



**《微机原理》实验报告**

姓名：曹建钬

学号：20375177

# 实验内容

**（一）PC 机与 Cortex-M7 处理器之间串行通信**

* 定义全局变量（如图1所示）和串口初始化函数（如图2所示）：

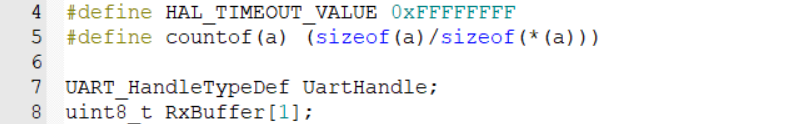
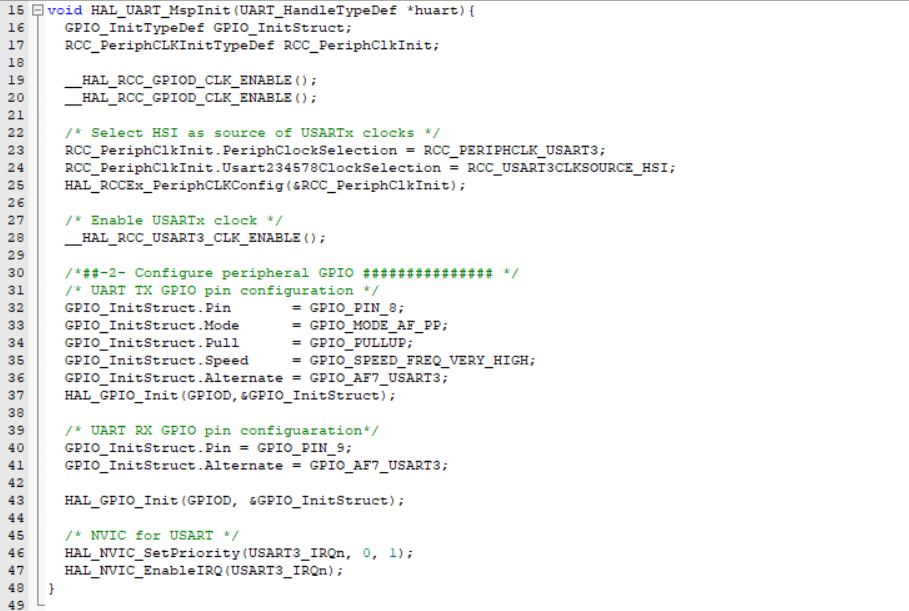


图 1：定义全局变量



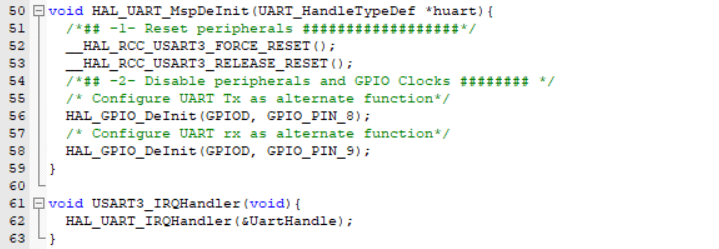


图 2：UART初始化函数

* 主函数程序编写：

初始化串口句柄，并调用串口初始化函数 HAL\_UART\_Init，对串口进行初始化，如图3所示：

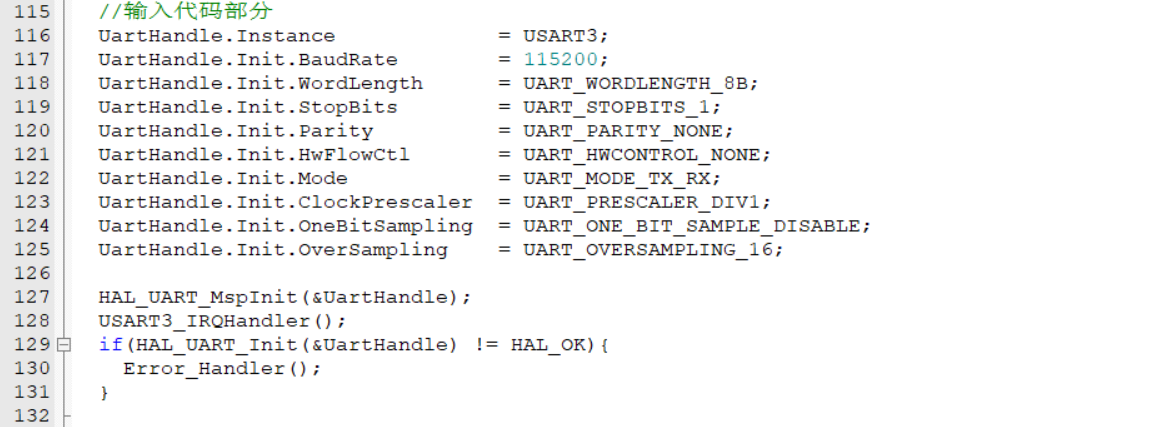
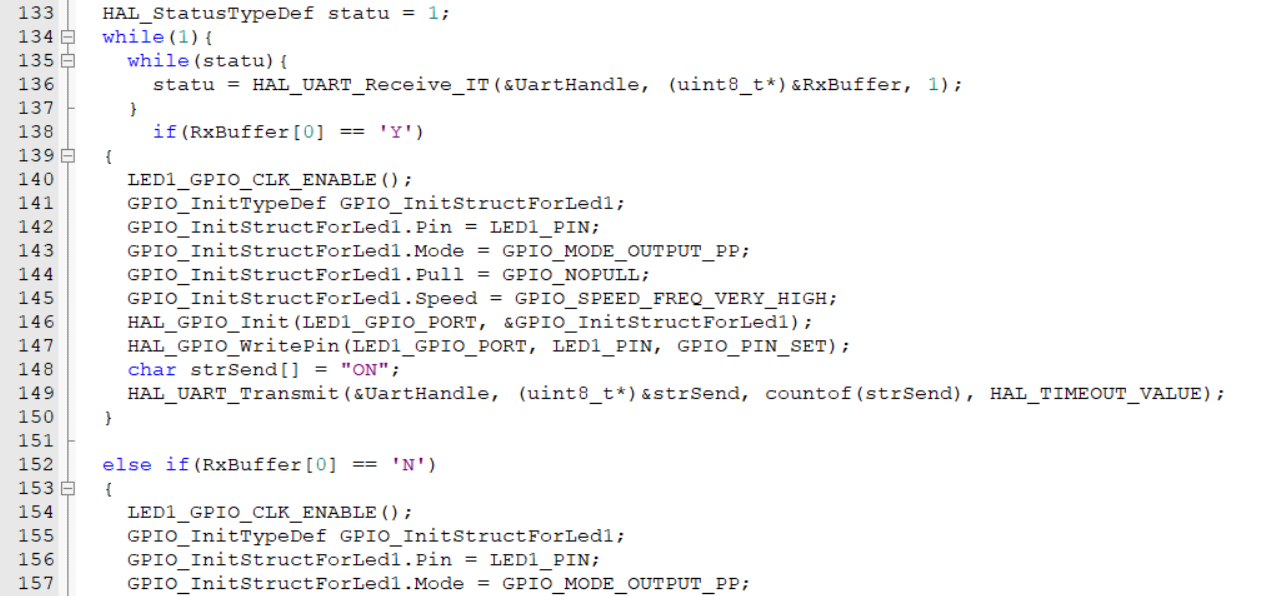


