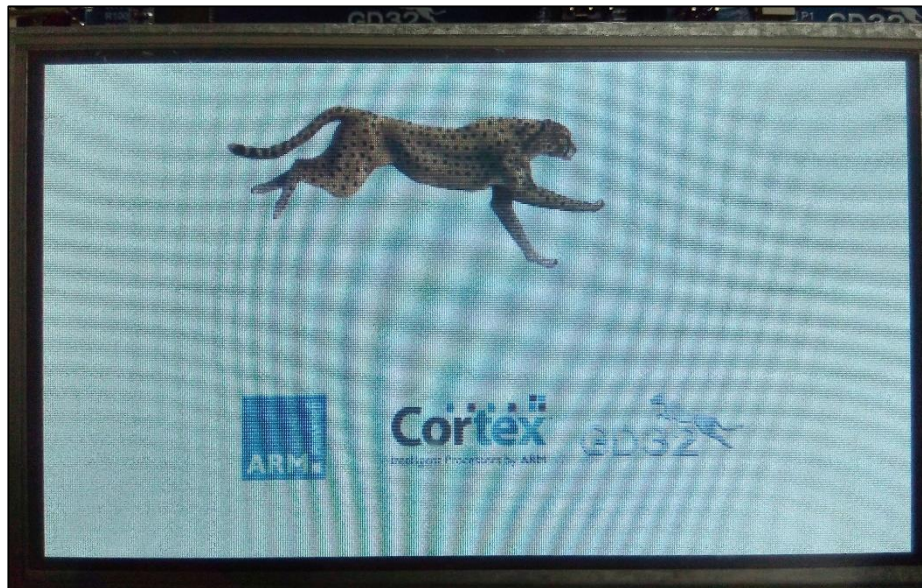
This demo includes the following functions of GD32 MCU:

- 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

Jump the JP13, JP15, JP20, JP23, JP25, JP35, JP36 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

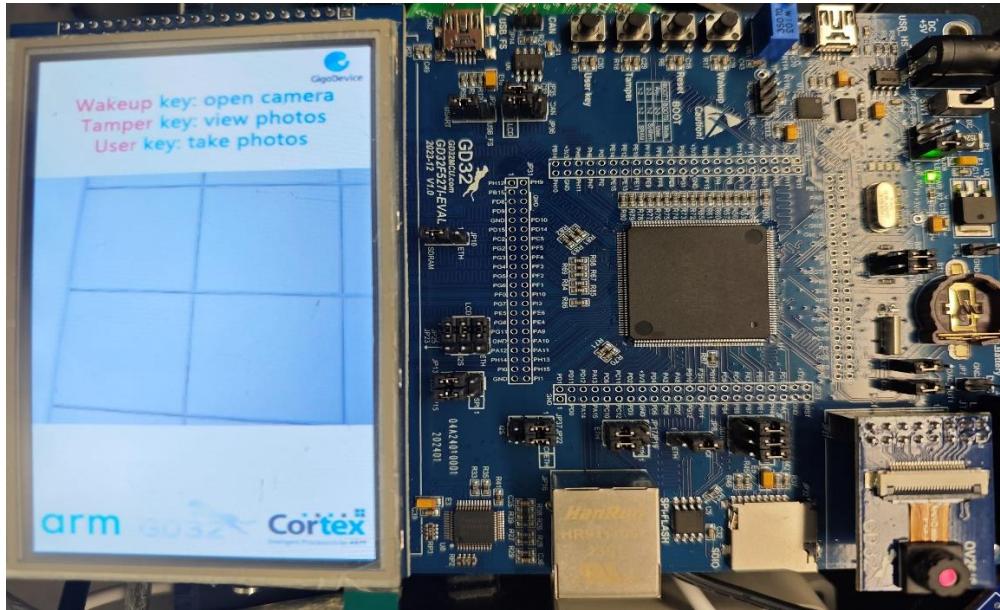
5.25.1. DEMO purpose

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 JP8/JP9/JP11/JP12/JP18/JP37 to DCI, jumper JP13/JP15/JP20/JP23/JP25/JP35/JP36 to LCD, jumper JP10 to SDRAM. Download the program<25_DCI_OV2640>to the GD32F527I-EVAL 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. CAU

5.26.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

- Learn DES, Triple-DES and AES algorithm
- Learn Electronic codebook (ECB) mode, Cipher block chaining (CBC) mode, Counter (CTR) mode, Galois/counter (GCM) mode, combined cipher machine (CCM) mode, Cipher Feedback (CFB) mode, and Output Feedback (OFB) mode
- Learn to use CAU to encrypt and decrypt
- Learn to communicate with PC by USART

