# Borealis v1.0 – Clear Frequency Search Implementation

D. Galeschuk[1], R. Rohel[1], T. Kolkman[1], S. Marei[1], P. Ponomarenko[1], G.C. Hussey[1], K.A. McWilliams[1]

[1] Institute of Space and Atmospheric Studies (ISAS)
Department of Physics and Engineering Physics

University of Saskatchewan SuperDARN Canada

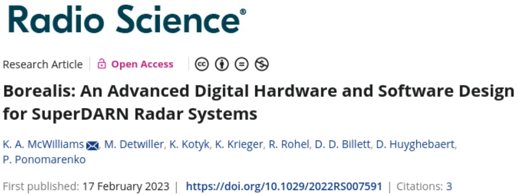
Saskatoon, SK Canada

**SuperDARN Workshop 2025 Roanoke, VA, USA, 2 June – 6 June 2025**

# Outline

- What is Borealis
- Designing Clear Frequency Search
- Staged Filtering
- Correcting Pulse Generation
- CFS Scan Results

# What is Borealis?



Radio Science®

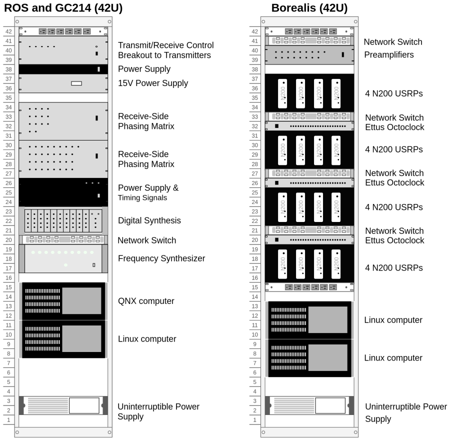
Research Article | Open Access

**Borealis: An Advanced Digital Hardware and Software Design for SuperDARN Radar Systems**

K. A. McWilliams, M. Detwiller, K. Kotyk, K. Krieger, R. Rohel, D. D. Billett, D. Huyghebaert, P. Ponomarenko

First published: 17 February 2023 | https://doi.org/10.1029/2022RS007591 | Citations: 3

- Software Defined Radio (SDR) system
- Replaces previous ROS system
- Utilizes digital signal modulation
- Enables more experiment flexibility
- Open source radio operation software



# New Radar Capabilities

SDRs expand the capability of our radar operations – All signal generation and modulation is done in software!

New experiment capabilities include:

- Full field-of-view (wide beam) experiments
- Simultaneous multi-frequency transmission
- Bi-static operation modes

Limited now only by the SDR transceiver and what we can program

## Clear Frequency Search

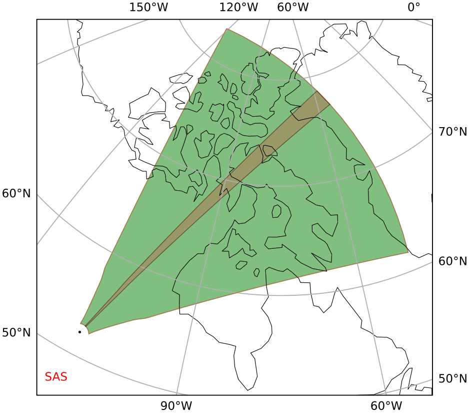
Development of a Clear Frequency Search (CFS) mode was a requirement for the first major release of Borealis

Clear Frequency Search is a mode that scans a frequency band and selects a transmit frequency from that band that has the lowest measured power and therefore least RF interference

- Important for meeting radio license requirements
- Provides information about the RF noise in the band
- Helps to avoid signal interference
- Was implemented in ROS

# Processing Chain

- Instead of the first pulse sequence perform Rx only scan
- Filter, down-sample, and beamform data
- Selects lowest power frequency bin
- Sets Tx to nearest kHz frequency
- Frequency is determined for each beam
- User control over frequency resolution of scan

