RFSoC Spectrum Monitoring Prototype Test Report

Yana Galina, Jaime Mohedano Aragon, Kakit Wong, Isaac Yamnitsky
Team 26
November 21, 2021

Overview

Our team has developed a web application which can display the contents of locally-stored data in the Digital RF format. Our prototype application visualizes stored data with two different graphs: the first is a spectrum graph which displays the power spectral density against frequency at a given time, and the second is a waterfall spectrogram plot, which displays power density as it varies over a period of time. These graphs are also interactive, specifically in that a logarithmic scale for the axes can be toggled on and off. During the test, we tested the overall functionality of our application, as well as the accuracy of the graphs it produced.

We tested the accuracy of our graphs by comparing them to our "ground truths", which were static plots outputted by the drf_plot.py script. The script originated from the Digital RF library, and is therefore known to produce accurate plots for the Digital RF data format.

Equipment and Setup

The only physical equipment that our prototype requires is a working host computer. To run the web app, Python 3.8, Anaconda, certain Python packages, and our software need to be installed. Detailed setup instructions can be found at our team Github repository at https://github.com/kitkatkandybar/RFSoC-Spectrum-Monitoring.

The data we used for our prototype was provided by our client John Swoboda, and was originally taken from the OpenRadar2020 workshop. This data needs to be saved locally onto the host computer in order for it to be displayed.

The last component of the setup is the "drf_plot.py" script, which is from the Digital RF library, found at https://github.com/MITHaystack/digital_rf. As mentioned before, this script produced static graphs against which we compared our graphs.

The first part of the setup was running the web app. The steps for this included:

1) In a command terminal, navigating to the location of the RFSoC-Spectrum-Monitoring repository.
2) Activating the 'rfsoc' Anaconda environment.
3) Running 'python ./app.py' to start locally hosting the web application.
4) Opening the web application in a browser by going to http://127.0.0.1:8050/.

The second part of the setup was obtaining the static graphs from drf_plot.py. This was done by navigating to the location of the script and running the following commands:

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

2) python drf_plot.py -i C:\Users\yanag\openradar\openradar_antennas_wb_hf\ -c discone:0 -r 0:1000000 -p spectrum -b 1024 -l
3) python drf_plot.py -i C:\Users\yanag\openradar\openradar_antennas_wb_hf\ -c discone:0 -r 0:1000000 -p spectrum -b 1024
4) python drf_plot.py -i C:\Users\yanag\openradar\openradar_antennas_wb_hf\ -c discone:0 -r 0:1000000 -p specgram -b 1024
5) python drf_plot.py -i C:\Users\yanag\openradar\openradar_antennas_wb_vhf -c discone:0 -r 0:1000000 -p spectrum -b 1024 -l
6) python drf_plot.py -i C:\Users\yanag\openradar\openradar_antennas_wb_vhf -c discone:0 -r 0:1000000 -p specgram -b 1024 -l
7) python drf_plot.py -i C:\Users\yanag\openradar\openradar_antennas_nb_vhf\ -c discone:0 -r 0:1000000 -p spectrum -b 1024 -l
8) python drf_plot.py -i C:\Users\yanag\openradar\openradar_antennas_nb_vhf\ -c discone:0 -r 0:1000000 -p specgram -b 1024 -l
9) python drf_plot.py -i C:\Users\yanag\openradar\openradar_antennas_lband_radar -c discone:0 -r 0:1000000 -p spectrum -b 1024 -l
10) python drf_plot.py -i C:\Users\yanag\openradar\openradar_antennas_lband_radar -c discone:0 -r 0:1000000 -p specgram -b 1024 -l

Detailed Measurements

Our main criteria for the graphs created by our web application is that they should look similar to the graphs created by drf_plot.py with the same parameters and the same data sources. By "look similar", we specifically mean that:

- The overall shapes should match
- Noticeable peaks should occur at the same frequency and amplitude
- The range of the x and the y axes should match

Some notes on the format of our graphs:

The screenshots of our spectrum graphs may have slightly different shapes or look more noisy than the corresponding drf_plot.py spectrum graph. This is because the drf_plot.py graphs show an average of the data over the entire time interval, while our graphs were screenshots taken at one instance of time during the playback of the data.

