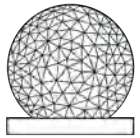
Haystack AeroVista REU

Hirani Sattenapalli, Aparna Rajesh, T. Lucas Briggs



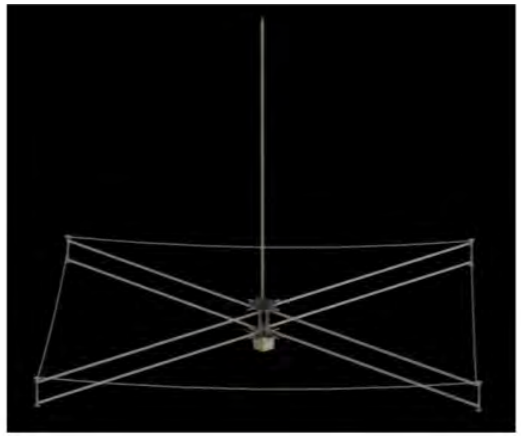
MIT
HAYSTACK
OBSERVATORY

“Aurora Touching Sunrise” from NASA archives

MIT HAYSTACK OBSERVATORY

AERO VISTA Mission Introduction

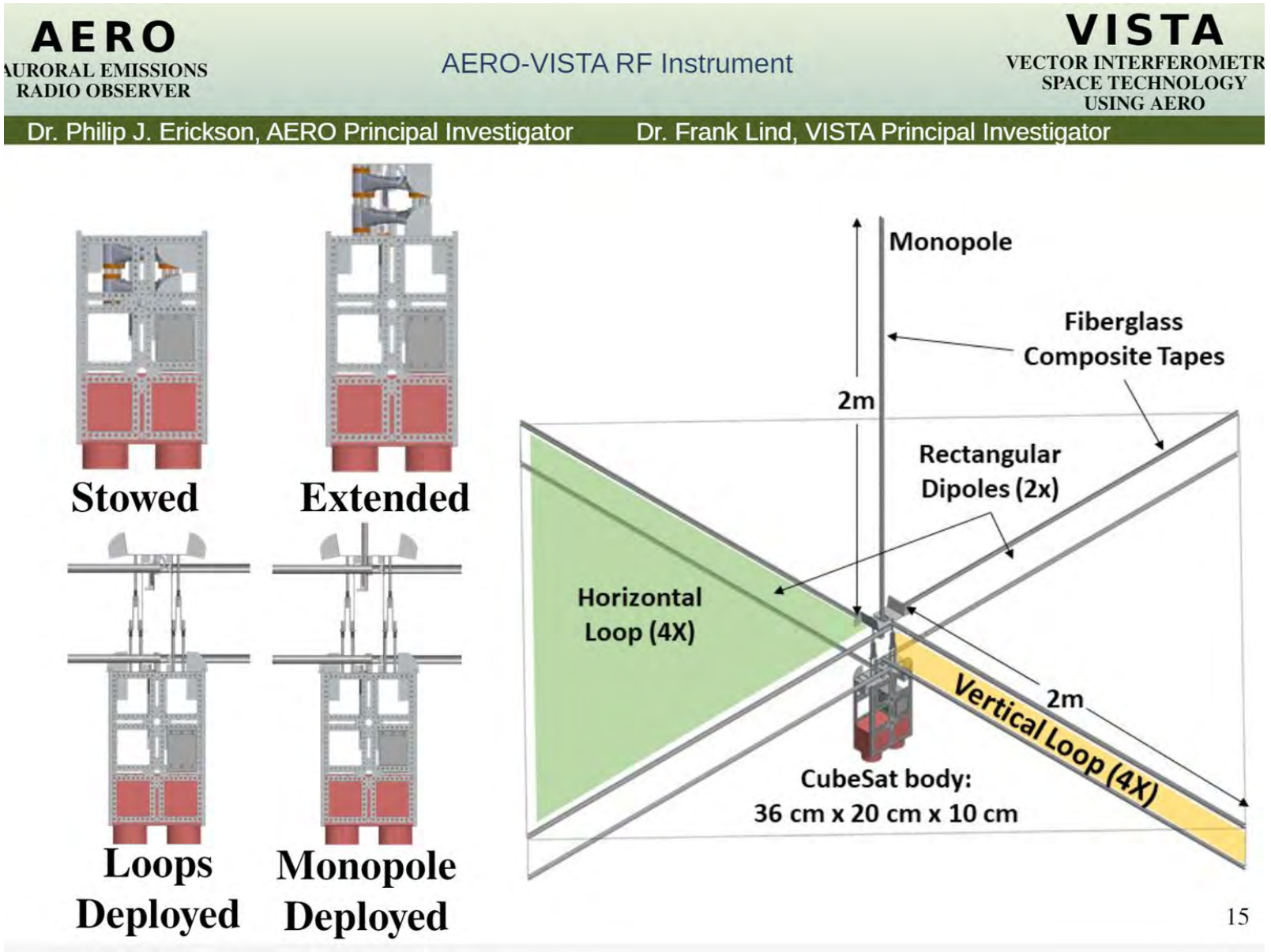
AERO VISTA Payload ←



Antenna after Deployment →

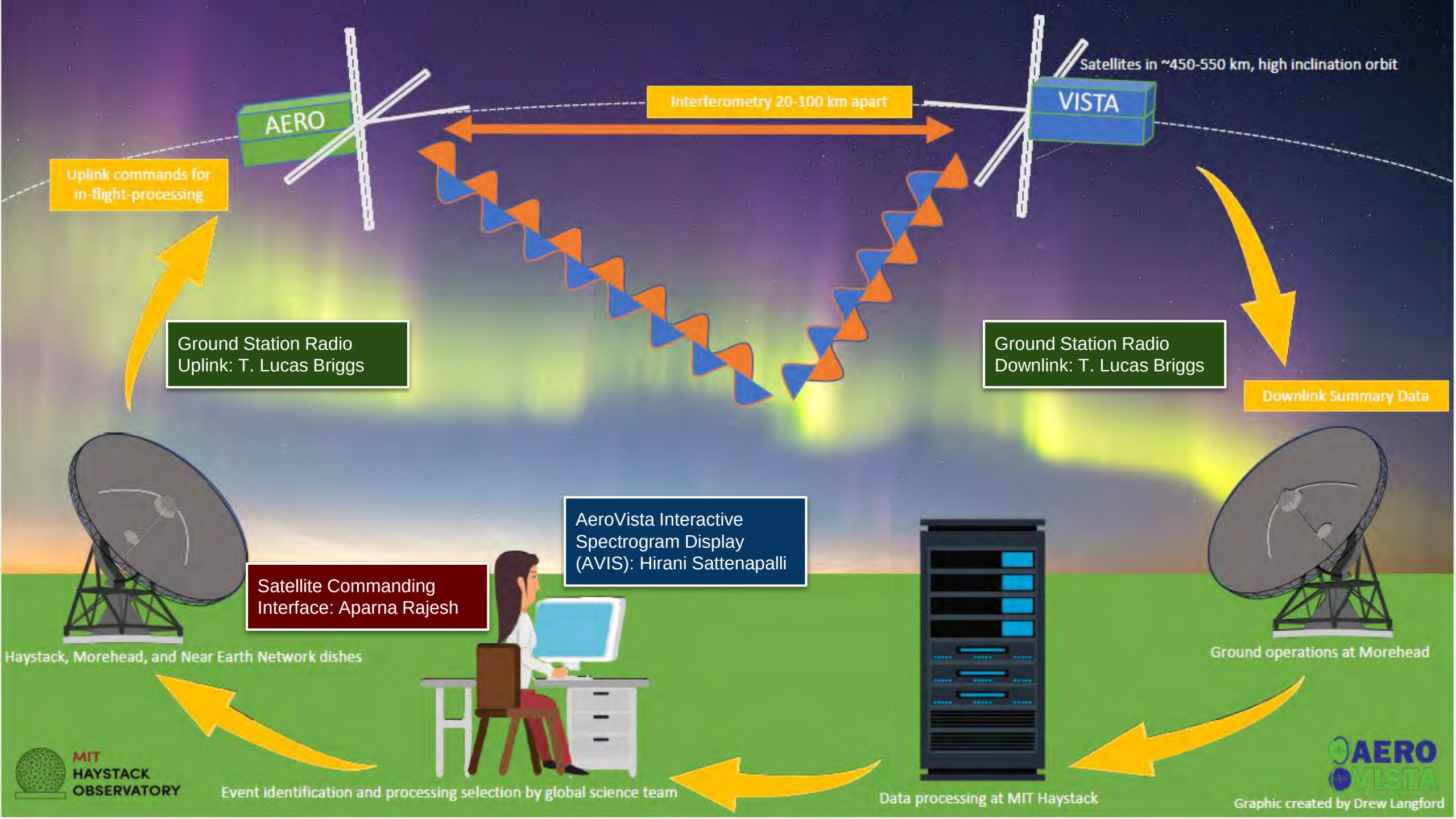
- AERO & VISTA satellites will collect radio frequency (RF) data from the auroral regions
- Data will be used to accomplish science and tech goals
  - Study emissions such as Auroral Kilometric Radiation (AKR), Medium Frequency Burst (MFB), Auroral Roar, and Auroral Hiss
  - Validate usage of Vector Sensor Interferometry and RFI survey

MIT HAYSTACK OBSERVATORY



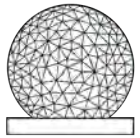
Monopole, Horizontal Loop, and Rectangular Dipoles correspond to channels on spectrogram

Slide from Mary Knapp AERO-VISTA presentation 2021


AERO
VISTA
Satellites in ~450-550 km, high inclination orbit
Interferometry 20-100 km apart
Uplink commands for in-flight-processing
Ground Station Radio Uplink: T. Lucas Briggs
Ground Station Radio Downlink: T. Lucas Briggs
Downlink Summary Data
AeroVista Interactive Spectrogram Display (AVIS): Hirani Sattenapalli
Satellite Commanding Interface: Aparna Rajesh
Haystack, Morehead, and Near Earth Network dishes
Ground operations at Morehead
MIT HAYSTACK OBSERVATORY
Event identification and processing selection by global science team
Data processing at MIT Haystack
AERO VISTA
Graphic created by Drew Langford

