Development of Back-End Software for the Grape 2

Cuong Nguyen KC3UAX[1], John Gibbons N8OBJ[2, 3], William Blackwell Jr. AB1XB[3], Nathaniel Frissell W2NAF[1]

[1]University of Scranton, [2]Case Western Reserve University, [3]HamSCI Community

Abstract

The Grape2 system is a three-channel radio receiver and a part of the larger Personal Space Weather Station project. This poster showcases several software tools developed to support the development and operation of the Grape 2 system. G2console is a terminal-based interface that communicates with the data collection system, providing users with valuable information such as software versions, amplitude, frequency, GPS, and magnetometer metrics for viewing and diagnostics. GrapeSpectrogram is a data processing script that generates Dopplergrams, aiding developers in validating the system's operation. Additionally, we will discuss future project developments, such as integration with the Linux GPS background service (gpsd) to provide accurate timing to the Raspberry Pi, and DigitalRF as a more efficient method of data storage.

Architecture Overview

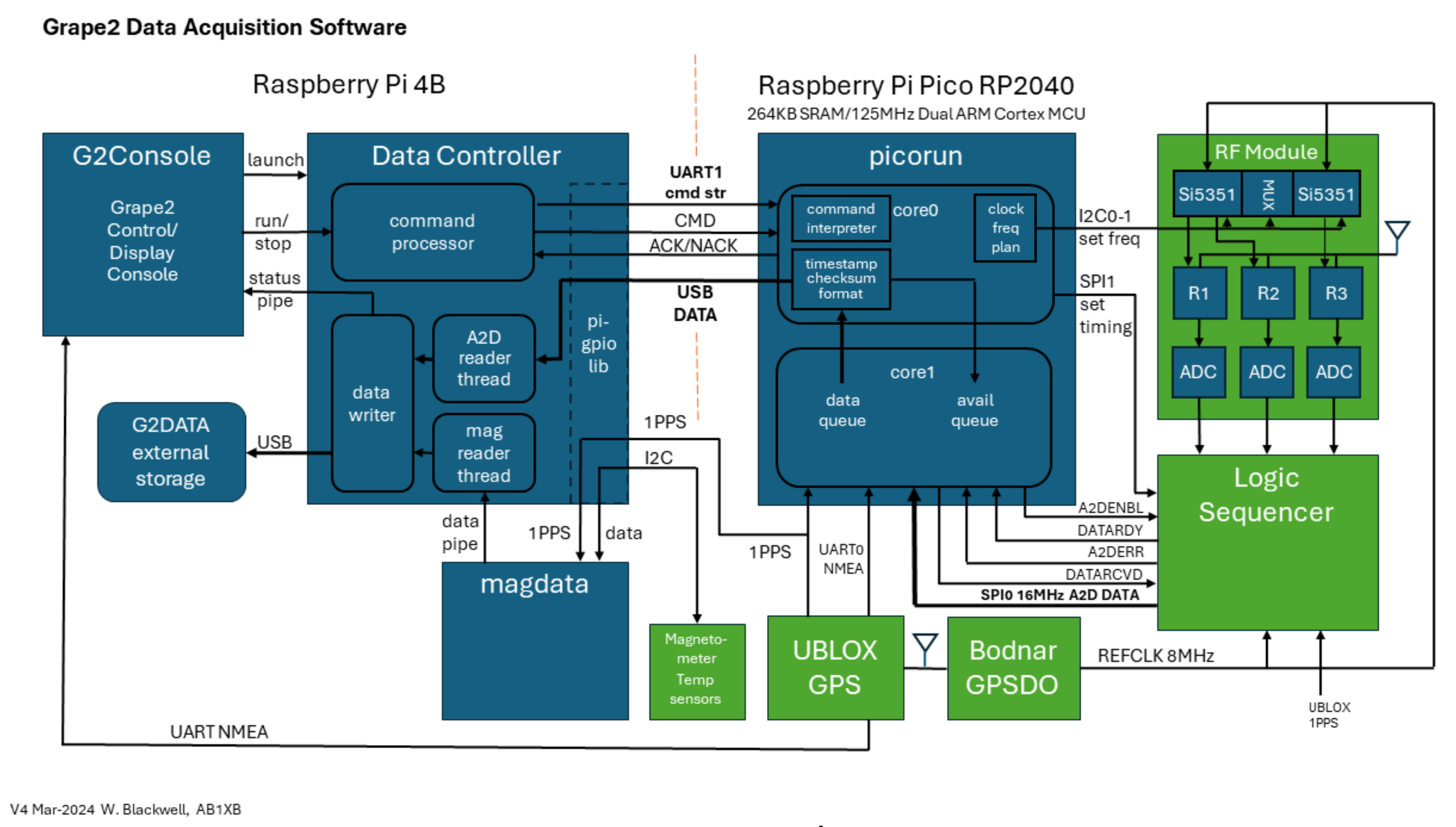


Figure 1. Grape2 Architecture

- The G2console program serves as the user interface for controlling and monitoring the Grape2 system's operations. It operates with three parallel threads:
  - GPS reader: communicates with the UBLOX GPS module to retrieve real-time GPS information.
  - Data reader: communicates with the data controller to retrieve data from the magnetometer and radio.
  - Main thread: is responsible for displaying the data obtained by the other two threads and handling user inputs.
- Built using the Python library curses, this software provides a terminal-based user interface, making it extremely lightweight. This design choice helps conserve computing power, especially for critical tasks such as data collection and writing.

Control/Display Console

Real-time Information

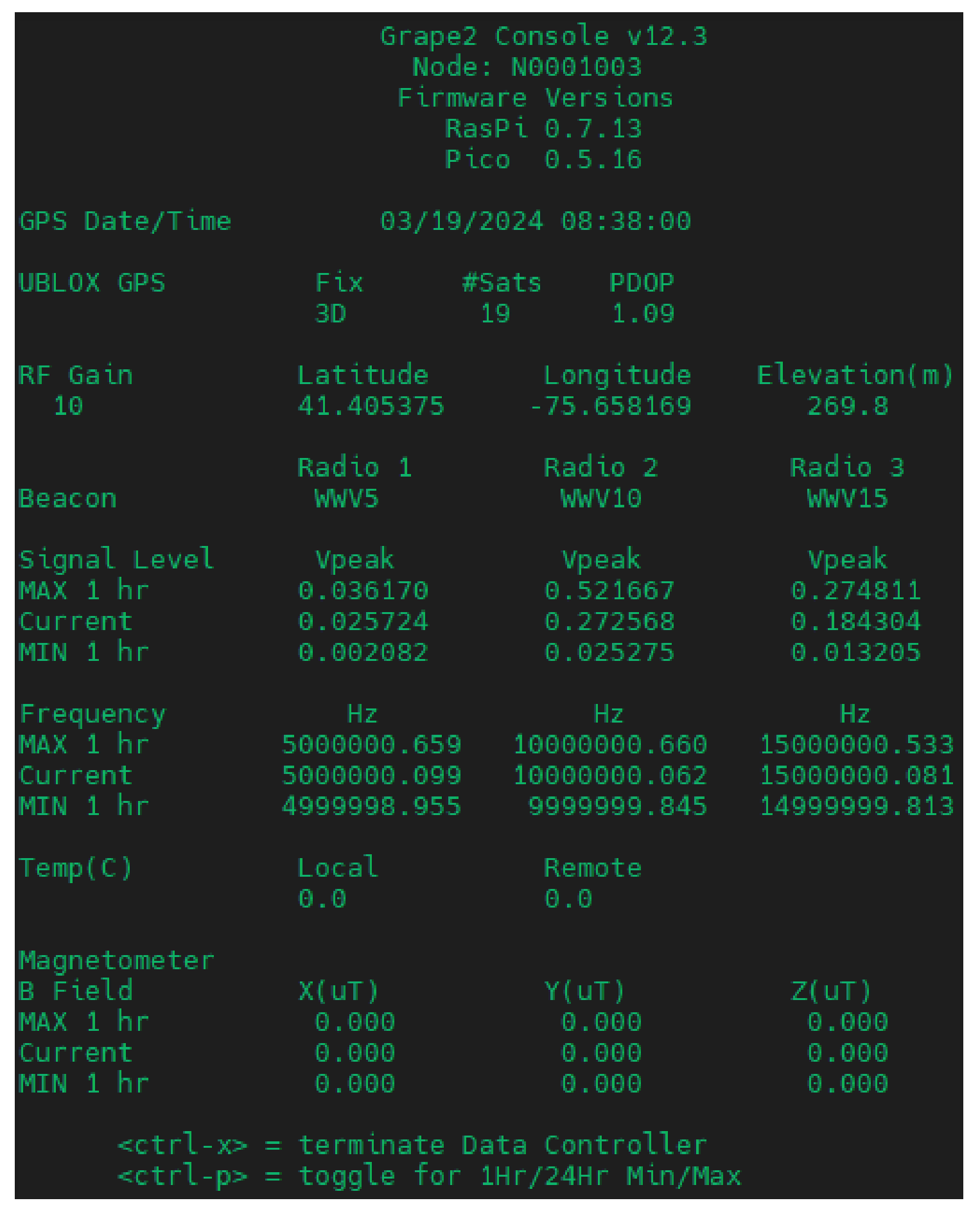
- Firmware/software versions
- Time
- GPS properties
- Radio amplitudes and frequencies
- Temperatures
- Magnetometer outputs

