ENG ECE463 First Prototype Testing Plan
11/16/21
Team 26
RFSoC Spectrum Monitoring
Yana Galina, Jaime Mohedano Aragon, Kakit Wong, Isaac Yamnitsky

Required Materials

- A host computer
- The software from our github successfully installed onto the host PC (https://github.com/kitkatkandybar/RFSoC-Spectrum-Monitoring/tree/plots)
- Data in the DigitalRF format from the "OpenRadar2020" workshop (provided to us by our clients) saved on the host PC. Data available on: https://www.dropbox.com/sh/yd7zoav86qphizz/AAA-wGaPfqd2v3YMdCJlGcwPa/Data/day2/04_antennas?dl=0&subfolder_nav_tracking=1
- The "drf_plot.py" script that comes with the radioconda version of Anaconda (from https://github.com/ryanvolz/radioconda)

Set Up

Since the current state of our project is software-only, our setup is fairly simple. We have a web application we created, which can "replay" the contents of (locally-stored) Digital RF data. This "replaying" takes the format of two graphs - a spectrum graph and waterfall spectrogram plot. We will be testing the overall functionality of the web app, and comparing the graph outputs of our app to our "ground truth", which is static plots outputted by the drf_plot.py script. This script comes from the Digital RF library, so it is known to produce accurate plots for the Digital RF data format.

Pre-testing Setup Procedure

To create the static Digital RF plots:

- In a radioconda shell, run the commands found in the "addendum" section of reports, to produce the plots found in that section

To start our web application:

- In a separate shell, navigate to the location of the RFSoC-Spectrum-Monitoring repo, activate the 'rfsoc' Anaconda environment, and run 'python ./app.py' to start locally hosting the web app.
- In a browser, go to http://127.0.0.1:8050/ to open the web application

Testing procedure

Note: Since we're testing on Yana's computer, paths are specific to her PC and where she has data saved.

For each one of the data sources below, repeat the following steps:

- C:\Users\yanag\openradar\openradar_antennas_wb_hf\
- C:\Users\yanag\openradar\openradar_antennas_wb_vhf\
- C:\Users\yanag\openradar\openradar_antennas_nb_vhf\
- C:\Users\yanag\openradar\openradar_antennas_lband_radar\

Procedure:

1) Input the directory path of the data into the "load data" form on the web application and click the "load data" button
2) Set "Log Scale" to "On" and click the "Playback data from beginning" button
3) When the graphs have finished updating, set "Log Scale" "Off" and click the "Playback data from beginning" button again

Measurable Criteria

1) When each data source is loaded, metadata for that source should appear on the app dashboard
2) Upon hitting "Plackback data from the beginning", both the spectrum graph and the spectrogram graph should start updating with data smoothly until completion
3) The graphs should successfully toggle between a logarithmic scale and a non logarithmic scale
4) Graphs created on the web application should look similar to graphs created by drf_plot.py (found in the addendum) using the same data sources
    a) The overall shapes should look similar
    b) Noticeable peaks should occur at the same frequency and amplitude
    c) Range of x axis should be similar

Scoring

Antenna Source	Graph type	Updates smoothly after activating “playback data from the beginning”	Successfully toggle between logarithmic and non logarithmic scale	Compare reasonably to the graphs created by drf_plot.ply
openradar_antennas_wb_hf\	spectrum			
	spectrogram			
openradar_antennas_wb_vhf\	spectrum			
	spectrogram			
openradar_antennas_nb_vhf\	spectrum			
	spectrogram			
openradar_antennas_lband_radar\	spectrum			
	spectrogram			