图 3：初始化串口句柄

编写程序（如图4），实现下述功能：通过 PC 机发送字符‘Y’，控制实验平台点亮 LED，并回传 PC 机“ON”；PC 机发送字符‘N’，实验平台熄灭 LED，并回传 PC 机“OFF”。且PC机发送其他字符时，实验平台返回发送的字符。



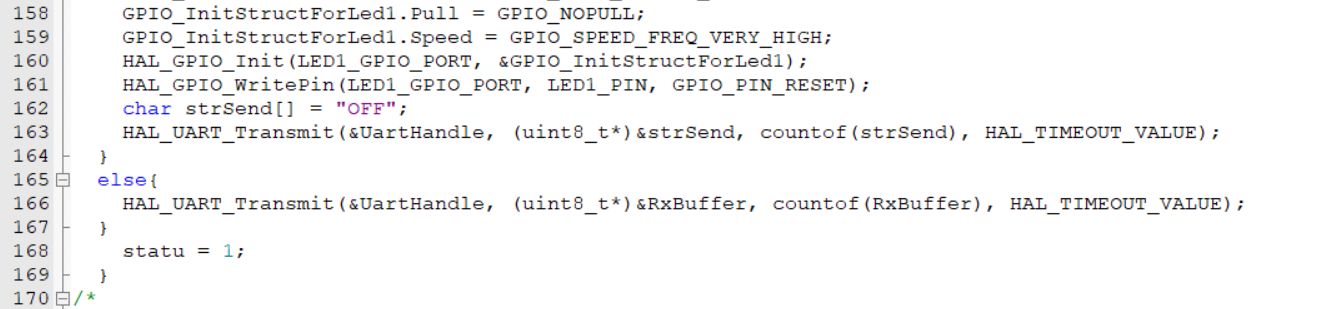


图 4：控制程序

思路为：串口要想持续工作，首先应该让串口收发数据函数内嵌于循环结构中；而要想让PC机发送字符后实验平台才执行对应任务，且实验平台发送完后应该处于等待输入而非持续输出的状态，这里我发现HAL\_UART\_Receive\_IT函数的返回值类型为HAL\_StatusTypeDef，如图5所示，而HAL\_StatusTypeDef为枚举类型，如图6所示，所以调用串口接收数据函数后会返回整数0，于是这里用变量statu作为接收数据的标志，当未接收数据时一直进行执行135-137行的循环体，持续接收数据，一旦接收到了数据，则执行138-167行的判断语句，实现单次任务的执行。

