

**GigaDevice Semiconductor Inc.**

**GD32350R-EVAL 评估板**

**用户手册**

**V2.1**

# 目录

目录.....	1
图.....	4
表.....	5
1. 简介.....	6
2. 功能引脚分配.....	6
3. 入门指南.....	7
4. 硬件设计概述.....	8
4.1. 供电电源.....	8
4.2. 启动方式选择 .....	8
4.3. LED 指示灯 .....	9
4.4. 按键.....	9
4.5. 串口 .....	10
4.6. RS485 .....	10
4.7. 模数转换器/数模转换器.....	11
4.8. I2S.....	12
4.9. I2C.....	12
4.10. SPI-TF CARD.....	13
4.11. SPI-TFT LCD .....	13
4.12. USBFS .....	14
4.13. 比较器.....	14
4.14. HDMI-CEC .....	15
4.15. TSI.....	15
4.16. 红外线接口 .....	15
4.17. RTC .....	16
4.18. GD-Link .....	17
4.19. 扩展电路.....	18
4.20. MCU .....	19
5. 例程使用指南.....	19
5.1. GPIO 流水灯 .....	19
5.1.1. DEMO 目的 .....	19

5.1.2. DEMO 执行结果.....	19
<b>5.2. GPIO 按键轮询模式 .....</b>	<b>20</b>
5.2.1. DEMO 目的 .....	20
5.2.2. DEMO 执行结果.....	20
<b>5.3. EXTI 按键中断模式 .....</b>	<b>20</b>
5.3.1. DEMO 目的 .....	20
5.3.2. DEMO 执行结果.....	20
<b>5.4. 串口打印.....</b>	<b>21</b>
5.4.1. DEMO 目的 .....	21
5.4.2. DEMO 执行结果.....	21
<b>5.5. 串口中断收发 .....</b>	<b>21</b>
5.5.1. DEMO 目的 .....	21
5.5.2. DEMO 执行结果.....	21
<b>5.6. 串口 DMA 收发 .....</b>	<b>22</b>
5.6.1. DEMO 目的 .....	22
5.6.2. DEMO 执行结果.....	22
<b>5.7. RS485 实验.....</b>	<b>22</b>
5.7.1. DEMO 目的 .....	22
5.7.2. DEMO 执行结果.....	23
<b>5.8. 定时器触发模数转换 .....</b>	<b>23</b>
5.8.1. DEMO 目的 .....	23
5.8.2. DEMO 执行结果.....	24
<b>5.9. DAC 输出电压值 .....</b>	<b>24</b>
5.9.1. DEMO 目的 .....	24
5.9.2. DEMO 执行结果.....	24
<b>5.10. 比较器输出获取指示灯 .....</b>	<b>24</b>
5.10.1. DEMO 目的 .....	24
5.10.2. DEMO 执行结果.....	24
<b>5.11. I2C 访问 EEPROM .....</b>	<b>25</b>
5.11.1. DEMO 目的 .....	25
5.11.2. DEMO 执行结果.....	25
<b>5.12. SPI TF 卡 Block 读写.....</b>	<b>26</b>
5.12.1. DEMO 目的 .....	26
5.12.2. DEMO 执行结果.....	26
<b>5.13. SPI TF 卡 FATFS 读写 .....</b>	<b>26</b>
5.13.1. DEMO 目的 .....	26
5.13.2. DEMO 执行结果.....	27
<b>5.14. SPI 驱动 LCD 液晶屏 .....</b>	<b>27</b>
5.14.1. DEMO 目的 .....	27

5.14.2.	DEMO 执行结果 .....	27
<b>5.15.</b>	<b>HDMI_CEC 通信 .....</b>	<b>28</b>
5.15.1.	DEMO 目的 .....	28
5.15.2.	DEMO 执行结果 .....	28
<b>5.16.</b>	<b>音频播放器 .....</b>	<b>29</b>
5.16.1.	DEMO 目的 .....	29
5.16.2.	DEMO 执行结果 .....	29
<b>5.17.</b>	<b>RCU 时钟输出 .....</b>	<b>29</b>
5.17.1.	DEMO 目的 .....	29
5.17.2.	DEMO 执行结果 .....	29
<b>5.18.</b>	<b>CTC 校准 .....</b>	<b>30</b>
5.18.1.	DEMO 目的 .....	30
5.18.2.	DEMO 执行结果 .....	30
<b>5.19.</b>	<b>PMU 睡眠模式唤醒 .....</b>	<b>30</b>
5.19.1.	DEMO 目的 .....	30
5.19.2.	DEMO 执行结果 .....	30
<b>5.20.</b>	<b>RTC 实时时钟 .....</b>	<b>30</b>
5.20.1.	DEMO 目的 .....	30
5.20.2.	DEMO 执行结果 .....	30
<b>5.21.</b>	<b>红外收发器 .....</b>	<b>32</b>
5.21.1.	DEMO 目的 .....	32
5.21.2.	DEMO 执行结果 .....	32
<b>5.22.</b>	<b>TIMER 呼吸灯 .....</b>	<b>32</b>
5.22.1.	DEMO 目的 .....	32
5.22.2.	DEMO 执行结果 .....	32
<b>5.23.</b>	<b>TSI 触摸按键 .....</b>	<b>32</b>
5.23.1.	DEMO 目的 .....	32
5.23.2.	DEMO 执行结果 .....	33
<b>5.24.</b>	<b>USB 设备 .....</b>	<b>33</b>
5.24.1.	设备虚拟串口 .....	33
5.24.2.	设备键盘 .....	34
<b>5.25.</b>	<b>USB 主机 .....</b>	<b>34</b>
5.25.1.	HID 主机 .....	34
5.25.2.	MSC 主机 .....	35
<b>6.</b>	<b>版本更新历史 .....</b>	<b>36</b>

## 图

图 4-1 供电电源原理图.....	8
图 4-2 启动方式选择原理图.....	8
图 4-3 LED 功能原理图.....	9
图 4-4 按键功能原理图.....	9
图 4-5 串口 0 功能原理图.....	10
图 4-6 RS485 功能原理图.....	10
图 4-7 模数/数模转换器功能原理图.....	11
图 4-8 I2S 功能原理图.....	12
图 4-9 I2C 功能原理图.....	12
图 4-10 SPI-TF CARD 功能原理图.....	13
图 4-11 SPI-TFT LCD 功能原理图.....	13
图 4-12 USBFS 功能原理图.....	14
图 4-13 比较器功能原理图.....	14
图 4-14 HDMI-CEC 功能原理图.....	15
图 4-15 TSI 功能原理图.....	15
图 4-16 红外线接口功能原理图.....	15
图 4-17 RTC 功能原理图.....	16
图 4-18 GD-Link 功能原理图.....	17
图 4-19 扩展电路功能原理图.....	18
图 4-20 MCU 功能原理图.....	19

# 表

表 2-1 引脚分配.....	6
表 4-1 启动方式配置.....	8
表 6-1 版本更新历史.....	36

## 1. 简介

GD32350R-EVAL 评估板使用 GD32F350RBT6 作为主控制器。该评估板为采用 Cortex™-M4 内核的 GD32F3x0 芯片提供了一个完整的开发平台，支持全方位的外围设备。评估板使用 mini-USB 接口或 5V 的 AC/DC 适配器作为供电电源。提供包括扩展引脚在内的以及 SWD、Reset、Boot、User button key、LED、I2C、I2S、USART、RS485、TFT-LCD、HDMI-CEC、LDR、TSI、IFRP LED、IFRP Receiver、RTC、SPI、USB、ADC、DAC 等外设资源。本文档提供详细的硬件原理图和相关应用程序。

## 2. 功能引脚分配

表 2-1 引脚分配

功能	引脚	描述
LED	PC10	LED1
	PC11	LED2
	PC12	LED3
	PD2	LED4
RESET		K1-Reset
KEY	PA0	K2-Wakeup
	PC13	K4-Temper
	PF7	K3-User Key
USB	PA11	USBDM
	PA12	USBDP
	PA9	USB VBUS
IR	PC6	IR_RX
	PB9	IR_TX
I2C	PB6	I2C0_SCL
	PB7	I2C0_SDA
I2S	PA4	I2S_WS
	PA5	I2S_CK
	PA7	I2S_DIN
	PA15	MSEL
	PB3	MCLK
	PB5	MDIN
	PA6	I2S_MCK
USART0	PA9	USART0_TX
	PA10	USART0_RX
RS485	PA2	RS485_TX
	PA3	RS485_RX
	PA1	RS485_DIR
TSI	PB11	TSI_G5_IO0
	PB14	TSI_G5_IO3

功能	引脚	描述
	PB13	TSI_G2_IO2
	PB12	TSI_G5_IO1
	PC5	TSI_G2_IO0
	PB0	TSI_G2_IO1
SPI	PB3	SPI0_SCK
	PB4	SPI0_MISO
	PB5	SPI0_MOSI
	PF5	TFT_CS
	PF4	TF_CARD_CS
	PC4	TFT_RESET
ADC	PC1	ADC_IN11
HDMI-CEC	PB8	CEC
COMPARATOR	PA1	COMP0_INP
DAC	PA4	DAC_OUT

### 3. 入门指南

评估板使用 mini-USB 或者 DC-005 连接器提供 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，LED6 将被点亮，表明评估板供电正常。

