Global Navigation Satellite System Positioning Performance Monitoring System

Final Design Review

Kevin Croissant, Jonathan Waters, Zachary Arnett, Hunter Fordyce
FPL: Dr. Michael Braasch

April 22, 2019

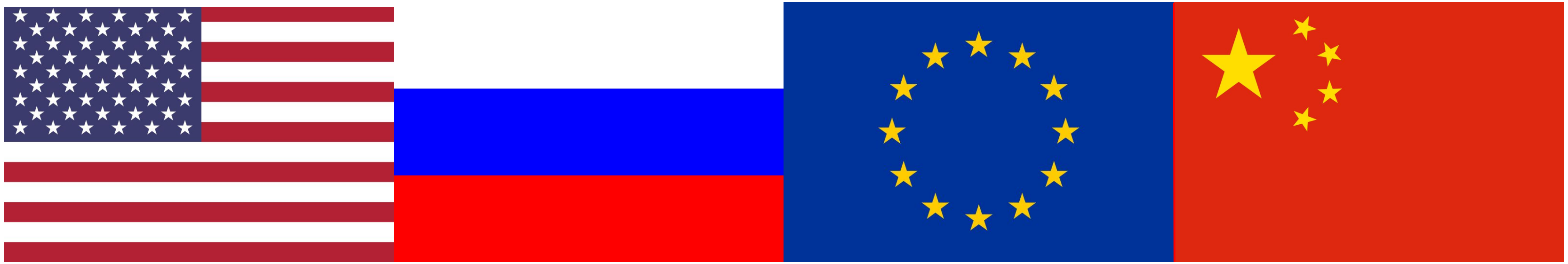
Outline of Talking Points

1. Introduction and Project Summary
   1.1. Project Description
   1.2. Summary of System Requirements
2. Final Design Summary
   2.1. Trade Studies Summary
   2.2. Final Design Technical Results
   2.3. Utilization of Engineering Standards for Design and Implementation
3. Verification and Test Results
   3.1. System Level Test Matrix
   3.2. Summary of Test Results
   3.3. Conclusions
4. Programmatic Summary
   4.1. Budget
   4.2. Schedule
   4.3. Personnel Time
   4.4. Intellectual Property
   4.5. Conclusions
5. Overall Conclusions and Recommendations
   5.1. Summary of how well the project me specifications and expectations
   5.2. Estimate the percentage of project completion
   5.3. Recommendations for future design/development
   5.4. Lessons Learned
6. Demonstration
7. Questions/Comments

1. Introduction and Project Summary

Introduction

GPS GLONASS Galileo BeiDou



1.1 Project Description

1.1.1 Customer Needs

Need Statement

The field of GNSS is quickly growing and evolving and the applications that GNSS is used for is quickly expanding. The major global GNSS producers are working on improving their systems with increasing satellites for better coverage in a race to improve the performance of the systems to impact humanity is a positive way. Public access to the individual positioning performance of multiple constellations over time is not easily accessible in a human-readable format. An easy to access human-readable format is needed for monitoring the individual positioning performance of GGGB to track the reliability each system provides.

1.1 Project Description

1.1.2 Goal Statement and Objectives

Goal Statement

Design and implement a system that monitors and displays the individual positioning performance of the GGGB constellations over Athens, Ohio in an easy to access human-readable format that is open to the public by the end of the 2018-2019 academic year.

Objectives

- Develop a system to simultaneously collect data from each GGGB constellation individually
- Develop a server to host the collected data
- Develop a website to display the positioning performance of the constellations
- Write a user manual for future maintenance of the system that will include all source code

1.1 Project Description

1.1.3 Final Deliverable

- An operational system that collects position data computed from each individual GGGB constellation
- Website capable of showing individual GNSS constellation position error over time
- System documentation and source code for ongoing maintenance and upkeep
  - All design documentation
  - All configuration information
  - All installation/deployment information
  - All software source code
  - Operations and maintenance documentation
  - Any other information required to replicate, install, and maintain the system
- Presentation and live demonstration at the Student Expo on April 11th, 2019
- Presentation and live demonstration for Final Design Review on April 23, 2019

1.1 Project Description

1.1.4 Statement of Work

1. In order to design a GNSS system with an Antenna and Receiver(s), the team shall acquire and install both receiver(s) and an antenna, using the funds provided by the customer, in a manner that will ensure long term maintainability, ease of access, and proper functionality of the system.
2. The team shall install server hardware and test the server hardware performance to ensure that all performance requirements are met, as well as ensure the reliability of the system.
3. The team shall design software in order to process and display the individual position performance data of each of the GGGB constellations.
4. Once the software has been created, the team shall configure automatic file backups for collected GNSS data, ensuring the software can utilize the data to update the static page generator which will give the public access to the data and ensuring that all the performance requirements are met.

1.1 Project Description

1.1.4 Statement of Work (cont.)

5. Once both the hardware and software have been implemented, the team shall test and debug both systems to ensure the integrity of the system.
6. In order to display the processed data, the team shall create a website that allows for users to view collected data in user friendly way.
7. Documentation will be created in order to ensure the longevity of the system.

1.2 Summary of System Requirements

1.2.1 Concept of Operation

In order to test the performance of different GNSS constellations, the precise location of the antenna needs to be surveyed to act as a base station. Once the location of the base station is known, a system can be developed to analyze the individual performance of these constellations. The antenna will receive signals from multiple constellations and the receiver will process the signals and output the signals into usable data format. The data will be stored in a database, and plots of the data will be generated. The plots will then be displayed on the web page, providing the easy to access human-readable format that is needed for monitoring the performance of these GNSS constellations

1.2 Summary of System Requirements

1.2.2 Functional Requirements

- Hardware:
  - Antenna must be mounted in an area with little-to-no obstructions nearby
  - The attenuation of the cable from the antenna to the receiver shall not drop the signal level below the threshold of the receiver
  - Receivers shall be connected to the server through USB cables
  - Server must be connected to the internet via Ethernet
- Software:
  - System shall provide a way to extract data via downloadable csv file from the website
  - System shall be capable of collecting data and plotting it without an internet connection