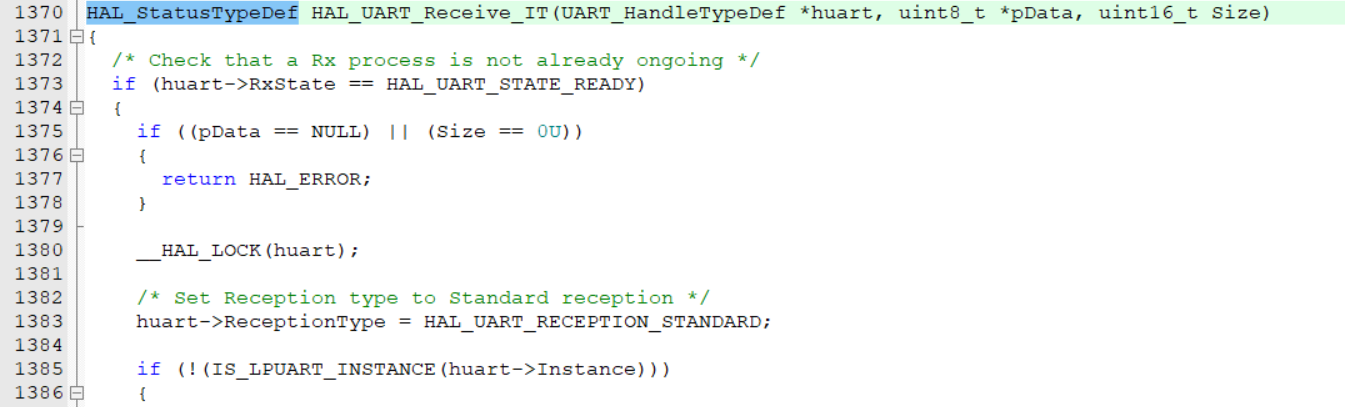


图 5：HAL\_UART\_Receive\_IT函数返回值类型

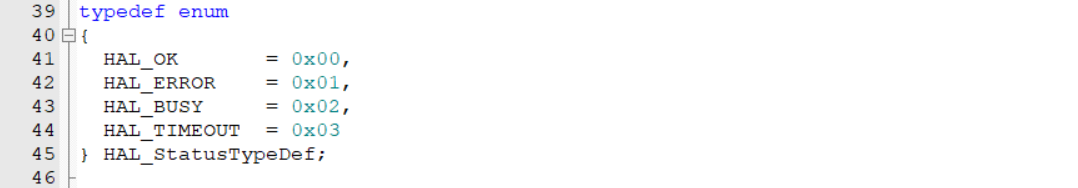


图 6：枚举类型

* 验证：

PC机通过端口与实验平台连接，如图7所示，连接成功，且使用COM7虚拟端口：

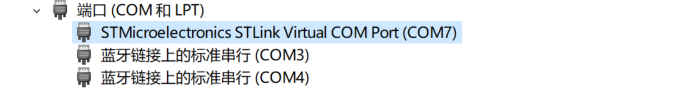


图 7：连接成功

打开串口调试工具sscom，并打开串口，如图8所示：

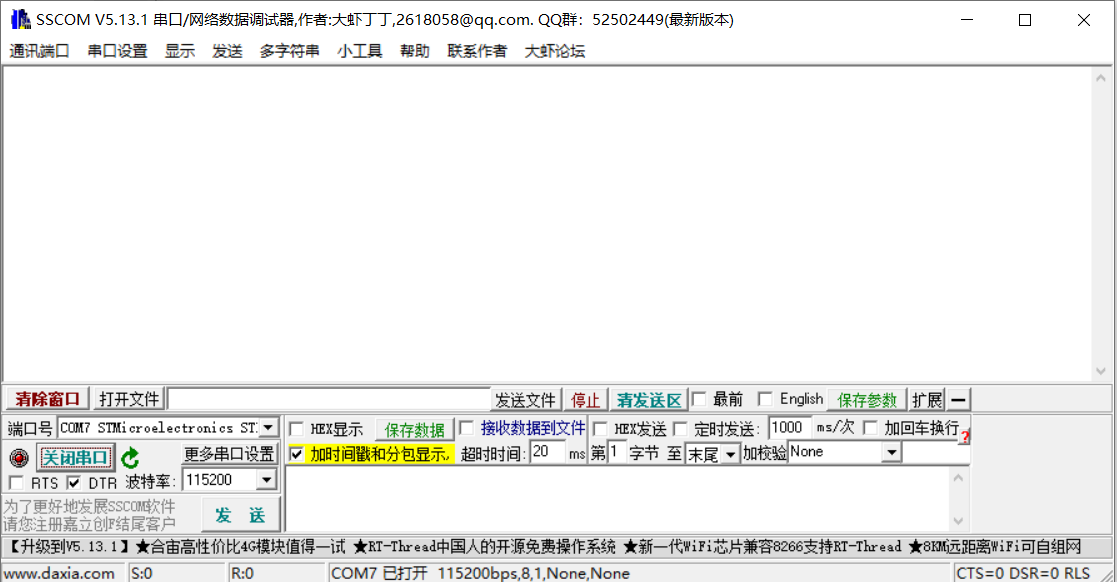


图 8：打开串口调试工具

发送字符‘Y’，实验平台LED1亮，如图9所示，返回“ON”，如图10所示：

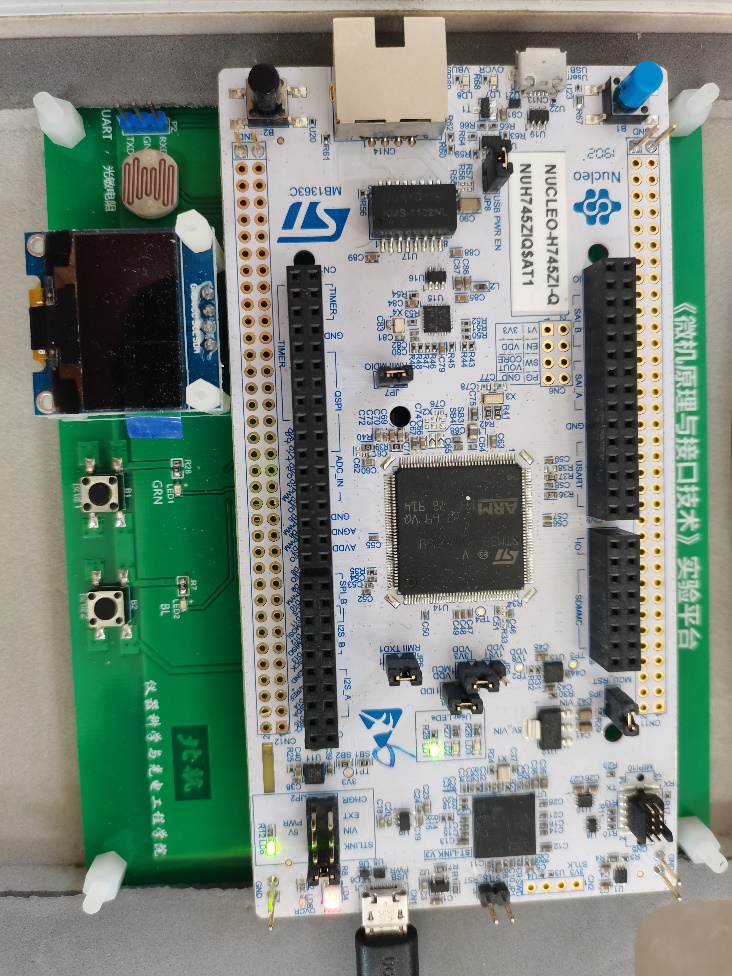


图 9：实验平台LED1亮

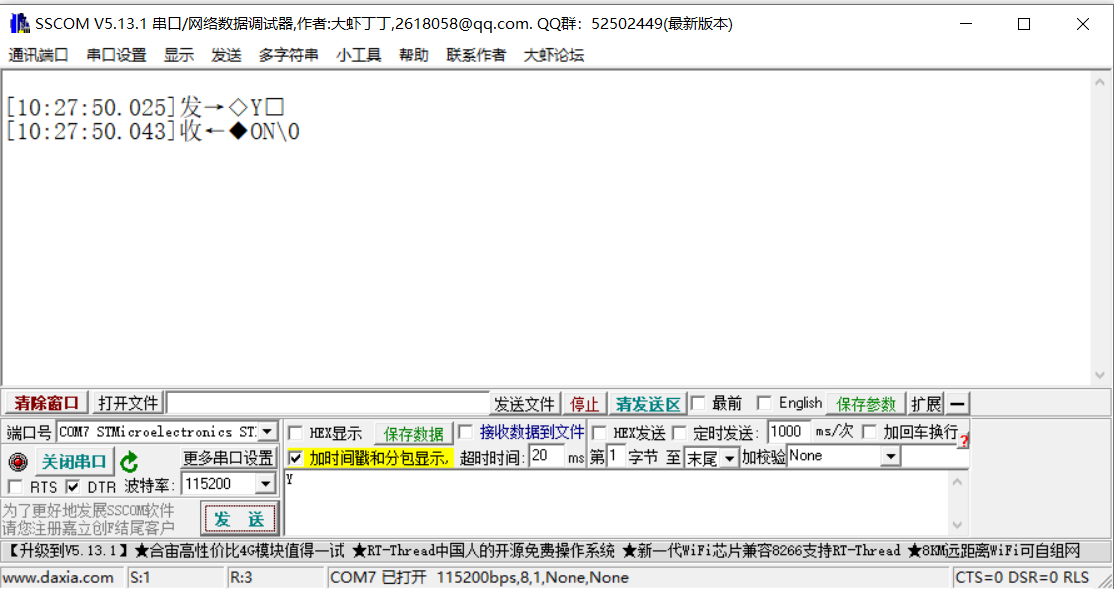


图 10：返回“ON”

