基于定时器与串口的红外通信系统设计

一、整体设计思路

发射端电脑通过串口向其STM32开发板发送信号。此STM32接收到不同的信号时，按照不同编码方式生成红外发射管的驱动信号。此信号通过GPIO端口抵达红外驱动电路，红外发射管发出红外信号。

接收端的红外接收管将红外信号转变为电信号，通过GPIO端口反馈到接收端的STM32开发板。开发板进行解码识别，识别出发送端发送的数据，再通过串口显示在接收端的电脑上。

串口通信

需要发送的数据

串口通信

解码后的数据

发送端计算机

接收端计算机

发送端STM32

接收端STM32

调制

编码

*串口界面*

*串口界面*

红外波

红外

发送

模块

接收

解调

解调后的信号

解码

编码

实物图如下：



二、串口通信设计

此部分实现STM32开发板与电脑之间的通信。分为两部分：发射端电脑通过串口向其STM32开发板发送信号、接收端STM32将解码后的数据发送到接收端电脑。两端都用LPUART1串口，设置为异步通信模式，115200波特率，8位字长，无奇偶校验，停止位1。串口发送引脚LPUART1\_Tx映射到PA2，接收引脚LPUART1\_Rx映射到PA3。

在软件代码部分对其进行重定向之后方可使用。

注：映射之后STM32发送接收与电脑的发送接收是在内部交叉连接的（LPUART1\_Tx连接Rx，LPUART1\_Rx连接Tx，不难理解STM32与电脑一收一发），不需要单独接线。但如果没有进行映射，保持在其默认引脚时，需要按照上述逻辑单独用杜邦线连接对应引脚。

三、红外发射管与接收管的驱动电路设计

10u

400Ω

1KΩ

5V

100Ω

5V

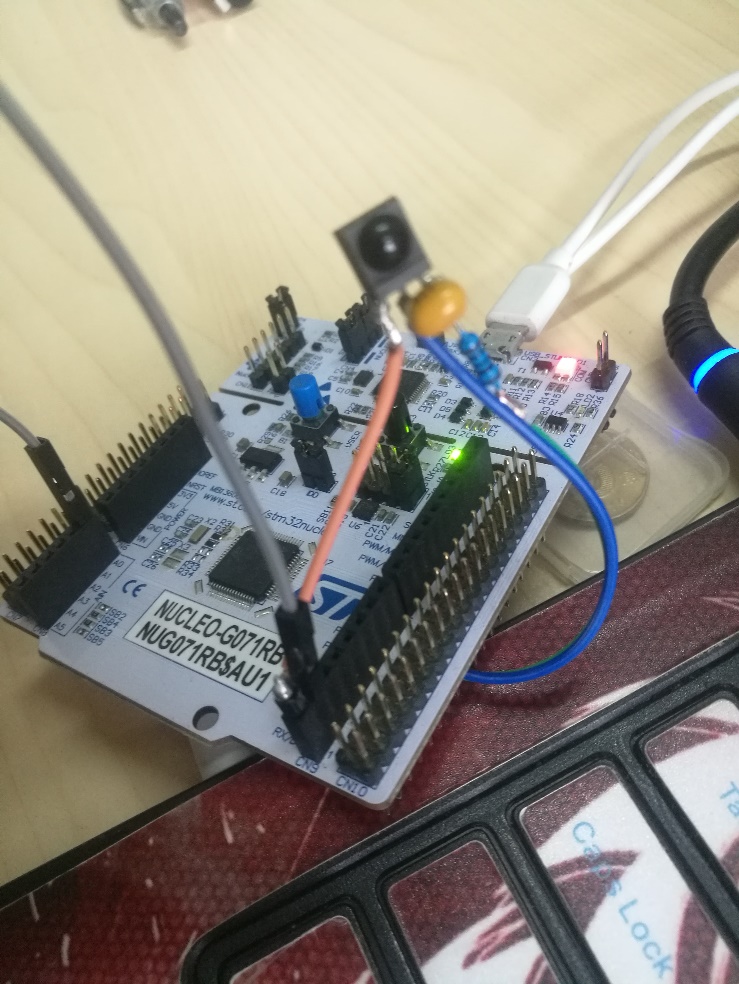
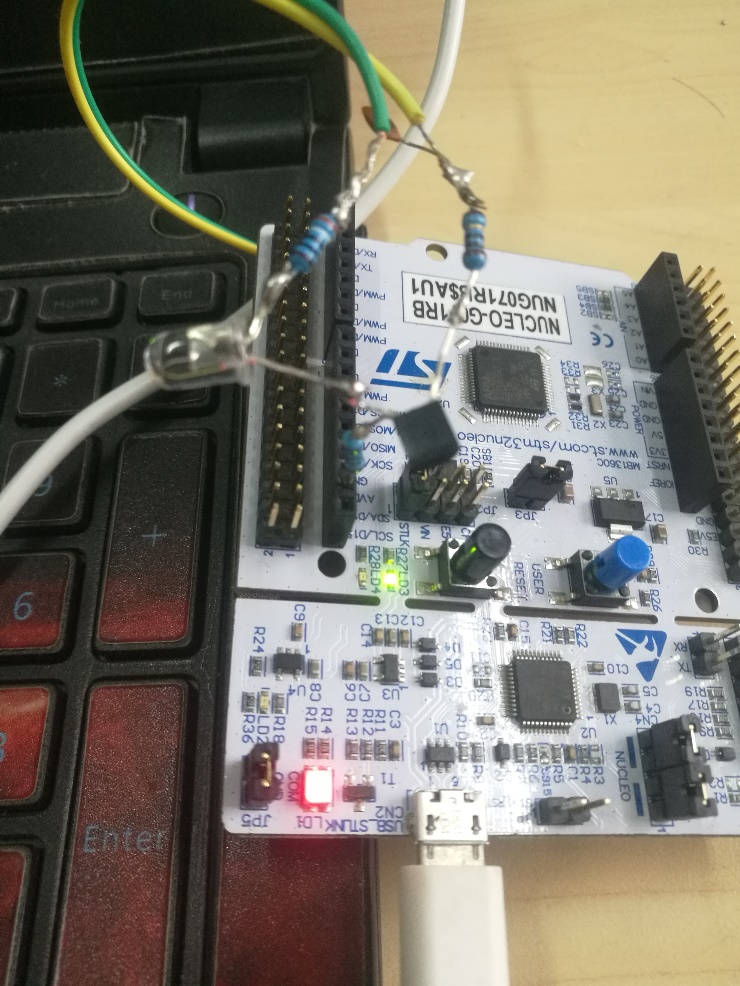
输入