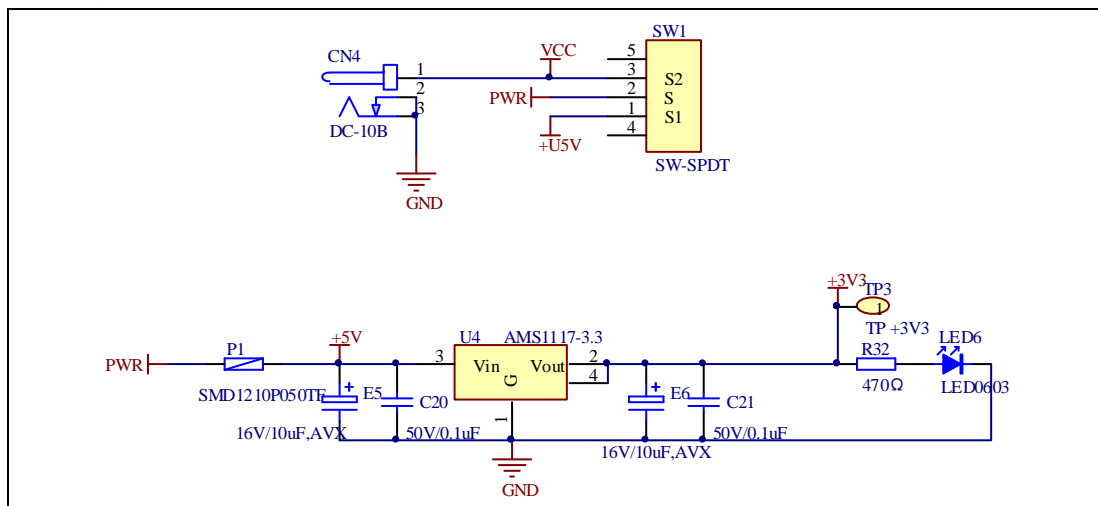
所有例程提供了 Keil 和 IAR 两个版本，其中 Keil 版的工程是基于 Keil MDK-ARM 4.74 uVision4 创建的，IAR 版的工程是基于 IAR Embedded Workbench for ARM 7.40.2 创建的。在使用过程中有如下几点需要注意：

- 1、如果使用 Keil uVision4 打开工程，安装 GD32F3x0\_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.GD32F3x0\_DFP.2.0.0.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\_GD32F3x0\_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

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

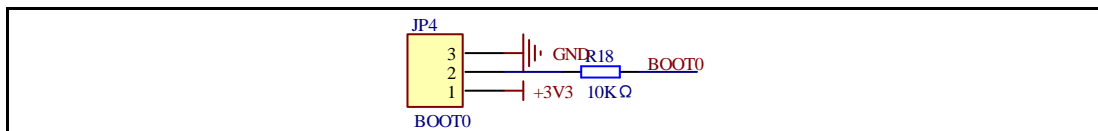
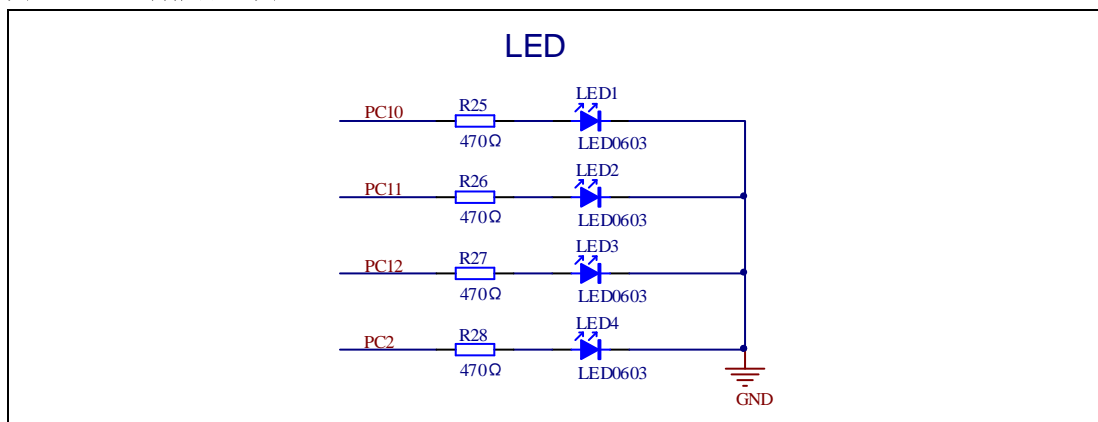