发送字符‘N’，实验平台LED1熄灭，如图11所示，返回“OFF”，如图12所示：

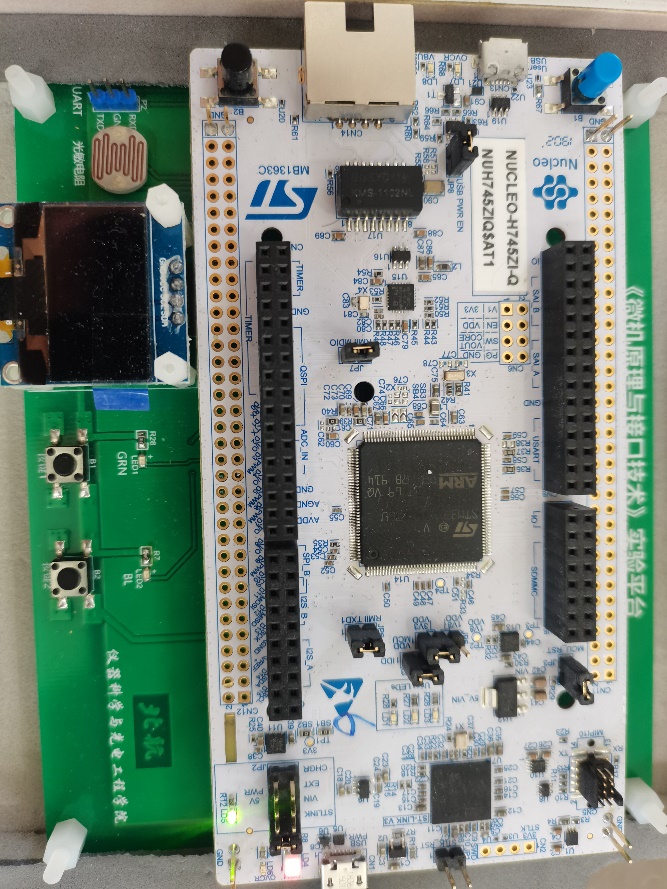


图 11：实验平台LED1熄灭

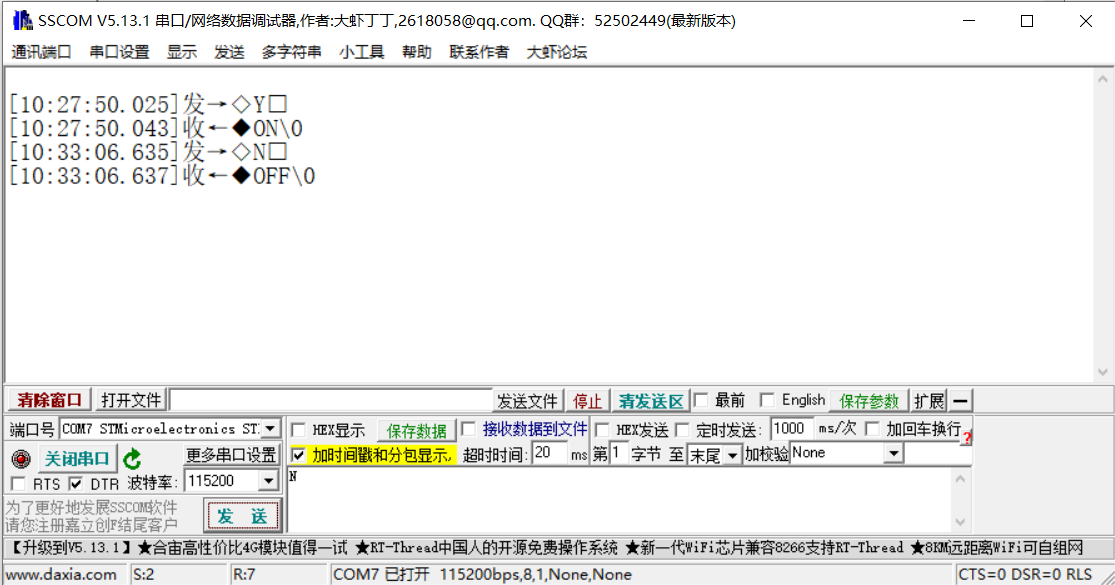


图 12：返回“OFF”

发送其他字符/字符串，实验平台无响应，返回发送的字符，如图13，14所示：

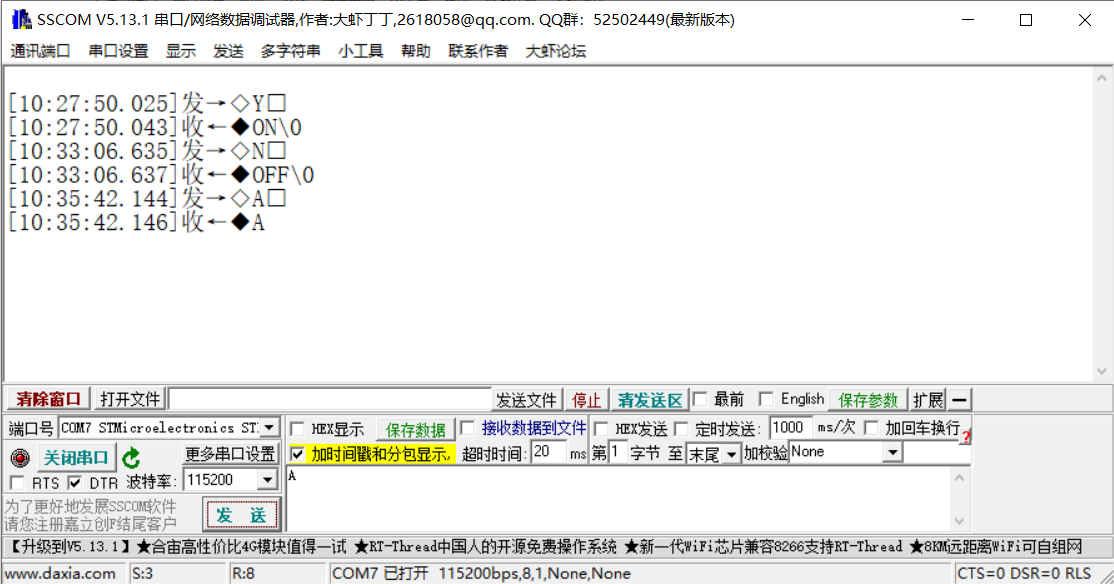


图 13：发送字符‘A’