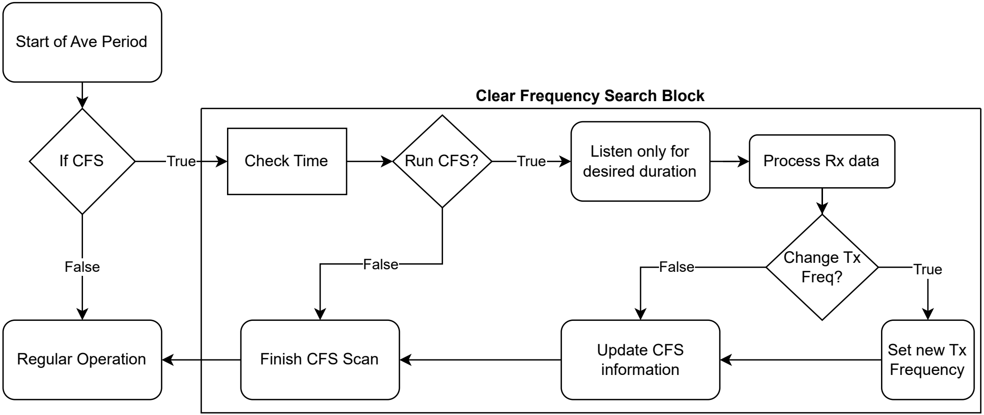


# User Parameters

Users designing experiments using CFS have control over the parameters:

| | |
|---|---|
| Band | Sets frequency band to search |
| Duration | Time to listen in RX only |
| Filter | Filter method for band analyzed |
| Stable Time | Tx frequency hold duration |
| Always Run | Toggle whether to always run even if Tx held |
| Power Threshold | Required power difference before Tx is changed |
| Frequency Resolution | Sets desired frequency bin size |

# Flowchart


Start of Ave Period
If CFS
True
False
Clear Frequency Search Block
Check Time
Run CFS?
True
Listen only for desired duration
Process Rx data
Change Tx Freq?
False
True
False
Set new Tx Frequency
Update CFS information
Finish CFS Scan
Regular Operation

# Challenges Encountered

Issues with development:

- Extra processing load was overloading GPU memory
- Extra processing time reduced number of pulse sequences in an averaging period
- Velocity data was skewed when Tx was not an interger multiple of 10 kHz

# Filtering Scheme

The staged filtering scheme represents how Borealis filters and down-samples Rx data to a final result

- N200 SDRs operated at 5 MHz receiver bandwidth
- Staged filtering is applied to isolate desired signal
- Staged down-sampling from 5 MHz to 3.3 KHz

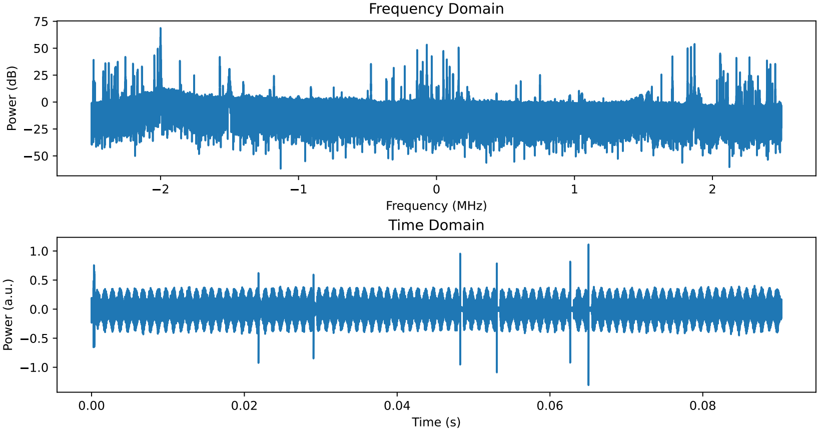
# New Filtering Scheme

- Switch from a 4-stage to 2-stage filtering scheme
- Faster and uses less processor memory
- Slightly higher side lobes (140 dB vs 200 dB down from the main lobe)