AERO-VISTA Interactive Spectrogram Display

Hirani Sattenapalli



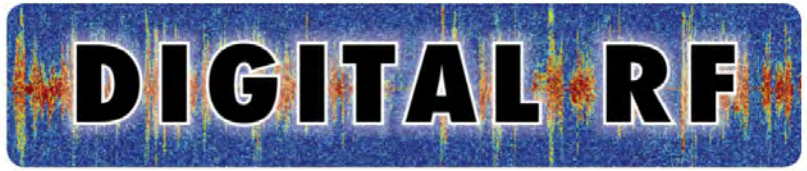
MIT
HAYSTACK
OBSERVATORY

AVIS Display & Objectives

- Provide a tool for the science team to visualize metadata
- Present spectrogram data in plotly graphs
- Allow science team to perform computation on channel data and send commands for in-flight processing

Libraries Used

redis

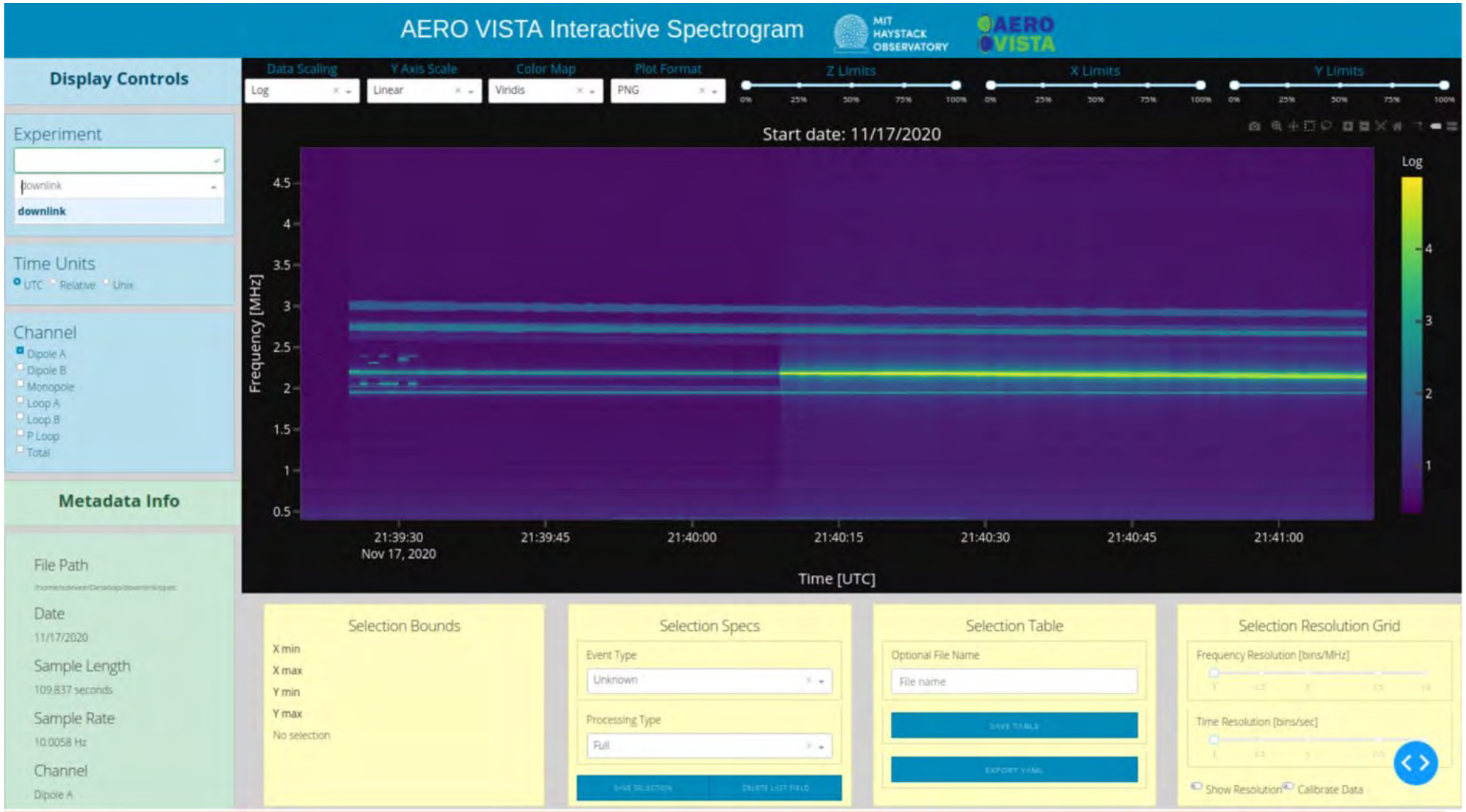


- Plotly
  - Graphing utility used for telemetry maps and spectrogram plots
- Redis
  - In-memory data structure used to store metadata
- Digital RF
  - Software used for reading and writing spectrogram metadata into digital RF format

Libraries Used

- Dash
  - Python framework to build web pages
  - Used to build and style layout and components of the dashboard
- Numpy
  - python library to work with arrays and matrices
- Xarray
  - python package that adds dimensions and coordinates to numpy arrays
  - used to organize metadata to place into redis

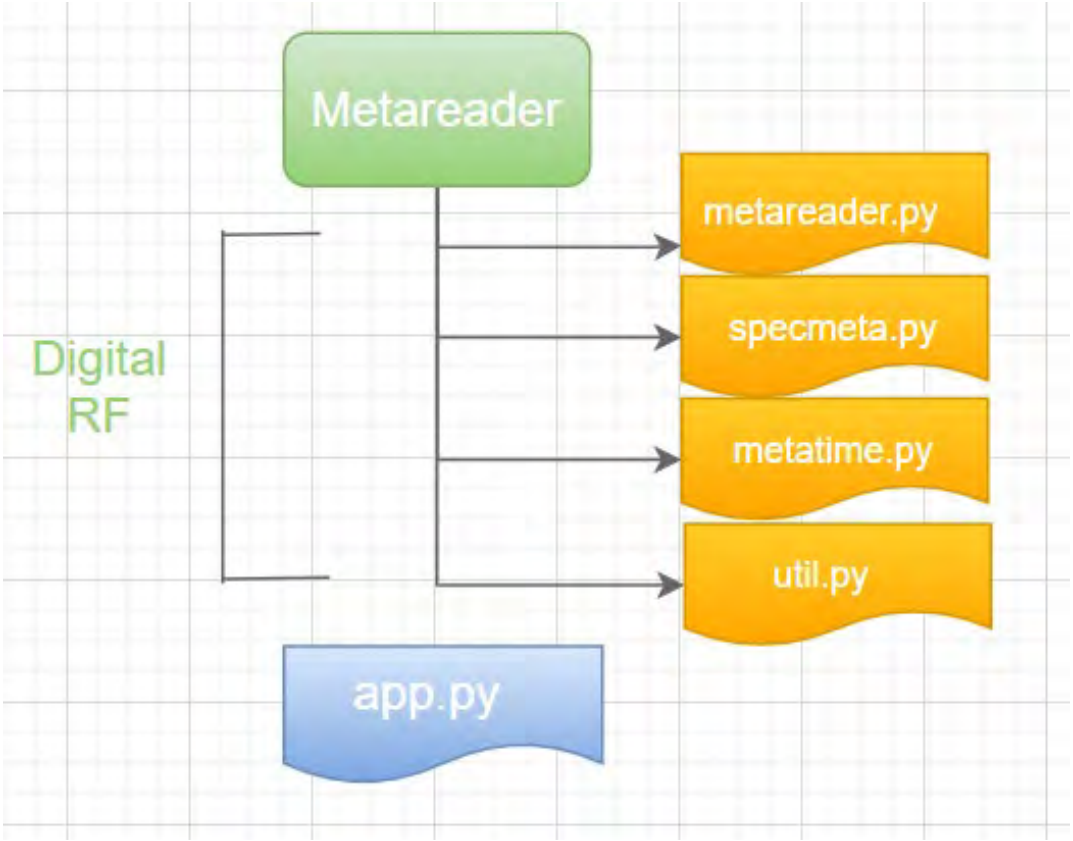
Existing AVIS Display



Goals for Dashboard Version 2.0

- Faster Loading of Data
  - Updating the spectrogram by retrieving and processing the summary data files presents a high computational load
- Display of Telemetry Data
  - Spacecraft speed, location, and altitude
  - Used to provide context for science team
- Generation of Subplots to do computation between channels
- Overall Design Changes to increase visual & user interactibility

File Structure

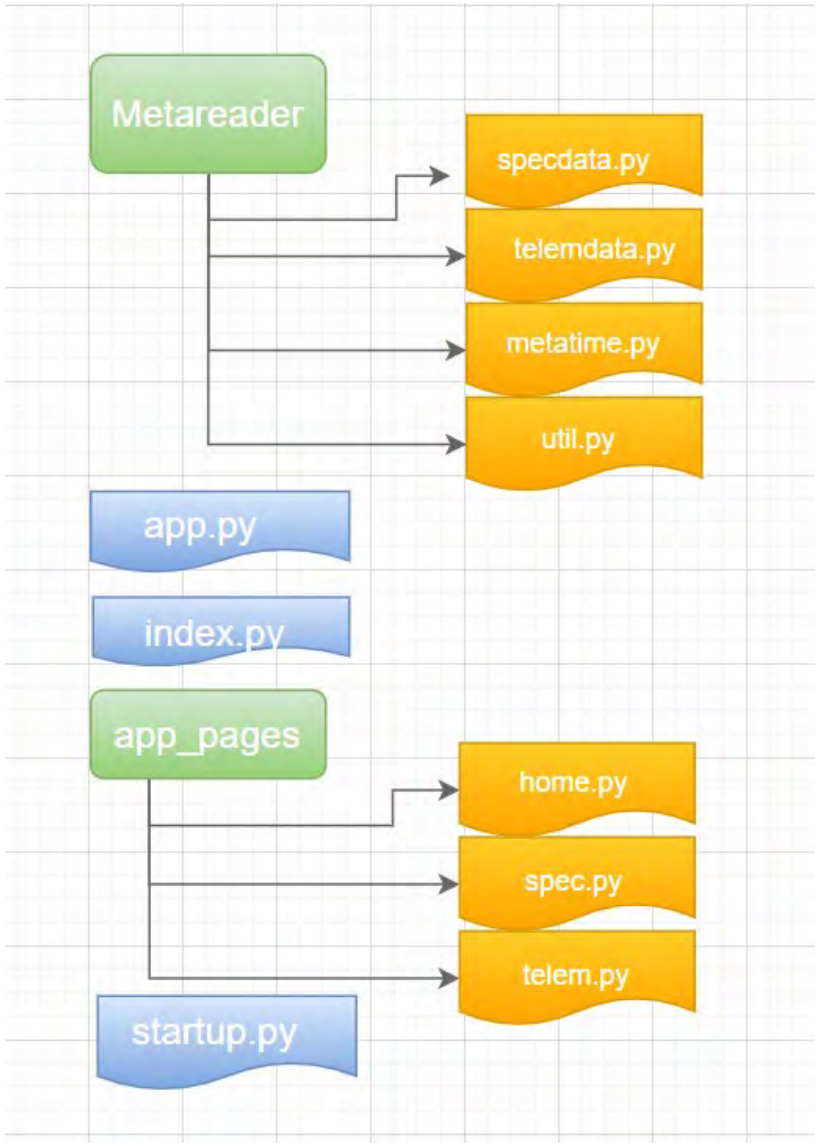


AVIS Version 1

- Metareader used to read summary data file
- Metatime provides timestamp data
- Specmeta creates spectrogram plot
- Util - creates dash components for the display