The screenshots of our spectrogram graphs have the axes flipped compared to the drf_plot.py spectrogram graphs. Our graphs have time on the vertical axis, while the drf_plot.py graphs have time on the horizontal axis.

All of the graphs have a logarithmic scale for the power, unless otherwise stated.

The comparison between the graphs are shown below. The graphs on the left of each figure correspond to our web app, while the graphs on the right correspond to the output of the Digital RF script drf_plot.py.

Wideband HF centered at 15 MHz With Logarithmic Power Scale:

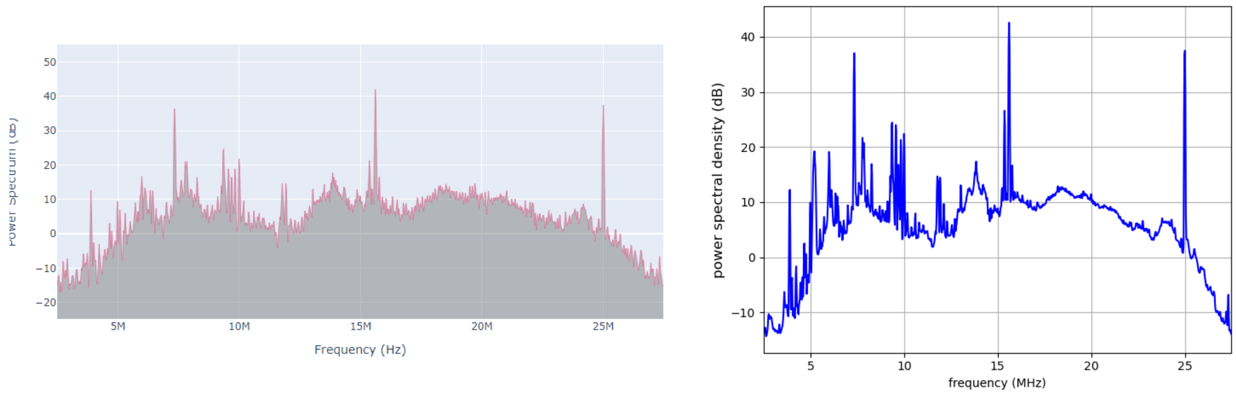

Figure 1: openradar_antennas_wb_hf spectrum graphs with logarithmic scale

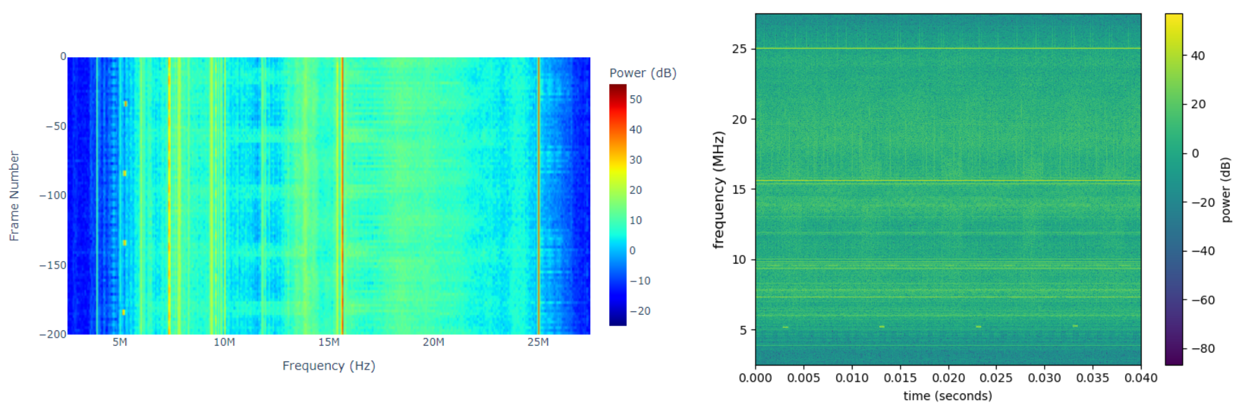

Figure 2: openradar_antennas_wb_hf spectrogram graphs with logarithmic scale

All four graphs show a central peak at around 15.6 MHz and an amplitude of about 42 dB. There are also two smaller peaks at 7.5 MHz and 25 MHz, having both an amplitude of around 38 dB. Both spectrogram plots show a periodic pulse at around 5MHz. All of the graphs have a frequency range going from around 2 MHz to 27 MHz.

Wideband HF centered at 15 MHz With Linear Power Scale:

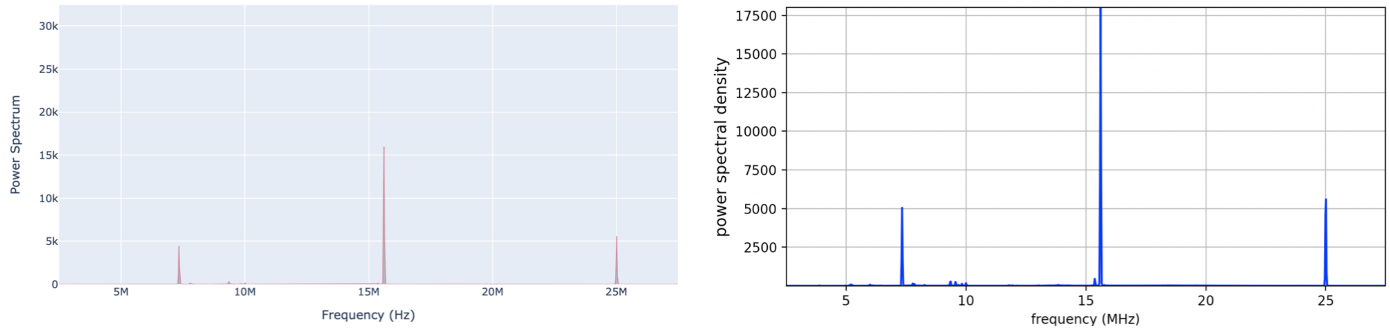

Figure 3: openradar_antennas_wb_hf spectrum graphs with linear scale

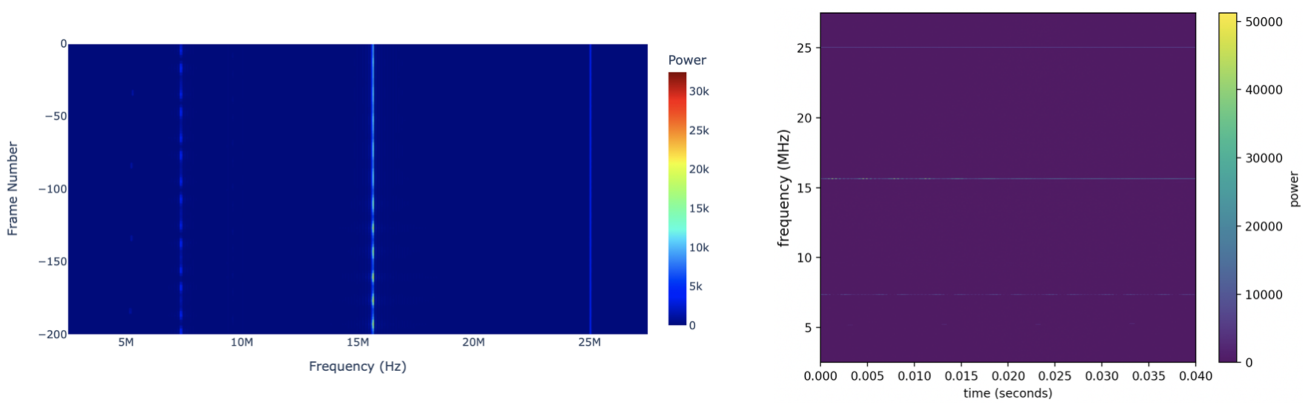

Figure 4: openradar_antennas_wb_hf spectrogram graphs with linear scale

One of the features of our application is the ability to toggle the logarithmic scale for the power density values on and off. These graphs show data with the logarithmic scale turned off, unlike all our other examples. All four graphs show a central peak at around 15.6 MHz and an amplitude ranging from 15-17k over the course of the time period covered by this data. There are also two smaller peaks at 7.5 MHz and 25 