1.2 Summary of System Requirements

1.2.3 Performance Requirements

- Hardware:
  - Amplifier shall be used to ensure signal strength is operating in the rated input sensitivity range of receiver as needed
  - All equipment being used outdoors shall be rated for outdoor use
  - The server shall have a processor capable of running all necessary software without consuming more than 75% of total processing capacity as measured by the system's 15 minute load average
- Software:
  - System shall receive GNSS data at a minimum of once per 30 seconds
  - Data plots shall be updated at a minimum of once per day
  - Web server shall serve pages in under one second as tested via a loopback connection
  - Website shall display correctly in all major and modern desktop web browsers and operating systems

1.2 Summary of System Requirements (cont.)

1.2.4 Operational Requirements

- Hardware
  - Server have a minimum of 4 GB of RAM
  - Server shall have enough storage for 20 years of data collection
  - Antenna shall be mounted in a location not affected by nearfield of other nearby antennas
  - Server shall be connected to an Uninterruptible Power Supply
- Software
  - Server shall have a publicly accessible IPv4 IP address.
  - Server shall have a firewall that automatically bans IPs that fail to log in more than 3 consecutive failed attempts

2. Final Design Summary

2.1 Trade Studies Update/Summary

- Four U-Blox receivers were purchased
- NovAtel GPS-704-X antenna was provided by Dr. van Graas
- Loss of cable did not drop signal strength below receiver signal strength sensitivity, so no in-line amplifier was used.
    - A pre-amplifier (JCA12-4189t) was used near the antenna to minimize noise.
- Cables were not studied due to existing cable runs and cable stock that was accessible for the group to utilize
    - Mounting materials were also available from Avionics
- Server was not trade studied due to having access to old server hardware. The hardware proved unreliable and had to be replaced.

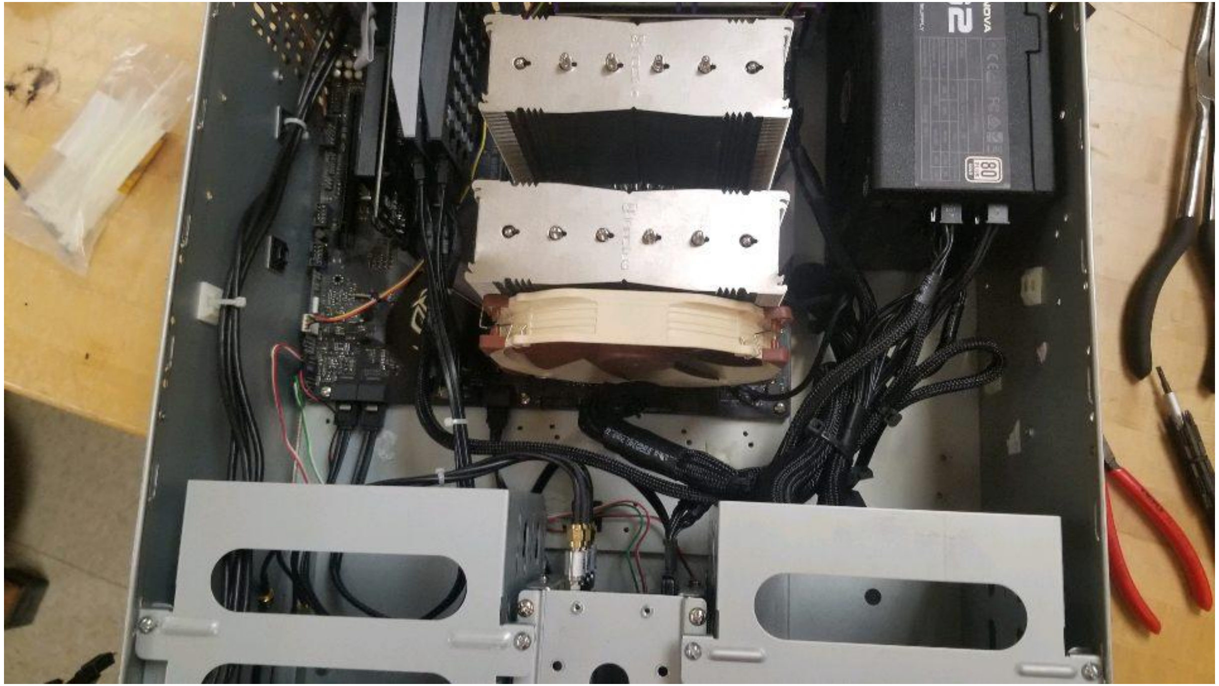
2.2 Final Design Technical Results

2.2.1 Hardware

- CPU: AMD Ryzen 7 2700X
- RAM: CORSAIR Vengeance LPX 32GB (2 x 16GB)
- Motherboard: ASUS ROG STRIX B350-F AM4 Socket
- GPU: GIGABYTE GeForce GT710
- Cooling:
  - Noctua NH-D15 (CPU Heatsink + 2x 120mm fans)
  - GELID Solutions GC-Extreme Thermal Compound
  - Noctua NM-AM4 (NH-D15 Bracket)
  - Noctua NF-F12 PWM (Chassis Fans)
- Uninterruptible Power Supply: Cyberpower OR700LCDRM1U
- Server PSU: EVGA SuperNOVA 550