5.26.2. DEMO running result

Download the program <26_CAU> to the EVAL board and run. When the program is running, the serial terminal tool will display the information, as shown in the following figure. Plaintext data value, the encryption algorithm, and the mode can be selected are shown. After the user setting the algorithm and mode according to the serial output information indicating, serial port will print out selected algorithm and mode, as shown below.

```

Plain data :
0x30 0x31 0x32 0x33 0x34 0x35 0x36 0x37
0x38 0x39 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 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 0x7A 0x7A
=====Choose CAU algorithm=====
1: DES algorithm
2: TDES algorithm
3: AES algorithm

You choose to use DES algorithm
=====Choose CAU mode=====
1: ECB mode
2: CBC mode
3: CTR mode only when choose AES algorithm
4: GCM mode only when choose AES algorithm
5: CCM mode only when choose AES algorithm
6: CFB mode only when choose AES algorithm
7: OFB mode only when choose AES algorithm

You choose to use ECB mode

```

After selection, the program starts encryption and decryption operations, the results are printed through the serial port.

```

Encrypted data with DES Mode ECB :

0xB3 0x9F 0xBD 0x94 0xC4 0xE7 0xC2 0xAA
0x2F 0x5E 0xDE 0x61 0x21 0x36 0x36 0x62
0x61 0x84 0xF8 0xCA 0x4D 0x4E 0x55 0x14
0x93 0x08 0xFC 0xE4 0x82 0x65 0x48 0x8F
0xC6 0x02 0x1C 0xAD 0xF9 0xA0 0xEB 0x51
0x3C 0x29 0xEF 0x55 0xDB 0x15 0x15 0x8F
0x6E 0x5E 0x78 0xAA 0x61 0xDD 0xEB 0xA6
0x2A 0xDA 0xBA 0x87 0x6C 0xD3 0xB1 0x23

Decrypted data with DES Mode ECB :

0x30 0x31 0x32 0x33 0x34 0x35 0x36 0x37
0x38 0x39 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 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 0x7A 0x7A

Example restarted...

```

And then restart for users to select a different algorithm and mode to repeat demo, as shown

below.

```
Example restarted...

Plain data :
0x30 0x31 0x32 0x33 0x34 0x35 0x36 0x37
0x38 0x39 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 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 0x7A 0x7A
=====Choose CAU algorithm=====
1: DES algorithm
2: TDES algorithm
3: AES algorithm
```

5.27. HAU

5.27.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

- Learn SHA-1, SHA-224, SHA-256 and MD5 algorithm.
- Learn HASH mode and HMAC (keyed-hash message authentication code) mode.
- Learn to use HAU to calculate digest for the input message.
- Learn to communicate with PC by USART.

5.27.2. DEMO running result

Jump the JP21 to USART with the jumper cap, Download the program <27_HAU> to the EVAL board and run. Connect USB cable to CN4. JP21 must be jumped to USART. When the program is running, the serial terminal tool will display the information, as shown in the following figure. After the user setting the algorithm and mode according to the serial output information indicating, If the hash calculation results are consistent with the expected result, then the LED1 will light up, and serial port will print out selected algorithm and mode, as shown below.

```

message to be hashed:

CHN GigaDevice Semiconductor IncCHN GigaDevice Semiconductor IncCHN
GigaDevice Semiconductor IncCHN GigaDevice Semiconductor IncCHN
GigaDevice Semiconductor IncCHN GigaDevice Semiconductor IncCHN
GigaDevice Semiconductor IncCHN GigaDevice Semiconductor
Inc=====Choose HAU algorithm=====
1: SHA1 algorithm
2: SHA224 algorithm
3: SHA256 algorithm
4: MD5 algorithm

You choose to use SHA1 algorithm
=====Choose HAU mode=====
1: HASH mode
2: HMAC mode

You choose to use HASH mode

```

After selection, the program starts digest calculation, the results are printed through the serial port. And then restart for users to select a different algorithm and mode to repeat demo, as shown below.

```

message digest with SHA-1 Mode HASH (160 bits):

0x06 0x9B 0x69 0x4C
0x14 0x07 0xDE 0x79
0xB7 0x2F 0x31 0xD9
0xB6 0xB4 0x97 0x84
0x6C 0x24 0x5A 0xB0

Example restarted...

message to be hashed:

CHN GigaDevice Semiconductor IncCHN GigaDevice Semiconductor IncCHN
GigaDevice Semiconductor IncCHN GigaDevice Semiconductor IncCHN
GigaDevice Semiconductor IncCHN GigaDevice Semiconductor IncCHN
GigaDevice Semiconductor IncCHN GigaDevice Semiconductor
Inc=====Choose HAU algorithm=====
1: SHA1 algorithm
2: SHA224 algorithm
3: SHA256 algorithm
4: MD5 algorithm

```

5.28. PKCAU

5.28.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

- Learn modular addition algorithm by interrupt

5.28.2. DEMO running result

Download the program <28_PKCAU_Modular_Addition_Interrupt> to the EVAL board. After system start-up, it is initialized first, then, perform modular addition computation, when the computation is completed, an interrupt will be generated, finally, read results from specific PKCAU RAM address and compare with the expected result, if the result is the same with the expected one, LED1 will on, or else, LED2 is on.

