

浙江大学

本科实验报告

课程名称： 嵌入式系统原理与设计

姓 名： 王若鹏

学 院： 信息与电子工程学院

专 业： 电子科学与技术

学 号： 3170105582

指导教师： 杜 歆

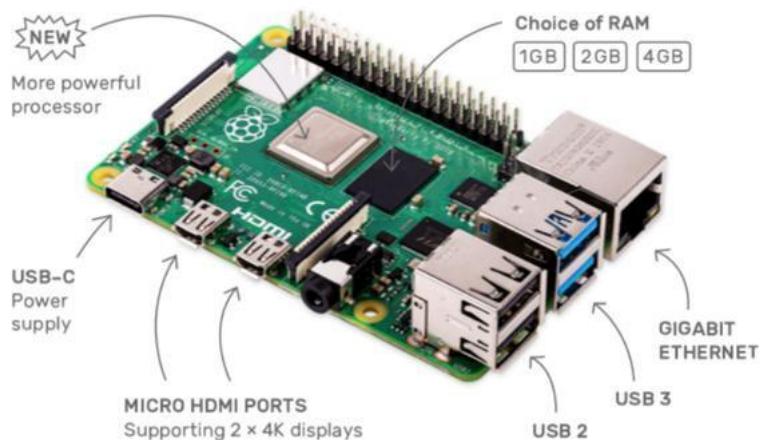
2020 年 3 月 26 日

基于树莓派的实时报告与监控系统

一、背景

树莓派 Raspberry PI 是一个采用 ARM 架构的开放式嵌入式系统，外形小巧，却具有强大的系统功能和接口资源。它是以 ARM 处理器为核心的单板计算机，拥有 256MB、512MB 甚至 1G 的内存，具有 USB 接口、快速以太网接口、SD 插槽、HDMI 输出接口。树莓派小巧玲珑，能提供 1080p 全高清影像输出。在搭载基于 ARM 的 Debian 和 Arch 等 Linux 的发行版后，便可使用大量现有的软件库，使用大量的开源软件，也便于实行开发扩展。树莓派支持多种语言进行应用开发，包括 C 语言和 Python 脚本等。

最新的树莓派 4B 搭载了博通最新的 BCM2711 处理器，它采用了 4 枚性能更为强劲的 ARM A72 核心。相比上一代树莓派 3B+，4B 的性能有着 2~4 倍的提升，处理任务要更为流畅。而在内存上，树莓派 4B 拥有 1GB/2GB/4GB 三种规格可选，相比以往的 1GB，大大拓展了多任务处理的可能。



(图 1：树莓派 4B)

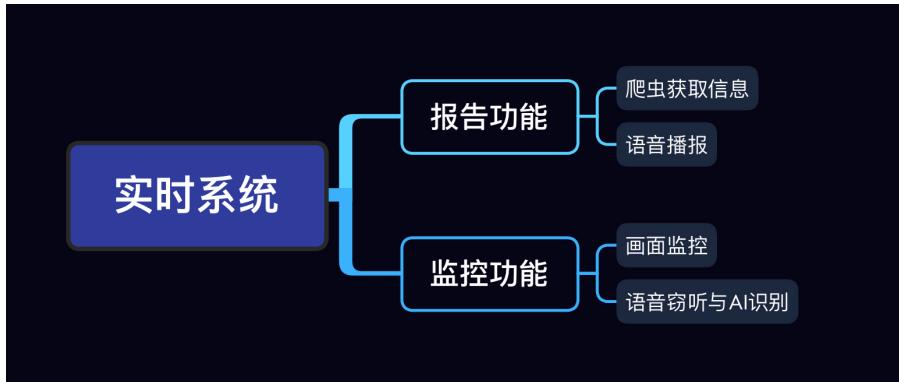
二、设计思路

在拿到树莓派之前，我就在思考，如何让这个“麻雀虽小五脏俱全”的微型计算机发挥出它应有的功能。虽然在网络上看到许多极客们用树莓派完成了大量令人叹为观止的作品，但在很长一段时间内我都没有什么合适的想法。想法不是说完全没有，只是因为难度或者设备的限制，一个接一个地被自己否定掉。

现在是一个网络互联的时代，获取实时信息是必不可少的一部分。因此，我萌生了将树莓派打造成获取实时信息的系统的想法。为充分发挥树莓派各个端口的功能，我将实时系统的设计分成两部分：实时报告系统和实时监控系统。实时报告系统通过采集疫情实时数据，