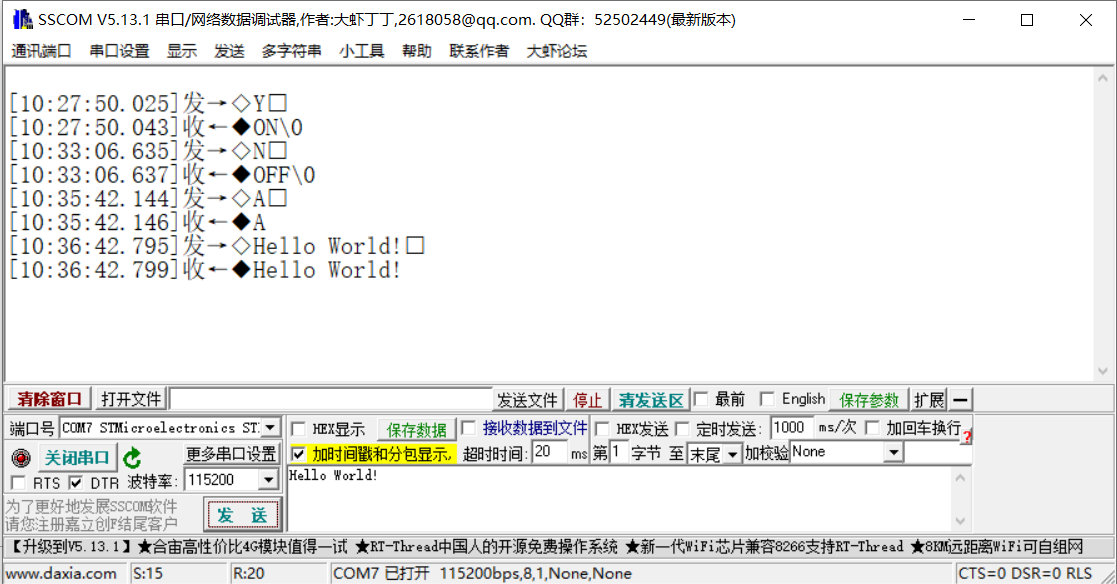


图 14：发送字符串“Hello World！”

