****

A级达标线上测试报告



**学院** 计算机科学与技术学院 **专业**  计算机科学与技术

**学号**  21009200162

**姓名** 李文卓

**手机** 18966576472 **完成日期** 2024-10-03

**成绩**

# 题目名称: 湿度监测仿真系统

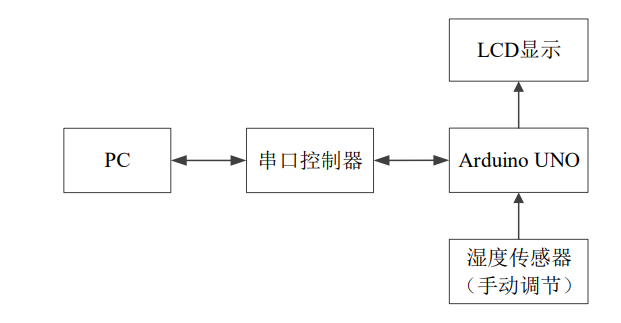
一、题目要求

软件环境：推荐采用 Proteus 8.9 SP2 及以上仿真软件，Arduino IDE，虚拟串口

驱动软件 Virtual Serial Port Driver（VSPD）。

实现功能：使用Arduino UNO微控制器，搭建一个PC上位机远程湿度监测系统。

·系统框图如下：



·功能：Arduino UNO（Atmega328P）通过串行接口组件与上位机 PC 进行双向

通信，PC 上位机软件向 Arduino UNO 发送学生自己的学号，Arduino UNO 收到

后在 LCD 上显示学生的学号，并且向 PC 机发送当前的湿度值。PC 上位机软件

显示收到的湿度值。

LCD 第一行显示 ID：学号，第二行显示 RH: 湿度值%