并用语音进行实时播报，来体现这一功能。实时监控系统通过树莓派连接摄像头与麦克风，实现对所处场景的实时监控与窃听。其中，我还将录音模块与百度 AI 的语音识别 API 相连接，从而实现语音转文字的功能。

以下是设计思路的逻辑框图：



(图 2：设计思路)

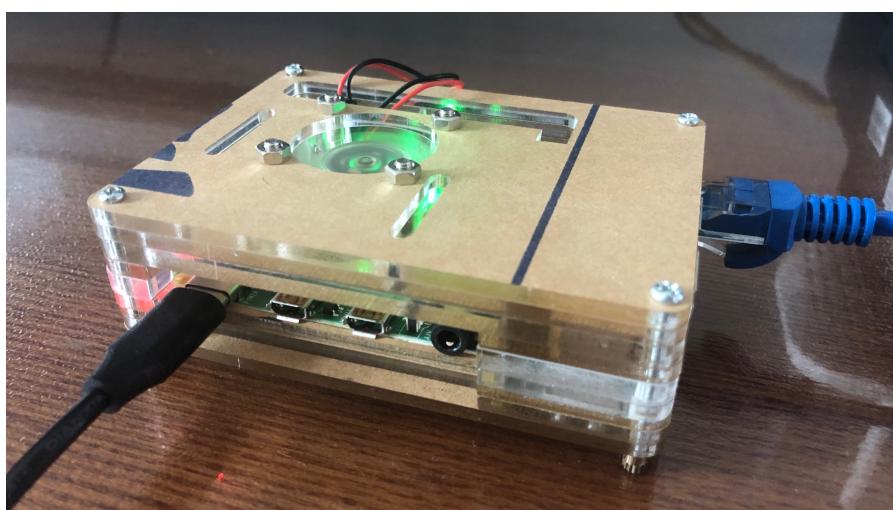
三、前期准备

3.1 实验器材

- 树莓派 4B（2GB 内存，32GB 存储）、装有 VNC viewer 的 MacBook Pro
- 电子配件：扬声器、麦克风、摄像头、网线、电源线
- 辅助组件：亚克力外壳一套、散热片若干、散热风扇一只
- 软件：2020 款 Raspbian 完整版(full)操作系统

3.2 硬件准备

考虑到可能的需求，我购买了 3 个散热片还有 1 个散热风扇。把散热片粘在芯片上，将散热风扇通过排针与树莓派相连，从而满足供电需求。最后参照说明书，把亚克力外壳组装好，为裸露的树莓派提供了一层很好的保护。如下图所示：

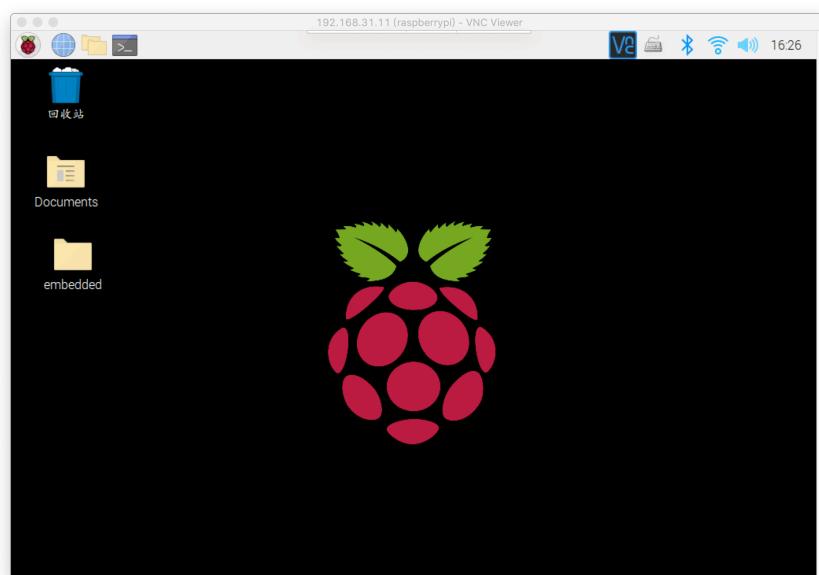


(图 3：组装好的树莓派)

3.3 软件准备

- (1) 下载所有的文件
- (2) 用 SDCardFormatter 格式化 SD 卡
- (3) 用 balenaEtcher 将解压 raspbian 固件 (IMG 文件) 烧写到 SD 卡
- (4) 在 SD 卡根目录 (boot 中) 新建名字为 SSH 文件 (无后缀)
- (5) 将 SD 卡插入到树莓派 4, 连上网线, 开机
- (6) 通过路由器配置网页找到树莓派 IP
- (7) 通过远程终端程序 VNC Viewer 登录到树莓派

