GOES-16 Satellite Receiver

SDmay20-03

Advisor/Client: Dr. Nathan Neihart

Members: Ted Mathews, Jonathan Massner, Riley Stuart, Yong Yi “Rudy” Lim, Nick Butts, Jordan Tillotson

http://sdmay20-03.sd.ece.iastate.edu/

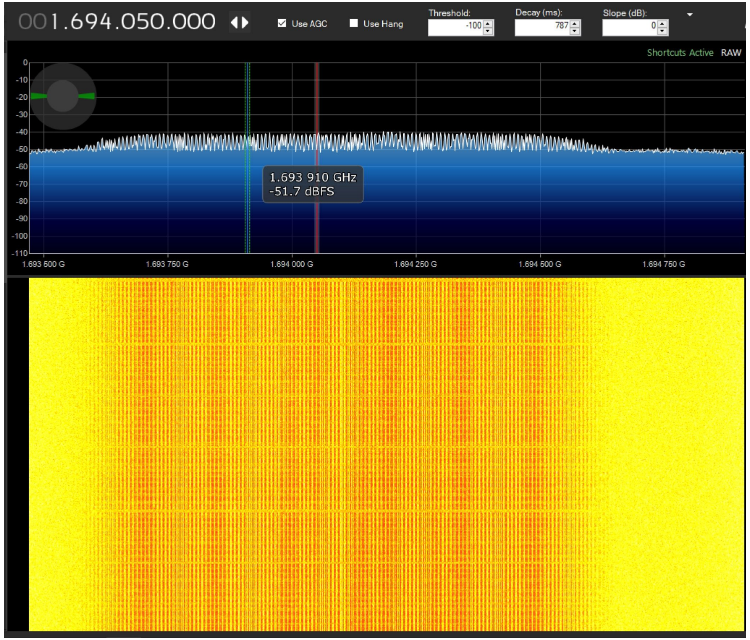
Project Plan

High Level Overview of the Project

The goal of this project is to receive, decode, and display weather products from NOAA's GOES-16 weather satellite.

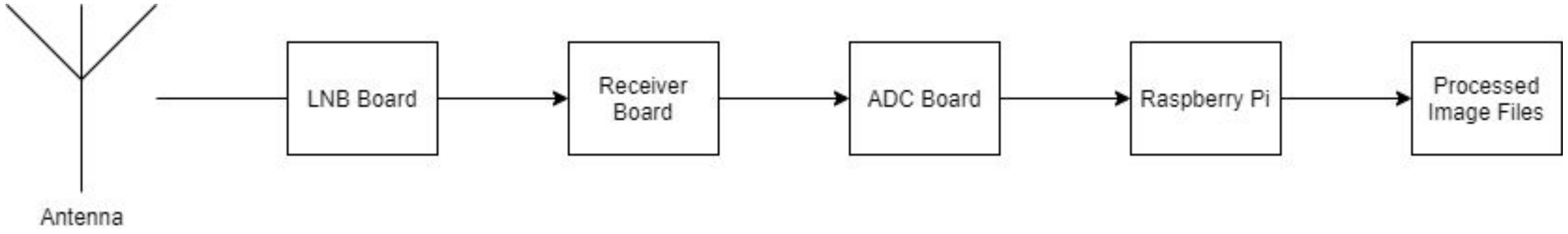
This project involves:

- Receiving and demodulating the analog signal (RF team)
- Digitizing the signal and transferring the data to Raspberry Pi 4 (ADC team)
- Decoding the data and image construction (software team)