MIT
HAYSTACK
OBSERVATORY

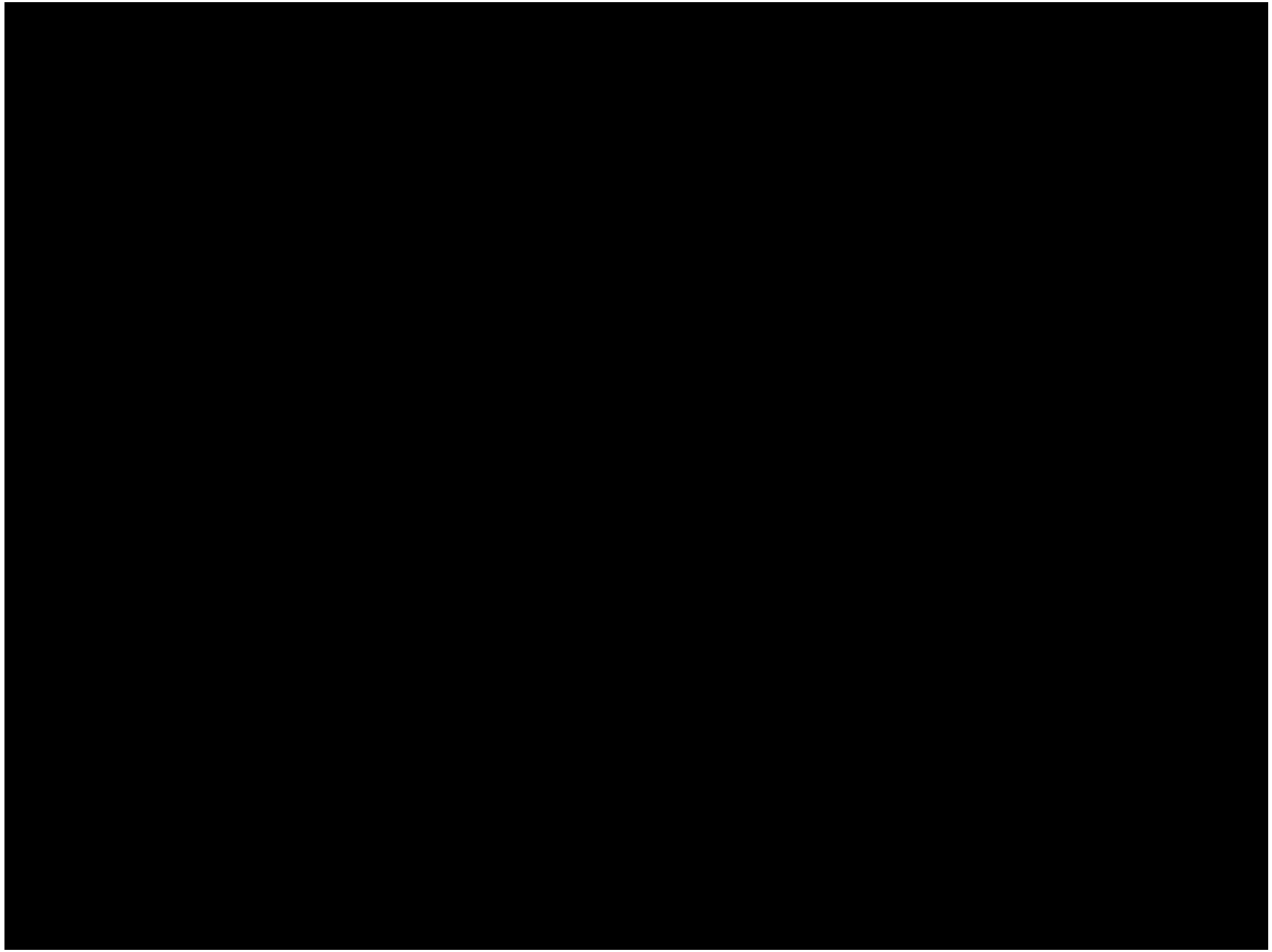
File Structure



AVIS Version 2.0

- Specmeta and Metareader files replaced with Specdata file
- Specdata:
  - used to enter summary data into redis for in-memory storage
- App.py files
  - Split into index.py, app.py, & app pages to accommodate multi-page dash app
  - Easier for future additions

Demo



MIT
HAYSTACK
OBSERVATORY

Summary of New Layout & Results

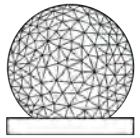
- App.py structure & index.py
  - Allows for easy addition of future improvements
- Redis interface & backend structure
  - Enters all spectrogram and time data into redis
  - Meant for future collaboration between Lucas and Aparna's work
  - Access files in redis and send uplink files into redis
- Speed
  - In memory storage of data did not speed up spectrogram plot generation as desired
  - Data input to redis makes it easier for data to be accessed and exported in the future

Summary of New Layout & Results Cont.

- Telemetry data page
  - Provides a data table of satellite speed, position/ location, & altitude
  - Provides context for science team when analyzing spectrogram data
- Subplots page
  - Continuous regeneration of spectrogram plots to use for computation between different channels (ex. sum(loops), mult(dipoles), division, linear combinations)
  - Used to classify if data is showing electrostatic or electromagnetic phenomena & type of emission

Future Work

- Computation between channels
  - Currently there is subplot generation; computation for channel math needs to be developed
- Improvements on redis structure
  - Data organization in redis and improvements on file access
- Satellite video/ display in home page
  - Future satellite data to be presented on the home page



MIT
HAYSTACK
OBSERVATORY

Citations

Erickson, et al. “AERO: Auroral Emissions Radio Observer”. *In: Conference on Small Satellites*. Aug. 2018. url: https : / / digitalcommons . usu . edu / smallsat/2018/all2018/453

Knapp, Mary. “AERO-VISTA and Low Frequency Radio Astronomy.” Haystack Observatory, July 2021.

Langford, Drew. (2020). “The AERO-VISTA Interactive Spectrogram Display: An Original Software Solution for Scientific Operations of Twin 6U CubeSats”

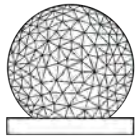
Frank Lind, et al. “AERO & VISTA: Demonstrating HF Radio Interferometry with Vector Sensors”. In: *Small Satellite Conference, Upcoming Missions, SSC19-WKV-09*. Aug.2019.

Volz, R., Rideout, W. C., Swoboda, J., Vierinen, J. P., & Lind, F. D. (2021). Digital RF (Version 2.6.6). MIT Haystack Observatory. Retrieved from https://github.com/MITHaystack/digital_rf

Zell, Holly. “Aurora Video GALLERY.” *NASA*, NASA, 10 Apr. 2015, www.nasa.gov/mission_pages/sunearth/aurora-videos/index.html.