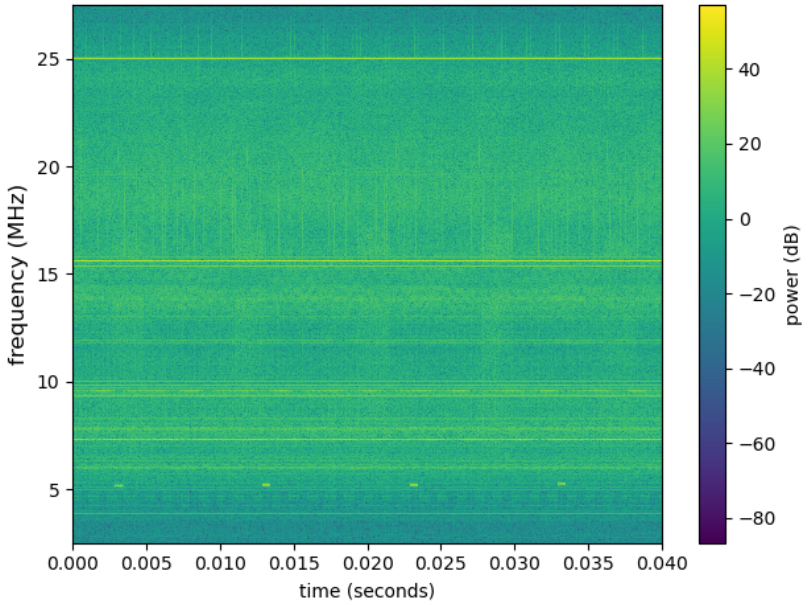
Addendum (Digital RF plots created using drf_plot.py)

Note: These plots were created on Yana’s computer and as such the paths are specific to her PC.

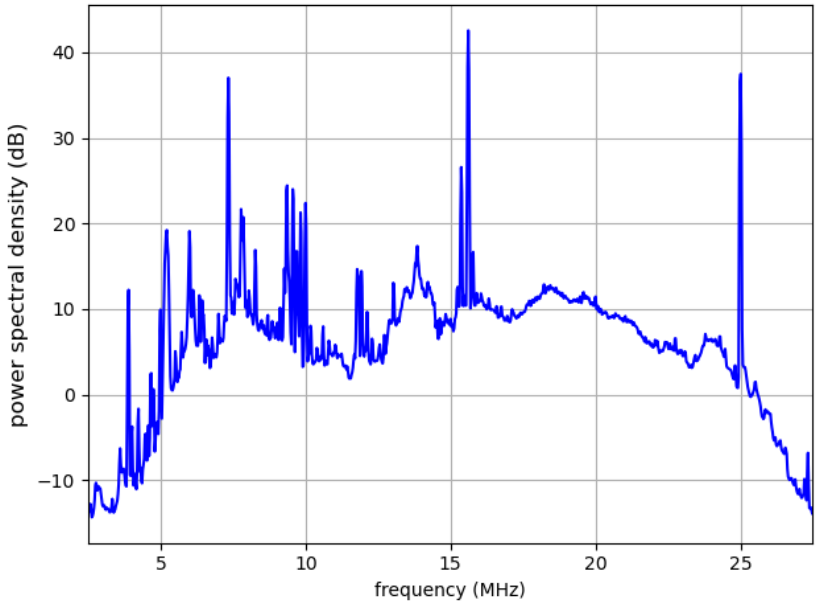
Command breakdown:
python C:\Users\yanag\radioconda\Scripts\drf_plot.py -i C:\Users\yanag\openradar\openradar_antennas_wb_hf\ -c discone:0 -r 0:1000000 -p specgram -b 1024 -l

- -i C:\Users\yanag\openradar\openradar_antennas_wb_hf\ : directory with the data.
- -c discone:0 : channel associated to a specific antenna with subchannel 0.
- -p specgram: plotting type (can be specgram or spectrum)
- -l: log scale on (remove it to have log scale off)