表 4-1 启动方式配置

BOOT1	BOOT0	启动模式
默认值	2-3	主 FLASH 存储器
	1-2	系统存储器
经过 ISP 修改	1-2	片上 SRAM

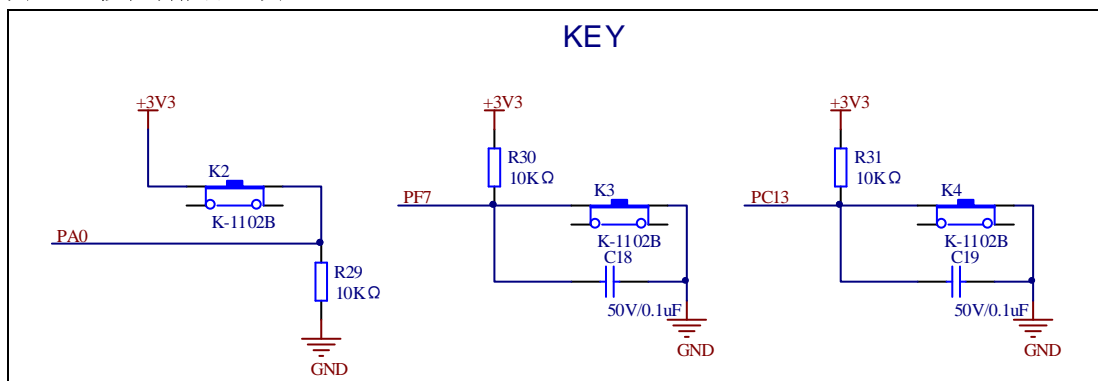
### 4.3. LED 指示灯

图 4-3 LED功能原理图



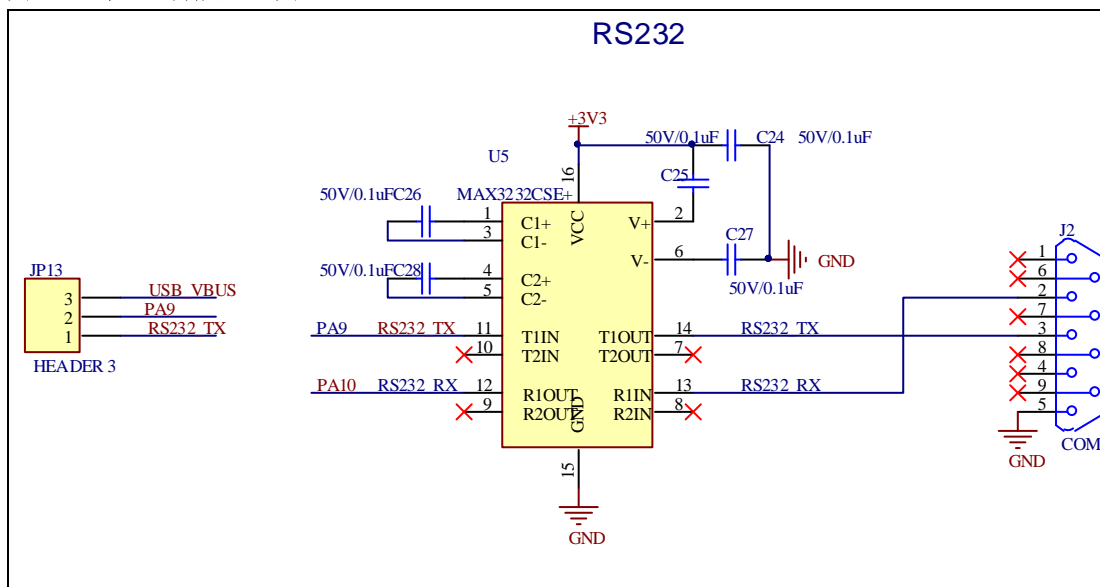
### 4.4. 按键

图 4-4 按键功能原理图



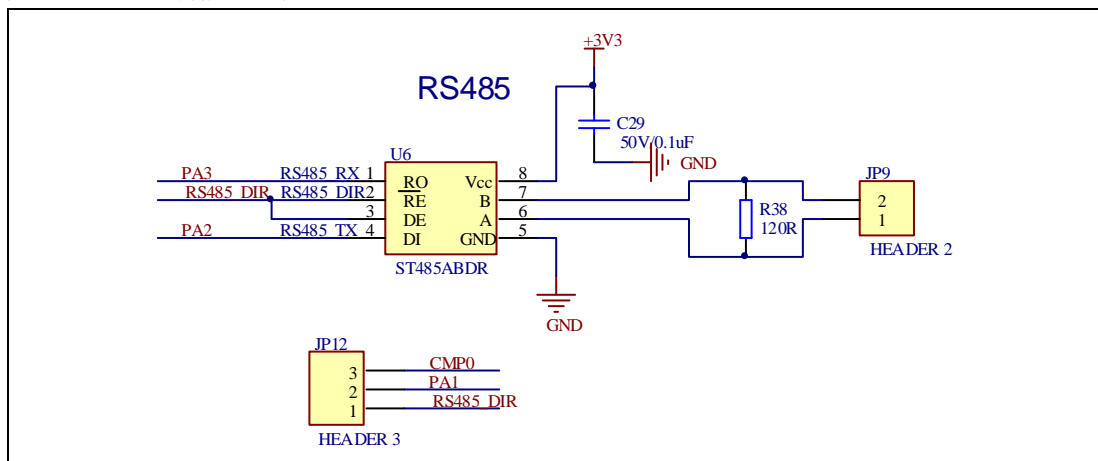
## 4.5. 串口

图 4-5 串口0功能原理图



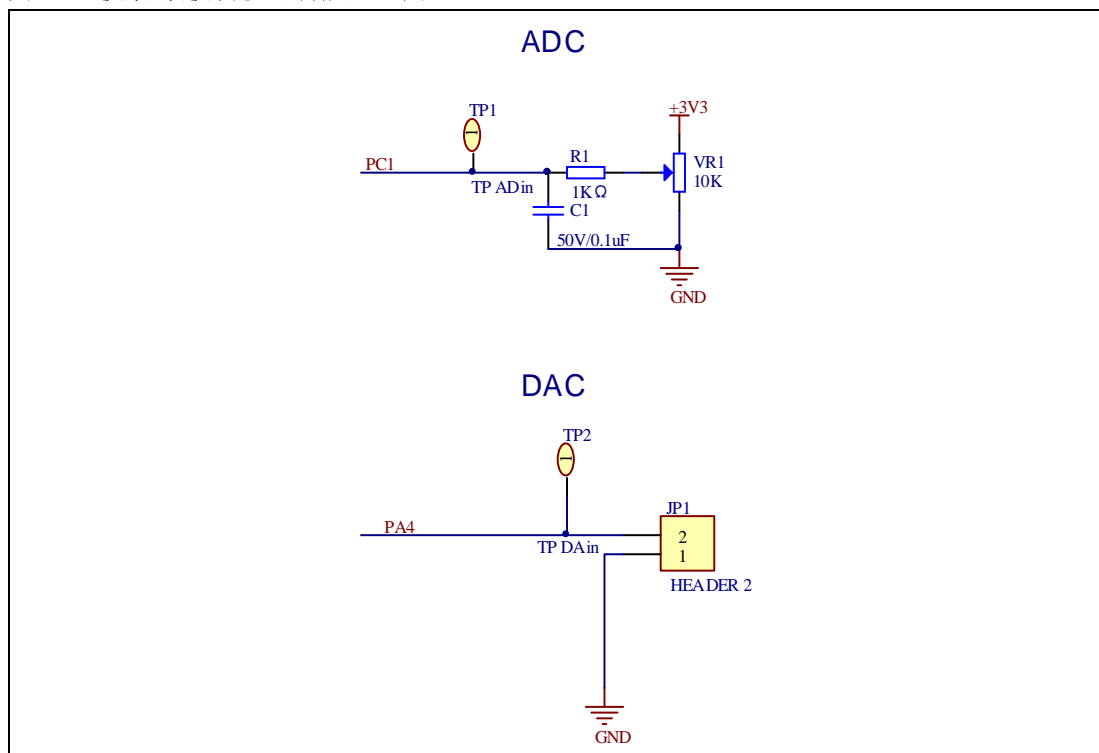
## 4.6. RS485

图 4-6 RS485功能原理图



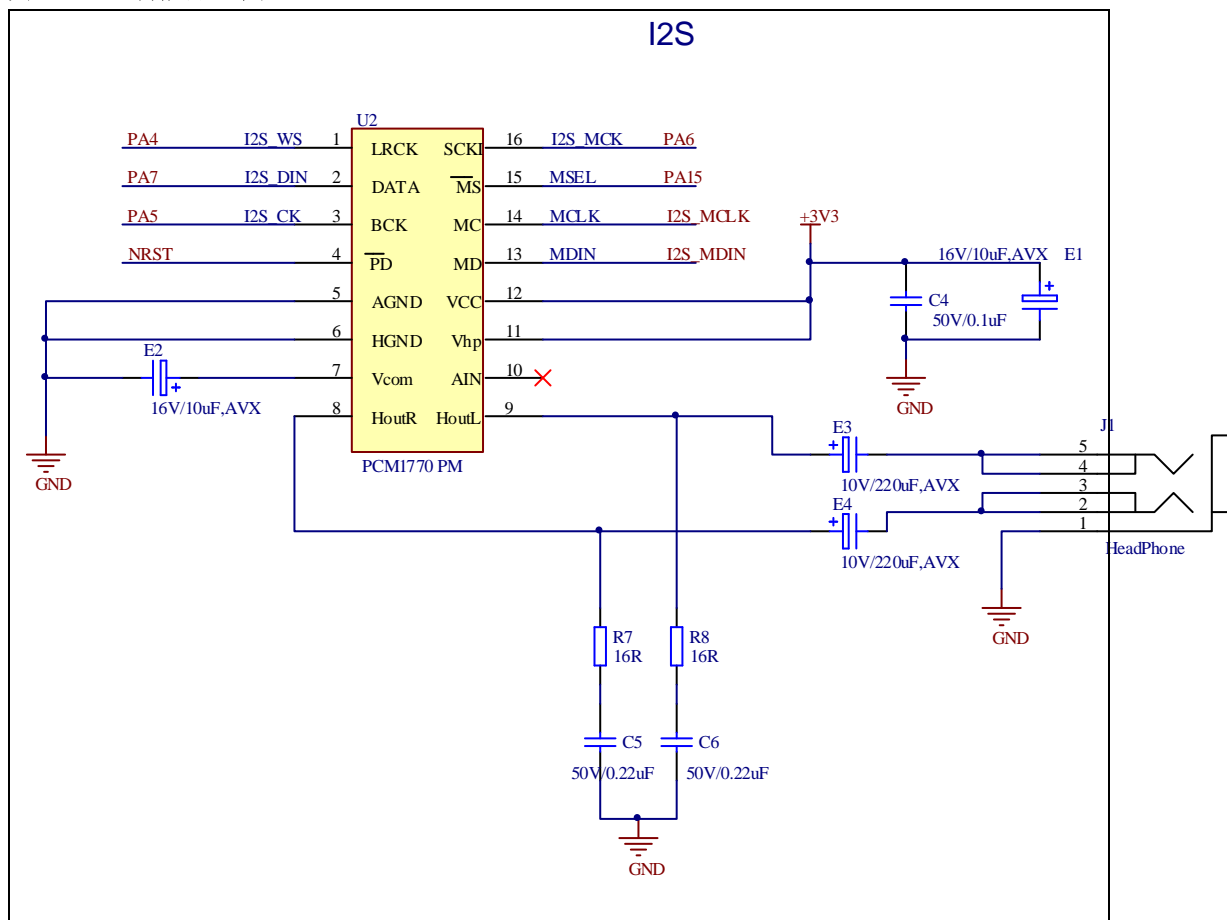
## 4.7. 模数转换器/数模转换器

图 4-7 模数/数模转换器功能原理图



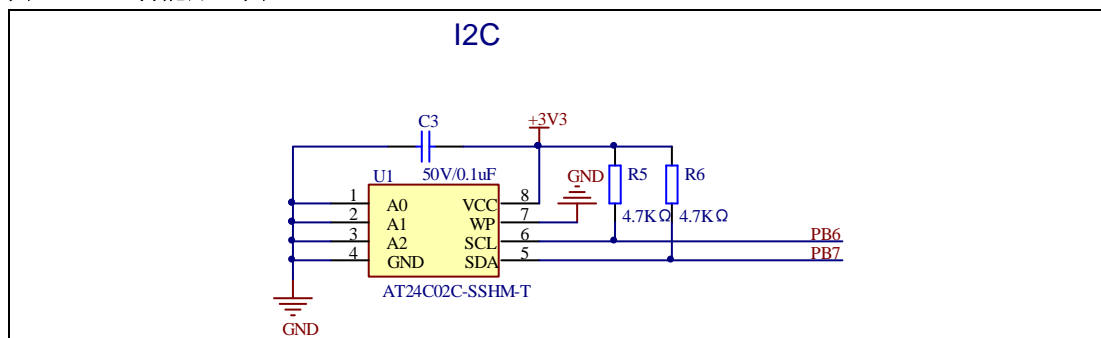
## 4.8. I2S

图 4-8 I2S功能原理图



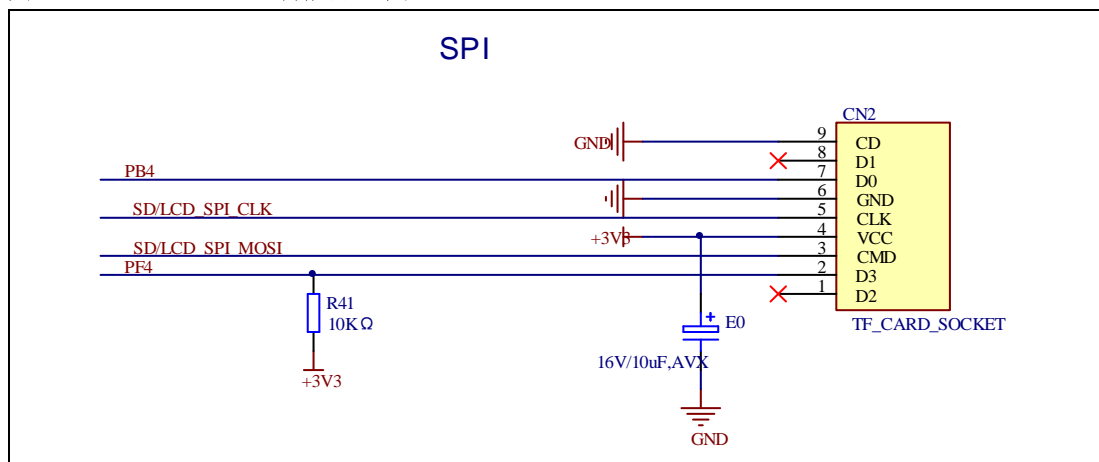
## 4.9. I2C

图 4-9 I2C功能原理图



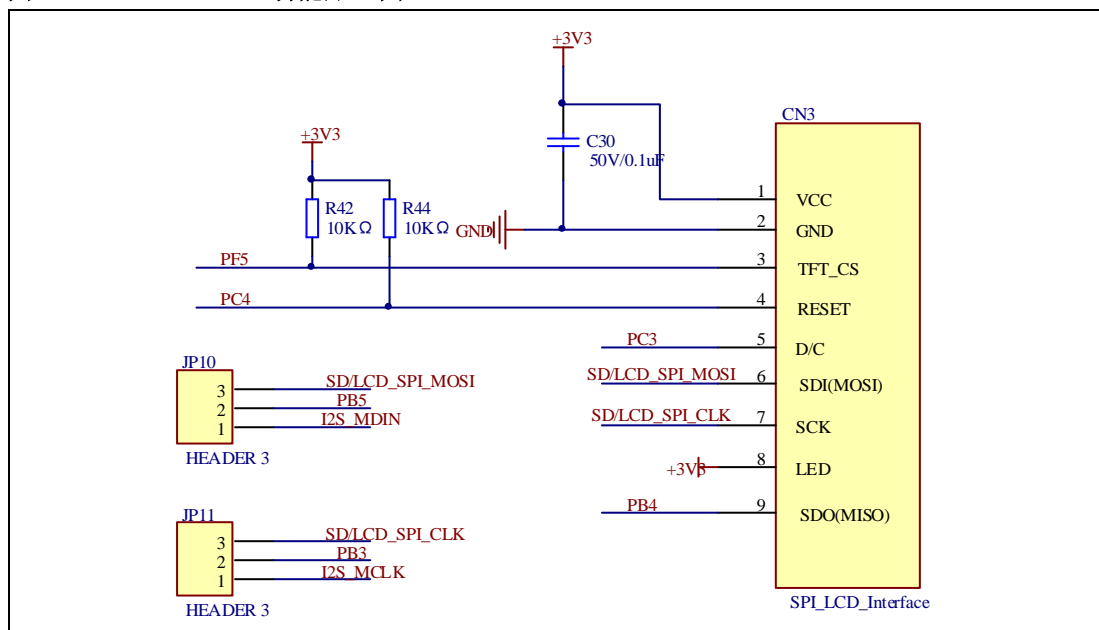
## 4.10. SPI-TF CARD

图 4-10 SPI-TF CARD功能原理图



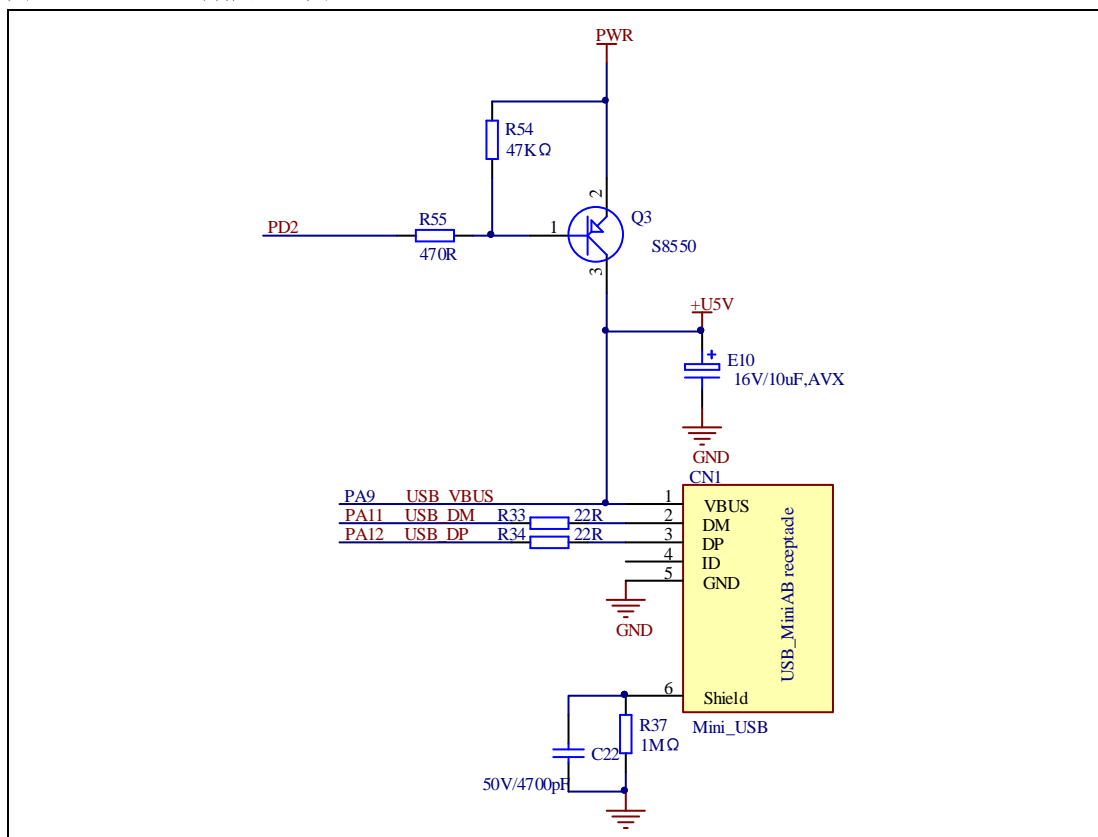
## 4.11. SPI-TFT LCD

图 4-11 SPI-TFT LCD功能原理图



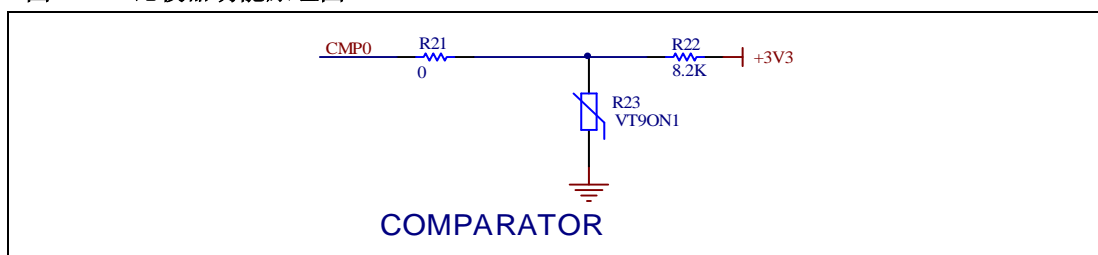
## 4.12. USBFS

图 4-12 USBFS功能原理图



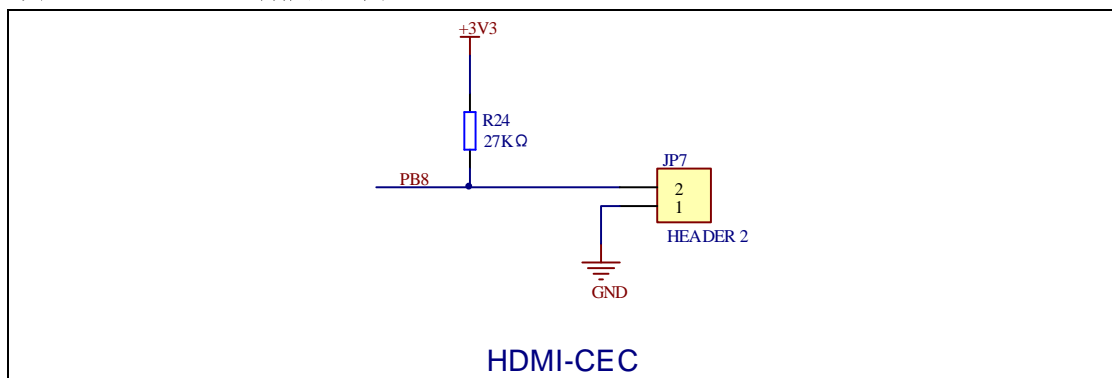
## 4.13. 比较器

图 4-13 比较器功能原理图



