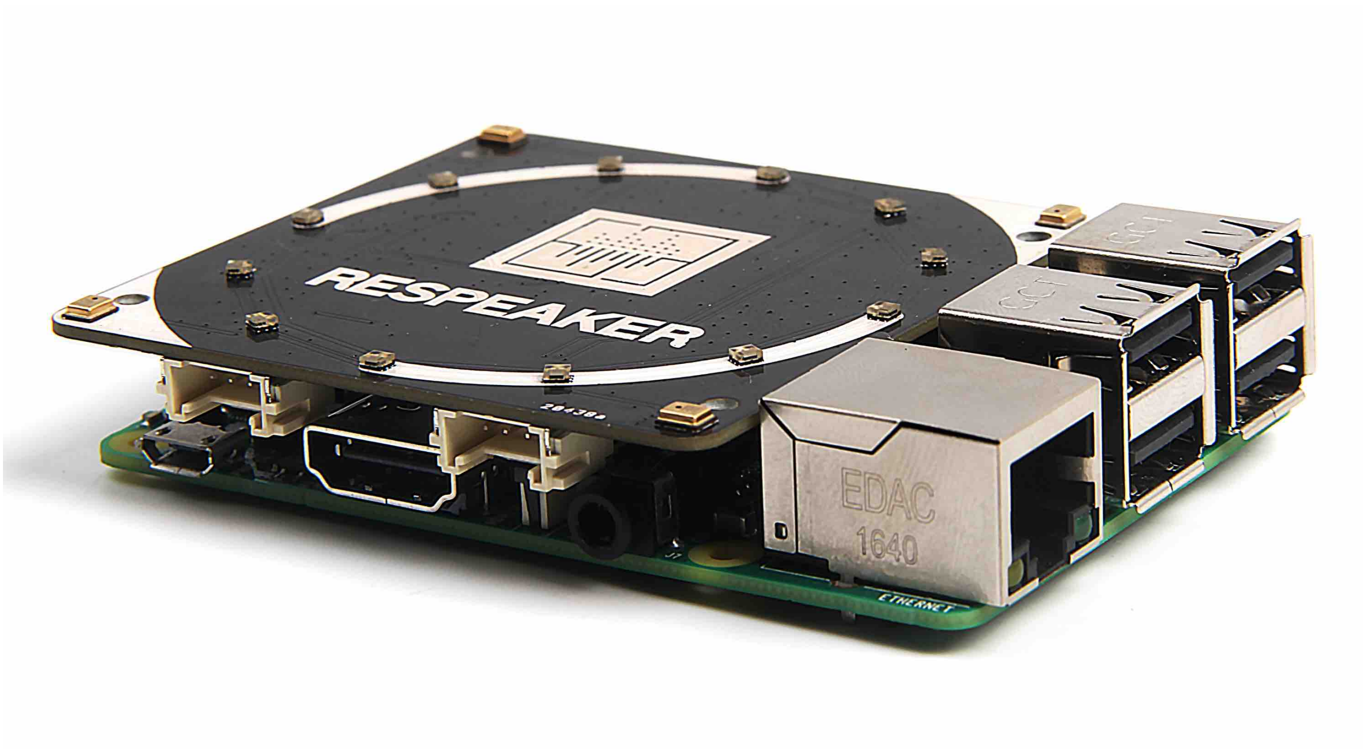
ReSpeaker 4-Mic Array for Raspberry Pi SKU: 103030216



ReSpeaker 4-Mic Array for Raspberry Pi is a quad-microphone expansion board for Raspberry Pi designed for AI and voice applications. This means that we can build a more powerful and flexible voice product that integrates Amazon Alexa Voice Service, Google Assistant, and so on.

Different from ReSpeaker 2-Mics Pi HAT, this board is developed based on AC108, a highly integrated quad-channel ADC with I2S/TDM output transition for high definition voice capture, which allows the device to pick up sounds in a 3 meters radius. Besides, this 4-Mics version provides a super cool LED ring, which contains 12 APA102 programable LEDs. With that 4 microphones and the LED ring, Raspberry Pi would have ability to do VAD(Voice Activity Detection), estimate DOA(Direction of Arrival), do KWS(Keyword Search) and show the direction via LED ring, just like Amazon Echo or Google Home.

https://www.youtube.com/embed/uAQf0RKBNHo

Features

- Raspberry Pi compatible(Support Raspberry Pi Zero and Zero W, Raspberry Pi B+, Raspberry Pi 2 B, Raspberry Pi 3 B, Raspberry Pi 3 B+ and Raspberry Pi 3 A+)
- 4 Microphones
- 3 meters radius voice capture
- 2 Grove Interfaces
- 12 APA102 User LEDs
- Software Algorithm: VAD(Voice Activity Detection), DOA(Direction of Arrival) and KWS(Keyword Search)

!!!Note There is no audio output interface on ReSpeaker 4-Mic Array for Raspberry Pi. It is only for voice capture. We can use the headphone jack on Raspberry Pi for audio output.