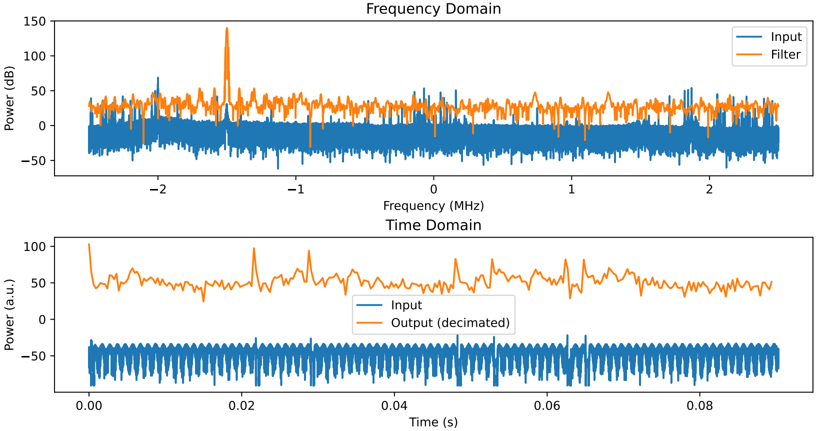
Performance on lab computer:

- 4-stage filter $= 22\ ms$
- 2-stage filter $= 10\ ms$

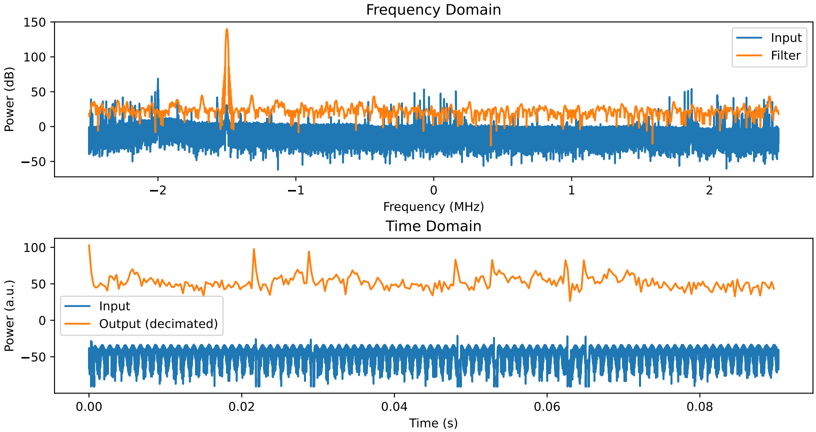
# Example 7 Pulse Sequence - 10.5 MHz Tx



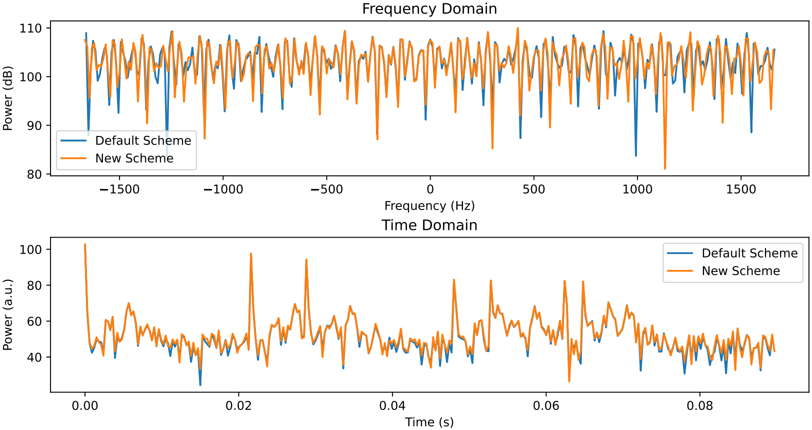
# Example 7 Pulse Sequence - 4 Stage Filter



# Example 7 Pulse Sequence - 2 Stage Filter



# 4 and 2 Stage Filter Result Comparison

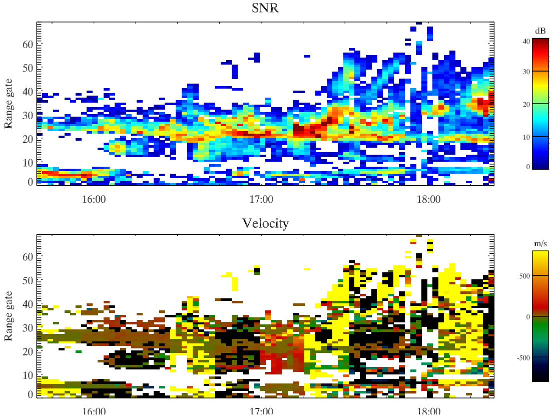


# Nonphysical Velocities

Initial CFS experiment SNR data looked great

However...