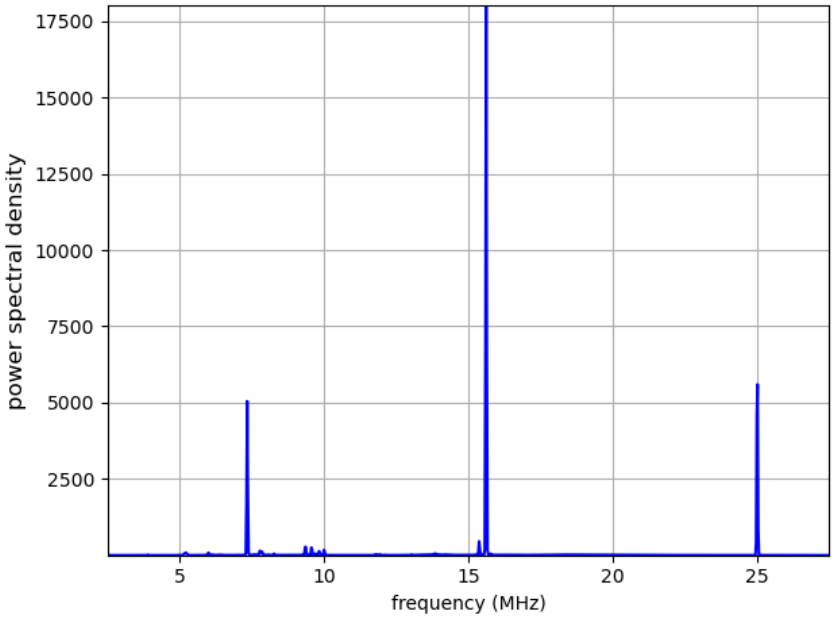
python C:\Users\yanag\radioconda\Scripts\drf_plot.py -i C:\Users\yanag\openradar\openradar_antennas_wb_hf\ -c discone:0 -r 0:1000000 -p specgram -b 1024 -l



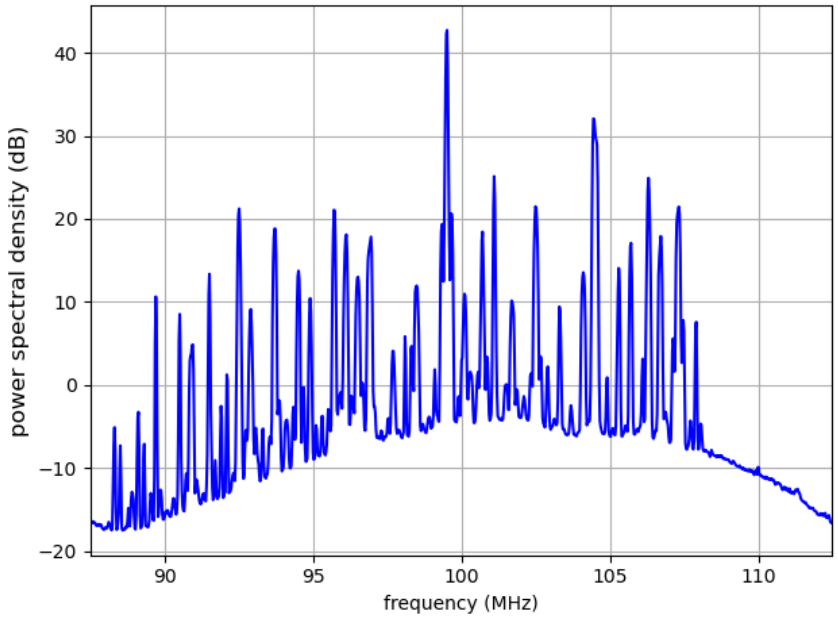
python C:\Users\yanag\radioconda\Scripts\drf_plot.py -i C:\Users\yanag\openradar\openradar_antennas_wb_hf\ -c discone:0 -r 0:1000000 -p spectrum -b 1024 -l