Satellite Commanding API

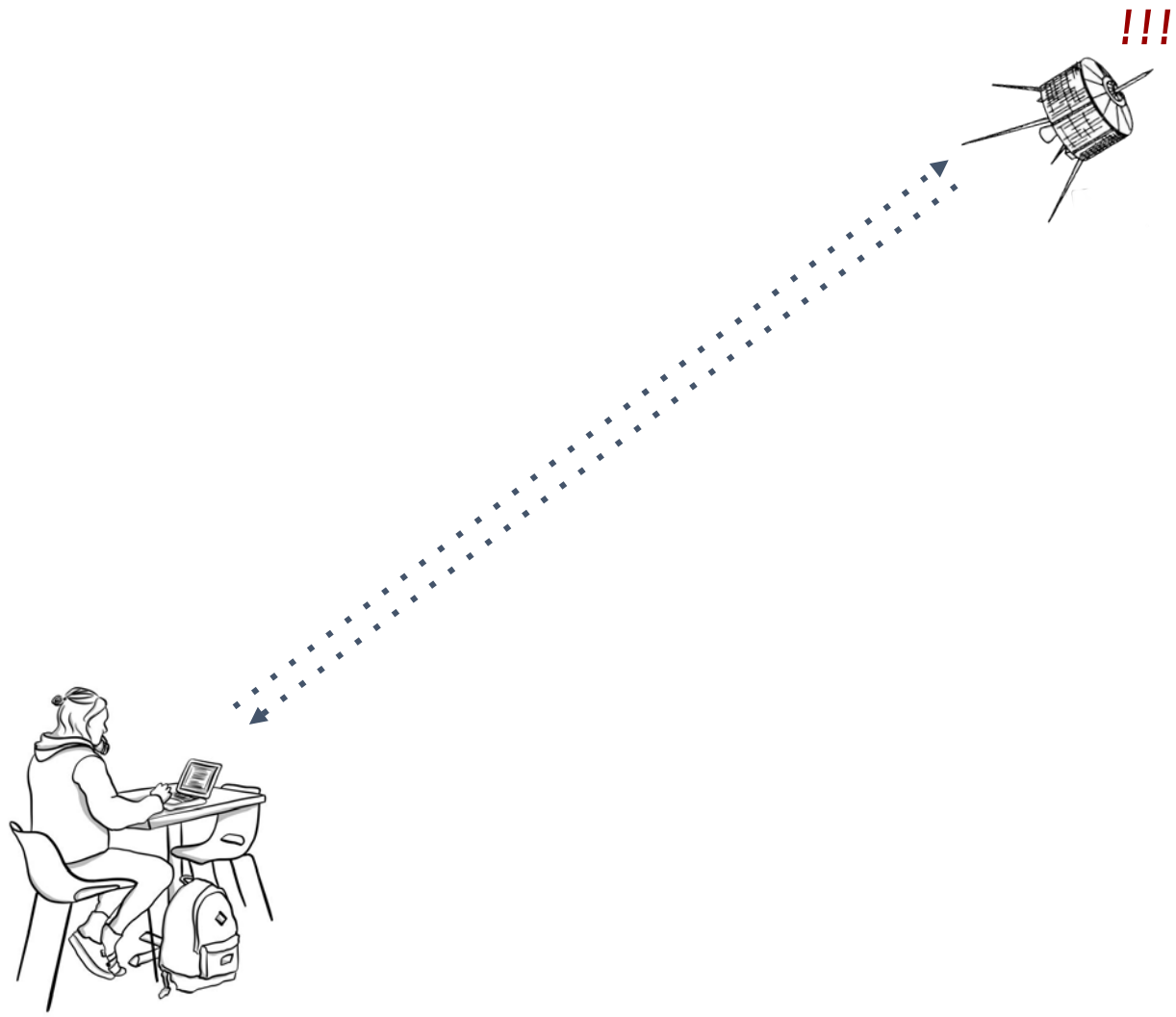
Aparna Rajesh



MIT
HAYSTACK
OBSERVATORY

Commanding?

- Once in orbit, satellites must maintain a connection with ground systems
- Example tasks:
  - Ping
  - File transfer
  - Data collection
- Commands are sent up via the uplink & a response is sent back via the downlink

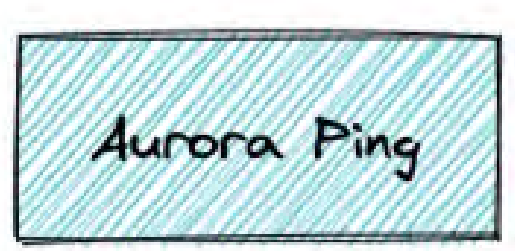


MIT HAYSTACK OBSERVATORY

satellite image: https://www.onlygfx.com/wp-content/uploads/2020/05/satellite-drawing-cover.jpg
person at desk image: https://media.istockphoto.com/vectors/thinking-and-typing-vector-id1154317230?k=6&m=1154317230&s=612x612&w=0&h=oVOhCOu4lYlmDjyJNfZIE1V7-giVsVADc7jRJNC_jhg=

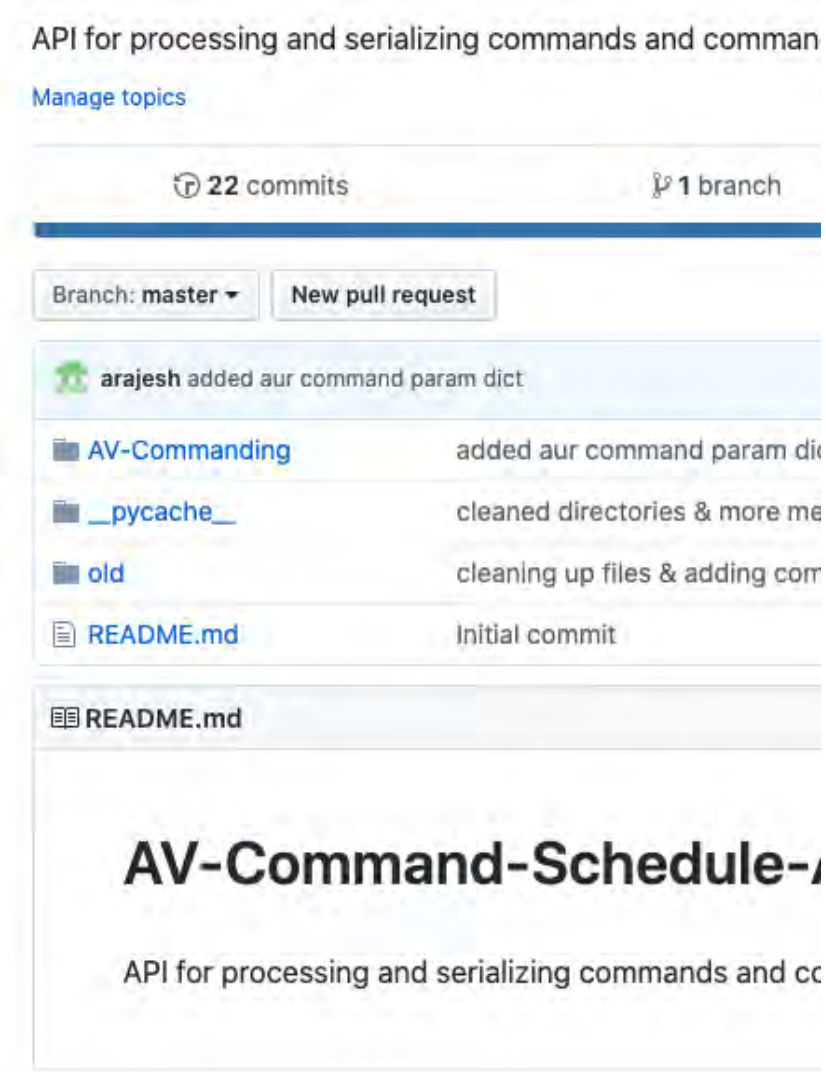
Goals

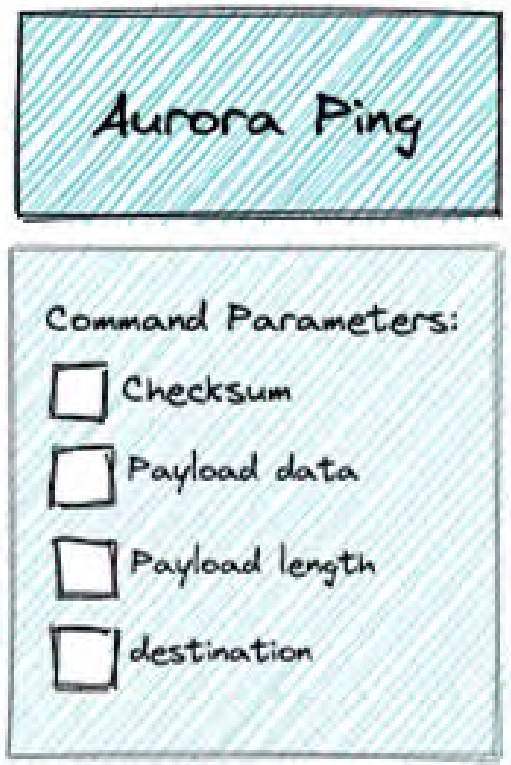
- Develop an API (Application Programming Interface, an intermediary software that takes data from a different software and then sends it to another API or program) that will:
  - neatly package command data and metadata
  - serialize all the information pertaining to a command or command schedule in a predetermined format
    - human readable format
    - hex format for the spacecraft
  - pass this byte stream to another program



AV-Command-Schedule-API

- The AV-Command-Schedule-API:
  - creates a command object with using instance-specific parameters and other data stored in config files
  - establishes a command “connection” using a connection modifier
    - *where will this command be sent?*
  - generates a human-readable component of the command object
  - generates a hex stream that is to be read by the spacecraft

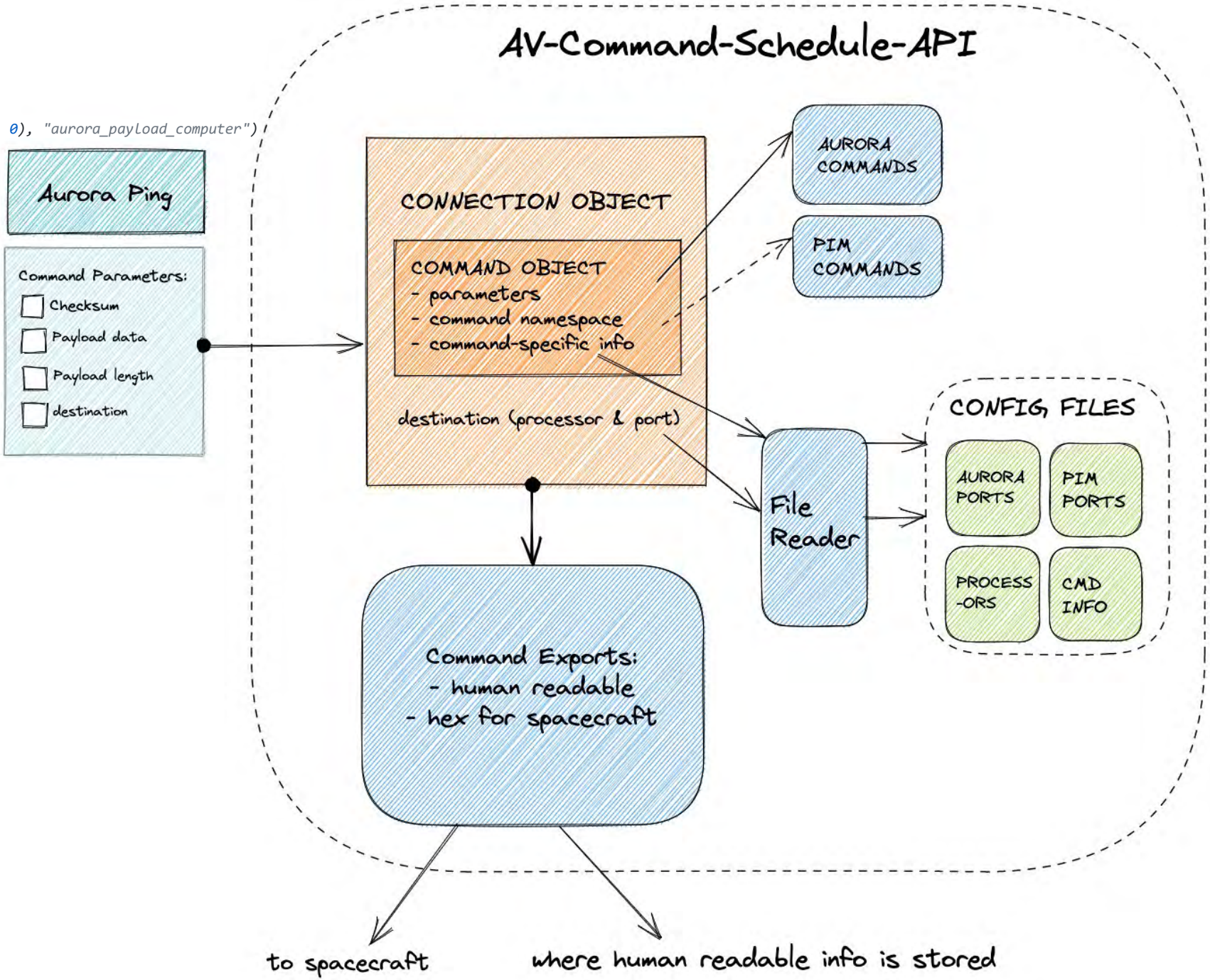





```
test = Connection(ping_aurora(5, 4, 2, 1, 0), "aurora_payload_computer")
```