Server Rebuild Day

Server Rebuild Day



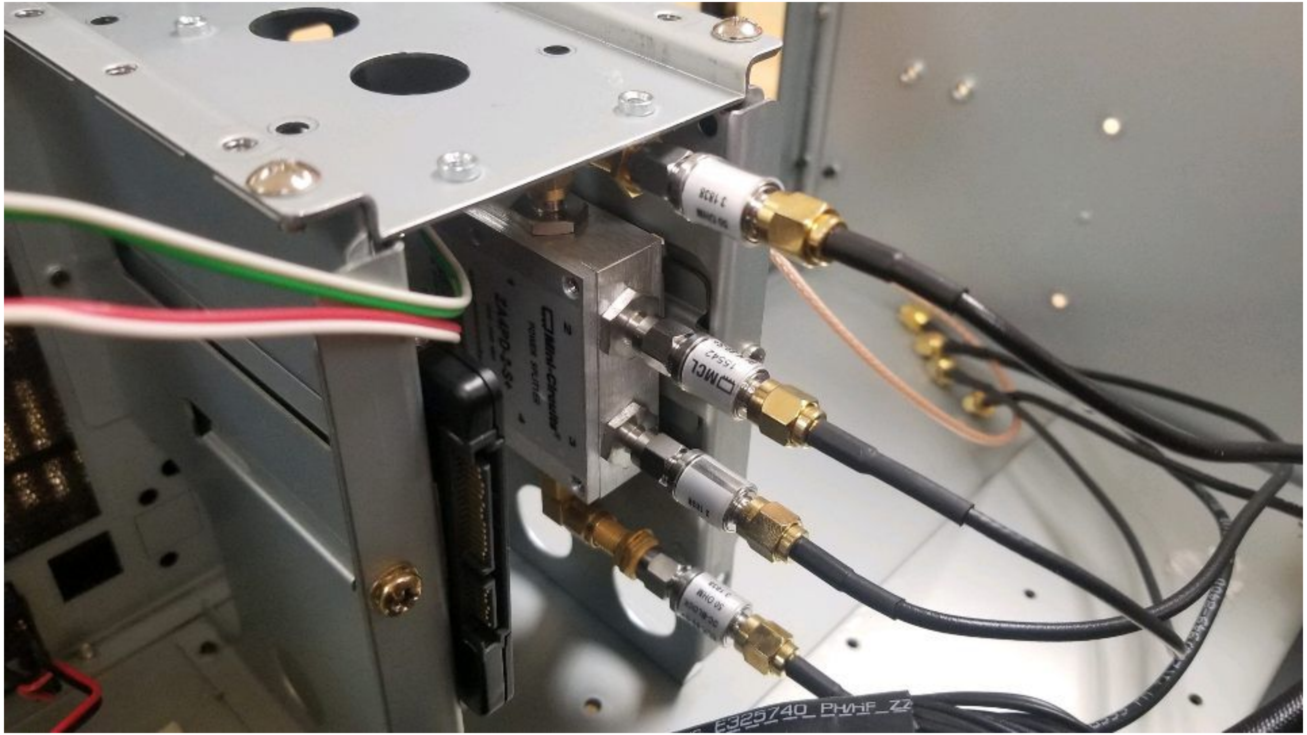
Server installed in Server Room

2.2 Final Design Technical Results

2.2.1 Hardware (cont.)

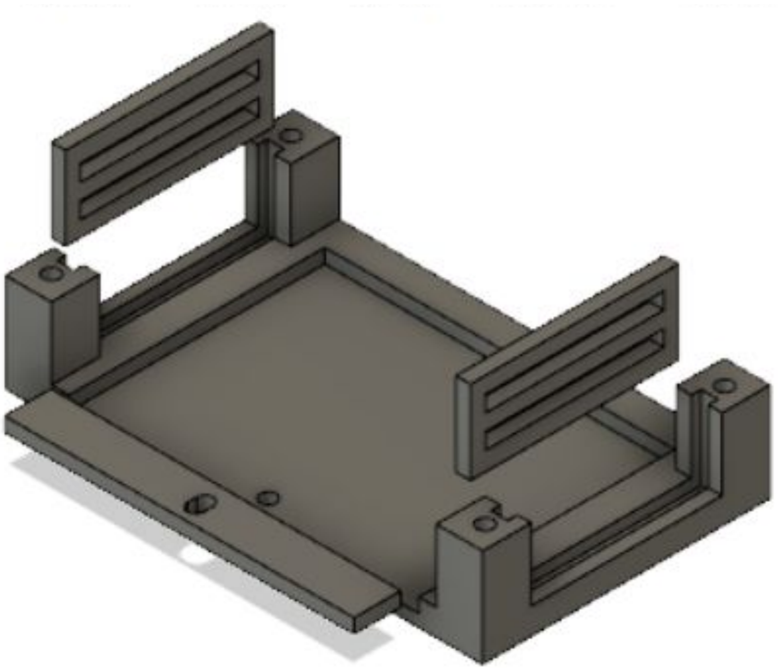
We chose to use the u-blox NEO-M8N receivers for this project.

- Supports reception of a single constellation (others do not)
- Small form-factor
- Affordable price ($180/receiver development kit)
- Professional grade receiver, used in many cars and consumer electronics
- Previously used for 2017 Senior Design “Surveillance Broadcast sUAS Payload for Low-Altitude Fleet Operations”



Four Way L-Band Power Splitter: ZA4PD-2-S+

Four U-Blox Receivers



U-blox stackable receiver bracket

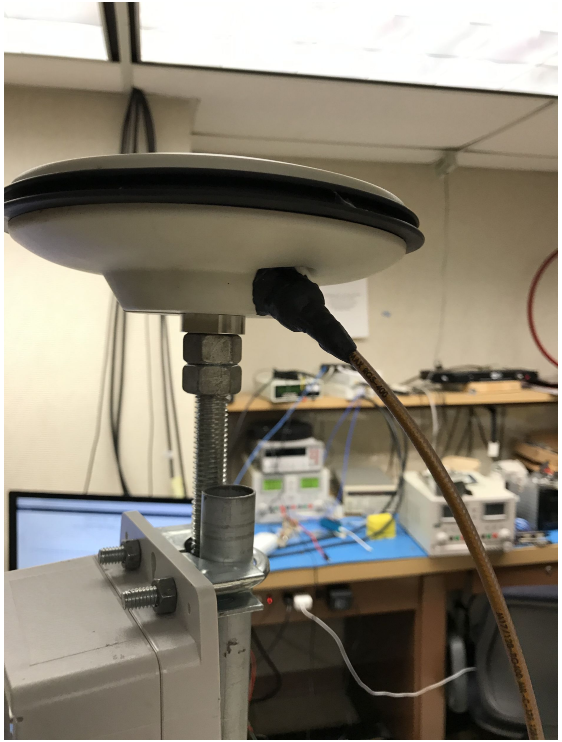
2.2 Final Design Technical Results

2.2.1 Hardware (cont.)

We used the NovAtel GPS-704-X antenna for our project

- Previously installed on Stocker Roof, but was inside a failing fiberglass enclosure
- We built a new enclosure and remounted it on the catwalk
- Antenna is connected to a distribution amplifier and is shared between many Avionics Engineering Center labs
- Even though we only use the L1 signal, we wanted to have the highest quality signal possible

