

GigaDevice Semiconductor Inc.

GD32H759I-EVAL 评估板
用户指南
Rev1.0

目录

目录.....	1
图	5
表	6
1. 简介.....	7
2. 功能引脚分配	7
3. 入门指南	11
4. 硬件设计概述	11
4.1. 供电电源.....	11
4.2. 启动方式选择.....	12
4.3. LED 指示灯.....	12
4.4. 按键	12
4.5. ADC	13
4.6. DAC	13
4.7. CAN	13
4.8. DCI.....	14
4.9. ENET.....	14
4.10. HPDF	15
4.11. I2C	15
4.12. I2S	15
4.13. SAI.....	16
4.14. OSPI	16
4.15. SPI_LCD.....	17
4.16. SDIO	17
4.17. SDRAM.....	18
4.18. TLI_LCD	19
4.19. USART.....	19
4.20. USB	20
4.21. Extension.....	20
4.22. GD-Link.....	21

4.23.	MCU.....	22
5.	例程使用指南	23
5.1.	GPIO 流水灯	23
5.1.1.	DEMO 目的	23
5.1.2.	DEMO 执行结果	23
5.2.	GPIO 按键轮询模式	23
5.2.1.	DEMO 目的	23
5.2.2.	DEMO 执行结果	23
5.3.	EXTI 按键中断模式	24
5.3.1.	DEMO 目的	24
5.3.2.	DEMO 执行结果	24
5.4.	串口打印	24
5.4.1.	DEMO 目的	24
5.4.2.	DEMO 执行结果	24
5.5.	串口中断收发	25
5.5.1.	DEMO 目的	25
5.5.2.	DEMO 执行结果	25
5.6.	串口 DMA 收发	25
5.6.1.	DEMO 目的	25
5.6.2.	DEMO 执行结果	25
5.7.	ADC 温度传感器_Vrefint.....	26
5.7.1.	DEMO 目的	26
5.7.2.	DEMO 执行结果	26
5.8.	ADC0 和 ADC1 跟随模式.....	27
5.8.1.	DEMO 目的	27
5.8.2.	DEMO 执行结果	27
5.9.	ADC0 和 ADC1 规则并行模式	28
5.9.1.	DEMO 目的	28
5.9.2.	DEMO 执行结果	28
5.10.	DAC 输出电压值	28
5.10.1.	DEMO 目的	错误!未定义书签。
5.10.2.	DEMO 执行结果	错误!未定义书签。
5.11.	I2C 访问 EEPROM	29
5.11.1.	DEMO 目的	29
5.11.2.	DEMO 执行结果	29
5.12.	HPDF_I2S 音频播放.....	30
5.12.1.	DEMO 目的	30
5.12.2.	DEMO 执行结果	31

5.13.	HPDF_SAI 音频播放	31
5.13.1.	DEMO 目的	31
5.13.2.	DEMO 执行结果	31
5.14.	SPI 四线 LCD	31
5.14.1.	DEMO 目的	31
5.14.2.	DEMO 执行结果	31
5.15.	OSPI 八线 Flash	32
5.15.1.	DEMO 目的	32
5.15.2.	DEMO 执行结果	32
5.16.	EXMC_SDRAM	33
5.16.1.	DEMO 目的	33
5.16.2.	DEMO 执行结果	33
5.17.	EXMC_SDRAM 与深度睡眠模式	34
5.17.1.	DEMO 目的	34
5.17.2.	DEMO 执行结果	35
5.18.	SD 卡测试	37
5.18.1.	DEMO 目的	37
5.18.2.	DEMO 执行结果	37
5.19.	CAN 网络通信	38
5.19.1.	DEMO 目的	38
5.19.2.	DEMO 执行结果	38
5.20.	RCU 时钟输出	38
5.20.1.	DEMO 目的	38
5.20.2.	DEMO 执行结果	38
5.21.	PMU 睡眠模式唤醒	39
5.21.1.	DEMO 目的	39
5.21.2.	DEMO 执行结果	39
5.22.	RTC 日历	39
5.22.1.	DEMO 目的	39
5.22.2.	DEMO 执行结果	39
5.23.	TIMER 呼吸灯	40
5.23.1.	DEMO 目的	40
5.23.2.	DEMO 执行结果	40
5.24.	TLI_IPA	40
5.24.1.	DEMO 目的	40
5.24.2.	DEMO 执行结果	40
5.25.	DCI_OV2640	41
5.25.1.	DEMO 目的	41
5.25.2.	DEMO 执行结果	41

5.26.	以太网	42
5.26.1.	FreeRTOS 上的服务器/客户端	42
5.26.2.	服务器/客户端	45
5.26.3.	web 服务器	47
5.27.	USB 设备	49
5.27.1.	虚拟串口	49
5.27.2.	HID_键盘	49
5.28.	USB 主机	50
5.28.1.	USB HID 主机	50
5.28.2.	USB MSC 主机	51
6.	版本历史	52

图

图 4-1. 供电电源原理图	11
图 4-2. 启动方式选择原理图	12
图 4-3. LED 功能原理图	12
图 4-4. 按键功能原理图	12
图 4-5. ADC 原理图	13
图 4-6. DAC 原理图	13
图 4-7. CAN 原理图	13
图 4-8. DCI 原理图	14
图 4-9. ENET 原理图	14
图 4-10. HPDF 原理图	15
图 4-11. I2C 原理图	15
图 4-12. I2S 原理图	15
图 4-13. SAI 原理图	16
图 4-14. OSPI 原理图	16
图 4-15. SPI_LCD 原理图	17
图 4-16. SDIO 原理图	17
图 4-17. SDRAM 原理图	18
图 4-18. TLI 原理图	19
图 4-19.USART 原理图	19
图 4-20. USB 原理图	20
图 4-21. Extension 原理图	20
图 4-22. GD-Link 原理图	21
图 4-23. MCU 原理图	22

表

表 2-1. 引脚分配.....	7
表 6-1. 版本历史.....	52

1. 简介

GD32H759I-EVAL 评估板使用 GD32H759IMK6 作为主控制器。评估板使用 GD-Link Mini USB 接口或者 DC-005 连接器提供 5V 电源。提供包括扩展引脚在内的及 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, USART 转 USB 接口等外设资源。更多关于开发板的资料可以查看 GD32H759I-EVAL-V1.1 原理图。

2. 功能引脚分配

表 2-1. 引脚分配

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. 入门指南

评估板使用 GD-Link Mini USBH 或者 DC-005 连接器提供 5V 电源。下载程序到评估板需要使用 J-Link 或 GD-Link 工具，在选择了正确的启动方式并且上电后，LEDPWR 将被点亮，表明评估板供电正常。

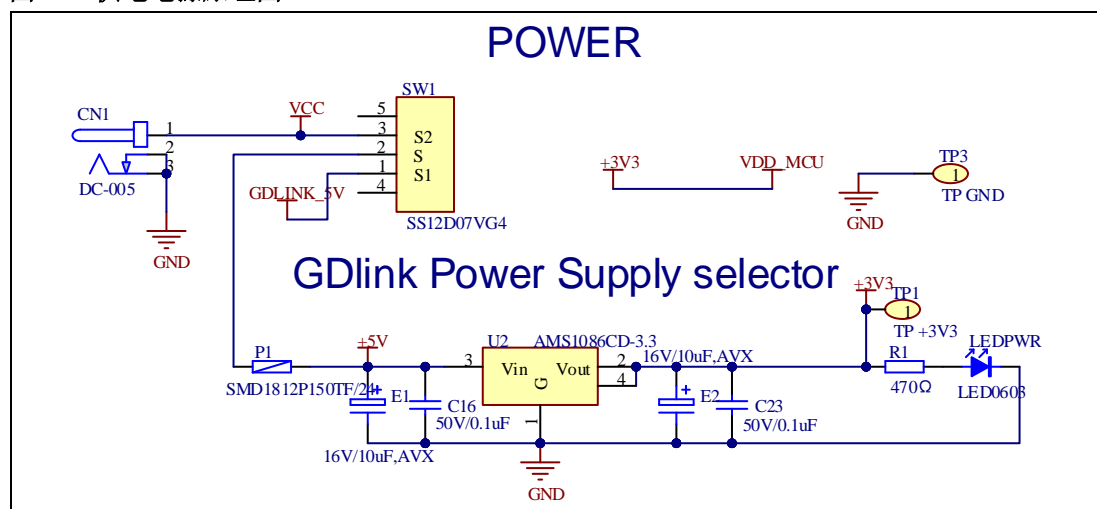
所有例程提供了 Keil 和 IAR 两个版本，Keil 版的工程是基于 Keil MDK-ARM 5.29 uVision5 创建的，IAR 版的工程是基于 IAR Embedded Workbench for ARM 8.32.1 创建的。在使用过程中有如下几点需要注意：

- 1、使用 Keil uVision5 打开工程，安装 GigaDevice.GD32H7xx_DFP.1.0.0.pack，以加载相关文件。
- 2、如果使用 IAR 打开工程，安装 IAR_GD32H7xx_ADDON_1.0.0.exe，以加载相关文件。