MHz, having both an amplitude of around 5k power spectral density. All of the graphs have a frequency range going from around 2 MHz to 27 MHz.

Wideband VHF centered at 100 MHz.

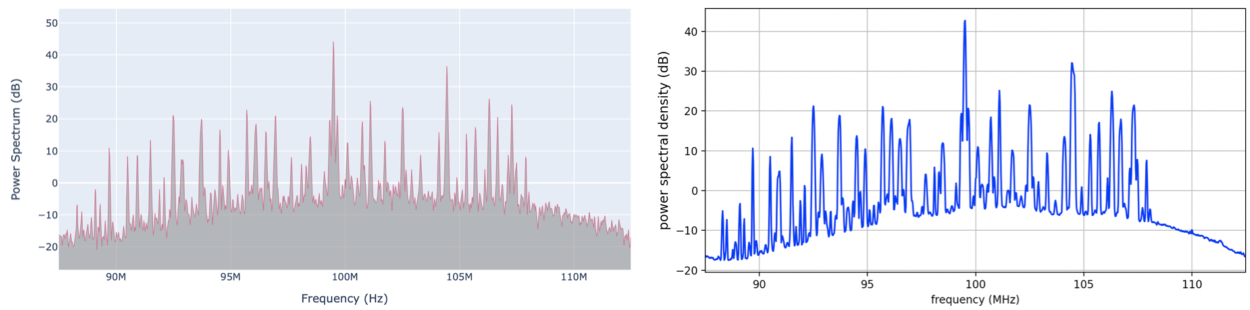

Figure 5: openradar_antennas_wb_vhf spectrum graphs

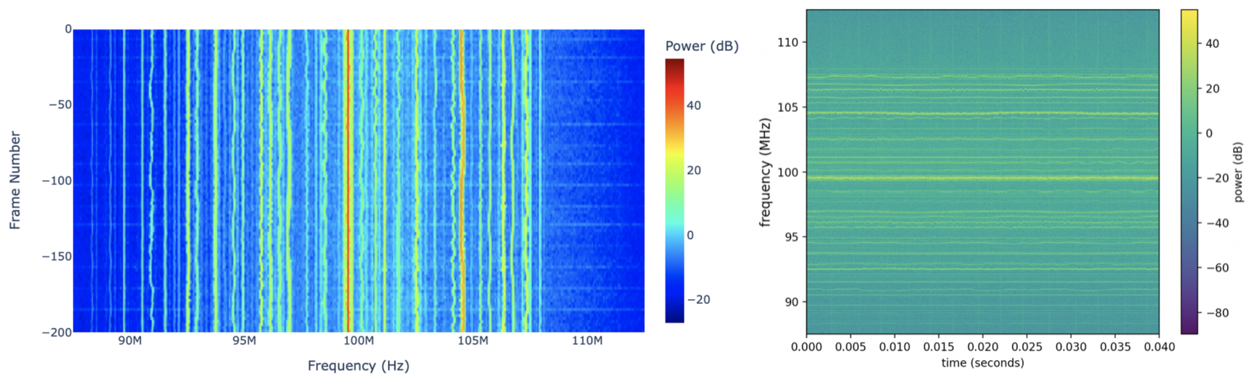

Figure 6: openradar_antennas_wb_vhf spectrogram graphs

The largest peak can be identified at about 99 MHz with an amplitude of around 42 dB on all four graphs. The shape of the various peaks math between the web application and the drf_plot versions. All of the graphs have a frequency range going from around 85 MHz to 115 MHz.