Antenna Preamplifier: JCA12-4189T



Antenna: NovAtel GPS-704-X

2.2 Final Design Technical Results

2.2.2 Software

- Quick stats:
  - Backend code (data collection, plotting) is approximately 10,000 lines of Python3 code
  - Frontend code (website) is approximately 2000 lines of PHP and HTML
- Languages used:
  - Python3 for all backend code
  - SQL used to calculate statistics
  - PHP for frontend code

2.2 Final Design Technical Results

2.2.2 Software (cont.)

- Python3 libraries used
  - ‘matplotlib’ for plotting
  - ‘schedule’ for the job scheduler core
  - ‘pymysql’ to interact with database
  - ‘numpy’ and ‘scipy’ for calculating statistics
  - ‘configparser’ for custom configuration files
  - ‘multiprocessing’ to parallelize plotting jobs
  - ‘ubx’ and ‘pyserial’ for interfacing with receivers for data collection
  - ‘pymap3d’ for coordinate transformations

2.2 Final Design Technical Results

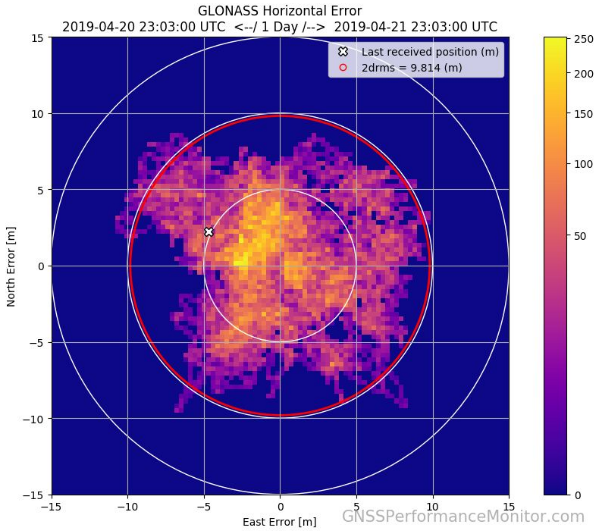
2.2.2 Software (cont.)

- Software used
  - MariaDB for data storage
  - ffmpeg to create 3D rotating scatterplots
  - Apache2 for web server
  - PHP for creating pages dynamically
  - Varnish as HTTP cache to reduce webserver load
  - Hitch used as a TLS termination proxy in front of varnish
  - nftables used as system firewall
  - Debian Linux used as operating system
  - Monitorix / rrdtool for server monitoring

2.2 Final Design Technical Results

2.2.3 Design Algorithms and Formulas

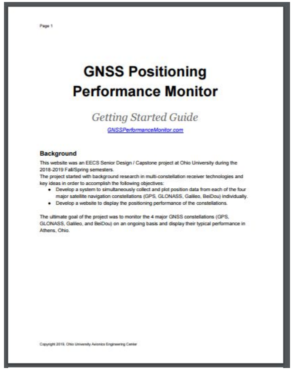
- Lat-Lon-Height (MSL) to East-North-Up (ENU) requires going LLH->ECEF->ENU
- ENU is a local reference frame. In our system, our antenna is defined as 0,0,0 ENU. Any deviations from the origin are position errors.
- These deviations are caused by error sources such as tropospheric delay, ionospheric delay, receiver noise, orbit errors, and multipath.



2.2 Final Design Technical Results

2.2.4 Getting Started Guide

- This guide was created to help first time users understand the heatmaps and graphs used to display the data
- This guide also shows in detail how users can download the data to be used for their own purposes.



Page 1

GNSS Positioning Performance Monitor

Getting Started Guide

GNSSPerformanceMonitor.com

Background

This website was an EECS Senior Design / Capstone project at Ohio University during the 2018-2019 Fall/Spring semesters.
The project started with background research in multi-constellation receiver technologies and key ideas in order to accomplish the following objectives:

- Develop a system to simultaneously collect and plot position data from each of the four major satellite navigation constellations (GPS, GLONASS, Galileo, BeiDou) individually.
- Develop a website to display the positioning performance of the constellations.

The ultimate goal of the project was to monitor the 4 major GNSS constellations (GPS, GLONASS, Galileo, and BeiDou) on an ongoing basis and display their typical performance in Athens, Ohio.

Copyright 2019, Ohio University Avionics Engineering Center

2.2 Final Design Technical Results

2.2.5 Operational Design

- Affordability
  - Supplies on hand saved cost
  - Material Cost, including unexpected costs of rebuilding server
- Usability
  - Beginners guide to website
  - Background on GNSS to help
  - Website inspired by similar online data services
- Manufacturability
  - N/A : All OEM parts and provides a service

2.3 Utilization of Engineering Standards

- IPC-620 - Wire Harness Standard (Quality and bend radius of wiring , etc)
- GPS Interface Specification Document: IS-GPS-200J
- GLONASS Interface Control Document
- Galileo Open Service Signal In Space Interface Control Document
- BeiDou Navigation Satellite System Signal In Space Interface Control Document
- Standard Coaxial Cable
- Standard NMEA-0183
- MIL-STD-348 Rev. B for Coaxial Connectors
    - SMA, N-Type, BNC, etc.

3. Verification and Test Results

3.1 System Level Test Matrix

- Test receivers with an antenna and coax run that is known to be functional
- Test coax run that is to be used for antenna to receiver connection for continuity and loss
- Test the growth rate of the database to ensure the storage that was chosen is sufficient for 20 years of data
- Test job scheduler for load management and apply adjustments to ensure the server hardware is not overloaded
- Test memory to ensure no memory leaks and system stability at memory level
- Test startup script for proper automatic start sequence upon a hard reset

3.2 Summary of Test Results

Receivers

- Testing the receivers on antenna that was already established to be working properly, allowing for the team to become familiar with the U-Blox receiver and the types of messages that the receiver sends
  - Confirmed functionality of receivers
  - Confirmed the receiver could receive signals from each constellation
  - Confirmed the receiver could provide a solution of one constellation independent of the other constellations
  - Provided insight into how the receivers should operate under normal conditions

3.2 Summary of Test Results

Coax

- Due to lack of signal generator for dB loss test of cable, alternative methods were used. A DMM was used to check for a short in the line.
    - Continuity testing checks for shorts or improper impedance
    - Cable contained no shorts
    - Cable impedance read ≃∞ and thus indicated a partial connection. Inspection revealed a corroded connection

3.2 Summary of Test Results

System Load and Memory Usage

- System is now stable with peak system load (1, 10, 15 min averages) <= 60% of CPU capacity.