4. 硬件设计概述

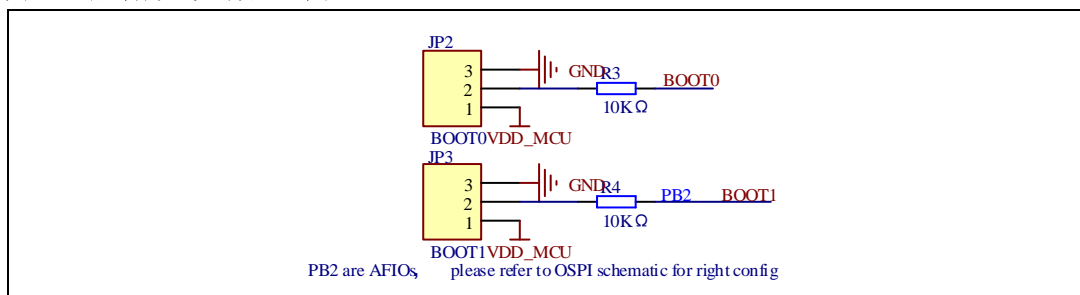
4.1. 供电电源

图4-1. 供电电源原理图



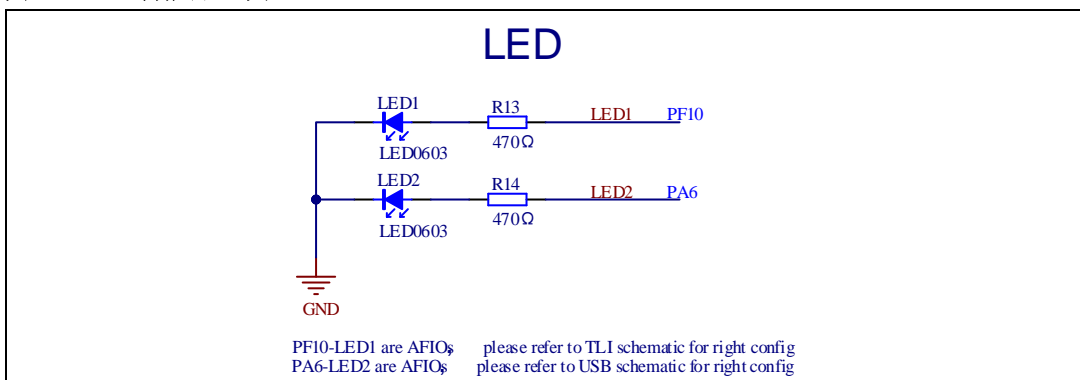
4.2. 启动方式选择

图4-2. 启动方式选择原理图



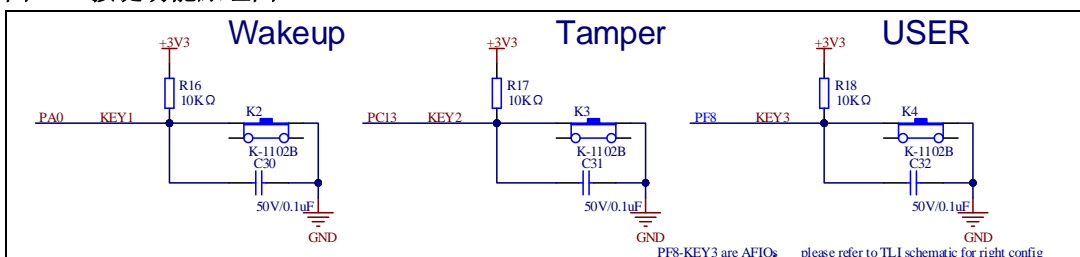
4.3. LED 指示灯

图4-3. LED功能原理图



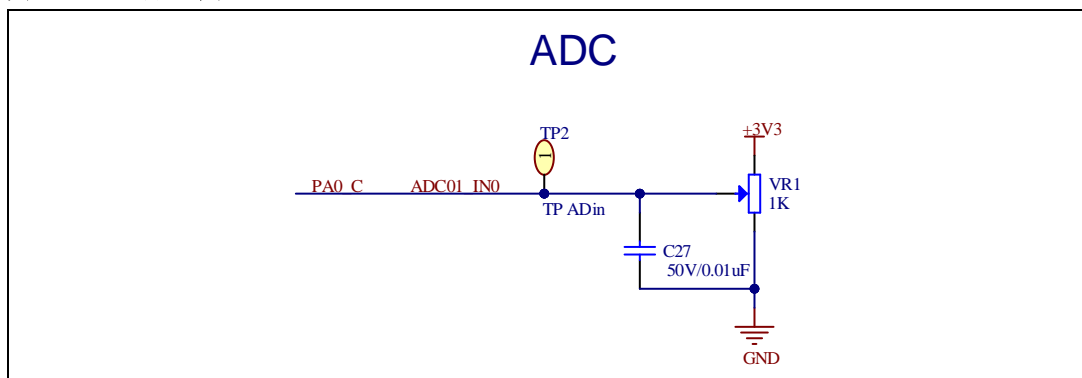
4.4. 按键

图4-4. 按键功能原理图



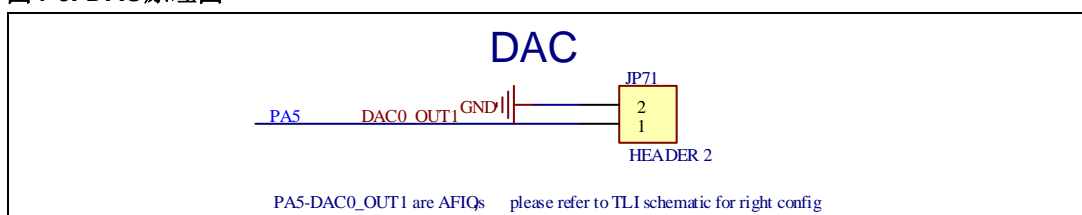
4.5. ADC

图4-5. ADC原理图



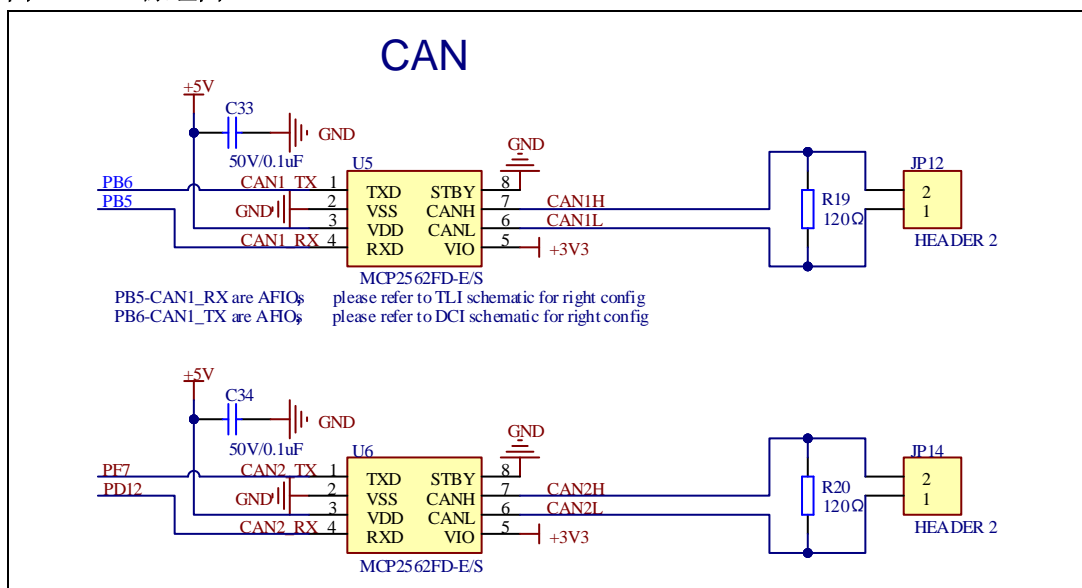
4.6. DAC

图4-6. DAC原理图



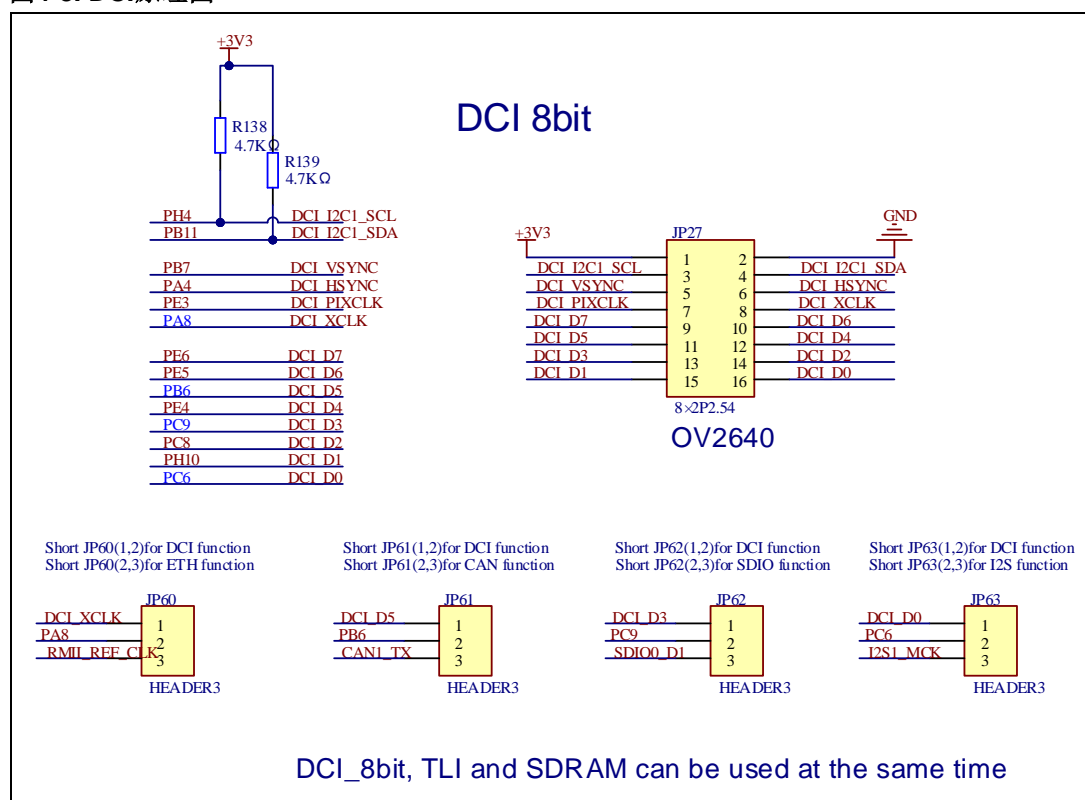
4.7. CAN

图4-7. CAN原理图



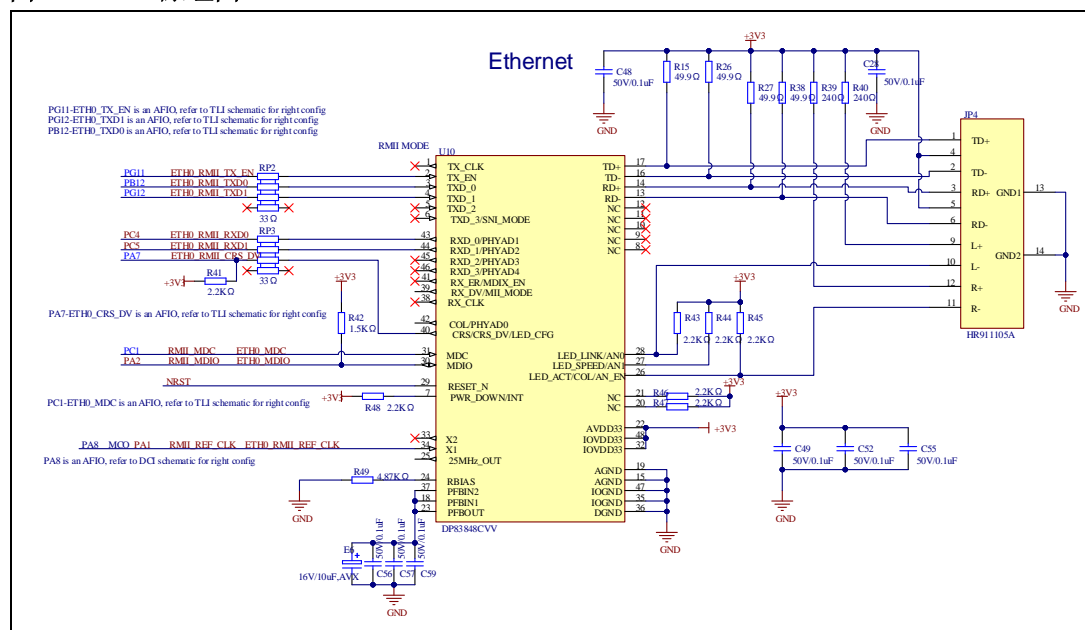
4.8. DCI

图4-8. DCI原理图



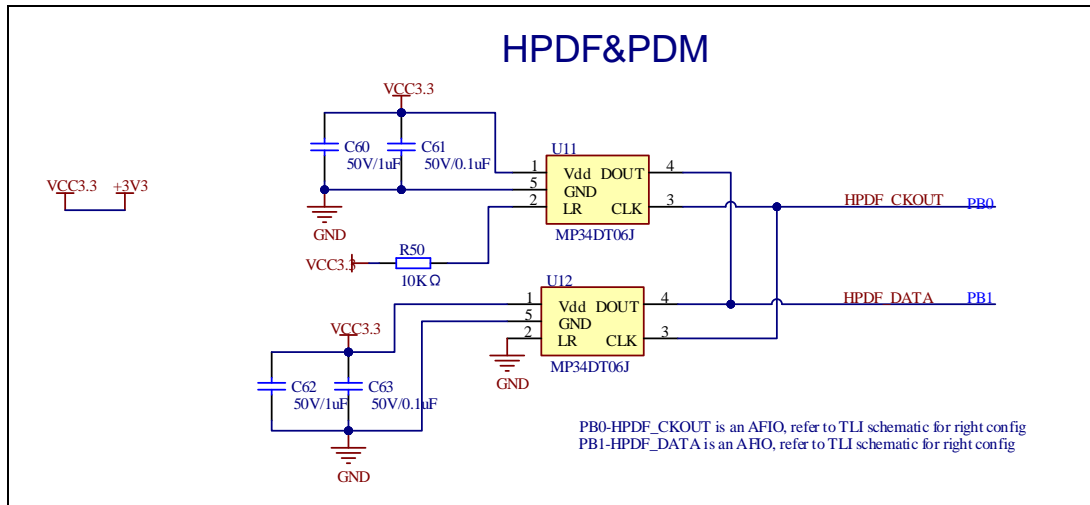
4.9. ENET

图4-9. ENET原理图



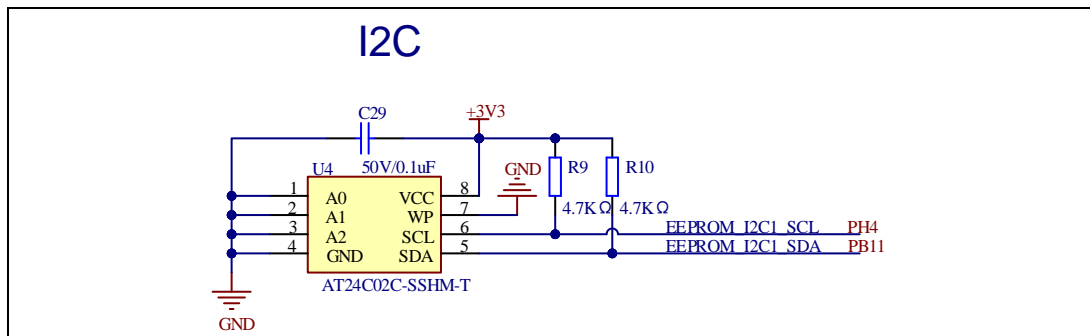
4.10. HPDF

图4-10. HPDF原理图



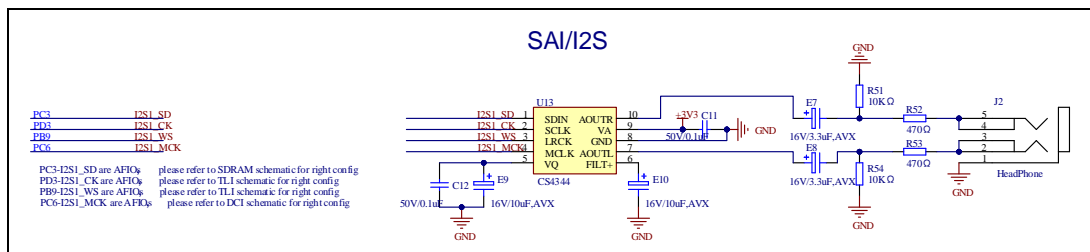
4.11. I2C

图4-11. I2C原理图



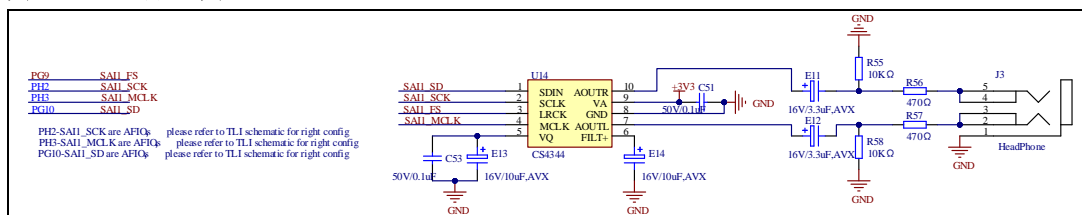
4.12. I2S

图4-12. I2S原理图



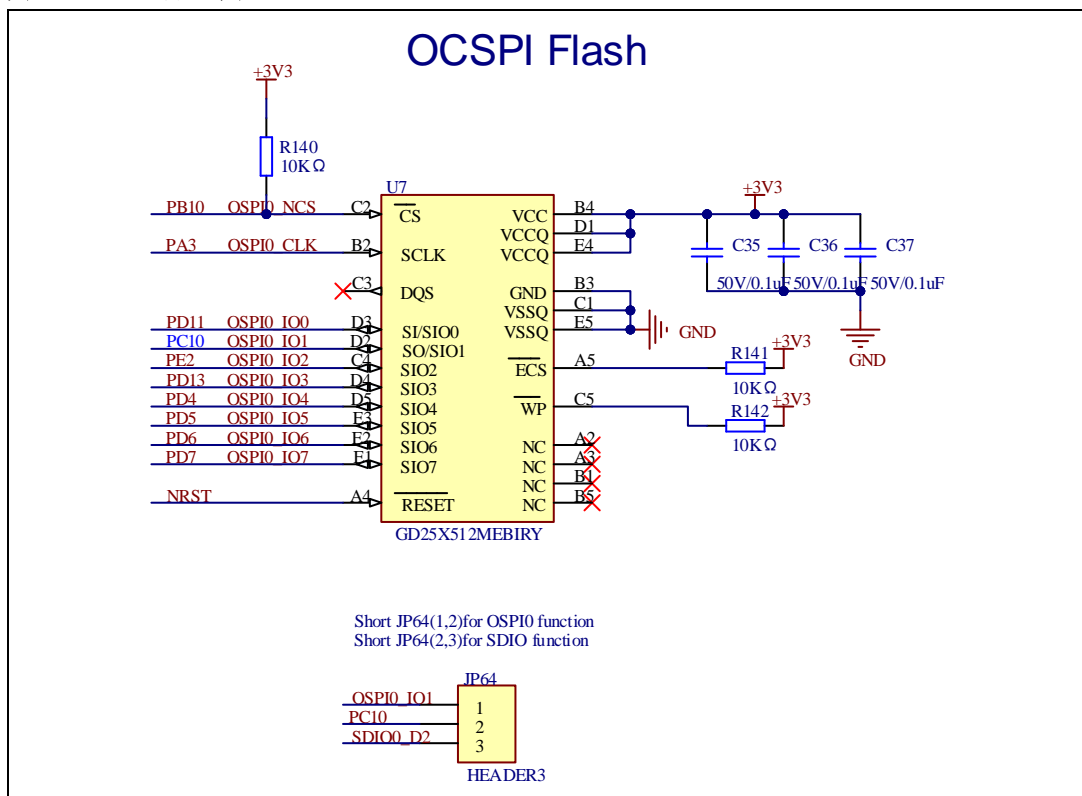
4.13. SAI