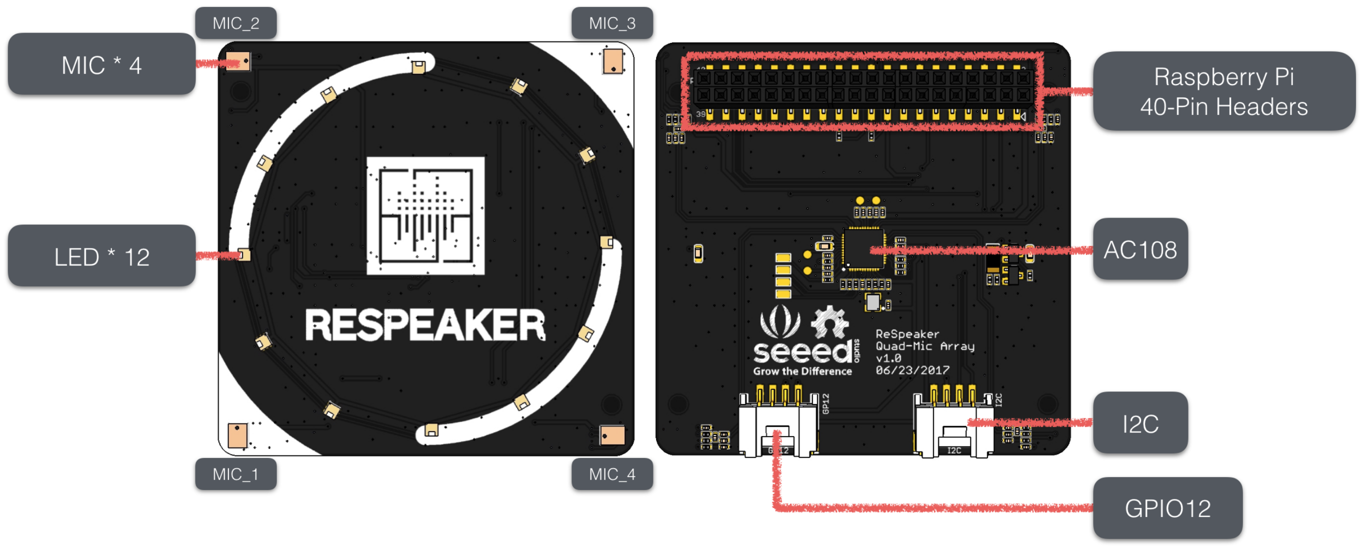
Specification

Parameter	Value/Range
Package size	L: 140mm W: 77mm H: 27mm
Gross Weight	38g

Application Ideas

- Voice Interaction Application
- AI Assistant

Hardware Overview


MIC_2
MIC_3
MIC * 4
LED * 12
MIC_1
MIC_4
RESPEAKER
Raspberry Pi 40-Pin Headers
AC108
I2C
GPIO12
seeed Grow the Difference
ReSpeaker Quad-Mic Array v1.0 06/23/2017

- MIC: 4 analog microphones
- LED: 12 APA102 programable RGB LEDs, connected to SPI interface
- Raspberry Pi 40-Pin Headers: support Raspberry Pi Zero, Raspberry Pi 1 B+, Raspberry Pi 2 B, Raspberry Pi 3 B and Raspberry Pi 3 B+
- AC108: highly integrated quad-channel ADC with I2S/TDM output transition
- I2C: Grove I2C port, connected to I2C-1
- GPIO12: Grove digital port, connected to GPIO12 & GPIO13
- Size: L:64mm W:64mm H:9mm
- Weight: 16.2g

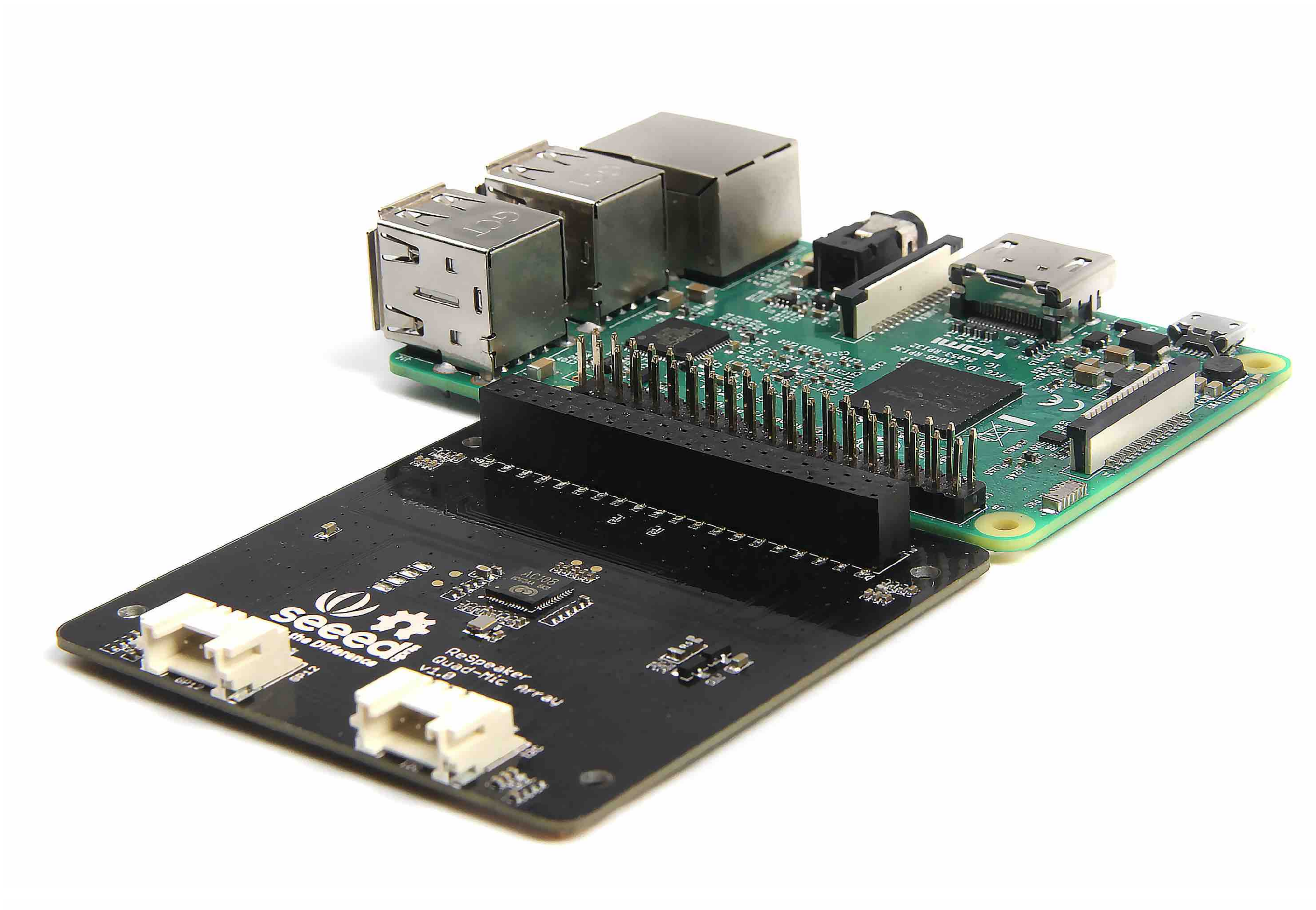
!!!Note If we want to use the APA102 RGB LEDs, please write HIGH to `GPIO5` first to enable VCC of the LEDs.

