

GigaDevice Semiconductor Inc.

GD32H75EY-EVAL 评估板
用户指南
Rev1.0

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.....	8
4.4. 按键	9
4.5. ADC	9
4.6. DAC	9
4.7. CAN	10
4.8. HPDF	11
4.9. I2C.....	11
4.10. TIMER.....	12
4.11. I2S	12
4.12. OSPI	13
4.13. SPI.....	13
4.14. USART.....	14
4.15. ETHC	15
4.16. USB	15
4.17. Extension.....	16
4.18. GD-Link.....	17
4.19. MCU.....	18
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

5.2. GPIO 按键轮询模式	19
5.2.1. DEMO 目的	19
5.2.2. DEMO 执行结果	19
5.3. EXTI 按键中断模式	20
5.3.1. DEMO 目的	20
5.3.2. DEMO 执行结果	20
5.4. 串口打印	20
5.4.1. DEMO 目的	20
5.4.2. DEMO 执行结果	20
5.5. 串口中断收发	21
5.5.1. DEMO 目的	21
5.5.2. DEMO 执行结果	21
5.6. 串口 DMA 收发	21
5.6.1. DEMO 目的	21
5.6.2. DEMO 执行结果	21
5.7. ADC 温度传感器_内部参考电压	22
5.7.1. DEMO 目的	22
5.7.2. DEMO 执行结果	22
5.8. ADC0 和 ADC1 跟随模式	23
5.8.1. DEMO 目的	23
5.8.2. DEMO 执行结果	23
5.9. ADC0 和 ADC1 常规并行模式	24
5.9.1. DEMO 目的	24
5.9.2. DEMO 执行结果	24
5.10. DAC 输出电压值	24
5.10.1. DEMO 目的	24
5.10.2. DEMO 执行结果	25
5.11. I2C 访问 EEPROM	25
5.11.1. DEMO 目的	25
5.11.2. DEMO 执行结果	25
5.12. HPDF_I2S 音频播放	26
5.12.1. DEMO 目的	26
5.12.2. DEMO 执行结果	27
5.13. SPI FLASH DMA	27
5.13.1. DEMO 目的	27
5.13.2. DEMO 执行结果	27
5.14. OSPI 八线 Flash	28
5.14.1. DEMO 目的	28
5.14.2. DEMO 执行结果	28

5.15. CAN 网络通信	29
5.15.1. DEMO 目的	29
5.15.2. DEMO 执行结果	30
5.16. RCU 时钟输出	30
5.16.1. DEMO 目的	30
5.16.2. DEMO 执行结果	30
5.17. PMU 睡眠模式唤醒	31
5.17.1. DEMO 目的	31
5.17.2. DEMO 执行结果	31
5.18. RTC 日历	31
5.18.1. DEMO 目的	31
5.18.2. DEMO 执行结果	31
5.19. 高级 TIMER PWM 输出	31
5.19.1. DEMO 目的	31
5.19.2. DEMO 执行结果	31
5.20. TIMER 呼吸灯	32
5.20.1. DEMO 目的	32
5.20.2. DEMO 执行结果	32
5.21. USB 设备	32
5.21.1. 虚拟串口	32
5.21.2. HID_键盘	33
5.22. USB 主机	34
5.22.1. USB HID 主机	34
5.22.2. USB MSC 主机	35
5.23. EtherCAT	36
5.23.1. DEMO 目的	36
5.23.2. DEMO 执行结果	36
6. 版本历史	37

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	8
图 4-4. 按键功能原理图	9
图 4-5. ADC 原理图	9
图 4-6. DAC 原理图	9
图 4-7. CAN 原理图	10
图 4-8. HPDF 原理图	11
图 4-9. I2C 原理图	11
图 4-10. TIMER 原理图	12
图 4-11. I2S 原理图	12
图 4-12. OSPI 原理图	13
图 4-13. SPI 原理图	13
图 4-14. USART 原理图	14
图 4-15. ETHC 原理图	15
图 4-16. USB 原理图	15
图 4-17. Extension 原理图	16
图 4-18. GD-Link 原理图	17
图 4-19. MCU 原理图	18

表

表 2-1. 引脚分配.....	6
表 6-1. 版本历史.....	37

1. 简介

GD32H75EY-EVAL 评估板使用 GD32H75EYMJ6 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, ETHC, HPDF, I2S, I2C, SPI, OSPI, USB 和 USART 转 USB 接口等外设资源。更多关于开发板的资料可以查看 GD32H75EY-EVAL-V1.0 原理图。

2. 功能引脚分配

表2-1. 引脚分配

功能	引脚	描述
LED	PC3	LED3
	PC4	LED4
	PC5	LED5
RESET	NRST	Reset
KEY	PA0	Wakeup
	PC13	Tamper
	PA2	User
ADC	PC0	ADC012_IN10
DAC	PA5	DAC0_OUT1
CAN	PB8	CAN0_RX
	PB9	CAN0_TX
	PB5	CAN1_RX
	PB13	CAN1_TX
	PF6	CAN2_RX
	PF7	CAN2_TX
HPDF	PB0	HPDF_CKOUT
	PB1	HPDF_DATAIN1
I2C	PH1	I2C3_SCL
	PH2	I2C3_SDA
TIMER	PF11	TIMER23_CH0
	PF12	TIMER23_CH1
	PF13	TIMER23_CH2
	PF14	TIMER23_CH3
	PE8	TIMER0_MCH0
	PE9	TIMER0_CH0
	PE10	TIMER0_MCH1
	PE11	TIMER0_CH1
	PE12	TIMER0_MCH2
	PE13	TIMER0_CH2

功能	引脚	描述
	PC9	TIMER0_MCH3
	PE14	TIMER0_CH3
	PE15	TIMER0_BRKIN0
I2S	PC12	I2S2_SD
	PC10	I2S2_CK
	PA4	I2S2_WS
	PC7	I2S2_MCK
OSPI	PB6	OSPIM_P0_NCS
	PB2	OSPIM_P0_CLK
	PD11	OSPIM_P0_IO0
	PC12	OSPIM_P0_IO1
	PA3	OSPIM_P0_IO2
	PD13	OSPIM_P0_IO3
	PD4	OSPIM_P0_IO4
	PD5	OSPIM_P0_IO5
	PD6	OSPIM_P0_IO6
	PD7	OSPIM_P0_IO7
SPI	PH6	SPI4_SCK
	PH5	SPI4_NSS
	PF9	SPI4_MOSI
	PH7	SPI4_MISO
	PH8	SPI4_IO2
	PH9	SPI4_IO3
USART	PB10	USART2_TX
	PB11	USART2_RX
ETHC	SYNC0_LATCH0	LATCH0
	SYNC1_LATCH1	LATCH1
USB	PA9	USBHS0_VBUS
	PA11	USB0_HS_DM
	PA12	USB0_HS_DP
	PB12	USBHS1_VBUS
	PB14	USB1_HS_DM
	PB15	USB1_HS_DP

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，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，Keil 版的工程是基于 Keil MDK-ARM 5.29 uVision5 创

