微机原理 final Project

智能语音助手

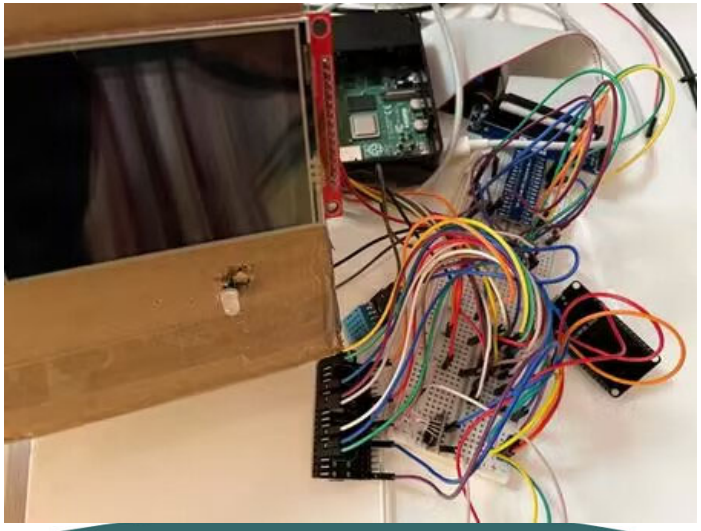
王烨祥 喻锦城

SUSTech Southern University of Science and Technology

主要硬件

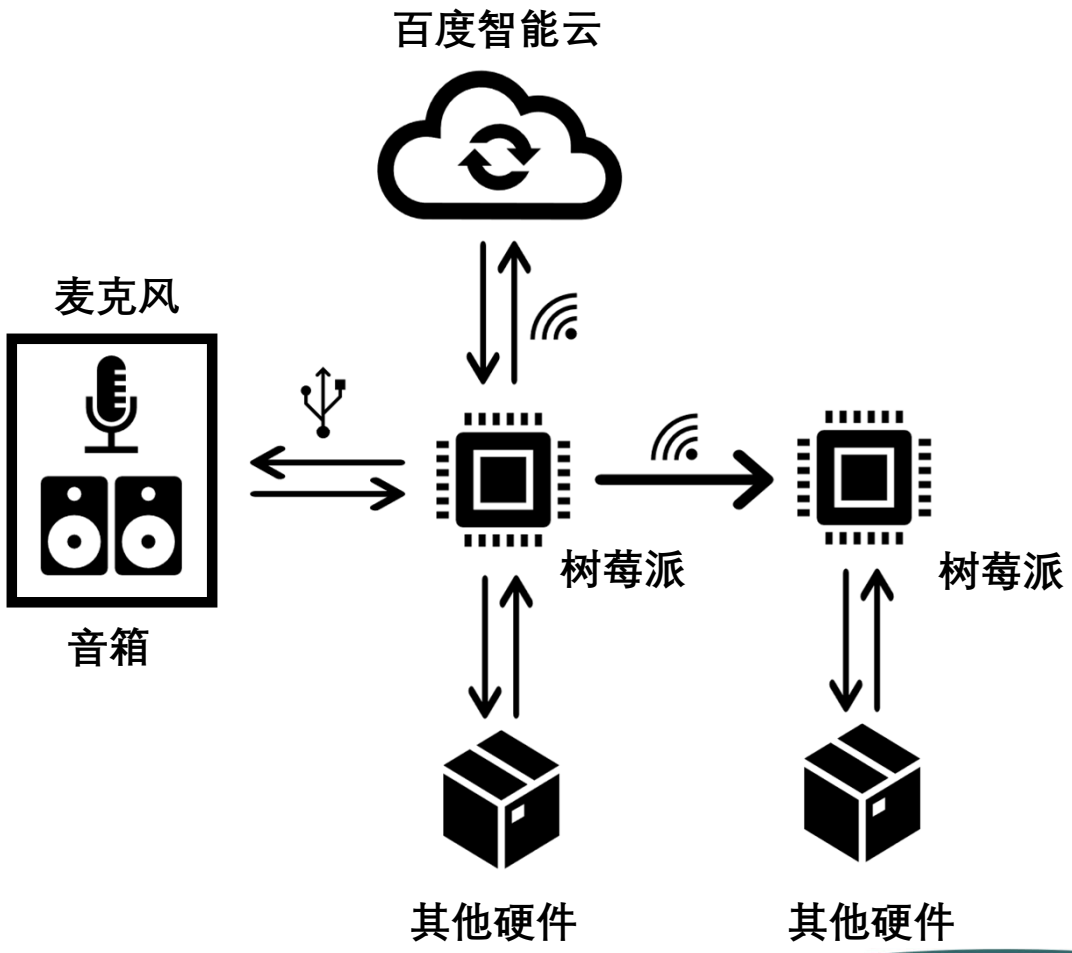
- 语音交互部分：麦克风(USB)，音箱(USB)，红绿LED灯(用于演示开关灯效果)
- 智能家居部分：第二部分详细说明

