GigaDevice Semiconductor Inc.

GD32103B-EVAL 评估板
用户指南
V2.1

目录

表

1 简介

GD32103B-EVAL 评估板使用 GD32F103VBT6 作为主控制器。评估板使用 Mini USB 接口提供 5V 电源。提供包括扩展引脚在内的及 SWD, Reset, Boot, User button key, LED, CAN, I2C, USART, RTC, LCD, SPI, ADC, EXMC, USBD, GD-Link 等外设资源。更多关于开发板的资料可以查看 GD32103B-EVAL-V1.2 原理图。

2 功能引脚分配

表 1. 引脚分配表

功能	引脚	描述
LED	PC6	LED2
	PC7	LED3
	PC8	LED4
	PC9	LED5
RESET		K1-Reset
KEY	PA0	K2-Wakeup
	PB9	K3-User key
	PC13	K4-Tamper
BEEP	PB8	BEEP_CTL
USART0	PA9	USART0_TX
	PA10	USART0_RX
USART1	PD5	USART1_TX
	PD6	USART1_RX
	PD3	USART1_CTS
	PD4	USART1_RTS
ADC	PC4	ADC01_IN14
I2C	PB6	I2C0_SCL
	PB7	I2C0_SDA
SPI	PA5	SPI0_SCK
	PA6	SPI0_MISO
	PA7	SPI0_MOSI
	PA4	SPI0_CS
CAN	PD0	CAN0_RX
	PD1	CAN0_TX
SRAM	PD14	EXMC_D0
	PD15	EXMC_D1
	PD0	EXMC_D2
	PD1	EXMC_D3
	PE7	EXMC_D4
	PE8	EXMC_D5

	PE9	EXMC_D6
	PE10	EXMC_D7
	PE11	EXMC_D8
	PE12	EXMC_D9
	PE13	EXMC_D10
	PE14	EXMC_D11
	PE15	EXMC_D12
	PD8	EXMC_D13
	PD9	EXMC_D14
	PD10	EXMC_D15
	PD11	EXMC_A16
	PD12	EXMC_A17
	PD13	EXMC_A18
	PD4	EXMC_NOE
	PD5	EXMC_NWE
	PD7	EXMC_NE0
	PE0	EXMC_NBL0
	PE1	EXMC_NBL1
	PB7	EXMC_NADV
LCD	PD14	EXMC_D0
	PD15	EXMC_D1
	PD0	EXMC_D2
	PD1	EXMC_D3
	PE7	EXMC_D4
	PE8	EXMC_D5
	PE9	EXMC_D6
	PE10	EXMC_D7
	PE11	EXMC_D8
	PE12	EXMC_D9
	PE13	EXMC_D10
	PE14	EXMC_D11
	PE15	EXMC_D12
	PD8	EXMC_D13
	PD9	EXMC_D14
	PD10	EXMC_D15
	PE2	EXMC_A23
	PD4	EXMC_NOE
	PD5	EXMC_NWE
	PD7	EXMC_NE0
USBD	PA11	USB_DM
	PA12	USB_DP

3 入门指南

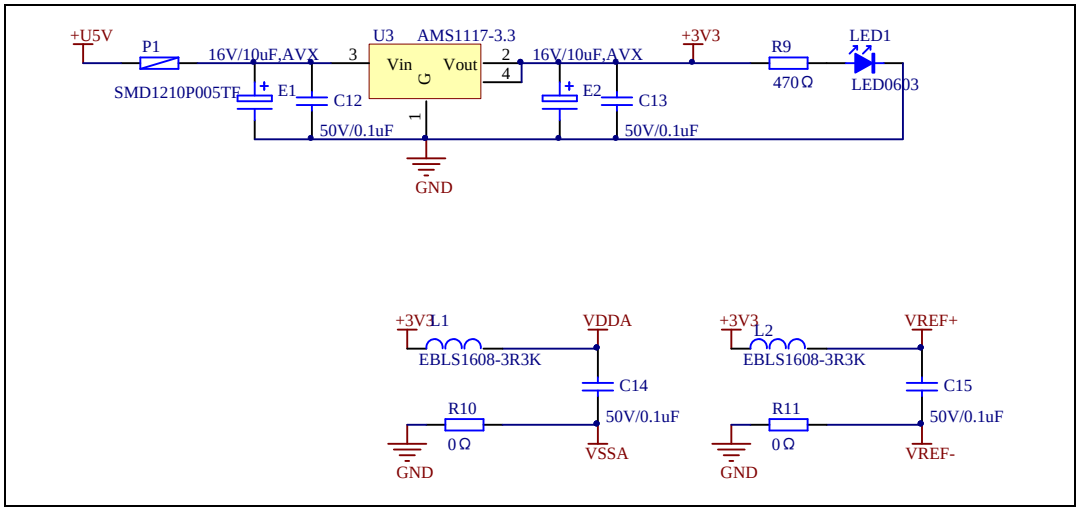
评估板使用 Mini USB 提供 5V 电源。下载程序到评估板需要一套 J-Link 或者使用 GD-Link 工具，在选择了正确的启动方式并且上电后，LED1 将被点亮，表明评估板供电正常。

所有例程提供了 Keil 和 IAR 两个版本，其中 Keil 版的工程是基于 Keil MDK-ARM 4.74 uVision4 创建的，IAR 版的工程是基于 IAR Embedded Workbench for ARM 7.40.2 创建的。在使用过程中有如下几点需要注意：

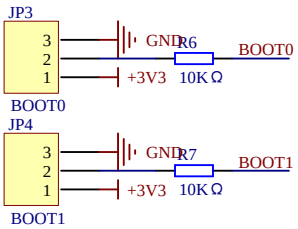
1、如果使用 Keil uVision4 打开工程，安装 GD32F10x_AddOn.2.0.0.exe，以加载相关文件；
2、如果使用 Keil uVision5 打开工程，有两种方法解决“Missing Device(s)”问题。第一种是方法先安装\Library\Firmware\GigaDevice.GD32F10x_DFP.2.0.1.pack，在 Project 菜单中选择 Manage 子菜单，点击 Migrate to Version 5 Format...菜单，将 Keil uVision4 工程转为 Keil uVision5 工程，同时在 Option for Target 的 C/C++ 中添加路径 C:\Keil_v5\ARM\Pack\ARM\CMSIS\4.2.0\CMSIS\Include；第二种方法是直接安装 Addon，在 Folder Selection 中的 Destination Folder 那一栏选择 Keil uVision5 软件的安装目录，如 C:\Keil_v5，然后在 Option for Target 的 Device 选择对应的器件，同时在 Option for Target 的 C/C++中添加路径 C:\Keil_v5\ARM\Pack\ARM\CMSIS\4.2.0\CMSIS\Include。
3、如果使用 IAR 打开工程，安装 IAR_GD32F10x_ADDON.2.0.0.exe，以加载相关文件。