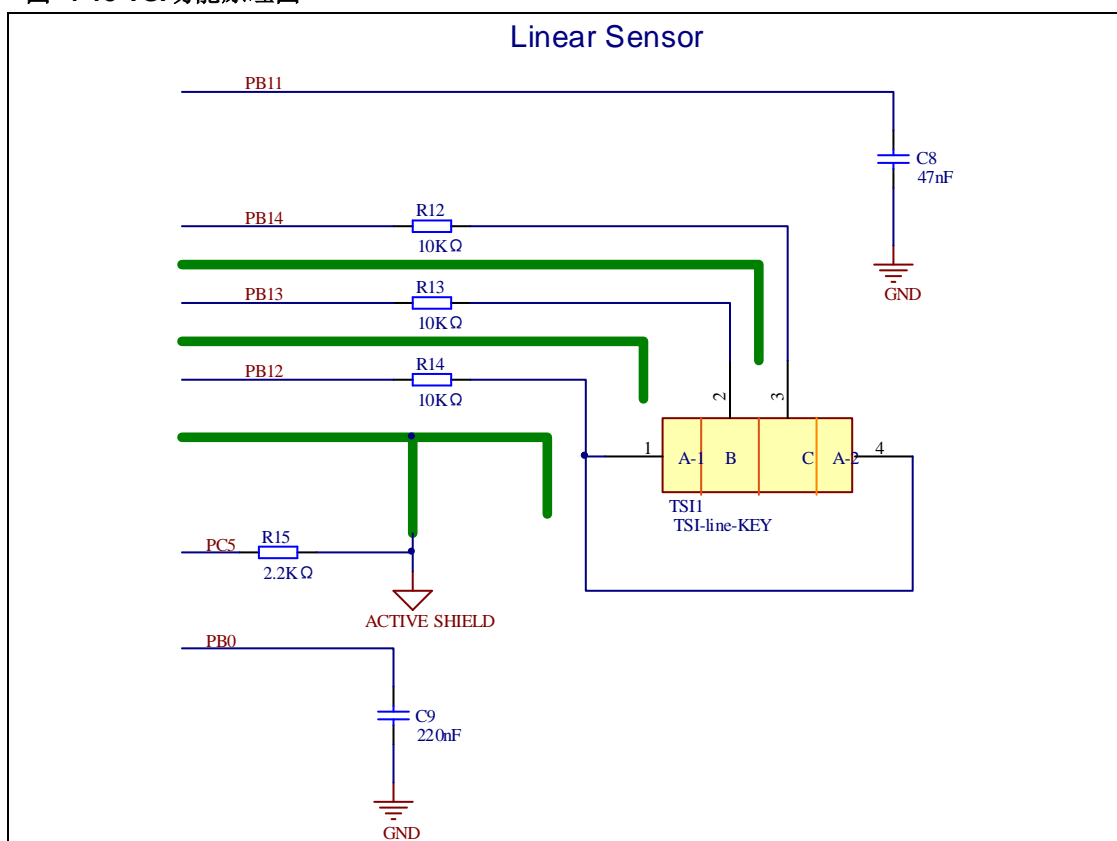
## 4.14. HDMI-CEC

图 4-14 HDMI-CEC功能原理图



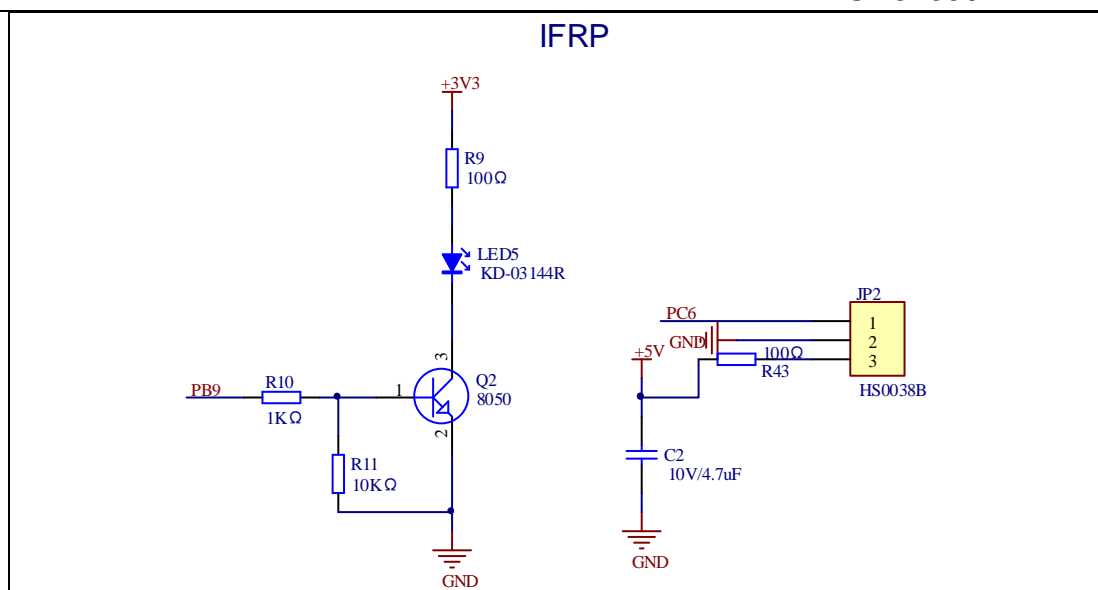
## 4.15. TSI

图 4-15 TSI功能原理图



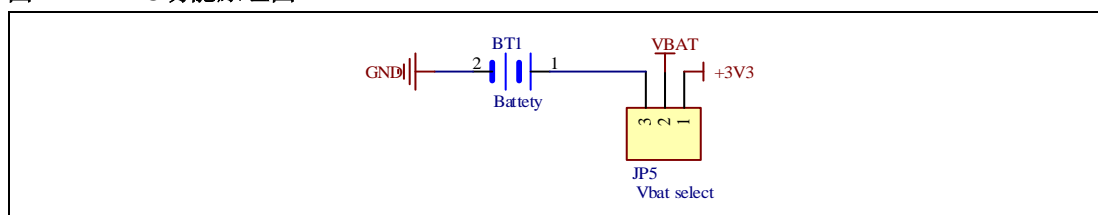
## 4.16. 红外线接口

图 4-16 红外线接口功能原理图



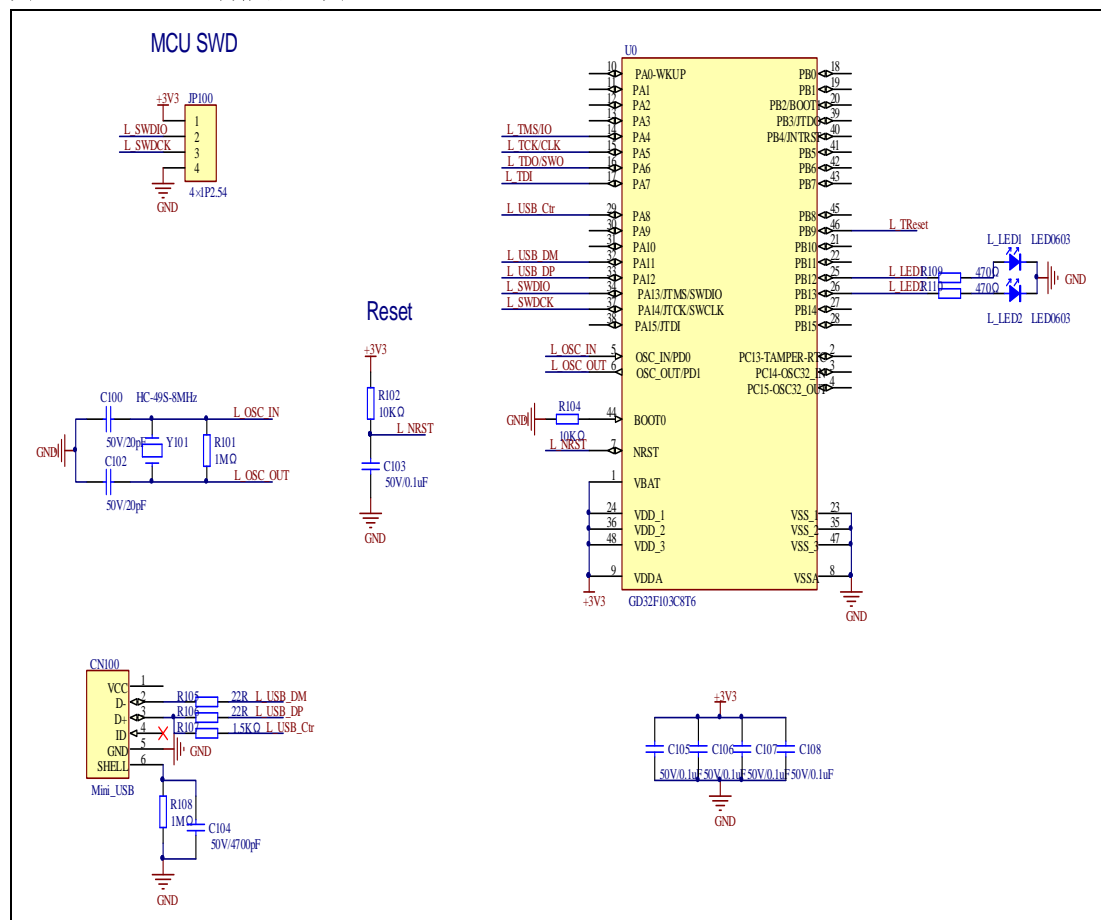
## 4.17. RTC

图 4-17 RTC功能原理图



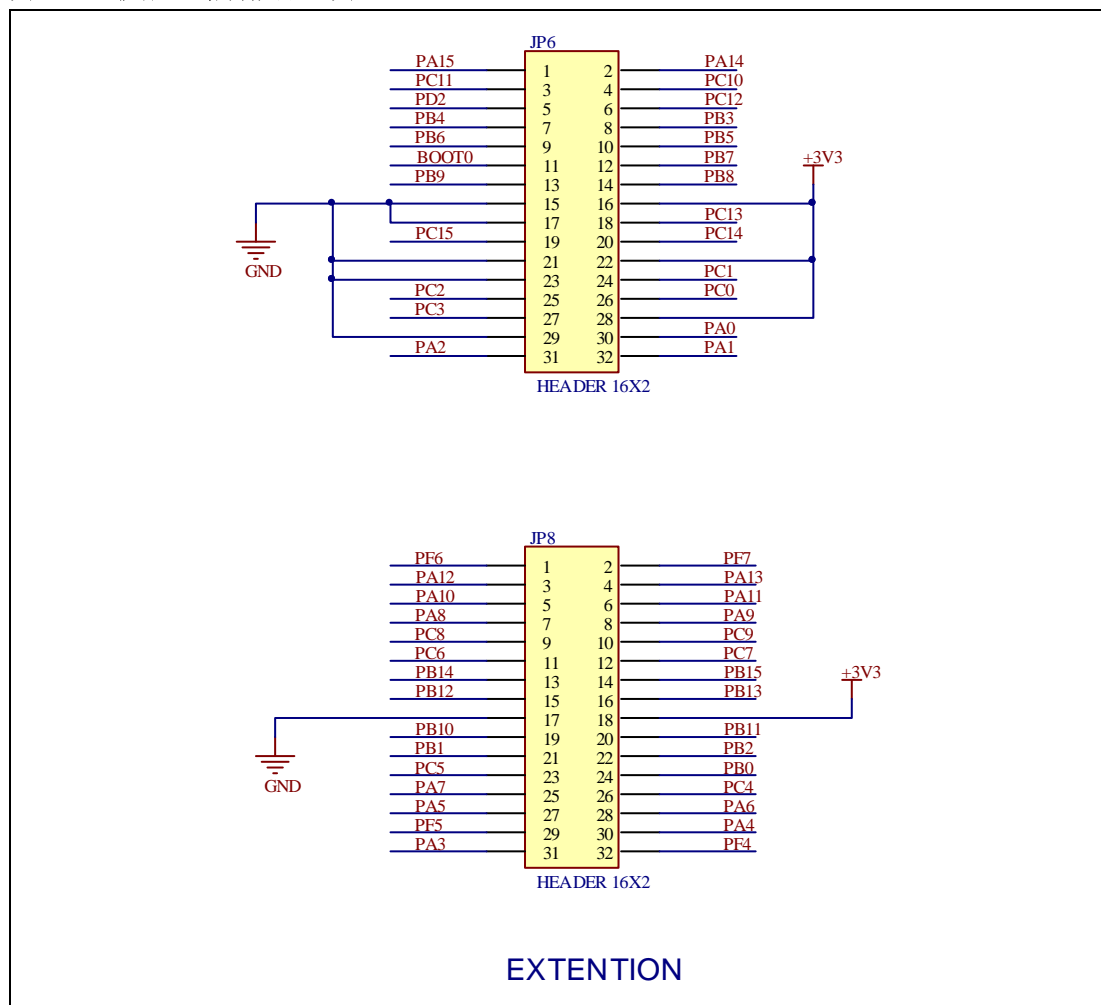
## 4.18. GD-Link

图 4-18 GD-Link功能原理图



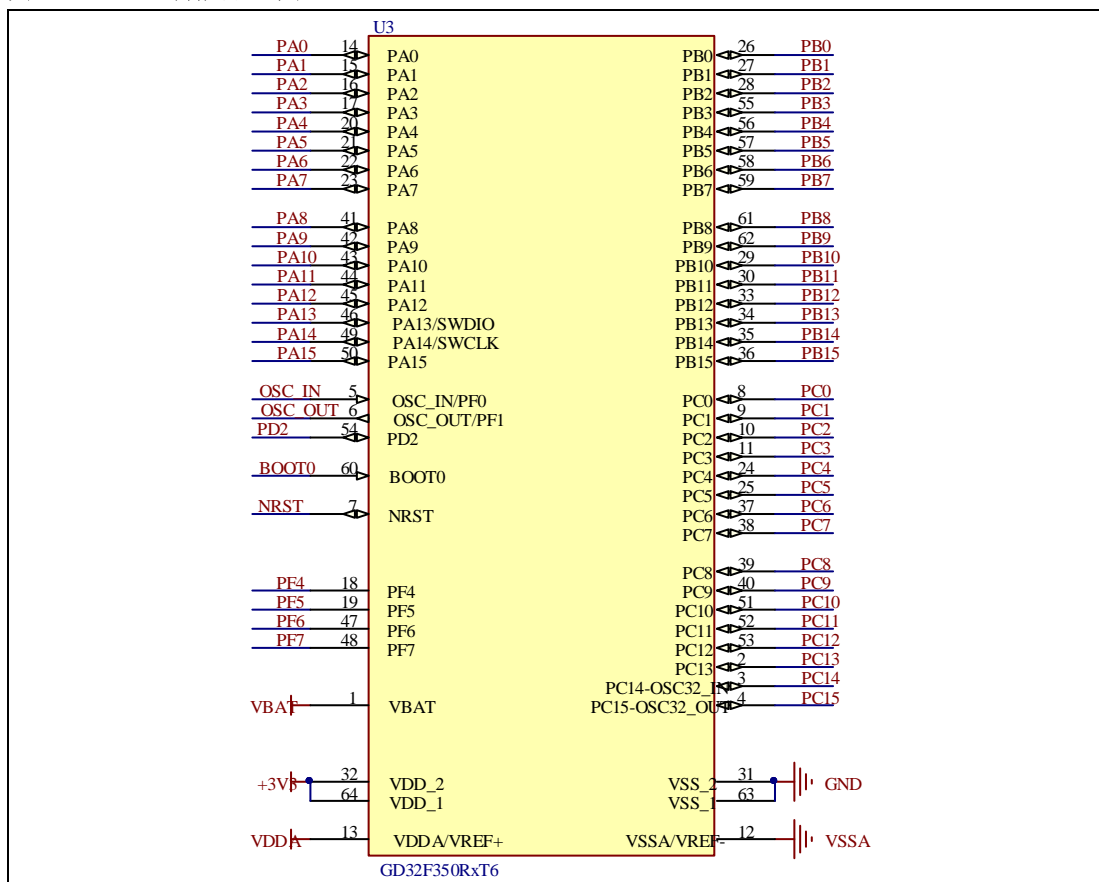
## 4.19. 扩展电路

图 4-19 扩展电路功能原理图



## 4.20. MCU

图 4-20 MCU功能原理图



## 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

GD32350R-EVAL 开发板上有 4 个 LED。LED1, LED2, LED3 和 LED4 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 到 LED4 每隔 200ms 依次点亮，然后全部熄灭，200ms 后，又重复这个过程。

## 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

GD32350R-EVAL 开发板有四个按键和四个 LED。这四个按键是 Reset 按键，Tamper 按键，User 按键，Wakeup 按键，LED1，LED2，LED3 和 LED4 通过 GPIO 控制着。