图4-13. SAI原理图



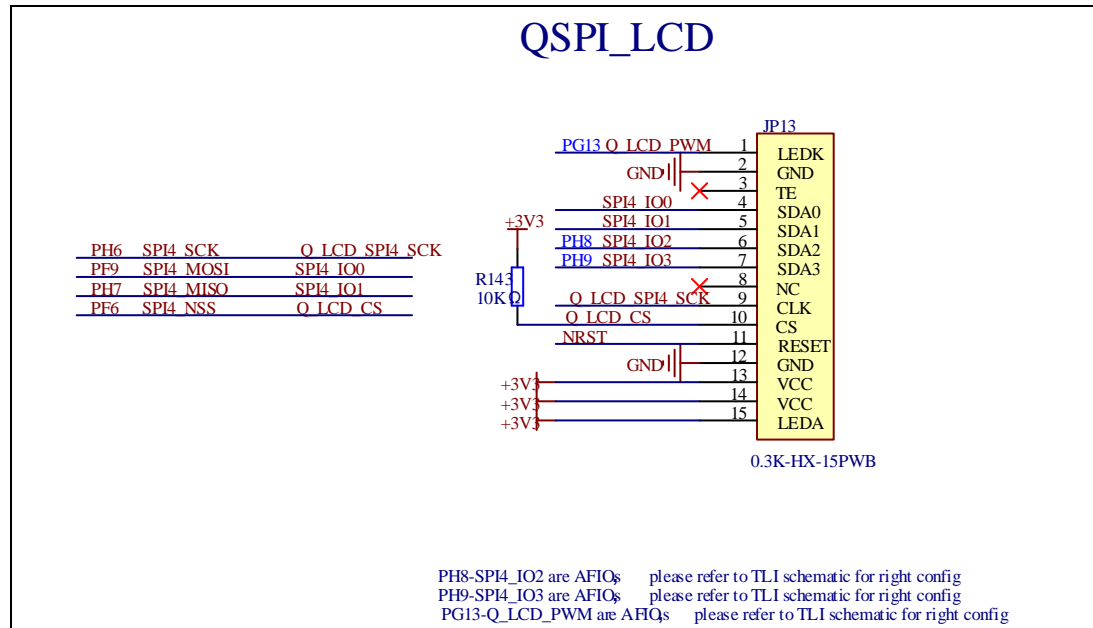
4.14. OSPI

图4-14. OSPI原理图



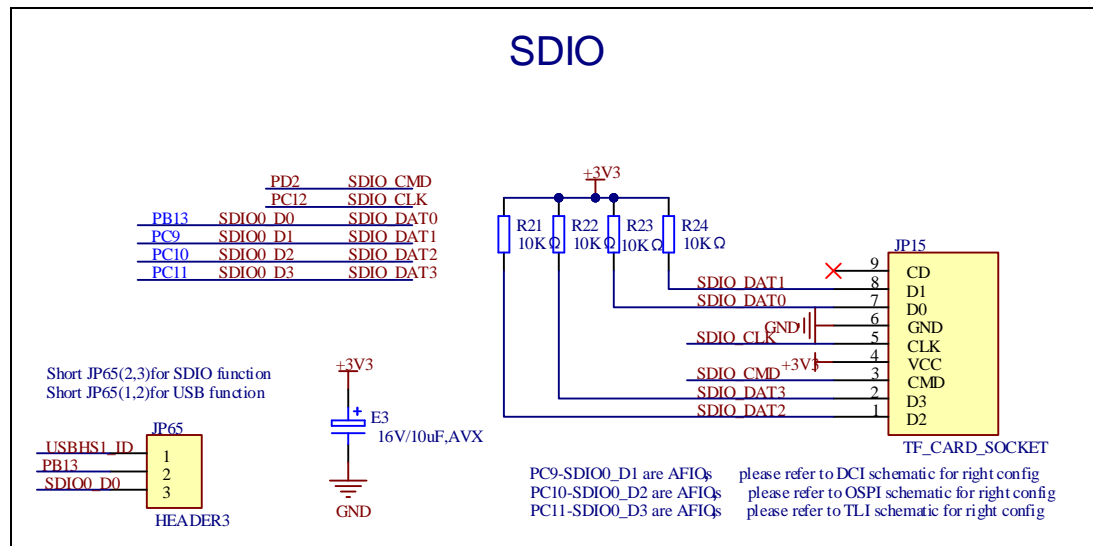
4.15. SPI_LCD

图4-15. SPI_LCD原理图



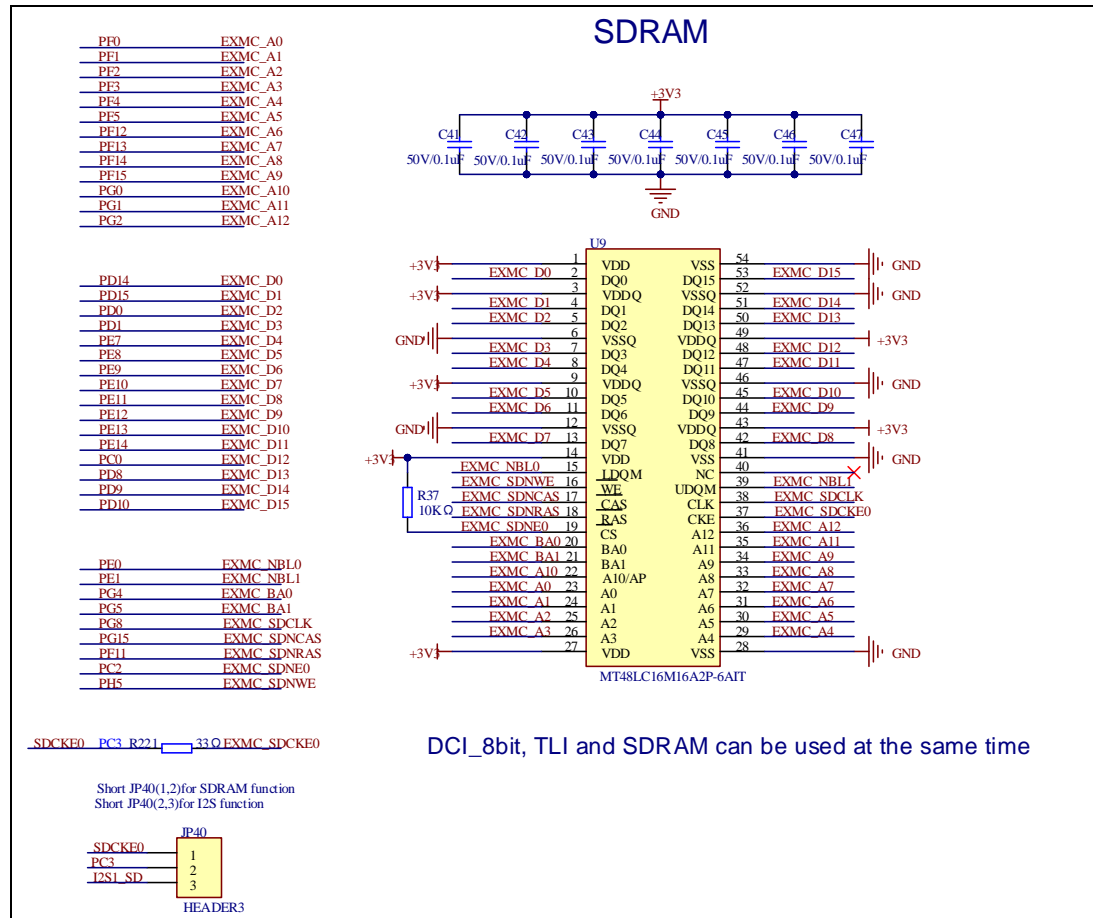
4.16. SDIO

图4-16. SDIO原理图



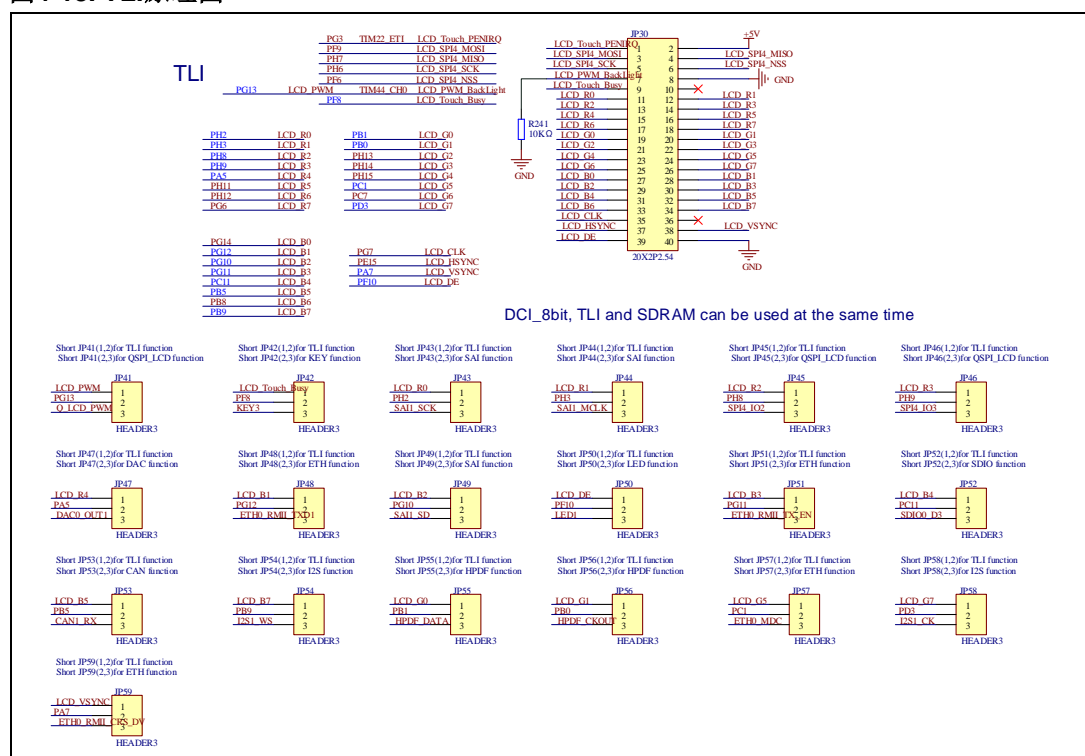
4.17. SDRAM

图4-17. SDRAM原理图



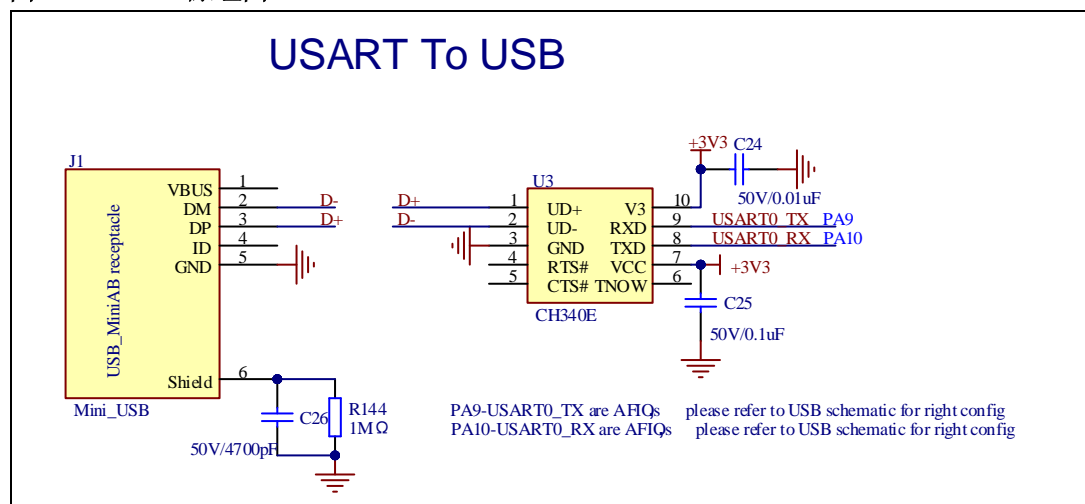
4.18. TLI_LCD

图4-18. TLI原理图



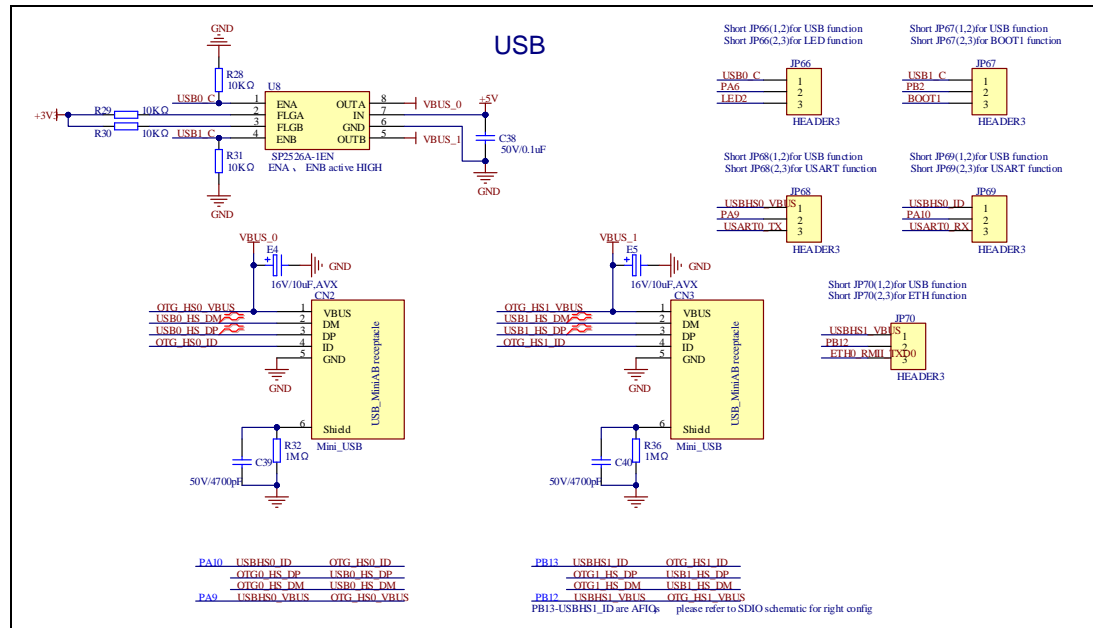
4.19. USART

图4-19.USART原理图



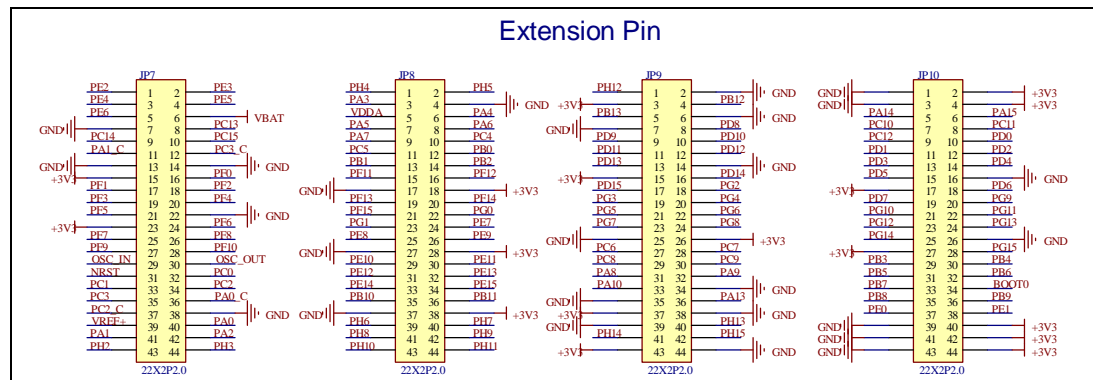
4.20. USB

图4-20. USB原理图



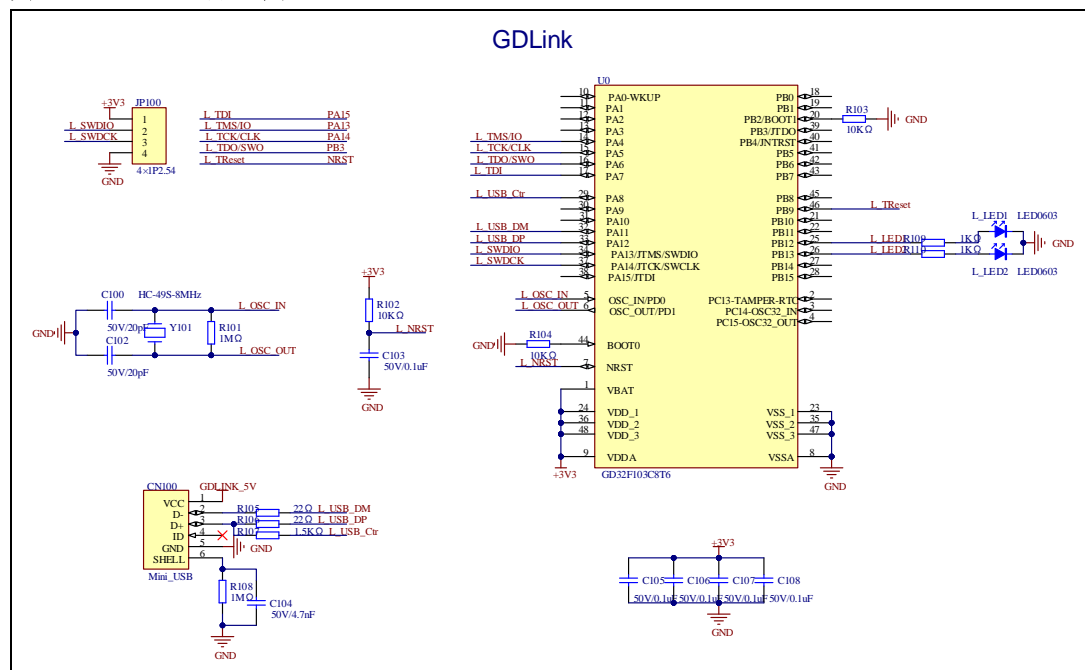
4.21. Extension

图4-21. Extension原理图



4.22. GD-Link

图4-22. GD-Link原理图



5. 例程使用指南

5.1. GPIO 流水灯

5.1.1. DEMO 目的

这个例程包括了 GD32 MCU 的以下功能：

- 学习使用 GPIO 控制 LED；
- 学习使用 SysTick 产生 1ms 的延时。

GD32H759I-EVAL-V1.1 开发板上有 2 个 LED。LED1，LED2 通过 GPIO 控制着。这个例程将讲述怎么点亮 LED。

5.1.2. DEMO 执行结果

下载程序<01_GPIO_Running_LED>到开发板上，LED1，LED2 将以流水的状态改变，然后循环重复整个过程。