最终效果如下图所示：



(图 4: 树莓派桌面)

四、实现过程

4.1 实时报告系统

现在是一个网络互联的时代，获取实时信息是其中必不可少的一部分。鉴于最近 COVID-19 是全球热点，之前我每天早上睁开眼第一件事就是打开手机看看疫情情况。我就想能不能也做一个类似于丁香园或者约翰霍普金斯大学的疫情实时数据获取系统。最终我设计了这样一套系统，它可以采集疫情实时数据，并进行语音播报。

4.1.1 疫情数据爬取

我查询了相关教程，首先我们需要获取指向数据的 url，并伪装成浏览器进行网络爬虫，主体程序为 PlagueData.py（详见附录）。以下是我选择爬取的网站：

https://view.inews.qq.com/g2/getOnsInfo?name=disease_h5

它是腾讯新闻的一个疫情数据统计网站，H5 样式的。可以从这些字符串中的花括号和双引号看到网站的数据组织方式，我的工作就是先分析这个数据结构，然后按照我想要的格式获取它们。以下截图由于截取时间不同，故数据不一致。



(图 5: 被爬取的网站, 分别用手机端和电脑端打开)

图 6 是我写的爬虫代码，核心思想就是伪装成客户端访问，然后把网页中 json 格式的字符串信息转换为字典存储，从而方便提取其中我们想获得的信息。url 全称叫 uniform resource locator，统一资源定位符，也就是网址。headers 代表着访问网址的客户端信息，里面有 linux 也有电脑端的 chrome 等等。接着就是先对网站进行 get 请求，在计算机网络的里，网站的请求方式主要有两种，分别是 GET、POST，由于 get 方式侧重于数据量小、速度快的访问，所以我选择这种方式。get 得到的数据默认是以 txt 文本文件的形式进行存储，于是为了后续处理方便，把它转换成字典。最后把字典中的 data 部分提取出来，就完成了。

```
import requests
import json
import os
import sys
import datetime

def Down_data():
    url = 'https://view.inews.qq.com/g2/getOnsInfo?name=disease_h5' #统一资源定位符(网址)
    headers = { #用户客户端
        'user-agent': 'Mozilla/5.0 (Linux; Android 6.0; Nexus 5 Build/MRA58N) \
                        AppleWebKit/537.36 (KHTML, like Gecko) Chrome/78.0.3904.108 Mobile Safari/537.36'
    }
    r = requests.get(url, headers) #有headers的get请求
    res = json.loads(r.text) #将json格式的数据(字符串)转换为字典
    data_res = json.loads(res['data']) #提取字典中的data部分
    return data_res
```

(图 6: 爬虫模块代码)

图 7 是爬取全国数据的模块，右上角的截图是网页中对全国数据进行描述的部分。我先分析了这些字符串的逻辑结构，然后根据这个结构，生成字典，然后从字典中提取所需的数据。数据的第一维读有两种，累计数据的关键字为 `chinaTotal`，新增数据为 `chinaAdd`。第二维度有很多，比如确诊 `confirm`、死亡 `death`、治愈 `heal` 等等。根据这两个维度来获取数据，我使用列表来存储，合并为字符串后，生成一个 `txt` 文件，文件命名由爬取时间决定。

```
def Parse_data1():
    data = Down_data() #爬取数据以字典形式存储
    list = [ datetime.datetime.now().strftime('截止%Y年%月%d日%H时') + ','
            '全国累计确诊: ' + str(data['chinaTotal']['confirm']) + '例,' #全国累计确诊
            '新增确诊: ' + str(data['chinaAdd']['confirm']) + '例,' #新增确诊
            '累计境外输入: ' + str(data['chinaTotal']['importedCase']) + '例,' #累计境外输入
            '新增境外输入: ' + str(data['chinaAdd']['importedCase']) + '例,' #新增境外输入
            '累计治愈: ' + str(data['chinaTotal']['heal']) + '例,' #累计治愈
            '新增治愈: ' + str(data['chinaAdd']['heal']) + '例,' #新增治愈
            '累计死亡: ' + str(data['chinaTotal']['dead']) + '例,' #累计死亡
            '新增死亡: ' + str(data['chinaAdd']['dead']) + '例.' #新增死亡
    result = ''.join(list) #字符串列表合并为字符串
    with open('全国实时数据'+str(data['lastUpdateTime'])+'.txt', 'a+', encoding="utf-8") as f:
        f.write(result + '\n') #写入txt文件
    f = open('全国实时数据'+str(data['lastUpdateTime'])+'.txt','r') #打开txt文件
    text = f.read() #读取txt文件
    cmd = "ilang " + text #朗读命令
    os.system(cmd)
```