Control

- Initiating and terminating the data collection system
- Switching between displaying aggregation of data within the last 1 hour or 24 hours

Error Notifications

- HDD failures
- Data reading/writing failures



```
                      Grape2 Console v12.3
                        Node: N0001003
                       Firmware Versions
                         RasPi 0.7.13
                         Pico  0.5.16

GPS Date/Time         03/19/2024 08:38:00

UBLOX GPS          Fix       #Sats     PDOP
                   3D         19       1.09

RF Gain           Latitude      Longitude    Elevation(m)
  10              41.405375     -75.658169     269.8

                  Radio 1       Radio 2       Radio 3
Beacon             WWV5          WWV10         WWV15

Signal Level       Vpeak         Vpeak         Vpeak
MAX 1 hr          0.036170      0.521667      0.274811
Current           0.025724      0.272568      0.184304
MIN 1 hr          0.002082      0.025275      0.013205

Frequency            Hz            Hz            Hz
MAX 1 hr         5000000.659  10000000.660  15000000.533
Current          5000000.099  10000000.062  15000000.081
MIN 1 hr         4999998.955   9999999.845  14999999.813

Temp(C)           Local         Remote
                  0.0           0.0

Magnetometer
B Field           X(uT)         Y(uT)         Z(uT)
MAX 1 hr           0.000         0.000         0.000
Current            0.000         0.000         0.000
MIN 1 hr           0.000         0.000         0.000

      <ctrl-x> = terminate Data Controller
      <ctrl-p> = toggle for 1Hr/24Hr Min/Max
```

