**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：**4

**实验名称：**超声波测距

**使用语言：**Python

1. **实验目的**

1、 学习超声波测距原理

2、 了解超声波测距模块的控制方式

3、 了解 4 位 LED 数码管的显示控制方法

4、 通过树莓派编程，完成超声波测距功能

5、 编写 Tkinter 界面，将测距显示在界面中

1. **实验内容**

本实验以树莓派开发板作为控制板，通过树莓派的 GPIO 口连接超声波模块和 4 位 LED

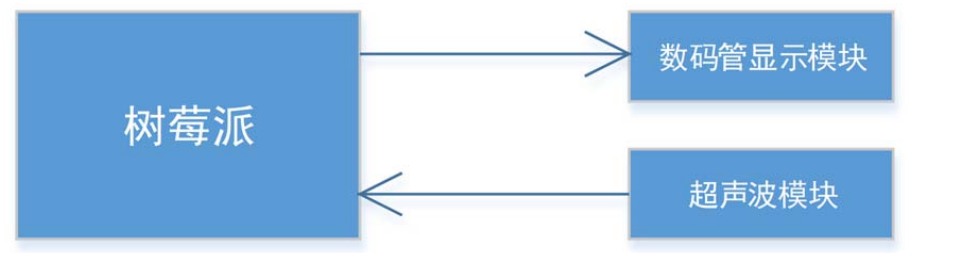


图1 系统框图

数码管显示模块，通过超声波模块测量距离并在数码管模块上显示距离。

1. **实验步骤**

1、 连接：用杜邦线，将各部分连接起来；

2、 编写程序驱动超声波模块测量距离并在数码管上显示出来。

3、 编写 TK 程序，能在显示屏上显示距离，要求基本界面美观。

1. **硬件连接图,软件流程图**

硬件连接图：

|  |  |
| --- | --- |
| 模块引脚 | 树莓派引脚(BCM) |
| 数码管 |  |
| SCLK | 13 |
| RCLK | 19 |
| DIO | 26 |
| 超声波 |  |
| TRIG | 2 |
| ECHO | 3 |

图1 硬件连接图

软件流程图：

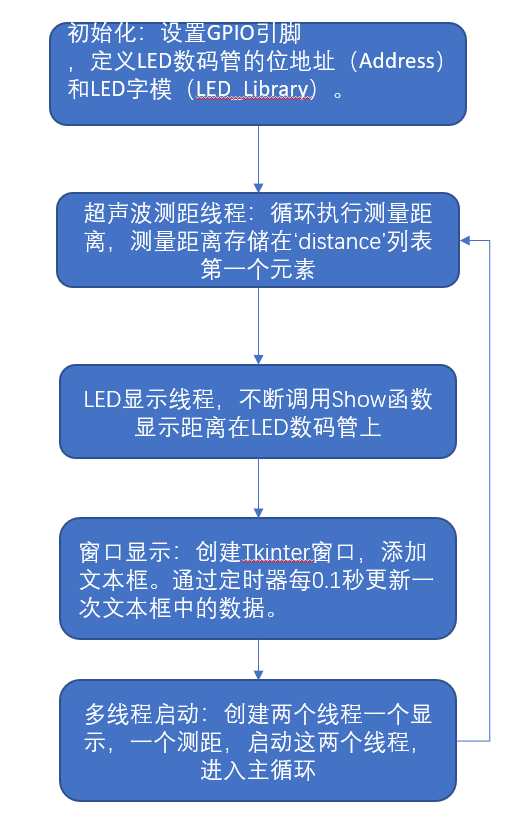


图2 软件流程图

1. **关键代码分析**

import time

import RPi.GPIO as GPIO

from threading import Thread

import ttkbootstrap as ttk

from ttkbootstrap.constants import \*

import tkinter as tk

# 超声波测距

def checkdist():

while True:

distance[0] = CheckDist()

time.sleep(0.5)

def CheckDist():

GPIO.output(TRIG, GPIO.HIGH) # 发出触发信号

time.sleep(0.000015) # 持续15微秒

GPIO.output(TRIG, GPIO.LOW)

while not GPIO.input(ECHO):

pass

t1 = time.time() # 获取当前时间戳（以秒为单位）

while GPIO.input(ECHO):

pass

t2 = time.time()

return (t2 - t1) \* 34000 / 2 # 返回距离，以厘米为单位

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

count = 0

while (i\_data > 10000): # 限制数据大小

i\_data -= 10000

list\_i\_data = list(str(i\_data)) # 将数据转化为列表变量

a = 0

for i in list\_i\_data:

if (i == "."):

count = a

break

a += 1

list\_i\_data.pop(count)

length = len(list\_i\_data)

for i in range(0, length):

Showdata[i] = int(list\_i\_data[i])

for i in range(0, 4):

LED\_Display(Showdata[i], Address[i], SCLK, RCLK, DIO, count, i)

def LED\_Display(i\_index, z\_location, SCLK, RCLK, DIO, index, i):

if ((index - 1) == i):

LED\_OUT(LED\_Library[i\_index] - 0x80, SCLK, DIO)

else:

LED\_OUT(LED\_Library[i\_index], SCLK, DIO) # 输出字模

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

GPIO.output(RCLK, GPIO.LOW)

