**西南交通大学**

**嵌入式系统实验日志**

****

**学号：**杨志成

**姓名：**2023211042

**组号：**3组

**队号：**6队

**实验序号：**2

**实验名称：**数码管控制

**使用语言：**Python

1. **实验目的**

1、 了解 4 位 LED 显示模块的结构和控制方法

2、 掌握利用 Python 语言，通过树莓派的 GPIO 口控制 4 位 LED 模块的显示

3、 学习 I2C 接口规范

4、 学习 PCF8591 AD/DA 转换模块的使用方法，并利用树莓派的 I 2 C 接口读取 PCF8591 的 数据，并显示在 LED 显示模块中

5、 利用 TK 编写一个窗口界面，在界面中显示 PCF8591 模块的数据

1. **实验内容**

本实验以树莓派作为控制器，以 Python 作为开发工具，利用树莓派的 GPIO，控制 2 个 4 位 LED 显示模块，让其显示指定数据，利用 AD/DA 转换模块，读取其板载的热敏电 阻值、光敏电阻值、可变电阻的分压值，并在 LED 模块中显示；利用 TK 编写一个窗口界 面，将 PCF8591 模块的三个数据实时显示出来。

1. **实验步骤**

1、 连接：用杜邦线，将树莓派、电源模块、2 个 LED 模块、PCF8591 模块的相应引脚连 起来；

2、 编制 LED 模块显示程序，能显示给定数字；

3、 编制 PCF8591 模块控制程序，将光敏电阻、可变电阻的分压值读出，由两个 LED 模块 显示，调节光敏电阻和可变电阻，用多线程编程，一个线程采集数据，一个线程显示数据， 调解可变电阻或者改变环境光线（可用手机电筒照射光电管），观察数字的变化情况；

4、 编制 TK 的窗口界面，在显示器上显示热敏电阻、光敏电阻、可变电阻的分压值，调节光敏电阻和可变电阻，观察数字的变化情况。

1. **硬件连接图,软件流程图**

硬件连接图：

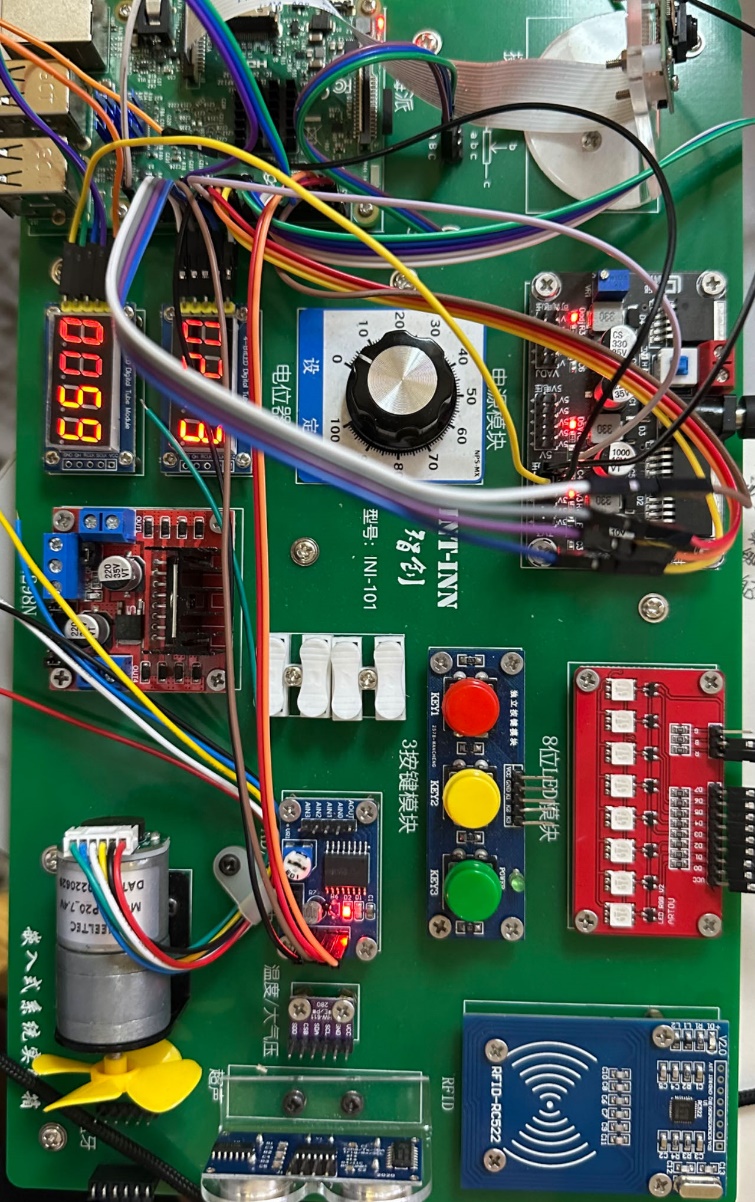
****

图1 硬件连接图

软件流程图：

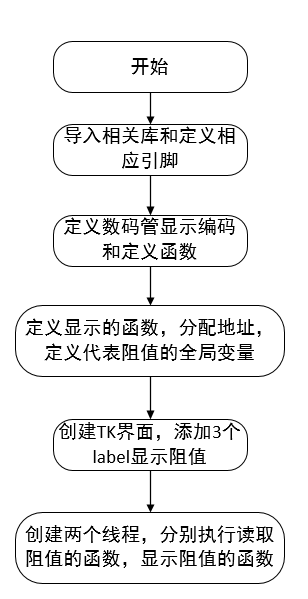


图2 软件流程图

1. **关键代码分析**

import RPi.GPIO as GPIO

import time

import smbus

import threading

GPIO.setmode(GPIO.BCM)