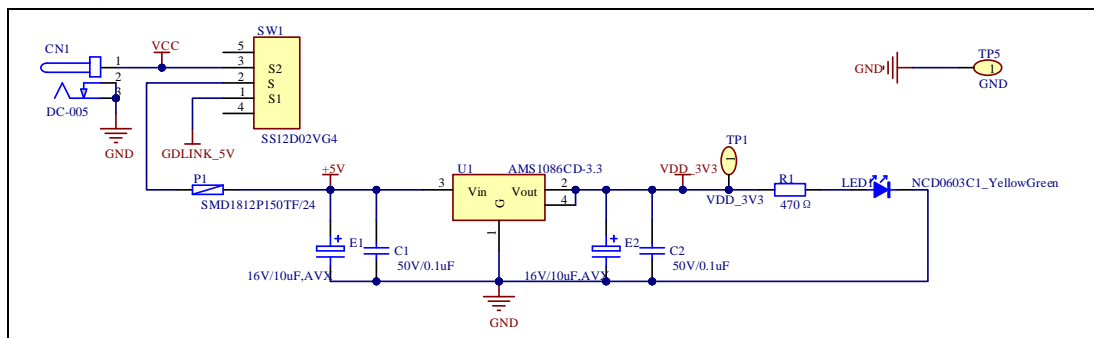
建的，IAR 版的工程是基于 IAR Embedded Workbench for ARM 8.32.1 创建的。在使用过程中有如下几点需要注意：

1. 使用 Keil uVision5 打开工程，安装（网址：<https://www.gd32mcu.com>）最新版本 GigaDevice.GD32H75E_DFP，以加载相关文件。
2. 如果使用 IAR 打开工程，安装（网址：<https://www.gd32mcu.com>）最新版本 IAR_GD32H75E_ADDON，以加载相关文件。