GPIO.output(RCLK, GPIO.HIGH) # 向RCLK输出一个向上的脉冲信号

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

for i in range(0, 8):

if (x & 0x80):

GPIO.output(DIO, GPIO.HIGH)

else:

GPIO.output(DIO, GPIO.LOW)

GPIO.output(SCLK, GPIO.LOW)

GPIO.output(SCLK, GPIO.HIGH) # 向SCLK输出一个向上的脉冲信号，用于数据在移位寄存器中进行移位操作

x <<= 1

# LED数码管显示

def LED\_Show():

while (True):

Show(distance[0], SCLK0, RCLK0, DIO0)

# 窗口显示

def Interface\_Show():

top = ttk.Window(themename='superhero')

top.title('实验四超声波测距') # 设置界面标题

top.geometry('400x80+400+200') # 设置界面大小与位置

label1 = ttk.Label(top, text="distance：", bootstyle="warning")

text1 = ttk.Text(top, width=10, height=1)

label1.place(x=65, y=22)

text1.place(x=140, y=16)

def updatedata():

text1.delete(1.0, "end")

text1.insert("insert", distance)

top.after(100, updatedata) # 在给定时间后调用函数一次

top.update()

top.after(100, updatedata) # 0.1s更新一次数据

# 进入消息循环

top.mainloop()

# 定义GPIO引脚

SCLK0 = 13

RCLK0 = 19

DIO0 = 26

TRIG = 3

ECHO = 4

LED\_PINS = [SCLK0, RCLK0, DIO0]

Address = [0x08, 0x04, 0x02, 0x01] # 定义LED数码管位地址

distance = [0] # 距离

Showdata = [0]\*50

LED\_Library = [0xC0, 0xF9, 0xA4, 0xB0, 0x99, 0x92, 0x82, 0xF8, 0x80, 0x90, 0x8C, 0xBF, 0xC6, 0xA1, 0x86, 0x8E, 0xBF] # 定义字模

GPIO.setmode(GPIO.BCM) # 设置引脚编号方式为BCM

GPIO.setup(TRIG, GPIO.OUT, initial=GPIO.LOW)

# 定义GPIO引脚为输出

for i in LED\_PINS:

GPIO.setup(i, GPIO.OUT)

GPIO.setup(ECHO, GPIO.IN, GPIO.PUD\_DOWN) # 设置为下拉输入

thread1 = Thread(target=LED\_Show) # 创建新线程

thread2 = Thread(target=checkdist)

thread1.start() # 启动新线程

thread2.start()

Interface\_Show()

1. **实验结果**

超声波模块检测前方的物体距离，根据放松波段出发时间和回收时间差值，再将其乘以34000得到单位为cm的距离数，并将这个距离显示在与树莓派相连的数码管上。

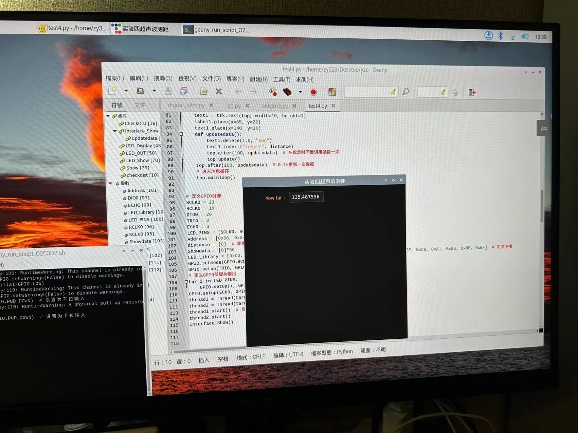
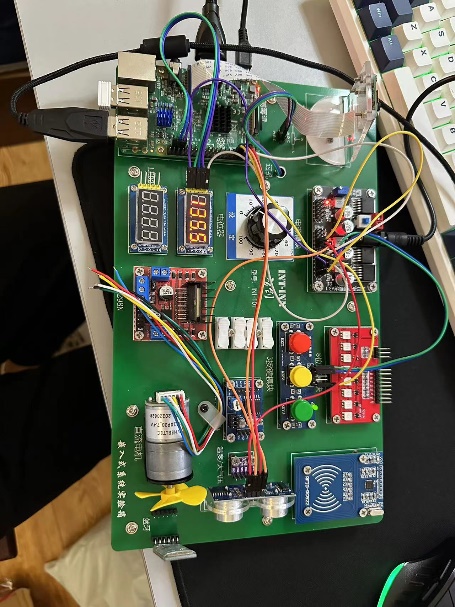
 

图3 实验结果

1. **实验遇到的问题及处理方法**

在实验时，超声波测距有巨大偏差，仔细排查后发现时周围存在可能干扰超声波信号的电磁干扰源，排除干扰源测量值变得准确。

1. **实验中，自己的创意部分**

显示的位数始终为4位，并且会根据显示的数字的整数部分的位数，自动显示小数点的位置。

1. **实验中的收获**

1、 学习了超声波测距原理

2、 了解了超声波测距模块的控制方式

3、 了解了 4 位 LED 数码管的显示控制方法

4、 通过树莓派编程，完成了超声波测距功能

5、 学会了编写 Tkinter 界面，将测距显示在界面中