MIT
HAYSTACK
OBSERVATORY

```
test = Connection(ping_aurora(5, 4, 2, 1, 0), "aurora_payload_computer")
```

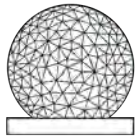


Next Steps

- Command schedules
- Hex headers
- Command dictionary parameter data types
- Config files

Ground Station Radio Communications

T. Lucas Briggs

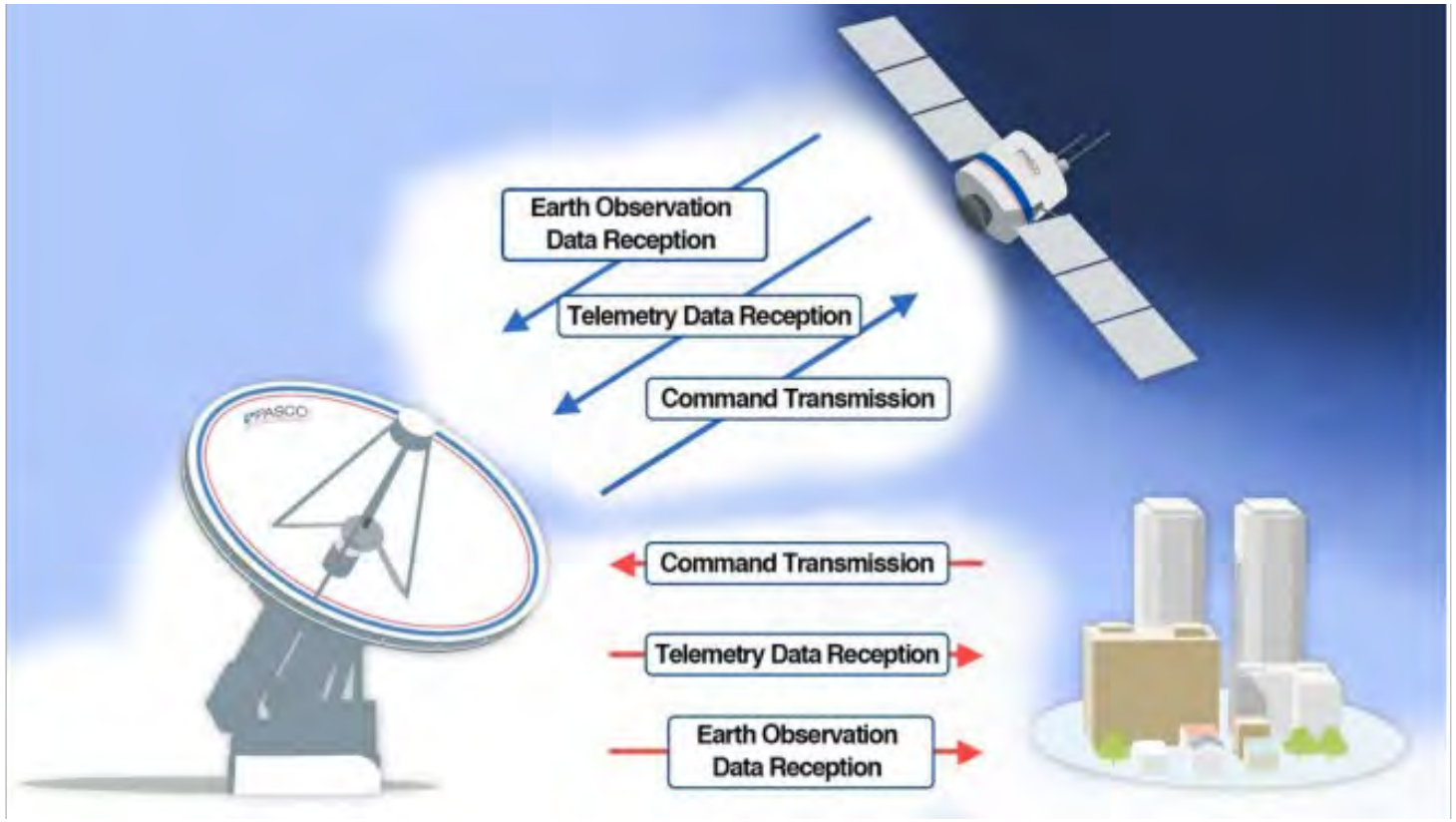


MIT
HAYSTACK
OBSERVATORY

What is a Ground Station?

- Enables communication between spacecraft and mission operations

- Handles all radio operation, command scheduling, and data verification tasks



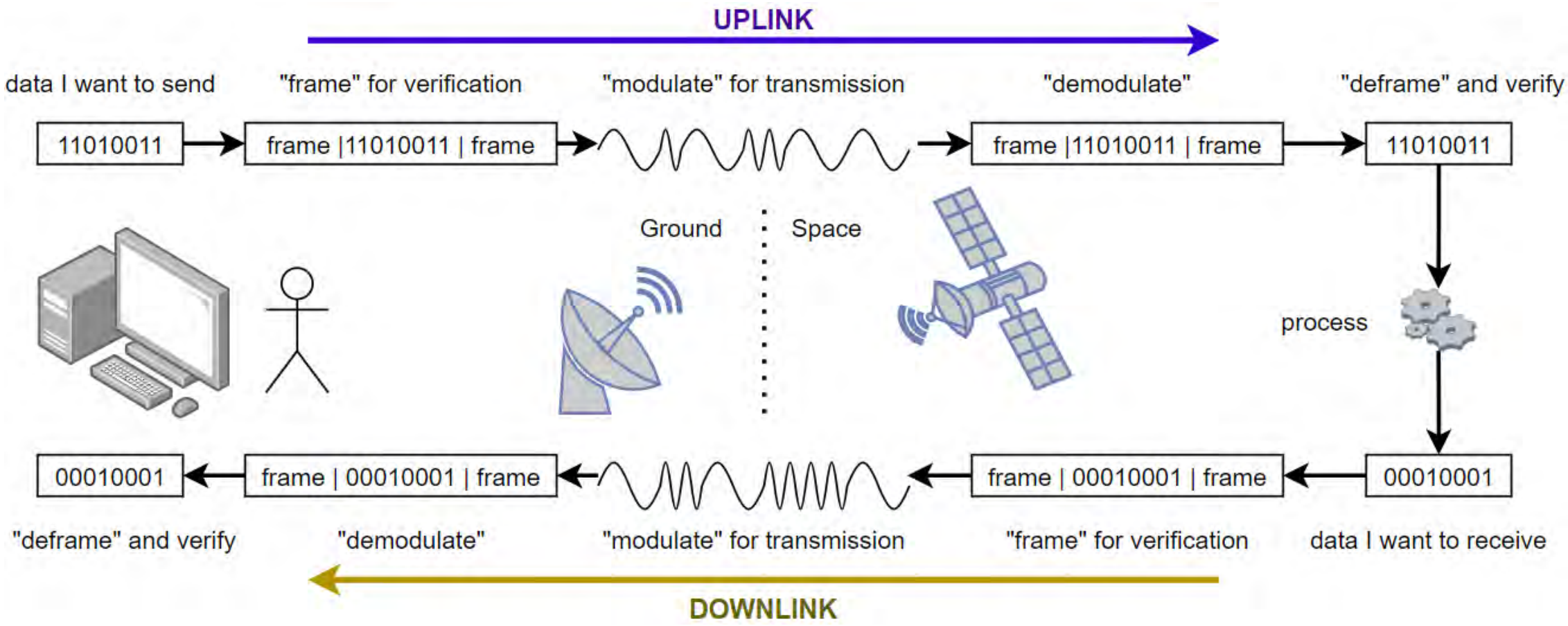
Source: https://www.gim-international.com/content/news/pasco-provides-rental-service-for-satellite-ground-station-facilities

MIT HAYSTACK OBSERVATORY

For AERO / VISTA

- Multiple possible Ground Stations in different locations with a wide range of radio configurations
- Many different types of data with varying sizes of payload
  - Commands: small, carries data necessary for spacecraft to execute desired action.
  - Acknowledgements: very small, carries data necessary to say “command received”
  - Telemetry: large, carries as much information as possible about spacecraft state
  - Science: very large, carries a full time-series of spectrum data from auroral emissions