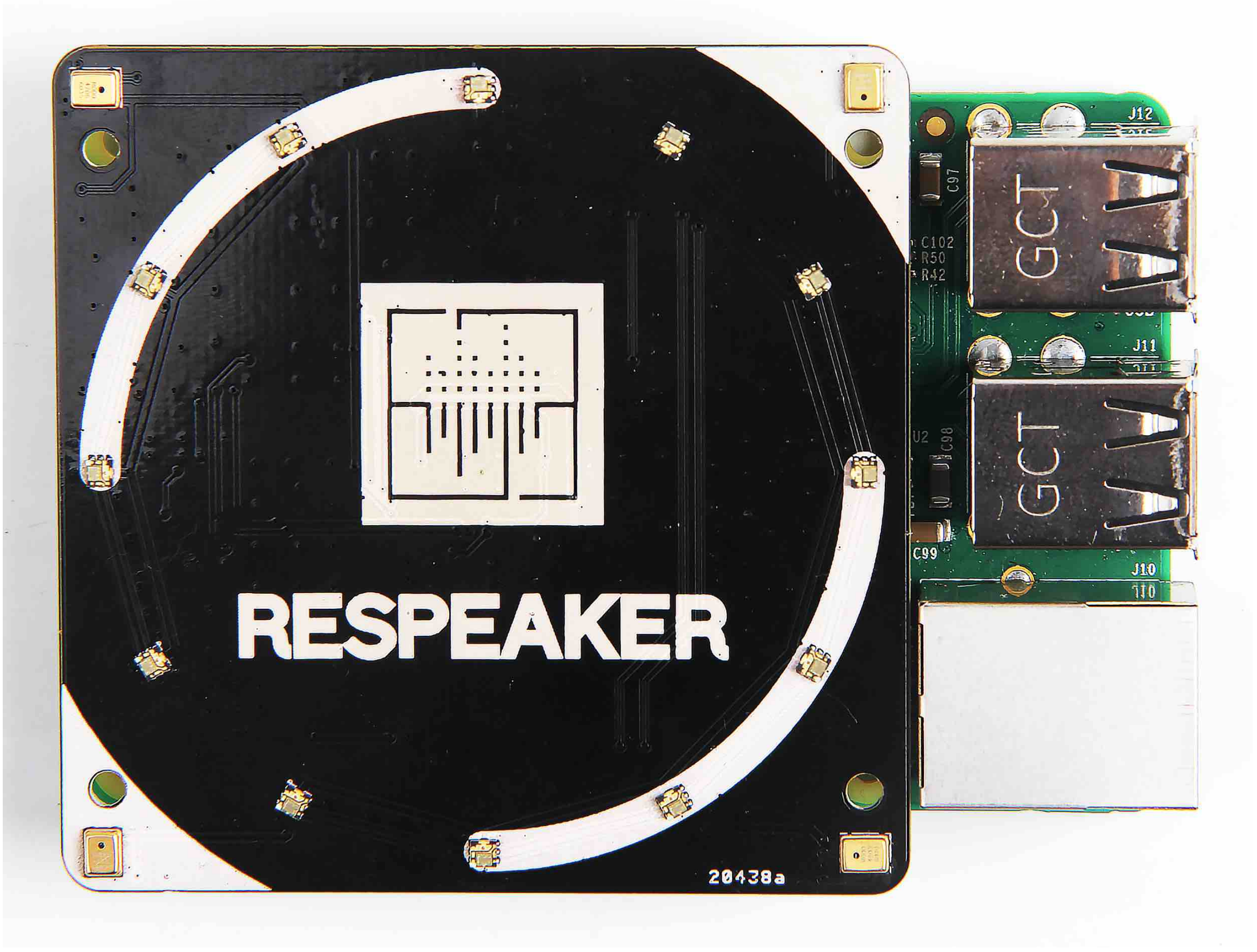
Getting Started

Connect ReSpeaker 4-Mic Array to Raspberry Pi

Mount ReSpeaker 4-Mic Array on Raspberry Pi, make sure that the pins are properly aligned when stacking the ReSpeaker 4-Mic Array for Raspberry Pi.

!!!Note Hot-plugging ReSpeaker 4-Mic Array is not allowed.It will damage the respeaker.


seeed
ReSpeaker
Quad-Mic Array
v1.0


RESPEAKER
20438a
GCT
GCT
J12
J11
J10
C97
C102
R50
R42
U2
C98
C99

Install driver

The AC108 codec is not supported by Pi kernel builds currently, we have to build it manually.

- Step 1. Please Make sure running the lastest Raspbian Operating System(debian 9) on Pi. *(updated at 2018.11.13)*
- Step 2. Get the seeed voice card source code.

```
sudo apt-get update
sudo apt-get upgrade
git clone https://github.com/respeaker/seeed-voicecard.git
cd seeed-voicecard
sudo ./install.sh
reboot
```

- Step 3. Then select the headphone jack on Raspberry Pi for audio output:

```
sudo raspi-config
# Select 7 Advanced Options
# Select A4 Audio
# Select 1 Force 3.5mm ('headphone') jack
# Select Finish
```