5.29. TRNG_Get_Random

5.29.1. DEMO purpose

This demo includes the following functions of GD32 MCU:

- Learn to use TRNG generate the random number.
- Learn to communicate with PC by USART.

5.29.2. DEMO running result

Jump the JP21 to USART with the jumper cap, and download the program <29_TRNG_Get_Random> to the EVAL board and run. Connect serial cable to COM0, open the serial terminal tool supporting hex format communication. When the program is running, the serial terminal tool will display the initial information. User can use the serial terminal tool to input the minimum and maximum values (for example, the minimum value is 0x00, the maximum value is 0x09), then application will generate random number in the input range and display it by the serial terminal tool.

Information via a serial port output as following:

```

/=====Gigadevice TRNG test=====/
TRNG init ok
Please input min num (hex format):
Please input max num (hex format):
Input min num is 0
Input max num is 9
Generate random num1 is 9
Generate random num2 is 8
Please input min num (hex format):

```

5.30. ENET

5.30.1. 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 GD32F527I-EVAL 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.

JP8, JP10, JP17, JP19, JP20, JP22 must be fitted. JP21 jump to Usart.

It is configured in RMII mode, and 25MHz oscillator is used, the system clock is configured to 200MHz.

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.

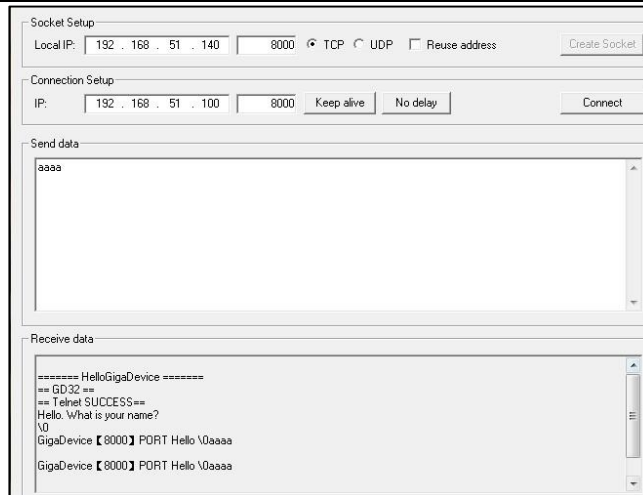
If users need dhcp function, it can be configured from the private defines in main.h. This function is closed by default.

Note: Users should configure ip address, mask and gw of GD32F527I-EVAL board or served according to the actual net situation from the private defines in main.h.

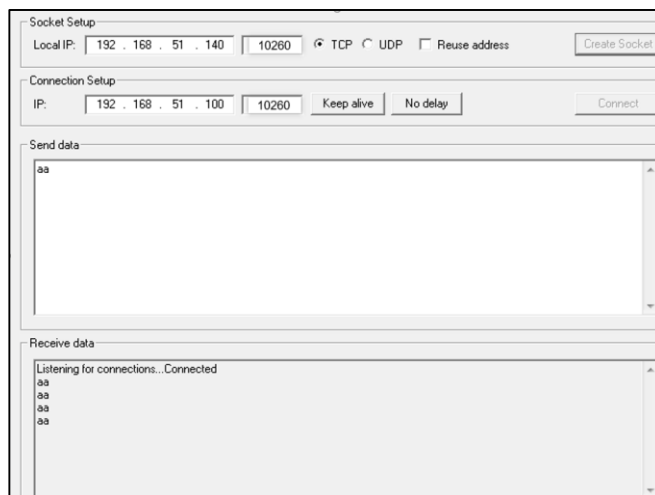
DEMO running result

Download the program <FreeRTOS_tcpudp> to the EVAL board, LED3 will light every 250ms.

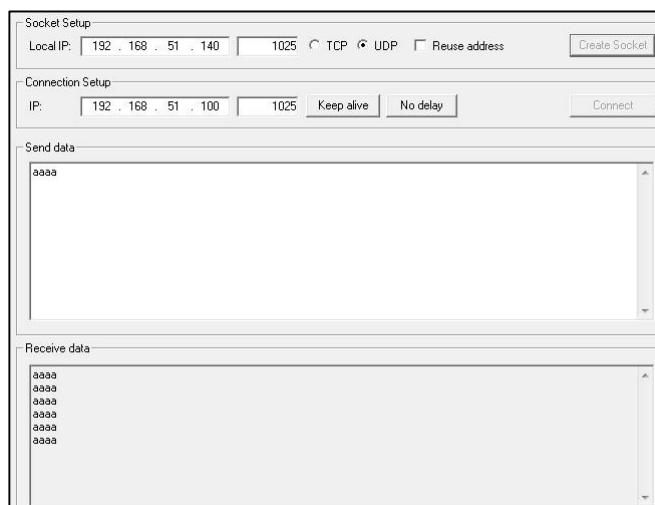
Using Network assistant software, configure the pc side to tcp client, using 8000 port, and when send something through the assistant, users can see the reply from the server:



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.