- Memory leaks were traced down and resolved, and system typically uses ~6GB RAM continuously.
- This will grow slowly over time - the motherboard can be upgraded to 64GB later.

3.2 Summary of Test Results

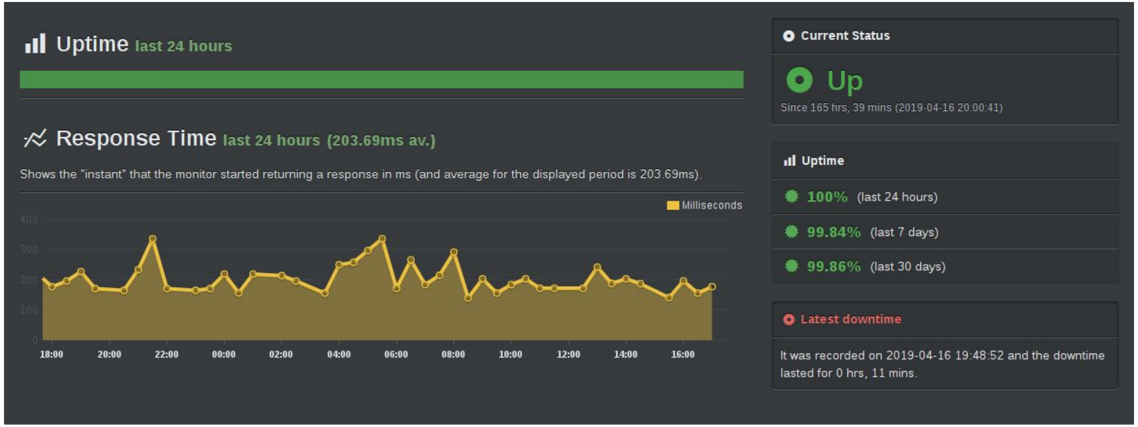
Database

- Database is growing at expected rate (~20MB/day), rate, which would allow for 100+ years of data collection
- Database server is able to handle current load and forecasted load for next 20 years.

3.2 Summary of Test Results

Website Uptime

- UptimeRobot.com monitors website 24/7 and continuously tests response time and record outages.
- Over last 30 days, website has had an uptime of 99.86%.
- Note: UptimeRobot response time is unusually high due to poor peering between UR servers and OU network.



3.3 Conclusions

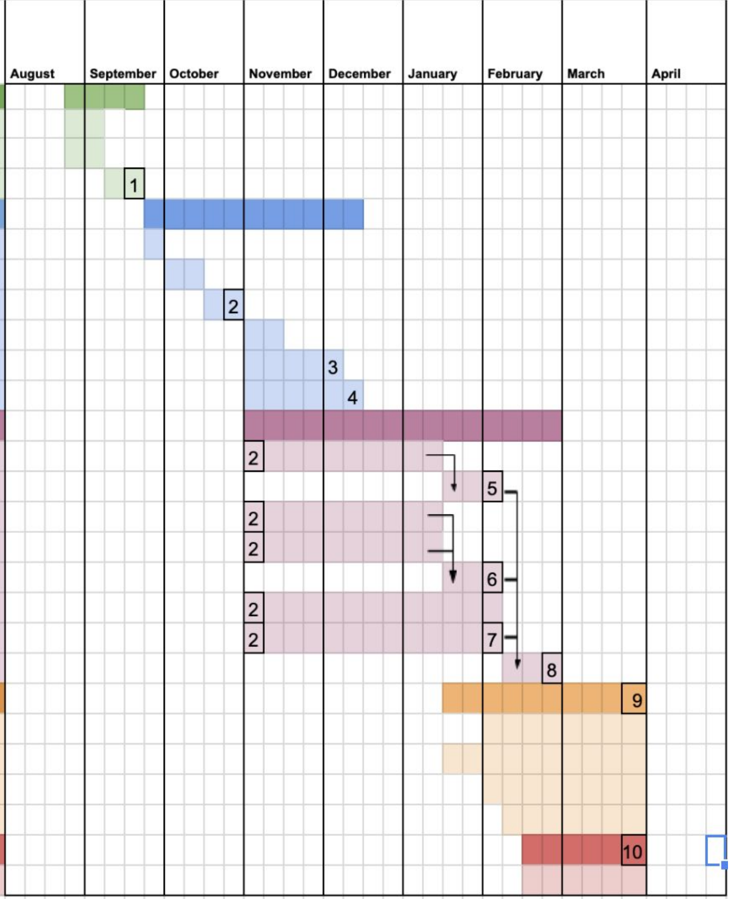
- The hardware and software of the project have been tested and verified to meet the functional, performance, and operational requirements of the proposal

4. Programmatic Summary

4.1 Budget

- The $500 budget from EECS was used to purchase some of the server hardware, as well as some small items such as coax adapters.
- The receivers and the majority of the server hardware was purchased by Dr. Braasch using his research funds (RI-PI funding).
- Total Funds used for project:
  - EECS Funding: Spent $499.86 / $500.00 (99.97%).
  - Dr. Braasch RI-PI funding: Spent $1,888.81
  - Total: **$2,388.67**
  - Original estimate from project proposal: $3,575.00

Project Phase Breakdown GNSS Team	WBS	Date Start (year-mm-dd)	Estimated Completion Date (year-mm-dd)
1.) Project Initiation		**2018-08-28**	**2018-09-21**
1.1) Memorandum of Understanding		2018-08-28	2018-09-07
1.2) Concept of Operation		2018-08-28	2018-09-07
1.3) System Requirement Review		2018-09-07	2018-09-21
2.) Project planning		**2018-09-21**	**2018-12-04**
2.1) Technical Approach		2018-09-21	2018-09-28
2.2) Management Plan		2018-10-02	2018-10-12
2.3) Project Proposal and Contract		2018-10-12	2018-10-30
2.4) Preliminary Design Review		2018-10-30	2018-11-13
2.5) Critical Design Review		2018-10-30	2018-12-04
4.1) Interim Progress Report		2018-10-30	2018-12-14
3.) Project Execution		**2018-10-30**	**2019-03-01**
3.1) Acquire RF Components		2018-10-30	2019-01-15
3.2) Install RF Components		2019-01-15	2019-02-04
3.3) Build server hardware		2018-10-30	2019-01-15
3.4) Build server database		2018-10-30	2019-02-04
3.5) Implement Server		2019-02-04	2019-03-01
3.6) Configure Software		2018-10-30	2019-02-04
3.7) Design website		2018-10-30	2019-02-04
3.8) Website and data management		2019-02-04	2019-03-01
4.) Project Monitoring and Control		**2018-10-30**	**2019-03-01**
4.2) Test and Monitor RF components		2019-01-15	2019-03-01
4.3) Test and Monitor Server Hardware		2019-02-01	2019-03-01
4.4) Test and Monitor Server Database		2019-02-01	2019-03-01
4.5) Test and Monitor Website		2019-02-01	2019-03-01
5.) Project Closure		**2019-02-15**	**2019-03-29**
5.1) System Documentation		2019-02-15	2019-03-29



