**西南交通大学**

**嵌入式系统实验日志**



**实验名称：**蓝牙通信实验

**实验序号：**7

**实验组号：**3组-6队

**使用语言：**Python

**姓名学号：**杨志成 2023211042

**实验时间：**2023年11月24日

**指导老师：**徐图

**一、实验目的：**

1、 了解蓝牙模块和树莓派板载蓝牙的通信方式

2、 了解树莓派串口控制方法

3、 掌握树莓派的编程方法

**二、实验步骤：**

1、 连接：用杜邦线，将各部分连接起来；

2、 安装蓝牙相关的软件，连接板载蓝牙和 HC-05 蓝牙模块

3、 通过蓝牙模块发送和接收信息，点亮 LED 灯；

4、 要求做两个程序，一个用于输入 0-255 的数字，并通过蓝牙发送；另一个是接收程序， 接收到数字以后，将其转化为二进制数，并用于点亮 8 位流水灯，其中高电平点亮灯， 低电平熄灭。

**三、硬件连接图，软件流程图：**

树莓派：接口定义（BCM）

17——三色LED D0

1——三色LED D1

27——三色LED D2

22——三色LED D3

23——三色LED D4

24——三色LED D5

25——三色LED D6

4——三色LED D7

蓝牙模块

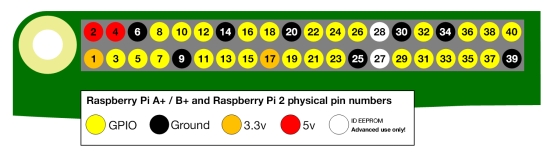
TXD——USB模块RXD

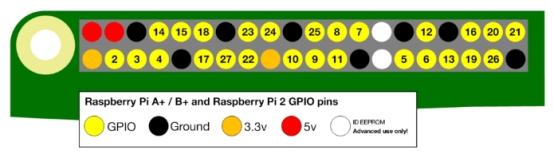
RXD——USB模块TXD

USB转串口模块

TXD——蓝牙模块RXD

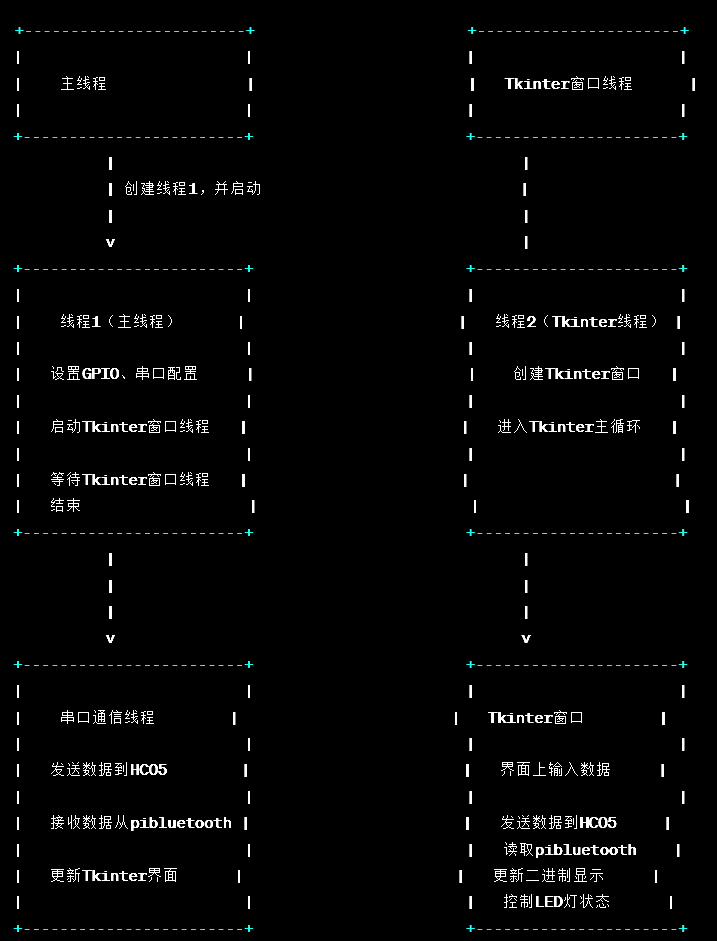
RXD——蓝牙模块TXD







软件流程图：



**四、关键代码分析：**

import RPi.GPIO as GPIO

import threading

import tkinter as tk

from tkinter import ttk

import serial

GPIO.setwarnings(False)

GPIO.setmode(GPIO.BCM)#设置GPIO口工作模式

LED\_List=[17, 1, 27, 22, 23, 24, 25, 4]

GPIO.setup(LED\_List, GPIO.OUT, initial=GPIO.HIGH)

#serial configuration

ser\_HC05 = serial.Serial('/dev/ttyUSB0',9600,timeout = 0.5) #根据串口的连接路径，创建一个“serial”对象，用于USB连接串行口

ser\_pibluetooth = serial.Serial('/dev/rfcomm0',9600,timeout = 0.5) #创建一个“serial”对象，使用蓝牙连接串行口，以及超时时间为0.5s

ser\_HC05.isOpen() #打开端口

ser\_pibluetooth.isOpen()

def control\_leds(value):

binary\_str = format(value, '08b') # Convert to 8-bit binary string

for i in range(8):

if binary\_str[i] == '1':

GPIO.output(LED\_List[i], GPIO.LOW) # Turn on LED

else:

GPIO.output(LED\_List[i], GPIO.HIGH) # Turn off LED

def bluetooth\_communication():

global text2

message = text1.get("1.0","end-1c")#从text组件上读取输入的数据，int类型

ser\_HC05.write(message.encode()) # 发送数据

print("已发送数据:" + message)