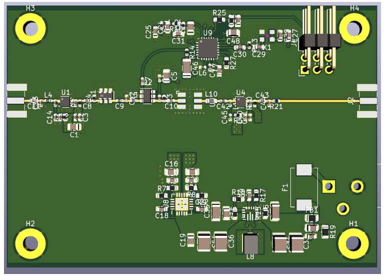


Spectrum of GOES-16 signal as received by our antenna

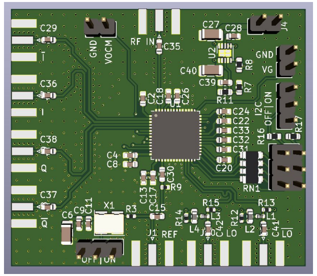
Conceptual Sketch



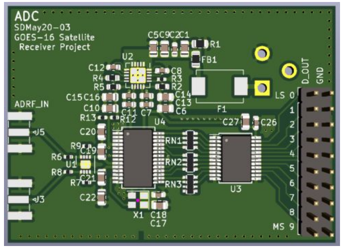
Antenna



LNB



Receiver



ADC

Rpi and SDR

Requirements

Functional

- The system must download a current Earth image from the GOES-16 satellite
- RPi4 will host a website to display information about the downloading process and full images

Non-functional

- RF section:
    - System must receive 1694.1MHz signal with at least 1.205MHz of bandwidth from NOAA's satellite and output a baseband signal

- ADC section:
    - System must digitize the baseband signal and output received binary to Raspberry Pi, binary data stream must be stored on Raspberry Pi4 for further processing
- Software section:
    - Software must receive the binary information from the ADC, decode the information, and generate an image file

Technical Considerations

Receiver architecture takes advantage a mix of SDR and superheterodyne strengths

ADC capabilities

- Data rate
- Resolution
- Sampling rate
- Communication

Software must compile and run on Raspbian

Computation limitations of RPi4

- If the program is not well optimized the runtime can be unacceptably long

Data stream storage

Potential Risks and Mitigation

Completion Risks

- Risk to hardware exists by improper connection and powering of system components.
- Signal strength matching - ensuring that the boosted signal is within tolerance for other system components.

Physical and Environmental Risks

- Low - Physical risk present when improperly transporting equipment such as the antenna.
- Risk of injury during PCB assembly
- Risk of damage to PCBs and test equipment from ESD events

Risk Mitigation

- Ensure AC coupling on inputs and outputs for RF systems and observe proper ESD mitigations
- Through calculations, testing, and datasheet analysis, ensure signal strength levels are appropriate
- Properly store and transport the system within Coover.

Resource/Cost Estimate

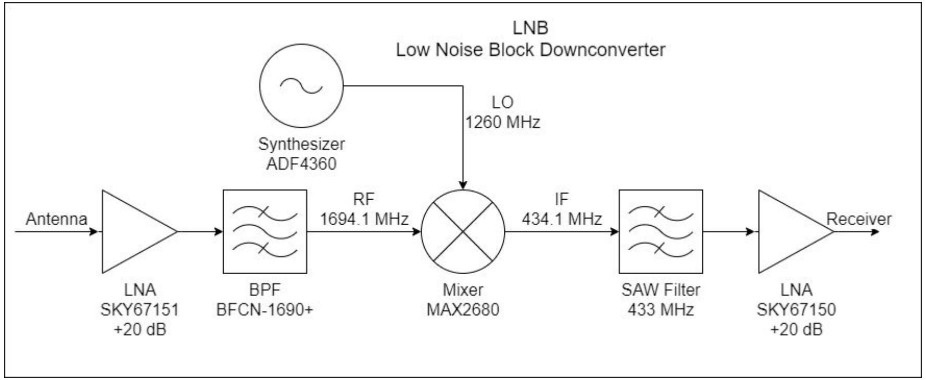
- Most of the costs are for hardware
  - Antenna
  - RF Boards, Components, RF Cables, and Connectors
  - ADC board and filtering components
  - Raspberry Pi 4

- Software is open-source
  - libcorrect from Github
  - Source code will be implemented in parts on an as needed basis

