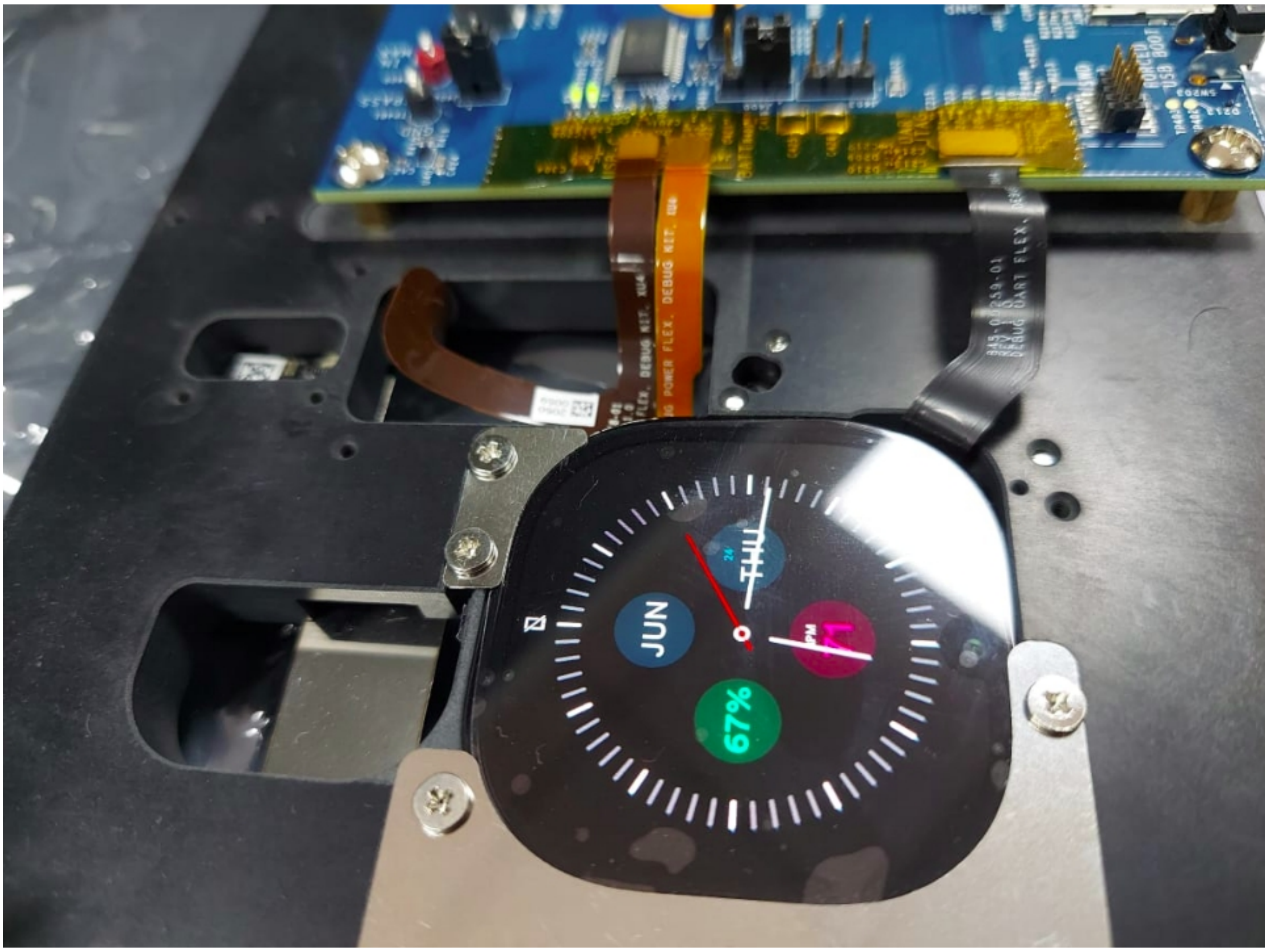
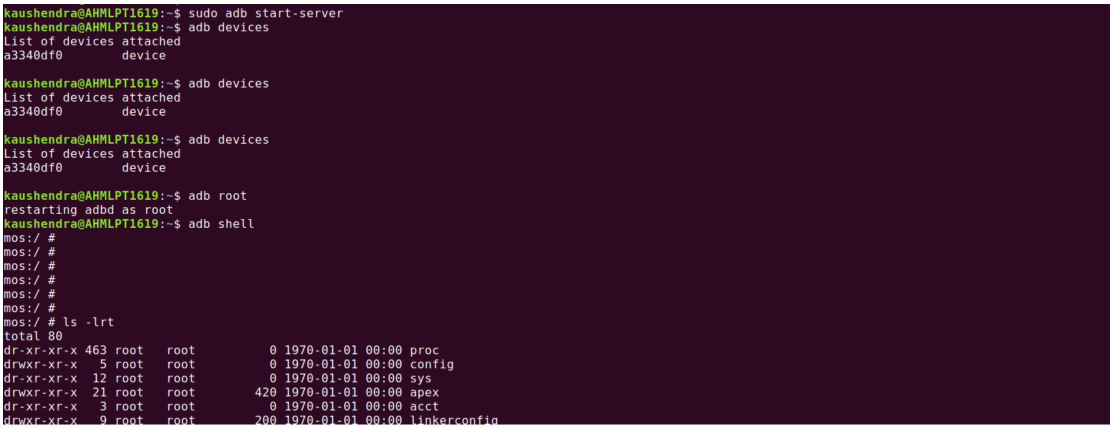
NED BOARD

Step 1: Connect USB-Type-B with Power Socket and USB-Type-C to Host Machine and press power button to turn on the NED board.