5.30.2. 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 GD32F527I-EVAL 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.

JP8, JP10, JP17, JP19, JP20, JP22 must be fitted. JP21 jump to Usart.

It is configured in RMI mode, and 25MHz oscillator is used, the system clock is configured to 200MHz.

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.

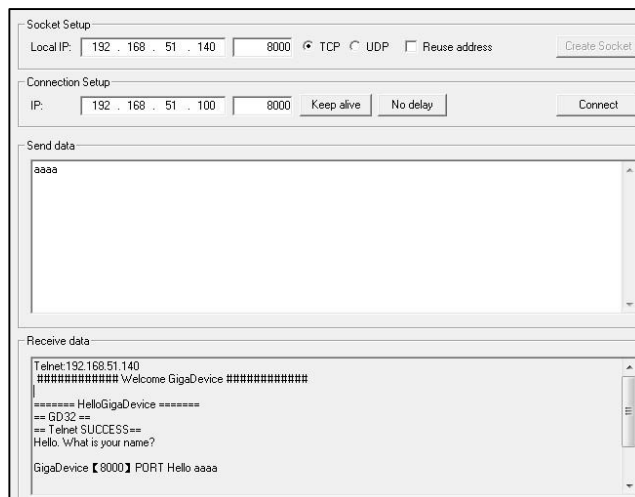
Note: Users should configure ip address, mask and gw of GD32F527I-EVAL-V1.0 board, or server according to the actual net situation from the private defines in main.h.

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

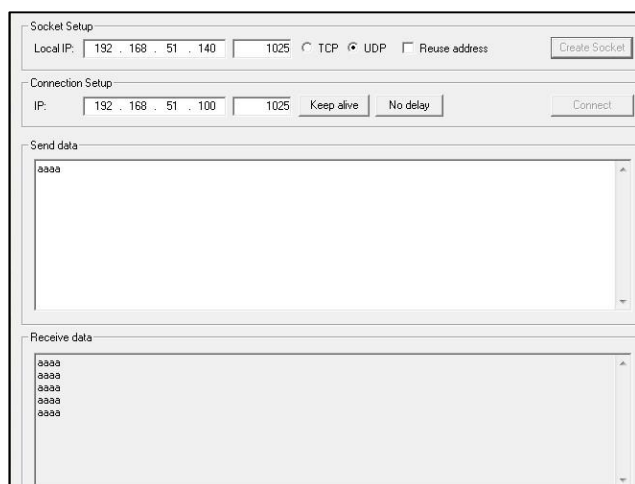
when send something through the assistant, users can see the reply from the server:



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.

5.30.3. 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 GD32F527I-EVAL 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.

JP8, JP10, JP17, JP19, JP20, JP22 must be fitted. JP21 jump to Usart.

It is configured in RMII mode, and 25MHz oscillator is used, the system clock is configured to 200MHz.

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.

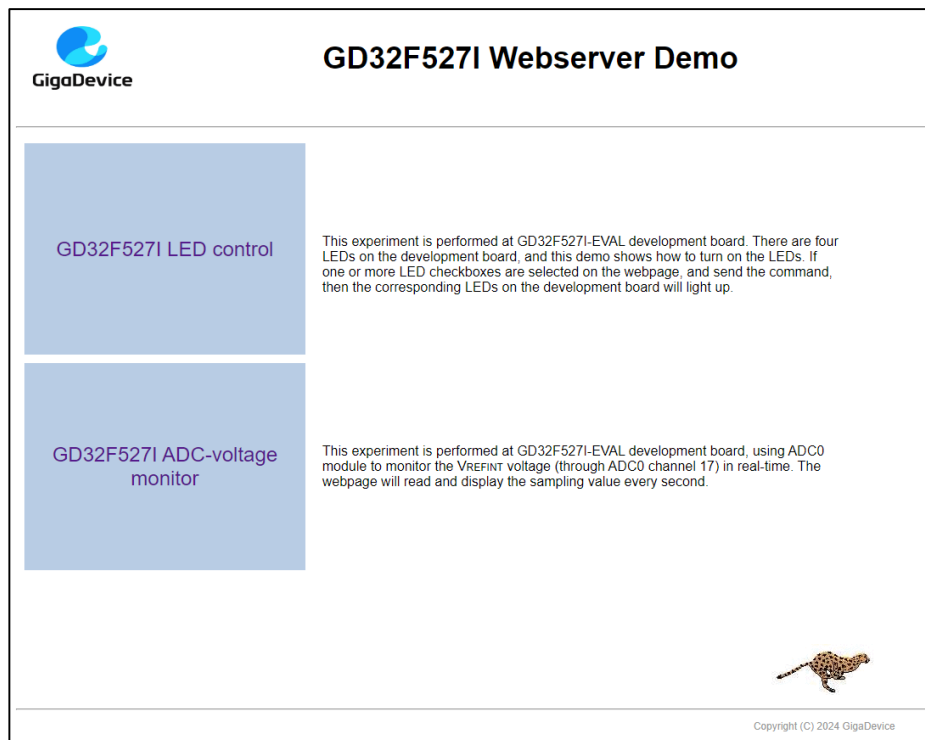
By default, the packet reception is polled in while(1). If users want to receive packet in interrupt service, uncomment the macro define `USE_ENET_INTERRUPT` in main.h.