自行编写 PC 上位机软件，实现 PC 与 Arduino 的双向数据传输及管理控制。编

程语言不限，推荐采用 C#。

上位机软件 GUI 界面需要有发送窗口显示发送的学号，有接收窗口显示接收到

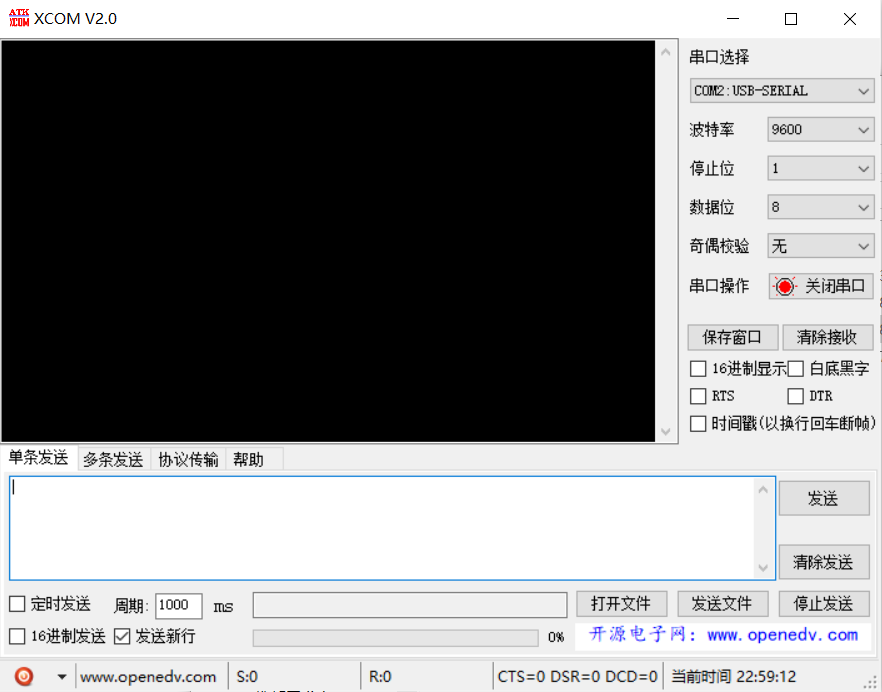
的湿度值，GUI 界面上需要有串口选择和串口打开关闭功能。

二、设计思路

采用两种方式实现上位机软件的UI界面：

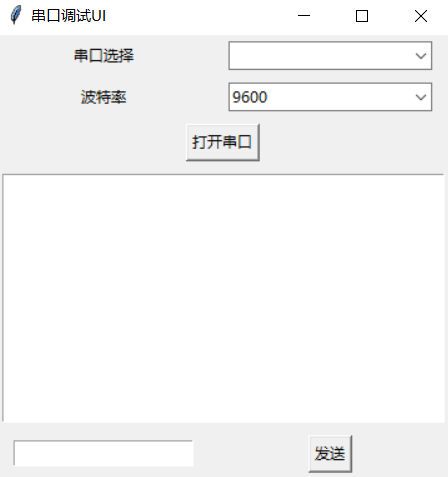
1．采用XCOM开源软件实现

上位机软件GUI界面截图

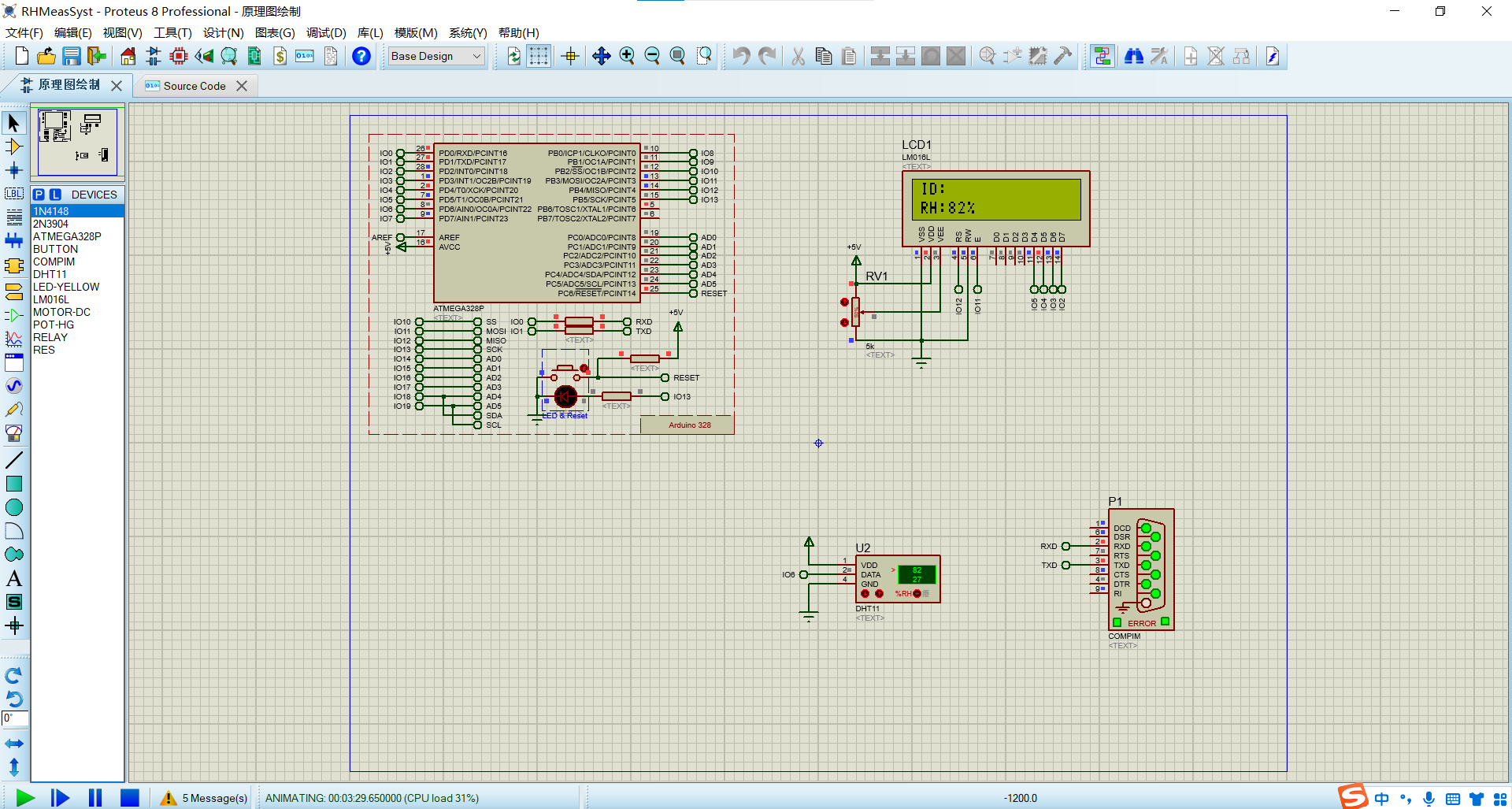


2.采用python实现UI界面：

上位机软件GUI界面截图

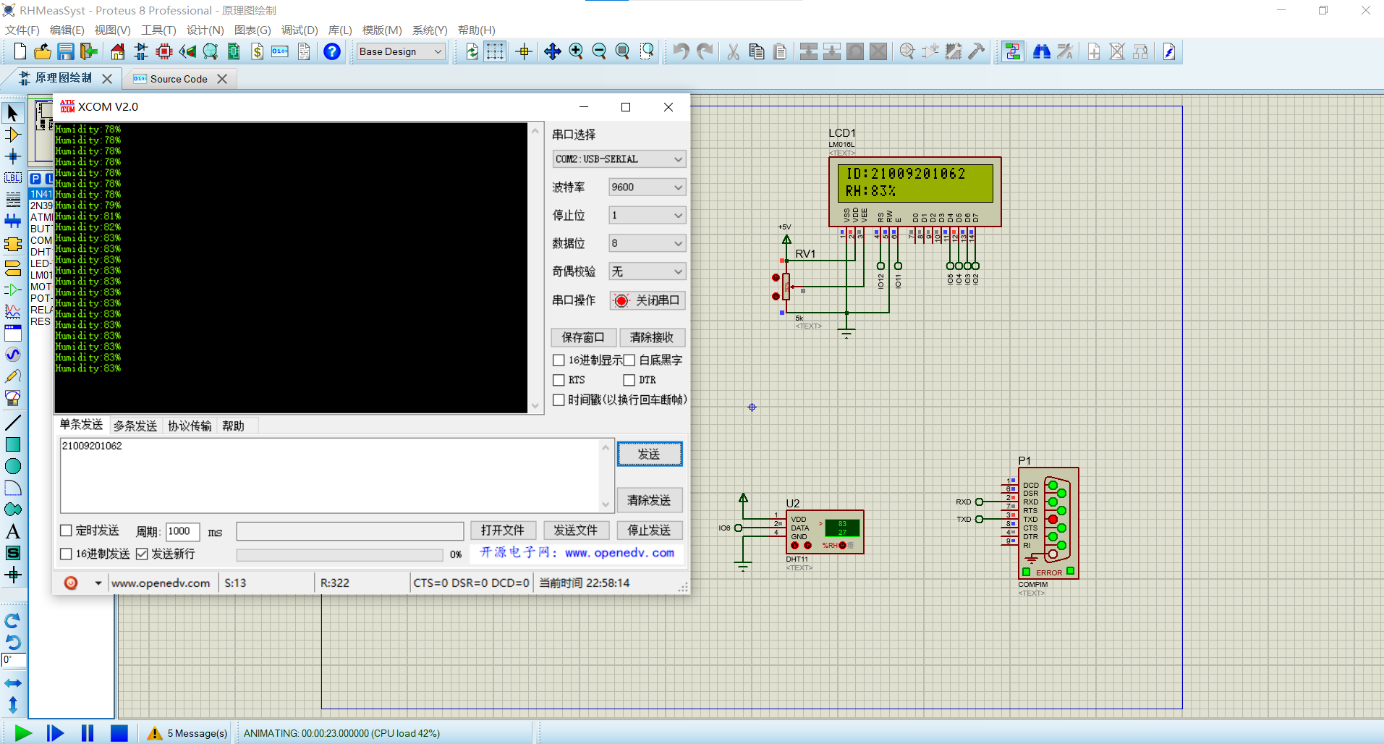


完整仿真电路截图

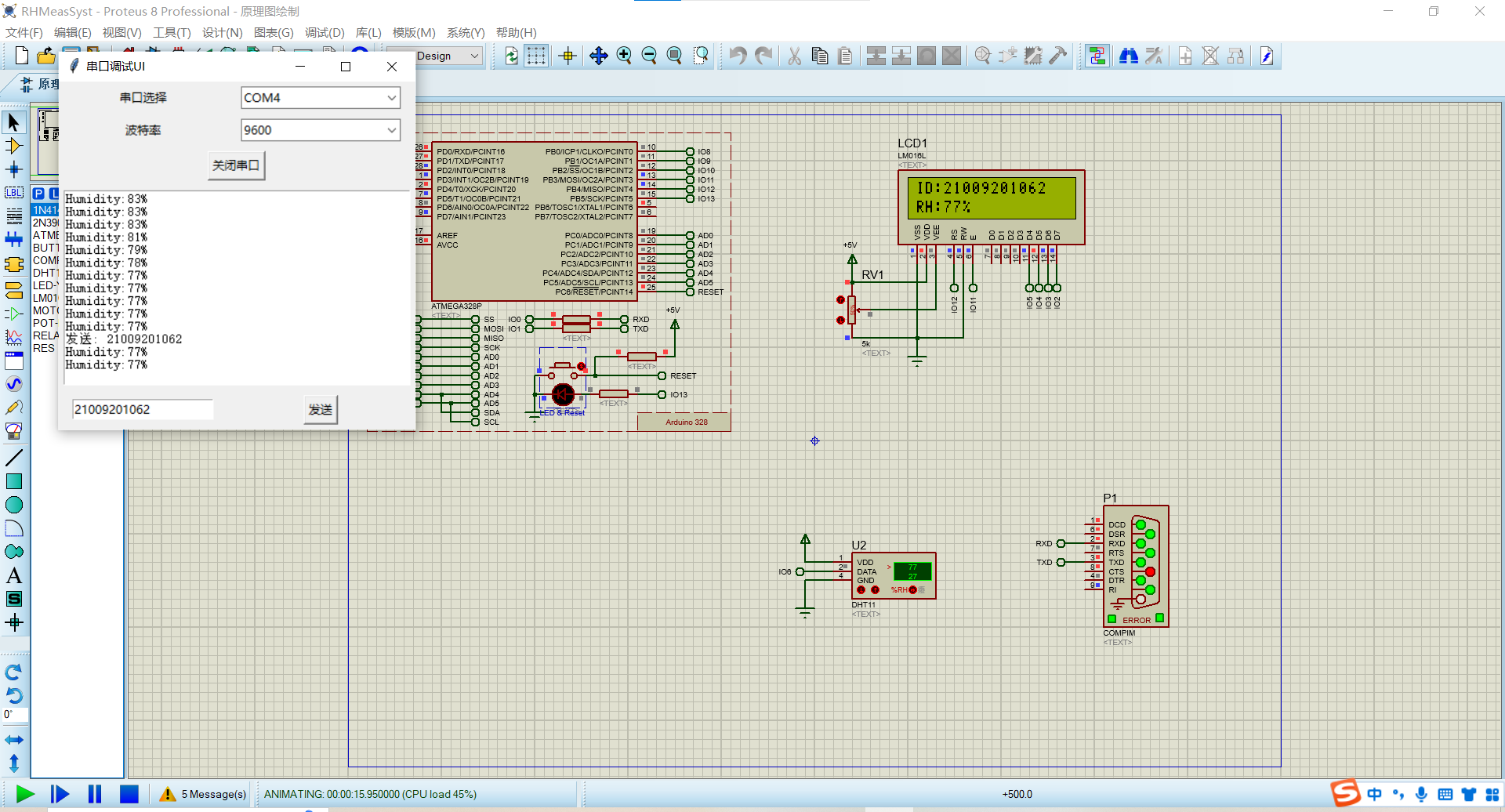


三、仿真结果展示

采用方法1的结果截图：



采用方法2的结果截图：



四、参考文献

1. Proteus 安装与使用文档

