声音信号的采集与播放

【必要硬件】

1. nodemcu-32s
2. 麦克风扩展板(https://lw.hachina.io/)
3. 耳机或耳机插口有源音箱

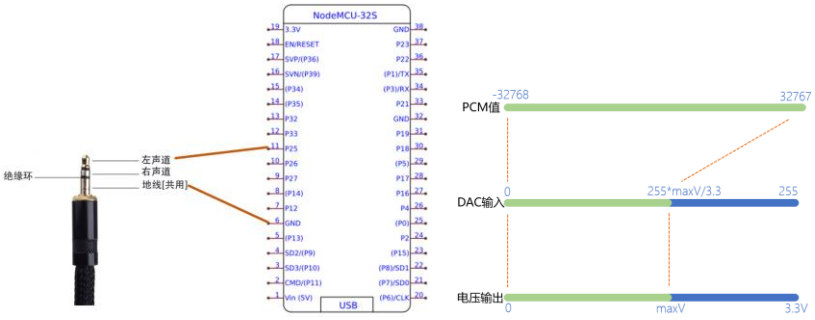

【操作步骤】

1. 在 Arduino 环境中观察声音输入与波形

Arduino 中麦克风信号采集程序

https://github.com/zhujisheng/Home-Assistant-DIY/blob/master/%E5%8F%82%E8%80%83%E6%96%87%E6%A1%A3%EF%BC%8821-30%EF%BC%89/example_24_audio_input/example_24_audio_input.ino

2. 将 PCM 信号转化为模拟信号输出给喇叭



Arduino 中 PCM 信号播放程序

https://github.com/zhujisheng/Home-Assistant-DIY/blob/master/%E5%8F%82%E8%80%83%E6%96%87%E6%A1%A3%EF%BC%8821-30%EF%BC%89/example_24_audio_output/example_24_audio_output.ino

【参考】

- 课程视频《Arduino 与 ESP 硬件》

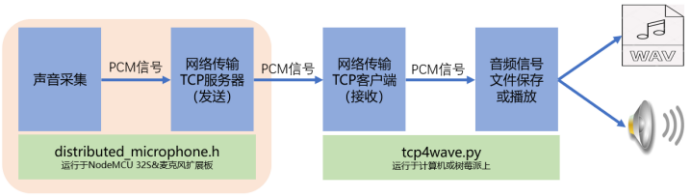
接入 HomeAssistant 的远程麦克风

【操作步骤】

1. 硬件连接

2. 在 ESPHome 中接入自定义的分布式麦克风程序
   a) 下载 distributed_microphone.h

https://raw.githubusercontent.com/zhujisheng/audio-reactive-led-strip/master/DistributedMicrophone/distributed_microphone.h



   b) 生成 ESPHome 的 yaml 配置文件
   c) 编译、UPLOAD
3. 使用 nc 进行连接测试
4. 接入 HomeAssistant 进行控制

【参考】

- ESPHome 操作视频
  - 《ESPHome——不编程，集成 ESP8266》
  - 《使用 NFC 识别不同的 ID 卡》
  - 《音乐灯带》
- 参考配置

```
esphome:
  name: distributed_microphone
  platform: ESP32
  board: nodemcu-32s
  includes:
    - distributed_microphone.h

......

switch:
  - platform: custom
    lambda: |-
      auto my_custom_switch = new MicrophoneSwitch();
      App.register_component(my_custom_switch);
      return {my_custom_switch};
    switches:
      name: "RemoteMic"

# a LED on GPIO17, optional for indicating microphone's state
output:
  - platform: ledc
    id: mic_led
    pin: GPIO17

light:
  - platform: monochromatic
    name: "MicLight"
```

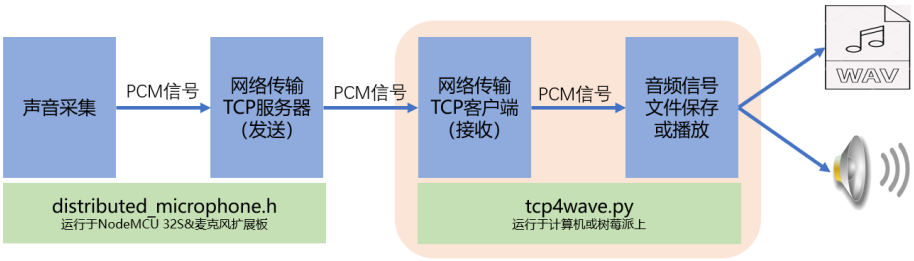
```
    output: mic_led
    default_transition_length: 0s
    effects:
      - strobe:
```

- ESPHome 中自定义开关组件

  https://esphome.io/components/switch/custom.html

使用远程麦克风——监听与录音

【架构】



【操作步骤】

1. 安装 pyaudio 库
   pip install pyaudio 或 sudo pip3 install pyaudio（linux 下 python3 环境）
2. 下载 tcp4wave.py 程序
   https://github.com/zhujisheng/audio-reactive-led-strip/blob/master/DistributedMicrophone/tcp4wave.py
3. 运行 tcp4wave.py
4. 解释 tcp4wave.py

【参考】

- pyaudio 库
  https://people.csail.mit.edu/hubert/pyaudio/
- wave 库
  https://docs.python.org/zh-cn/3/library/wave.html

给智能音箱配上远程麦克风

【操作步骤】

1. 以前智能音箱必要环节回顾
   1）本地麦克风
   2）SpeechRecongnition 库
   3）Snowboy 唤醒词服务
   4）具有 tts 和媒体播放能力的 HomeAssistant
   5）配置 HomeAssistant 的 conversation 和 intent_script 组件
   6）图灵机器人的 API 服务
2. 阅读新的 voice_assistant.py 程序

   https://github.com/zhujisheng/audio-reactive-led-strip/blob/master/DistributedMicrophone/%E6%99%BA%E8%83%BD%E9%9F%B3%E7%AE%B1/voice_assistant.py

   https://github.com/zhujisheng/audio-reactive-led-strip/blob/master/DistributedMicrophone/%E6%99%BA%E8%83%BD%E9%9F%B3%E7%AE%B1/tcp_mic.py
3. 程序下载与配置
4. 演示

【参考】

- REMOTE_MIC_CONFIG 配置说明
  - 当没有远程麦克风时，配置为：

    ```
    REMOTE_MIC_CONFIG = []
    ```
  - 当有多个远程麦克风时，配置为：

    ```
    REMOTE_MIC_CONFIG = [('IP1',port1, RemoteMicCB1), ('IP2',port2, RemoteMicCB2),…]
    ```

    其中，IP 和 port 是远程麦克风的 IP 地址与端口号，RemoteMicCB 是状态指示函数

    RemoteMicCB(0) 在进入等待唤醒时被调用

    RemoteMicCB(1) 在被唤醒后被调用

    RemoteMicCB(2) 在输入语音指令后被调用
- LOCAL_MIC_CONFIG 配置说明

  LOCAL_MIC_CONFIG=True 使用本地麦克风

  LOCAL_MIC_CONFIG=False 不使用本地麦克风