(图 7: 全国数据爬虫代码)

接下来是各个省份的数据，首先是要输入查询的省份，然后在字典中搜索，与全国数据的获取方法类似，最后存入 txt。代码如图 8 所示：

```

\\"name\\":\\"浙江\\",\\"today\\":{\\\"confirm\\":0,\\\"confirmCuts\\":0,\\\"isUpdated\\":true,\\\"tip\\":\\"浙江累计报告境外输入确诊病例49例。\\"},\\\"total\\":
\\\"nowConfirm\\":30,\\\"confirm\\":1267,\\\"suspect\\":0,\\\"dead\\":1,\\\"deadRate\\":\\\"0.08\\",\\\"showRate\\":false,\\\"heal\\":1236,\\\"healRate\\":\\\"97.55\\",\\\"showHeal\\":true},\\"
children": [{"\\name\\":\\"境外输入\\", \\\"today\\":{\\\"confirm\\":0,\\\"confirmCuts\\":0,\\\"isUpdated\\":true},\\\"total\\":
\\\"nowConfirm\\":29,\\\"confirm\\":149,\\\"suspect\\":0,\\\"dead\\":0,\\\"deadRate\\":\\\"0.00\\",\\\"showRate\\":false,\\\"heal\\":20,\\\"healRate\\":\\\"40.82\\",\\\"showHeal\\":true}},\\
{\\name\\":\\"嘉兴\\", \\\"today\\":{\\\"confirm\\":0,\\\"confirmCuts\\":0,\\\"isUpdated\\":true},\\\"total\\":
\\\"nowConfirm\\":29,\\\"confirm\\":146,\\\"suspect\\":0,\\\"dead\\":0,\\\"deadRate\\":\\\"0.00\\",\\\"showRate\\":false,\\\"heal\\":45,\\\"healRate\\":\\\"97.83\\",\\\"showHeal\\":true}},\\
{\\name\\":\\"金华\\", \\\"today\\":{\\\"confirm\\":0,\\\"confirmCuts\\":0,\\\"isUpdated\\":true},\\\"total\\":
\\\"nowConfirm\\":29,\\\"confirm\\":55,\\\"suspect\\":0,\\\"dead\\":0,\\\"deadRate\\":\\\"0.00\\",\\\"showRate\\":false,\\\"heal\\":55,\\\"healRate\\":\\\"100.00\\",\\\"showHeal\\":true}},\\
{\\name\\":\\"台州\\", \\\"today\\":{\\\"confirm\\":0,\\\"confirmCuts\\":0,\\\"isUpdated\\":true},\\\"total\\":
\\\"nowConfirm\\":29,\\\"confirm\\":146,\\\"suspect\\":0,\\\"dead\\":0,\\\"deadRate\\":\\\"0.00\\",\\\"showRate\\":false,\\\"heal\\":146,\\\"healRate\\":\\\"100.00\\",\\\"showHeal\\":true}},\\
{\\name\\":\\"舟山\\", \\\"today\\":{\\\"confirm\\":0,\\\"confirmCuts\\":0,\\\"isUpdated\\":true},\\\"total\\":
\\\"nowConfirm\\":10,\\\"confirm\\":146,\\\"suspect\\":0,\\\"dead\\":0,\\\"deadRate\\":\\\"0.00\\",\\\"showRate\\":false,\\\"heal\\":146,\\\"healRate\\":\\\"100.00\\",\\\"showHeal\\":true}},\\
{\\name\\":\\"衢州\\", \\\"today\\":{\\\"confirm\\":0,\\\"confirmCuts\\":0,\\\"isUpdated\\":true},\\\"total\\":
\\\"nowConfirm\\":10,\\\"confirm\\":14,\\\"suspect\\":0,\\\"dead\\":0,\\\"deadRate\\":\\\"0.00\\",\\\"showRate\\":false,\\\"heal\\":14,\\\"healRate\\":\\\"100.00\\",\\\"showHeal\\":true}},\\
{\\name\\":\\"丽水\\", \\\"today\\":{\\\"confirm\\":0,\\\"confirmCuts\\":0,\\\"isUpdated\\":true},\\\"total\\":
\\\"nowConfirm\\":10,\\\"confirm\\":14,\\\"suspect\\":0,\\\"dead\\":0,\\\"deadRate\\":\\\"0.00\\",\\\"showRate\\":false,\\\"heal\\":14,\\\"healRate\\":\\\"100.00\\",\\\"showHeal\\":true}},\\
{\\name\\":\\"绍兴\\", \\\"today\\":{\\\"confirm\\":0,\\\"confirmCuts\\":0,\\\"isUpdated\\":true},\\\"total\\":
\\\"nowConfirm\\":10,\\\"confirm\\":42,\\\"suspect\\":0,\\\"dead\\":0,\\\"deadRate\\":\\\"0.00\\",\\\"showRate\\":false,\\\"heal\\":42,\\\"healRate\\":\\\"100.00\\",\\\"showHeal\\":true}},\\
{\\name\\":\\"省十里丰监狱\\", \\\"today\\":{\\\"confirm\\":0,\\\"confirmCuts\\":0,\\\"isUpdated\\":true},\\\"total\\":
\\\"nowConfirm\\":10,\\\"confirm\\":36,\\\"suspect\\":0,\\\"dead\\":0,\\\"deadRate\\":\\\"0.00\\",\\\"showRate\\":false,\\\"heal\\":36,\\\"healRate\\":\\\"100.00\\",\\\"showHeal\\":true}},\\
{\\name\\":\\"宁波\\", \\\"today\\":{\\\"confirm\\":0,\\\"confirmCuts\\":0,\\\"isUpdated\\":true},\\\"total\\":
\\\"nowConfirm\\":10,\\\"confirm\\":157,\\\"suspect\\":0,\\\"dead\\":0,\\\"deadRate\\":\\\"0.00\\",\\\"showRate\\":false,\\\"heal\\":157,\\\"healRate\\":\\\"100.00\\",\\\"showHeal\\":true}},\\
{\\name\\":\\"杭州\\", \\\"today\\":{\\\"confirm\\":0,\\\"confirmCuts\\":0,\\\"isUpdated\\":true},\\\"total\\":
\\\"nowConfirm\\":10,\\\"confirm\\":181,\\\"suspect\\":0,\\\"dead\\":0,\\\"deadRate\\":\\\"0.00\\",\\\"showRate\\":false,\\\"heal\\":181,\\\"healRate\\":\\\"100.00\\",\\\"showHeal\\":true}},\\
\\\"nowConfirm\\":0
def Parse_data2():
    data = Down_data()['areaTree'][0]['children'] #载入爬到的数据children子树
    path = str(input('请输入你要查询的省份：')) #查询路径
    for i in data:
        if path in i['name']:
            for item in i['children']:
                list_city = [
                    '地区：' + str(item['name']),
                    '累计确诊：' + str(item['total']['confirm']),
                    '\t新增确诊：' + str(item['today']['confirm']) + '\n'
                ] #字符串列表记录查得数据
                res_city = ''.join(list_city) #字符串列表合并为字符串
                with open(path+'\\省数据'+str(datetime.datetime.now().strftime('%m月%d日'))+'.txt', 'a+', encoding='utf-8') as f:
                    f.write(res_city) #写入txt文件