4 硬件设计概述

4.1 供电电源

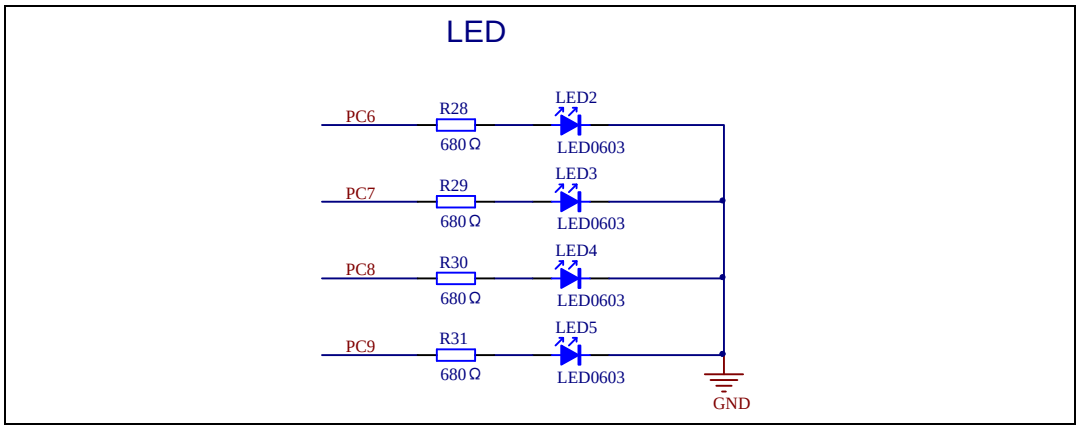


4.2 启动方式选择

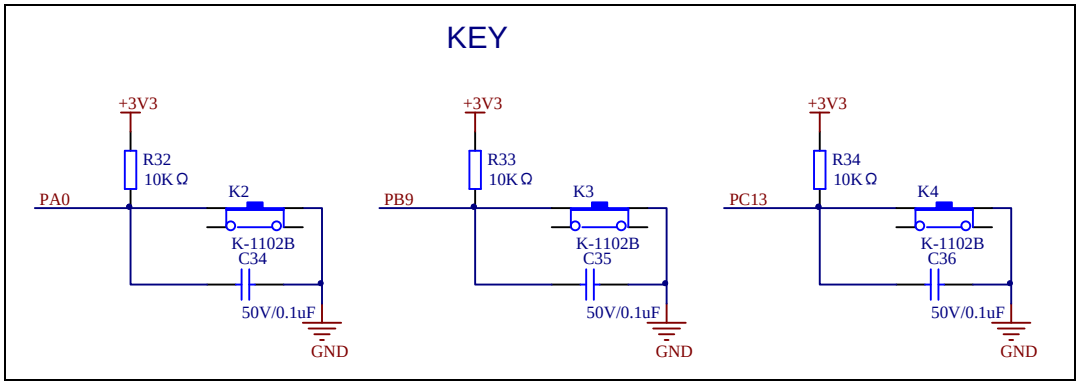


BOOT1	BOOT0	Boot Mode
Any	2-3	User memory
2-3	1-2	System memory
1-2	1-2	SRAM memory