语音交互部分

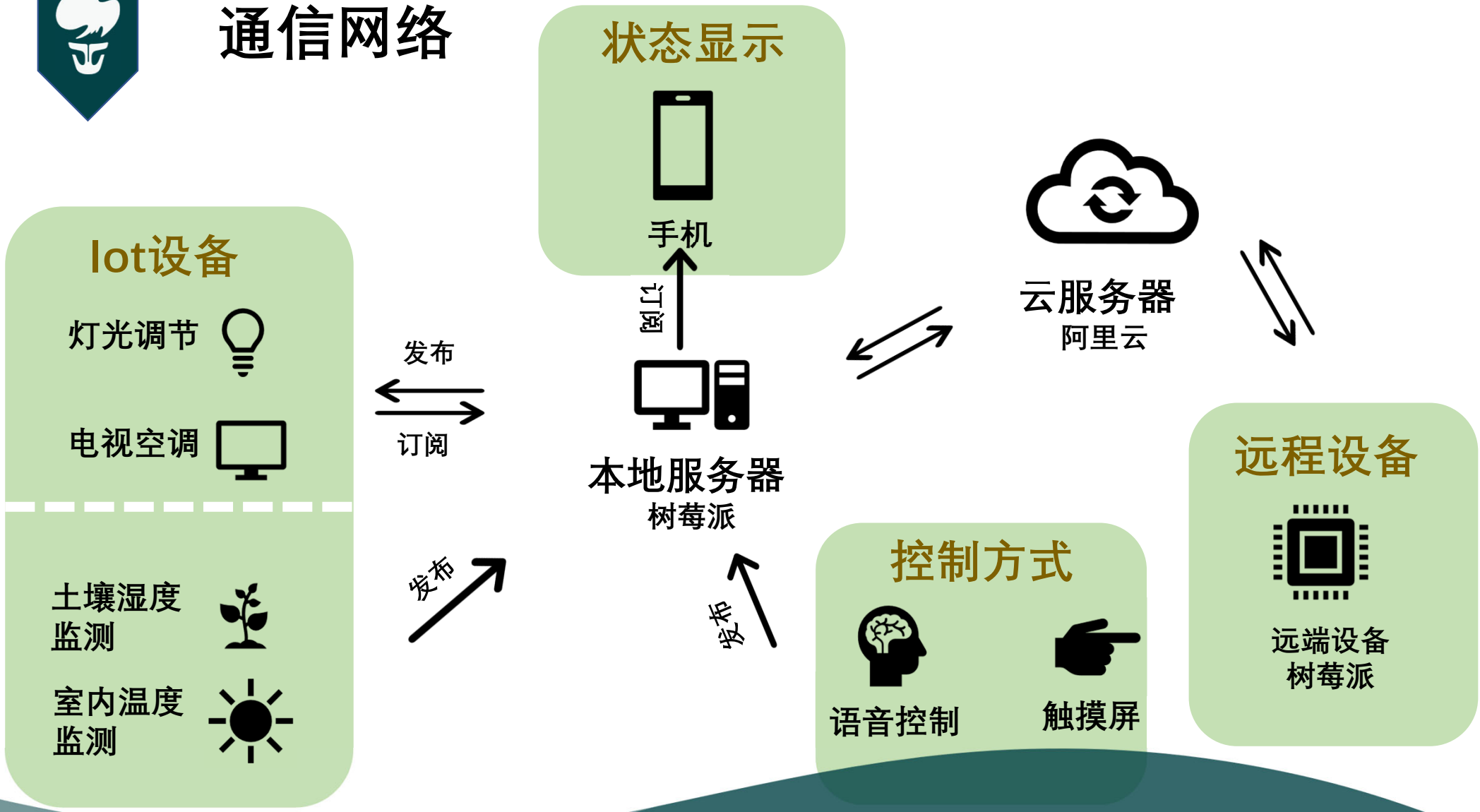


智能家居部分

系统框图



通信网络


状态显示
手机
订阅
lot设备
灯光调节
电视空调
发布
订阅
本地服务器
树莓派
云服务器
阿里云
远程设备
远端设备
树莓派
发布
发布
控制方式
语音控制
触摸屏
土壤湿度
监测
室内温度
监测

功能列表

- 1.唤醒和语音交互
- 2.声控开灯/关灯
- 3.查询天气
- 4.调用图灵机器人聊天
- 5.远程/本地服务器
- 6.红外控制设备
- 7.室内温湿度查询
- 8.移动端控制家居设备

PART 1 语音交互

声卡配置

- 在使用外接USB音频设备前，需要提前配置声卡。需进入树莓派的configuration中指定设备自带的声卡(不能使用树莓派自带的声卡)

```
pi@raspberrypi:~ $ arecord -l
**** List of CAPTURE Hardware Devices ****
card 3: Device [USB PnP Sound Device], device 0: USB Audio [USB Audio]
  Subdevices: 0/1
  Subdevice #0: subdevice #0
```