4. 硬件设计概述

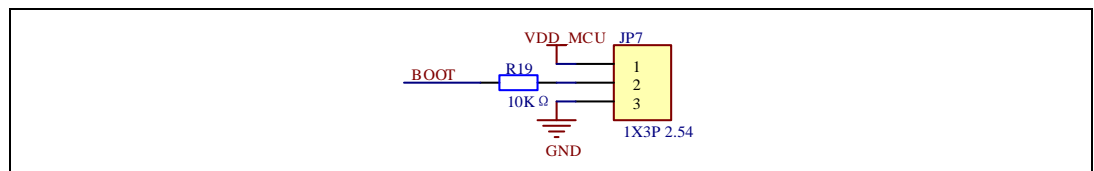
4.1. 供电电源

图4-1. 供电电源原理图



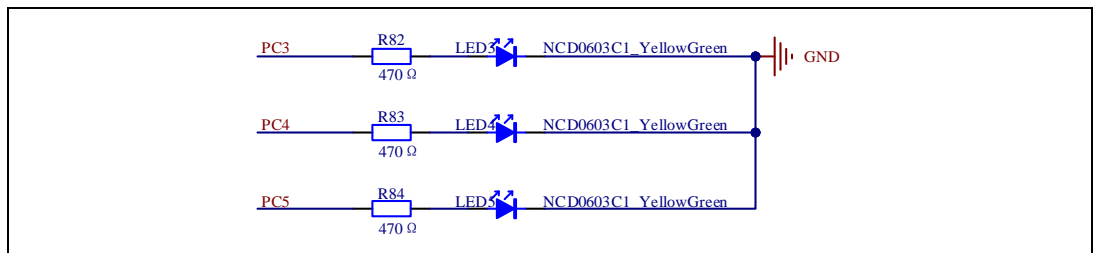
4.2. 启动方式选择

图4-2. 启动方式选择原理图



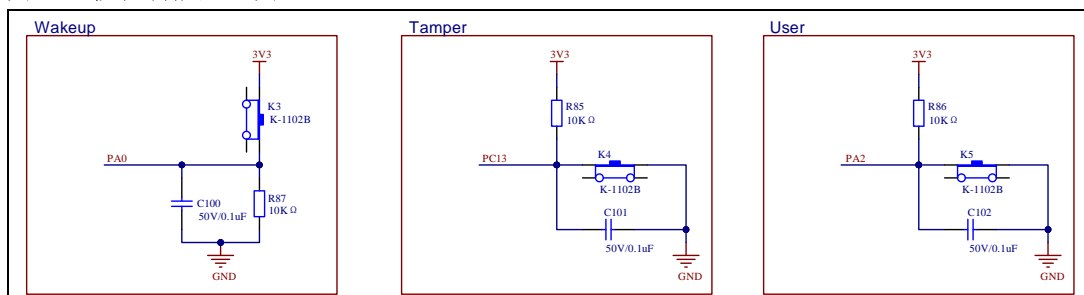
4.3. LED 指示灯

图4-3. LED功能原理图



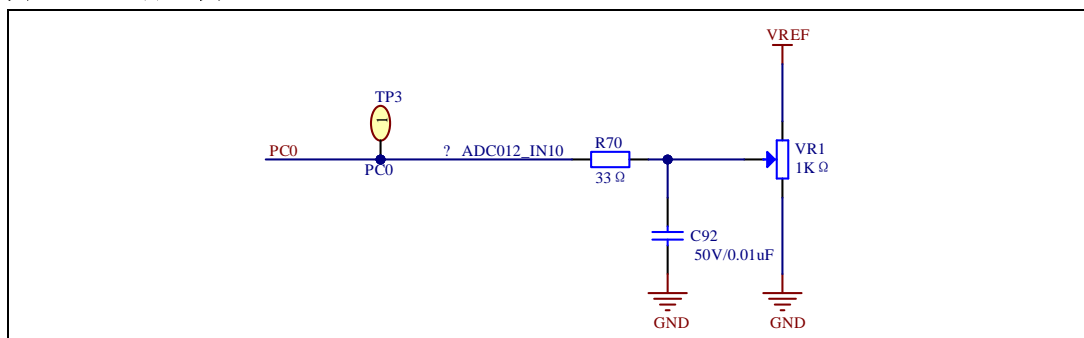
4.4. 按键

图4-4. 按键功能原理图



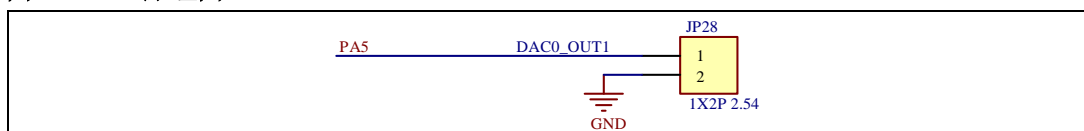
4.5. ADC

图4-5. ADC原理图



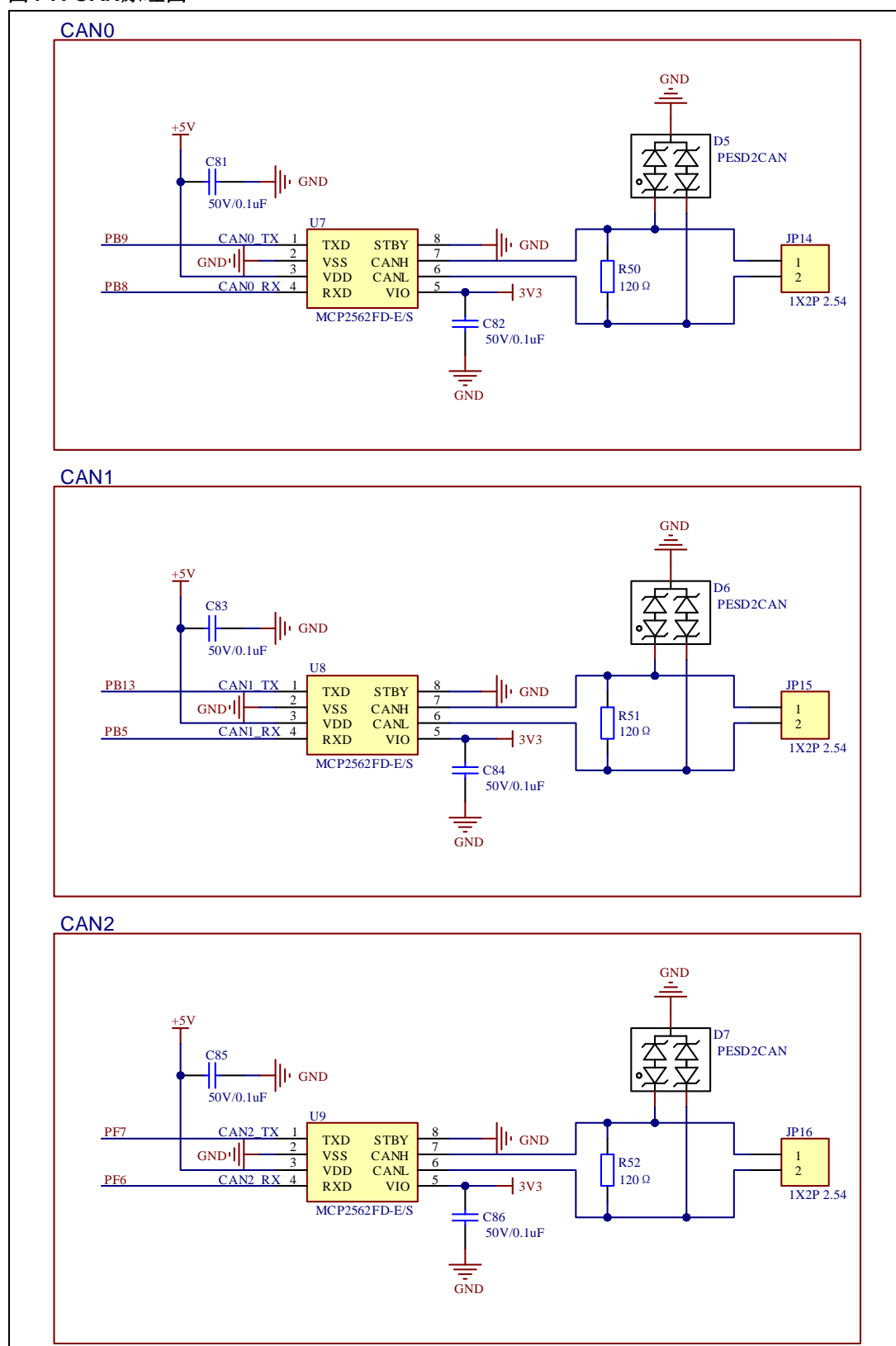
4.6. DAC

图4-6. DAC原理图



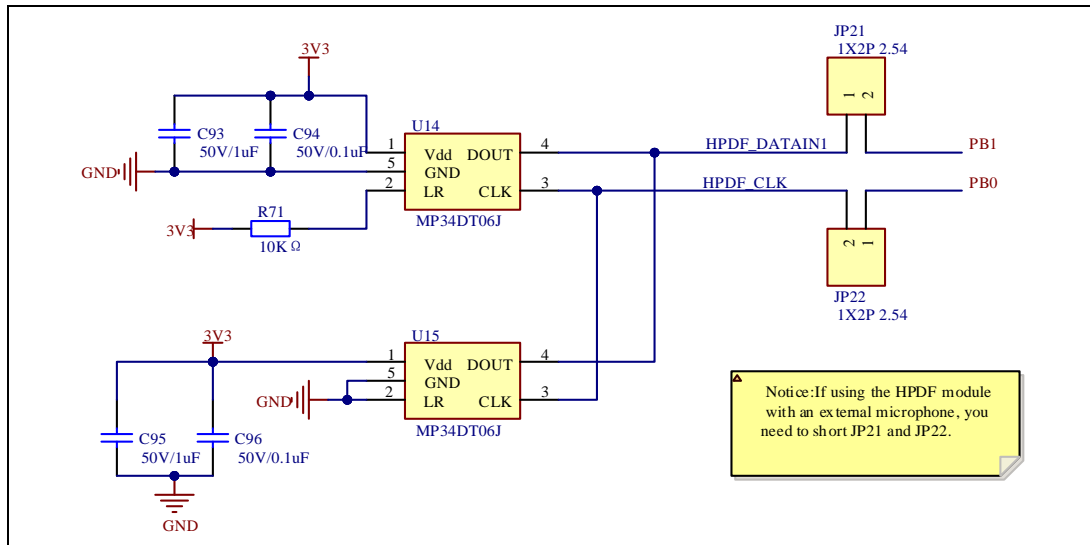
4.7. CAN

图4-7. CAN原理图



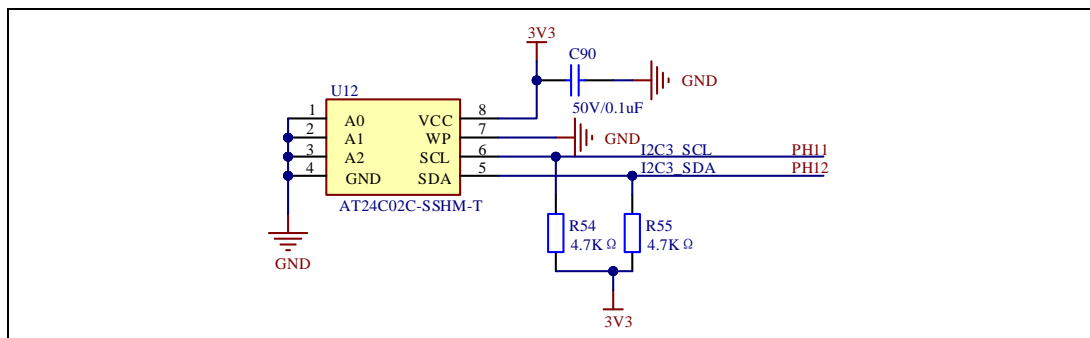
4.8. HPDF

图4-8. HPDF原理图



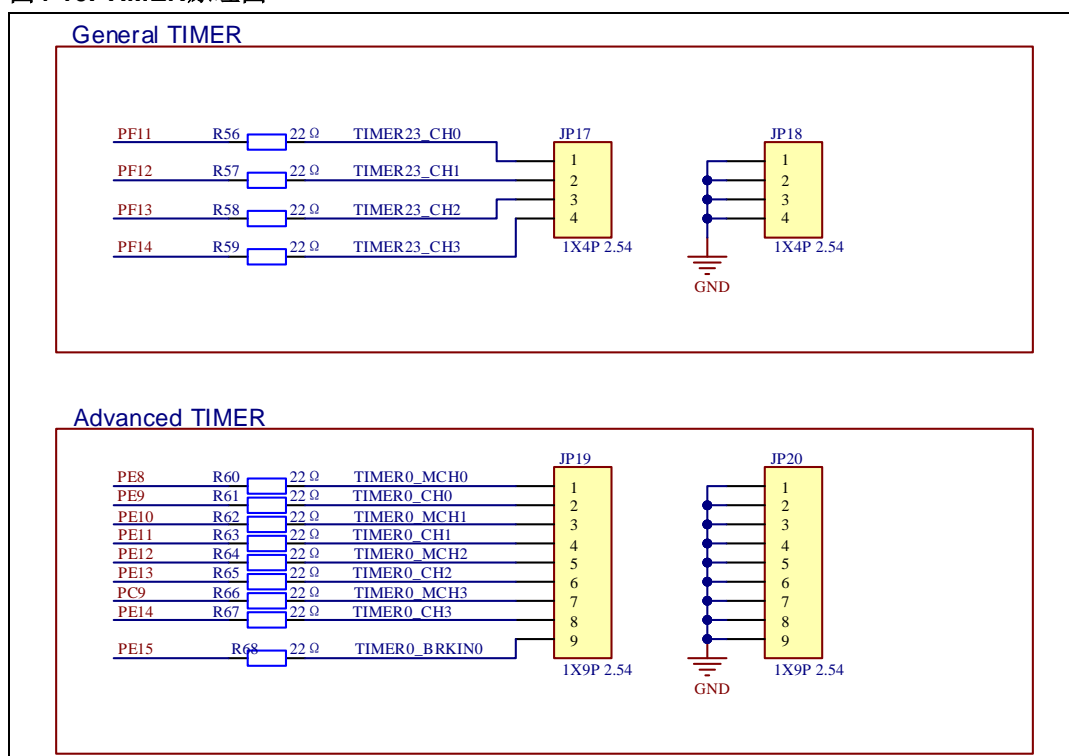
4.9. I2C

图4-9. I2C原理图



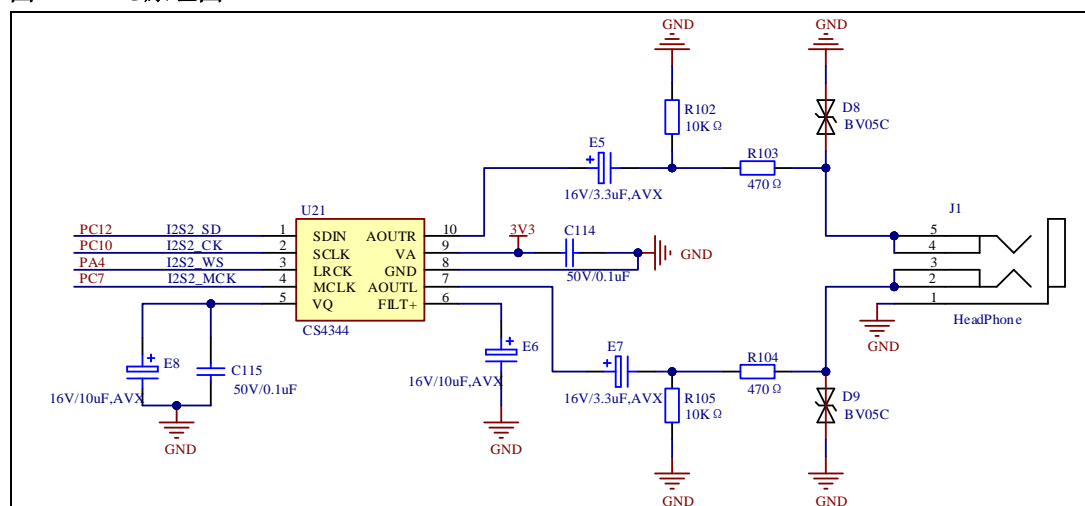
4.10. TIMER

图4-10. TIMER原理图



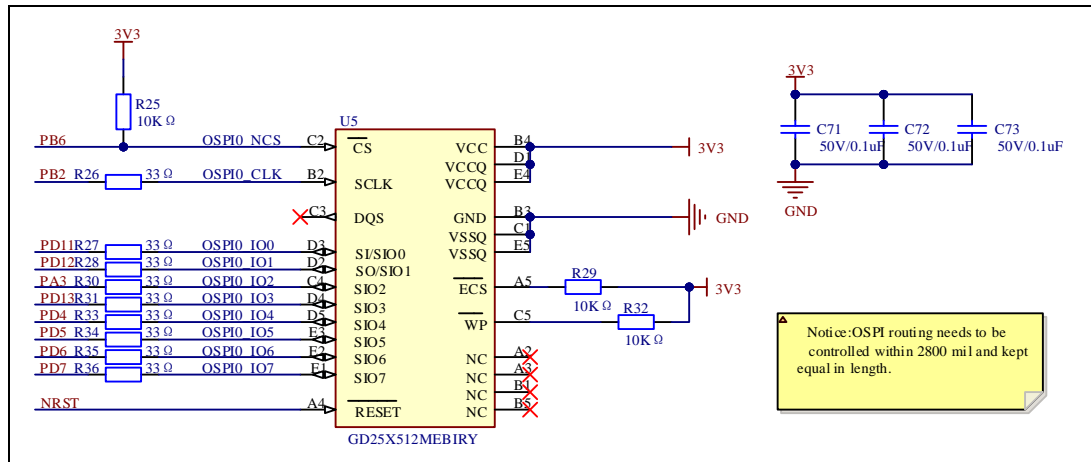
4.11. I2S

图4-11. I2S原理图



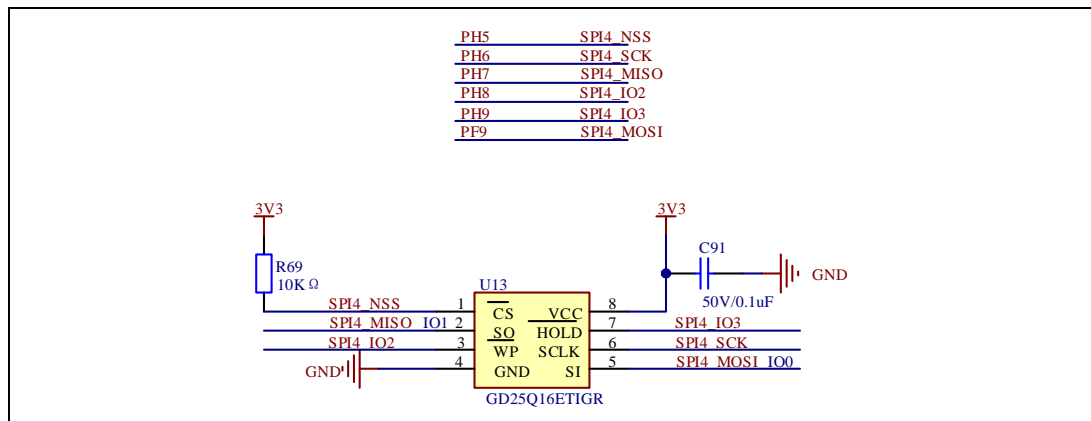
4.12. OSPI

图4-12. OSPI原理图



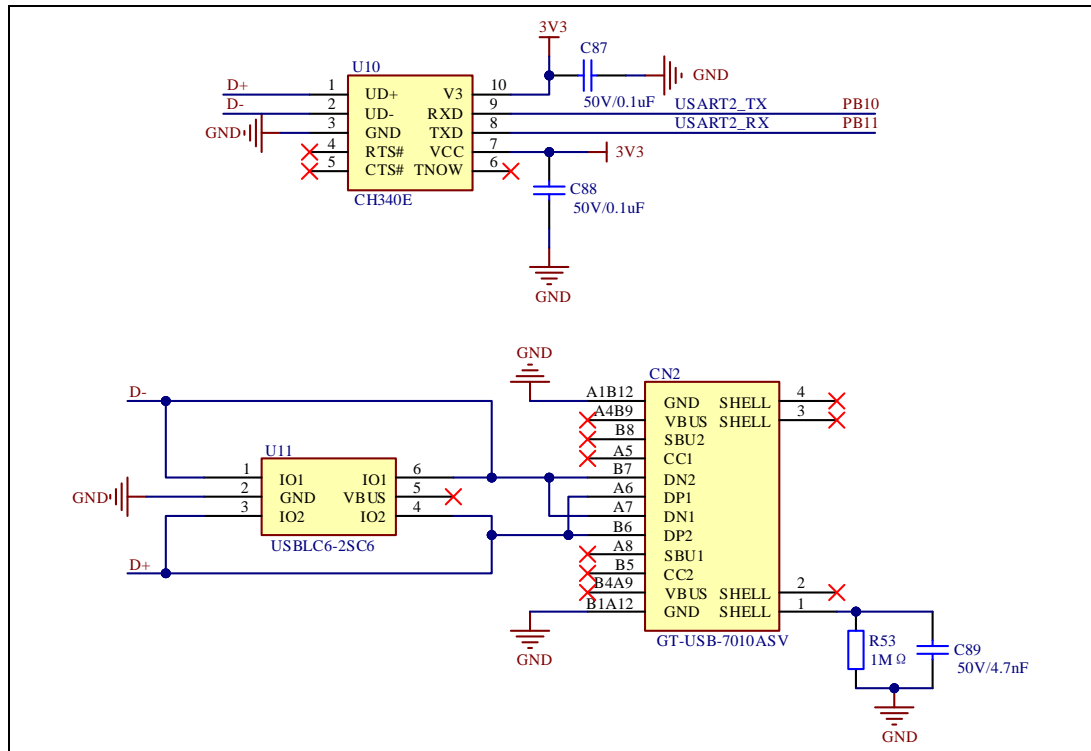
4.13. SPI

图4-13. SPI原理图



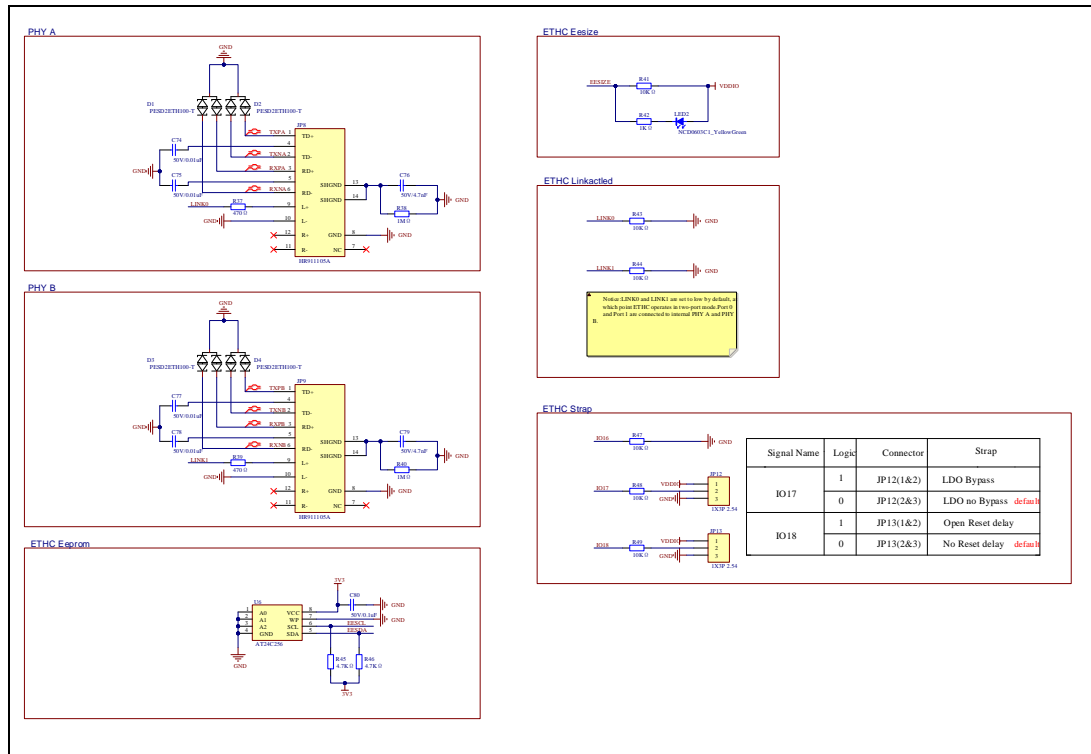
4.14. USART

图4-14.USART原理图



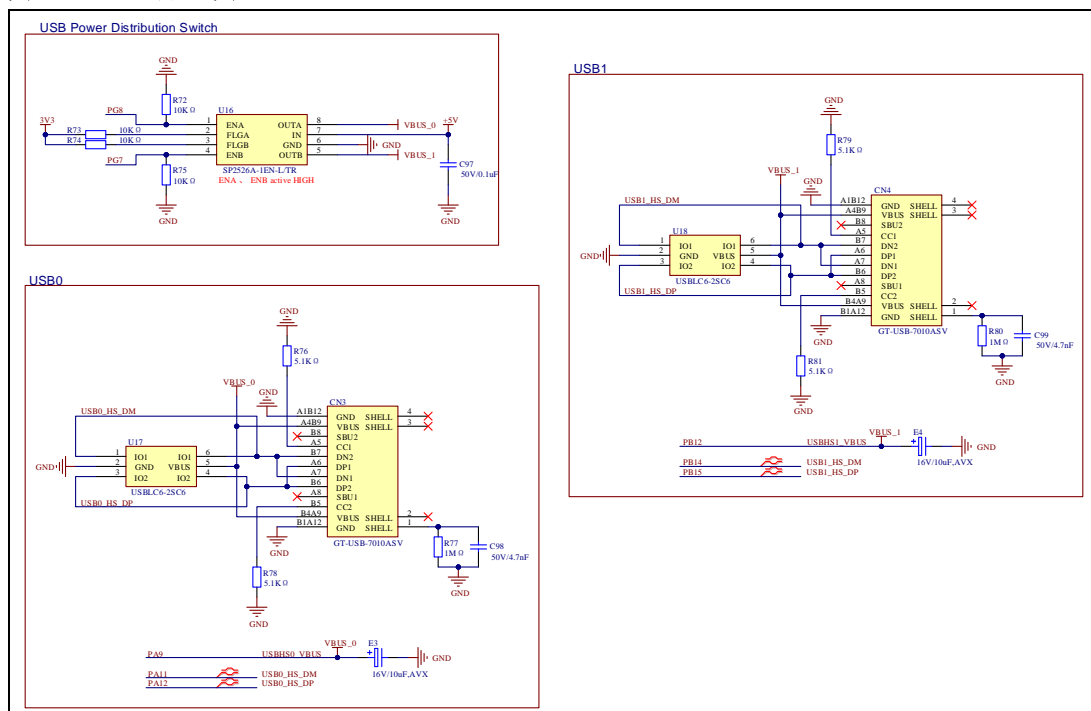
4.15. ETHC

图4-15. ETHC原理图



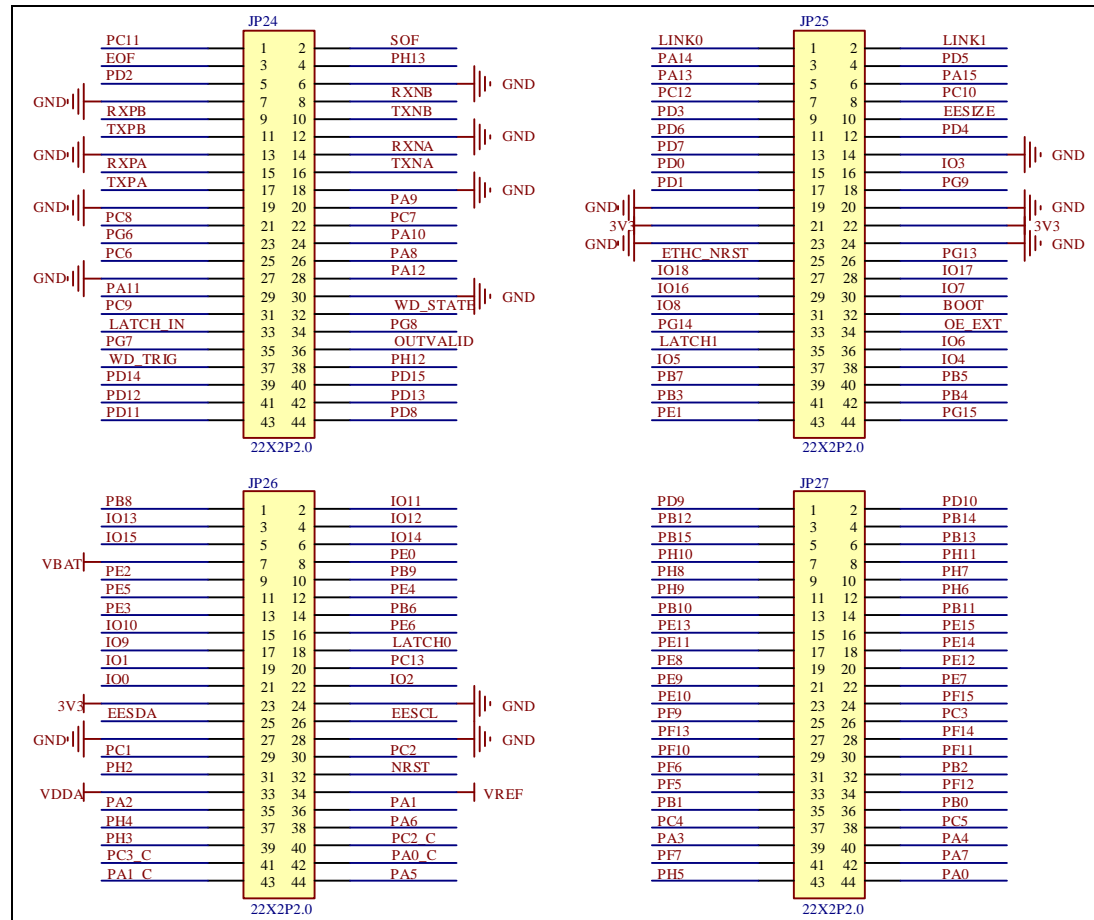
4.16. USB

图4-16. USB原理图



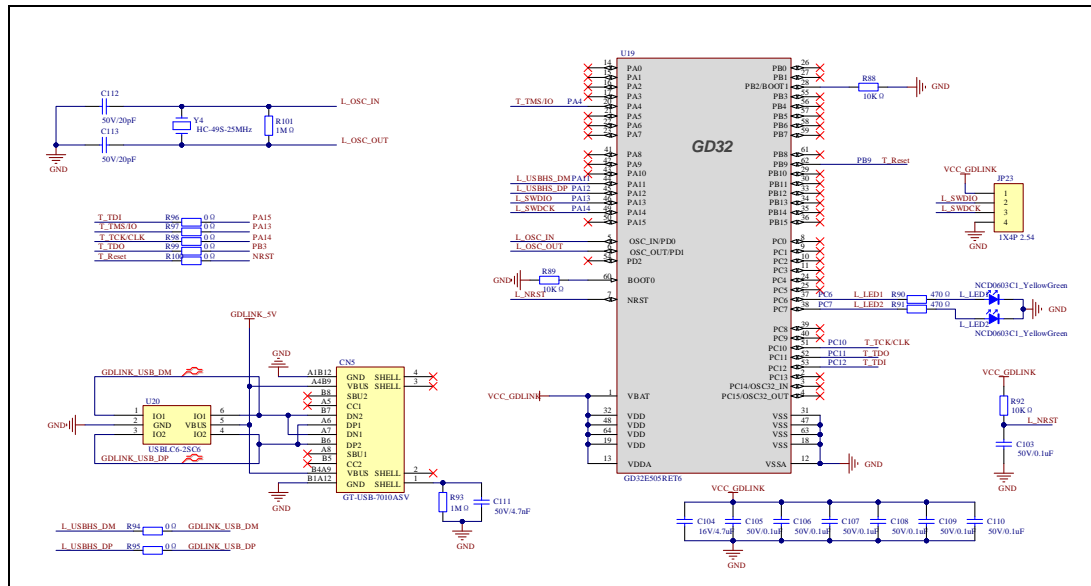
4.17. Extension