```

(图 8：各省数据爬虫代码)

具体过程详见演示视频。操作步骤为第一步：运行，获取全国数据。第二步：输入“浙江”，查询浙江省的当日数据。得到的结果存储在两个 txt 文件中，文件命名根据查询时间而定，如下所示：



(图 9：爬取的全国和浙江省疫情实时数据)

4.1.2 语音播报

为使语音播报的内容简短，仅朗读全国疫情数据，作为示例。通过 ilang 命令实现（具

体效果请观看演示视频），在 Python 中的代码如下：即先读取 txt 文档，用 text 记录其中的文字信息，然后调用 ilang 命令实现朗读。ilang 命令属于 shell 命令，格式按照“ilang+空格+文本”，即可朗读文本内容。需要特别注意：txt 文件中不能有换行符，否则会报错。

以下为语言播报模块的关键代码：

```
f = open('全国实时数据'+str(data['lastUpdateTime'])+'.txt', "r")
text = f.read()
cmd = "ilang " + text
os.system(cmd)
```

4.2 实时监控系统

对于一个实时监控系统，我们需要实时获取设备所在场景的信息，其中最重要的就是视频和音频信息。但由于预算有限，我购买的摄像头和麦克风不是性能最佳的，但已经能足够体现该系统的设计理念。因此，实时监控系统主要分为两个部分：画面监控系统和语音窃听系统。其中语音窃听系统与百度 AI 语音识别相结合，可以实现实时地语音转文字，为窃听提供了极大的方便。