麦克风(录音设备)的配置

```
pi@raspberrypi:~ $ aplay -l
**** List of PLAYBACK Hardware Devices ****
card 0: b1 [bcm2835 HDMI 1], device 0: bcm2835 HDMI 1 [bcm2835 HDMI 1]
  Subdevices: 4/4
  Subdevice #0: subdevice #0
  Subdevice #1: subdevice #1
  Subdevice #2: subdevice #2
  Subdevice #3: subdevice #3
card 1: b2 [bcm2835 HDMI 2], device 0: bcm2835 HDMI 2 [bcm2835 HDMI 2]
  Subdevices: 2/2
  Subdevice #0: subdevice #0
  Subdevice #1: subdevice #1
card 2: Headphones [bcm2835 Headphones], device 0: bcm2835 Headphones [bcm2835 Headphones]
  Subdevices: 2/2
  Subdevice #0: subdevice #0
  Subdevice #1: subdevice #1
```

音箱(输出设备)的配置

录音参数

- 完成相关配置后，可以使用系统自带命令：
- arecord -d 3 -r 16000 -c 1 -f S16_LE listen_word.wav
- 来进行录音。-d 3 表示录音3s，-r 16000 表示采样率为16k，-c 1 表示通道数为1，-f S16_LE 表示以16bit格式进行录音。
- 在本次项目中，必须严格以16k采样率，1通道，16bit格式进行录音，否则将无法语音识别(百度智能云严格要求音频格式)。
- Ps:开始我认为使用32bit音质更好，会有更好的识别效果，多次实验之后发现无法识别。(百度智能云文档中并未指定16bit，这里调参尝试了好久)

语音唤醒 和 Snowboy

- 许多产品都用有语音唤醒功能，例如iPhone的siri，小米音箱的小爱同学等等。
- Snowboy，KITT.AI开发的人工智能软件工具包。通过Snowboy软件，开发人员可以在一些硬件设备上添加 “语音热词探测” 功能。
- Snowboy 可以用于实时嵌入式系统，并且始终监听（即使**离线**）。当前，它可以运在 **Raspberry Pi**、（Ubuntu）Linux 和 Mac OS X 系统上。
- 目前，Snowboy已经开源。本项目使用Snowboy作为语音唤醒框架。

Snowboy的回调函数(call_back)

- 执行各项功能的主函数main.py即写在Snowboy的回调函数中，每次检测到唤醒词，即执行主函数，完成各项功能，达到人机交互的效果。

```
# main loop
detector.start(detected_callback=snowboydecoder.play_audio_file,
               interrupt_check=interrupt_callback,
               sleep_time=0.03)
```

Snowboy框架的回调函数

```
stream_out.stop_stream()
stream_out.close()
audio.terminate()
os.system('python3 main.py')
```

回调函数内部执行主函数main.py

语音识别

- 语音识别技术是指机器自动将人的语音的内容转成文字，又称Automatic Speech Recognition，即**ASR技术**。
- 在深度学习出现之后，**RNNLM**语言模型开始广泛使用。
- 语音识别技术**早期的应用主要是语音听写**，d用户说一句，机器识别一句。后来发展成语音转写，随着AI的发展，语音识别开始作为**智能交互应用**中的一环。
- 本项目使用了百度智能云的语音识别API，来完成语音识别(ASR)功能。

语音识别的效果

```
开始录音:
Recording WAVE 'listen_word.wav' : Signed 16 bit Little Endian, Rate 16000 Hz, Mono
{'corpus_no': '7185721187959765753', 'err_msg': 'success.', 'err_no': 0, 'result': ['开下灯'], 'sn': '4165441444716730560 89'}
```

识别的效果

```
if(result['err_msg']=='success.'):
        # print('bbb')
        word=result['result']
        do_something(word[0], client_remote)
```

将语音识别内容
(字符串)提取出来

```
if ('开' in word) and ('灯' in word):
        # '好的，这就为您开灯。'
        os.system('mplayer ./resources/open_light.mp3')
        led('1')
        publish(client,1)
```

以开灯为例

语音合成

- 语音合成是通过机械的、电子的方法产生人造语音的技术。
- TTS技术（又称文语转换技术）隶属于语音合成，它是将计算机自己产生的、或外部输入的文字信息转变为可以听得懂的、流利的汉语口语输出的技术。
- 本项目使用了百度智能云的语音合成API，来完成语音合成(TTS)功能。具体效果见演示视频。