2.VSPD使用教程

3. DHT11传感器介绍<https://projecthub.arduino.cc/arcaegecengiz/using-dht11-12f621>

4. Arduino IDE介绍文档

五、程序设计

Arduino程序源代码：

# include <Wire.h>

# include <dht11.h> // DHT11库

# include <LiquidCrystal.h> // LCD控制相关库

// 定义引脚

# define DHT11PIN 6

# define EMPIN 7

LiquidCrystal lcd(12, 11, 5, 4, 3, 2); // 初始化LCD

dht11 DHT11;

String str = ""; // 定义字符串str，接收学号

int length\_, last = 0;

int setHumidity, realHumidity = 0;

void setup() {

// put your setup code here, to run once:

Serial.begin(9600); // 初始化串口，设置波特率为9600

lcd.begin(16, 2);

pinMode(EMPIN, OUTPUT); // 设置连接直流电机引脚工作模式

}

void loop() {

// put your main code here, to run repeatedly:

DHT11.read(DHT11PIN);

realHumidity = (int)DHT11.humidity;

lcd.print("ID:");

if (Serial.available() > 0) { // 串口接收到数据

while (Serial.available() > 0) {

str += (char)Serial.read();

delay(10);

}

lcd.println(str);

length\_ = str.length();

last = str.substring(length\_ - 3, length\_ - 2).toInt(); // 取末尾数字

setHumidity = 30 + last;

str = "";

length\_ = 0;

last = 0;