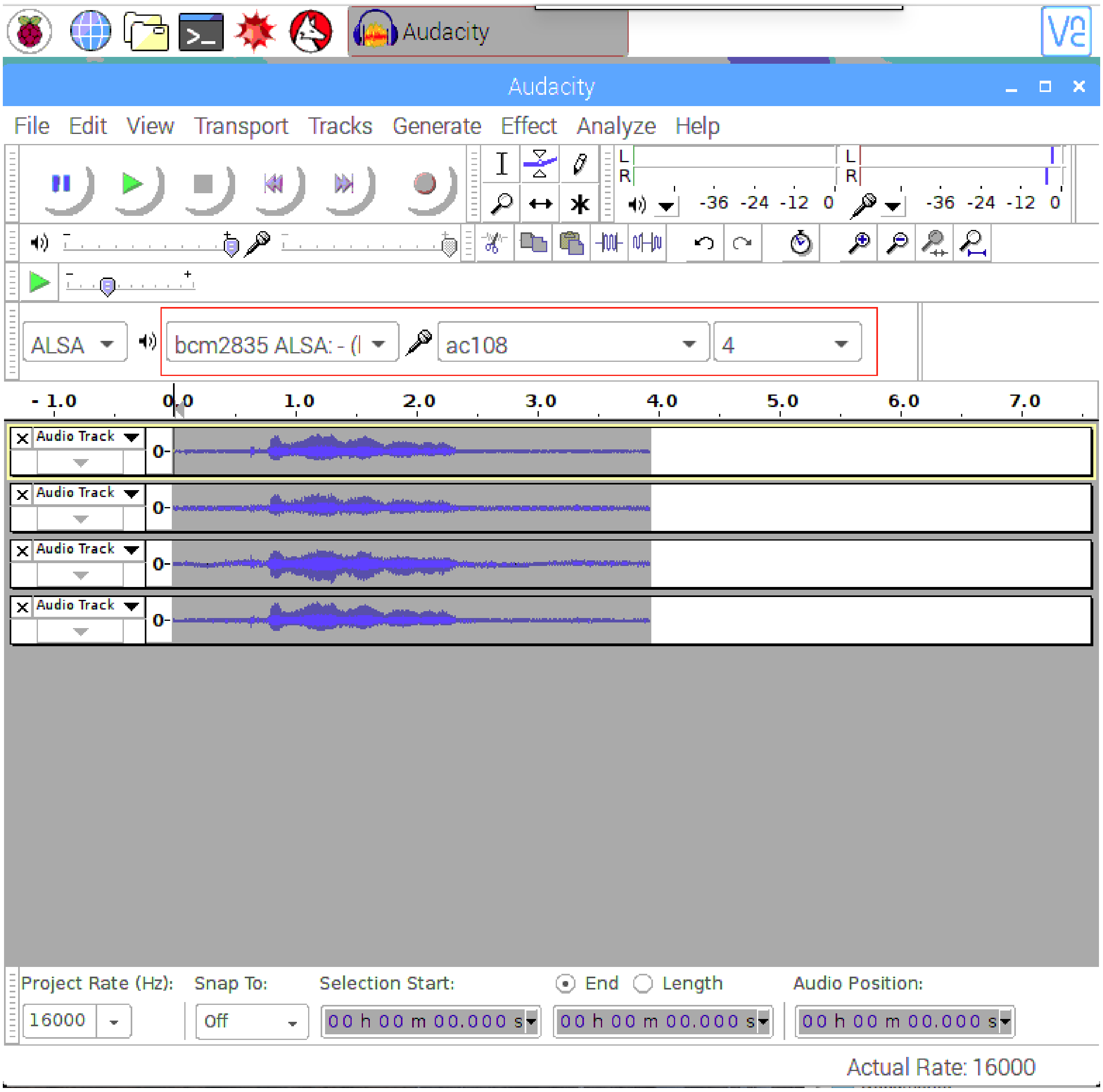
- Step 4. Check that the sound card name looks like this:

```
pi@raspberrypi:~/seeed-voicecard $ arecord -L
null
    Discard all samples (playback) or generate zero samples (capture)
playback
capture
dmixed
array
ac108
default:CARD=seeed4micvoicec
    seeed-4mic-voicecard,
    Default Audio Device
sysdefault:CARD=seeed4micvoicec
    seeed-4mic-voicecard,
    Default Audio Device
dmix:CARD=seeed4micvoicec,DEV=0
    seeed-4mic-voicecard,
    Direct sample mixing device
dsnoop:CARD=seeed4micvoicec,DEV=0
    seeed-4mic-voicecard,
    Direct sample snooping device
hw:CARD=seeed4micvoicec,DEV=0
    seeed-4mic-voicecard,
    Direct hardware device without any conversions
plughw:CARD=seeed4micvoicec,DEV=0
    seeed-4mic-voicecard,
    Hardware device with all software conversions
```

If we want to change the alsa settings, we can use `sudo alsactl --file=ac108_asound.state store` to save it. And when we need to use the settings again, copy it to: `sudo cp ~/seeed-voicecard/ac108_asound.state /var/lib/alsa/asound.state`

- Step 5. Open Audacity and select **AC108 & 4 channels** as input and **bcm2835 alsa: - (hw:0:0)** as output to test:

```
$ sudo apt update
$ sudo apt install audacity
$ audacity                      // run audacity
```