10Ω

输出

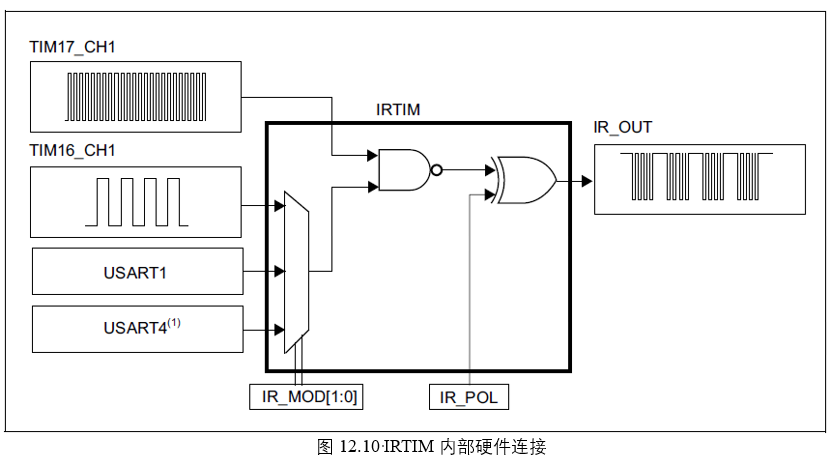
发射管输入信号为高电平时，三极管导通，三极管与红外发射管连接处电压被拉低二极管导通。发射管输入信号为低电平时，三极管截止，三极管与红外发射管连接处电压被拉高二极管截止。由下图可知，输入（第一路天蓝色信号）、二极管与三极管连接处（第二路紫色信号）呈现反相关系。





四、信号的调制与解调

STM32内部的IRTIM可实现调制功能。



在其输入端用定时器17通道1来生成载波，用定时器16通道1、串口1或串口4生成包络。此实验中我们选用定时器16通道1，两路信号经过与非门后调制成功，后边的异或门可以将生成的信号反相输出，但此实验中不用反相。上述四选一数据选择器与反相输出的设置分别在SYSCFG\_CFGR1的IR\_MOD[1:0]位与IR\_POL位。

与非门电路分析：包络（定时器16）为高电平，其调制输出端口信号IR\_OUT为载波信号反相；包络为低电平，其调制输出端口信号IR\_OUT为高电平。如下图中第一路黄色信号为定时器16输出信号（包络），第二路天蓝色信号为IRTIM调制载之后的信号。



通过查询资料不难发现，一般家用红外遥控器都是38Khz调制的红外信号。此实验中将系统时钟设置为最高的64Mhz，用于产生载波的定时器17工作在“通道1生成PWM”的模式下，预分频系数128，计数周期16，反转8（周期的一半）。载波信号频率为64Mhz/128/16=31.25Khz，周期为32us，占空比为50%，可应用于红外发射管。

产生信号包络的定时器16，同样工作在“通道1生成PWM”的模式下，预分频系数设置为128，计数周期320，反转160（周期的一半）。包络信号频率为64Mhz/128/320=1.56Khz，周期为641us，占空比为50%，便于接收端检测。