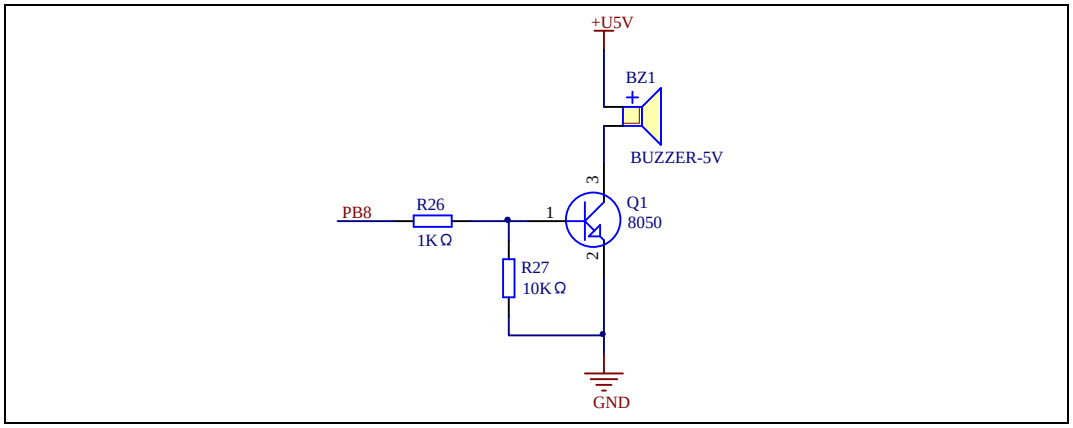
4.3 LED 指示灯



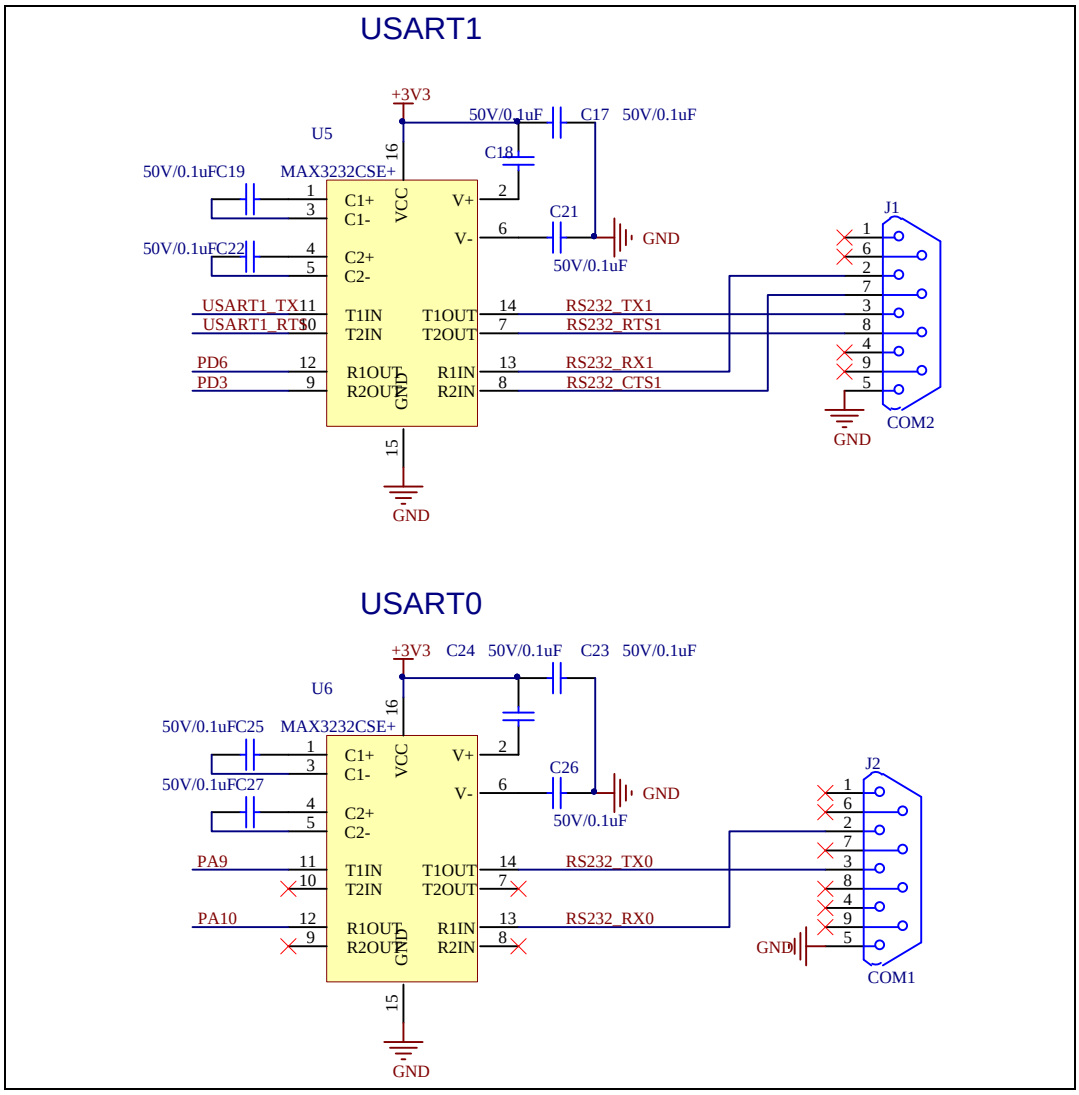
4.4 按键



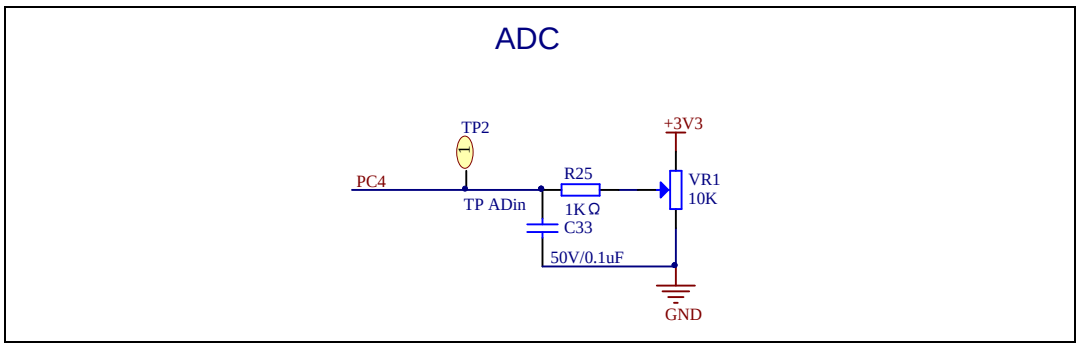
4.5 蜂鸣器



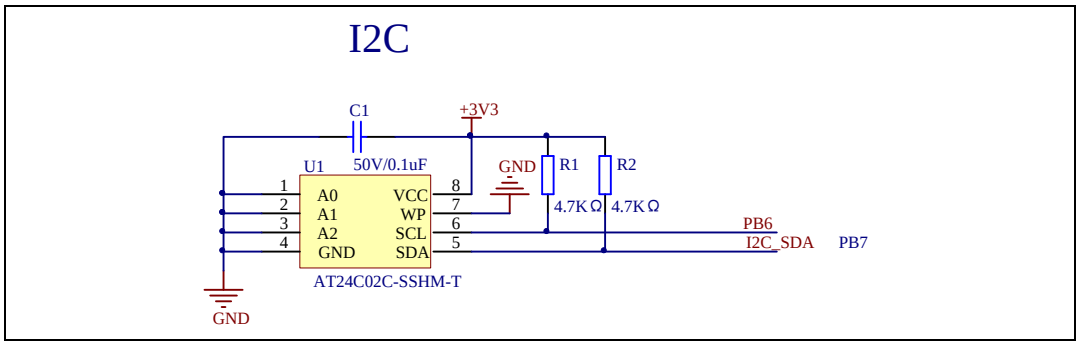
4.6 串口



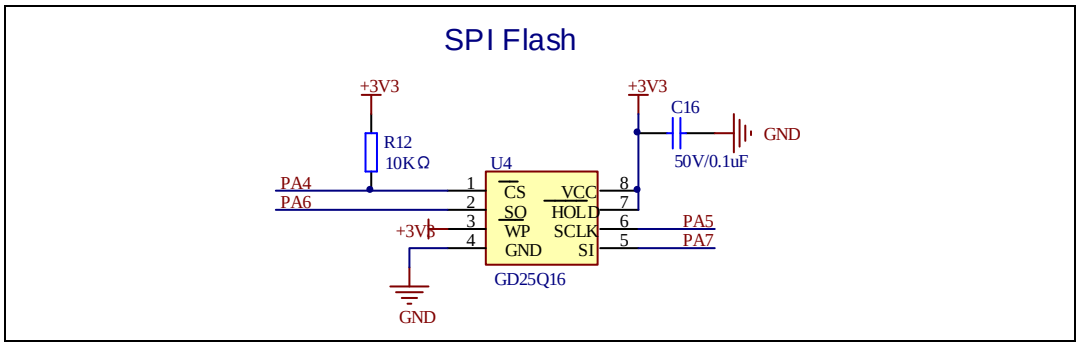
4.7 模数转换器



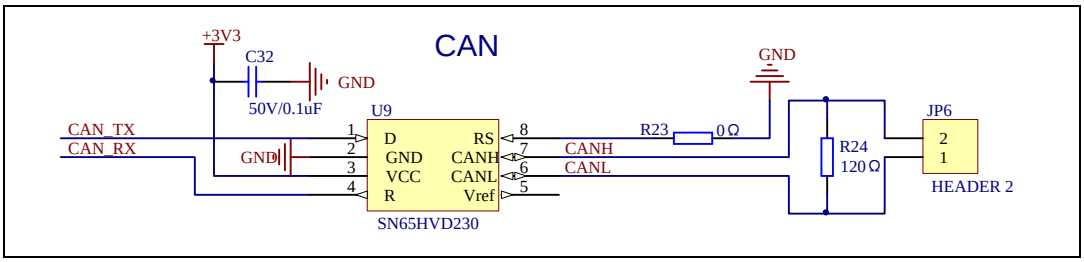
4.8 I2C



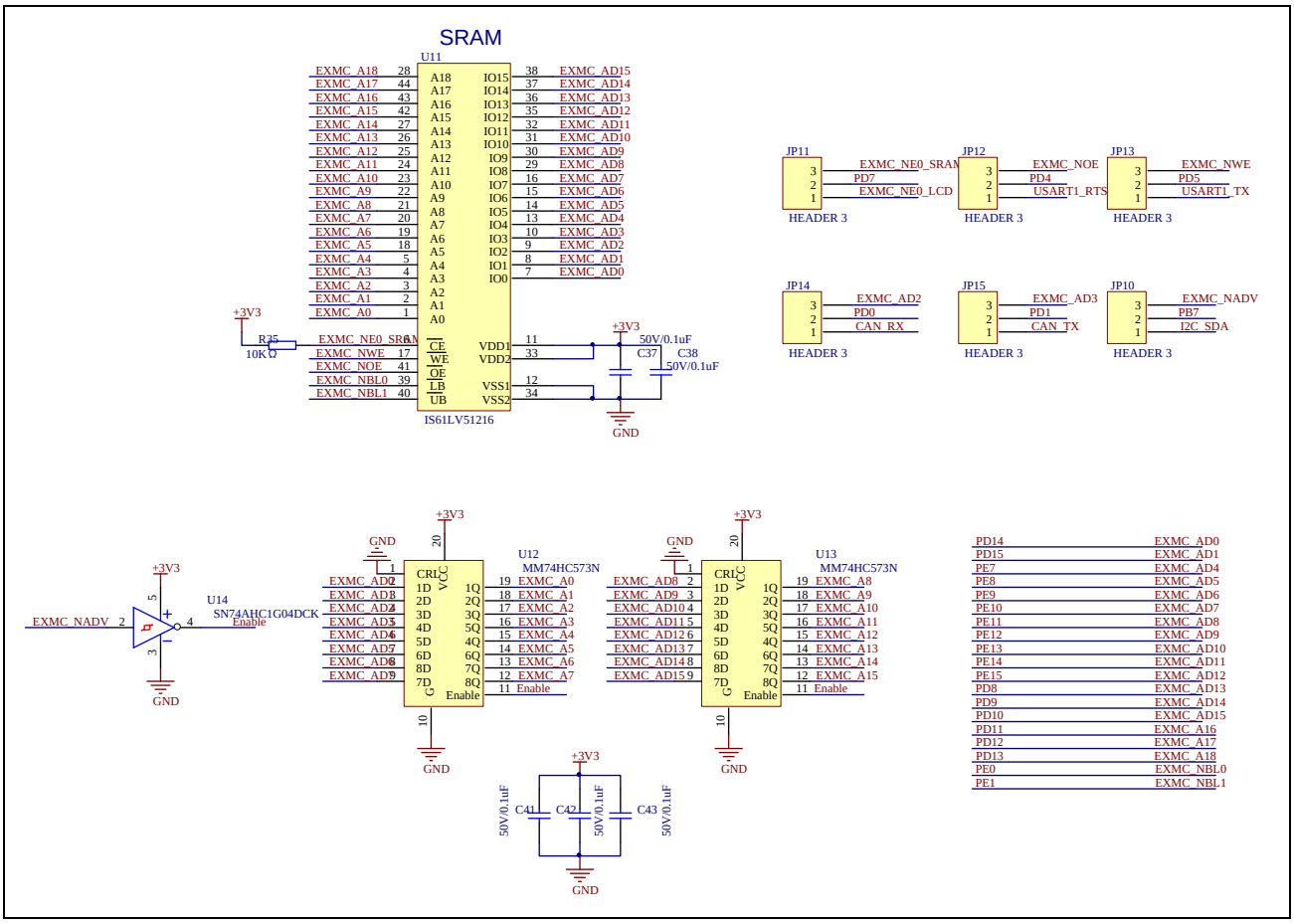
4.9 SPI



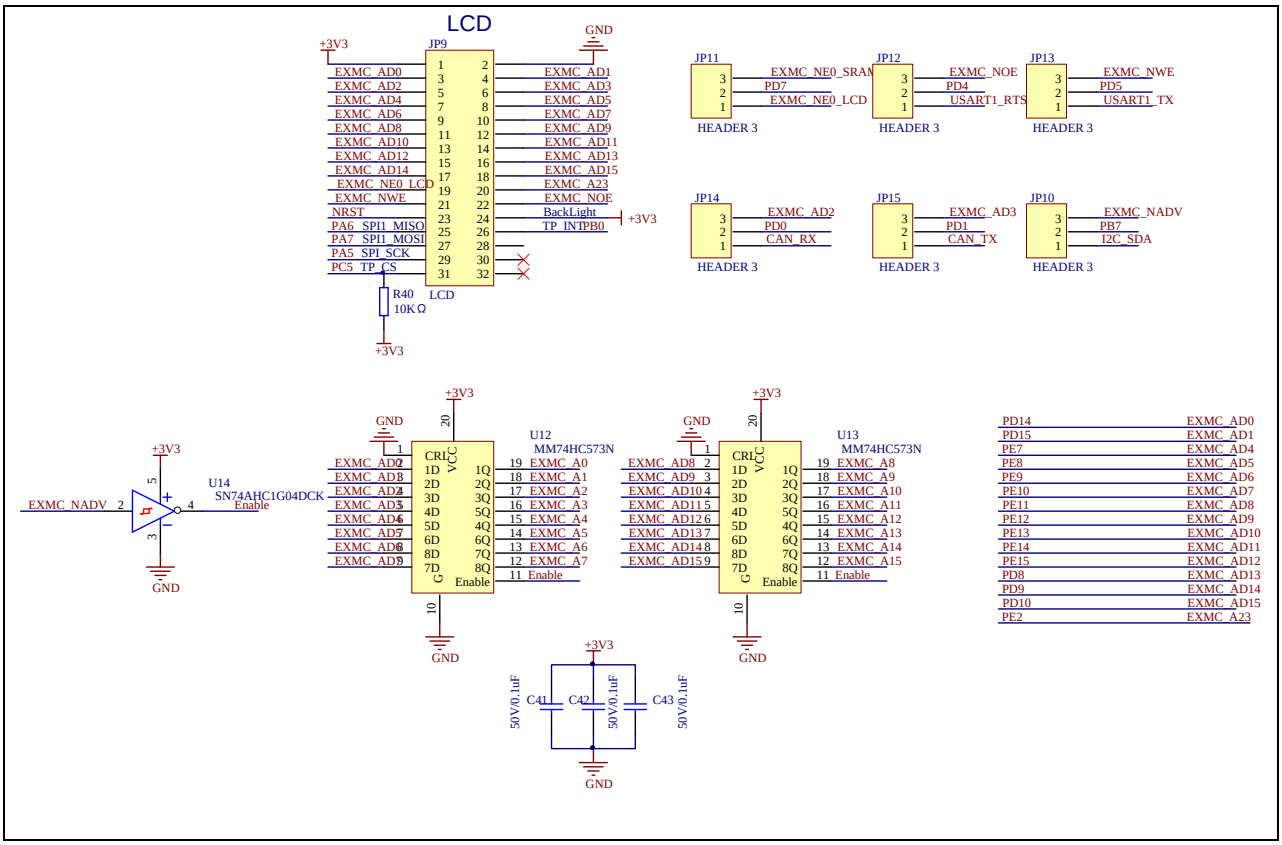
4.10 CAN



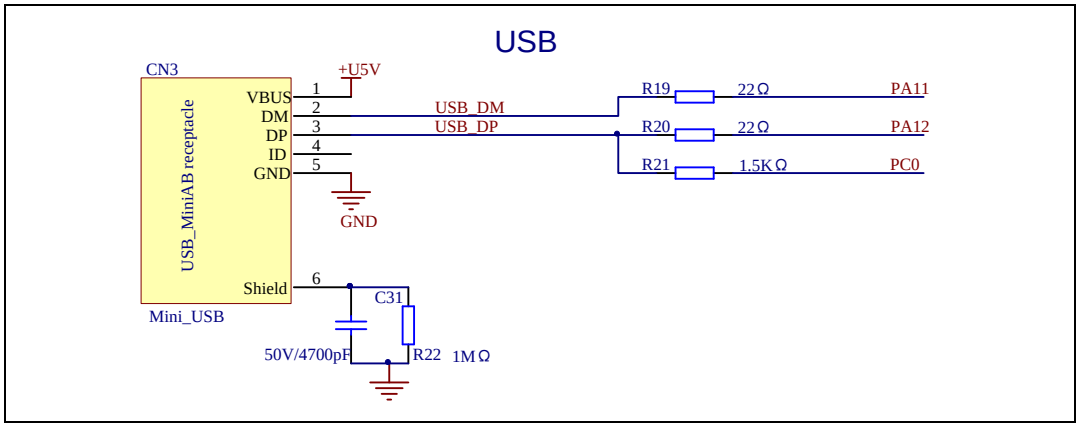