综上，需求满足。

**（二）定时器：利用定时器完成 LED 灯的闪烁、利用定时器的 PWM 输出模式，实现 LED 灯渐变（呼吸灯）**

1. 完成 LED 灯闪烁，通过修改 arr 和 psc，达到快闪和慢闪的效果，其中快闪周期 1s，慢闪周期 5s：

* 创建定时器句柄和定时器初始化函数，如图15所示：

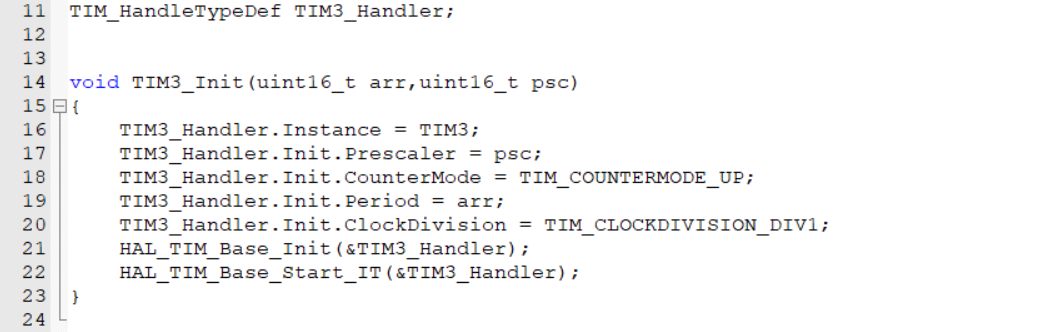


图 15：创建定时器句柄和初始化函数

* 初始化定时器中断相关函数，如图16所示：

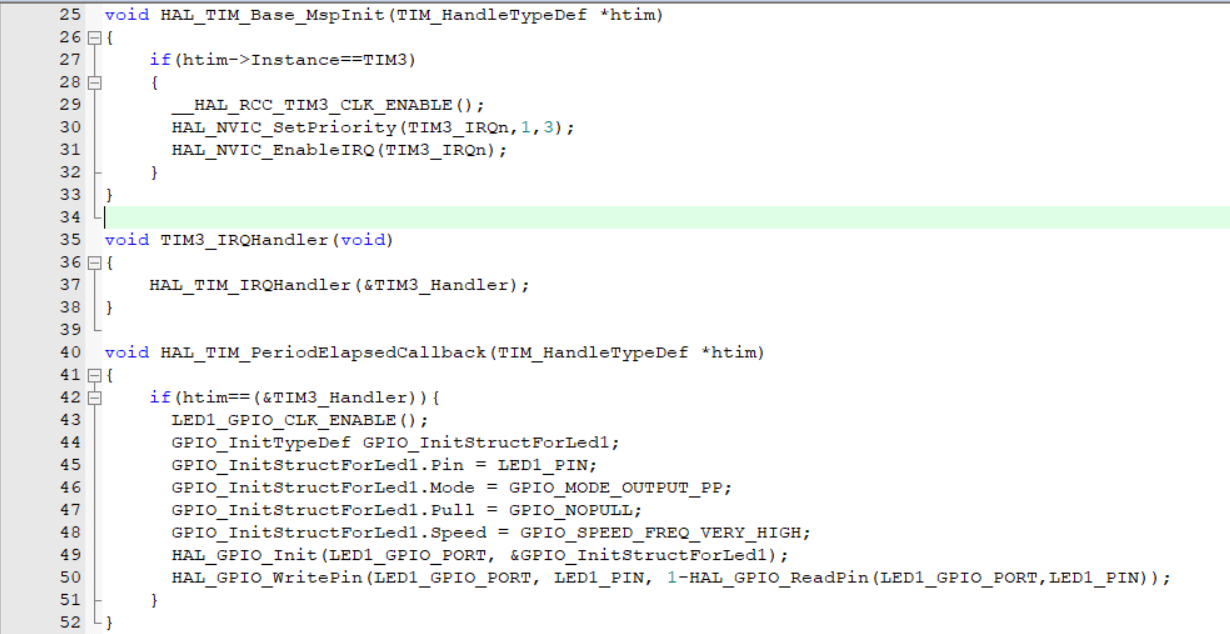


图 16：定时器中断相关函数

* 快闪主函数程序如图17所示：

快闪时，其中，可取。



图 17：快闪T=1s

快闪视频见附件。

* 慢闪主函数程序如图18所示：

慢闪时，其中，可取。

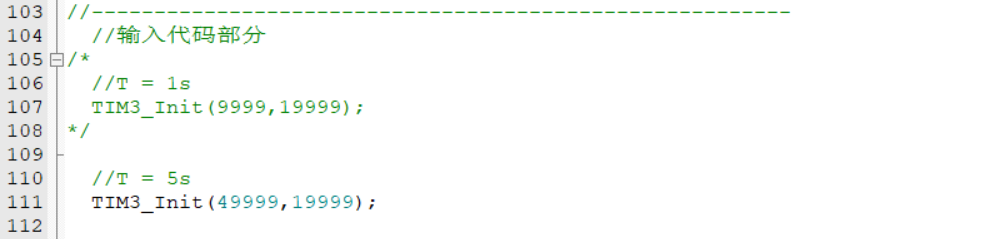


图 18：慢闪T=5s

慢闪视频附件。

2. 完成Nucleo板的底板**LED1（PE11，TIM1\_CH2）**呼吸灯实验：