4.2.1 画面监控系统



(图 10: USB 摄像头)

首先我们需要将摄像头通过 USB 接口与树莓派连接，从下面的命令，能看出设备中是否有 video 的设备了，能说明 USB 摄像头是否成功被树莓派检测到了：

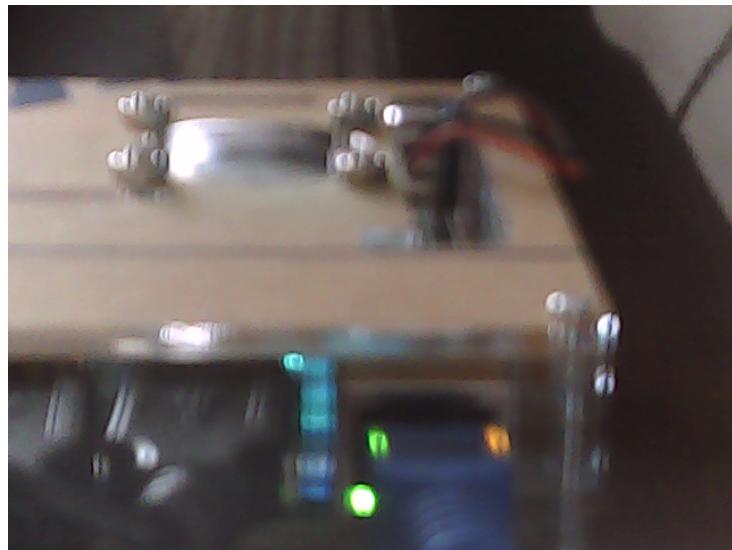
```
$ ls -l /dev/video*
```

当 USB 摄像头成功挂载到树莓派上之后，下一步就是拍一些照片来验证它的功能了。要想拍摄照片，需要安装 fswebcam，这是一款小型摄像头程序。fswebcam 安装完成后，在终端中运行下面的命令来抓去一张来自摄像头的照片。

```
$ sudo apt-get install fswebcam
$ fswebcam --no-banner -r 640x480 image.jpg
```

为方便起见，将上述指令编写进 python 脚本 camera.py（详见附录），运行即可拍照，

抓取一张 640x480 分辨率的照片，并且用 jpg 格式保存，将图片命名为 image.jpg，效果如下图所示。由于预算有限，仅购买了一只廉价的摄像头，分辨率低且不可调焦，但是它带有一个夹子，能够夹在任何地方、传输线可以扯很长，可以充分体现了监控的“隐蔽性”。



(图 11：拍摄的照片)

4.2.2 语音窃听系统



(图 12：USB 麦克风)

对于录音模块，我买的 USB 麦克风是免安装驱动的，但是在使用之前需要在 shell 里调试一下灵敏度，测测录音效果。配置好后就可以录音了。

4.2.2.1 录音模块

首先将麦克风通过 USB 与树莓派相连。在终端执行：

```
$ sudo arecord -D "plughw:1,0" -d 10 test.wav
```

录音 10 秒，之后生成一个文件 test.wav。这个音频文件需要用到播放器才能播放，所以安装 omxplayer，安装完以后就可以播放录音文件了。

```
$ sudo apt-get install omxplayer
```

```
$ omxplayer -o local test.wav
```

就能播放本文件夹下的音频文件 test.wav。如果录音效果不佳，噪音特别大，因为还需要再次配置一下麦克风，在终端输入命令进行配置：\$ alsamixer

为方便起见，将上述指令编写进 python 脚本 microphone.py（详见附录），运行即可录音并保存。

4.2.2.2 语音识别模块

首先进入百度 AI 管理平台：<https://console.bce.baidu.com/ai>

在产品服务-语音技术创建语音识别应用 raspberry-wrp，获得 AppID, APIKey, Secret Key 信息，这些信息在调用 API 接口时候需要用到，并下载 python 资源库 aip，如下图所示：

应用名称	AppID	API Key	Secret Key	创建时间	操作
1 raspberry-wrp	19163157	EOh0214yMBDs3Yjuz5feRY4n	q29Ql47tYwWZG5xn5YFndLTRNImBRrzn	2020-03-30 14:49:50	报表 管理 删除

(图 13：百度 AI 语音识别设置页面)

编写语音转文字程序 wav2text.py（详见附录），调用百度 AI 语音识别的 API 接口，

记录获得的 AppID, API Key, Secret Key，通过以下代码记录客户信息：

```
client = AipSpeech(APP_ID, API_KEY, SECRET_KEY)
```