Digital Communications over Radio

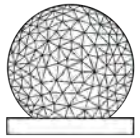


MIT HAYSTACK OBSERVATORY

Summary of Requirements

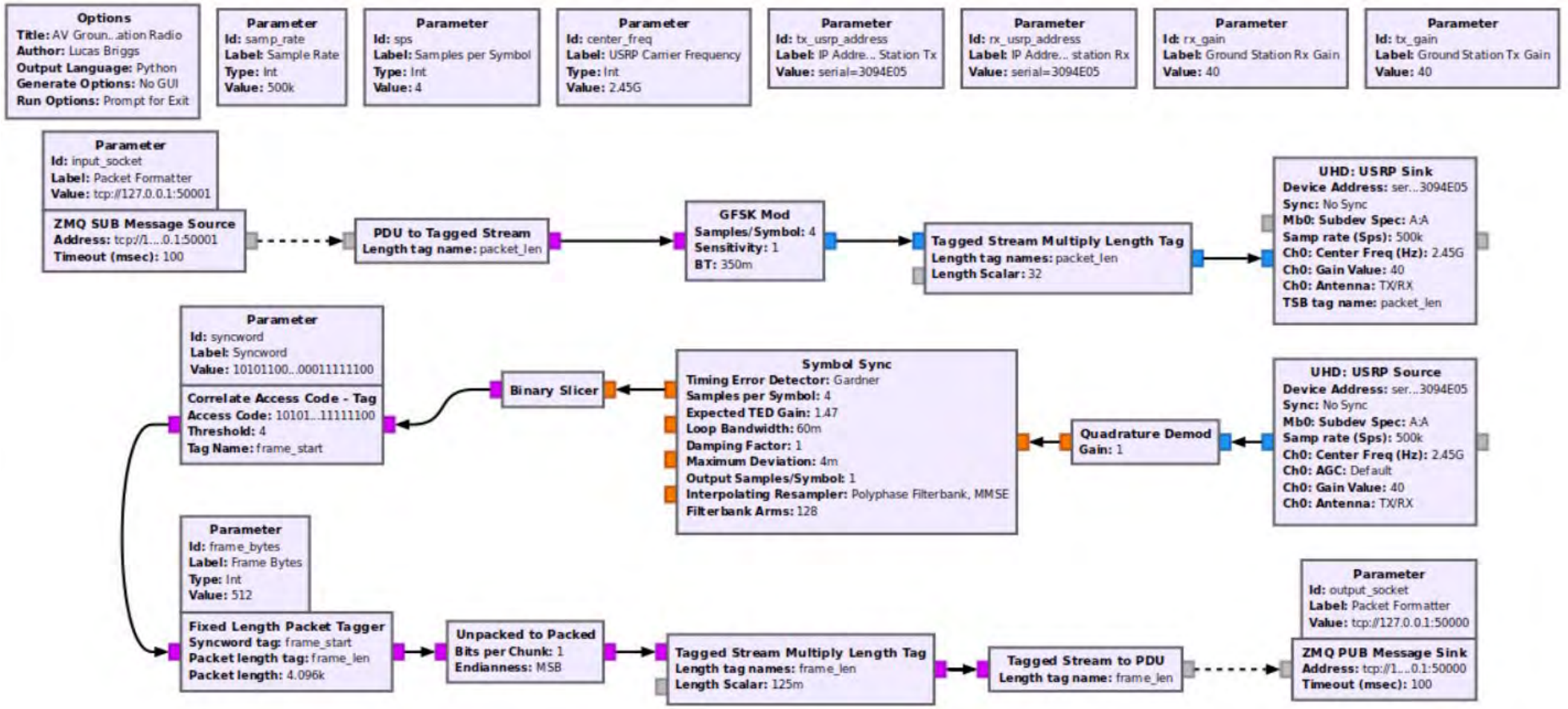
Need a software package that...

- Enables framing/deframing and mod./demod. on uplink/downlink.
- Allows for a wide range of data payload lengths.
- “Drops in”. Can be imported, started, and maintained anywhere.
- Provides a way to access data remotely and asynchronously.
- Is as configurable as possible
- Is as extendable as possible



MIT
HAYSTACK
OBSERVATORY

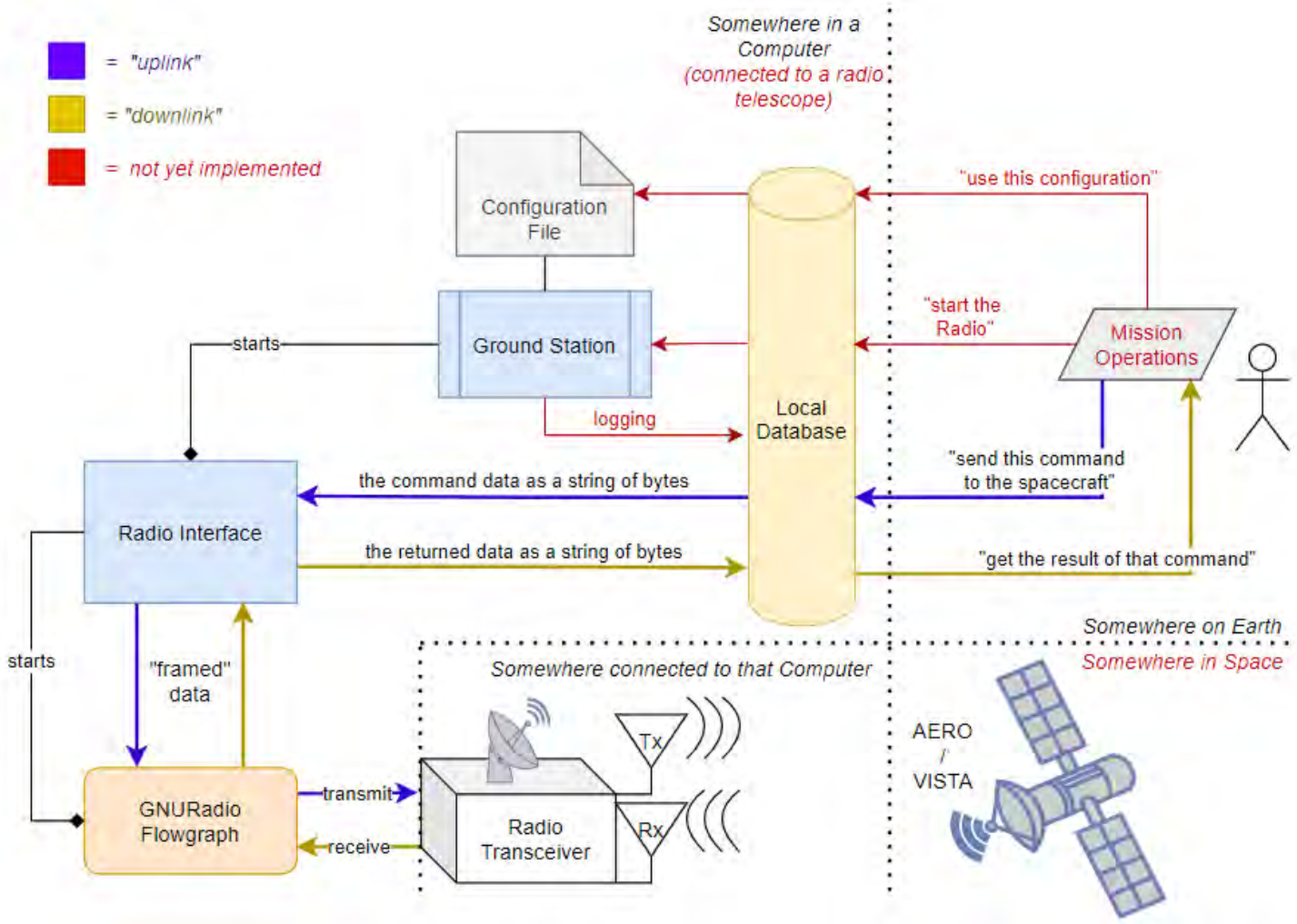
Implementation - GNURadio Flowgraph



MIT HAYSTACK OBSERVATORY

Implementation - Software (simplified)

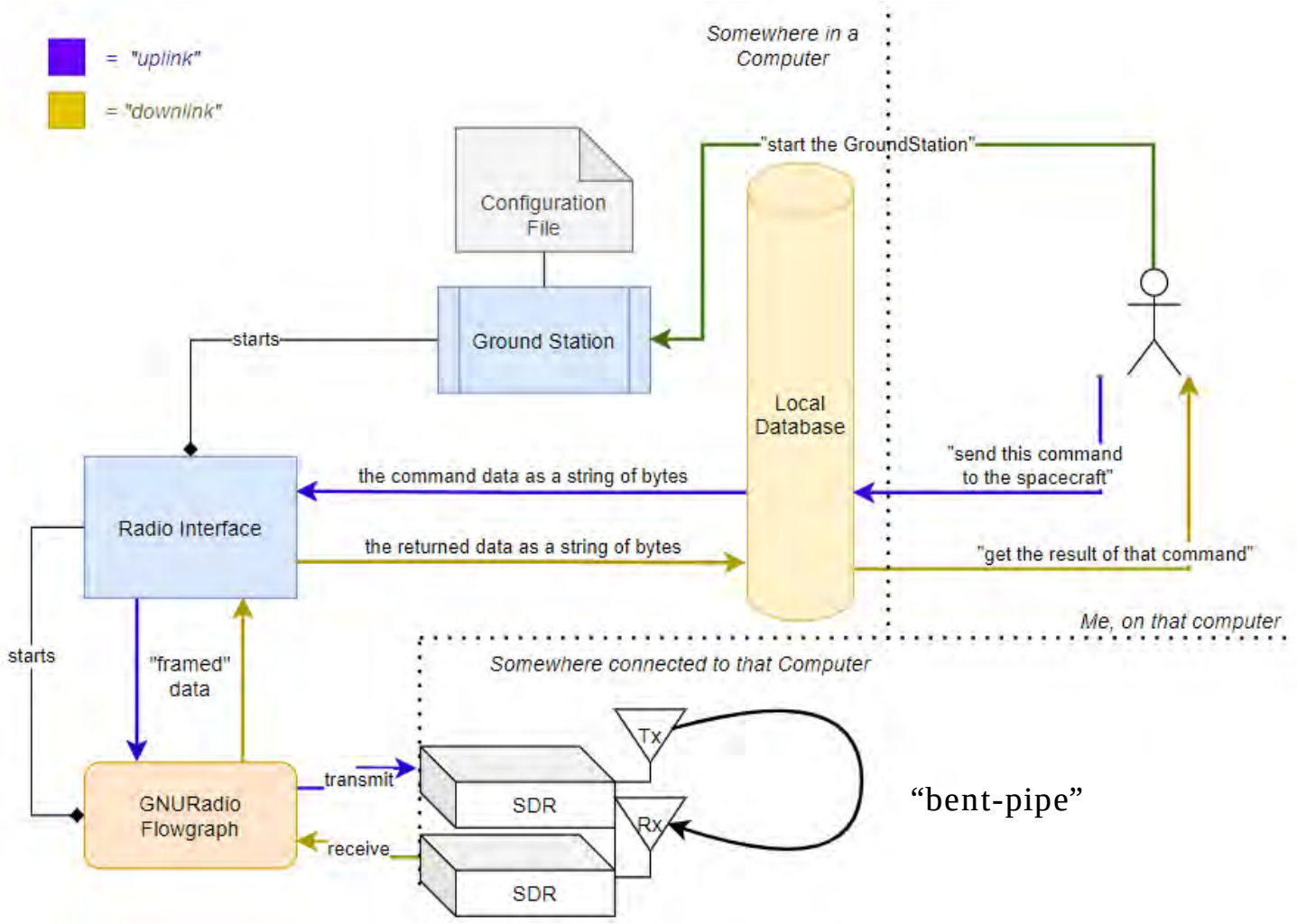
- Database provides asynchronous remote access.
- Ground Station provides configuration, handles top-level operation.
- Radio Interface manages framing/deframing data, uplinking/downlinking frames.
- Flowgraph operates the Radio Transceiver.