4.11 SRAM


SRAM
U11
IS61LV51216
R35
10K Ω
C37
C38
50V/0.1uF
JP11
JP12
JP13
JP14
JP15
JP10
HEADER 3
U12
U13
MM74HC573N
U14
SN74AHC1G04DCK
C41
C42
C43
+3V3
GND
EXMC_NADV
EXMC_NE0_SRAM
EXMC_NE0_LCD
EXMC_NOE
EXMC_NWE
EXMC_NBL0
EXMC_NBL1
USART1_RTS
USART1_TX
CAN_RX
CAN_TX
I2C_SDA
Enable

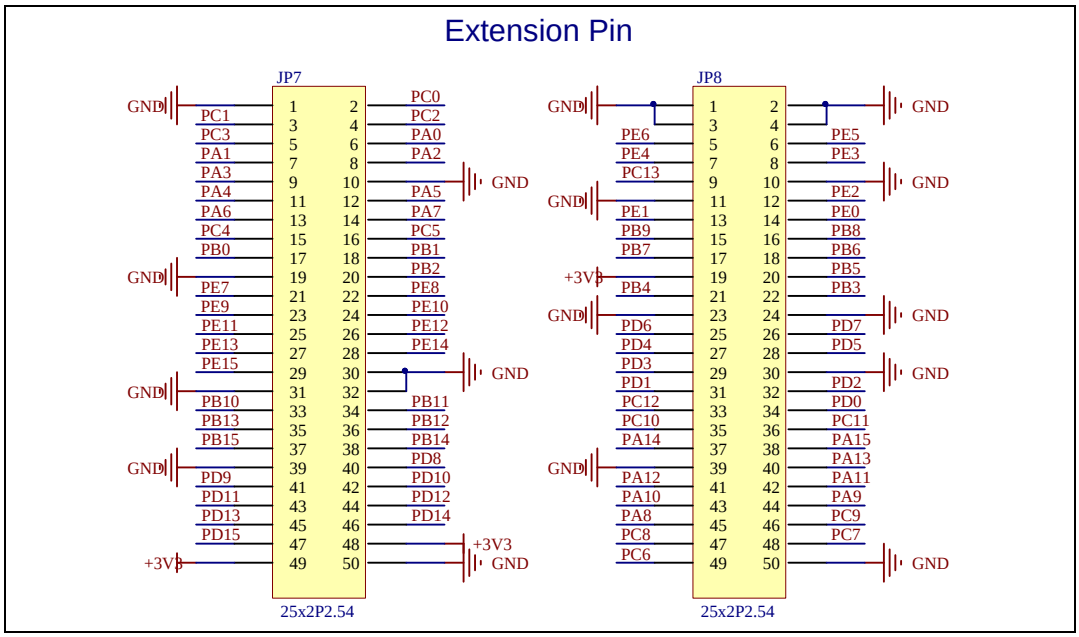
4.12 LCD



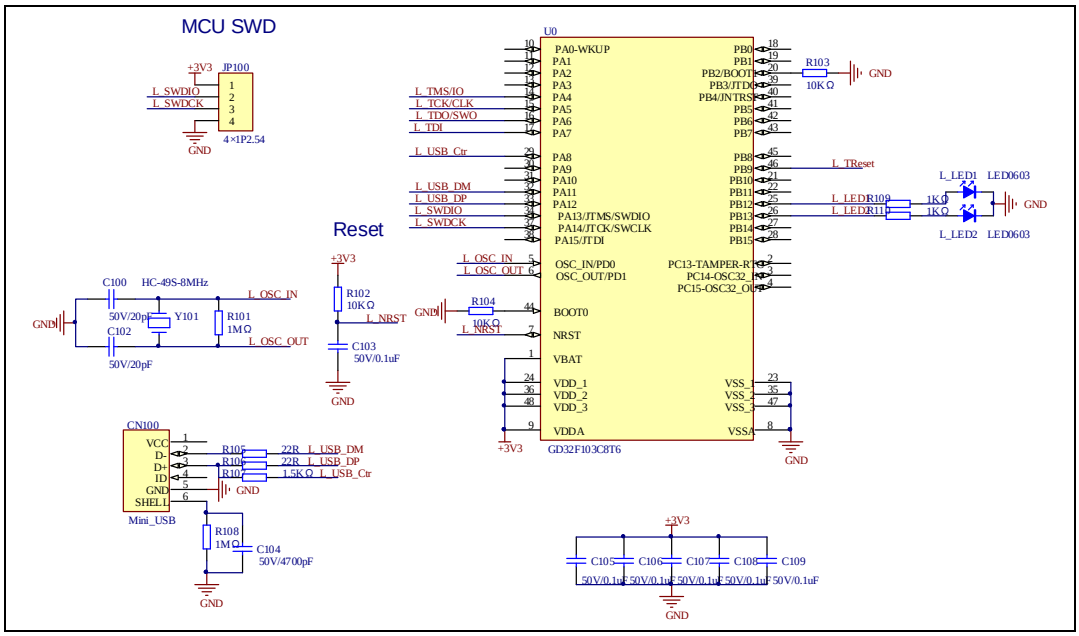
4.13 USBD



4.14 扩展电路



4.15 GD-Link



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

GD32103B-EVAL 开发板上有 4 个 LED。LED2, LED3, LED4, LED5 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_Runing_Led>到开发板上，LED2, LED3, LED4, LED5 将顺序每间隔 200 毫秒点亮，然后一起熄灭，200ms 之后，重复前面的过程。

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

GD32103B-EVAL 开发板有四个按键和四个 LED。其中，四个按键是 Reset 按键，Tamper 按键，Wakeup 按键，User 按键；LED2, LED3 和 LED4,LED5 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 GPIO 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