这个例程将讲述怎么使用 Tamper 按键控制 LED2。当按下 Tamper 按键，将检测 IO 端口的输入值，如果输入值为 0，将等待 50ms。再次检测 IO 端口的输入值。如果数值仍然为 0 表示按键成功按下，并点亮 LED2。

### 5.2.2. DEMO 执行结果

下载程序<02\_GPIO\_Key\_Polling\_mode>到开发板上，首先，所有的 LED 亮灭一次用于测试。然后，按下 Tamper 按键，LED2 将被点亮。再按下 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 实现控制 LED 和按键
- 学习使用 EXTI 产生外部中断

GD32350R-EVAL 有四个按键和四个 LED 灯。这四个按键分别为 Reset 按键、Wakeup 按键、User 按键和 Tamper 按键。LED1，LED2，LED3 和 LED4 通过 GPIO 控制。

这个例程实现怎样使用 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，首先，所有的 LED 亮灭一次用于测试。然后，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

使用跳线帽 JP13 跳线到 USART。下载程序< 04\_USART\_Printf >到开发板，并将串口线连到开发板的 COM 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。

```
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：

- 学习串口的发送和接收中断的使用

### 5.5.2. DEMO 执行结果

使用跳线帽 JP13 跳线到 USART。下载程序<05\_USART\_HyperTerminal\_Interrupt>到开发板并运行。首先，所有的 LED 灯亮灭一次用于测试。然后，COM 将首先输出数组 tx\_buffer 的内容（从 0x00 到 0xFF）到超级终端并等待接收数据。接收到的数据将存在数组 rx\_buffer，接收缓冲区的最大字节数为 BUFFER\_SIZE。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 和 LED2 灯亮，LED3 和 LED4 灯灭，否则，LED3 和 LED4 灯亮，LED1 和 LED2 灯灭。程序输出如下：