MIT HAYSTACK OBSERVATORY

Implementation - Software (simplified)

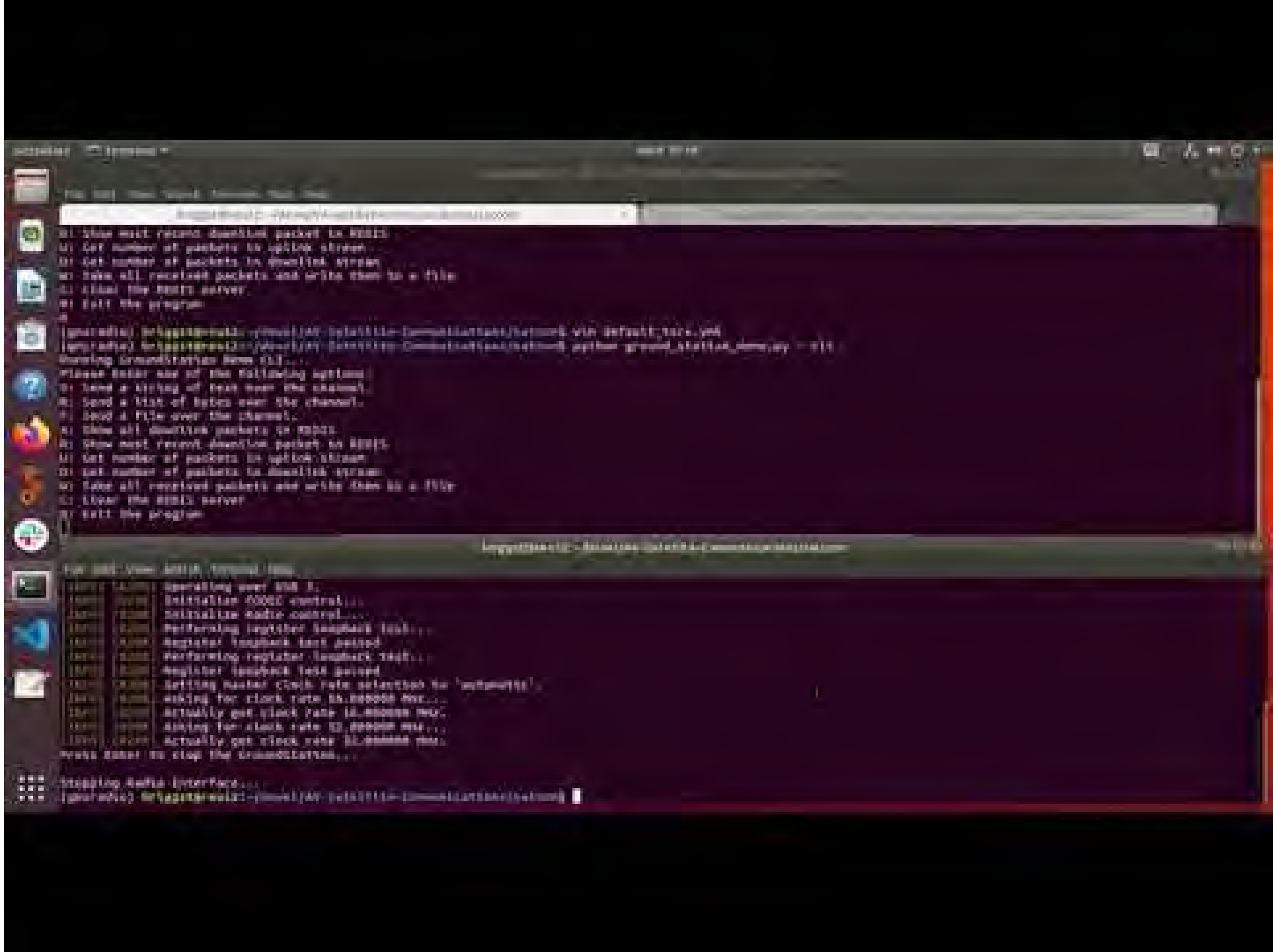
- Database provides asynchronous remote access.
- Ground Station provides configuration, handles top-level operation.
- Radio Interface manages framing/deframing data, uplinking/downlinking frames.
- Flowgraph operates the Radio Transceiver.



"bent-pipe"

MIT HAYSTACK OBSERVATORY

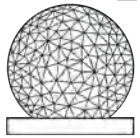
Live Demo - PNG File Transfer



MIT HAYSTACK OBSERVATORY

Next Steps

- Interaction of the Ground Station with the Database needs to be expanded
  - Operation commands, logging, configuration, and satellite command scheduling, all remote-asynchronous
- GNURadio Flowgraph needs to be expanded
  - More/better signal processing for higher SNR (configurable)
- Develop a “mock satellite” version of the Ground Station
  - Responds to known commands with dummy data
  - Potentially applies a channel model to the signal to test non-ideal conditions
- Integrate with a real transceiver and satellite engineering model

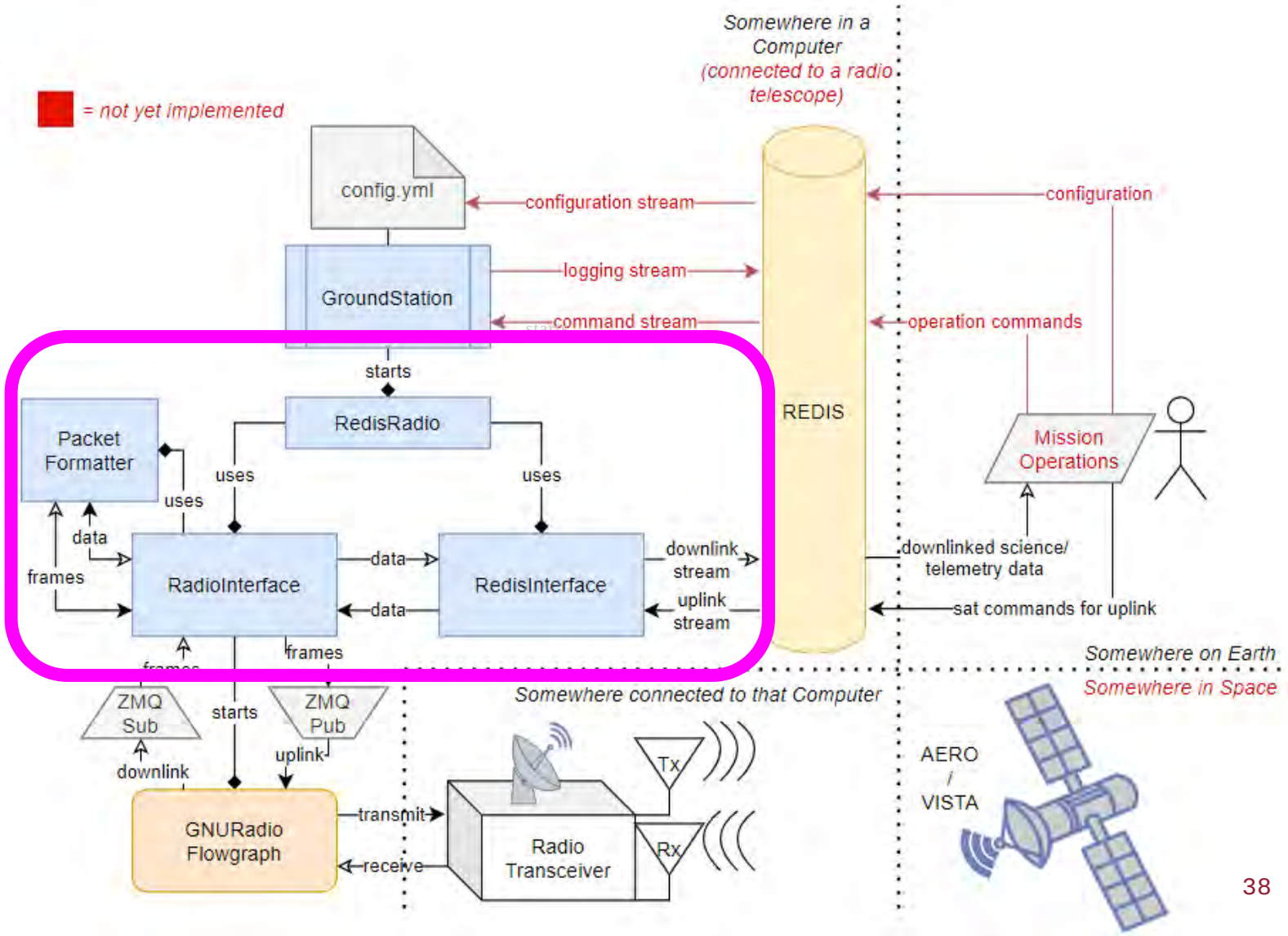


MIT
HAYSTACK
OBSERVATORY

backup

Implementation - Software (expanded)