5.2. GPIO 按键轮询模式

5.2.1. DEMO 目的

这个例程包括了 GD32 MCU 的以下功能：

- 学习使用 GPIO 控制 LED 和按键；
- 学习使用 SysTick 产生 1ms 的延时。

GD32H759I-EVAL-V1.1 开发板有四个按键和两个 LED。其中，四个按键是 Reset 按键，Tamper 按键，Wakeup 按键，User 按键；LED1 和 LED2 可通过 GPIO 控制。

这个例程讲述如何使用 Tamper 按键控制 LED2。当按下 Tamper 按键，将检测 IO 端口的输入值，如果输入为低电平，将等待延时 100ms。之后，再次检测 IO 端口的输入状态。如果输入仍然为低电平，表明按键成功按下，翻转 LED2 的输出状态。

5.2.2. DEMO 执行结果

下载程序<02_GPIO_Key_Polling_mode>到开发板上，按下 Tamper 按键，LED2 将会点亮，再次按下 Tamper 按键，LED2 将会熄灭。

5.3. EXTI 按键中断模式

5.3.1. DEMO 目的

这个例程包括了 GD32 MCU 的以下功能：

- 学习使用 GPIO 控制 LED 和按键
- 学习使用 EXTI 产生外部中断

GD32H759I-EVAL 开发板有四个按键和两个 LED。其中，四个按键是 Reset 按键，Tamper 按键，Wakeup 按键，User 按键；LED1 和 LED2 可通过 GPIO 控制。

这个例程讲述如何使用 EXTI 外部中断线控制 LED2。当按下 Tamper 按键，将产生一个外部中断，在中断服务函数中，应用程序翻转 LED2 的输出状态。

5.3.2. DEMO 执行结果

下载程序<03_EXTI_Key_Interrupt_mode>到开发板。启动后，LED2 闪烁一次，按下 Tamper 按键，LED2 将会点亮，再次按下 Tamper 按键，LED2 将会熄灭。