* 创建定时器句柄和定时器1通道2句柄，如图19：



图 19：创建定时器句柄和通道

* 编写初始化TIM1的PWM函数，如图20所示：

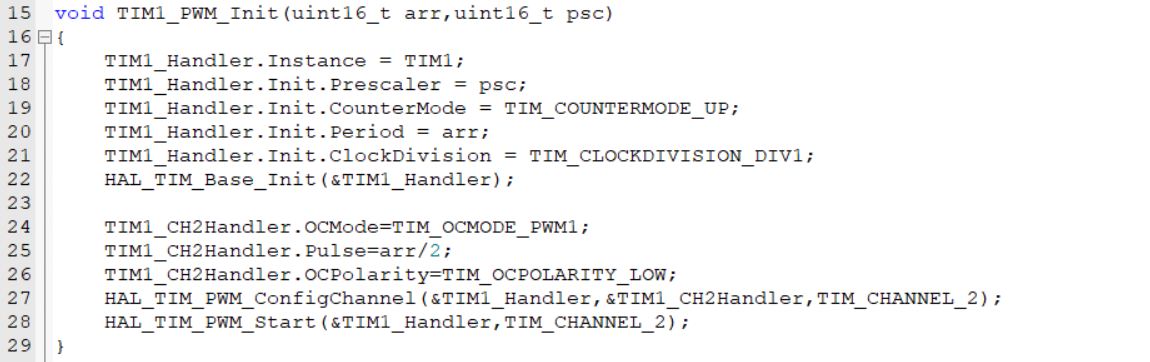


图 20：初始化TIM1的PWM函数

* 编写定时器底层驱动，如图21所示：

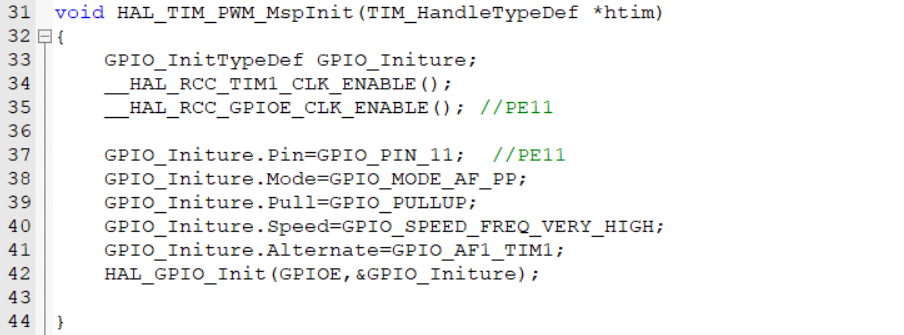


图 21：定时器底层驱动

* 设置TIM1通道2的占空比函数，如图22所示：

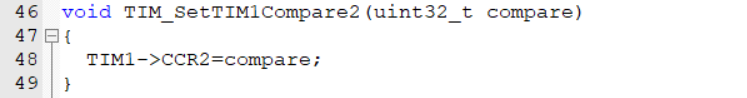


图 22：设置占空比函数

* 主函数编写：

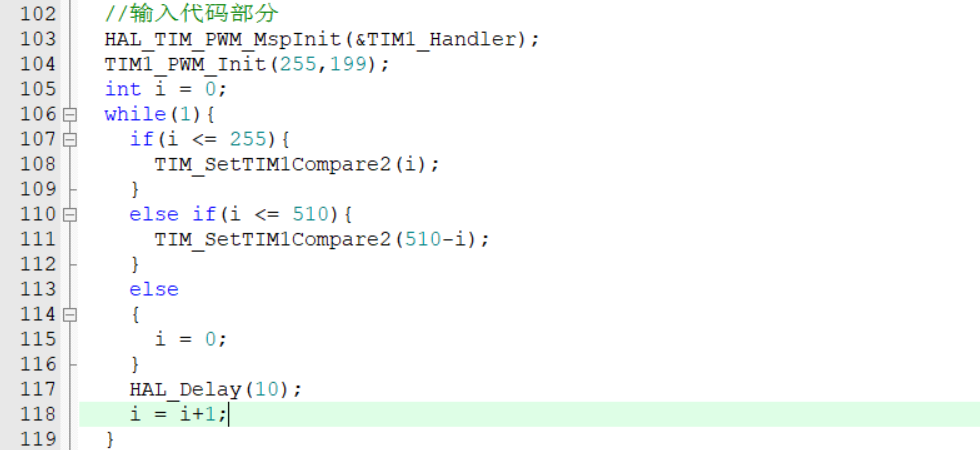


图 23：主函数

这里103行直接调用了HAL\_TIM\_PWM\_MspInit且程序能正常运行，视频见附件，按理说应该使用HAL\_TIM\_PWM\_Init，但因为实验平台在队友手里，不想麻烦他再跑一遍，因此没有验证HAL\_TIM\_PWM\_Init函数是否可行。