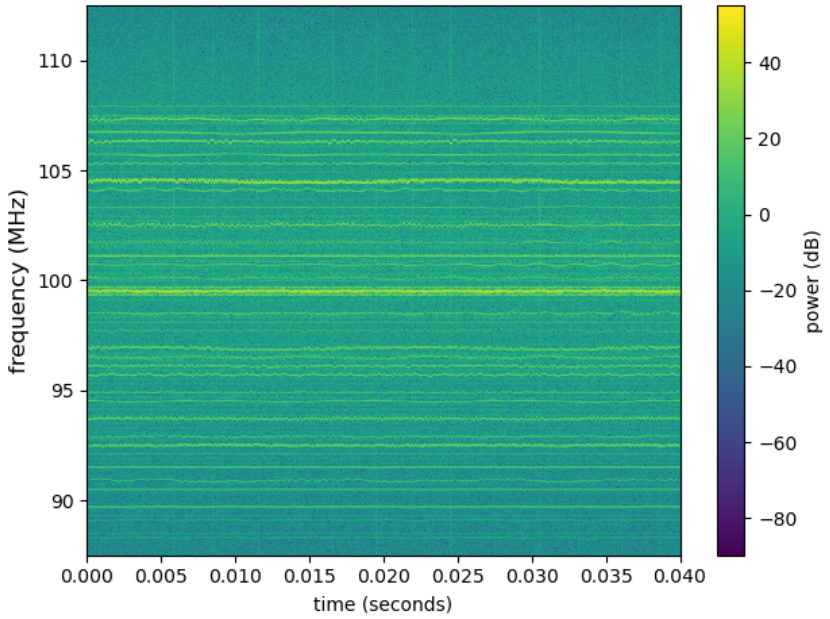
python C:\Users\yanag\radioconda\Scripts\drf_plot.py -i C:\Users\yanag\openradar\openradar_antennas_wb_hf\ -c discone:0 -r 0:1000000 -p spectrum -b 1024



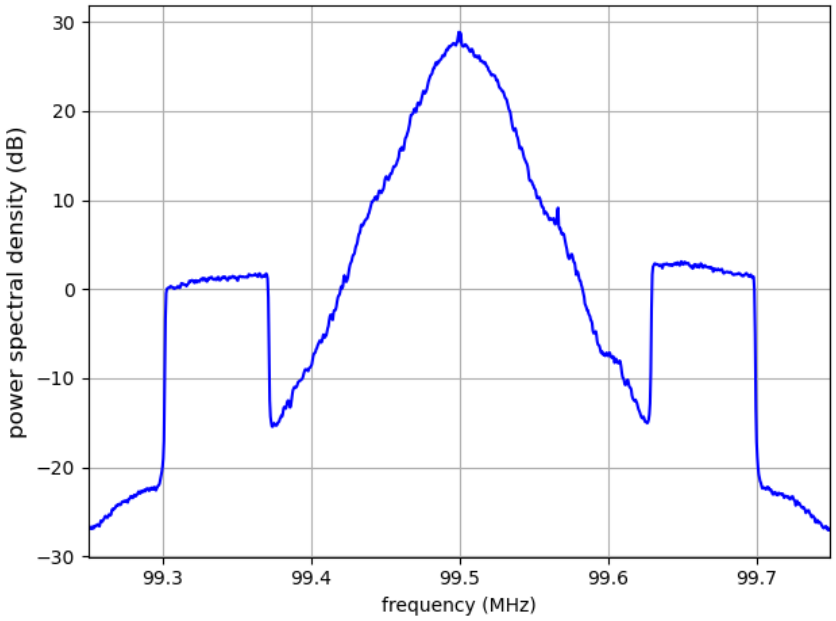
C:\Users\yanag\radioconda\Scripts\drf_plot.py -i C:\Users\yanag\openradar\openradar_antennas_wb_vhf -c discone:0 -r 0:1000000 -p spectrum -b 1024 -l