如下图所示，被转换的录音内容为我录制的一段 10 秒钟的音频“这里是树莓派第一次录音测试”。在点击运行后的几秒，成功输出相应的文字内容。

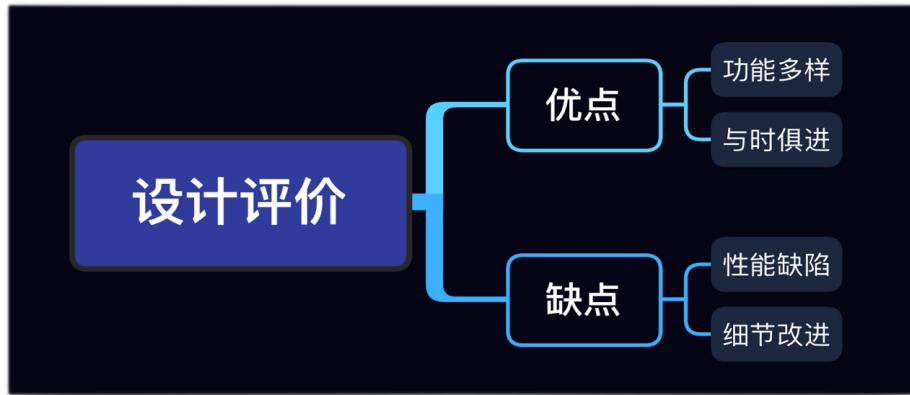
```
#!/usr/bin/python
# -*- coding: UTF-8 -*-
from aip import AipSpeech
APP_ID = '19163157'
API_KEY = 'EOh0214yMBDs3Yjuz5feRY4n'
SECRET_KEY = 'q29Ql47tYwWZG5xn5YFndLTRNImBRrzn'
client = AipSpeech(APP_ID, API_KEY, SECRET_KEY)
# 读取文件
def get_file_content(filePath):
    with open(filePath, 'rb') as fn:
```

Shell

```
Python 3.7.3 (/usr/bin/python3)
>>> %Run wav2text.py
这里是树莓派第一次录音测试。
>>>
```

(图 14：成功实现语音转文字)

五、评价



(图 15：设计评价)

5.1 优点

(1) 功能多样。虽然本系统名为基于树莓派的实时监测系统，但其功能涵盖了“线上”和“线下”两个方面，具备多样的功能，并且易于拓展。对于线上信息的监测，实现手段就是通过 python 进行爬虫。对于线下信息的监测，就需要用到许多设备，比如摄像头和麦克风。由于树莓派的可扩展性，还可以通过其他接口添加更多的设备。

(2) 与时俱进。近几年，语音识别是人工智能领域的一个火热的分支。因此，在设计语音监听系统时，我与时俱进，融入了语音识别模块，在获得语音信息的同时能够实时转换成文字信息，使得系统的功能更加完善、先进。

5.2 缺点

(1) 性能缺陷。由于预算有限，我购买的摄像头和麦克风的性能不是特别好。在拍照的时候，镜头没有自动对焦功能，导致拍出来的照片非常模糊。麦克风是单声道录音，生成的是 wav 文件而不是 m4a 文件，因而无法获取声音传来的位置这一信息。

(2) 细节改进。针对某些功能，我有更好的想法，但由于未知的原因无法实现。比如我最开始想做的是可以通过 chrome 浏览器远程访问的视频监控系统，但不知是我的 chrome 版本问题还是摄像头是否支持视频拍摄的问题，这项功能我做了很久，最终没有实现，可以说非常遗憾了。

六、心得体会

这是我完成的第一份嵌入式项目，从产生思路到动手把它实现处理，整个过程学到了很多东西，不仅仅是具体的技能，更是发现问题解决问题的一套技术方法论。

我是在 3 月下旬开始着手开始这项课程设计，尽管已经上了一个月的课，但由于嵌入式方向多而杂，在设计过程中需要具体问题具体分析。我产生过很多想法，包括人脸识别、智

能家居助手等等，但最终锁定了实时监测系统这个主题，一是可以结合当下热点即新冠病毒疫情，二是可以充分使用树莓派的功能以及人工智能技术。

起初我连 python 怎么写都不会，由于 python 是在树莓派系统中非常方便的语言，因此我从零开始在“菜鸟教程”网站学习 python 的语法和编程规范。记得当时在 shell 上实现了各种 python 语句操作，心里是十分开心的。

