现代电子系统设计

Psoc 提高实验

学　　号　　**2017011589**

姓　　名　　**吾尔开希**

专　　业　　**自动化**

日　　期　　**2019.7.1---7.12**

目录

一、Psoc 提高实验

1. 实验内容

1.1 必做内容

基于 freeRTO 完成双核通信实验，具体为：用一个核通过 Capsense 滑条控制 RGBLed 的亮度并显示在 E-INK 上，同时将滑条位置信息通过 BLE 模块发送给计算机或者手机，另一个核通过 IPC 通道获取 RGBLed 亮度信息并通过 UART 发送给计算机。

1.2 自选内容

增加另一种控制模式，用手机通过 BLE 控制 RGBLed 的颜色和亮度，两种模式利用 SW 开关切换。

2. 设计方案

2.1 Capsense 滑条

滑条位置从左到右对应 0 到 100, 滑条下的按钮分别对应 0 和 100。Capsense 有独立的 task 来控制，该 task 通过三条队列向 EINK 显示 task，LED 亮度控制 task（pwmTask）与 BLE 的 task 发送 0 到 100 的信息。

2.2 LED 亮度控制

在基础功能模式下，LED 亮度控制的 task 接收 Capsen 滑条 task 发来的队列信息控制红灯亮度；在自选功能模式下，LED 亮度控制的 task 接收 BLE 的 task 发来的队列信息控制 LED 灯的颜色与亮度。

2.3 BLE 蓝牙控制

在基础功能模式下，BLE 蓝牙的 task 接收 Capsen 滑条 task 发来的队列信息并将其通过 notification 的方式发送给手机/电脑端；在自选功能模式下，手机端通过 BLE 蓝牙向 Psoc 写入 rgb 颜色与亮度，BLE 蓝牙 task 将其通过队列发送给 LED 控制的 task。