图4-17. Extension原理图



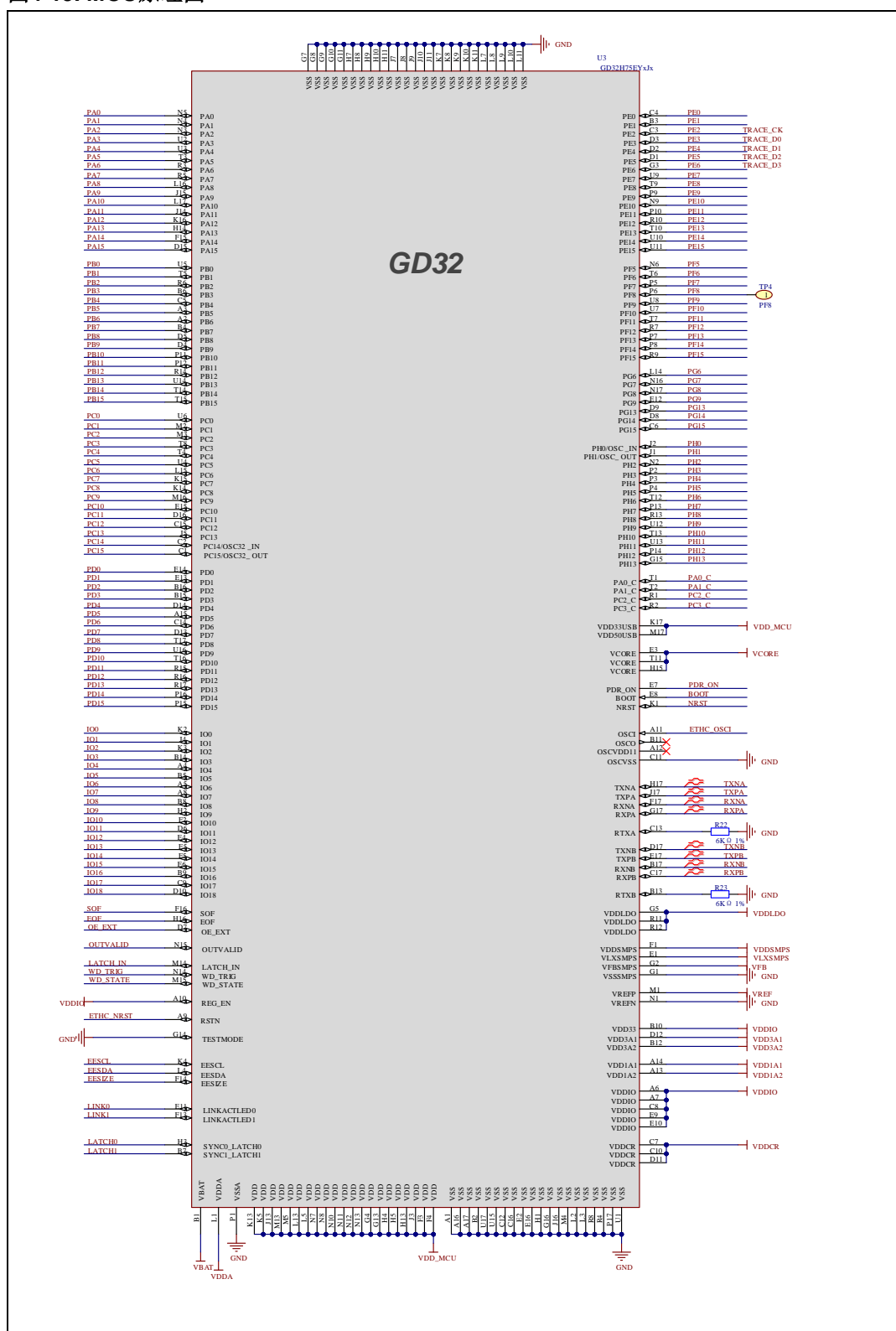
4.18. GD-Link

图4-18. GD-Link原理图



MCU

图4-19. MCU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。

GD32H75EY-EVAL-V1.0 开发板上有 3 个 LED。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_Running_LED>到开发板上，LED3, LED4, LED5 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。

GD32H75EY-EVAL-V1.0 开发板有四个按键和三个 LED。其中，四个按键是 Reset 按键，Tamper 按键，Wakeup 按键，User 按键；LED3, LED4 和 LED5 可通过 GPIO 控制。

这个例程讲述如何使用 Tamper 按键控制 LED3。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 LED3 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，LED3 将会点亮，再次按下 Tamper 按键，LED3 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

GD32H75EY-EVAL 开发板有四个按键和三个 LED。其中，四个按键是 NRST 按键，Tamper 按键，Wakeup 按键，User 按键；LED3，LED4 和 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 LED3。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 LED3 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，LED3 闪烁一次，按下 Tamper 按键，LED3 将会点亮，再次按下 Tamper 按键，LED3 将会熄灭。

5.4. 串口打印

5.4.1. DEMO 目的

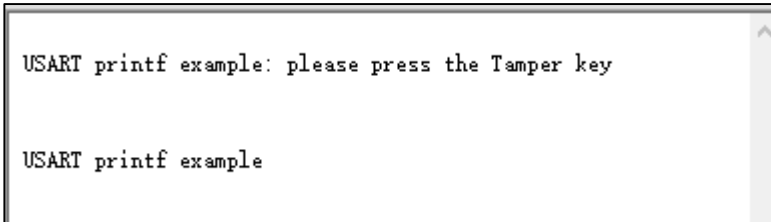
这个例程包括了 GD32 MCU 的以下功能：