云端-边缘

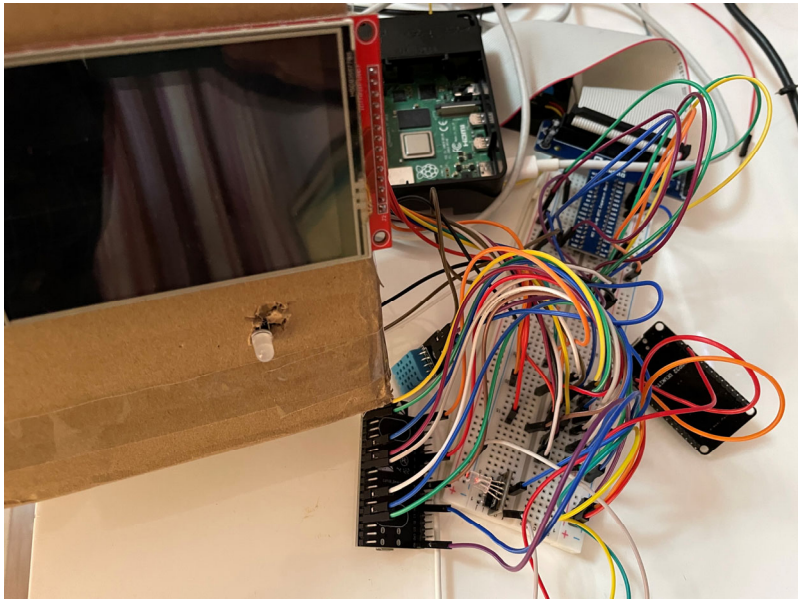
- 本次project中，树莓派的算力并不支撑完成语音识别的深度学习，所以我们采用了调用百度云的API，云端完成识别之后将，结果返回，由树莓派进行进一步的操作。
- 这比较好的体现了目前所提倡的边缘-云端的架构，突出了嵌入式轻量级的，灵活的优势。

第一部分功能演示(视频)