ser\_pibluetooth.inWaiting() #检查缓冲区的字节内容，判断是否有数据待读取

receive\_msg = ser\_pibluetooth.read(3).decode() # 将接收数据解码，并只读取前三位，因为只传了三位

print("已接收到数据:" + receive\_msg)

binary\_str = format(int(receive\_msg), '08b') # 转换为8位二进制

text2.delete('1.0', tk.END) # 清除文本框中的旧内容

text2.insert(tk.INSERT, binary\_str) # 插入新的二进制数据

control\_leds(int(receive\_msg))

lock = threading.Lock() # 设置线程锁

def thread2\_tk():

global text1, text2,bt\_auto,windows#全局变量的组件和窗口句柄

if lock. acquire(1): #判断是否能获取到锁

windows=tk. Tk()#窗口句柄

windows. title("蓝牙通信")#窗口名称

windows.geometry('400x200') # 规定窗口大小像素

windows.resizable(False, False) # 规定窗口不可缩放

# 创建一个容器Frame来居中放置所有的元素

center\_frame = ttk.Frame(windows)

center\_frame.place(relx = 0.5, rely = 0.5, anchor = 'center')#设置frame的大小位window的1/2，且中心重合

# 使用自定义样式

style = ttk.Style()

style.configure('TLabel', font = ('Helvetica', 12))#设置标签的格式

style.configure('TButton', font = ('Helvetica', 12),foreground ='green')

lab1 = ttk.Label(center\_frame, text = '输入发送的数据(0~255): ', style = 'TLabel')

lab1.grid(row = 0, column = 0, padx = 10, pady = 5)

text1 = tk.Text(center\_frame, width = 10, height = 1, font = ('Helvetica', 11))

text1.grid(row = 0, column = 1, padx = 10, pady = 10)

lab2 = ttk.Label(center\_frame, text = '八位二进制形式: ', style = 'TLabel')

lab2.grid(row = 1, column = 0, padx = 5, pady = 5)

text2 = tk.Text(center\_frame, width = 10, height = 1, font = ('Helvetica', 11))

text2.grid(row = 1, column = 1, padx = 10, pady = 5)

bt\_send = ttk.Button(center\_frame, text = '发送', style = 'TButton',command = bluetooth\_communication ) #添加按钮，并绑定回调函数

bt\_send.grid(row = 2, column = 0,columnspan=2, padx = 20, pady = 10,sticky="ew")#横跨两行，居中

lock.release() # 释放锁

windows. mainloop()#循环显示界面内容

if \_\_name\_\_ == "\_\_main\_\_":