- Total cost estimate is approximately
  - 650$

System Design

Detailed Design

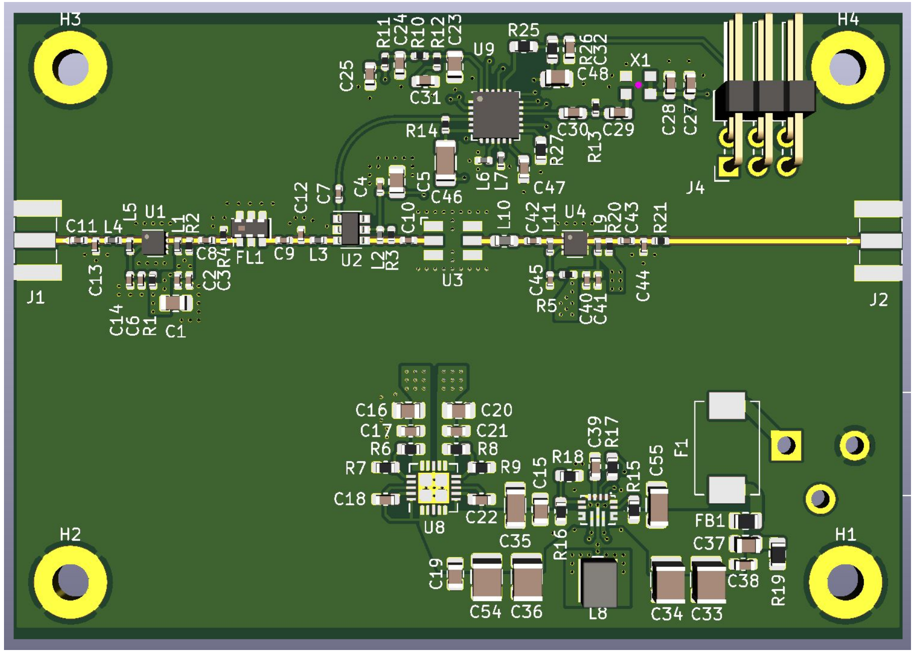


Analysis

Output Power (rms)	-33.67	dBm
Output Voltage (rms)	4.63	mVrms
Output Voltage (pp)	13.09	mVpp
OP1dB	15.49	dBm
IP1dB	-29.8	dBm
Power Gain	46.33	dB
Voltage Gain	46.33	dB

Noise Figure	0.84	dB
Output NSD	-126.8	dBm/Hz
Output NSD	101.9	nV/rtHz
Output Noise Floor	-63.8	dBm
SNR	30.1	dB
Input Rx Sensitivity	-100.1	dBm

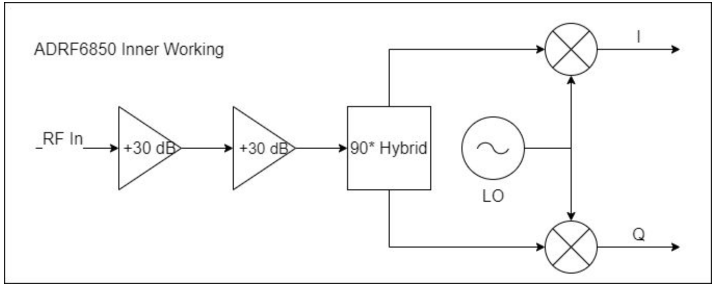
OIP3	15.68	dBm
IIP3	-30.6	dBm
IMD3 ((Pin-3dB) per tone)	-104.7	dBc
SFDR	52.9	dB
ACLR (est.)	-30	dB
Pwr Consumption	0.79	W