红外接收管与电容电阻搭建的红外接收模块可实现包络检波，输出解调信号。

通过示波器测量如下：



上图信号依次为：定时器16产生的包络信号（深蓝色），定时器17产生的载波信号（紫色），红外发射管发射信号（天蓝色），红外接模块输出的信号（金黄色）。其周期、频率非常接近理论值。红外发射管发射信号经过IRTIM与发送设备的三极管两次反向，所以其包络即定时器16产生的信号，并且相位相同。

五、信号的编码与解码

编码：通过控制定时器打开关闭的时间间隔，来控制发送的脉冲个数。本实验中对数字1到9进行了编码，需要发送的数据为INPUT。打开定时器16，延时INPUT\*5ms，关闭定时器16，延时200-INPUT\*5ms。即发送一个数据，需要200ms，其中定时器工作INPUT\*5ms。例如：发送数据1，定时器16工作5ms，即5ms/641us=7.8个周期，约为8个周期（定时器16产生8个脉冲）。以此类推，发送数据2时产生16个脉冲……

解码：解码部分系统时钟为64Mhz，解码时用到定时器2产生接收复位信号，其工作在“通道1生成PWM”的模式下，预分频系数1024，计数周期62500，反转31250（周期的一半）。接收复位信号频率为64Mhz/1024/62500=1hz，周期为1s，占空比为50%。当收到发送端发出信号的那一刻，打开定时器2开始计时,并记录收到的下降沿个数（也即高电平个数）。定时器2开始工作500ms后产生第一个下降沿触发中断，先根据记录的高电平数解码输出，再进行接收复位并关闭它自己，直到下一个发送端发来信号。解码算法：加四除八后赋值给整型数data（自动取整）。例如：上述编码部分提到发送数据1，定时器16工作5ms，产生8个脉冲，（8+4）/8=1.5取整为1，成功解码。

六、此红外通信系统的抗干扰设计

发送设备串口通信与发送端红外信号时序：接收到串口数据后，先延时100ms再对定时器16进行操作（此过程持续200ms即编码中提到的发送一个数据所需时间）以保证每个数据顺利发送，最后提示用户可以发送下一个数据，避免了上一个数据还没成功发送完毕，就又更新了串口数据内容。

红外发射管接收管工作时序：发送一个数据红外管工作时间为200ms，接收管复位时间为500ms大于发射管。确保了红外发射管发出的所有脉冲动能被接收管收到。

解码时不是一一对应而是采用加四除八的方式，极大程度上降低了误码率：例如收到4到11个脉冲，都可以解码为1，即便发送端没有准确地发出8个脉冲而是小范围波动依旧可以正确解码，其次此解码算法确保了每次发送数据的标准脉冲个数都在该接收区间的正中央（例如8在4-11的正中央，16在12到19正中央……）。还避免了红外发射管的误触发造成的影响，收到1、2或3个脉冲时，加四除八取整后为0即没接收到任何数据。

1. 工程创建与设计代码

发射端工程创建：①在时钟配置界面下，首先将系统时钟复用更改为锁相环时钟PLLCLK，系统时钟设置为64Mhz，AHB总线与APB总线分频系数均为1，此时可以看到送给定时器的时钟就是系统时钟64Mhz，LPUART1时钟复用选用PCLK其工作频率也是64Mhz。②定时器设置：定时器16通道1工作模式为“PWM Generation CH1”，预分频系数127，计数周期320，脉冲160。定时器17通道1工作模式为“PWM Generation CH1”，预分频系数127，计数周期16，脉冲8。③IRTIM输出不反相，信号来源选择为定时器16，调制信号输出IR\_OUT映射在PB9引脚。④串口LPUART1工作在异步模式，波特率115200，字长8位，无奇偶校验，停止位1位，将LPUART1\_Tx映射在PA2引脚，将LPUART1\_Rx映射在PA3引脚。

发射端硬件连接：串口引脚不需要单独接线，红外发射模块的输入信号线连接PB9引脚，并连接VCC（5V供电）与GND。

发射端设计代码：

1. 添加头文件、串口重定向声明

#include "stdio.h"

#include "string.h"

#define unit8\_t unsigned char

#define PUTCHAR\_PROTOTYPE int fputc(int ch,FILE \*f)

#define GETCHAR\_PROTOTYPE int fgetc(FILE \*f)

#define BACKSPACE\_PROTOTYPE int \_backspace(FILE \*f)

1. 主函数编写

在进入循环工作模式前打开定时器17

HAL\_TIM\_PWM\_Start(&htim17,TIM\_CHANNEL\_1);