Figure 2. G2console

GrapeSpectrogram

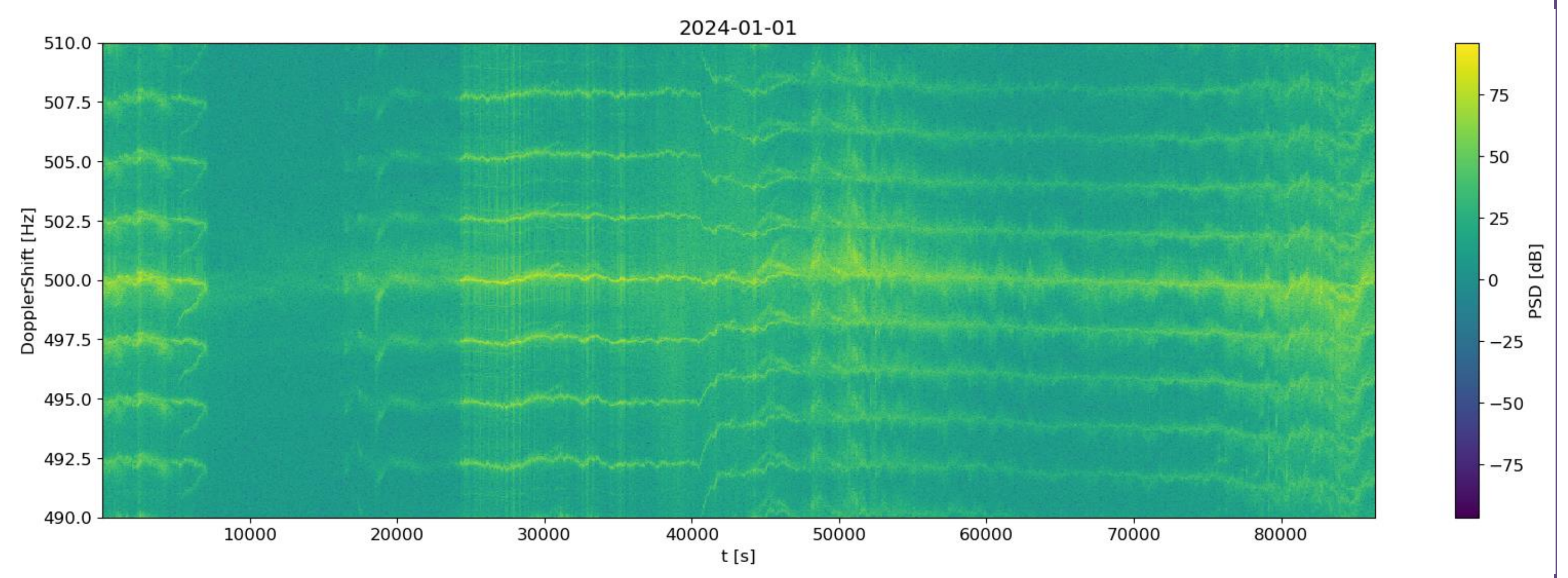


Figure 3. PSD Spectrogram for Channel 1 on 2024-01-01

- The power spectral density spectrogram presented above vividly illustrates the presence of the Doppler shift, a crucial phenomenon in our system.
- This graphical representation is invaluable as a visual validation of the system's functionality and accuracy.
- The graph was created using a Python notebook leveraging processing libraries such as SciPy and NumPy. Due to the computational intensity of the task, the script was run on an external Linux computer to ensure optimal performance and resource management.

Future Work

GPSD

- The current system operates in a semi-realtime manner. Although each data point is timestamped with information directly from the GPS module, the operating system's clock synchronizes its time via the Internet.
- The GPS daemon is a Linux background service that acts as a bridge between the GPS module and the Raspian OS, enabling the system to achieve true realtime functionality.

DigitalRF

- Our current system stores raw data in CSV files, which are often considered inefficient for this purpose due to their tendency to consume excess storage and present readability challenges for computers.
- Developed by MIT Haystack Observatory, DigitalRF (DRF) is an HDF5 file format specifically designed for reading and writing radio frequency data. Our preliminary conversion to the DRF format has shown that it can reduce storage requirements by up to 4 times.
- Furthermore, our central database already includes DRF data analysis programs for post-processing, such as automatically generating the spectrograms.

G2console Rework

- Over time, the G2console program has evolved to become the primary user interface. Initially, it was intended as a quick-and-dirty method to provide a live feed of system operations, primarily for diagnostic purposes. Consequently, the foundational code was not designed with maintainability in mind. A rework of the program using software engineering principles would undoubtedly benefit the future of the Grape2 system.

References

digital_rf. *MIT Haystack Observatory*. https://github.com/MITHaystack/digital_rf.git

HDF5. https://www.hdfgroup.org/solutions/hdf5/

Acknowledgements

The authors acknowledge NIST, NRC, and their members for the operation of the WWV/WWVH/CHU stations. The authors also acknowledge William Engelke AB4EJ for his advices on DRF, and Jonathan Rizzo KC3EEY for his inputs on GPSD. Computations and plotting were done using JupyterLab and open-source Python tools including but not up to SciPy, NumPy, Pandas, matplotlib, curses. We also acknowledge the support of NSF grants AGS-2230345, AGS-2230346, and AGS-2002278.