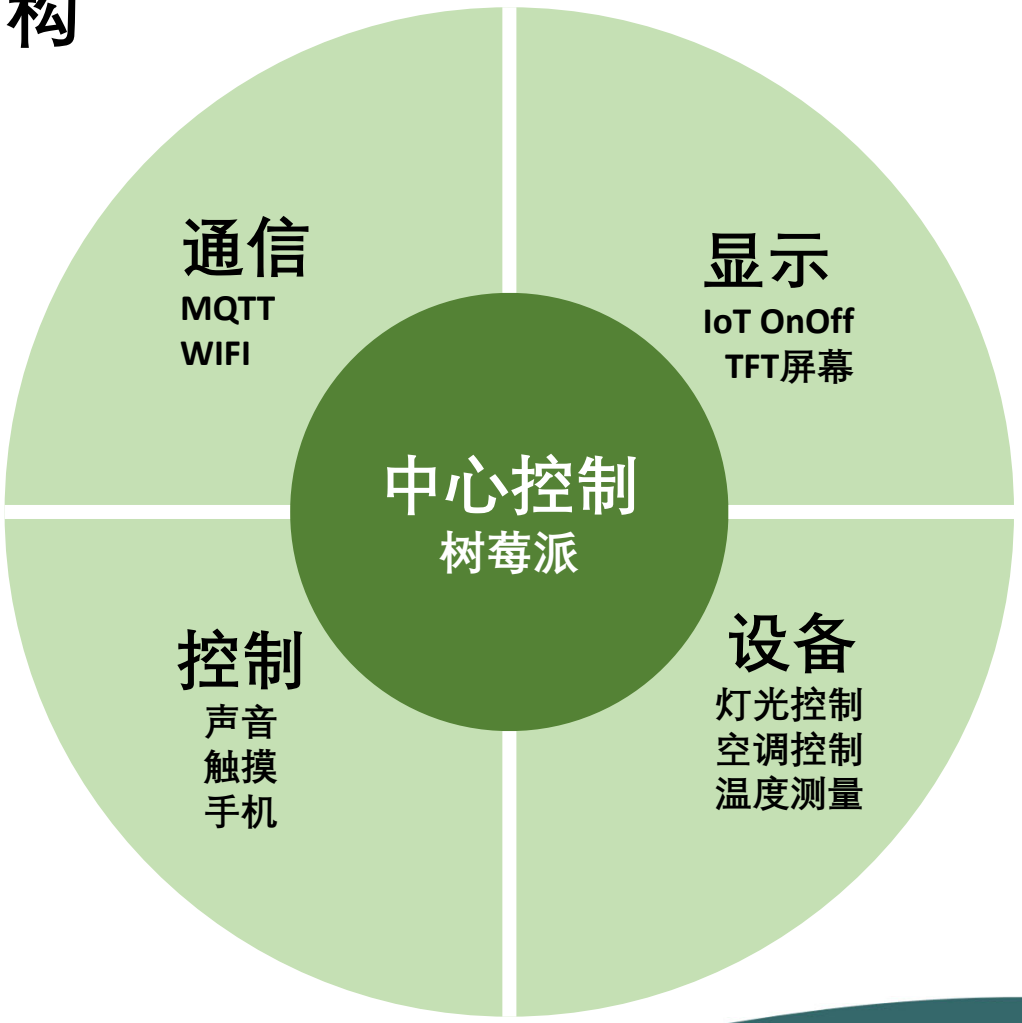
PART 2 智能家居

主要设备

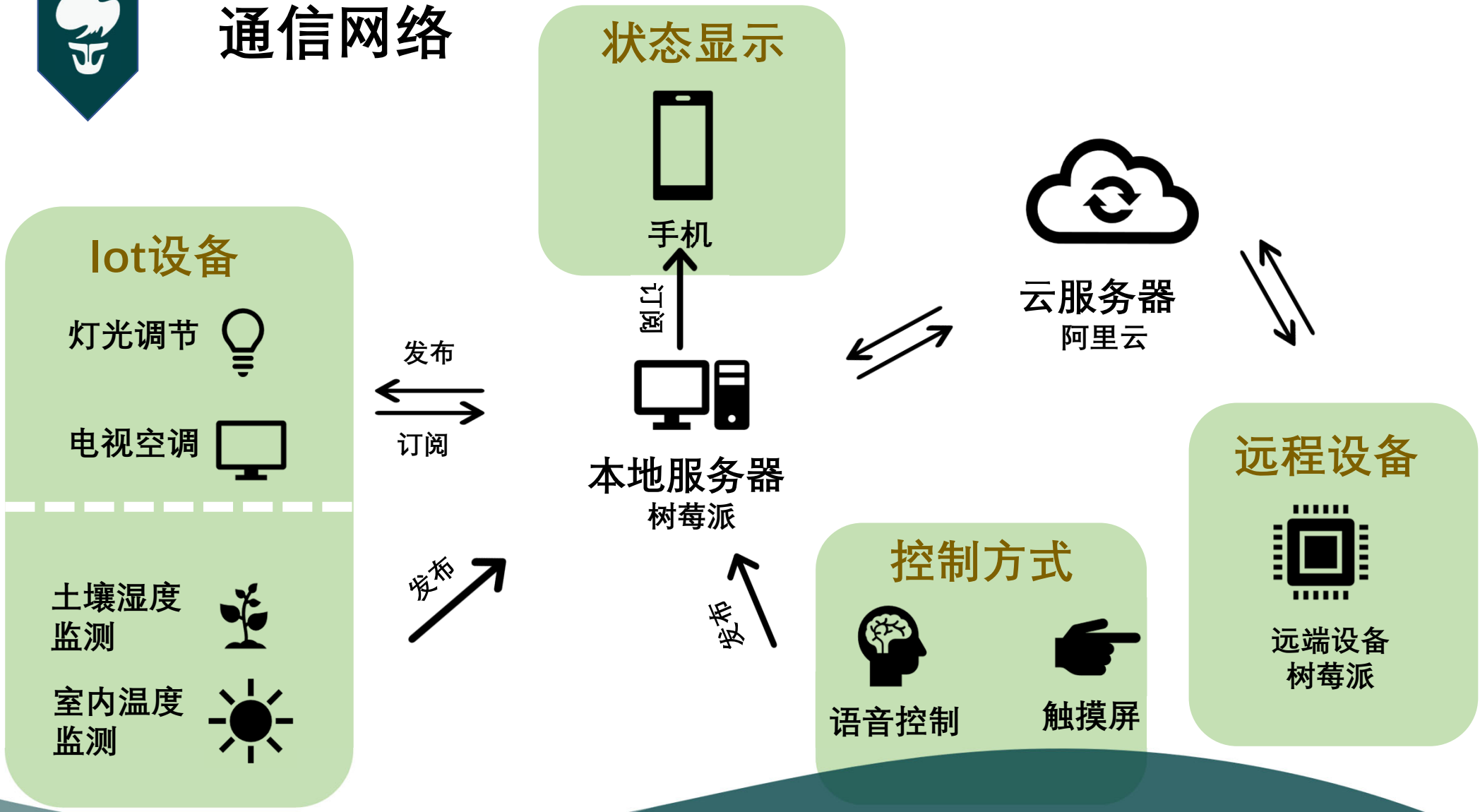
- 树莓派
- 大功率红外发射模块
- 红外接收模块
- 光耦继电器模块
- ESP32-DEVKITC-32D
- ILI9341TFT显示屏
- SG90舵机
- DHT11温湿度传感器
- LED灯



整体结构


通信
MQTT
WIFI
显示
IoT OnOff
TFT屏幕
中心控制
树莓派
控制
声音
触摸
手机
设备
灯光控制
空调控制
温度测量

通信网络


状态显示
手机
订阅
lot设备
灯光调节
电视空调
发布
订阅
本地服务器
树莓派
云服务器
阿里云
远程设备
远端设备
树莓派
发布
发布
控制方式
语音控制
触摸屏
土壤湿度
监测
室内温度
监测