Step 2:get the adb of the Board using below commands.

$sudo adb start-server
$adb devices
$adb root
$adb shell



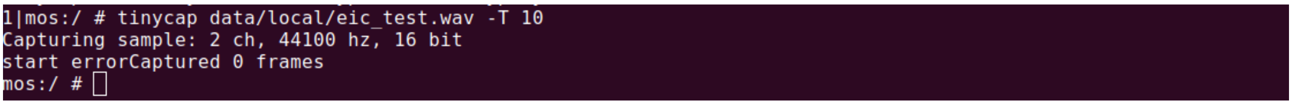
```
kaushendra@AHMLPT1619:~$ sudo adb start-server
kaushendra@AHMLPT1619:~$ adb devices
List of devices attached
a3340df0        device

kaushendra@AHMLPT1619:~$ adb devices
List of devices attached
a3340df0        device

kaushendra@AHMLPT1619:~$ adb devices
List of devices attached
a3340df0        device

kaushendra@AHMLPT1619:~$ adb root
restarting adbd as root
kaushendra@AHMLPT1619:~$ adb shell
mos:/ #
mos:/ #
mos:/ #
mos:/ #
mos:/ #
mos:/ #
mos:/ # ls -lrt
total 80
dr-xr-xr-x 463 root   root          0 1970-01-01 00:00 proc
drwxr-xr-x   5 root   root          0 1970-01-01 00:00 config
dr-xr-xr-x  12 root   root          0 1970-01-01 00:00 sys
drwxr-xr-x  21 root   root        420 1970-01-01 00:00 apex
dr-xr-xr-x   3 root   root          0 1970-01-01 00:00 acct
drwxr-xr-x   9 root   root        200 1970-01-01 00:00 linkerconfig
```

Step 3: use the tinycap utility to generate an audio file.
$ tinycap filename -T timeout

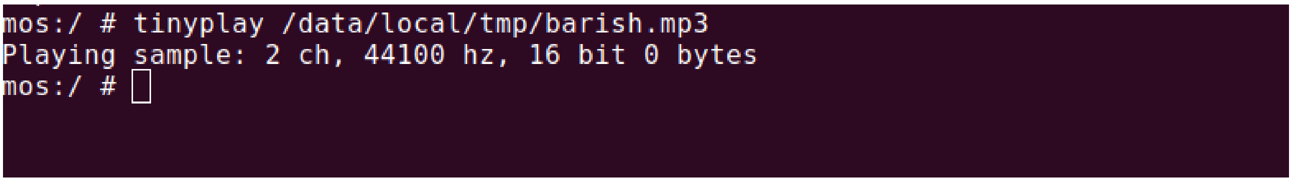


```
1|mos:/ # tinycap data/local/eic_test.wav -T 10
Capturing sample: 2 ch, 44100 hz, 16 bit
start errorCaptured 0 frames
mos:/ # 
```

Note: we found error while capturing the audio,need help to understand this scenario.

Step 5: Running the audio file over the board.

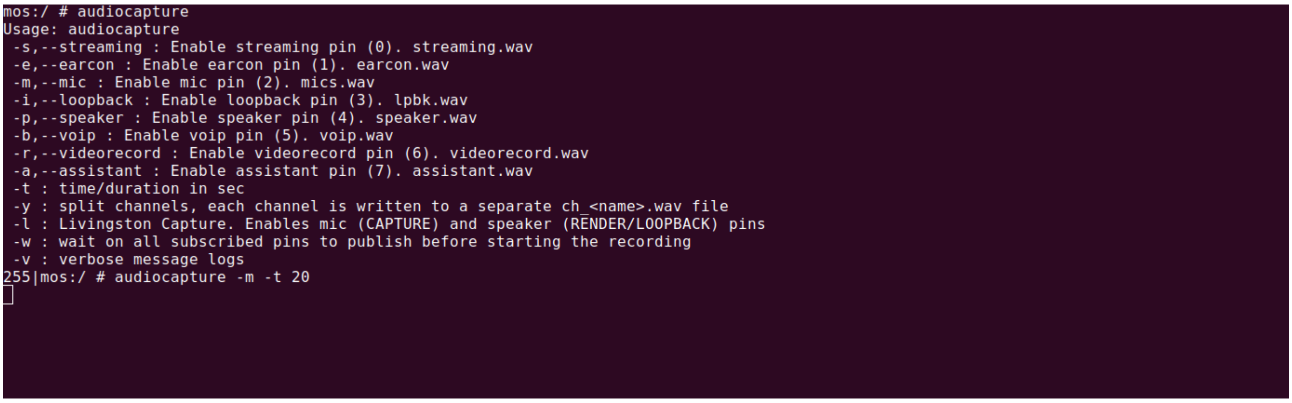
$tinyplay file_name_audio



```
mos:/ # tinyplay /data/local/tmp/barish.mp3
Playing sample: 2 ch, 44100 hz, 16 bit 0 bytes
mos:/ # 
```