- 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，LED3 和 LED4 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”，同时 LED3 被点亮，否则 LED3 熄灭。

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，LED3 和 LED4 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，LED3，LED4 轮流闪烁；如果结果不相同，LED3，LED4 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_内部参考电压

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 30 degrees Celsius
the reference voltage data is 1.200V

the temperature data is 30 degrees Celsius
the reference voltage data is 1.200V

the temperature data is 30 degrees Celsius
the reference voltage data is 1.200V

the temperature data is 30 degrees Celsius
the reference voltage data is 1.200V

the temperature data is 30 degrees Celsius
the reference voltage data is 1.200V
```

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 和 PA0_C 引脚接外部电压。

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 常规并行模式

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 工作在常规并行模式

5.9.2. DEMO 执行结果

下载<09_ADC0_ADC1_Routine_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 和 PA0_C 引脚接外部电压。

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，并行转换常规序列。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 DAC0_OUT1 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。

LED3 和 LED4 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 PA5 引脚输出。

PA5 输出的电压可以通过示波器观测。

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

GD32H75EY-EVAL-V1.0 开发板集成了 HPDF 和 I2S 模块, JP21 / JP22 跳线帽跳到 HPDF, 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_I2S_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. SPI FLASH DMA

5.13.1. 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 主模式通过 DMA 方式读写带有 SPI 接口的 NOR Flash。

GD32H75EY-EVAL 开发板上集成的 SPI 模块可以和外部的 SPI NOR Flash 存储器设备进行通信。GD25Q16BS 是一款 16Mb 容量的 SPI NOR Flash 存储器，存储容量：16Mb（2MB），适用于各种嵌入式系统存储需求。

5.13.2. DEMO 执行结果

把电脑串口线连接到开发板的 COM 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 <13_SPI_SPI_Flash_DMA> 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，串口打印出“SPI-GD25Q16 Test Passed!”，否则，串口打印出“Err: Data Read and Write aren't Matching.”。最后，LED3 灯闪烁。

下图为实验结果：

```

#####
gd32h75e System is starting up...
gd32h75e Flash:3840Klock:600000000Hz
gd32h75e The CPU Unique Device ID:[22E13043-51050733-33363239]
gd32h75e 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

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

```