当基本功都锻炼好后，为后续的功能实现奠定了基础。我的第一项功能是做网页爬虫，由于第一次接触爬虫，我在网络上寻找了很多教程，根据提供的模板写了爬取疫情实时数据的代码，经历了一段时间的 debug，最终成功实现了对数据的爬取。其他的功能比如拍照录音都不难。语音识别也能在百度 AI 上找到官方指南和 API 接口，做起来也方便。

整个项目都成功完成时，我的心里是十分自豪的。回首整个经历，最开始我连 python 语言和 Linux 命令都不会，到现在我也能成功完成一项嵌入式系统的设计了。对此我非常感谢杜老师，您在课堂上一直分享项目经验，对我们的循循善诱。希望以后自己能做出更优秀的嵌入式作品，这仅仅是个开始，未来的路还很长……

附录：Python 代码与功能说明

(1) 实时疫情数据获取：PlagueData.py

```
import requests
import json
import os
import sys
import datetime

def Down_data():
    url = 'https://view.inews.qq.com/g2/getOnsInfo?name=disease_h5'
    headers = {
        'user-agent': 'Mozilla/5.0 (Linux; Android 6.0; Nexus 5 Build/MRA58N) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/78.0.3904.108 Mobile Safari/537.36'
    }
    r = requests.get(url, headers)
    res = json.loads(r.text)
    data_res = json.loads(res['data'])
```

```

return data_res

def Parse_data1():
    data = Down_data()
    list = [ datetime.datetime.now().strftime('截至%m 月%d 日%H 时') + ','
            '全国累计确诊: ' + str(data['chinaTotal']['confirm']) + '例,'
            '新增确诊: ' + str(data['chinaAdd']['confirm']) + '例,'
            '累计境外输入: ' + str(data['chinaTotal']['importedCase']) + '例,'
            '新增境外输入: ' + str(data['chinaAdd']['importedCase']) + '例,'
            '累计治愈: ' + str(data['chinaTotal']['heal']) + '例,'
            '新增治愈: ' + str(data['chinaAdd']['heal']) + '例,'
            '累计死亡: ' + str(data['chinaTotal']['dead']) + '例,'
            '新增死亡: ' + str(data['chinaAdd']['dead']) + '例.']

    result = ''.join(list)

    with open('全国实时数据'+str(data['lastUpdateTime'])+'.txt', 'a+', encoding="utf-8") as f:
        f.write(result + '\n')

    f = open('全国实时数据'+str(data['lastUpdateTime'])+'.txt', "r")
    text = f.read()
    cmd = "ilang " + text
    os.system(cmd)

def Parse_data2():
    data = Down_data()['areaTree'][0]['children']
    path = str(input('请输入你要查询的省份: '))
    for i in data:
        if path in i['name']:
            for item in i['children']:
                list_city = [
                    '地区: ' + str(item['name']),
                    '\t 累计确诊: ' + str(item['total']['confirm']),
                    '\t 新增确诊: ' + str(item['today']['confirm']) + '\n'
                ]

```

```
res_city = ''.join(list_city)

with open(path+'省数据'+str(datetime.datetime.now().strftime('%m 月%d 日'))+'.txt', 'a+', encoding="utf-8") as f:
    f.write(res_city)

Down_data()
Parse_data1()
Parse_data2()
```

(2) 拍照功能 camera.py

```
import os
import sys
import datetime

name = datetime.datetime.now().strftime('%m 月%d 日%H:%M:%S.jpg')
cmd = "fswebcam --no-banner -r 640x480 "+name
os.system(cmd)
```

(3) 录音功能 microphone.py

```
import os
import sys
import datetime

name = datetime.datetime.now().strftime(' %m 月%d 日%H:%M:%S.wav')
time = 10
cmd = "sudo arecord -D 'plughw:1,0' -d " + str(time) + name
os.system(cmd)
```

(4) 语言转文字功能 wav2text.py

```
from aip import AipSpeech

#从百度 AI 开放平台创建应用处获取
APP_ID = '19163157'
API_KEY = 'EOh02I4yMBDs3Yjuz5feRY4n'
```

```
SECRET_KEY = 'q29Ql47tYwWZG5xn5YFndLTRNImBRrzn'  
client = AipSpeech(APP_ID, API_KEY, SECRET_KEY)  
  
# 读取文件  
def get_file_content(filePath):  
    with open(filePath, 'rb') as fp:  
        return fp.read()  
  
# 识别本地文件  
def get_text():  
    result = client.asr(get_file_content('test.wav'), 'wav', 8000, {  
        'dev_pid': 1537,})  
    #print(result)  
    text = result['result'][0]  
    return text  
  
print(get_text())
```