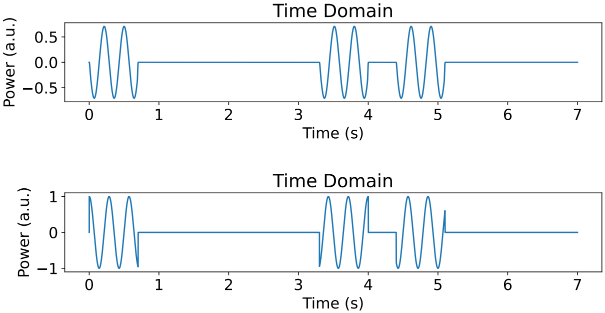
Velocity data was unreasonable



# Pulse Generation

The problem is a fundamental issue in the way pulses were generated:

- Borealis built the pulse sequence using identical copies of a pulse
- In physical RF hardware, the source of the pulse would be sampling from a continuous oscillator

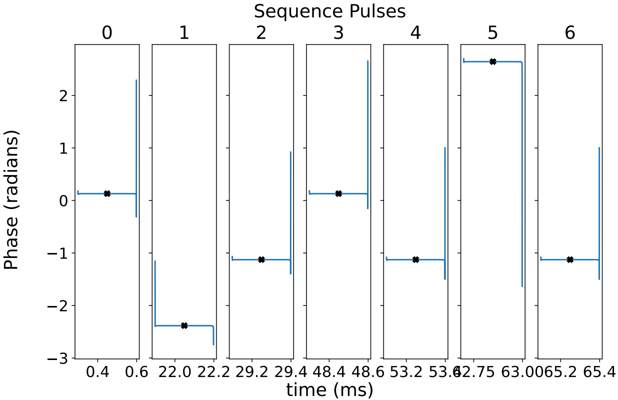


## Pulse Phase Correction

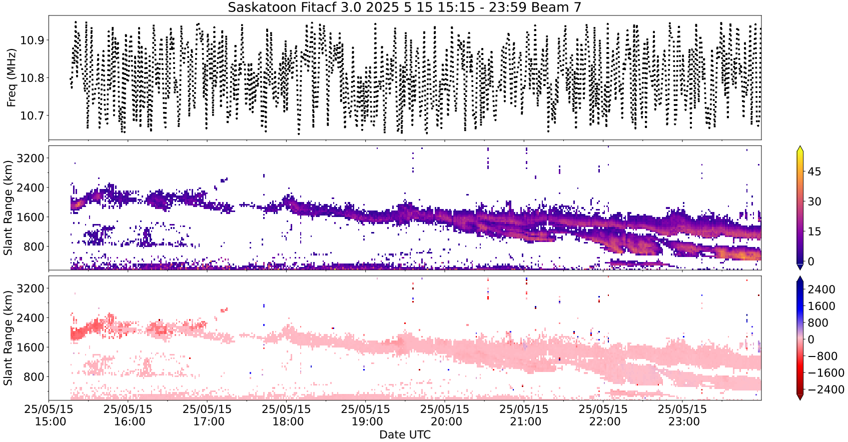
Implication: Downshifting Borealis pulses imposes a phase offset

- Phase pulse, $\phi$, should be is easy to predict
- Depends on minimum pulse separation, $\tau$, and Tx frequency, $f$