5.14. OSPI 八线 Flash

5.14.1. 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

GD32H75EY-EVAL 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。

5.14.2. 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。

下载程序<14_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:
GD32H75EY_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:
GD32H75EY_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:

GD32H75EY_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:
GD32H75EY_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:
GD32H75EY_EVAL octal-flash memory mapped read test!

The data read
in memory mapped mode from flash is:
GD32H75EY_EVAL octal-flash memory mapped read test!

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

5.15. CAN 网络通信

5.15.1. DEMO 目的

这个例程包括了 GD32 MCU 的以下功能：

- 学习使用 CAN 实现三个节点之间的通信；
- 学习使用 USART 模块与上位机进行通讯。

5.15.2. DEMO 执行结果

下载程序<15_CAN_Network>到开发板上。将 JP14、JP15 和 JP16 的 CANL 引脚和 CANH 引脚分别相连。将串口线连到开发板的 USART。用户按下 TAMPER 或 WAKEUP 键，数据帧将通过 CAN1 或 CAN2 发送出去，并将数据内容通过串口打印出来。当 CAN0 接收到数据帧时，接收到的数据通过串口打印，同时 LED3 或 LED4 状态翻转一次。通过串口输出的信息如下图所示。

```
Communication test CAN0, CAN1 and CAN2, please press WAKEUP or TAMPER key to start!