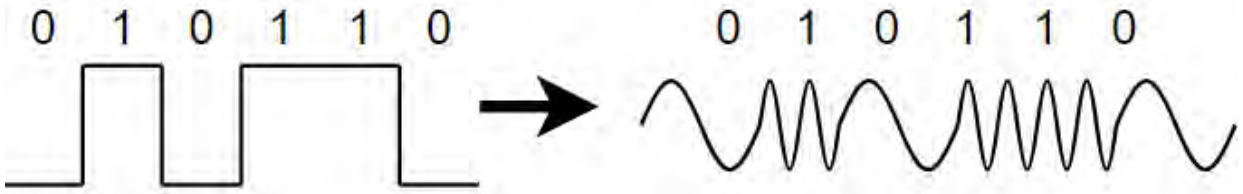
- Radio Interface exists as 3 important blocks
- Each has a single responsibility, promotes extendability
- See report for more details



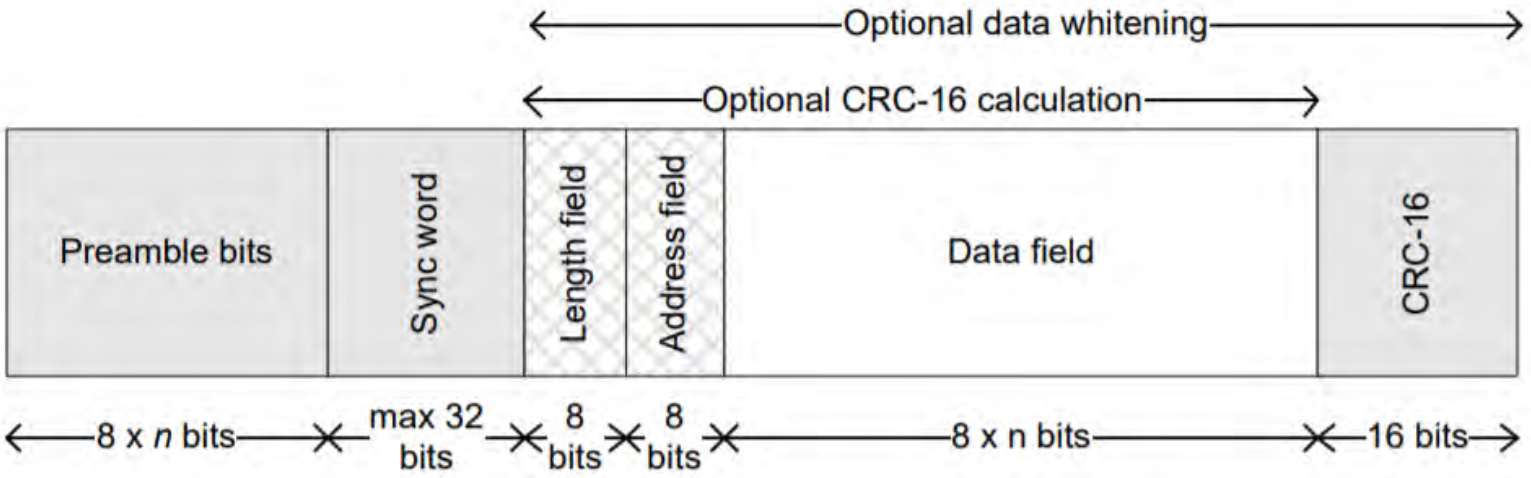
MIT HAYSTACK OBSERVATORY

Modulation and Framing for A/V

- Modulation using Gaussian Frequency Shift Keying (GFSK)
  - Looks a bit like this:



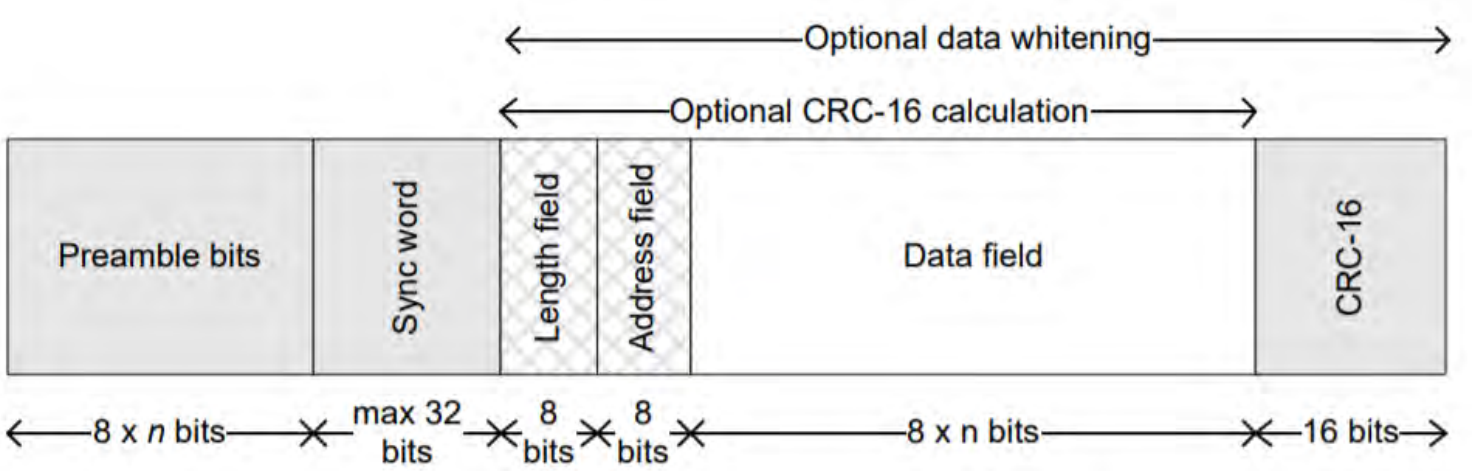
- Flexible data framing with the following template:



MIT HAYSTACK OBSERVATORY

Reliable Data Transfer: Framing

- Preamble: recognize symbols, “lock on” to binary data
- Syncword: find the start of the data packet
- Length: how many bytes to expect in the data
- Address: where should this data go



- CRC (Cyclical Redundancy Check): Verify that the data has not changed
- Whitening: Deterministically randomize the data

MIT HAYSTACK OBSERVATORY

Configuration Through YAML Files

Configurable parameters are categorized into “packet”, “radio”, and “redis” groupings

default_txrx.yml M

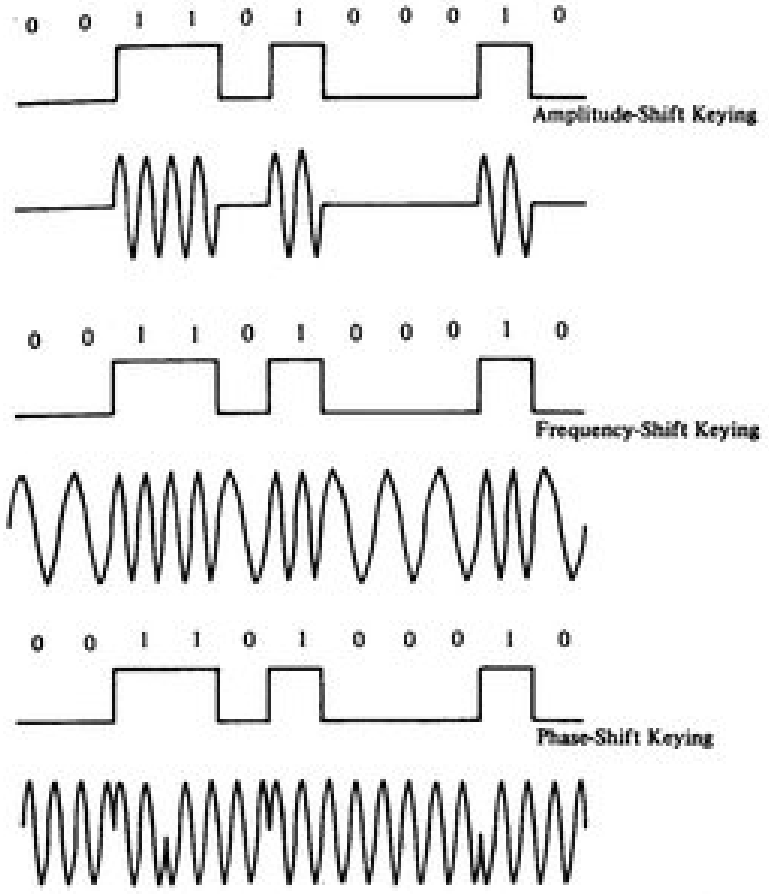
AV-Satellite-Communications > satcom > default_txrx.yml

```yaml
packet:
  formatter: "default"
  preamble: "1010010011110010" # ""
  syncword: "10101100110111011010010011100010111100101000110000100000011111100"
  whiten: True
  whitener_offset: 0
  crc: True

radio:
  center_frequency: 2450000000
  sample_rate: 500000
  samples_per_symbol: 4
  rx_gain: 40.0
  tx_gain: 40.0
  # These addresses can be the same (sat-in-the-loop test)
  rx_usrp_address: "serial=3094DF7"
  tx_usrp_address: "serial=3094E05"
  # These addresses must be different
  rx_socket_address: "tcp://127.0.0.1:50000"
  tx_socket_address: "tcp://127.0.0.1:50001"
  receive_waits_s: [0.1, 1, 3, 5]
  max_payload_bytes: 512

redis:
  host: "localhost"
  port: 6379
  db: 0
  pkt_uplink_stream_name: "pkt-uplink"
  pkt_downlink_stream_name: "pkt-downlink"
  clear_on_start: True
```

MIT HAYSTACK OBSERVATORY

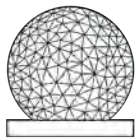
Digital Communications: Modulation

- Binary data representation on a “constant wave”, accomplished through **keying**
  - Amplitude-Shift Keying (ASK)
  - Frequency-Shift Keying (FSK)
  - Phase-Shift Keying (PSK)
  - ...many more
- Our solution: Gaussian FSK (GFSK)
  - FSK with no sudden jumps



MIT HAYSTACK OBSERVATORY

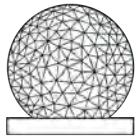
Final Group Demo



MIT
HAYSTACK
OBSERVATORY

Thank you to our mentors

Mary Knapp, Ryan Volz, John Swoboda, Frank Lind, Phil Erickson, Toby Gedenk, and Geoff Crew



MIT
HAYSTACK
OBSERVATORY