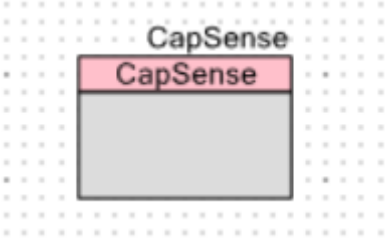
2.4 EINK 显示

EINK 显示的 task 接收 Capsense 滑条控制 task 通过队列发送来的信息并将其以一定的格式显示在 EINK 墨水屏上。

3. 电路图

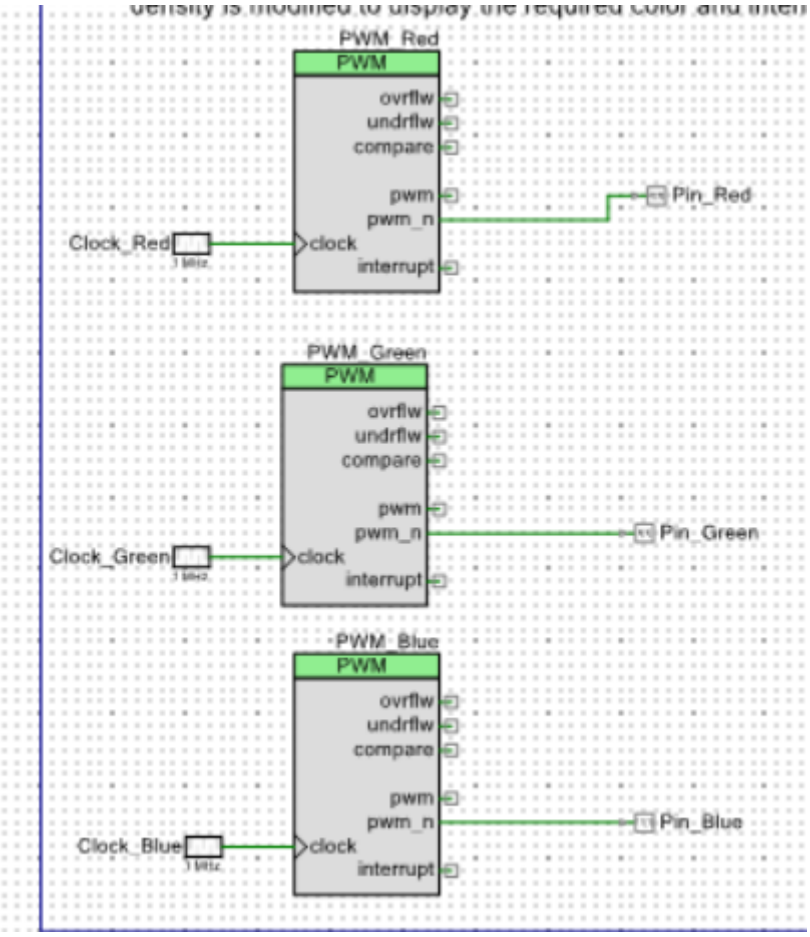
3.1 Capsense 滑条

Capsense 滑条与按钮有封装好的模块，拖入设计图并正确连接引脚即可。



3.2 LED 亮度控制

由于自选功能需要控制 LED 灯的颜色，所以通过三个 PWM 模块连接 RGB 引脚，PWM 的占空比在程序中软件写入。

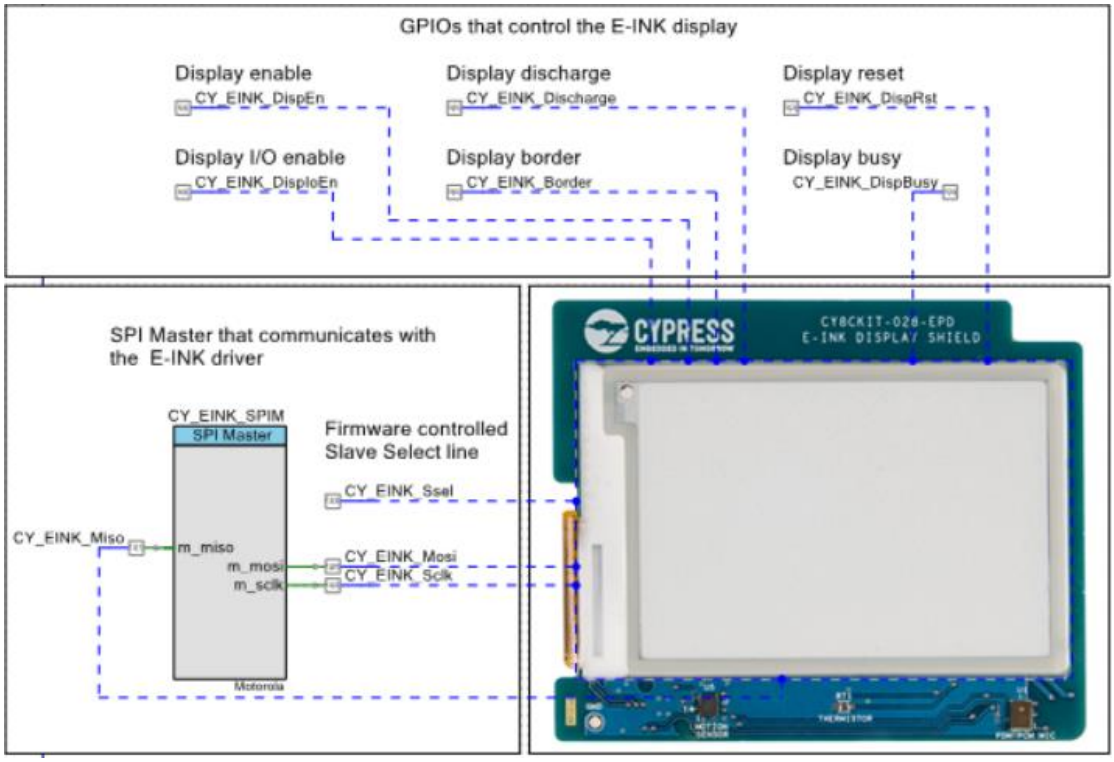


3.3 BLE 蓝牙控制