Note: Users should configure ip address, mask and gw of GD32F527I-EVAL board according to the actual net situation from the private defines in main.h.

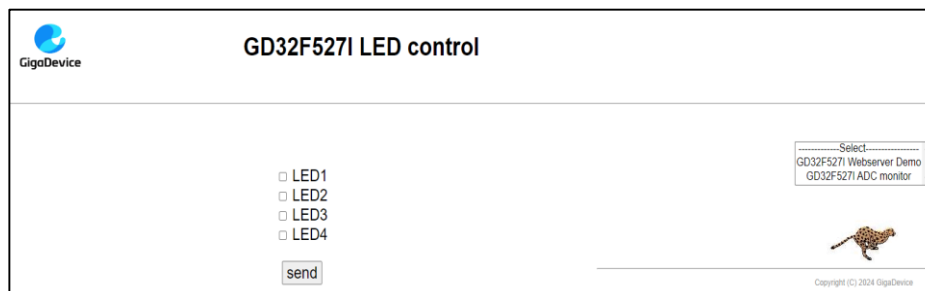
DEMO running result

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

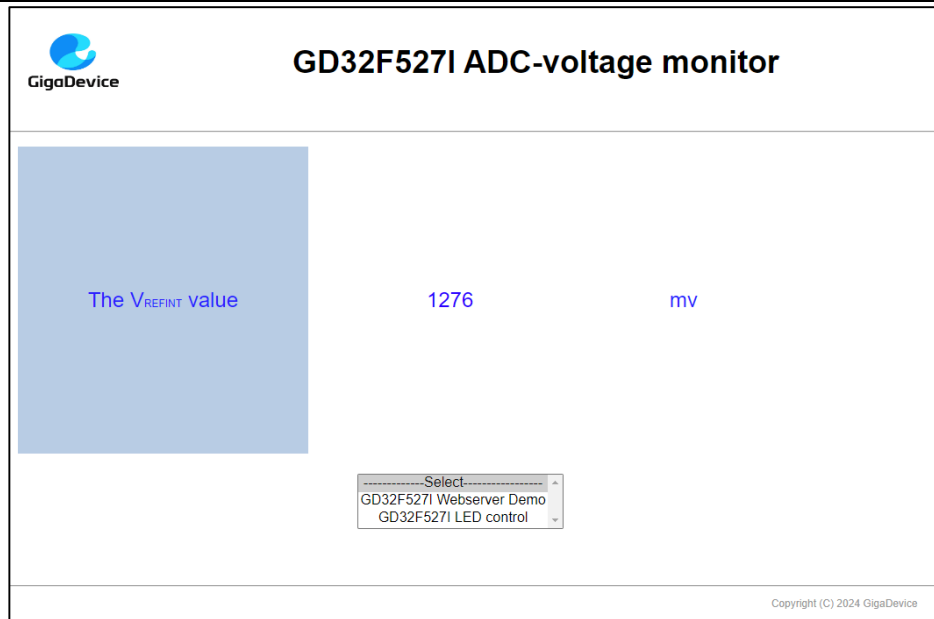
The web home page shows as below:



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.

5.31. USB_Device

5.31.1. 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

GD32F527I-EVAL board has four keys, one USB_FS interface and one USB_HS interface. The four keys are Reset key, Wakeup key, User key and Tamper key. In this demo, the GD32F527I-EVAL 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

Download the program <31_USB_Device\HID_Keyboard> to the EVAL board and run. If user press the Wakeup key, will output 'b'. If user press the User key, will output 'c'. If user press the Tamper key, will output 'a'.

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.