通信网络

□ 通信协议:
MQTT 3.1.1

□ 服务器:
云服务器：
阿里云-----》EMQX
实现公网MQTT数据交换

本地服务器：
树莓派-----》mosquito
实现局域网的iot设备连接

□ 硬件设备：
ESP32，树莓派

物联网设备

ESP32控制灯光

组成设备：
ESP32：负责与本地服务器通信
并控制舵机
舵机：负责控制开关
5V锂电池：负责供电
三脚开关：手动控制供电

供电方案：
锂电池正负极连接VIN与GND引脚

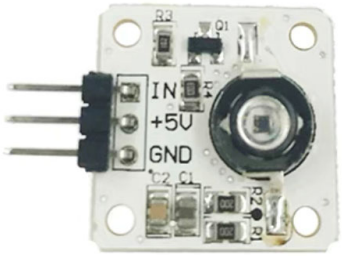
休眠功能：
长时间不使用时进入休眠模式，节约电量

物联网设备

红外控制器：

ESP32：负责与本地服务器通信并控制编解码
大功率红外发射模块：发射红外信号，5m范围都可接收

红外接收模块：负责解码



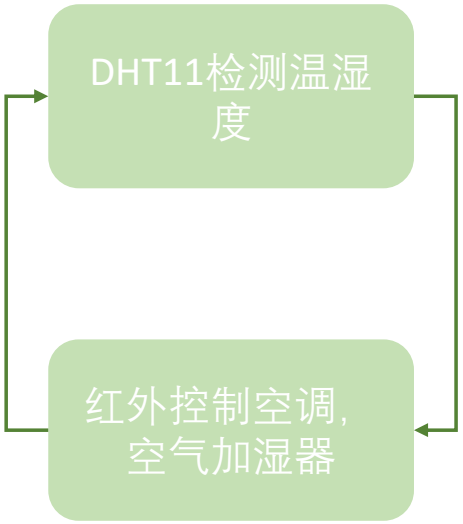
触摸控制器：

ILI9341TFT显示屏：显示并接收触摸信号

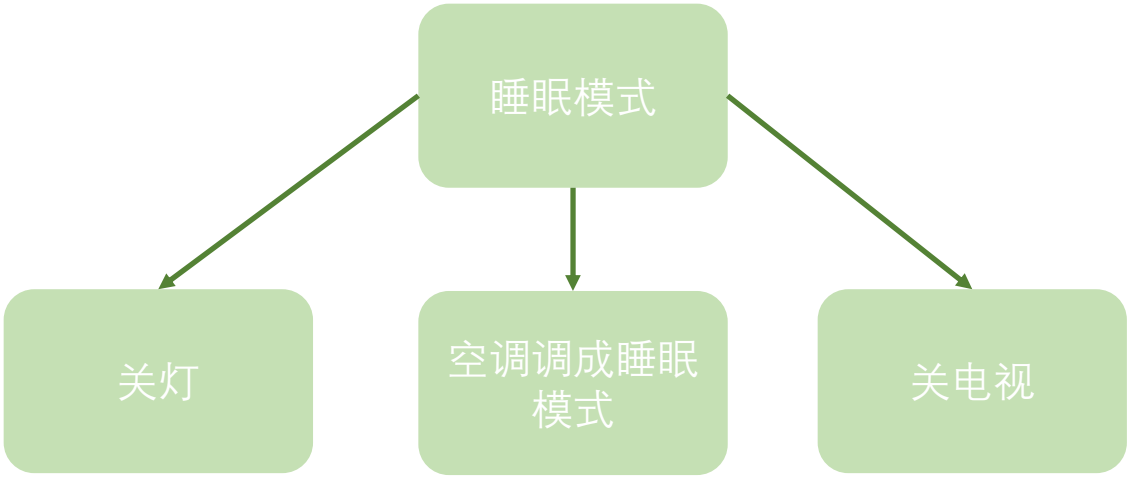
ESP32：负责与本地服务器通信并控制显示屏