CAN1 transmit data:
a0 a1 a2 a3 a4 a5 a6 a7
CAN0 Mailbox receive data:
a0 a1 a2 a3 a4 a5 a6 a7
CAN2 transmit data:
b0 b1 b2 b3 b4 b5 b6 b7
CAN0 FIFO receive data:
b0 b1 b2 b3 b4 b5 b6 b7
```

5.16. RCU 时钟输出

5.16.1. 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

5.16.2. DEMO 执行结果

下载程序<16_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
```

5.17. PMU 睡眠模式唤醒

5.17.1. 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

5.17.2. DEMO 执行结果

下载程序<17_PMU_Sleep_Wakeup>到开发板上，并将串口线连到开发板的 USART 上。板上上电后，LED4 和 LED5 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。LED4 和 LED5 灯同时闪烁。

5.18. RTC 日历

5.18.1. 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

5.18.2. DEMO 执行结果

下载程序<18_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。

5.19. 高级 TIMER PWM 输出

5.19.1. DEMO 目的

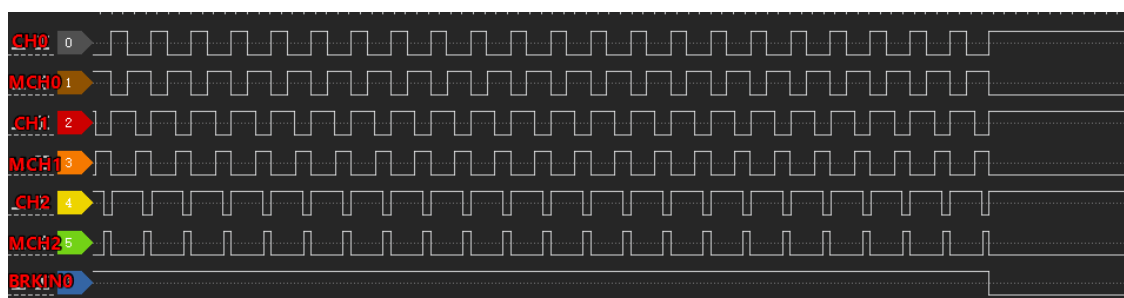
这个例程包括了 GD32 MCU 的以下功能：

- 学习使用高级定时器输出互补 PWM 波
- 学习高级定时器中止功能

5.19.2. DEMO 执行结果

下载程序<19_Advanced_Timer_PWM_Output>到 GD32H75EY-EVAL-V1.0 开发板，并运行程序。当程序运行时，可以看到 CH0/MCH0，CH1/MCH1 和 MCH2/MCH2 分别输出不同占空比的互补 PWM 波形。如果给 BRKIN0 引脚（PE15）输入低电平，高级定时器的 PWM 输出就会被中止。

互补 PWM 输出如下所示:



5.20. TIMER 呼吸灯

5.20.1. 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

5.20.2. DEMO 执行结果

下载程序<20_TIMER_Breath_LED>到 GD32H75EY-EVAL-V1.0 开发板, 并运行程序。当程序运行时, 可以看到 LED3 由暗变亮, 由亮变暗, 往复循环, 就像人的呼吸一样有节奏。

5.21. USB 设备

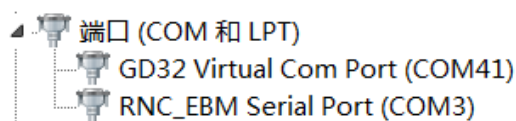
5.21.1. 虚拟串口

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

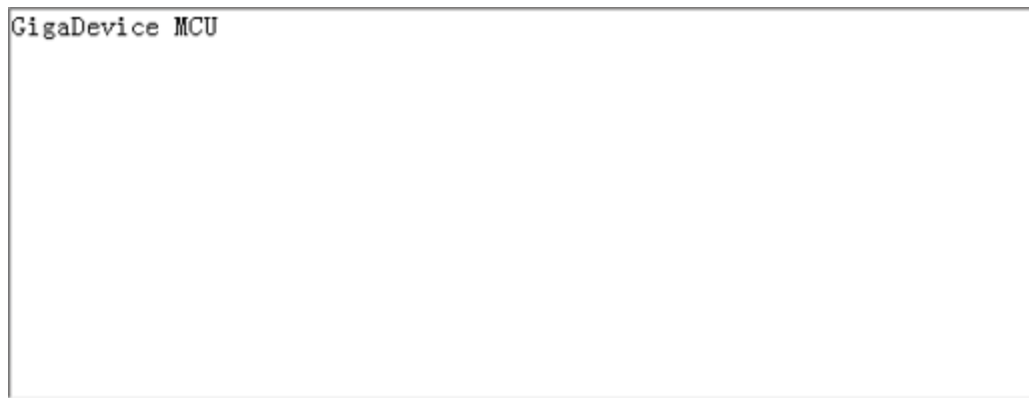
GD32H75EY-EVAL-V1.0 开发板具有两个 USBFS/HS 接口。在本例程中, GD32H75EY-EVAL-V1.0 开发板的 USBHS0 被 USB 主机枚举为一个 USB 虚拟串口, 如下图所示, 可在 PC 端设备管理器中看到该虚拟串口。该例程使得 USB 接口看起来像是个串口, 也可以通过 USB 口回传数据。通过键盘输入某些信息, 虚拟串口可以接收并显示这些信息。



DEMO 执行结果

将< 21_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。



5.21.2. 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

GD32H75EY-EVAL-V1.0 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，GD32H75EY-EVAL-V1.0 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 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

JP42 跳到 KEY，将 < 21_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。

5.22. USB 主机

5.22.1. 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

GD32H75EY-EVAL-V1.0 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 < 22_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 type characters:

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

5.22.2. 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

GD32H75EY-EVAL-V1.0 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 跳到 USART，JP70 跳到 USB。将 OTG 电缆线插入到 USB 接口，然后将 <22_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.
```

5.23. EtherCAT

5.23.1. DEMO 目的

这个例程包括了 GD32 MCU 的以下功能：

- 学习使用 Ethercat 模块作为从站
- 学习主站和 Ethercat 从站之间的操作

GD32H75EY-EVAL-V1.0 开发板内部包含 EtherCAT 模块，并且该模块可以处理 EtherCAT 协议。该示例主要展示了如何使用 GD32H75E 中 EtherCAT 模块作为一个从站设备，运行 CIA402 程序和外部主站设备进行通信。

5.23.2. DEMO 执行结果

下载程序<23_EtherCAT >到 GD32H75EY-EVAL-V1.0 开发板，并运行程序。当程序运行时，连接网线到开发板上，使用主站软件与当前从站板进行通信，当系统正常运行后板子上 LED2 灯保持常亮。

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	2025 年 06 月 20 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 according to the laws of the People's Republic of China and other applicable laws. The Company reserves all rights under such laws and no Intellectual Property Rights are transferred (either wholly or partially) or licensed by the Company (either expressly or impliedly) herein. The names and brands of third party referred thereto (if any) are the property of their respective owner and referred to for identification purposes only.

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

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