GD32103B-EVAL 开发板有四个按键和四个 LED。其中，四个按键是 Reset 按键，Tamper 按键，Wakeup 按键，User 按键；LED2, LED3 和 LED4,LED5 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，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

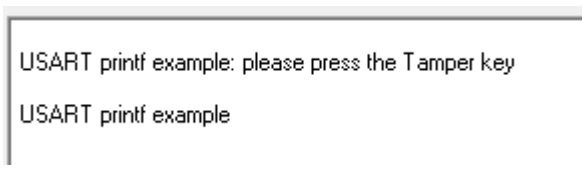
这个例程包括了 GD32 MCU 的以下功能：

- 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，并将串口线连到开发板的 COM0 上。例程首先将输出“USART printf example: please press the Tamper key”到超级终端。按下 Tamper 键，串口继续输出“USART printf example”。

通过串口输出的信息如下图所示。



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串口助手之间的通信

5.5.2 DEMO 执行结果

下载程序< 05_USART_HyperTerminal_Interrupt >到开发板，并将串口线连到开发板的 COM0 上。首先，所有灯亮灭一次用于测试。然后 EVAL_COM0 将首先输出数组 tx_buffer 的内容（从 0x00 到 0xFF）到支持 hex 格式的串口助手并等待接收由串口助手发送的 BUFFER_SIZE 个字节的数据。MCU 将接收到的串口助手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，LED2，LED3，LED4，LED5 轮流闪烁；如果结果不相同，LED2，LED3，LED4，LED5 一起闪烁。

通过串口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 串口硬件流控

5.6.1 DEMO 目的

这个例程包括了 GD32 MCU 的以下功能：

- 学习使用串口硬件流控功能发送和接收数据

5.6.2 DEMO 执行结果

下载程序< 06_USART_HardwareFlowControl >到开发板，并将串口线连到开发板的 COM2 上。首先，所有灯亮灭一次用于测试。然后 EVAL_COM2 将输出字符串“USART Hyperterminal Hardware Flow Control Example: USART - Hyperterminal communication using hardware flow control”到支持字符串格式并且打开了 CTS 和 RTS 功能的串口助手，并等待接收由串口助手发送的以回车符结尾的任意字符串。MCU 将接收到的串口助手发来的字符串发回给串口助手并打印出来。

通过串口输出的信息如下图所示。

```
USART Hyperterminal Hardware Flow Control Example: USART - Hyperterminal communication using hardware flow control

The words send from the Hyper Terminal
```

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 16（温度传感器通道）、内部通道 17（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 COM0 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、内部参考电压和电池电压的值。

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 50 degrees Celsius
the reference voltage data is 1.175V

the temperature data is 50 degrees Celsius
the reference voltage data is 1.174V

the temperature data is 50 degrees Celsius
the reference voltage data is 1.177V

the temperature data is 50 degrees Celsius
the reference voltage data is 1.176V

the temperature data is 50 degrees Celsius
the reference voltage data is 1.175V
```

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 COM0 口连接到电脑，打开电脑串口软件。

TIMER0_CH0 作为 ADC0 和 ADC1 的触发源。当 TIMER0_CH0 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 TIMER0_CH0 的第一个上升沿到来，ADC0 转换的 PC4 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 PC5 引脚的电压值存储到 adc_value[0]的高半字。当 TIMER0_CH0 的第二个上升沿到来，ADC0 转换的 PC5 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 PC4 引脚的电压值存储到 adc_value[1]的高半字。

当程序运行时，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 0FDF0681
the data adc_value[1] is 06AC0FE3

the data adc_value[0] is 0FDF0681
the data adc_value[1] is 06AC0FE2

the data adc_value[0] is 0FDF06A0
the data adc_value[1] is 06AE0FE0

the data adc_value[0] is 0FDF06A0
the data adc_value[1] is 06AC0FE2

the data adc_value[0] is 0FDF06A0
the data adc_value[1] is 06AC0FE3

the data adc_value[0] is 0FDF06A0
the data adc_value[1] is 06AC0FE2

the data adc_value[0] is 020406A1
the data adc_value[1] is 06AC037E

the data adc_value[0] is 00AA0681
the data adc_value[1] is 06AC00E5

the data adc_value[0] is 00000681
the data adc_value[1] is 06AC0000
```

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 COM0 口连接到电脑，打开电脑串口软件。

TIMER0_CH0 作为 ADC0 和 ADC1 的触发源。当 TIMER0_CH0 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 TIMER0_CH0 的第一个上升沿到来，ADC0 转换的 PC4 引脚的电压值存储到 adc_value[0] 的低半字，并且 ADC1 转换的 PC5 引脚的电压值存储到 adc_value[0]的高半字。当 TIMER0_CH0 的第二个上升沿到来，ADC0 转换的 PC5 引脚的电压值存储到 adc_value[1]的低半字，并且 ADC1 转换的 PC4 引脚的电压值存储到 adc_value[1]的高半字。

当程序运行时，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 0000066D
the data adc_value[1] is 06790000

the data adc_value[0] is 0000066D
the data adc_value[1] is 067A0000

the data adc_value[0] is 0000063F
the data adc_value[1] is 064F0000

the data adc_value[0] is 00000627
the data adc_value[1] is 06330000

the data adc_value[0] is 00000626
the data adc_value[1] is 06310000

the data adc_value[0] is 000005FF
the data adc_value[1] is 06100000

the data adc_value[0] is 000005BF
the data adc_value[1] is 05EB0000

the data adc_value[0] is 000005BF
the data adc_value[1] is 05E90000
```

5.10 I2C 访问 EEPROM

5.10.1 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

5.10.2 DEMO 执行结果

下载程序<10_I2C_EEPROM >到开发板上。将开发板的 COM0 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!”，同时开发板上的四个 LED 灯开始顺序闪烁，否则串口打印出“Err: data read and write aren't matching.”，同时四个 LED 全亮。

通过串口输出的信息如下图所示。