<u>Narrowband VHF centered at 99.5 MHz:</u>

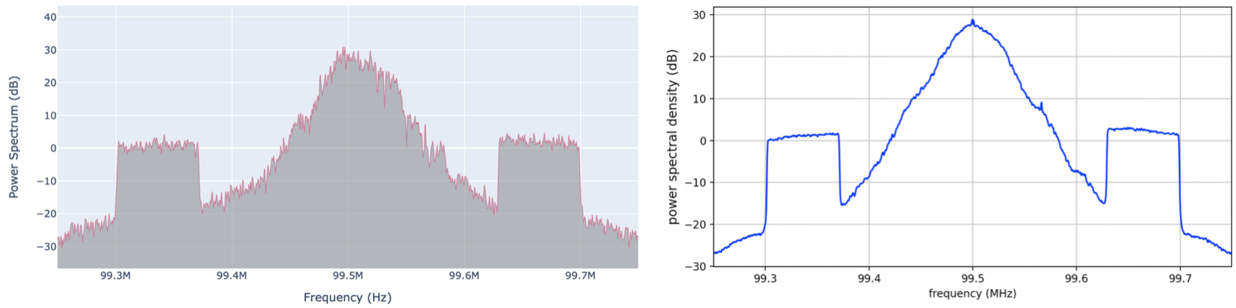

Figure 7: openradar_antennas_nb_vhf spectrum graphs

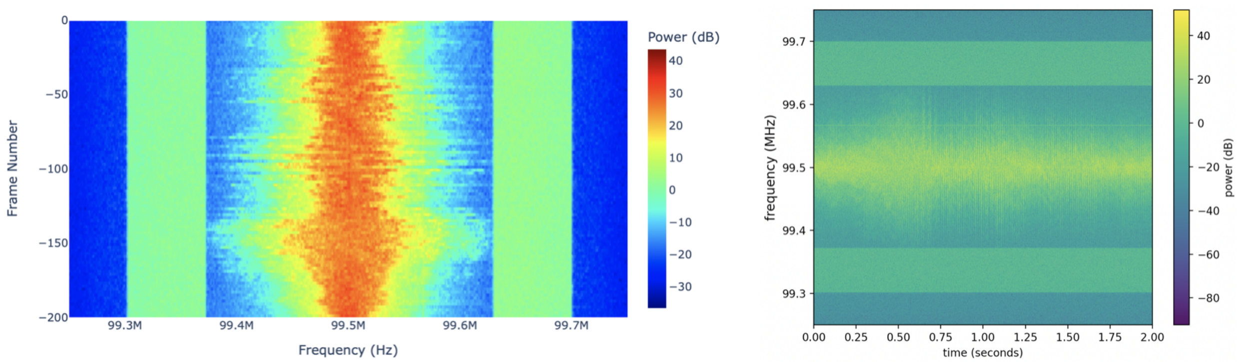

Figure 8: openradar_antennas_nb_vhf spectrogram graphs

The spectrum graphs both show a center frequency at 99.5 MHz and a power density of around 30 dB. There are two symmetric bands surrounding the center, going (roughly) from 99.3GHz to 99.375GHz and from 99.625GHz to 99.7GHz. They both have a power density of around 0 dB.

In the spectrogram graphs, you can also clearly distinguish the same center peak at 30 dB with fluctuating bandwidth and the two sidebands at 0 dB in both versions. All of the graphs have a frequency range going from around 99.25 MHz to 99.75 MHz.