4.3 Personnel Time

Team Member	Total Hours
Kevin	700+ hours
Zac	450 hours
Jon	450 hours
Hunter	450 hours

4.4 Intellectual Property

- All data products (raw data, computed statistics, plot images) are available for download and usage according to Creative Commons CC BY 4.0.
- CC BY 4.0 allows for anyone to share the data products, as well as remix them for any purpose, including commercial purposes under the following terms: the licensee must give appropriate credit, provide a link to the license, and indicate if any changes were made to the data products. Additionally, the licensee must not apply legal or technological measures that restrict others from doing anything that the CC BY 4.0 license permits.

4.5 Differences between planned and finished product

- Data export: Rather than pre-generating downloadable chunks, .CSVs are created on-demand.
- NMEA did not have suitable precision for our application, we had to switch to the proprietary UBX protocol.
- Server upgrade was necessary to support new features and ensure system capacity for years to come.

5. Overall Conclusions and Recommendations

5.1 Summary of how well the project met specifications and expectations

- The project met all specifications
- The project exceeded expectations as many “wishlist” items were also completed
  - Automated notifications sent via email in the event of errors
  - Historical data viewer

5.2 Percentage of Completion

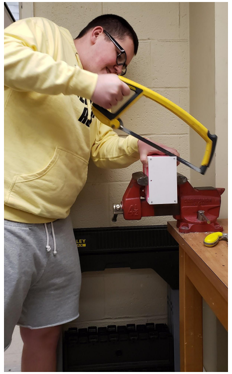
	Functionality	Performance	Operation
Data Plotting	100%	100%	100%
Job Scheduler	100%	100%	100%
Data Collection	100%	100%	100%
Website	100%	100%	100%
Management Systems	95%	100%	100%

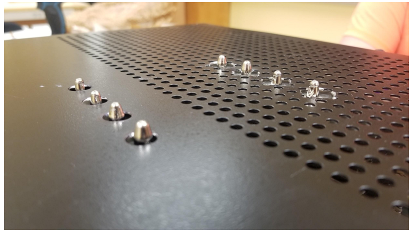
5.3 - Recommendations for future work

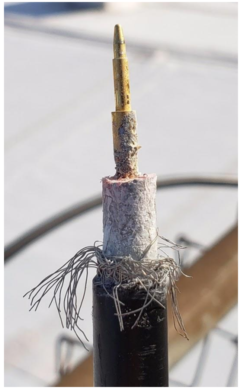
1. Better receivers / Redesign data collection
2. Establish another monitoring location -- preferably 100s of miles away
3. Optimize database operations as needed
4. Improve website design, perform UX/UI studies
5. New server with real server hardware, redundant PSUs, etc.
6. Future iterations should be a team of CpEs or CS students

5.4 Lessons Learned

- u-blox's UBX protocol is very difficult to interface with
- The survey proved to be much more complicated than expected
- Bugtracker / feature tracker spreadsheet was extremely helpful for tracking items
- Set up monitoring systems early to help catch bugs (memory leaks, runaway processes)
- Measure once, cut twice...







Software Bugtracker / Feature Tracker

Project To-Do

File Edit View Insert Format Data Tools Add-ons Help Last edit was 7 days ago

fx DONE?