```
I2C-24C02 configured....
The I2C0 is hardware interface
The speed is 400000
AT24C02 writing...
0x00 0x01 0x02 0x03 0x04 0x05 0x06 0x07 0x08 0x09 0x0A 0x0B 0x0C 0x0D 0x0E 0x0F
0x10 0x11 0x12 0x13 0x14 0x15 0x16 0x17 0x18 0x19 0x1A 0x1B 0x1C 0x1D 0x1E 0x1F
0x20 0x21 0x22 0x23 0x24 0x25 0x26 0x27 0x28 0x29 0x2A 0x2B 0x2C 0x2D 0x2E 0x2F
0x30 0x31 0x32 0x33 0x34 0x35 0x36 0x37 0x38 0x39 0x3A 0x3B 0x3C 0x3D 0x3E 0x3F
0x40 0x41 0x42 0x43 0x44 0x45 0x46 0x47 0x48 0x49 0x4A 0x4B 0x4C 0x4D 0x4E 0x4F
0x50 0x51 0x52 0x53 0x54 0x55 0x56 0x57 0x58 0x59 0x5A 0x5B 0x5C 0x5D 0x5E 0x5F
0x60 0x61 0x62 0x63 0x64 0x65 0x66 0x67 0x68 0x69 0x6A 0x6B 0x6C 0x6D 0x6E 0x6F
0x70 0x71 0x72 0x73 0x74 0x75 0x76 0x77 0x78 0x79 0x7A 0x7B 0x7C 0x7D 0x7E 0x7F
0x80 0x81 0x82 0x83 0x84 0x85 0x86 0x87 0x88 0x89 0x8A 0x8B 0x8C 0x8D 0x8E 0x8F
0x90 0x91 0x92 0x93 0x94 0x95 0x96 0x97 0x98 0x99 0x9A 0x9B 0x9C 0x9D 0x9E 0x9F
0xA0 0xA1 0xA2 0xA3 0xA4 0xA5 0xA6 0xA7 0xA8 0xA9 0xAA 0xAB 0xAC 0xAD 0xAE 0xAF
0xB0 0xB1 0xB2 0xB3 0xB4 0xB5 0xB6 0xB7 0xB8 0xB9 0xBA 0xBB 0xBC 0xBD 0xBE 0xBF
0xC0 0xC1 0xC2 0xC3 0xC4 0xC5 0xC6 0xC7 0xC8 0xC9 0xCA 0xCB 0xCC 0xCD 0xCE 0xCF
0xD0 0xD1 0xD2 0xD3 0xD4 0xD5 0xD6 0xD7 0xD8 0xD9 0xDA 0xDB 0xDC 0xDD 0xDE 0xDF
0xE0 0xE1 0xE2 0xE3 0xE4 0xE5 0xE6 0xE7 0xE8 0xE9 0xEA 0xEB 0xEC 0xED 0xEE 0xEF
0xF0 0xF1 0xF2 0xF3 0xF4 0xF5 0xF6 0xF7 0xF8 0xF9 0xFA 0xFB 0xFC 0xFD 0xFE 0xFF
AT24C02 reading...
0x00 0x01 0x02 0x03 0x04 0x05 0x06 0x07 0x08 0x09 0x0A 0x0B 0x0C 0x0D 0x0E 0x0F
0x10 0x11 0x12 0x13 0x14 0x15 0x16 0x17 0x18 0x19 0x1A 0x1B 0x1C 0x1D 0x1E 0x1F
0x20 0x21 0x22 0x23 0x24 0x25 0x26 0x27 0x28 0x29 0x2A 0x2B 0x2C 0x2D 0x2E 0x2F
0x30 0x31 0x32 0x33 0x34 0x35 0x36 0x37 0x38 0x39 0x3A 0x3B 0x3C 0x3D 0x3E 0x3F
0x40 0x41 0x42 0x43 0x44 0x45 0x46 0x47 0x48 0x49 0x4A 0x4B 0x4C 0x4D 0x4E 0x4F
0x50 0x51 0x52 0x53 0x54 0x55 0x56 0x57 0x58 0x59 0x5A 0x5B 0x5C 0x5D 0x5E 0x5F
0x60 0x61 0x62 0x63 0x64 0x65 0x66 0x67 0x68 0x69 0x6A 0x6B 0x6C 0x6D 0x6E 0x6F
0x70 0x71 0x72 0x73 0x74 0x75 0x76 0x77 0x78 0x79 0x7A 0x7B 0x7C 0x7D 0x7E 0x7F
0x80 0x81 0x82 0x83 0x84 0x85 0x86 0x87 0x88 0x89 0x8A 0x8B 0x8C 0x8D 0x8E 0x8F
0x90 0x91 0x92 0x93 0x94 0x95 0x96 0x97 0x98 0x99 0x9A 0x9B 0x9C 0x9D 0x9E 0x9F
0xA0 0xA1 0xA2 0xA3 0xA4 0xA5 0xA6 0xA7 0xA8 0xA9 0xAA 0xAB 0xAC 0xAD 0xAE 0xAF
0xB0 0xB1 0xB2 0xB3 0xB4 0xB5 0xB6 0xB7 0xB8 0xB9 0xBA 0xBB 0xBC 0xBD 0xBE 0xBF
0xC0 0xC1 0xC2 0xC3 0xC4 0xC5 0xC6 0xC7 0xC8 0xC9 0xCA 0xCB 0xCC 0xCD 0xCE 0xCF
0xD0 0xD1 0xD2 0xD3 0xD4 0xD5 0xD6 0xD7 0xD8 0xD9 0xDA 0xDB 0xDC 0xDD 0xDE 0xDF
0xE0 0xE1 0xE2 0xE3 0xE4 0xE5 0xE6 0xE7 0xE8 0xE9 0xEA 0xEB 0xEC 0xED 0xEE 0xEF
0xF0 0xF1 0xF2 0xF3 0xF4 0xF5 0xF6 0xF7 0xF8 0xF9 0xFA 0xFB 0xFC 0xFD 0xFE 0xFF
I2C-AT24C02 test passed!
```

5.11 SPI FLASH

5.11.1 DEMO 目的

这个例程包括了 GD32 MCU 的以下功能：

- 学习使用 SPI 模块的 SPI 主模式读写带有 SPI 接口的 NOR Flash。

5.11.2 DEMO 执行结果

把电脑串口线连接到开发板的 COM0 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。

下载程序 <11_SPI_SPI_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，写入和读出 FLASH 的 256 字节数据。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 1 打印出“SPI-GD25Q16 Test Passed!”，否则，串口打印出"Err: Data Read and Write aren't Matching."。最后，四个 LED 灯依次循环点亮。

下图是实验结果图。

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

System is Starting up...

Flash:128K

The CPU Unique Device ID:[38363452-30313737-140080]

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!
```

5.12 SRAM 存储器

5.12.1 DEMO 目的

这个例程包括了 GD32 MCU 的以下功能：

- 学习使用 EXMC 控制 SRAM

5.12.2 DEMO 执行结果

GD32103B-EVAL 开发板使用 EXMC 模块来控制 SRAM。在运行例程之前，JP10、JP12、JP13、JP14 和 JP15 连接到 EXMC，JP11 连接到 sram。下载程序<12_EXMC_SRAM>到开发板。这个例程演示 EXMC 对 SRAM 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 LED2，否则点亮 LED5。超级终端输出信息如下：

```
SRAM test successed!
the data is:
  1215  1216  1217  1218
```

5.13 LCD 触摸屏

5.13.1 DEMO 目的

这个例程包括了 GD32 MCU 的以下功能：

- 学习使用 EXMC 控制 LCD

5.13.2 DEMO 执行结果

GD32103B-EVAL 开发板使用 EXMC 模块来控制 LCD。在运行例程之前，JP10、JP12、JP13、JP14 和 JP15 连接到 EXMC，JP11 连接到 lcd。下载程序<13_EXMC_TouchScreen>到开发板。这个例程将通过 EXMC 模块在 LCD 屏上显示 GigaDevice 的 logo 和 4 个绿色按钮。用户可以通过触摸屏上的按钮来点亮开发板中对应的 LED，同时屏上触摸过的按钮颜色将变成红色。