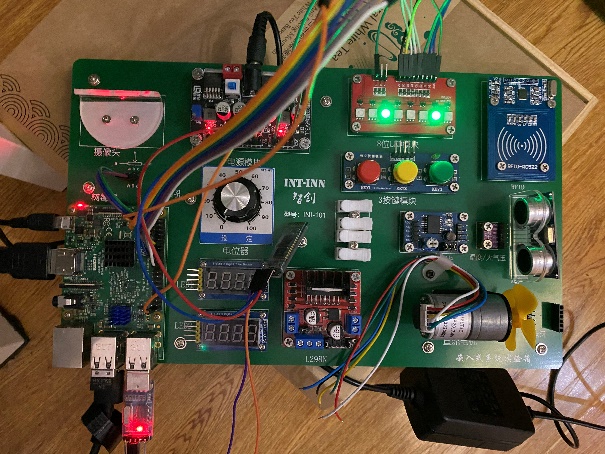
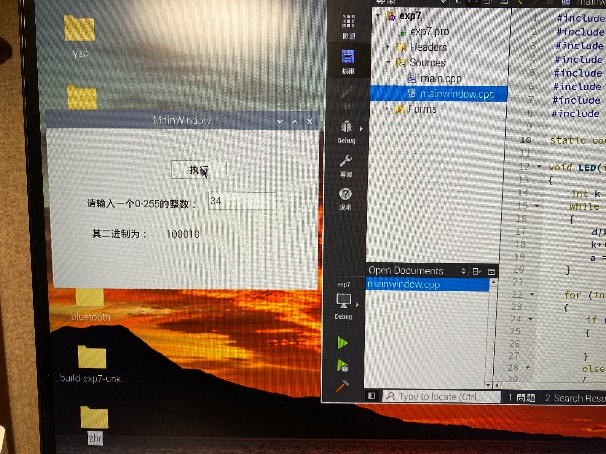
thread1 = threading.Thread(target = thread2\_tk, args = ())

thread1.start()

thread1.join()

**五、实验结果：**

实现了将蓝牙与树莓派串口通信，输入一个0-255的数字，通过蓝牙发送之后，将其转化为二进制字符串，根据每一个字符串对相应的LED灯的引脚加高低电平。

可以看到在ui界面显示的是发送的二进制字符串，在树莓派LED上显示的是相应二进制字符串的亮与灭。

**六、实验中遇到的问题及处理方法：**

根据实验指导书上的连接步骤搜索连接蓝牙，发现连接不上，蓝牙模块一直找不到，后来发现是蓝牙模块坏了，找组长更换蓝牙之后就好了。

**七、实验中，自己的创意部分：**

多线程设计： 使用了多线程来处理不同的任务。主线程用于初始化硬件配置和启动 Tkinter 窗口线程，而串口通信线程负责实际的数据通信和界面更新，这样可以使界面保持响应，不会因为串口通信而阻塞。

Tkinter GUI： 使用了 Tkinter 构建了一个简单的 GUI 界面，这对于嵌入式系统控制来说是一种直观的用户界面。用户可以通过界面输入数据，同时能够实时看到数据的二进制形式，提高了用户交互的友好性。

**八、实验中的收获：**

1、 了解了蓝牙模块和树莓派板载蓝牙的通信方式

2、 了解了树莓派串口控制方法

3、 掌握了树莓派的编程方法