	DONE?	Date Added	Date Finished	Assigned Person(s)	Relevant Subsystem(s)	Category (Bug/Feature)	Priority (Wish, Low, Medium, High, Critical)	Description	Status
2	NO	3/22/2019		Kevin	Linux	Feature	Medium	Set up drivers for temperature sensors, fix monitorix	
3	NO	2/1/2018		Kevin	Linux	Feature	Medium	Monthly emails with various information in it	Not started
4	NO	12/3/2018		Kevin	Linux	Feature	Medium	Udev rules for each receiver so their tty names dont change	Unsure why this is not working, worked previously
5	NO	3/11/2018		Kevin	dataPlotter	Feature	Medium	If there is no data available, output a png that says "no data available"	Lineplot logic is somewhat difficult, maybe move this i
6	NO	2/18/2019		Kevin	Linux	Feature	Low	Create backup image of boot drive	Waiting until last minute
7	NO	2/18/2019		Kevin	Linux/Database	Feature	Low	Create backup script for database using GDrive	In-progress
8	NO	2/17/2019		Kevin	Linux	Feature	Low	Unattended updates, get on LTS software as much as possible	Waiting until last minute
9	NO	1/23/2019		Kevin	Linux	Feature	Low	sendmail w/ GSuite for Fail2ban, environmental sensor alerts, etc	Partially done
10	NO	2/17/2019		Kevin	Linux	Feature	Low	SMART monitoring with email updates	https://wiki.archlinux.org/index.php/S.M.A.R.T.
11	NO	11/24/2019		Kevin	Linux	Feature	Low	System Startup Script	In-Progress
12	YES	1/15/2019	4/9/2019	Zac, Kevin	Database	Feature	High	After antenna survey, correct all old data by appropriate amount	DONE
13	YES	1/24/2019	3/26/2019	Kevin	Linux	Feature	High	Set up software to connect to UPS for auto-shutdown, etc	apcupsd is reporting COMMLOST
14	YES	2/11/2019	3/22/2019	Kevin	dataPlotter / functions / histogram	Bug	High	BeiDou histograms do not mask correctly	maxNum = max(yAsNPArray.max(), abs(yAsNPArray.r
15	YES	1/23/2019	3/21/2019	Kevin	dataExport, admin	Feature	Low	Special Access DataExport	let them query the DB with unlimited timespans. keep the files around a lot longer. allow access to full database dumps (must be triggere
16	YES	1/7/2019	3/21/2019	Kevin	dataExport	Feature	Low	Anti-Abuse: rate limiting, max data export folder size, disable submit button after submit, error handling	CAPTCHA prevents double submission, added error ha
17	YES	12/3/2018	3/21/2019	Zac	functions/dataPlotter	Feature	Medium	Plot normalized error with DOPs	Out of scope of this project
18	YES	3/11/2019	3/11/2019	Kevin	dataPlotter	Feature	Medium	Plot DOPs under respective error lineplot	Plots are overcrowded, this feature was left out.
19	YES	2/18/2019	3/11/2019	Kevin	jobScheduler	Feature	Medium	Configurable timeouts for each plotting job, plus lockfiles.	https://dreamix.eu/blog/webothers/timeout-function-in-p https://docs.python.org/2/library/multiprocessing.html https://stackoverflow.com/questions/26063877/python-
20	YES	12/3/2018	3/11/2019	Kevin	dataPlotter	Feature	Wishlist	Direct comparison of all constellations on all plots using subplots, with all time ranges	Kinda implemented via custom view, but would be nice
21	YES	3/10/2019	3/10/2019	Kevin	dataPlotter	Feature	Wishlist	Added historical data viewer	
22	YES	2/16/2018	3/10/2019	Kevin	dataPlotter	Feature	Wishlist	Skyplot (points, no tracks) for any constellation and timestamp	Would require collecting azimuth and elevation of each Decided not to do this, it is just way too difficult.
23	YES	12/3/2018	3/10/2019	Kevin	functions/ getENUDataFromDB	Feature	Low	Optimize transform from SQL datatype to Python lists	Probably best to stay with pymysql -- moving to Panda
24	YES	2/18/2019	3/10/2019	Kevin	getGGGBData_parallel	Bug	Medium	Mask not working properly, lines are drawn during missing data	
25	YES	2/16/2019	3/10/2019	Kevin	dataPlotter, website	Feature	Medium	Add 5, 10, and 20 year plotting options.	Probably need to do this with datashader and make a r Avoid downsampling / subsampling if possible.
26	YES	2/11/2019	3/10/2019	Kevin	dataCollection	Bug	CRITICAL	Glitches with BeiDou and Galileo position errors	Nothing to be done, just wait for the constellations to f
27	YES	2/16/2018	3/10/2019	Kevin	dataExport	Bug	Medium	PHP needs longer execution time	
28	YES	2/19/2019	2/19/2019	Kevin	dataPlotter, dataExport	Feature	High	Delay data by 24 hours	Per Braasch's request
29								PHP Errors in Data Export:	

6. Demonstration of Project

Website is online and is publicly accessible:

https://gnssperformancemonitor.com/

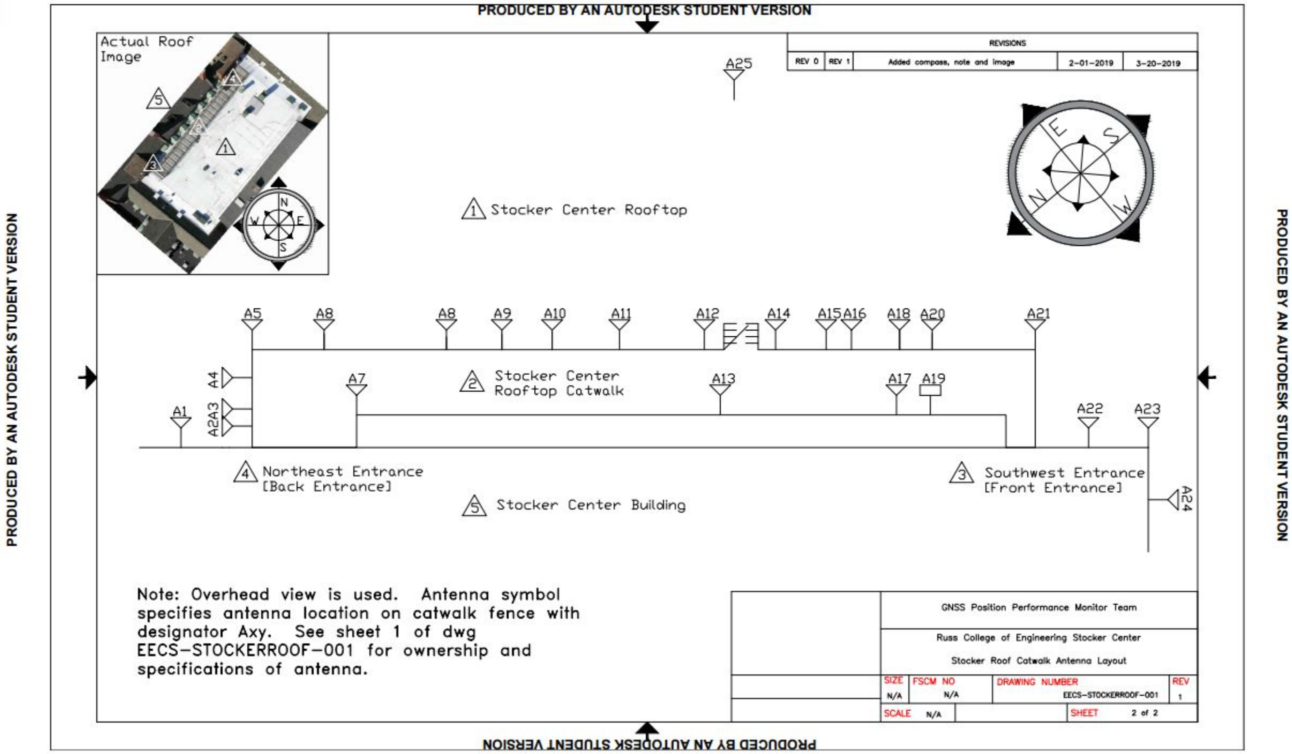
Please go check it out, browse around, give us feedback!