5.4. 串口打印

5.4.1. DEMO 目的

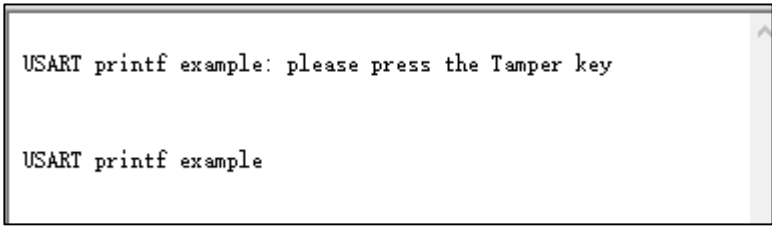
这个例程包括了 GD32 MCU 的以下功能：

- 学习使用 GPIO 控制 LED
- 学习将 C 库函数 Printf 重定向到 USART

5.4.2. DEMO 执行结果

下载程序<04_USART_Printf>到开发板，将串口线连到开发板的 USART 上。首先，所有灯亮灭 2 次用于测试。然后 USART 将输出“USART printf example: please press the Tamper key”到超级终端。按下按键 Tamper 键，串口继续输出“USART printf example”。

超级终端输出的信息如下图所示：



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

USART printf example
```

5.5. 串口中断收发

5.5.1. DEMO 目的

这个例程包括了 GD32 MCU 的以下功能：

- 学习使用串口发送和接收中断与超级终端之间的通信

5.5.2. DEMO 执行结果

下载程序< 05_USART_HyperTerminal_Interrupt >到开发板，将串口线连到开发板的 USART 上。首先，所有灯亮灭一次用于测试。然后 USART 将输出数组 tx_buffer 的内容（从 0x00 到 0xFF）到支持 hex 格式的超级终端并等待接收由超级终端发送的 BUFFER_SIZE 个字节的数据。MCU 将接收到的超级终端发来的数据存放在数组 rx_buffer 中。在发送和接收完成后，将比较 tx_buffer 和 rx_buffer 的值，如果结果相同，LED1，LED2 轮流闪烁；如果结果不相同，LED1，LED2 一起闪烁。

超级终端输出的信息如下图所示：

```
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. 串口 DMA 收发

5.6.1. DEMO 目的

这个例程包括了 GD32 MCU 的以下功能：

- 学习使用串口 DMA 功能发送和接收

5.6.2. DEMO 执行结果

下载程序< 06_USART_DMA >到开发板，将串口线连到开发板的 USART 上。首先，USART 将输出“USART DMA interrupt receive and transmit example, please input 32 bytes:”并等待接收由超级终端发送 32 个字节的数据。MCU 接收到数据后，串口将接收到的数据继续输出到超级终端。

超级终端输出的信息如下图所示：



5.7. ADC 温度传感器_Vrefint

5.7.1. DEMO 目的

这个例程包括了 GD32 MCU 的以下功能：

- 学习使用 ADC 将模拟量转换成数字量
- 学习如何获取 ADC 内部通道 18（温度传感器通道）、内部通道 19（内部参考电压 Vrefint 通道）

5.7.2. DEMO 执行结果

下载<07_ADC_Temperature_Vrefint>至开发板并运行。将开发板的 USART 口连接到电脑，打开电脑串口软件。当程序运行时，串口软件会显示温度和内部参考电压值。

注意：由于温度传感器存在偏差，如果需要测量精确的温度，应该使用一个外置的温度传感器来校准这个偏移错误。

```
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 和 ADC1 跟随模式

5.8.1. DEMO 目的

这个例程包括了 GD32 MCU 的以下功能：

- 学习使用 ADC 将模拟量转换成数字量
- 学习 ADC0 和 ADC1 工作在跟随模式

5.8.2. DEMO 执行结果

下载<08_ADC0_ADC1_Follow_up_mode>至开发板并运行。将开发板的 USART 口连接到电脑，打开电脑串口软件。PA4 引脚接外部电压，PA0_C 在开发板上接 VR1。

TIMER1_CH1 作为 ADC0 和 ADC1 的触发源。当 TIMER1_CH1 的上升沿到来，ADC0 立即启动，经过几个 ADC 时钟周期后，ADC1 启动。ADC0 和 ADC1 的值通过 DMA 传送给 adc_value[0]和 adc_value[1]。

当 TIMER1_CH1 的第一个上升沿到来，ADC0 转换的 PA0_C 引脚的电压值存储到 adc_value[0]的低半字，经过几个 ADC 时钟周期后，ADC1 转换的 PA4 引脚的电压值存储到 adc_value[0]的高半字。当 TIMER1_CH1 的第二个上升沿到来，ADC0 转换的 PA4 引脚的电压值存储到 adc_value[1]的低半字，经过几个 ADC 时钟周期后，ADC1 转换的 PA0_C 引脚的电压值存储到 adc_value[1]的高半字。

当程序运行时，串口软件会显示 adc_value[0]和 adc_value[1]的值。

```
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 0x20E91F0B

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

5.9. ADC0 和 ADC1 规则并行模式

5.9.1. DEMO 目的

这个例程包括了 GD32 MCU 的以下功能：

- 学习使用 ADC 将模拟量转换成数字量
- 学习 ADC0 和 ADC1 工作在规则并行模式

5.9.2. DEMO 执行结果

下载<09_ADC0_ADC1_Regular_Parallel_mode>至开发板并运行。将开发板的 USART 口连接到电脑，打开电脑串口软件。PA4 引脚接外部电压，PA0_C 在开发板上接 VR1。

TIMER1_CH1 作为 ADC0 和 ADC1 的触发源。当 TIMER1_CH1 的上升沿到来，ADC0 和 ADC1 会立即启动，并行转换规则组通道。ADC0 和 ADC1 的值通过 DMA 传送给 `adc_value[0]` 和 `adc_value[1]`。

当 TIMER1_CH1 的第一个上升沿到来，ADC0 转换的 PA0_C 引脚的电压值存储到 `adc_value[0]` 的低半字，并且 ADC1 转换的 PA4 引脚的电压值存储到 `adc_value[0]` 的高半字。当 TIMER1_CH1 的第二个上升沿到来，ADC0 转换的 PA4 引脚的电压值存储到 `adc_value[1]` 的低半字，并且 ADC1 转换的 PA0_C 引脚的电压值存储到 `adc_value[1]` 的高半字。

当程序运行时，串口软件会显示 `adc_value[0]` 和 `adc_value[1]` 的值。

```
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 输出电压值

5.10.1. DEMO 目的

这个例程包括了 GD32 MCU 的以下功能：

- 学习使用 DAC 在 DAC 输出端生成电压

5.10.2. DEMO 执行结果

下载程序<10_DAC_Output_Voltage_Value>至评估板并运行。将数字量设置为 0x7FF0，它的转换值应该为 1.65V ($V_{REF}/2$)，使用电压表测量 PA5 引脚 DAC_OUT1 引脚，得知其值为 1.65V。也可通过示波器观察输出波形。

5.11. I2C 访问 EEPROM

5.11.1. DEMO 目的

这个例程包括了 GD32 MCU 的以下功能：

- 学习使用 I2C 模块的主机发送模式
- 学习使用 I2C 模块的主机接收模式
- 学习读写带有 I2C 接口的 EEPROM

5.11.2. DEMO 执行结果

下载程序<11_I2C_EEPROM >到开发板上。将开发板的 USART 口连接到电脑，通过超级终端显示打印信息。

程序首先从 0x00 地址顺序写入 256 字节的数据到 EEPROM 中，并打印写入的数据，然后程序又从 0x00 地址处顺序读出 256 字节的数据，最后比较写入的数据和读出的数据是否一致，如果一致，串口打印出“I2C-AT24C02 test passed!”，同时开发板上的两个 LED 灯开始顺序闪烁，否则串口打印出“Err:data read and write aren't matching.”，同时两个 LED 全亮。

通过串口输出的信息如下图所示。

```

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 音频播放

5.12.1. DEMO 目的

这个例程包括了 GD32 MCU 的以下功能：

- 学习使用 I2S 接口输出音频数据
- 学习使用 HPDF 接口处理 PDM 数据

GD32H759I-EVAL-V1.1 开发板集成了 HPDF 和 I2S 模块，JP55 / JP56 跳线帽跳到 HPDF，JP40/JP54/JP58/JP63 跳线帽跳到 I2S，两个模块可以相互配合，播放麦克风的音频信号。这个例程演示了通过开发板的 HPDF 采集双通道的音频数据，并使用 I2S 接口实现双通道播

放的流程。

5.12.2. DEMO 执行结果

下载程序<12_HPDI_F2S_Audio>到开发板并运行，插上耳机可听到麦克风传入的声音。

5.13. HPDI_SAI 音频播放

5.13.1. DEMO 目的

这个例程包括了 GD32 MCU 的以下功能：

- 学习使用 SAI 接口输出音频数据
- 学习使用 HPDI 接口处理 PDM 数据

GD32H759I-EVAL-V1.1 开发板集成了 HPDI 和 SAI 模块，JP55 / JP56 跳线帽跳到 HPDI，JP34 / JP43 / JP49 跳线帽跳到 SAI，两个模块可以相互配合，播放麦克风的音频信号。这个例程演示了通过开发板的 HPDI 采集双通道的音频数据，并使用 SAI 接口实现双通道播放的流程。

5.13.2. DEMO 执行结果

下载程序<13_HPDI_SAI_Audio>到开发板并运行，插上耳机可听到麦克风传入的声音。

5.14. SPI 四线 LCD

5.14.1. DEMO 目的

这个例程包括了 GD32 MCU 的以下功能：

- 学习使用 SPI 模块带有的单 / 四线制功能对 GC9B71 LCD 进行控制指令 / 像素数据传输。

GD32H759I-EVAL-V1.1 开发板上集成的 SPI 模块可以和外部的 SPI LCD 设备进行通信。GC9B71 为 SPI 接口，320X386 分辨率，RGB565 的 LCD，支持单线制指令/地址/参数传输，四线制像素数据传输，最大传输速率为 50MHz。

5.14.2. DEMO 执行结果

将 JP41, JP45, JP46 跳线到 QSPI，将 LCD 屏接到 JP13 上。下载程序<14_SPI_Quad_LCD>到开发板，通过观察 LCD 屏可观察运行情况

下图为实验结果：



5.15. OSPI 八线 Flash

5.15.1. DEMO 目的

这个例程包括了 GD32 MCU 的以下功能：

- 学习使用 OSPI 模块读写带有 Octal-SPI 接口的 Nor Flash

GD32H759I-EVAL-V1.1 开发板上集成的 OSPI 模块可以和外部的 Nor Flash 设备进行通信。SPI Nor Flash 为 512Mbit 的 Flash 存储芯片 GD25X512ME，该芯片支持标准 SPI 和 8 线 SPI 的读写指令。

5.15.2. DEMO 执行结果

把电脑串口线连接到开发板的 USART 口，设置超级终端（HyperTerminal）软件波特率为 115200，数据位 8 位，停止位 1 位。同时，将 JP68，JP69 跳线到 USART，将 JP50，JP66 跳线到 LED，将 JP64 跳线到 OSPI。

下载程序<15_OSPI_Octal_Flash>到开发板上，通过超级终端可观察运行状况，会显示 Flash 的 ID 号，写入 tx_buffer1，tx_buffer2，tx_buffer3 中的数据并读出。然后比较写入的数据和读出的数据是否一致。如果一致，串口会打印出成功的信息，否则，串口打印出失败的信息。最后，两个 LED 循环点亮。

下图是实验结果图：

```
#####
```

```
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 目的

这个例程包括了 GD32 MCU 的以下功能:

- 学习使用 EXMC 控制 SDRAM

5.16.2. DEMO 执行结果

GD32H759I-EVAL 开发板使用 EXMC 模块来控制 SDRAM。下载程序<16_EXMC_SDRAM>

到开发板。这个例程演示 EXMC 对 SDRAM 的读写操作，最后会把读写的操作进行比较，如果数据一致，点亮 LED1，否则点亮 LED2。超级终端输出信息如下：