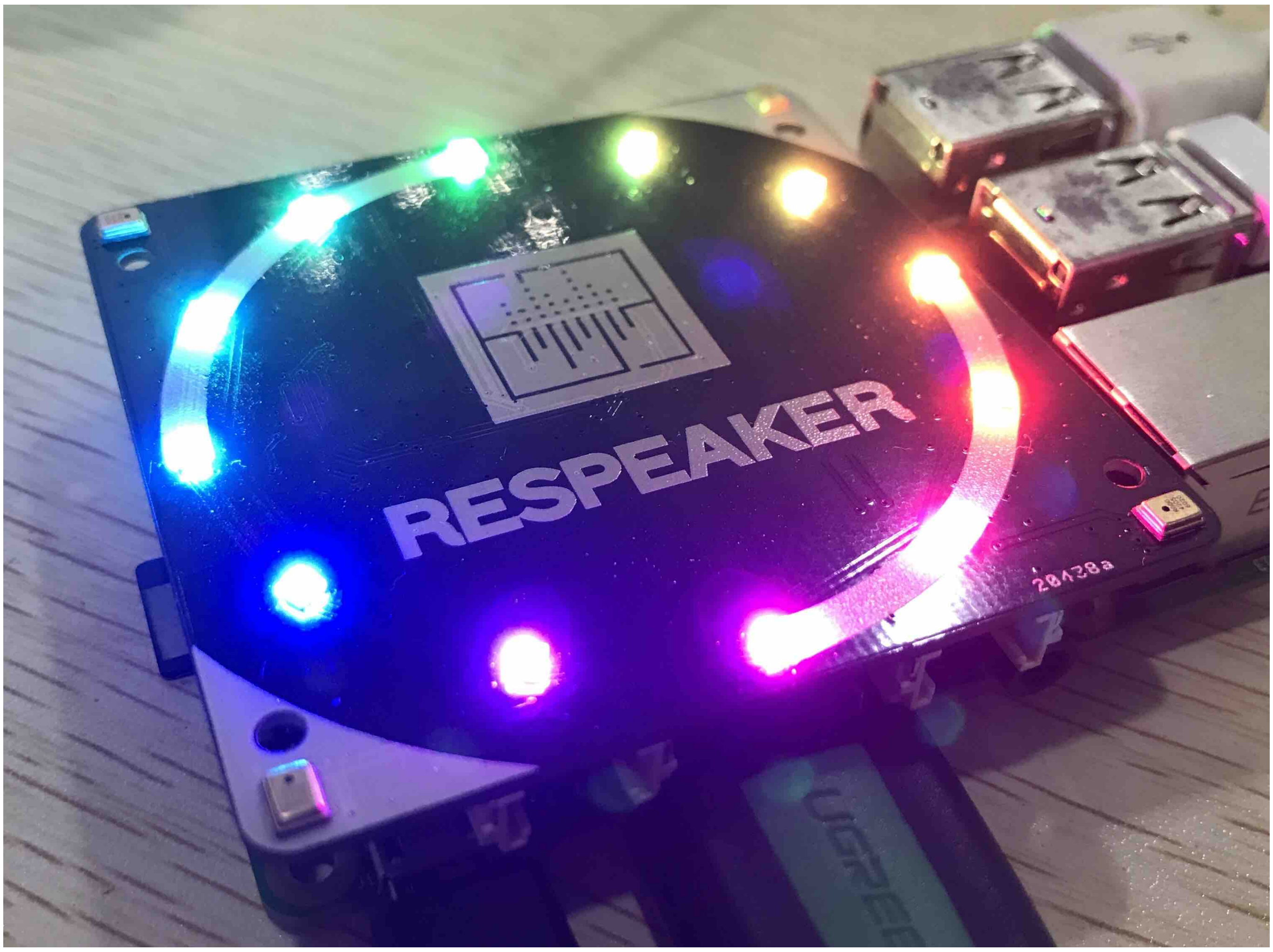
- Step 6. Or we could record with `arecord` and play with `aplay`:

```
arecord -Dac108 -f S32_LE -r 16000 -c 4 hello.wav    // only support 4 channels
aplay hello.wav                                      // make sure default device
                                                     // Audio will come out via

audio jack of Raspberry Pi
```

Play with APA102 LEDs

Each on-board APA102 LED has an additional driver chip. The driver chip takes care of receiving the desired color via its input lines, and then holding this color until a new command is received.



- Step 1. Activate SPI:
    - sudo raspi-config
    - Go to "Interfacing Options"
    - Go to "SPI"
    - Enable SPI
    - Exit the tool
- Step 2. Get APA102 LEDs Library and examples

```
pi@raspberrypi:~ $ cd /home/pi
pi@raspberrypi:~ $ git clone https://github.com/respeaker/4mics_hat.git
pi@raspberrypi:~ $ cd /home/pi/4mics_hat
pi@raspberrypi:~/4mics_hat $ sudo apt install python-virtualenv          # install python virtualenv tool
pi@raspberrypi:~/4mics_hat $ virtualenv --system-site-packages ~/env     # create a virtual python environment
pi@raspberrypi:~/4mics_hat $ source ~/env/bin/activate                   # activate the virtual environment
(env) pi@raspberrypi:~/4mics_hat $ pip install spidev gpiozero           # install spidev and gpiozero
```

- Step 3. Then run the example code under virtualenv, now we can see the LEDs blink like Google Assistant.

```
(env) pi@raspberrypi:~/4mics_hat $ python pixels_demo.py
```

DoA

DOA with Keywords

With DoA(Direction of Arrial), ReSpeaker 4-Mic Array is able to find the direction where the sound source is located.

- Step 1. setup voice engine

```
pi@raspberrypi:~ $ source ~/env/bin/activate                    # activate the
virtual, if we have already activated, skip this step
(env) pi@raspberrypi:~ $ cd ~/4mics_hat
(env) pi@raspberrypi:~/4mics_hat $ sudo apt install libatlas-base-dev     #
install snowboy dependencies
(env) pi@raspberrypi:~/4mics_hat $ sudo apt install python-pyaudio        #
install pyaudio
(env) pi@raspberrypi:~/4mics_hat $ pip install ./snowboy*.whl             #
install snowboy for KWS
(env) pi@raspberrypi:~/4mics_hat $ pip install ./webrtc*.whl              #
install webrtc for DoA
(env) pi@raspberrypi:~/4mics_hat $ cd ~/
(env) pi@raspberrypi:~ $ git clone https://github.com/voice-engine/voice-engine
(env) pi@raspberrypi:~ $ cd voice-engine/
(env) pi@raspberrypi:~/voice-engine $ python setup.py install
```

- Step 2. Run the demo under virtualenv. Please wake up respeaker with saying `snowboy` and we will see the direction.

```
(env) pi@raspberrypi:~/voice-engine $ cd ~/4mics_hat
(env) pi@raspberrypi:~/4mics_hat $ python kws_doa.py
```

DOA without Keywords

- Step 1. Setup the dependency

```
sudo apt-get install portaudio19-dev
sudo pip install pyaudio
sudo pip install webrtcvad
sudo apt-get install python-numpy
sudo pip install pyusb
```

- Step 2. Run the vad_doa.py

```
cd ~
git clone https://github.com/respeaker/mic_array.git
cd mic_array
nano vad_doa.py
#change CHANNELS = 8 to CHANNELS = 4 @line10
python vad_doa.py
```

- Step 3. Here is the output.

```
1111110000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000111111011111111111111111111
135
10000000000000000000000000000000000000000000000000000000000000011111111111111111
135
11111111111111111111
135
1001111111100000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000111111111100000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000111111111111111110000000
135
0000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000011111111111111111
135
```