Note:we are not able to hear any audio while playing

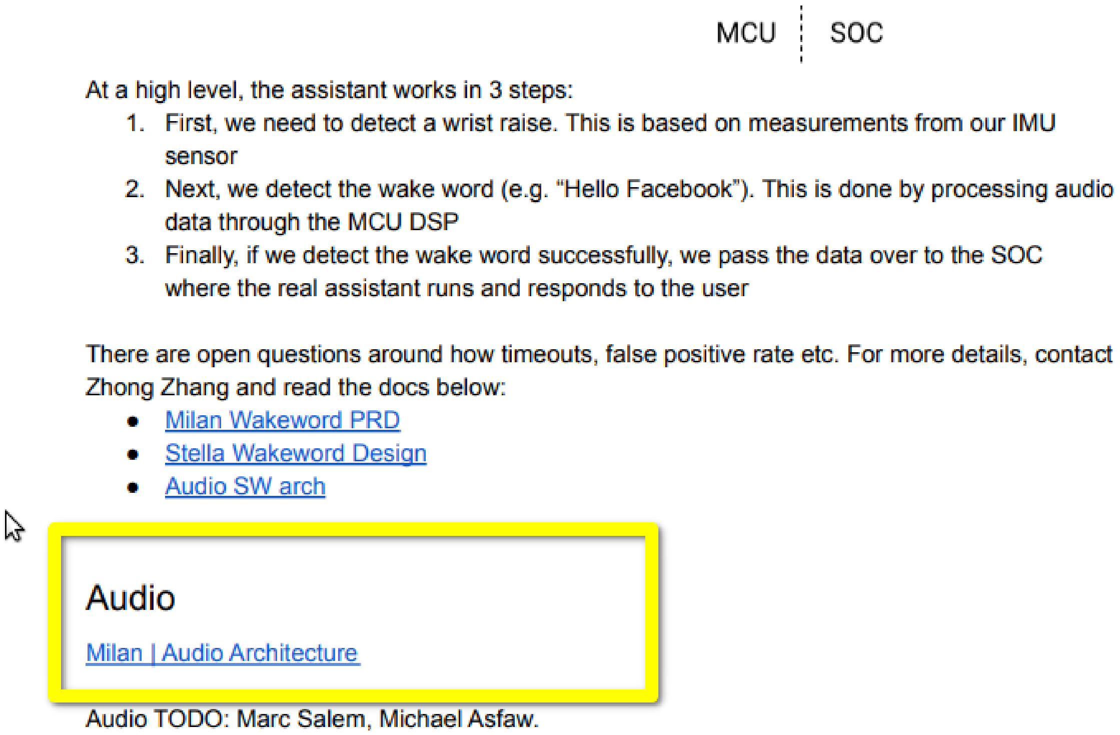
Step 6: Checking audiocapture binary over NED Board :



```
mos:/ # audiocapture
Usage: audiocapture
 -s,--streaming : Enable streaming pin (0). streaming.wav
 -e,--earcon : Enable earcon pin (1). earcon.wav
 -m,--mic : Enable mic pin (2). mics.wav
 -i,--loopback : Enable loopback pin (3). lpbk.wav
 -p,--speaker : Enable speaker pin (4). speaker.wav
 -b,--voip : Enable voip pin (5). voip.wav
 -r,--videorecord : Enable videorecord pin (6). videorecord.wav
 -a,--assistant : Enable assistant pin (7). assistant.wav
 -t : time/duration in sec
 -y : split channels, each channel is written to a separate ch_<name>.wav file
 -l : Livingston Capture. Enables mic (CAPTURE) and speaker (RENDER/LOOPBACK) pins
 -w : wait on all subscribed pins to publish before starting the recording
 -v : verbose message logs
255|mos:/ # audiocapture -m -t 20
```

Note: we need your help to understand how to use this utility and commands and their responses.

We Need the below document



At a high level, the assistant works in 3 steps:

1. First, we need to detect a wrist raise. This is based on measurements from our IMU sensor
2. Next, we detect the wake word (e.g. "Hello Facebook"). This is done by processing audio data through the MCU DSP
3. Finally, if we detect the wake word successfully, we pass the data over to the SOC where the real assistant runs and responds to the user

There are open questions around how timeouts, false positive rate etc. For more details, contact Zhong Zhang and read the docs below:

- Milan Wakeword PRD
- Stella Wakeword Design
- Audio SW arch

Audio

Milan | Audio Architecture

Audio TODO: Marc Salem, Michael Asfaw.

Note: We tried to run the Sound recoder.apk also finds bugs using this,like file not getting generated.