```
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 与深度睡眠模式

5.17.1. DEMO 目的

这个例程包括了 GD32 MCU 的以下功能：

- 学习使用 EXMC 控制 SDRAM
- 学习使用深度睡眠模式

5.17.2. DEMO 执行结果

GD32H759I-EVAL 开发板使用 EXMC 模块来控制 SDRAM。下载程序 <17_EXMC_SDRAM_DeepSleep> 到开发板。这个例程演示怎样在深度睡眠模式下使用 SDRAM。首先，MCU 工作在正常模式，SDRAM 自刷新的时钟由 MCU 提供，此时把指定的数据写入 SDRAM。然后使 MCU 进入深度睡眠模式并点亮 LED1，此时 SDRAM 的自刷新时钟由自己提供。最后按下 WAKEUP 按键唤醒 MCU，并读取相应数据进行比较。如果数据一致，点亮 LED2，否则熄灭 LED2。超级终端输出如下：

```

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. SD 卡测试

5.18.1. DEMO 目的

这个例程包括了 GD32 MCU 的以下功能：

- 学习使用 SDIO 单个数据块或多个数据块读写操作
- 学习使用 SDIO 对 SD 卡进行擦除、上锁和解锁操作

GD32H759I-EVAL 开发板有一个 SDIO 接口，它定义了 SD/SD I/O / eMMC 卡主机接口。这个例程讲述了如何使用 SDIO 接口来操作 SD 卡。

5.18.2. DEMO 执行结果

保证 GD32H759I-EVAL 开发板的 JP59/Jp60/Jp62 跳线帽跳到 SDIO，JP68/Jp69 跳线帽跳到 USART，下载<18_SDIO_SDCardTest>至评估板并运行。将开发板的 USART 口连接到电脑，打开超级终端。所有的 LED 灯先亮灭一次用于测试目的。然后初始化卡并打印卡的相关信息。接着再测试单块操作、上锁/解锁卡操作、擦除操作和多块操作。如果发生错误，打印错误信息并点亮 LED1 和 LED2。否则，熄灭所有 LED。

取消宏 DATA_PRINT 的注释，可以打印数据信息。通过选择不同的宏，可以设置不同的总线模式（1-bit 或 4-bit）、不同的速度模式（默认速度模式和高速模式）和数据传输模式（轮询模式或 DMA 模式）。

串口输出如下图所示：

```
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 网络通信

5.19.1. DEMO 目的

这个例程包括了 GD32 MCU 的以下功能：

- 学习使用 CAN 实现两个节点之间的通信；
- 学习使用 USART 模块与上位机进行通讯。

5.19.2. DEMO 执行结果

下载程序<19_CAN_Network>到开发板上。将 JP12 和 JP14 的 L 引脚和 H 引脚分别相连。用跳线帽将 JP61、JP53 跳 CAN 上，将 JP66 跳到 LED 上，将 JP68、JP69 跳到 USART 上，并将串口线连到开发板的 USART。用户按下 WAKEUP 键，数据帧将通过 CAN2 发送出去同时将数据内容通过串口打印出来。当接收到数据帧时，接收到的数据通过串口打印，同时 LED2 状态翻转一次。通过串口输出的信息如下图所示。

```
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 时钟输出

5.20.1. DEMO 目的

这个例程包括了 GD32 MCU 的以下功能：

- 学习使用 GPIO 控制 LED
- 学习使用 RCU 模块的时钟输出功能
- 学习使用 USART 模块与电脑进行通讯

5.20.2. DEMO 执行结果

下载程序<20_RCU_Clock_Out>到开发板上并运行。将开发板的 USART 口连接到电脑，打开超级终端。当程序运行时，超级终端将显示初始信息。之后通过按下 TAMPER 按键可以选择输出时钟的类型，对应的 LED 灯会被点亮，并在超级终端显示选择的模式类型。测量 PA8 和 PC9 引脚，可以通过示波器观测输出时钟的频率。

串口输出如下图所示：

```

/===== 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 睡眠模式唤醒

5.21.1. DEMO 目的

这个例程包括了 GD32 MCU 的以下功能：

- 学习使用串口接收中断唤醒 PMU 睡眠模式

5.21.2. DEMO 执行结果

下载程序<21_PMU_Sleep_Wakeup>到开发板上，并将串口线连到开发板的 USART 上。板子上电后，所有 LED 都熄灭。MCU 将进入睡眠模式同时软件停止运行。当从超级终端接收到一个字节数据时，MCU 将被 USART 接收中断唤醒。所有的 LED 灯同时闪烁。

5.22. RTC 日历

5.22.1. DEMO 目的

这个例程包括了 GD32 MCU 的以下功能：

- 学习使用 RTC 模块实现日历功能
- 学习使用 USART 模块实现时间显示

5.22.2. DEMO 执行结果

下载程序<22_RTC_Calendar>到开发板上，使用串口线连接电脑到开发板 USART 接口，打开串口助手软件。在开发板上电后，程序需要请求通过串口助手设置时间。日历会显示在串口助手上。

5.23. TIMER 呼吸灯

5.23.1. DEMO 目的

这个例程包括了 GD32 MCU 的以下功能：

- 学习使用定时器输出 PWM 波
- 学习更新定时器通道寄存器的值

GD32H759I-EVAL-V1.1 开发板上有 2 个 LED。LED1，LED2 通过 GPIO 控制着。

5.23.2. DEMO 执行结果

使用杜邦线连接 TIMER0_CH3（PC7）到 LED1（PF10），然后下载程序<23_TIMER_Breath_LED>到开发板，并运行程序。PC7 不要用于其他外设。

可以看到 LED1 由暗变亮，由亮变暗，往复循环，就像人的呼吸一样有节奏。

5.24. TLI_IPA

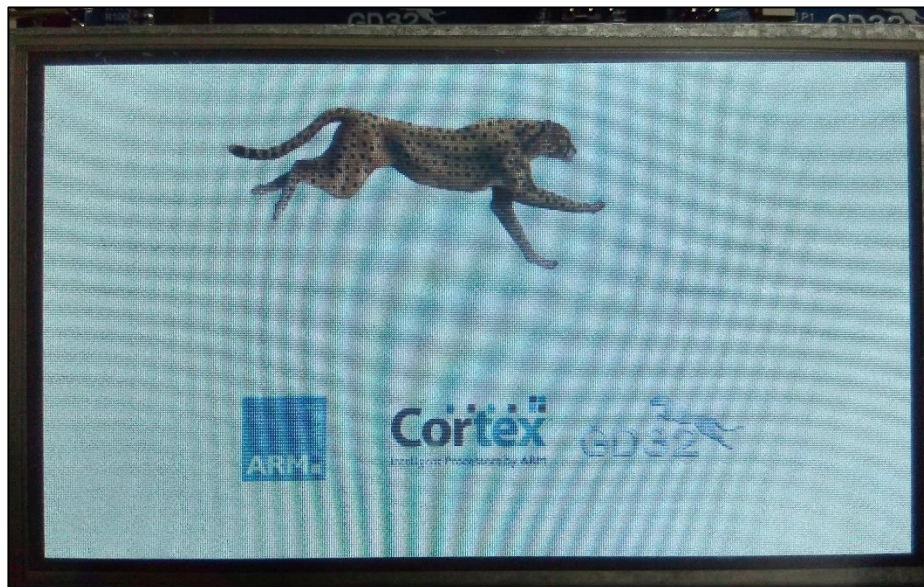
5.24.1. DEMO 目的

这个例程包括了 GD32 MCU 的以下功能：

- 学习使用 TLI 控制 LCD 显示不同的内容；
- 学习使用 IPA 处理图像数据。

5.24.2. DEMO 执行结果

将 JP41/Jp43/Jp44/Jp45/Jp46/Jp47/Jp48/Jp49/Jp50/Jp51/Jp52/Jp53/Jp54/Jp55/Jp56/Jp57/Jp58/Jp59 跳线帽跳到 LCD。下载<24_TLI_IPA>至评估板并运行。将在 LCD 上显示以 GD logo 为背景的奔跑的豹子。由于 LCD 屏消耗的电流较大，建议使用 DC-5V 供电。



5.25. DCI_OV2640

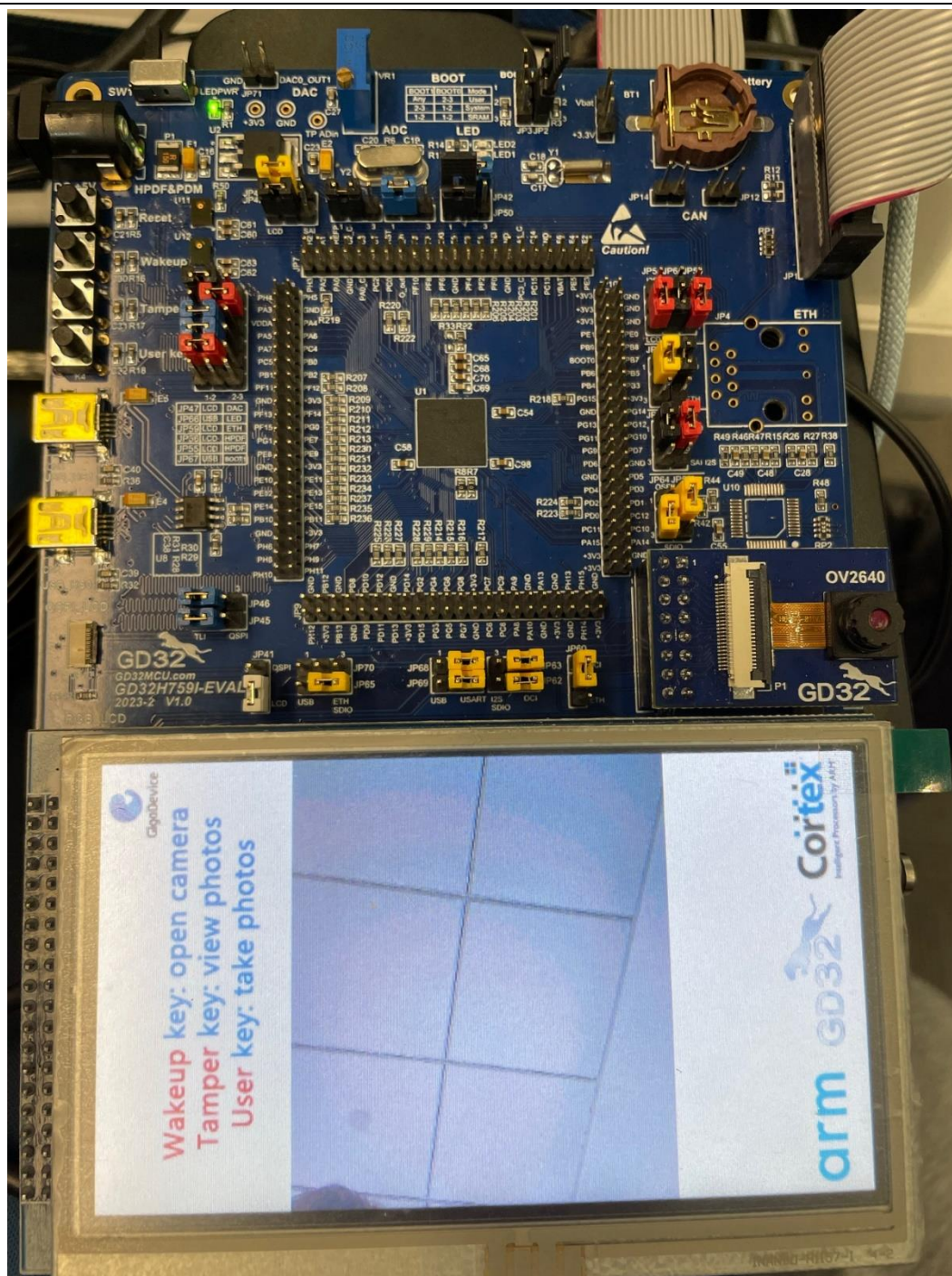
5.25.1. DEMO 目的

这个例程包括了 GD32 MCU 的以下功能：

- 学习使用 DCI 接口采集 OV2640 摄像头图像
- 学习使用 TLI 接口显示采集图像

5.25.2. DEMO 执行结果

保证 GD32H759I-EVAL-V1.1 开发板的 JP60/Jp61/Jp62/Jp63 跳线帽跳到 DCI，JP41/Jp43/Jp44/Jp45/Jp46/Jp47/Jp48/Jp49/Jp50/Jp51/Jp52/Jp53/Jp54/Jp55/Jp56/Jp57/Jp58/Jp59 跳线帽跳到 LCD，JP42 跳线帽跳到 User key，JP40 跳线帽跳到 SDRAM。将程序<25_DCI_OV2640>烧录到开发板，正确安装 LCD 显示屏和 OV2640 摄像头到开发板各接口插槽。上电后可观察到，摄像头采集的图像显示在 LCD 显示屏上，按下 User key 按键可以拍照存储，按下 Tamper 按键显示照片，按下 Wakeup 按键返回图像采集状态。



5.26. 以太网

5.26.1. FreeRTOS 上的服务器/客户端

DEMO 目的

这个例程包括了 GD32 MCU 的以下功能：

- 学习使用 Lwip 协议栈；
- 学习使用 FreeRTOS 操作系统；
- 学习使用 netconn 与 socket API 函数来处理任务；
- 学习怎样实现一个 tcp 服务器；
- 学习怎样实现一个 tcp 客户端；
- 学习怎样实现一个 udp 服务器/客户端；
- 学习使用 DHCP 来自动分配 ip 地址。