Alexa/Baidu/Snowboy SDK

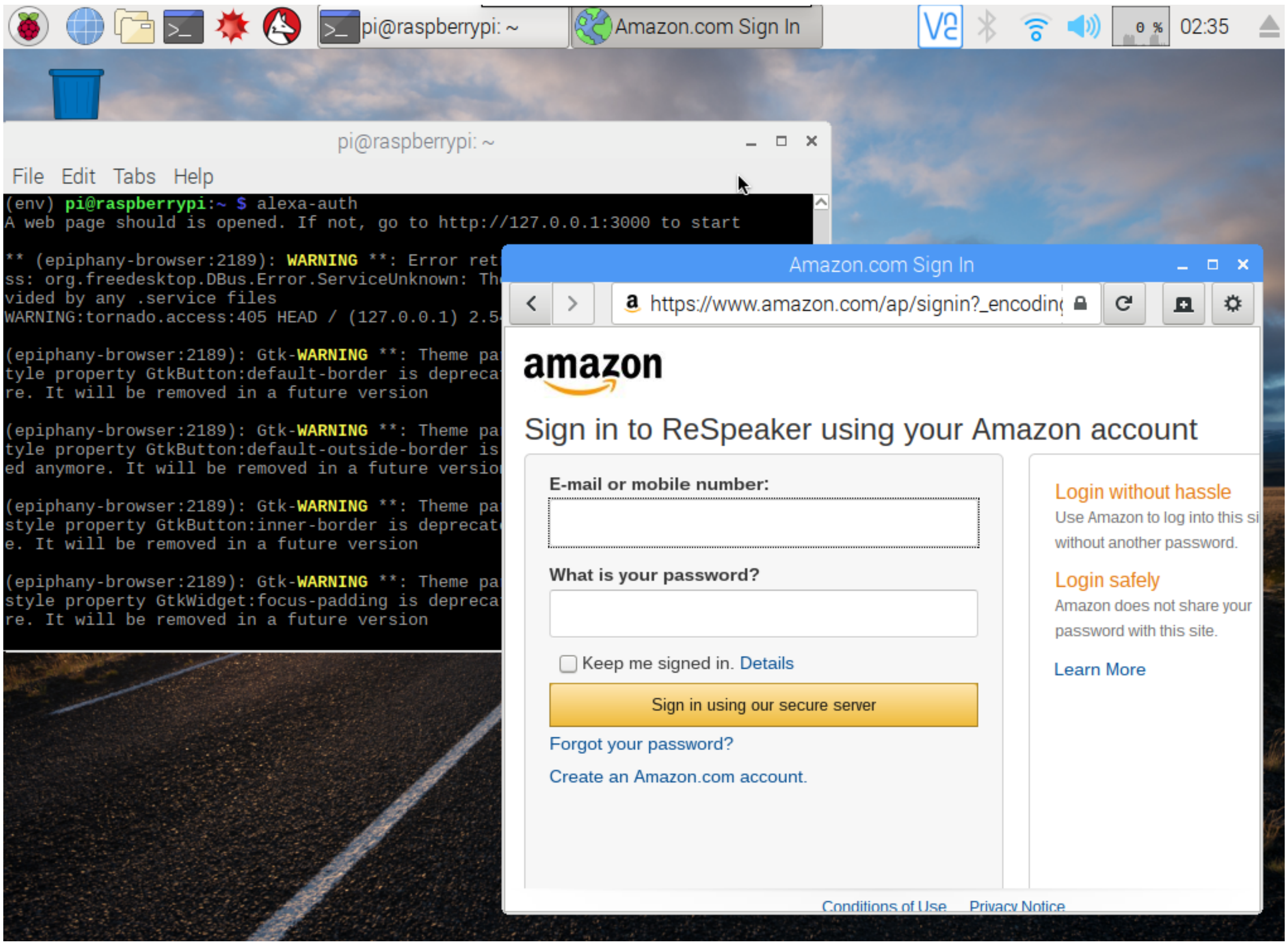
- Step 1. Get Alexa or Baidu authorization

```
pi@raspberrypi:~ $ source ~/env/bin/activate                    # activate the
virtual, if we have already activated, skip this step
(env) pi@raspberrypi:~ $ cd ~/
(env) pi@raspberrypi:~ $ git clone https://github.com/respeaker/avs
(env) pi@raspberrypi:~ $ cd avs                                 # install
Requirements
(env) pi@raspberrypi:~/avs $ python setup.py install
(env) pi@raspberrypi:~/avs $ sudo apt-get install gstreamer1.0-plugins-good
gstreamer1.0-plugins-bad gstreamer1.0-plugins-ugly gir1.2-gstreamer-1.0 python-gi
python-gst-1.0
(env) pi@raspberrypi:~/avs $ pip install tornado
```

Then open a terminal at VNC, switch to the python virtual env with

```
# note: this is in the VNC terminal
pi@raspberrypi:~ $ source ~/env/bin/activate
(env) pi@raspberrypi:~ $ alexa-auth
```

Run `alexa-auth` in the terminal to get Alexa authorization or run `dueros-auth` to get Baidu authorization. The authorization file will be saved in `/home/pi/.avs.json`.



```
(env) pi@raspberrypi:~ $ alexa-auth
A web page should is opened. If not, go to http://127.0.0.1:3000 to start

** (epiphany-browser:2189): WARNING **: Error ret
ss: org.freedesktop.DBus.Error.ServiceUnknown: Th
vided by any .service files
WARNING:tornado.access:405 HEAD / (127.0.0.1) 2.5

(epiphany-browser:2189): Gtk-WARNING **: Theme pa
tyle property GtkButton:default-border is depreca
re. It will be removed in a future version

(epiphany-browser:2189): Gtk-WARNING **: Theme pa
tyle property GtkButton:default-outside-border is
ed anymore. It will be removed in a future versio

(epiphany-browser:2189): Gtk-WARNING **: Theme pa
style property GtkButton:inner-border is deprecat
e. It will be removed in a future version

(epiphany-browser:2189): Gtk-WARNING **: Theme pa
style property GtkWidget:focus-padding is depreca
re. It will be removed in a future version
```

Sign in to ReSpeaker using your Amazon account

E-mail or mobile number:

What is your password?

Keep me signed in. Details

Sign in using our secure server

Forgot your password?

Create an Amazon.com account.

Login without hassle
Use Amazon to log into this si
without another password.

Login safely
Amazon does not share your
password with this site.