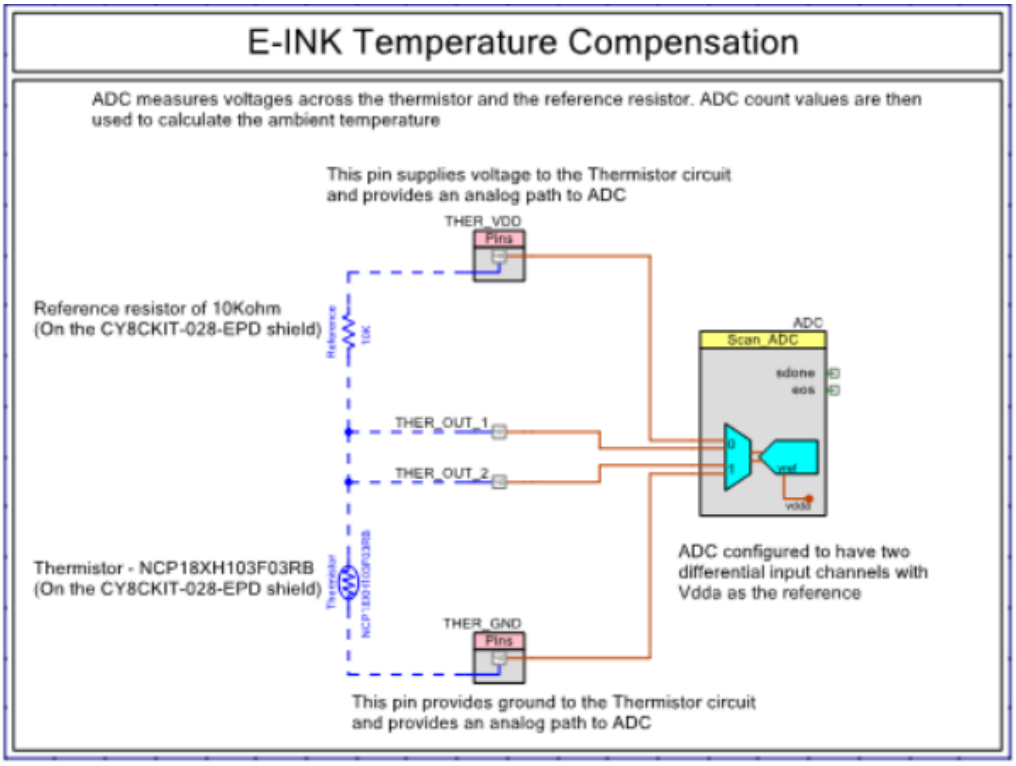
BLE 同样有封装好的模块，拖入设计图中即可。

3.4 EINK 显示

EINK 显示的硬件驱动一方面需要一系列的 GPIO 控制，另一方面需要 SPI_MASTER 模块来给它传输信息。



此外，EINK 正常显示还需要环境温度信息。



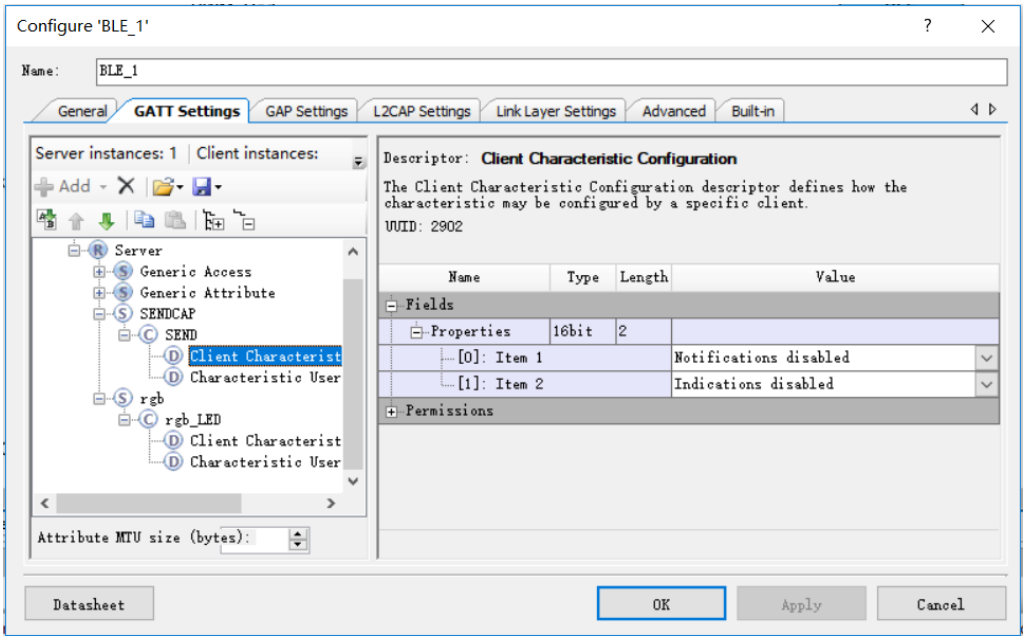
4. 模块选择与参数设置

4.1 PWM 模块

PWM 模块：时钟频率 1MHz，周期 32768，占空比由程序软件写入。

4.2 BLE 蓝牙模块

设置有两个 service，分别用于接收滑条信息与发送 rgb 值。滑条信息 service 允许 notification，rgb 值 service 允许 read，write 与 notification。