}

lcd.setCursor(0, 1); // 将光标定位在第0列，第1行

lcd.print("RH:");

lcd.print(realHumidity);

lcd.println("%");

lcd.home(); // 光标返回左上角

Serial.print("Humidity:");

Serial.print(realHumidity);

Serial.println("%");

if (realHumidity <= setHumidity) {

digitalWrite(EMPIN, HIGH); // 电机转动

} else {

digitalWrite(EMPIN, LOW);

}

delay(1000);

}

上位机程序源代码：

import tkinter as tk

from tkinter import ttk

from threading import Thread

try:

import serial

import serial.tools.list\_ports

pyserial\_installed = True

except ImportError:

pyserial\_installed = False

class SerialGUI:

def \_\_init\_\_(self, root):

self.root = root

self.root.title("串口调试UI")

# 串口相关

self.serial\_port = None

self.is\_open = False

# 检查 pyserial 是否安装

if not pyserial\_installed:

self.text\_area = tk.Text(root, height=15, width=50)

self.text\_area.grid(row=0, column=0, columnspan=2, padx=5, pady=5)

self.text\_area.insert(tk.END, "pyserial 库未安装，请运行以下命令安装:\n\npip install pyserial")

return # 停止初始化 GUI

# 串口选择框

self.port\_label = tk.Label(root, text="串口选择")

self.port\_label.grid(row=0, column=0, padx=5, pady=5)

self.port\_combo = ttk.Combobox(root, values=self.get\_serial\_ports())

self.port\_combo.grid(row=0, column=1, padx=5, pady=5)

# 波特率

self.baud\_label = tk.Label(root, text="波特率")

self.baud\_label.grid(row=1, column=0, padx=5, pady=5)

self.baud\_combo = ttk.Combobox(root, values=["9600", "115200"], state="readonly")

self.baud\_combo.set("9600")

self.baud\_combo.grid(row=1, column=1, padx=5, pady=5)

# 打开/关闭串口按钮

self.toggle\_btn = tk.Button(root, text="打开串口", command=self.toggle\_serial)

self.toggle\_btn.grid(row=2, column=0, columnspan=2, padx=5, pady=5)

# 数据显示区

self.text\_area = tk.Text(root, height=15, width=50)

self.text\_area.grid(row=3, column=0, columnspan=2, padx=5, pady=5)

# 发送数据区

self.entry = tk.Entry(root)

self.entry.grid(row=4, column=0, padx=5, pady=5)

self.send\_btn = tk.Button(root, text="发送", command=self.send\_data)

self.send\_btn.grid(row=4, column=1, padx=5, pady=5)

def get\_serial\_ports(self):

"""获取可用串口"""

if pyserial\_installed:

ports = serial.tools.list\_ports.comports()

return [port.device for port in ports]

return []

def toggle\_serial(self):

"""打开或关闭串口"""

if self.is\_open:

self.close\_serial()

else:

self.open\_serial()

def open\_serial(self):

"""打开串口"""

port = self.port\_combo.get()

baudrate = self.baud\_combo.get()

try:

self.serial\_port = serial.Serial(port, baudrate, timeout=1)

self.is\_open = True

self.toggle\_btn.config(text="关闭串口")

self.start\_reading()

except Exception as e:

self.text\_area.insert(tk.END, f"无法打开串口: {e}\n")

def close\_serial(self):

"""关闭串口"""

if self.serial\_port:

self.serial\_port.close()

self.is\_open = False

self.toggle\_btn.config(text="打开串口")

def start\_reading(self):

"""启动串口读取线程"""

self.read\_thread = Thread(target=self.read\_data)

self.read\_thread.daemon = True

self.read\_thread.start()

def read\_data(self):

"""从串口读取数据"""

while self.is\_open:

try:

data = self.serial\_port.readline().decode('utf-8')

if data:

self.text\_area.insert(tk.END, data)

except Exception as e:

self.text\_area.insert(tk.END, f"读取错误: {e}\n")

def send\_data(self):

"""发送数据"""

data = self.entry.get()

if self.serial\_port and self.is\_open:

self.serial\_port.write(data.encode('utf-8'))

self.text\_area.insert(tk.END, f"发送: {data}\n")

else:

self.text\_area.insert(tk.END, "串口未打开\n")

# 启动GUI

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

root = tk.Tk()

gui = SerialGUI(root)

root.mainloop()