Learn More

Conditions of Use Privacy Notice

!!!Note If we want to switch between `alexa-auth` and `dueros-auth`, please delete `/home/pi/.avs.json` first. The file is hidden and use the `ls -la` to list the file.

- Step 2. Let's Enjoy!

Now run `python ns_kws_doa_alexa_with_light.py` under virtualenv, we will see lots of debug message rasing in the terminal. When we see **status code: 204**, try to wake up respeaker with saying `snowboy`. Then the leds will light up, and now we can speak to it, for example, asking "who is the most handsome guy in the world?" or "Do you know how to beat-box?". Just enjoy it!

```
(env) pi@raspberrypi:~/avs $ cd ~/4mics_hat
(env) pi@raspberrypi:~/4mics_hat $ python ns_kws_doa_alexa_with_light.py
```

STT(Speech to Text) SDK

This part will introduce Baidu STT(Speech to Text) functions together with GPIO control. Here is the GPIOs configuration. If you do not have a fan, You can connect 2 LEDs on GPIO12/GPIO13 to demonstrate.

GPIO	Turn On	Faster	Slower	Turn Off
GPIO12	1	0	1	0
GPIO13	0	1	0	0

- **Step 1. Install dependiencies**

```
sudo apt install mpg123
pip install baidu-aip monotonic pyaudio
```

- **Step 2. Get Baidu key from Here.**
- **Step 3. Download the Smart_Fan.py**

```
cd ~
wget
https://github.com/SeeedDocument/MIC_HATv1.0_for_raspberrypi/raw/master/src/baidu_
STT.zip
unzip baidu_STT.zip
cd baidu_STT
python Smart_Fan.py
```

!!!Warning Please add baidu key @ line 36,37,38 before running python Smart_Fan.py. You also can generate your owner voice by running the synthesis_wav.py. Please do add baidu keys at line 6,7,8 and modify string to what you want to generate.

- **Step 4. Let's say '开风扇'.**
- **Step 5. You will see the fan moving.**
- **Step 6. Let's try '快一点', '慢一点' and '关风扇'.**

Extract Voice

We use PyAudio python library to extract voice.

- Step 1, We need to run the following script to get the device index number of 4 Mic pi hat:

```
sudo pip install pyaudio
cd ~
nano get_index.py
```

- Step 2, copy below code and paste on get_index.py.

```
import pyaudio

p = pyaudio.PyAudio()
info = p.get_host_api_info_by_index(0)
numdevices = info.get('deviceCount')

for i in range(0, numdevices):
        if (p.get_device_info_by_host_api_device_index(0,
i).get('maxInputChannels')) > 0:
            print "Input Device id ", i, " - ",
p.get_device_info_by_host_api_device_index(0, i).get('name')
```

- Step 3, press Ctrl + X to exit and press Y to save.
- Step 4, run 'sudo python get_index.py' and we will see the device ID as below.

```
Input Device id  2  -  seeed-4mic-voicecard: - (hw:1,0)
```

- Step 5, change `RESPEAKER_INDEX = 2` to index number. Run python script record.py to record a speech.

```
import pyaudio
import wave

RESPEAKER_RATE = 16000
RESPEAKER_CHANNELS = 4
RESPEAKER_WIDTH = 2
# run getDeviceInfo.py to get index
RESPEAKER_INDEX = 2  # refer to input device id
CHUNK = 1024
RECORD_SECONDS = 5
WAVE_OUTPUT_FILENAME = "output.wav"

p = pyaudio.PyAudio()

stream = p.open(
            rate=RESPEAKER_RATE,
            format=p.get_format_from_width(RESPEAKER_WIDTH),
            channels=RESPEAKER_CHANNELS,
            input=True,
            input_device_index=RESPEAKER_INDEX,)

print("* recording")

frames = []

for i in range(0, int(RESPEAKER_RATE / CHUNK * RECORD_SECONDS)):
    data = stream.read(CHUNK)
    frames.append(data)

print("* done recording")

stream.stop_stream()
stream.close()
p.terminate()

wf = wave.open(WAVE_OUTPUT_FILENAME, 'wb')
wf.setnchannels(RESPEAKER_CHANNELS)
wf.setsampwidth(p.get_sample_size(p.get_format_from_width(RESPEAKER_WIDTH)))
wf.setframerate(RESPEAKER_RATE)
wf.writeframes(b''.join(frames))
wf.close()
```

- Step 6. If you want to extract channel 0 data from 4 channels, please follow below code. For other channel X, please change [0::4] to [X::4].

```
import pyaudio
import wave
import numpy as np

RESPEAKER_RATE = 16000
RESPEAKER_CHANNELS = 4
RESPEAKER_WIDTH = 2
# run getDeviceInfo.py to get index
RESPEAKER_INDEX = 2  # refer to input device id
CHUNK = 1024
RECORD_SECONDS = 3
WAVE_OUTPUT_FILENAME = "output.wav"

p = pyaudio.PyAudio()

stream = p.open(
            rate=RESPEAKER_RATE,
            format=p.get_format_from_width(RESPEAKER_WIDTH),
            channels=RESPEAKER_CHANNELS,
            input=True,
            input_device_index=RESPEAKER_INDEX,)

print("* recording")

frames = []

for i in range(0, int(RESPEAKER_RATE / CHUNK * RECORD_SECONDS)):
    data = stream.read(CHUNK)
    # extract channel 0 data from 4 channels, if you want to extract channel 1,
please change to [1::4]
    a = np.fromstring(data,dtype=np.int16)[0::4]
    frames.append(a.tostring())
```

```
print("* done recording")

stream.stop_stream()
stream.close()
p.terminate()

wf = wave.open(WAVE_OUTPUT_FILENAME, 'wb')
wf.setnchannels(1)
wf.setsampwidth(p.get_sample_size(p.get_format_from_width(RESPEAKER_WIDTH)))
wf.setframerate(RESPEAKER_RATE)
wf.writeframes(b''.join(frames))
wf.close()
```