#定义引脚

SCLK1=25

RCLK1=8

DIO1=7

SCLK2=5

RCLK2=4

DIO2=17

GPIO.setwarnings(False)#不报警告

GPIO.setup([SCLK1,SCLK2,RCLK1,RCLK2,DIO1,DIO2] ,GPIO.OUT)#设置模式

#数码管显示数字定义

LED\_Library=[0xC0,0xF9,0xA4,0xB0,0x99,0x92,0x82,0xf8,0x80,0x90,

0xBF,0xC6,0xA1,0x86,0x8E,0xbf]

#输入显示的数据和显示数字的led引脚列表

def Show(i\_data,RCLK,SCLK,DIO):

i\_show4=i\_data % 10

i\_data=i\_data // 10

i\_show3=i\_data % 10

i\_data=i\_data // 10

i\_show2=i\_data % 10

i\_data=i\_data // 10

i\_show1=i\_data % 10

i\_data=i\_data // 10

LED4\_Display(i\_show1,0x08,RCLK,SCLK,DIO)#显示第一位

LED4\_Display(i\_show2,0x04,RCLK,SCLK,DIO)#显示第二位

LED4\_Display(i\_show3,0x02,RCLK,SCLK,DIO)#显示第三位

LED4\_Display(i\_show4,0x01,RCLK,SCLK,DIO)#显示第四位

# 显示一个数字在其中一个数码管上，以及显示的位置

def LED4\_Display(i\_index,hx\_location,RCLK,SCLK,DIO):

LED\_OUT(LED\_Library[i\_index],SCLK,DIO)#输入显示的数码位

LED\_OUT(hx\_location,SCLK,DIO)

GPIO.output(RCLK,GPIO.LOW)#先低再高脉冲

GPIO.output(RCLK,GPIO.HIGH)

# 字模和位地址都从DIO口输入，然后通过SCLK和RCLK对数据进行锁存

def LED\_OUT(X,SCLK,DIO):

for i in range(0,8):

if(X&0x80):#从高到底依次检查位数高输出1，低输出0

GPIO.output(DIO,GPIO.HIGH)

else:

GPIO.output(DIO,GPIO.LOW)

GPIO.output(SCLK,GPIO.LOW)

GPIO.output(SCLK,GPIO.HIGH)

X <<=1

address = 0x48

A0=0x40

A1=0x41

A2=0x42

A3=0x43

bus=smbus.SMBus(1)

value1=0

value2=0

value3=0

def Read\_data():

global value1,value2,value3

while True:

#读取3个模拟值

bus.write\_byte(address,A1)

value1=bus.read\_byte(address)

bus.write\_byte(address,A2)

value2=bus.read\_byte(address)

bus.write\_byte(address,A3)

value3=bus.read\_byte(address)

def Display\_data():

while True:

disp1=value2

disp2=value3

label1.config(text=value1)

label1.config(text=value2)

label1.config(text=value3)

for i in range(1000):

#显示模拟值

Show(disp1,RCLK1,SCLK1,DIO1)

Show(disp2,RCLK2,SCLK2,DIO2)

import tkinter as tk

import threading

# 创建一个400x400像素的窗口

window = tk.Tk()

window.geometry("400x400")

#创建3个标签并将它们放置在窗口中央

label1 = tk.Label(window, text="0",font=("Arial",16))

label1.place(relx=0.5, rely=0.33, anchor="center")

label2 = tk.Label(window, text="0",font=("Arial",16))

label2.place(relx=0.5, rely=0.5, anchor="center")

label3 = tk.Label(window, text="0",font=("Arial",16))

label3.place(relx=0.5, rely=0.67, anchor="center")

threadd1=threading.Thread(target=Read\_data,args=())

threadd1.start()

threadd2=threading.Thread(target=Display\_data,args=())

threadd2.start()

# 运行窗口

window.mainloop()。

1. **实验结果**

在树莓派上可以看到读取的光敏电阻和热敏电阻的阻值正常显示在树莓派的数码管上，在raspibian系统上的界面中也可以正常显示3个阻值，包括光敏阻值和热敏阻值，以及可变电阻阻值显示在创建好的TK界面中。

1. **实验遇到的问题及处理方法**
2. 安装了I2C-Tool之后查不到设备的地址

解决方法：重新连接好硬件电路就能正常显示。

2.听过SSH连接开启I2C服务

解决方法： 执行下列命令进行树莓派配置。Sudo raspi-config

选择第3个Interface Options，开启I2C服务，然后用i2cdetect -y 1查询设备。

1. **实验中，自己的创意部分**

多线程自由控制读取和显示的数据。

1. **实验中的收获**

1、 了解 了4 位 LED 显示模块的结构和控制方法

2、 掌握了利用 Python 语言，通过树莓派的 GPIO 口控制 4 位 LED 模块的显示

3、 学习了 I2C 接口规范

4、 学习了 PCF8591 AD/DA 转换模块的使用方法，并利用树莓派的 I 2 C 接口读取 PCF8591 的 数据，并显示在 LED 显示模块中

5、 学会了利用 TK 编写一个窗口界面，在界面中显示 PCF8591 模块的数据