7. Questions

Thanks to our stakeholders for helping drive the project forward

Dr. Braasch
Daniel Allwine
Jason Wright
Tianyi Cai
Ohio University Russ College of Engineering
Ohio University - Avionics Engineering Center
GNSS Team EECS Senior Design
The international Positioning, Navigation, and Timing community

Extra Slides



Roof Diagram

2.2 Final Design Technical Results

Data Collection software continuously monitors output of receiver and performs several sanity checks, including:

- Drift rate and time delta between server time (NTP) and receiver time (constellation specific)
- Receiver oscillator drift
- Timing and frequency accuracy estimate
- Protocol specific checks (header integrity checks, etc)

```
---------------- Constellation: GPS ----------------
Receiver Timestamp: 2019-04-23 04:28:06  |  Server Timestamp: 2019-04-23 04:28:07.033683
Time delta: 1.0337 Sec | Time since startup: 534716.3073 Sec | Server-Receiver drift rate: 0.0000 s/s
Clock Bias: 97728ns / 97.728us  |  Receiver Internal Clock Drift: 65ns/s
Time accuracy estimate: 4ns  / 0.004us | Frequency Accuracy Estimate: 639ns

PVT epoch:188904.0
East: 0.6477695451775538  |  North: 0.949632707047962   |  Up: -1.668000104816413
LatLon: 39.3260865,-82.1070469 Alt 205.511
NumSV: 7

DOP epoch:188904.0
EDOP: 0.73 | NDOP: 1.1 | VDOP: 1.99
HDOP: 1.33 | PDOP: 2.39 | GDOP: 2.75 | TDOP: 1.35

SVINFO epoch: 188904.0  SVs: 1;2;5;7;13;15;16;20;21;25;26;29;30

Database ops time: 0.004344 Seconds
Messages processed: 530877
-------------------------------------------------------
```

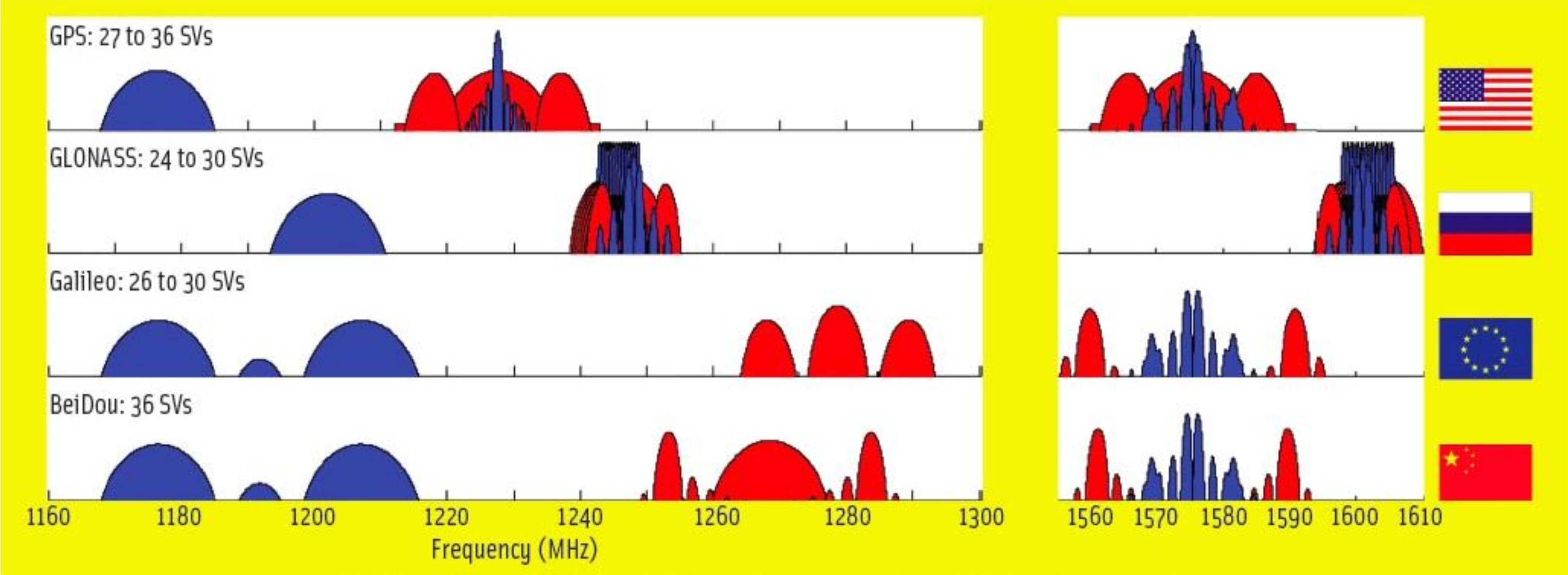
2.2.4 Software Details (Job Scheduler)

- Job Scheduler is based on the open source ‘schedule’ module
- Modifications were made to add support for multiple concurrent jobs, an external configuration file, and job timeouts.
- Supports email notifications in case of failed or aborted job

```
schedule.every(30).minutes.do(run_multiprocessing, plot_GNSS, '1Day', int(jobConfigurationData['1DayTimeout']))
schedule.every(5).hours.do(run_multiprocessing, plot_GNSS3D, '1Day', int(jobConfigurationData['1Day3DTimeout']))
schedule.every(2).hours.do(run_multiprocessing, plot_GNSS, '1Week', int(jobConfigurationData['1WeekTimeout']))
schedule.every(14).hours.do(run_multiprocessing, plot_GNSS3D, '1Week', int(jobConfigurationData['1Week3DTimeout']))
schedule.every(11).hours.do(run_multiprocessing, plot_GNSS, '1Month', int(jobConfigurationData['1MonthTimeout']))
schedule.every(15).hours.do(run_multiprocessing, plot_GNSS, '1Year', int(jobConfigurationData['1YearTimeout']))
schedule.every(1).hours.do(run_multiprocessing, plot_GNSS, 'AllData', int(jobConfigurationData['AllDataTimeout']))
schedule.every(1).minute.do(run_multiprocessing, cleanup_exports, 'cleanup', int(jobConfigurationData['cleanupTimeout']))
schedule.every(5).minutes.do(run_multiprocessing, runHistoricalPlotter, 'historical', int(jobConfigurationData['historicalPlotterTimeout']))
```

Stocker Roof

Signal Structures and Band Frequency for Constellation



Original Image Credit: http://insidegnss.com/something-old-something-new/ Modified by KMC

Stocker Attic Coax Conduit and Demarcation Point