Realtime Sound Source Localization and Tracking

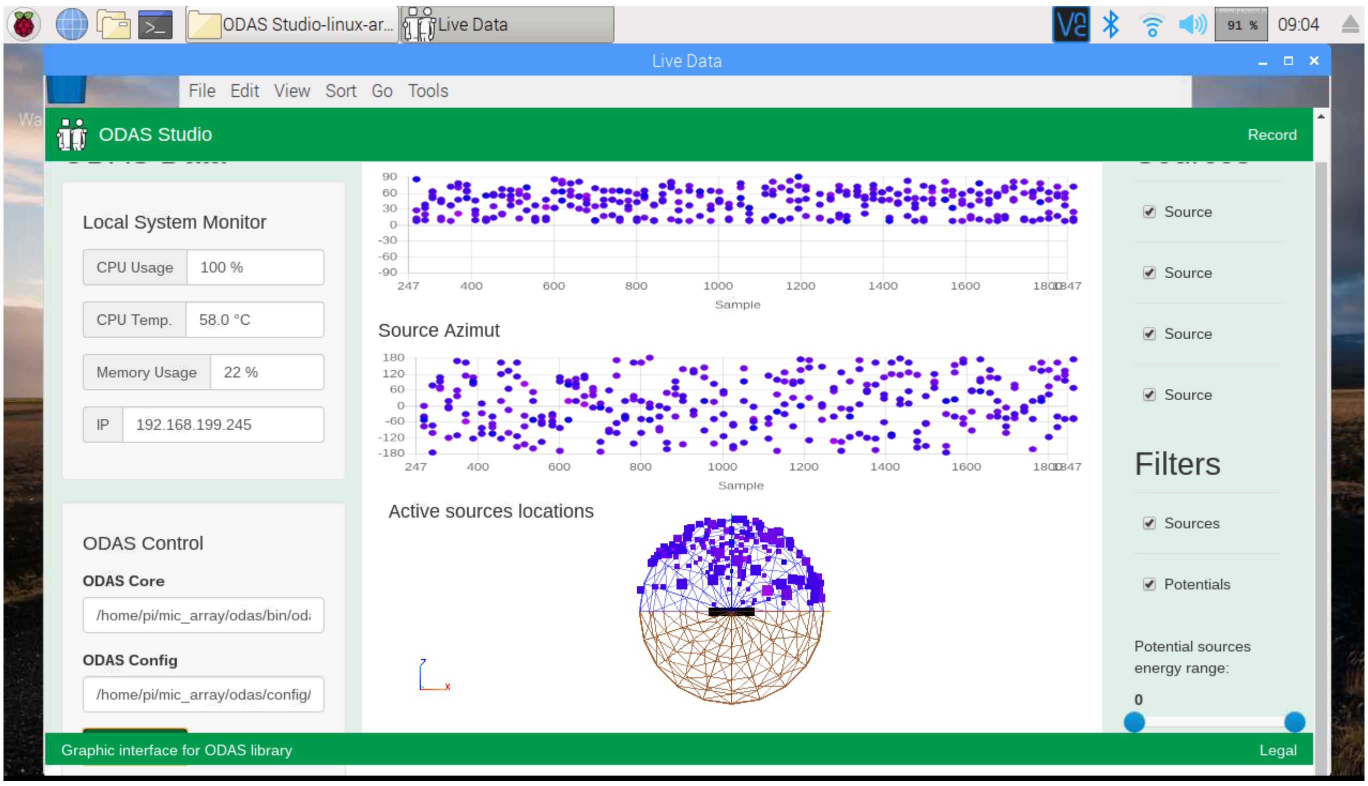
ODAS stands for Open embeddeD Audition System. This is a library dedicated to perform sound source localization, tracking, separation and post-filtering. Let's have a fun with it.

- Step 1. Get ODAS and build it.

```
sudo apt-get install libfftw3-dev libconfig-dev libasound2-dev libgconf-2-4
sudo apt-get install cmake
git clone https://github.com/introlab/odas.git
mkdir odas/build
cd odas/build
cmake ..
make
```

- Step 2. Get ODAS Studio and open it.

- Step 3. The odascore will be at **odas/bin/odaslive**, the config file is at **odas/config/odaslive/respeaker_4_mic_array.cfg**.



FAQ

Q1: How to change the Raspbian Mirrors source?

A1: Please refer to Raspbian Mirrors and follow below instructions to modify the source at begining.

```
pi@raspberrypi ~ $ sudo nano /etc/apt/sources.list
```

For example, we suggest use the tsinghua source for China users. So please modify the sources.list as below.

```
deb http://mirrors.tuna.tsinghua.edu.cn/raspbian/raspbian/ stretch main non-free
contrib
deb-src http://mirrors.tuna.tsinghua.edu.cn/raspbian/raspbian/ stretch main non-
free contrib
```

Q2: We can hear the voice by aplay from the 3.5mm audio jack but we can't hear the voice when running ns_kws_doa_alexa_with_light.py

A2: We have 3 players (mpv, mpg123 and gstreamer) to use. SpeechSynthesizer and Alerts prefer mpg123 which is more responsive. AudioPlayer likes gstreamer > mpv > mpg123. Gstreamer supports more audio format and works well on raspberry pi. We can also specify the player of AudioPlayer using the environment variable PLAYER. So please try below commands to enable the voice.

```
sudo apt install mpg123
PLAYER=mpg123 python ns_kws_doa_alexa_with_light.py
```

Q3: There is no response When we run kws_doa.py and say snowboy

A3: Please run audacity to make sure 4 channels are good. If there is one channel without data, there will be no response when we say snowboy.

Q4: #include "portaudio.h" Error when run "sudo pip install pyaudio".

A4: Please run below command to solve the issue.

```
sudo apt-get install portaudio19-dev
```

Resources

- **[PDF]** ReSpeaker 4-Mic Array for Raspberry Pi(PDF)
- **[DXF]** ReSpeaker 4-Mic Array for Raspberry Pi v1.0
- **[3D]** ReSpeaker 4-Mic Array for Raspberry Pi v1.0 3D Model
- **[AC108]** AC108 DataSheet
- **[Driver]** Seeed-Voice Driver
- **[Algorithms]** Algorithms includes DOA, VAD, NS
- **[Voice Engine]** Voice Engine project, provides building blocks to create voice enabled objects
- **[Algorithms]** AEC

Tech Support

Please submit any technical issue into our forum or drop mail to techsupport@seeed.cc.