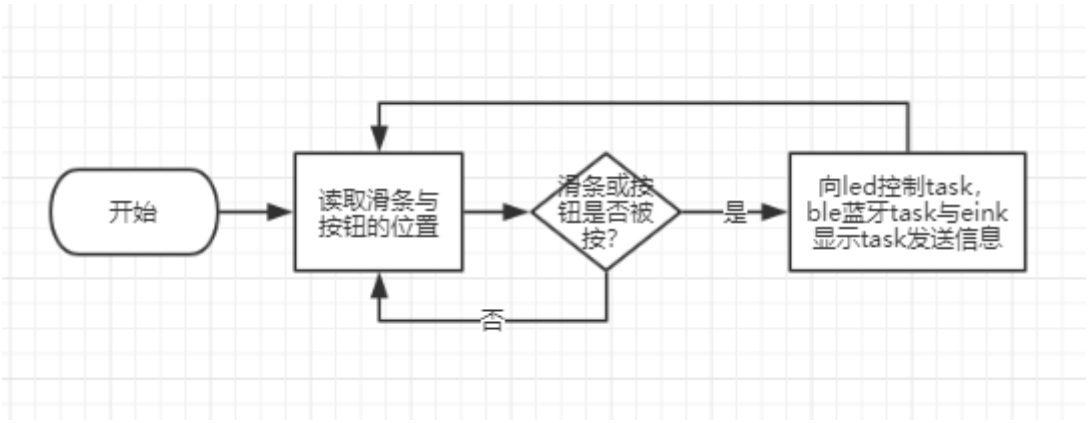


4.3 其他模块

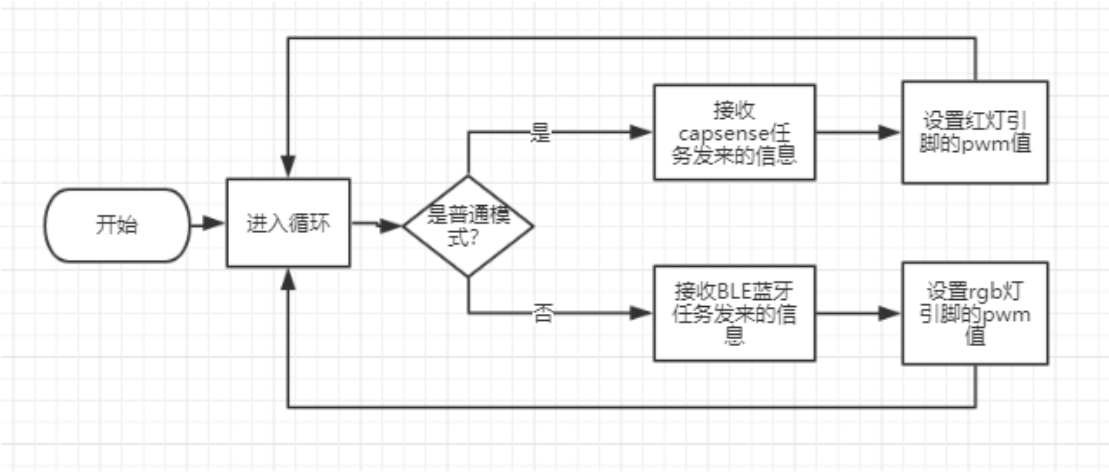
此外还有 Capsense 模块，uart 模块，eink 模块与温度检测模块，没有进行特殊设置。