```

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：

- 学习串口 DMA 发送和接收功能的使用

### 5.6.2. DEMO 执行结果

使用跳线帽 JP13 跳线到 USART。下载程序 < 06\_USART\_DMA > 到开发板并运行。首先，所有的 LED 灯亮灭一次用于测试。然后，COM 将首先输出数组 tx\_buffer 的内容（从 0x00 到 0xFF）到超级终端并等待接收数据。接收到的数据将存在数组 rx\_buffer，接收缓冲区的最大字节数为 BUFFER\_SIZE。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 和 LED2 灯亮，LED3 和 LED4 灯灭，否则，LED3 和 LED4 灯亮，LED1 和 LED2 灯灭。程序输出如下：

```

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

```

## 5.7. RS485 实验

### 5.7.1. DEMO 目的

这个例程包括了 GD32 MCU 的以下功能：

- 学习使用串口 RS485 功能

### 5.7.2. DEMO 执行结果

使用跳线帽 JP12 跳线到 RS485，跳线帽 JP13 跳线到 USART。此例程需要准备两块 GD32350R-EVAL 开发板，一个作为发送方，另一个作为接收方。先使用杜邦线将通过 RS485 的 A，B 两个接口将两块开发板连接起来，然后下载程序< 07\_RS485\_Test >到开发板上运行。当按下 Wakeup 键时，设置开发板作为 RS485 发送端同时 LED2 亮，当按下 Tamper 键时，设置开发板作为 RS485 接收端同时 LED3 亮。