<u>L-band centered at 1295 MHz</u>

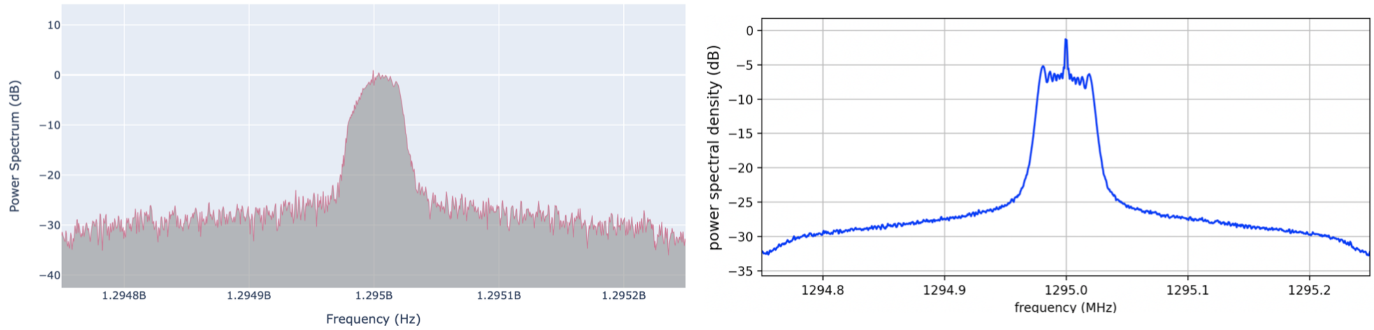

Figure 9: openradar_antennas_lband_radar spectrum graphs

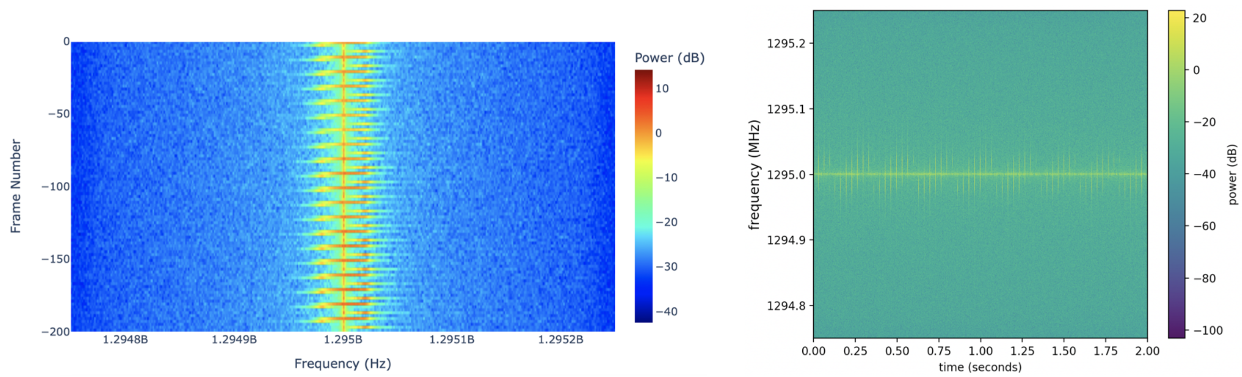

Figure 10: openradar_antennas_lband_radar specgram graphs

The spectrum graphs seem to look different at first glance, but this discrepancy is just an artifact of the data. The peak shifts rapidly, and as we mentioned earlier in the report, the drf_plot.py graphs show an average of the data, while our graphs are screenshots taken at one particular instance.

The central peak can be identified in all four graphs at 1.295 GHz and a power density of around 0 dB. Both spectrogram graphs display the shifting pulses. All of the graphs have a frequency range going from around 1.2945 GHz to 1.29525 GHz.

Conclusions

We have created a successful proof-of-concept application which visualizes Digital RF data in a clear and accurate manner. Our graphs display accurate axes and data compared to the ground truth, with similar shapes and peaks occurring at the same frequencies and amplitudes. During the live demo, we also demonstrated that our web application runs smoothly, with no major lagging or bugs. Our current prototype is a solid foundation for future improvements and additions. Going forward, we will extend our web application to include more user interaction and a more visually appealing UI.

Our ultimate goal for this project is to be able to visualize and process data from an RFSoC board. Up until now, we did not have access to one of these boards, which is why we focused on displaying stored data in the DigitalRF format for this prototype. However, our clients have recently obtained an RFSoC board that we will have remote access into. Therefore, our next major step will be to add live RFSoC data as an input to our web application.