该例程是基于 GD32H759I-EVAL-V1.1 开发板，演示怎样配置以太网模块为常规描述符模式来进行收发数据包，以及如何使用 Lwip tcp / ip 协议栈来实现 ping, telnet, 服务器/客户端功能。

JP48, JP51, JP57, JP59, JP60, JP70 跳线帽必须匹配。JP68, JP69 跳线帽连到 Usart。

该例程中以太网配置为 RMII 模式，使用 25MHz 晶振，系统时钟配为 600MHz。

该例程实现了三个应用：

- Telnet 应用，开发板作为 tcp 服务器。用户可以将客户端与开发板服务器相连接，通信采用 8000 端口，在客户端界面可以看到来自服务器的回复，客户端可以发送姓名到服务器，服务器进行应答。
- tcp 客户端应用，开发板作为 tcp 客户端。用户可以将服务器与开发板客户端相连接，通信采用 10260 端口，用户从服务器发送信息给开发板，开发板将所收到的信息发回。
- udp 应用。用户可以将开发板与其他站点进行 udp 连接，使用 1025 端口通信，用户从站点发送信息给开发板，开发板将所收到的信息发回。

如果用户要使用 DHCP 功能，需在 main.h 文件中将相应的宏去屏蔽，并重新编译。该功能默认为关闭。

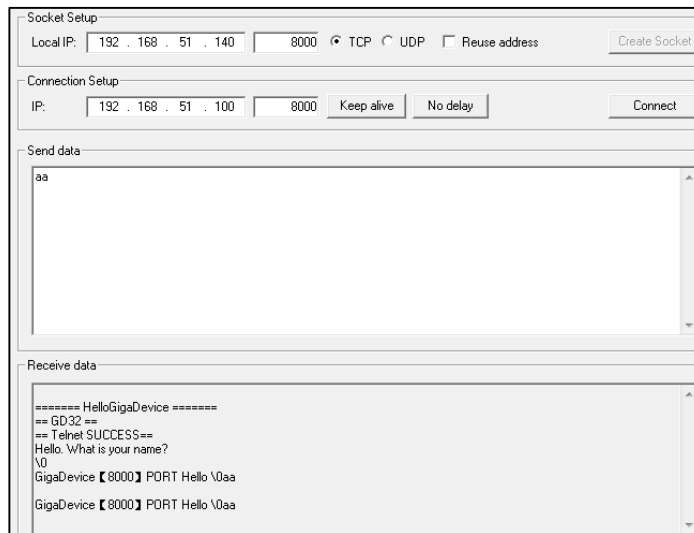
注意：

- 用户需要根据实际的网络情况在 main.h 文件中为开发板以及服务器配置 ip 地址，网络掩码和网关地址。
- USE_ENET0 和 USE_ENET1 宏定义不能同时开启。

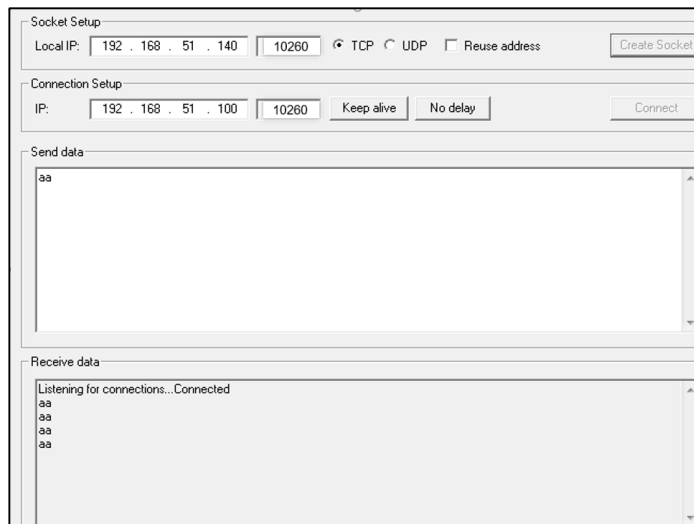
DEMO 执行结果

将例程<FreeRTOS_tcpudp>下载到开发板，LED1 每 250ms 亮一次。

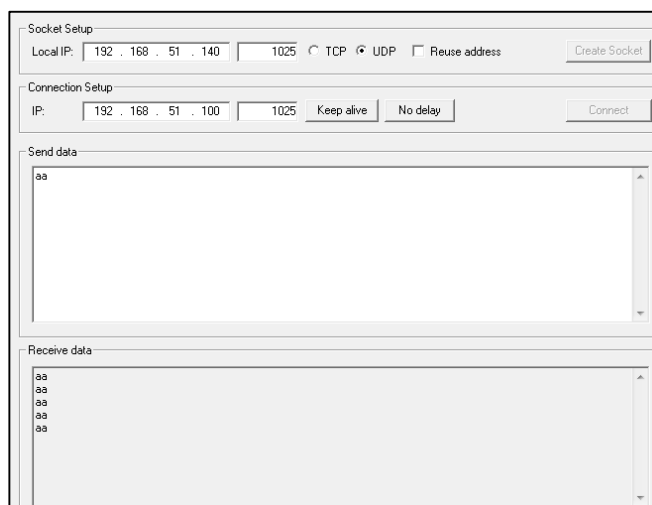
使用网络调试助手，并将电脑端配置为 tcp 客户端，端口配为 8000，连接上服务器后用户可以看到服务器的回复，在客户端发送姓名到服务器，可以看到服务器的应答：



使用网络调试助手，并将电脑端配置为 **tcp** 服务器，端口配为 **10260**，连接上客户端后在服务器端发送信息到客户端，可以看到客户端的回显应答：



使用网络调试助手，配置使用 **udp** 协议，端口配为 **1025**，连接上开发板后在电脑端发送信息到开发板，可以看到开发板的回显应答：



在 `main.h` 中打开 DHCP 功能后，并将板子与电脑都接在路由器上，用户可以通过串口调试助手看到自动分配给开发板的 ip 地址。

5.26.2. 服务器/客户端

DEMO 目的

这个例程包括了 GD32 MCU 的以下功能：

- 学习使用 Lwip 协议栈；
- 学习使用 raw API 函数来处理任务；
- 学习怎样实现一个 tcp 服务器；
- 学习怎样实现一个 tcp 客户端；
- 学习怎样实现一个 udp 服务器/客户端；
- 学习使用 DHCP 来自动分配 ip 地址；
- 学习使用轮询方式和中断方式来进行包的接收。

该例程是基于 GD32H759I-EVAL-V1.1 开发板，演示怎样配置以太网模块为常规描述符模式来进行收发数据包，以及如何使用 Lwip tcp / ip 协议栈来实现 ping, telnet, 服务器/客户端功能。

JP48, JP51, JP57, JP59, JP60, JP70 跳线帽必须匹配。JP68, JP69 跳线帽连到 Usart。

该例程中以太网配置为 RMII 模式，使用 25MHz 晶振，系统时钟配为 600MHz。

该例程实现了三个应用：

- Telnet 应用，开发板作为 tcp 服务器。用户可以将客户端与开发板服务器相连接，通信采用 8000 端口，在客户端界面可以看到来自服务器的回复，客户端可以发送姓名到服务器，服务器进行应答。
- tcp 客户端应用，开发板作为 tcp 客户端。用户可以将服务器与开发板客户端相连接，通信采用 10260 端口，用户从服务器发送信息给开发板，开发板将所收到的信息发回。如果服务器在一开始没有打开，或者在通信过程中发生了中断，当服务器再次准备好的时候，用户可以通过按 Tamper 键来重新建立客户端与服务器的连接。
- udp 应用。用户可以将开发板与其他站点进行 udp 连接，使用 1025 端口通信，用户从站点发送信息给开发板，开发板将所收到的信息发回。

默认包的接收采用在 `while(1)` 中轮询的模式，用户如果想要在中断中处理接收包，可将 `main.h` 中 `USE_ENET_INTERRUPT` 宏去屏蔽。

如果用户要使用 DHCP 功能，需在 `main.h` 文件中将相应的宏去屏蔽，并重新编译。该功能默认为关闭。

注意：

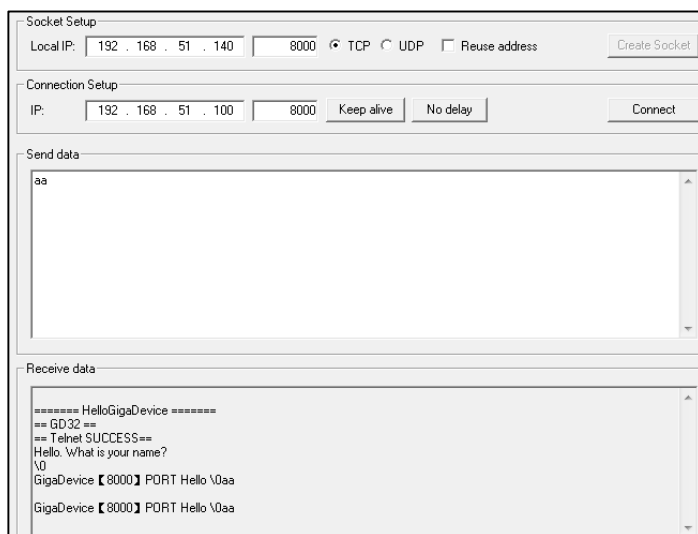
- 用户需要根据实际的网络情况在 `main.h` 文件中为开发板以及服务器配置 ip 地址，网络掩码和网关地址。
- `USE_ENET0` 和 `USE_ENET1` 宏定义不能同时开启。

DEMO 执行结果

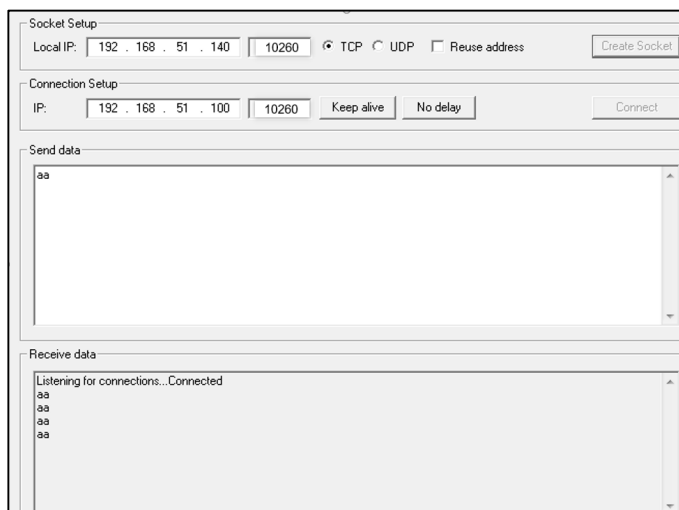
将例程 < Raw_tcpudp > 下载到开发板。

使用网络调试助手，并将电脑端配置为 tcp 客户端，端口配为 8000，连接上服务器后用户

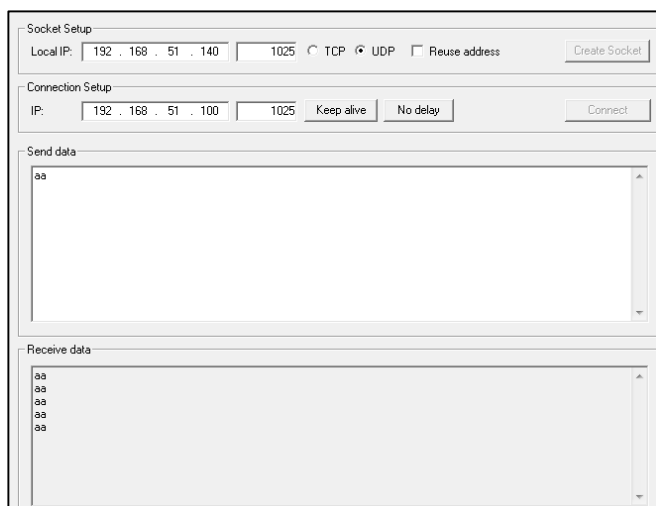
可以看到服务器的回复，在客户端发送姓名到服务器，可以看到服务器的应答：



使用网络调试助手，并将电脑端配置为 **tcp** 服务器，端口配为 **10260**，连接后，按 **Tamper** 键，在服务器端发送信息到客户端，可以看到客户端的回显应答：