下载程序< 07\_RS485\_Test >到开发板并运行后，程序输出如下：

```
-----  
GD32F350R_EVAL RS485_Test  
--> Press down KEY_WAKEUP to set GD32F350R_EVAL as RS485 transmitter  
--> Press down KEY_TAMPER to set GD32F350R_EVAL as RS485 receiver  
-----
```

根据提示，设置两块开发板一为发送端（按下 Wakeup 键），一位接收端（按下 Tamper 键），发送端输出如下：

```
RS485 transmitter is enabled  
Data is being transmitted: GD32MCU  
Data is being transmitted: GD32MCU  
Data is being transmitted: GD32MCU  
Data is being transmitted: GD32MCU
```

接收端输出如下：

```
RS485 receiver is enabled  
Waiting for received data  
The received data: [GD32MCU]  
The received data: [GD32MCU]  
The received data: [GD32MCU]  
The received data: [GD32MCU]
```

## 5.8. 定时器触发模数转换

### 5.8.1. DEMO 目的

该例程包含 GD32 MCU 以下功能：

- 学会使用 ADC 转换模拟量为数字量
- 学会使用定时器生成比较事件
- 学会使用 LCD 液晶屏来显示 ADC 转换结果

定时器 1 的比较事件 1 触发 ADC 转换，ADC 转换的结果将随着模拟值输入的改变而改变。转换结果由 DMA 搬运到 SRAM 中，最后在 LCD 上画出相应曲线。

### 5.8.2. DEMO 执行结果

将<08\_ADC\_conversion\_triggered\_by\_timer>程序下载至评估板，定时器 1 的比较捕获事件 1 触发 ADC 转换，调节电位器改变输入，ADC 的转换结果将会改变，同时可以在 LCD 液晶屏上看到转换的电压值曲线随着电位器调节而变化。

## 5.9. DAC 输出电压值

### 5.9.1. DEMO 目的

这个例程包括了 GD32 MCU 的以下功能：

- 学习使用 DAC 在 DAC0\_OUT0 输出端生成电压；

### 5.9.2. DEMO 执行结果

下载程序<10\_DAC\_Output\_Voltage\_Value>至评估板并运行。

所有的 LED 灯先亮灭一次用于测试。数字量 0x7FF0，在 3.3V（VREF/2）的参考电压下，它的转换值应该为 1.65V，将会在 PA4 引脚输出。

PA4 输出的电压可以通过示波器观测。

## 5.10. 比较器输出获取指示灯

### 5.10.1. DEMO 目的

该例程包含 GD32 MCU 以下功能：

- 学会使用比较器输出比较结果

比较器有两个输入端，其中一个输入设置为 PA1，另一个是参考电压，比较这两个输入电压，输出高电平或低电平，然后 LED2 灯就会执行相应动作。

### 5.10.2. DEMO 执行结果

下载程序<10\_Comparator\_Obtain\_Brightness>到开发板中，比较两个输入电压大小，如果输出电平为高，LED2 亮，否则，LED2 灭。

## 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

使用跳线帽 JP13 跳线到 USART。下载程序<11\_I2C\_EEPROM>到开发板上。将开发板的 COM 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!
```

## 5.12. SPI TF 卡 Block 读写

### 5.12.1. DEMO 目的

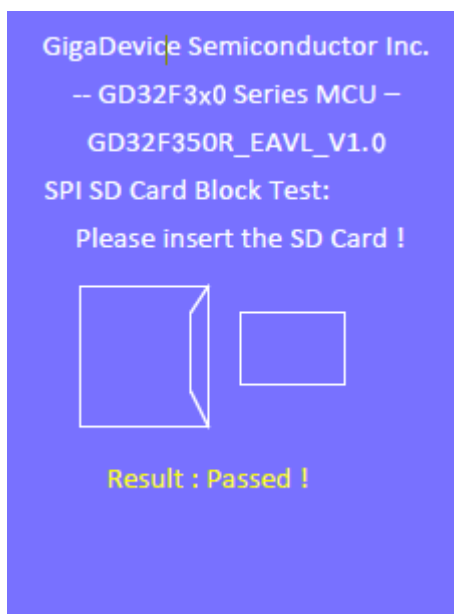
这个例程包括了 GD32 MCU 的以下功能：

- 学习使用如何利用 SPI 进行 TF 卡 Block 读写

Demo 中使用 SPI 接口，从 TF 卡中读写数据。直接向 TF 卡写入 8 次 0~255，共 2048 个字节。然后从原地址读出数据，检查写入的数据和读出的数据是否一致。需要注意的是，这个 Demo 中没有使用文件系统，在无文件系统下读写 TF 卡，将破坏 TF 原来的文件系统，请在测试前做好备份。

### 5.12.2. DEMO 执行结果

首先要确保 JP10 和 JP11 都跳线到 SPI，然后下载程序<12\_SPI\_TF\_Card\_Block\_Operation>到开发板中。所有的 LED 先被打开然后关闭，接着插入 TF 卡，不同的测试结果将会在 LCD 屏上显示。



## 5.13. SPI TF 卡 FATFS 读写

### 5.13.1. DEMO 目的

这个例程包括了 GD32 MCU 的以下功能：

- 学习使用如何利用 SPI 在有 FAT 文件系统的情况下对 TF 卡进行读写操作

Demo 中使用 SPI 接口，从 TF 卡中读写数据。确保 TF 卡有 FAT 文件系统，否则，LCD 屏上会有相应的信息提示用户。

### 5.13.2. DEMO 执行结果

首先要确保 JP10 和 JP11 都跳线到 SPI, 然后下载程序<13\_SPI\_TF\_Card\_FATFS\_Operation> 到开发板中。所有的 LED 先被打开然后关闭, 接着插入一张已经被格式化后的 TF 卡, 然后一个文本文件将会在 TF 卡中被创建。当屏幕上出现“FATFS FILE Create Success”后, 取出 TF 卡。可以在计算机上用读卡器读出数据, 测试文件是否被成功创建。当出现异常时, 请根据屏幕提示操作。



## 5.14. SPI 驱动 LCD 液晶屏

### 5.14.1. DEMO 目的

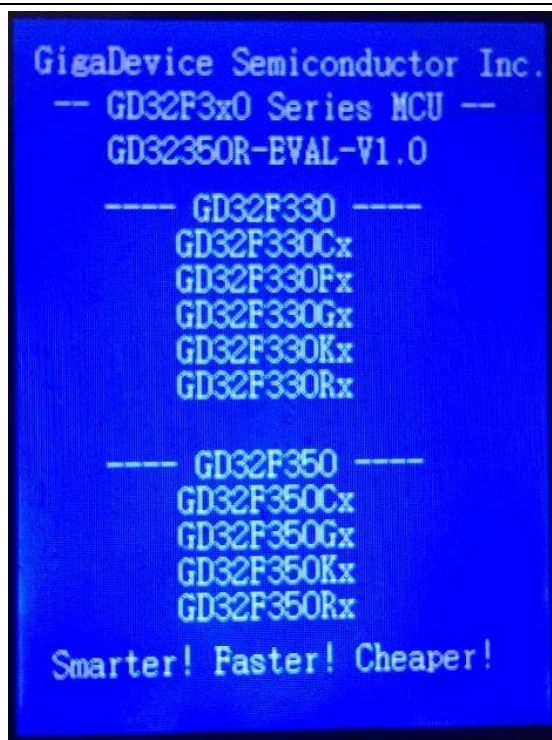
这个例程包括了 GD32 MCU 的以下功能:

- 学习使用如何利用 SPI 驱动 TFT LCD 屏并显示

GD32350R-EVAL 开发板上有一个 TFT LCD 显示屏, 它支持 SPI 接口。在这个 Demo 中, 分别进行了文字测试、数字测试、画图测试和颜色测试, 最终在 LCD 屏上显示。

### 5.14.2. DEMO 执行结果

首先要确保 JP10 和 JP11 都跳线到 SPI 端, 然后下载程序<14\_SPI\_TFT\_LCD\_Driver>到开发板并运行。所有的 LED 先被打开然后关闭, 接着 LCD 屏循环显示 GUI 测试项目。



## 5.15. HDMI\_CEC 通信

### 5.15.1. DEMO 目的

这个例程包括了 GD32 MCU 的以下功能：

- 学习 HDMI-CEC 的通信功能

通信过程中通过按键中断发送方将数据发给接收方，接收方在 CEC 中断中进行数据接收。整个通信过程中不作错误处理。

### 5.15.2. DEMO 执行结果

此例程需要准备两个 GD32350R 开发板，一个作为发送方，另一个作为接收方。先使用杜邦线将两块开发板上的 HDMI-CEC 的 CEC 总线（PB8）和地线（GND）两个引脚连接起来，然后下载程序<15\_HDMI\_CEC\_HostSlaveCommunication>到开发板上运行。当程序运行时，首先开发板的 LCD 显示的都是数据 0，按下其中一块开发板的 TAMPER 键，另一块开发板上 LCD 的数字会递增，这说明一次数据传输结束。每次递增到 9 后会清 0 重新递增；按下 WAKEUP 键后，数字会递减，这同样说明一次数据传输结束。每次递减到 0 后会回到数字 9 重新递减。

## 5.16. 音频播放器

### 5.16.1. DEMO 目的

这个例程包括了 GD32 MCU 的以下功能：

- 学习使用 I2S 接口输出音频文件

GD32350R-EVAL 开发板集成了 I2S 模块，该模块可以和外部设备通过音频协议通信。这个例程演示了如何通过开发板的 I2S 接口播放音频文件。

### 5.16.2. DEMO 执行结果

首先要确保 JP10 和 JP11 都跳线到 I2S 端，然后下载程序 <16\_I2S\_Audio\_Player> 到开发板并运行，将耳机插到 J1 端口，即可听到播放的音频文件声音。

## 5.17. RCU 时钟输出

### 5.17.1. 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

### 5.17.2. DEMO 执行结果

下载程序<17\_RCU\_Clock\_Out>到开发板上并运行。将开发板的 COM 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 USER 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 user key to select clock output source
CK_OUT: IRC28M, DIV:1
CK_OUT: IRC40K, DIV:1
CK_OUT: LXTAL, DIV:1
CK_OUT: CKSYS, DIV:4
CK_OUT: IRC8M, DIV:1
CK_OUT: HXTAL, DIV:1
CK_OUT: CKPLL, DIV:4
```