5.31.2. MSC_Udisk

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 MSC (mass storage) device

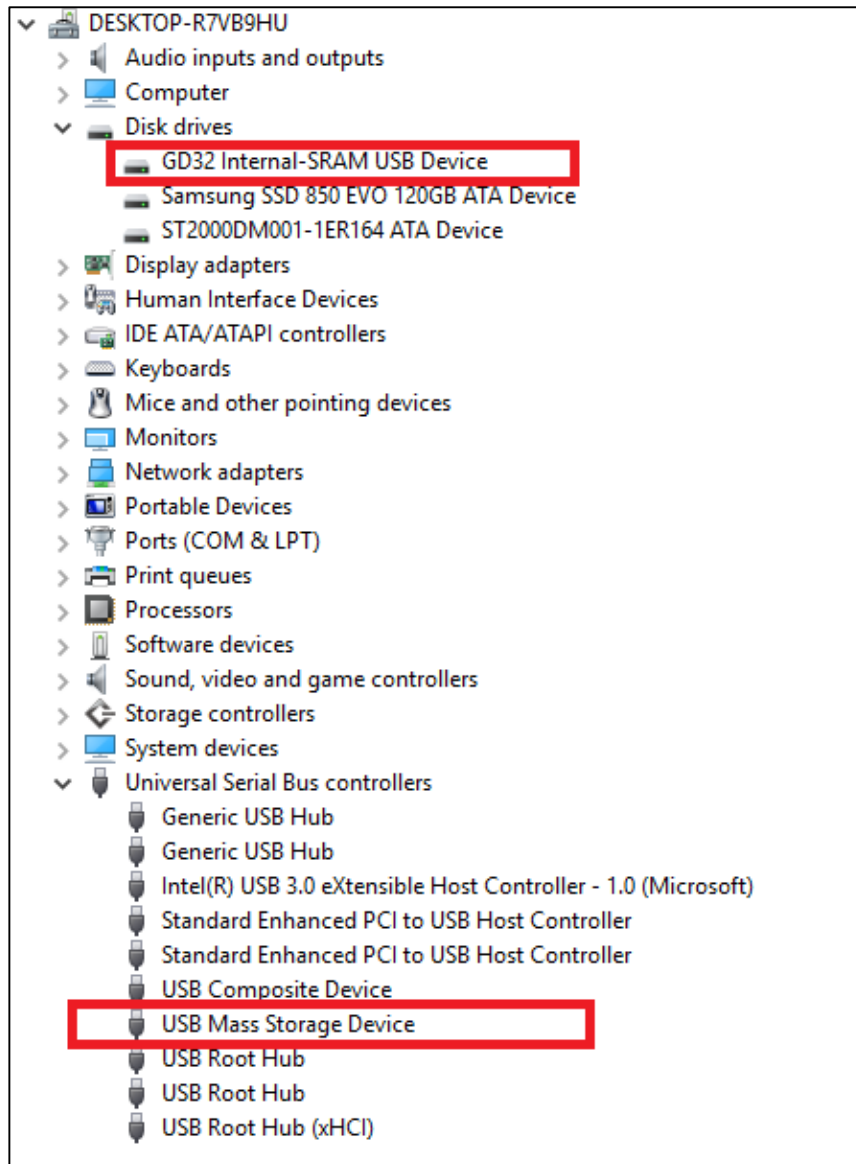
This demo mainly implements a U disk. U disk is currently very widely used removable MSC devices. MSC, the Mass Storage device Class, is a transport protocol between a computer and mobile devices, which allow a universal serial bus (USB) equipment to access a host computing device, file transfer between 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 SRAM 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.

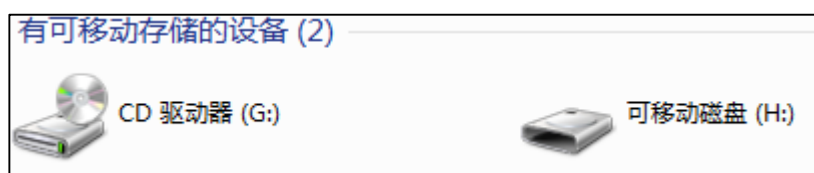
DEMO Running Result

Download the program <31_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 one more disk drives in the device manager of PC, as shown below:



Then, after opening the resource manager, user will see one more disk, as shown in the following diagram:



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

5.32. USB_Host

5.32.1. HID_Host

DEMO Purpose

This demo includes the following functions of GD32 MCU:

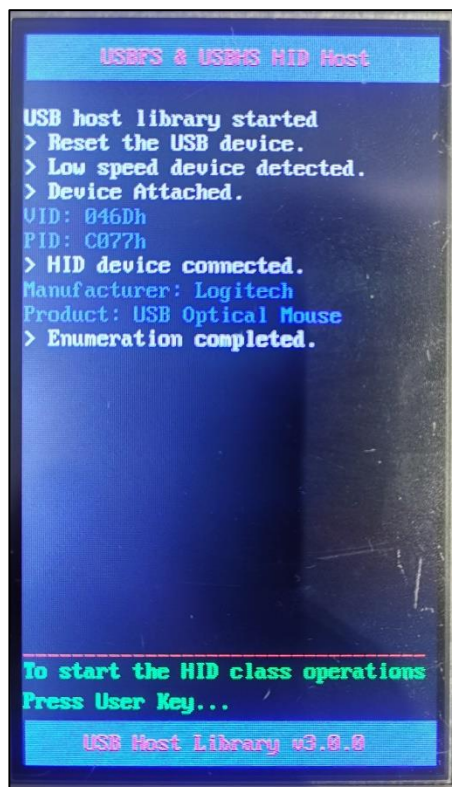
- Learn to use the USBFS/USBHS as a HID host
- Learn the operation between the HID host and the mouse device
- Learn the operation between the HID host and the keyboard device

GD32F527I-EVAL evaluation board integrates the USBFS module and 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 USBFS/USBHS as a USB HID host to communicate with external USB HID device.