物联网设备

反馈调节：



自定义协同模式：



控制方式

语音控制

通过唤醒树莓派利用部署好的的语音识别功能控制指令的发出

触摸控制

通过TFT触摸屏按下相应按键发出相应指令

手机控制

通过可视界面，完成化监视与控制一体化的功能

控制方式

语音控制

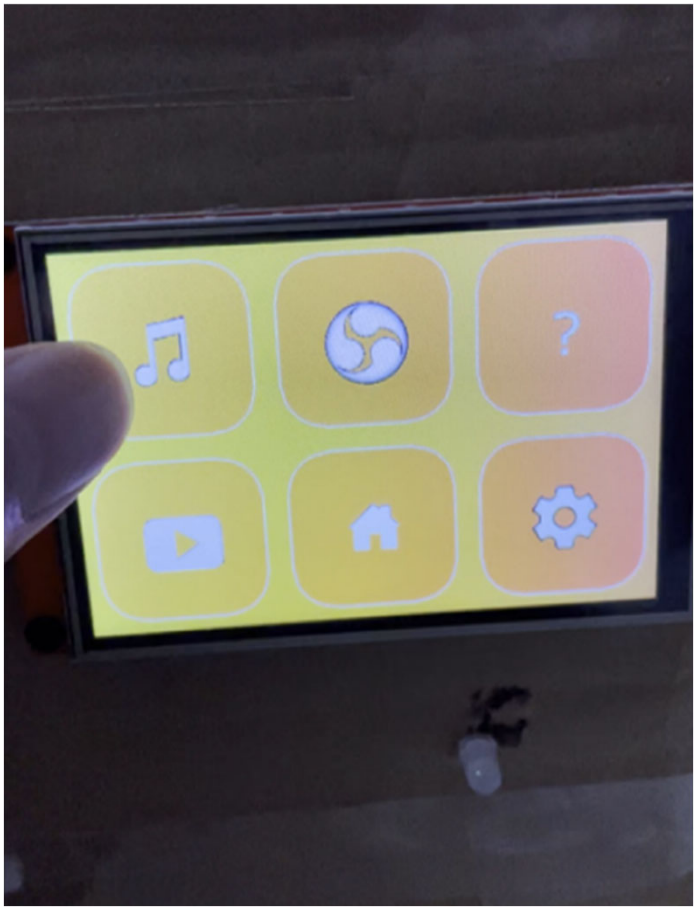
控制方式

触摸控制

主界面六个主题，每个主题下有五个功能按键，每个按键对应相应的指令

- 由ESP32实现触摸屏功能
- 可以与树莓派分离移动携带
- 单独的电源供电
- 30个按键指令

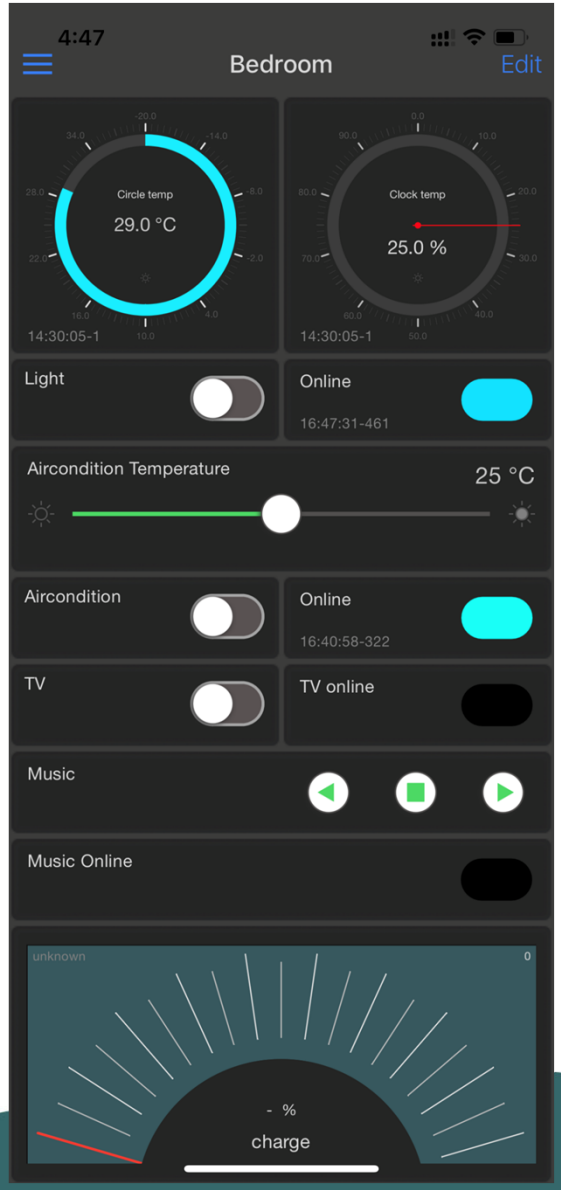
控制方式



显示界面

在手机端设计的可视化界面IoT OnOff

- 温度
- 湿度
- 灯光控制
- 温度控制
- 电视控制
- 音乐播放
- 设备在线情况

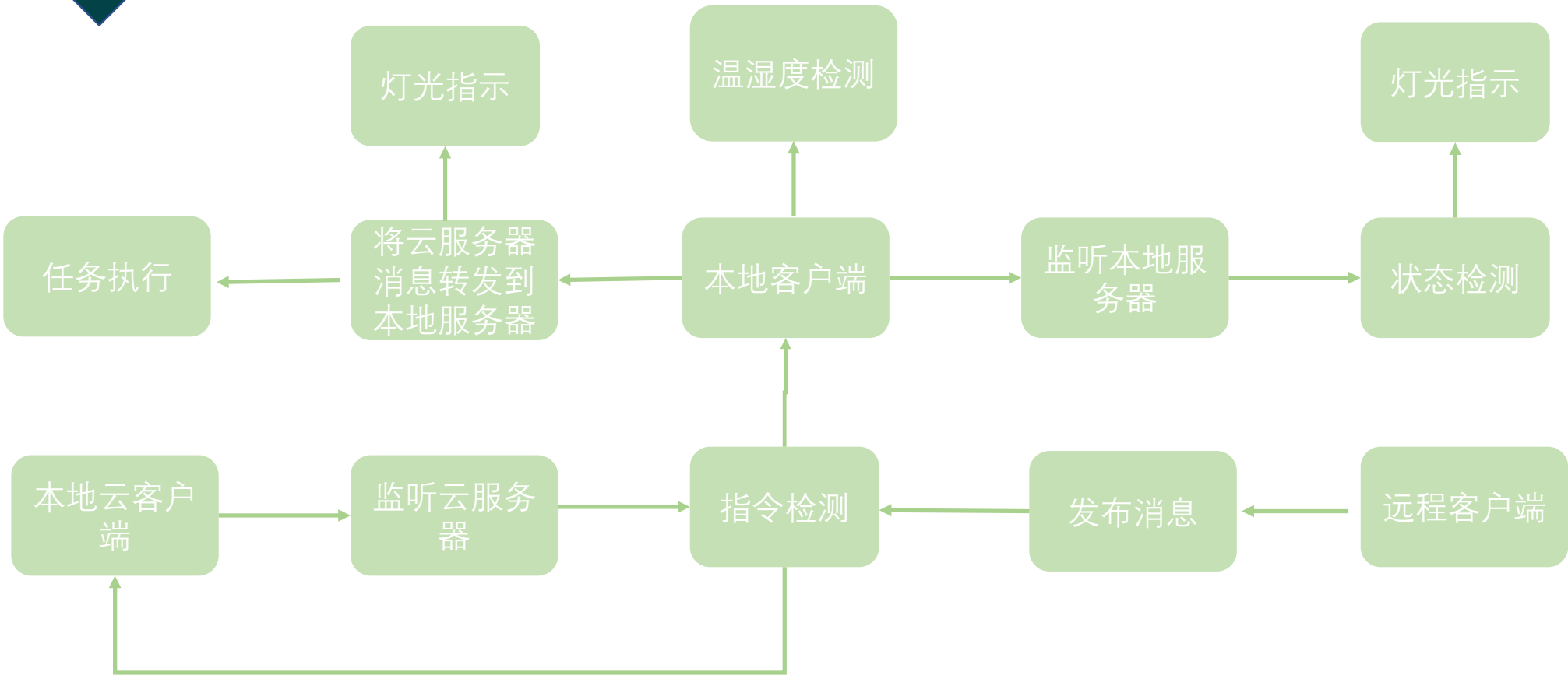


显示界面

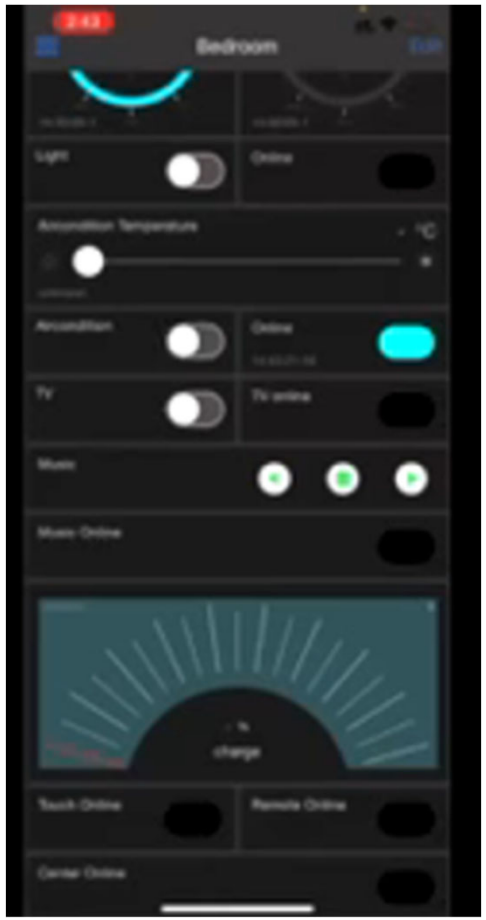
在手机端设计的可视化界面IoT OnOff

- 温度
- 湿度
- 灯光控制
- 温度控制
- 电视控制
- 音乐播放
- 设备在线情况

主程序逻辑


灯光指示
温湿度检测
灯光指示
任务执行
将云服务器消息转发到本地服务器
本地客户端
监听本地服务器
状态检测
本地云客户端
监听云服务器
指令检测
发布消息
远程客户端

远程控制


Bedroom
Light
Online
Air-conditioner Temperature
Air-conditioner
Online
TV
TV online
Music
Music Online
charge
Touch Online
Remote Online
Center Online

还可以优化的地方

- 1.目前录音功能为定时5s，之后优化可以不断检测音量，若音量变小了一段时间后，自动停止录音。
- 2.麦克风可以换成麦克风阵列，可以不必贴近麦克风录音，录音质量也会变好。
- 3.开关灯的功能由于硬件条件的限制，只能通过led来演示功能。之后加上相关的控制系统，完成物联。
- 4.由于物流原因，部分器件没有收到，搭建的模型较为简陋


Thanks For Watching!