使用网络调试助手，配置使用 **udp** 协议，端口配为 **1025**，连接上开发板后在电脑端发送信息到开发板，可以看到开发板的回显应答：



在 main.h 中打开 DHCP 功能后，并将板子与电脑都接在路由器上，用户可以通过串口调试助手看到自动分配给开发板的 ip 地址。

5.26.3. web 服务器

DEMO 目的

这个例程包括了 GD32 MCU 的以下功能：

- 学习使用 Lwip 协议栈；
- 学习使用 raw API 函数来处理任务；
- 学习怎样实现一个 web 服务器；
- 学习使用 web 服务器来控制 LED；
- 学习使用 web 服务器来监控开发板 V_{REFINT} 电压；
- 学习使用 DHCP 来自动分配 ip 地址；
- 学习使用轮询方式和中断方式来进行包的接收。

该例程是基于 GD32H759I-EVAL-V1.1 开发板，演示怎样配置以太网模块为常规描述符模式来进行收发数据包，以及如何使用 Lwip tcp / ip 协议栈来实现 web 服务器应用。

JP48, JP51, JP57, JP59, JP60, JP70 跳线帽必须匹配。JP68, JP69 跳线帽连到 Usart。

该例程中以太网配置为 RMII 模式，使用 25MHz 晶振，系统时钟配为 600MHz。

该例程实现了 web 服务器应用：

用户可以通过网页浏览器来访问开发板，开发板此时作为一个 web 服务器，网址是开发板的 ip 地址。web 服务器中实现了 2 个实验，一个为 LED 灯的控制，另一个为通过 ADC 实时监测开发板 V_{REFINT} 电压。

如果用户需要 DHCP 功能，可通过 main.h 中相关宏进行配置，该功能默认关闭。如果打开了该功能，用户可以使用路由器连接开发板，并由串口调试助手打印自动为开发板分配的 ip 地址，然后将手机连上路由器发的 wifi，这样手机与开发板就在一个网段了。用户可以在手机上通过浏览器访问开发板的 ip 地址，来控制开发板 LED 灯以及实时监测 Vref 电压。

默认包的接收采用在 while(1) 中轮询的模式，用户如果想要在中断中处理接收包，可将 main.h 中 USE_ENET_INTERRUPT 宏去屏蔽。

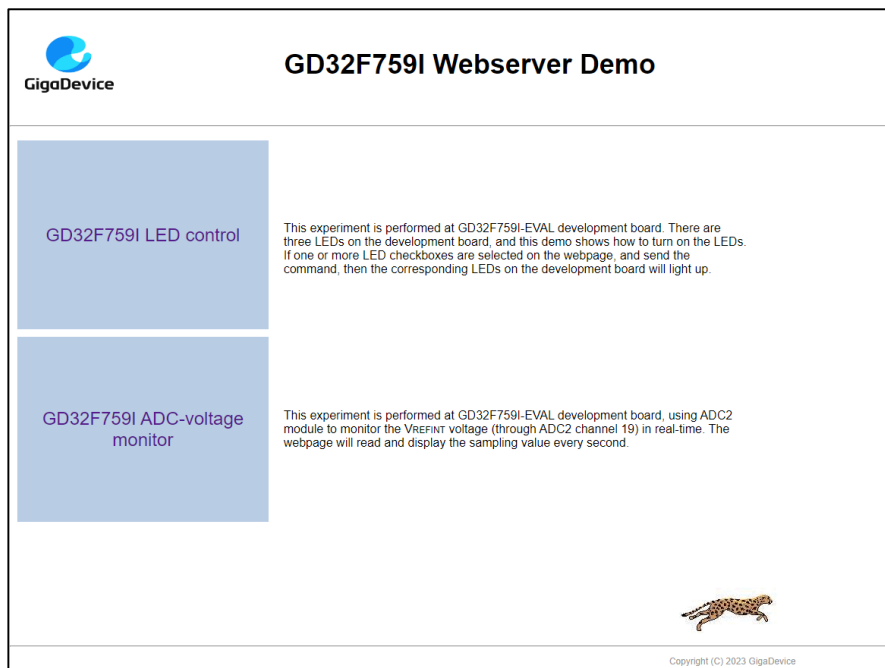
注意：

- 用户需要根据实际的网络情况在 main.h 文件中为开发板以及服务器配置 ip 地址，网络掩码和网关地址。
- USE_ENET0 和 USE_ENET1 宏定义不能同时开启。

DEMO 执行结果

将例程<Raw_webserver>下载到开发板，使用浏览器，访问开发板的 ip 地址，在网页中点击 LED 控制的链接，在新的 LED 灯控制页眉中选择要点亮的灯的复选框，并点击发送，则板上相应的 LED 将被点亮。点击 ADC 监控电压的连接，则网页将实时显示开发板所采集到的 V_{REFINT} 电压，每秒自动刷新一次。

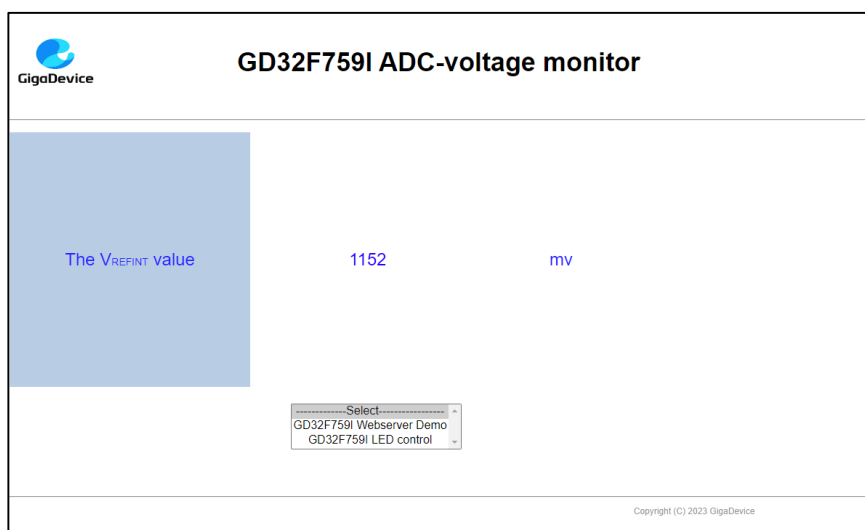
网页主页显示如下：



LED 控制页面显示如下：



ADC 检测电压页面显示如下：



在 `main.h` 中打开 DHCP 功能，使用路由器连接开发板，由串口调试助手打印自动为开发板分配的 ip 地址，然后将手机连上路由器发的 wifi。此时用户可以在手机上通过浏览器访问开发板的 ip 地址，并控制开发板。

5.27. USB 设备

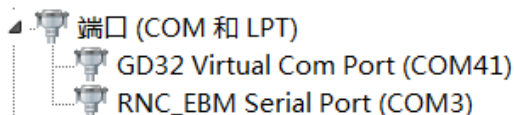
5.27.1. 虚拟串口

DEMO 目的

这个例程包括了 GD32 MCU 的以下功能：

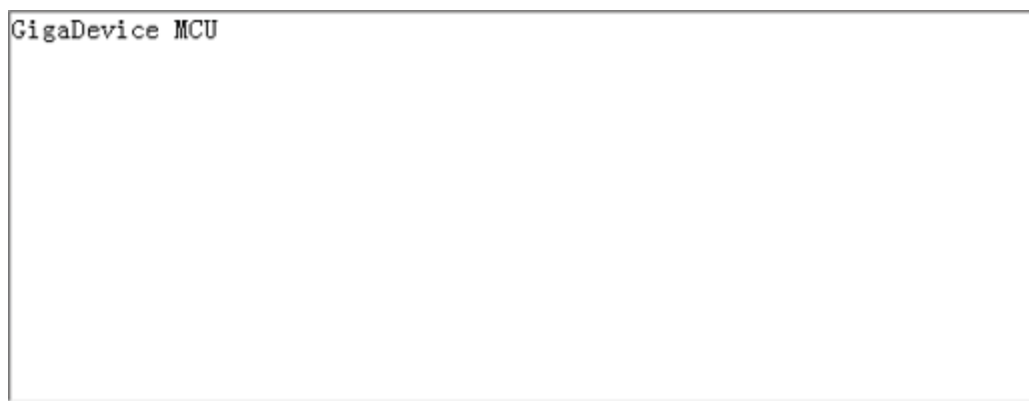
- 学习如何使用 USBFS/HS 设备
- 学习如何实现 USB CDC 设备

GD32H759I-EVAL-V1.1 开发板具有两个 USBFS/HS 接口。在本例程中，GD32H759I-EVAL-V1.1 开发板的 USBHS0 被 USB 主机枚举为一个 USB 虚拟串口，如下图所示，可在 PC 端设备管理器中看到该虚拟串口。该例程使得 USB 接口看起来像是个串口，也可以通过 USB 口回传数据。通过键盘输入某些信息，虚拟串口可以接收并显示这些信息。



DEMO 执行结果

将 < 27_USB_Device\CDC_ACM > 例程下载到开发板中，并运行。通过键盘输入某些数据，虚拟串口可以接收并显示这些数据。比如通过虚拟串口的输入框输入“GigaDevice MCU”，PC 回传这些信息给虚拟串口，并得以显示。



5.27.2. HID_键盘

DEMO 目的

这个例程包括了 GD32 MCU 的以下功能：

- 学习如何使用 USBFS/USBHS 的设备模式
- 学习如何实现 USB HID（人机接口）设备

GD32H759I-EVAL-V1.1 开发板具有四个按键、两个 USBFS/HS 接口，这四个按键分别是 Reset 按键、Wakeup 按键、User 按键和 Tamper 按键。在本例程中，GD32H759I-EVAL-V1.1 开发板的 USBHS0 被 USB 主机利用内部 HID 驱动枚举为 USB 键盘，如下图所示，USB 键盘利用 Wakeup 键、Tamper 键和用户键输出三个字符（‘b’，‘a’ 和 ‘c’）。另外，本例程支持 USB 键盘远程唤醒主机，其中 Wakeup 按键被作为唤醒源。



DEMO 执行结果

完成这些之后，将 < 27_USB_Device\HID_Keyboard > 例程下载到开发板中，并运行。按下 Wakeup 键，输出 ‘a’；按下 Tamper 键，输出 ‘b’；按下 User 键，输出 ‘c’。

可利用以下步骤所说明的方法验证 USB 远程唤醒的功能：

- 手动将 PC 机切换到睡眠模式；
- 等待主机完全进入睡眠模式；
- 按下 Wakeup 按键；
- 如果 PC 被唤醒，表明 USB 远程唤醒功能正常，否则失败。

5.28. USB 主机

5.28.1. USB HID 主机

DEMO 目的

这个例程包括了 GD32 MCU 的以下功能：

- 学习使用 USBHS 模块作为 HID 主机
- 学习 HID 主机和鼠标设备之间的操作
- 学习 HID 主机和键盘设备之间的操作

GD32H759I-EVAL-V1.1 开发板内部包含 USBFS 模块和 USBHS 模块，并且该模块可以被使用作为一个 USB 设备、一个 USB 主机或者一个 OTG 设备。该示例主要展示了如何使用 USBHS1 作为一个 USB HID 主机和外部 USB HID 设备进行通信。

DEMO 执行结果

将 < 28_USB_Host\Host_HID > 代码下载到开发板并运行。

如果一个鼠标被连入，用户将会看到鼠标枚举的信息。首先按下 User 按键，将会看到插入的设备是鼠标；然后移动鼠标，将会在串口调试助手看到鼠标坐标位置。

```
> 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.
```

如果一个键盘被连入，用户将会看到键盘枚举的信息。首先按下 **User** 按键，将会看到插入的设备是键盘；然后按下键盘按键，将会通过串口调试助手显示按键状态。

```
> 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 tape 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. USB MSC 主机

DEMO 目的

这个例程包括了 GD32 MCU 的以下功能：

- 学习使用 USBFS/USBHS 作为 MSC 主机
- 学习 MSC 主机和 U 盘之间的操作

GD32H759I-EVAL-V1.1 开发板包含 USBFS 模块和 USBHS 模块，并且这两个模块可以被用于作为 USB 设备、USB 主机或 OTG 设备。本示例主要显示如何使用 USBHS1 作为 USB MSC 主机来与外部 U 盘进行通信。

DEMO 执行结果

将 JP68/69 跳到 USART，JP70 跳到 USB。将 OTG 电缆线插入到 USB 接口，然后将 <28_USB_Host\Host_MSC> 代码下载到开发板并运行。

如果 U 盘被连入，用户将会在串口助手上看到 U 盘枚举信息。

首先会看到 U 盘信息；之后按下 **Tamper** 按键将会看到 U 盘根目录内容；然后按下 **Wakeup** 按键将会向 U 盘写入文件；最后用户将会看到 MSC 主机示例结束的信息。

```

++++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. 版本历史

表 6-1. 版本历史

版本号	说明	日期
1.0	初稿发布	2023 年 03 月 31 日

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.