进入while循环后输出提示信息，获取串口数据，通过控制定时器16产生包络，自动调制并生成红外发射电路驱动信号，驱动红外发射模块发射红外波。

printf("\r\n 请输入需要发送的数据: ");

scanf("%d",&INPUT);

HAL\_Delay(100);

if(INPUT>0&&INPUT<10)

{

HAL\_TIM\_PWM\_Start(&htim16,TIM\_CHANNEL\_1);

HAL\_Delay(INPUT\*5);

HAL\_TIM\_PWM\_Stop(&htim16,TIM\_CHANNEL\_1);

HAL\_Delay(200-INPUT\*5);

}

1. 串口重定向

PUTCHAR\_PROTOTYPE //重定向 fputc()函数

{

HAL\_UART\_Transmit(&hlpuart1,(unit8\_t\*) &ch,1,0xFFFF); //调用串口发送函数

return ch; //返回发送的字符

}

GETCHAR\_PROTOTYPE //重定向 fgetc()函数

{

uint8\_t value; //定义无符号字符型变量 value

while((LPUART1->ISR & 0x00000020)==0){} //判断串口是否接收到字符

value=(uint8\_t)LPUART1->RDR; //读取串口接收到的字符

HAL\_UART\_Transmit(&hlpuart1,(uint8\_t \*)&value,1,0x1000); //回显接收到的字符

return value; //返回接收到的值 value

}

BACKSPACE\_PROTOTYPE //重定向\_\_backspace 函数

{

return 0;

}

接收端工程创建：①接收端时钟配置与发射端时钟配置相同均为64Mhz。②定时器2通道1工作在“PWM Generation CH1”模式下，预分频系数1023，计数周期62500，脉冲31250。③串口LPUART1设置与输入端串口设置相同。④将PC4、PC5引脚均设置为下降沿触发的外部中断，并在NVIC界面下使能4到15号外部中断。

接收端硬件连接：定时器输出引脚PA0连接外部中断引脚PC4，红外接收模块反馈信号线接外部中断引脚PC5，并连接VCC（5V供电）与GND。

接收端设计代码：

1. 添加头文件、串口重定向声明、定义变量

#include "stdio.h"

#include "string.h"

#define unit8\_t unsigned char

#define PUTCHAR\_PROTOTYPE int fputc(int ch,FILE \*f)

#define GETCHAR\_PROTOTYPE int fgetc(FILE \*f)

#define BACKSPACE\_PROTOTYPE int \_backspace(FILE \*f)

volatile int number=0; //收到的脉冲个数

int data=0; //解码出的数据

int flag\_TIM2=0; //定时器2通道1标志位 1打开0关闭

1. 主函数编写

进入循环前提示用户接收准备就绪，复位按键按下之后执行。

printf("开始接收 \r\n");

所有操作均通过中断的回调函数执行，所以进入循环之后等待即可。

HAL\_Delay(10);

③串口重定向

PUTCHAR\_PROTOTYPE //重定向 fputc()函数

{

HAL\_UART\_Transmit(&hlpuart1,(unit8\_t\*) &ch,1,0xFFFF); //调用串口发送函数

return ch; //返回发送的字符

}

GETCHAR\_PROTOTYPE //重定向 fgetc()函数

{

uint8\_t value; //定义无符号字符型变量 value

while((LPUART1->ISR & 0x00000020)==0){} //判断串口是否接收到字符

value=(uint8\_t)LPUART1->RDR; //读取串口接收到的字符

HAL\_UART\_Transmit(&hlpuart1,(uint8\_t \*)&value,1,0x1000); //回显接收到的字符

return value; //返回接收到的值 value

}

BACKSPACE\_PROTOTYPE //重定向\_\_backspace 函数

{

return 0;

}

④编写中断回调函数

void HAL\_GPIO\_EXTI\_Falling\_Callback(uint16\_t GPIO\_Pin)

{

if(GPIO\_Pin==0x0010) //PC4 定时器中断触发-----输出+清零

{

//计时结束关闭定时器2通道1，并更新定时器2标志位

HAL\_TIM\_PWM\_Stop(&htim2,TIM\_CHANNEL\_1);

flag\_TIM2=0;

data=(number+4)/8; //根据脉冲个数解码出数据

if(data!=0) //数据为0说明：发送数据或红外干扰不输出忽视即可

printf(" 收到数据 %d \r\n",data); //否则通过串口显示在接收端电脑

number=0; //脉冲计数清零准备新一轮接收

}

else if(GPIO\_Pin==0x0020) //PC5红外所接收中断触发

{

if(flag\_TIM2==0) //如果定时器处于关闭状态则将其打开

{ //如果定时器已经打开则不对其进行操作

HAL\_TIM\_PWM\_Start(&htim2,TIM\_CHANNEL\_1);

flag\_TIM2=1;

}

number++; //PC5中断每触发一次说明接收到一个脉冲，脉冲计数加一

}

}

八、结果展示