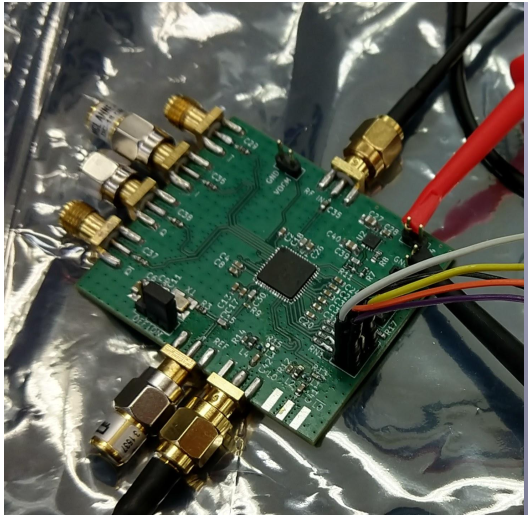


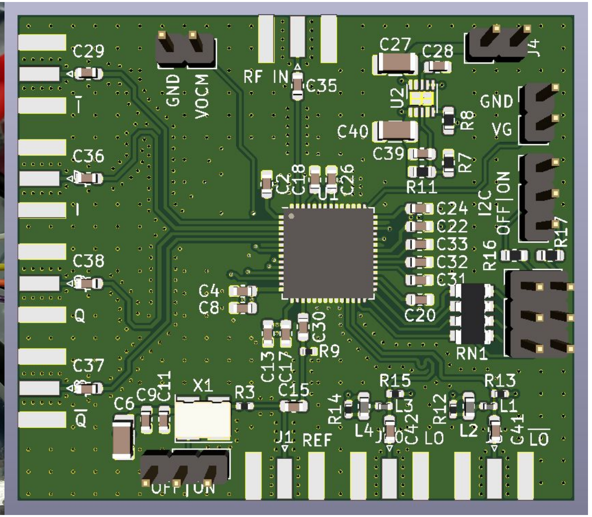
LNB

- ~46 dB of gain
- 0.84 dB NF
- ~30 dB SNR

Detailed Design



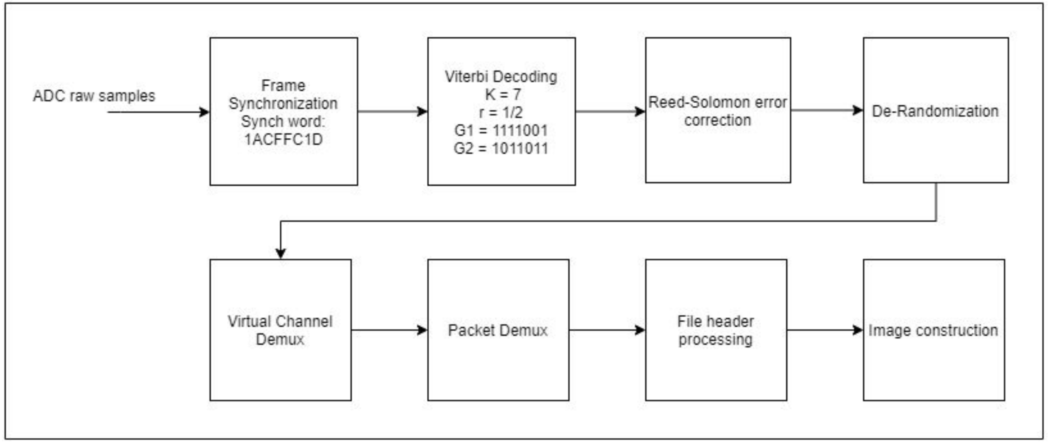




Receiver

- Input: 433.1 MHz from LNB
- Up to 60 dBo of gain
- ~ 11 dB noise figure at 39 dB of gain