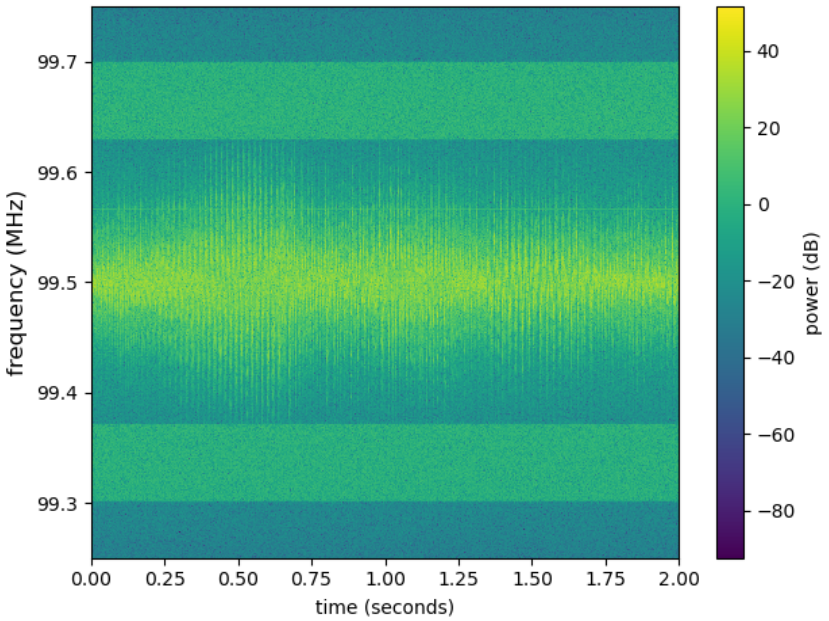
python C:\Users\yanag\radioconda\Scripts\drf_plot.py -i C:\Users\yanag\openradar\openradar_antennas_wb_vhf -c discone:0 -r 0:1000000 -p specgram -b 1024 -l



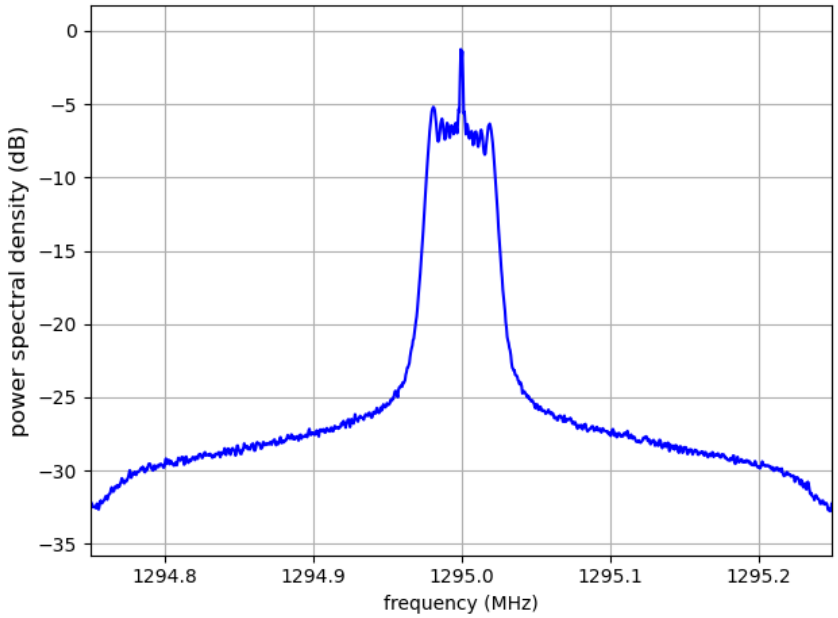
python C:\Users\yanag\radioconda\Scripts\drf_plot.py -i C:\Users\yanag\openradar\openradar_antennas_nb_vhf\ -c discone:0 -r 0:1000000 -p spectrum -b 1024 -l



python C:\Users\yanag\radioconda\Scripts\drf_plot.py -i C:\Users\yanag\openradar\openradar_antennas_nb_vhf\ -c discone:0 -r 0:1000000 -p specgram -b 1024 -l



python C:\Users\yanag\radioconda\Scripts\drf_plot.py -i C:\Users\yanag\openradar\openradar_antennas_lband_radar -c discone:0 -r 0:1000000 -p spectrum -b 1024 -l



C:\Users\yanag\openradar>python C:\Users\yanag\radioconda\Scripts\drf_plot.py -i C:\Users\yanag\openradar\openradar_antennas_lband_radar -c discone:0 -r 0:1000000 -p specgram -b 1024 -l