DEMO Running Result

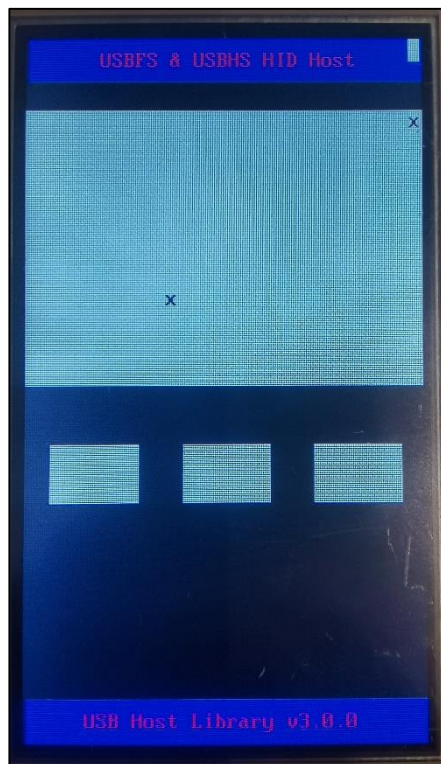
Jump the JP10 to SDRAM, jump the JP13/JP15/JP20/JP23/JP25/JP35/JP36 to LCD, and jump the JP26 to USB_HS in USBHS target, then insert the OTG cable to the USB port, download the program <32_USB_Host\HID_Host> to the EVAL board and run.

If a mouse has been attached, the user will see the information of mouse enumeration.

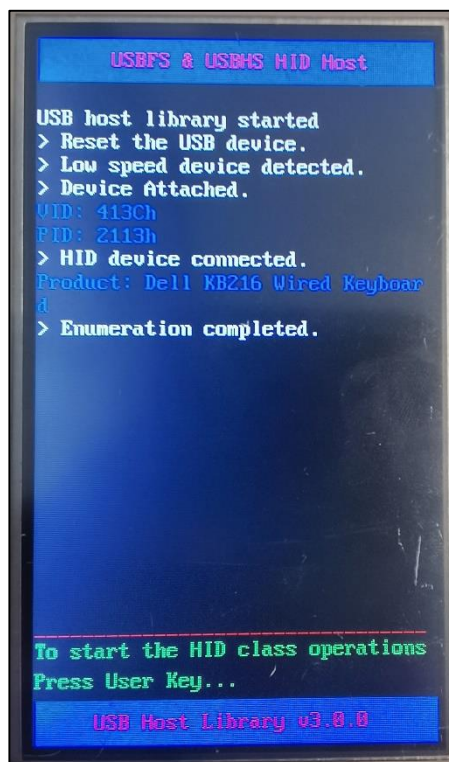


First pressing the User key will see the inserted device is mouse, and then moving the mouse will move the 'x' in the LCD screen and pressing the button will show the magenta color

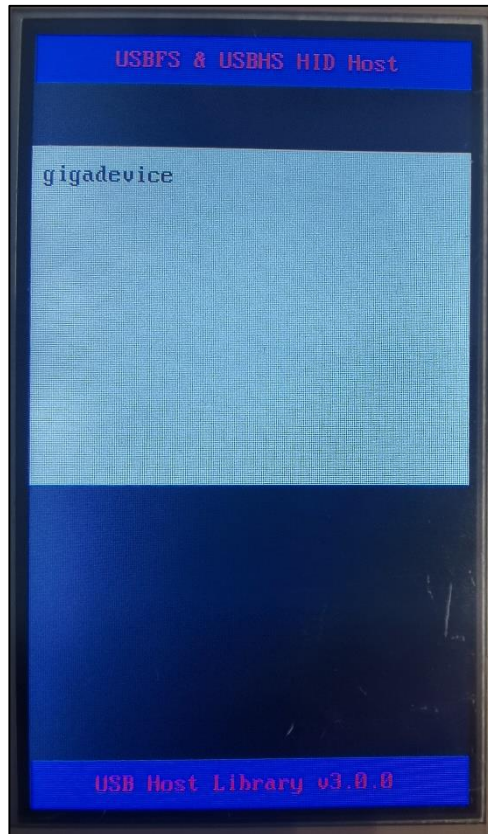
rectangle in the LCD screen.



If a keyboard has been attached, the user will see the information of keyboard enumeration.



First pressing the User key will see the inserted device is keyboard, and then pressing the keyboard will print the char of the button in the LCD screen.