**（三）ADC实验：将光敏电阻两端电压通过ADC转换为数字信号，通过数字信号的大小控制PWM输出，从而控制底板LED1亮度**

* 初始化ADC并创建ADC句柄：

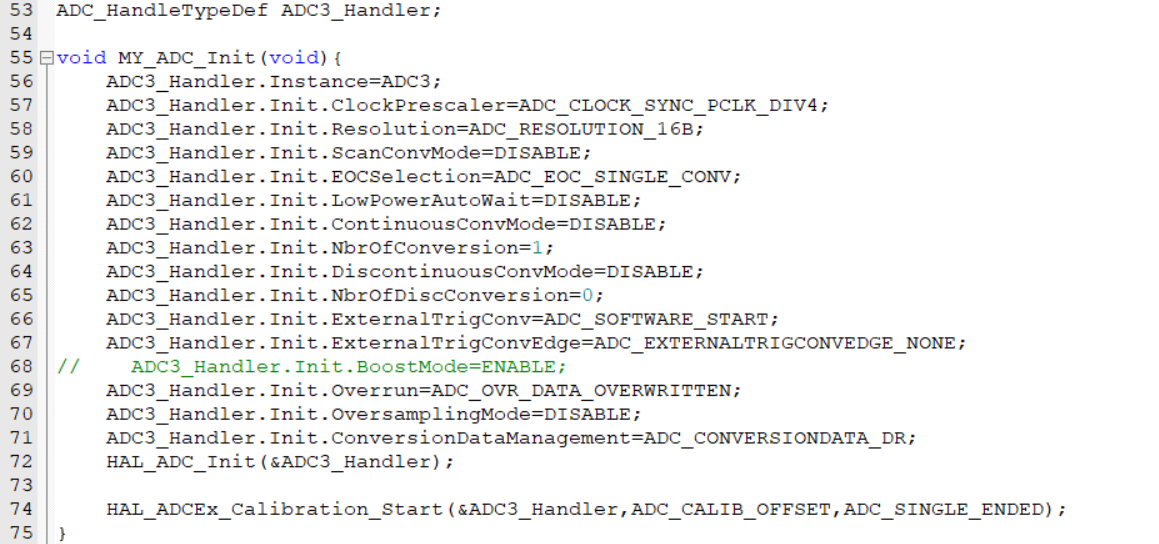


图 24：初始化ADC

* 编写ADC底层驱动：

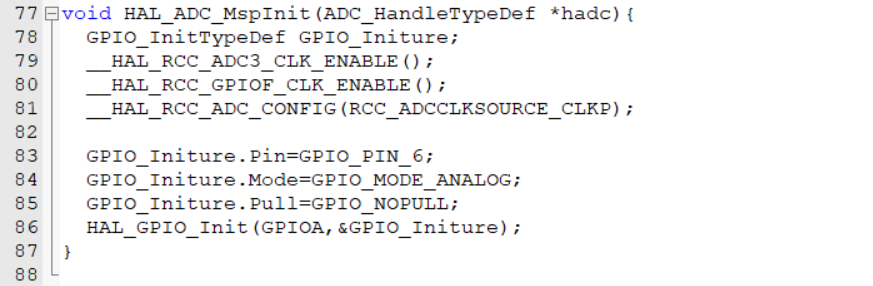


图 25：编写ADC底层驱动

* 获取ADC值的函数：

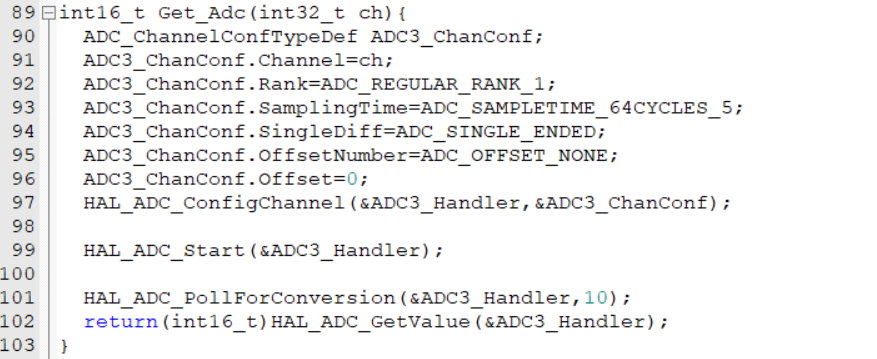


图 26：单次获取ADC值

* 获取指定通道的转换值，取times次，然后平均，后续主函数调用时times取5：

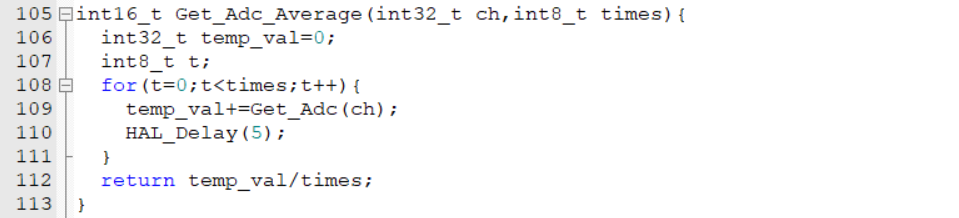


图 27：获得指定通道的转换值

* 主函数代码如图28所示，用手慢慢遮挡光敏电阻，此时光强减小，则底板LED1也越来越暗，视频见附件。