Detailed Design



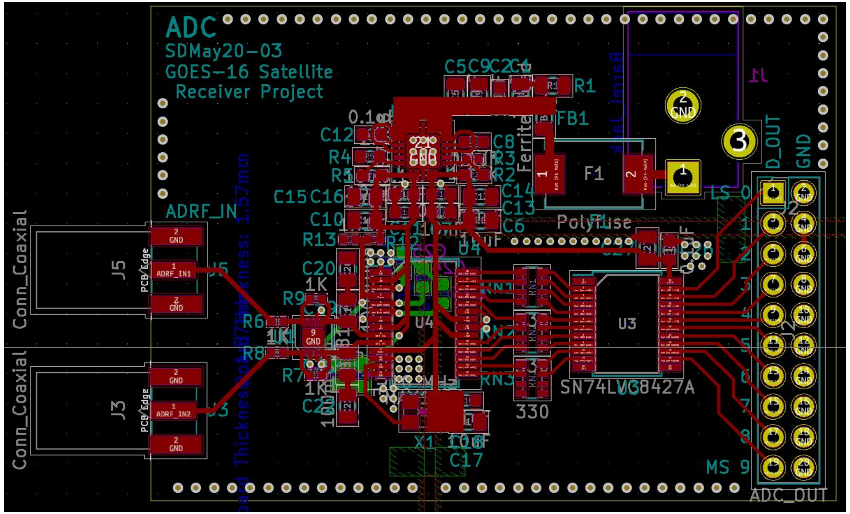
ADC and the Raspberry Pi

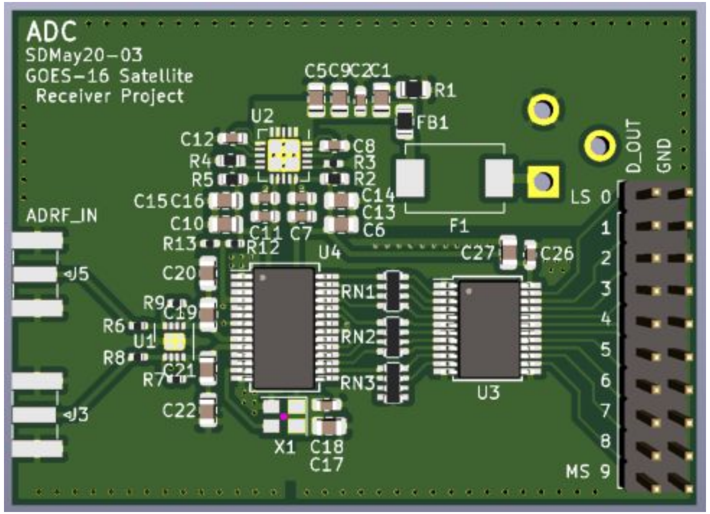
- **Input**: analog I & Q data from the ADRF6850
- **Output**: binary data stream to RPi4 for further processing

Software on the Raspberry Pi

- **Input**: 8 bit raw samples from the ADC
- **Output**: full-disk images of Earth

Detailed Design





- Process began with designing a Sallen-Key low-pass filter built around a fully-differential op-amp
- MAX1426 ADC chip was chosen because of its bit-depth and sampling capabilities (10-bit, 10 Msps)
- Component choices based on low noise and compatibility
- MAX1425/1426 Eval board documentation used for layout reference

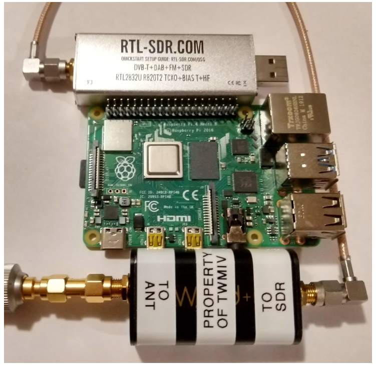
Hardware Platforms Used

Prototyping

- STM32F303 Nucleo (microcontroller development board)
    - Changed to MAX1426 DS
- Raspberry Pi3/Pi4
- ADRF6850 I/Q Demodulator development board

Final Implementation

- ADRF6850 I/Q Demodulator
- LCOM 2.4GHz 24 dBi Dish Antenna
- SKY67151 LNA
- BFCN-1690 BPF
- MAX2680 Mixer
- ADF4360 Synthesizer
- MAX1426 DS ADC
- Raspberry Pi4

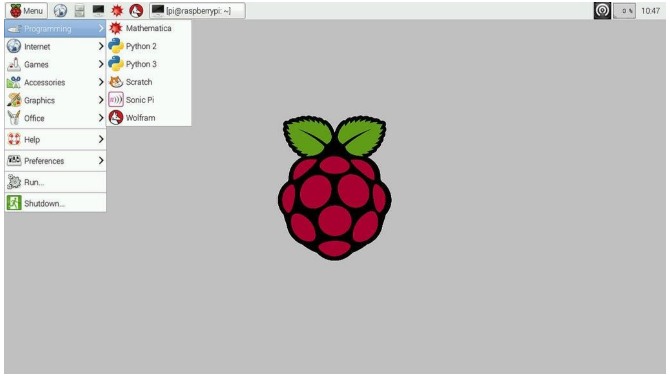


LCOM 2.4GHz 24 dBi Dish

Software Platforms Used

Raspbian OS

- Operating system for Raspberry Pi4
- Code::blocks for cross-platform software development
- Python for packet demux and image construction



