EE123
Digital Signal Processing

Lecture 13A
Project

* Beautiful handwritten figures by Prof. Murat Arcak
Project

• Project Teams this Friday, April 13th
  – Can be in teams up to 3 — scope increases accordingly
  – Submit 1 paragraphs project proposal (More info later)
• There will be Weekly 5-10min project followups
  – Period I 04/16-20, Period II 04/23-27
• Projects are Due May 04th.

• Project Deliverables
  – Software
  – Demo
  – A few slides / Poster
  – A 3min video explaining your project and a demo
Competition Project

• Image communication
  – We will give you an image
  – You will need to transmit it with the best quality over a limited amount of time (75 seconds)
  – Evaluation is based on PICSNR and visual quality score

• You can use ANY method you write yourself
  – Compression
  – Filtering, image recovery....
  – Modulation (digital or analog), detection, ......
Competition project

• Evaluation based on
  – Comparison to a baseline implementation with packet APRS -- slow and low res
  – Scope
  – Creativity
  – Presentation

• Winners gets a prize: radio

• If you want to do a different project — you must get prior approval
More Details

• Project proposal: One paragraph
  – Who is in the team
  – What will you attempt — who will do what.

• Weekly meetings
  – Make 2-3 slides on progress for every meeting
  – Slides should show progress and preliminary results, experimentations an TODO’s for next meeting

• https://www.youtube.com/watch?v=WaeTTmh2IRg
Project Webpage

• https://inst.eecs.berkeley.edu/~ee123/sp18/project.html
Lab 4 Part I

- New and old interfaces
- Allow you to connect the Pi to the radio
- Purpose of this lab:
  - Learn how to debug transmission and recording for a more successful part II and III
Radio Webpage

- https://inst.eecs.berkeley.edu/~ee123/sp18/radio.html
Lab 4 Part I

In [5]: `sd.query_devices()`

Out[5]:
0 bcm2835 ALSA: - (hw:0,0), ALSA (0 in, 2 out)
1 bcm2835 ALSA: IEC958/HDMI (hw:0,1), ALSA (0 in, 2 out)
2 USB PnP Sound Device: Audio (hw:1,0), ALSA (1 in, 2 out)
3 sysdefault, ALSA (0 in, 128 out)
4 dmix, ALSA (0 in, 2 out)
* 5 default, ALSA (1 in, 2 out)

Manually set the audio device numbers.

`builtin_idx` should be the bcm2835 ALSA, but not the one with HDMI

`usb_idx` should be the USB audio

In [6]:
   `builtin_idx = 0`
   `usb_idx = 2`
```python
def queuereplay_callback(indata, outdata, frames, time, status):
    if status:
        print(status)
    outdata[:] = indata
Qin.put(indata)  # Global queue

# create an input FIFO queue
Qin = Queue.Queue()

st = sd.Stream(device=(USB_idx, builtin_idx), callback=queuereplay_callback)
st.start()
```
Calibrating the SDR frequency

- Disconnect the SDR antenna when doing this experiment
- Radio should ALWAYS have antenna connected
- Transmit with the radio — receive with the SDR.
  - Find the frequency offset

shift in Hz: [-31258.53658537]
shift in ppm: [ 71.]

![Graph showing calibrated frequency shift](image-url)
Calibrating output audio level

- Transmit tone with increasing audio amplitude
- FM BW should increase

spectrogram of FM

spectrogram of demodulated FM
Calibrating output audio level

- Transmit tone with increasing audio amplitude

![Spectrogram of demodulated FM](image1)

- Linear regime
- Saturated

![Audio amplitude curve](image2)

Find output audio amp here
Measure Input Audio BP filter

• Radio filters audio coming in.
  – Pre-emphasizes higher frequencies, since they have more noise in FM

• Measure by:
  – Playing a chirp and receiving with SDR
  – Playing noise and looking at average power spectrum
Measure Input Audio BP filter
Measure Input Audio BP filter
Write a function to generate morse code

```python
def text2Morse(text, fc, fs, dt):
    CODE = {
        'A': '.-', 'B': '-...', 'C': '-.-.',
        'D': '-....', 'E': '...', 'F': '..-.',
        'G': '--.', 'H': '....', 'I': '.',
        'J': '--.-', 'K': '-..', 'L': '..',
        'M': '--', 'N': '-.', 'O': '---',
        'P': '.--.', 'Q': '--.-', 'R': '.-.
        'S': '...', 'T': '-', 'U': '..-',
        'V': '...-', 'W': '.--', 'X': '-..-',
        'Y': '-.--', 'Z': '--..',
        '0': '-----', '1': '.----', '2': '..---',
        '3': '...--', '4': '....-', '5': '.....',
        '6': '-....', '7': '--...', '8': '---..',
        '9': '-----',
        ':': '---..', ';': '---..', '(': '----.',
        ')': '-----',
    }
```

- Implement a function `sig = text2Morse(text, fc, fs, dt)`. The function will take a string and convert it to a tone
- Generate Morse code from text
- Can be used to identify yourself