5.14 CAN 网络通信

5.14.1 DEMO 目的

这个例程包括了 GD32 MCU 的以下功能：

- 学习使用 CAN0 实现两个板子之间的通信

GD32103B-EVAL 开发板集成了 CAN(控制器局域网络)总线控制器，他是一种常用的工业控制总线。CAN 总线控制器遵循 2.0A 和 2.0B 总线协议。该例程演示了在两个板子之间通过 CAN0 进行通信。

5.14.2 DEMO 执行结果

该例程的测试需要两个 GD32103B-EVAL 开发板。用跳线帽将 JP14、JP15 跳到 CAN 上。将两个板子的 JP6 的 L 引脚和 H 引脚分别相连，用于发送或者接收数据帧。下载程序<14_CAN_Network>到两个开发板中，并将串口线连到开发板的 COM0 上。例程首先将输出“please press the Tamper key to transmit data!”到超级终端。按下 Tamper 键，数据帧通过

CAN0 发送出去同时通过串口打印出来。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。

```
please press the Tamper key to transmit data!
CAN0 transmit data: ab,cd
CAN0 receive data: ab,cd
```

5.15 RCU 时钟输出

5.15.1 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

5.15.2 DEMO 执行结果

下载程序<15_RCU_Clock_Out>到开发板上并运行。将开发板的 COM0 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 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_OUT0: system clock
CK_OUT0: IRC8M
CK_OUT0: HXTAL
CK_OUT0: system clock
```

5.16 PMU 睡眠模式唤醒

5.16.1 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

5.16.2 DEMO 执行结果

下载程序<16_PMU_sleep_wakeup >到开发板上，并将串口线连到开发板的 COM0 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。

5.17 RTC 日历

5.17.1 DEMO 目的

这个例程包括了 GD32 MCU 的以下功能：

- 学习使用 RTC 模块实现日历和闹钟功能
- 学习使用 USART 模块实现时间显示

5.17.2 DEMO 执行结果

下载程序<17_RTC_Calendar>到开发板上，使用串口线连接电脑到开发板 COM0 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。同时程序会将设置的时间增加 10 秒作为闹钟时间。在 10 秒以后，闹钟产生，会在串口助手上显示并且点亮 LED 灯。

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

This is a RTC demo!

 RTC not yet configured....
 RTC configured....
==============Time Settings=====================================
  Please Set Hours:  20
  Please Set Minutes:  20
  Please Set Seconds:  20
 Set Alarm Time: 20:20:30
 Time: 20:20:20
 Time: 20:20:20
 Time: 20:20:21
 Time: 20:20:22
 Time: 20:20:23
 Time: 20:20:24
 Time: 20:20:25
 Time: 20:20:26
 Time: 20:20:27
 Time: 20:20:28
 Time: 20:20:29

==============RTC Alarm and turn on LED=====================================
 Time: 20:20:30
 Time: 20:20:31
```

5.18 呼吸灯

5.18.1 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

5.18.2 DEMO 执行结果

使用杜邦线连接 TIMER0_CH0（PA8）和 LED2（PC6），然后下载程序<18_TIMER_Breath_LED>到开发板，并运行程序。

PA8 不要用于其他外设。

可以看到 LED2 由暗变亮，由亮变暗，往复循环，就像人的呼吸一样有节奏。

5.19 USBD 键盘

5.19.1 DEMO 目的

这个例程包括了 GD32 MCU 的以下功能：

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

在本例程中，GD32103B 开发板被 USB 主机利用内部 HID 驱动枚举为一个 USB 键盘，如下图所示，按不同按键可以输出三个字符（‘a’，‘b’ 和 ‘c’）。另外，本例程支持 USB 键盘远程唤醒主机，其中‘Tamper’按键被作为唤醒源。

5.19.2 DEMO 执行结果

将<19_USBD_Keyboard>例程下载到开发板中，并运行。按下 Tamper 键，输出‘a’；按下 Wakeup 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 Tamper 按键；

-- 如果 PC 被唤醒，表明 USB 远程唤醒功能正常，否则失败。

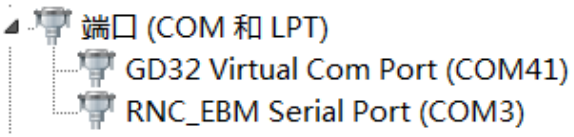
5.20 USBD 虚拟串口

5.20.1 DEMO 目的

这个例程包括了 GD32 MCU 的以下功能：

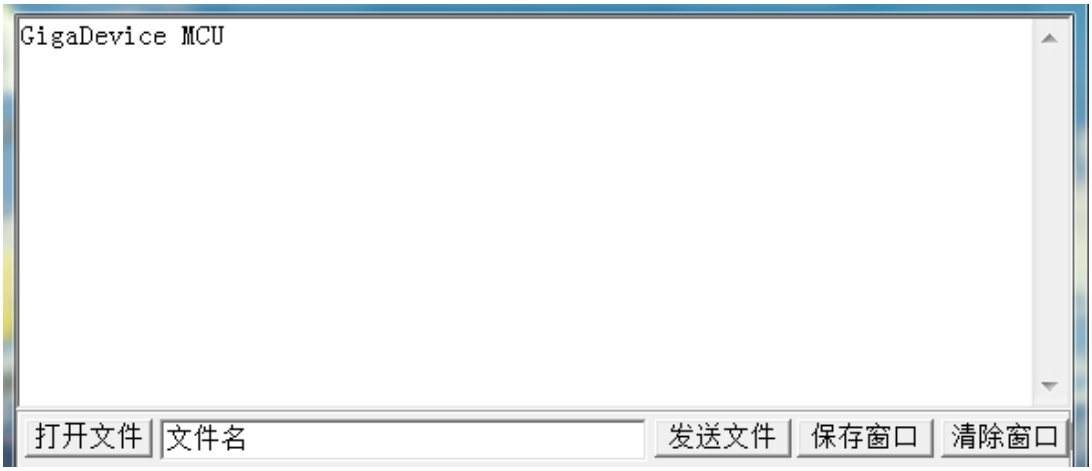
- 学习如何使用 USBD 设备
- 学习如何实现 USB CDC 设备

GD32103B 开发板具有一个 USBD 接口。在本例程中，GD32103B 开发板被 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 键盘看起来像是个串口，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。



5.20.2 DEMO 执行结果

将<20_USBD_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。


GigaDevice MCU
打开文件
文件名
发送文件
保存窗口
清除窗口

6 版本更新历史

表 2. 版本更新历史

版本号	描述	日期
1.0	初始发布版本	2014 年 12 月 26 日
2.0	固件库更新	2017 年 6 月 30 日
2.1	固件库更新，例程名更新，LCD 例程更新 logo，SD 卡驱动更新	2021 年 4 月 30 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 it’s 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 it’s 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.