## 5.18. CTC 校准

### 5.18.1. DEMO 目的

这个例程包括了 GD32 MCU 的以下功能：

- 学习使用外部晶振 LXTAL 来实现 CTC 校准功能
- 学习使用 CTC 校准控制器校准内部 48MHz RC 振荡器时钟

CTC 模块基于外部高精度的参考信号源来校准 IRC48M 的时钟频率，通过自动的或手动的调整校准值，可以得到一个精准的 IRC48M 时钟。

### 5.18.2. DEMO 执行结果

下载程序<18\_CTC\_Calibration>到开发板上，运行程序，如果内部 48MHz RC 校准成功，LED1 将会点亮。

## 5.19. PMU 睡眠模式唤醒

### 5.19.1. 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

### 5.19.2. DEMO 执行结果

下载程序<19\_PMU\_sleep\_wakeup>到开发板上，并将串口线连到开发板的 COM 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。

## 5.20. RTC 实时时钟

### 5.20.1. DEMO 目的

这个例程包括了 GD32 MCU 的以下功能：

- 学习使用 RTC 模块实现实时时钟的功能
- 学习使用串口模块实现显示时间的功能

### 5.20.2. DEMO 执行结果

使用跳线帽 JP13 跳线到 USART，下载程序<20\_PMU\_sleep\_wakeup>到开发板上，并将串

口线连到开发板的 COM 上。串口输出信息如下图所示，如果开发板是第一次运行该程序，串口输出如下信息“RTC not yet configured....”，要求用户进行时间设置。

```

XXXXXXXXXXXXXXXX RTC calendar demo XXXXXXXXXXXXXXXXXXXX
=====Configure RTC Time=====
please input hour:

```

根据串口输出信息提示，设置时间后，串口会打印信息如下图所示：

```
***** RTC calendar demo *****
=====Configure RTC Time=====
please input hour:
12
please input minute:
00
please input second:
00
** RTC time configuration success! **
Current time: 12:00:00
```

如果开发板此前已经设置好时间，在系统复位后，如下图所示，串口会输出提示“No need to configure RTC....”，串口继续打印时间信息。

```

XXXXXXXXXXXXXXXXXXXX RTC calendar demo XXXXXXXXXXXXXXXXXXXX
power on reset occurred...
no need to configure RTC....
Current time: 12:02:41

```

## 5.21. 红外收发器

### 5.21.1. DEMO 目的

这个例程包括了 GD32 MCU 的以下功能：

- 学习使用通用定时器输出 PWM 的方法
- 学习使用通用定时器更新中断的方法
- 学习使用通用定时器捕获中断功能
- 学习使用定时器 TIMER15 和 TIMER16 实现红外功能

### 5.21.2. DEMO 执行结果

下载程序<21\_IRInfrared\_Transceiver>到开发板上并运行。当程序运行时，如果红外接收器接收到正确信号，可以看到 LED1~LED4 依次点亮，否则，可以看到 LED1~LED4 同时翻转，即同时点亮和熄灭。

## 5.22. TIMER 呼吸灯

### 5.22.1. 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

### 5.22.2. DEMO 执行结果

使用杜邦线连接 TIMER0\_CH0（PA8）和 LED1（PC10），然后下载程序<22\_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 LED1 由暗变亮，由亮变暗，往复循环，就像人的呼吸一样有节奏。

## 5.23. TSI 触摸按键

### 5.23.1. DEMO 目的

这个例程包括了 GD32 MCU 的以下功能：

- 学习使用 TSI 模块实现触摸按键的功能

### 5.23.2. DEMO 执行结果

使用跳线帽 JP10 和 JP11 跳到 SPI，下载程序<23\_TSI\_TouchKey\_leds>到开发板上并运行。当程序运行时，使用手指触摸 TouchKey（A1，B，C 和 A2），然后相应的 LED 灯点亮且 LCD 显示当前进度条。

## 5.24. USB 设备

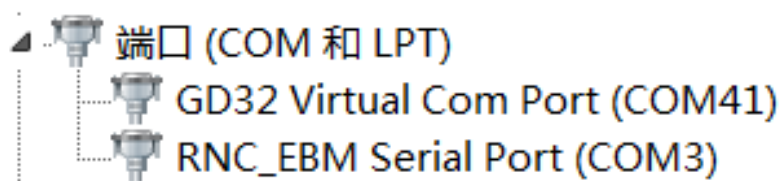
### 5.24.1. 设备虚拟串口

#### DEMO 目的

这个例程包括了 GD32 MCU 的以下功能：

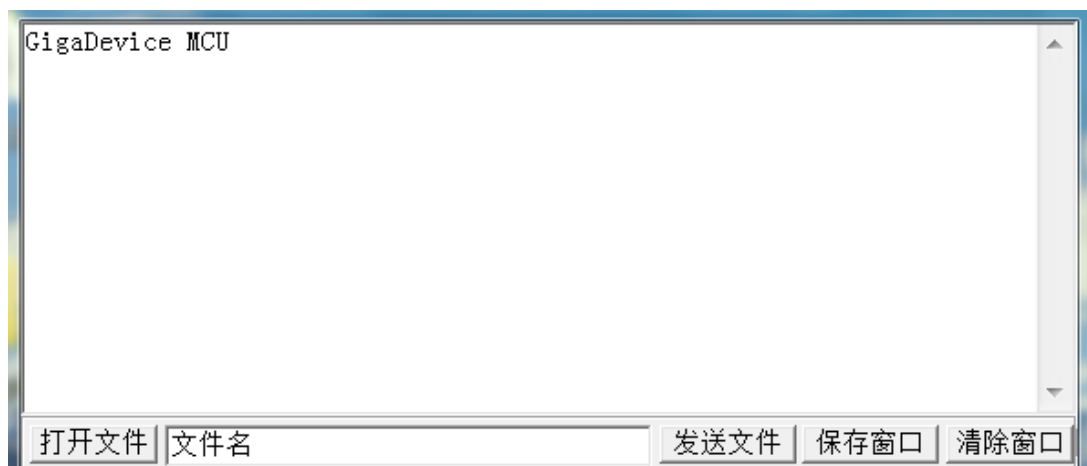
- 学习如何使用 USBFS 设备
- 学习如何实现 USBFS CDC 设备

开发板具有一个 USBFS 接口。在本例程中，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。



#### DEMO 执行结果

将<24\_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。



## 5.24.2. 设备键盘

### DEMO 目的

这个例程包括了 GD32 MCU 的以下功能：

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

开发板具有四个按键和一个 USBFS 接口，这四个按键分别是 Reset 按键、Wakeup 按键、Tamper 按键、User 按键。在本例程中，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，USB 键盘利用 Wakeup 键、Tamper 键和 User 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

在运行程序之前，确保将 JP13 跳到 USB，然后将 <24\_USB\_Device\HID\_Keyboard > 例程下载到开发板中，并运行。按下 Wakeup 键，输出 'b'；按下 User 键，输出 'c'；按下 Tamper 键，输出 'a'。

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。

## 5.25. USB 主机

### 5.25.1. HID 主机

#### DEMO 目的

这个例程包括了 GD32 MCU 的以下功能：

- 学习使用 USBFS 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

评估板内部包含 USBFS 模块，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 USBFS 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 JP13 引脚跳到 USB，将<25\_USB\_Host\USBH\_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，将会在液晶上看到鼠标的位置和按键的状态。

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，将会通过液晶显示按键状态。

### 5.25.2. 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 作为 MSC 主机
- 学习 MSC 主机和 U 盘之间的操作

评估板包含 USBFS 模块，并且该模块可以被用于作为一个 USB 设备、一个 USB 主机或一个 OTG 设备。本示例主要显示如何使用 USBFS 作为一个 USB MSC 主机来与外部 U 盘进行通信。

#### DEMO 执行结果

将 JP13 引脚跳到 USB。然后将 OTG 电缆线插入到 USB 接口，将<25\_USB\_Host\USBH\_MSC > 工程下载到开发板中并运行。

如果一个 U 盘被连入，用户将会看到 U 盘枚举信息。首先按下 **User** 按键将会看到 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。

## 6. 版本更新历史

表 6-1 版本更新历史

版本号	说明	日期
1.0	初稿发布	Jun.28, 2017
2.0	更新文档格式	Jun.1, 2019
2.1	更新 <b>5.9</b> 和 <b>5.10</b> 小节内容	Dec.31, 2023

## Important Notice

This document is the property of GigaDevice Semiconductor Inc. and its subsidiaries (the "Company"). This document, including any product of the Company described in this document (the "Product"), is owned by the Company under the intellectual property laws and treaties of the People's Republic of China and other jurisdictions worldwide. The Company reserves all rights under such laws and treaties and does not grant any license under its patents, copyrights, trademarks, or other intellectual property rights. The names and brands of third party referred thereto (if any) are the property of their respective owner and referred to for identification purposes only.

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

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