Test Plan

RF Team

- 3 stages of testing for both the LNB and receiver.
  - **Electrical:** Ensuring that the board was assembled properly. Checking power supplies, IC outputs, and signal levels.
  - **Functional:** Inputting generated RF signals to determine the performance of the signal path.
  - **Operational:** Attach LNB and Receiver to the antenna and view the received signal on a spectrum analyzer.

ADC Team

- **Receiver to ADC:** Spice design and testing of Sallen-key lowpass filter (fc ~ 1.5MHz), interface with the follow-on ADC and PLL circuitry and test in Spice simulations
- **ADC to PI:** Test arbitrary 1.2 MHz signal coming into GPIO of Pi, ensuring that a binary file with correct bit formatting will be written in the desired storage location

Software Team

- Used sample data file from Lucas Teske's blog to test code as it was developed

Prototype Implementations

RF Team

- Antenna assembled, tested with SDR
- Receiver prototype has been implemented and tested
- LNB board schematic and layout finished

ADC Team

- First prototype included the use of an STM32 microcontroller for ADC and communication
- Second approach removed the use of the microcontroller and replaced it with a discrete ADC
    - ADC board schematic and layout finished

Software Team

- Frame synchronization, Viterbi decoding, Derandomization, Virtual channel demuxer functional
- Packet demuxer script tested but not fully functional

Engineering Standards and Design Practices

- Agile Development
- IEEE 145-2013: IEEE Standard for Definitions of Terms for Antennas
- IEEE 149-1979: IEEE Standard Test Procedures for Antennas
- IEEE Standard 211-2018: IEEE Standard Definitions of Terms for Radio Wave Propagation

Conclusion

Project Status

RF Team

- **Antenna**
  - Antenna assembled, tested, and functional
- **Receiver**
  - Board fabricated, testing began, error found and fixed, further testing halted
- **LNB**
  - Schematic and layout complete

ADC Team

- ADC board schematic and layout completed
- Upon course instruction moving to strictly online, the ADC team was able to focus more time assisting the Software team

Software Team

- Frame synchronization, Viterbi decoder, de-randomization completed and tested
- Virtual channel demuxer functional and outputs demuxed channel data in binary files
- We have identified Python scripts that may do the rest of the processing, but were unable to get it working

Team Member Contributions

Ted Mathews - RF Design and Testing

- SME, All system component recommendations, Antenna tripod build, Antenna assembly and modification, RF circuit design and layout

Jonathan Massner - RF Design and Testing, System Design

- Antenna assembly, RF circuit design and layout, system integration, managerial tasks

Riley Stuart - ADC Design and Testing

- Initial implementation, configuration, testing of microcontroller; data transfer and communication from ADC to PI

Jordan Tillotson - ADC Design and Testing

- STM32F303 Nucleo communication with Raspberry Pi4, ADC circuit design and layout

Nick Butts - Software/DSP Development

- Wrote program to do frame synch, decode, de-randomize, and demux virtual channels

Yong Yi Lim - Software/DSP Development

- Finished Reed-Solomon Error Correction and Viterbi Decoder Code

Future of the Project

- Project will be carried forward into future semesters and completed by a different team.
- All existing documentations and code have been uploaded to GitLab for use.
- Substantial progress has been made in all areas of the project and everything has been documented well, so we are confident another Senior Design team can finish this project.

Works Cited

Teske. “Satellite Projects.” *Lets Hack It*, 28 Oct. 2016, lucasteske.dev/satcom-projects/satellite-projects.

Quiet. “Quiet/Libcorrect.” *GitHub*, 10 Oct. 2018, github.com/quiet/libcorrect.

Extra Slides