5. 流程图

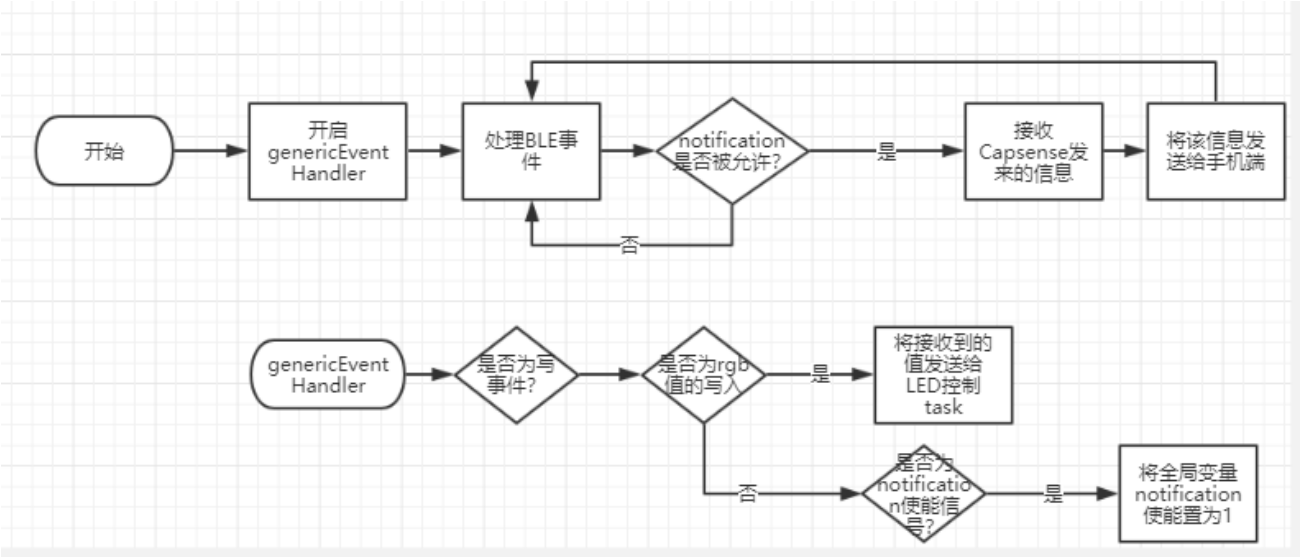
5.1 Capsense 滑条 task



5.2 LED 亮度控制 task



5.3 BLE 蓝牙控制



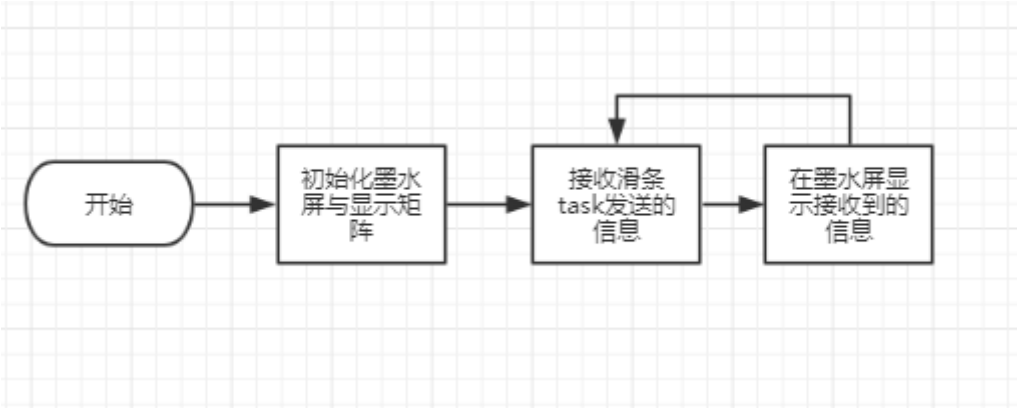
rgb 值包含四个信息：红绿蓝三种颜色值和亮度信息，分别为一字节数值，因此将其表示为 32 位四字节来处理。

```
if ((writeRequest->handleValPair.attrHandle == CY_BLE_RGB_RGB_LED_CHAR_HANDLE) && (!mode_read))
{

    /* Extract the write value sent by the Client for RGB LED Color
       characteristic */
    memcpy(rgbData.valueArray,writeRequest->handleValPair.value.val,
           RGB_DATA_LEN);

    printf("rgb 0:%d\r\n", rgbData.valueArray[0]);//red
    printf("rgb 1:%d\r\n", rgbData.valueArray[1]);//green
    printf("rgb 2:%d\r\n", rgbData.valueArray[2]);//blue
    printf("rgb 3:%d\r\n", rgbData.valueArray[3]);
    /* Update the RGB LED color and intensity */
    xQueueOverwrite(rgbLedDataQ, &rgbData.colorAndIntensity);
}
```