5.32.2. 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 U-disk

GD32F527I-EVAL evaluation board integrates the 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 USBFS and USBHS as a USB MSC host to communicate with external U-disk.

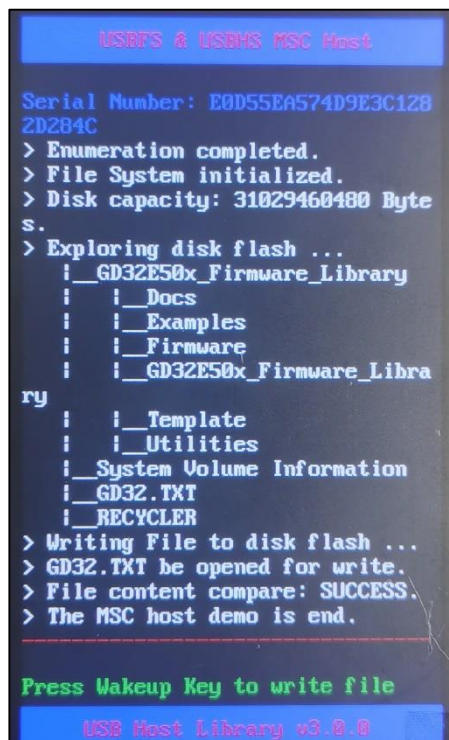
DEMO Running Result

Jump the JP10 to SDRAM, jump the JP13/JP15/JP20/JP23/JP25/JP35/JP36 to LCD, and jump the JP26 to USB_HS in USBHS target, then insert the OTG cable to the USB port, download the program <32_USB_Host\MSC_Host > to the EVAL board and run.

If a U-disk has been attached, the user will see the information of U-disk enumeration.



First pressing the User key will see the U-disk information, next pressing the Tamper key will see the root content of the U-disk, then press the Wakeup key will write file to the U-disk, finally the user will see information that the MSC host demo is end.



6. Revision history

Table 6-1 Revision history

Revision No.	Description	Date
1.0	Initial Release	Mar.18, 2024
1.1	Update chapter 3	Jan.25, 2025
1.2	Update chapter 3	Aug.08, 2025

Important Notice

This document is the property of GigaDevice Semiconductor Inc. and its subsidiaries (the "Company"). This document, including any product of the Company described in this document (the "Product"), is owned by the Company according to the laws of the People's Republic of China and other applicable laws. The Company reserves all rights under such laws and no Intellectual Property Rights are transferred (either wholly or partially) or licensed by the Company (either expressly or impliedly) herein. The names and brands of third party referred thereto (if any) are the property of their respective owner and referred to for identification purposes only.

To the maximum extent permitted by applicable law, the Company makes no representations or warranties of any kind, express or implied, with regard to the merchantability and the fitness for a particular purpose of the Product, nor does the Company assume any liability arising out of the application or use of any Product. Any information provided in this document is provided only for reference purposes. It is the sole responsibility of the user of this document to determine whether the Product is suitable and fit for its applications and products planned, and properly design, program, and test the functionality and safety of its applications and products planned using the Product. The Product is designed, developed, and/or manufactured for ordinary business, industrial, personal, and/or household applications only, and the Product is not designed or intended for use in (i) safety critical applications such as weapons systems, nuclear facilities, atomic energy controller, combustion controller, aeronautic or aerospace applications, traffic signal instruments, pollution control or hazardous substance management; (ii) life-support systems, other medical equipment or systems (including life support equipment and surgical implants); (iii) automotive applications or environments, including but not limited to applications for active and passive safety of automobiles (regardless of front market or aftermarket), for example, EPS, braking, ADAS (camera/fusion), EMS, TCU, BMS, BSG, TPMS, Airbag, Suspension, DMS, ICMS, Domain, ESC, DCDC, e-clutch, advanced-lighting, etc.. Automobile herein means a vehicle propelled by a self-contained motor, engine or the like, such as, without limitation, cars, trucks, motorcycles, electric cars, and other transportation devices; and/or (iv) other uses where the failure of the device or the Product can reasonably be expected to result in personal injury, death, or severe property or environmental damage (collectively "Unintended Uses"). Customers shall take any and all actions to ensure the Product meets the applicable laws and regulations. The Company is not liable for, in whole or in part, and customers shall hereby release the Company as well as its suppliers and/or distributors from, any claim, damage, or other liability arising from or related to all Unintended Uses of the Product. Customers shall indemnify and hold the Company, and its officers, employees, subsidiaries, affiliates as well as its suppliers and/or distributors harmless from and against all claims, costs, damages, and other liabilities, including claims for personal injury or death, arising from or related to any Unintended Uses of the Product.

Information in this document is provided solely in connection with the Product. The Company reserves the right to make changes, corrections, modifications or improvements to this document and the Product described herein at any time without notice. The Company shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them. Information in this document supersedes and replaces information previously supplied in any prior versions of this document.