5.4 EINK 显示



由于墨水屏显示函数还需要当前屏幕上的信息，所以创建两个数组。

```
//one for currentFrame, one for nextFrame
cy_eink_frame_t textPageBackgroundGroup[2][CY_EINK_IMAGE_SIZE];
```

一个存储当前屏幕上的信息，另一个存储欲显示的信息。

```
static uint8_t curframe = 0;

cy_eink_frame_t* frame;
if(curframe==0)
{
    frame = textPageBackgroundGroup[1];
}
else
{
    frame = textPageBackgroundGroup[0];
}

uint8_t textOrigin[2] =  {0x00u, 0x00u};
/* Load the frame buffer with the current text Page content */
Cy_EINK_TextToFrameBuffer(frame, text,
                          CY_EINK_FONT_16X16BLACK, (uint8_t*) textOrigin);

/* Perform a full update to avoid ghosting as the text pages differ
   significantly from one another and also from the main menu images */
Cy_EINK_ShowFrame(textPageBackgroundGroup[curframe], frame,
                  CY_EINK_FULL_2STAGE, CY_EINK_POWER_AUTO);
```

6. 实验结果

达到了必做内容与自选内容的要求，CM4 核通过 Capsense 滑条控制 RGBLed 的亮度并显示在 E-INK 上，同时将滑条位置信息通过 BLE 模块发送给计算机或者手机，另一个核通过 IPC 通道获取 RGBLed 亮度信息并通过 UART 发送给计算机。

然而由于 EINK 刷新需要一定时间，所以其显示有大概 0.5s 的延迟。

7. 实验中遇到的问题与解决方法

7.1 队列收发问题

滑条的位置信息通过 xQueue 队列发送给 EINK 显示的 task。由于 EINK 显示的刷新需要一段时间，所以当滑条位置变化较快时，EINK 的显示值可能会滞后，比如当滑条从 0 滑到 100 时，EINK 才显示到 60。

所以 capsenseTask 中，队列发送信息应该用 xQueueOverwrite 函数而不是 xQueueSend 函数，这样可以实时更新数值并且没有延时等待。

7.2 任务优先级问题

这次实验我用 freeRTOS 安排了四个任务，当添加第四个任务时，程序会卡死。

我猜想可能是任务优先级的问题，我隐约记得之前使用单片机编程时，第 x 优先级只能有 x 个任务，而我把四个任务都设置为第三优先级，可能就会出问题，于是我把蓝牙的优先级改为四，问题迎刃而解。

7.3 蓝牙显示滑条位置问题

蓝牙 bleTask 的 for 循环中，需要用 xQueueReceive 函数接收 capsense 的 task 发送的滑条数值，该函数的参数中有一个延迟时间参数，需要将其设为最大（portMAX_DELAY），否则手机端接收到的数据可能时断时续。该函数：

xQueueReceive(bleQueueHandle, &cap_value, portMAX_DELAY);

7.4 PSOC 读取 pin 值问题

由于我自选实验的模式切换要求，需要读取 GPIO 引脚值，我发现读取并不像我想象的那么简单，有些引脚无法输入或输出。我发现的另外一个问题是：PSOC 读取 PIN 值，对于输出引脚，需要用 Cy_GPIO_ReadOut 函数而不是 Cy_GPIO_Read 函数。

8. 体会、收获与建议

这次实验在软件方面，实践了并行编程，按我的理解，freeRTOS 应该是并行调度的一种实现，四个任务并行完成，效率较高，但也对任务之间的配合提出更

高的要求。传统的单片机中，并行任务只需要通过不同的时间中断来实现，并且不同任务直接的交流也可以通过简单的全局变量定义来实现，而 freeRTOS 系统直接在 Psoc 上实现 cpu 才有的并行功能，我觉得很不同寻常。

此外，还有通过 IPC 对双核系统的利用，虽然只是单字节变量的共享，但算是初次体验吧。

我认为这次最大的难点在于网络资料的不足，由于 Psoc6 版本较新，所以网上资源不太充足，只能依靠官网 datasheet 和文档。