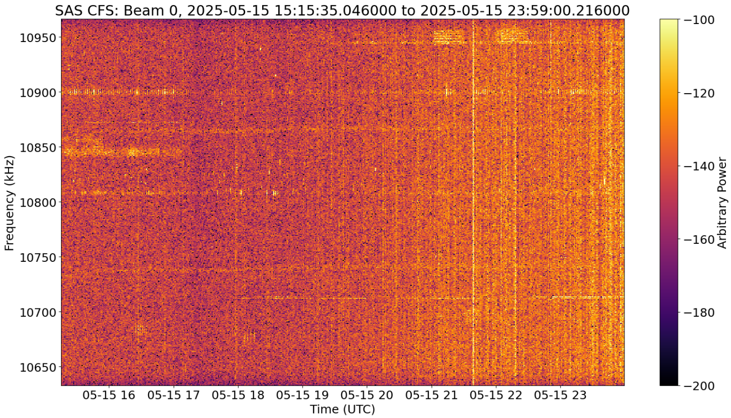
$$\phi = exp(2\pi f * n\tau * 1i)$$



# Fixed Velocities



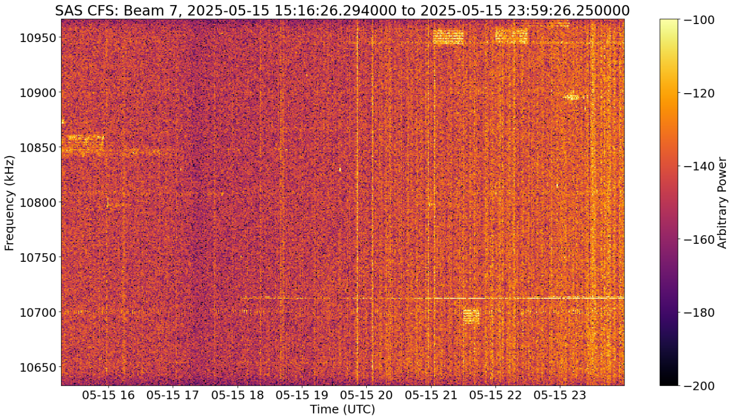
# Example Scan: Beam 0 SAS



## Run parameters

- Range $= 300\ kHz$
- Duration $= 90\ ms$
- Freq Res $= 1\ kHz$

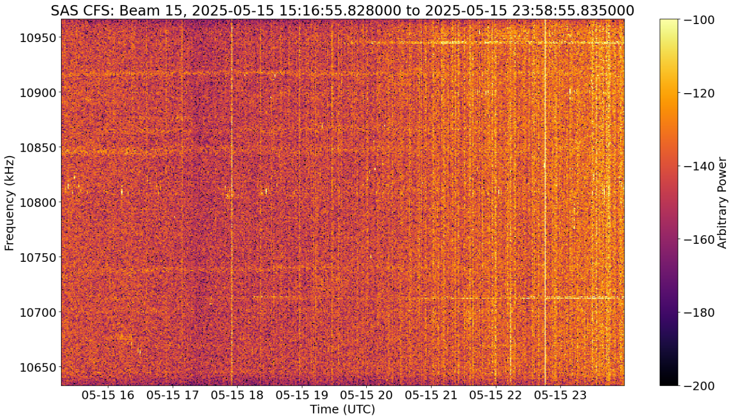
# Example Scan: Beam 7 SAS



## Run parameters

- Range = 300 *kHz*
- Duration = 90 *ms*
- Freq Res = 1 *kHz*

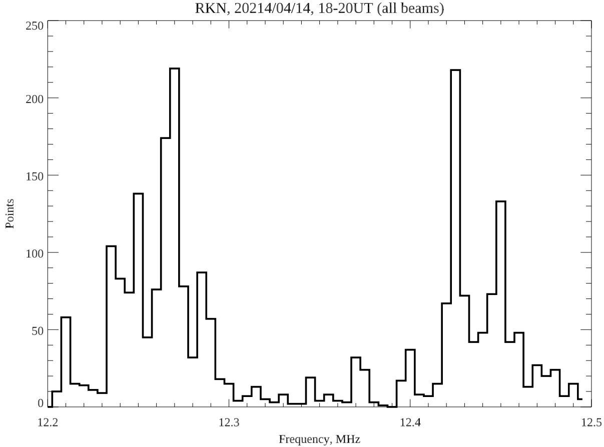
# Example Scan: Beam 15 SAS



## Run parameters

- Range $= 300\ kHz$
- Duration $= 90\ ms$
- Freq Res $= 1\ kHz$

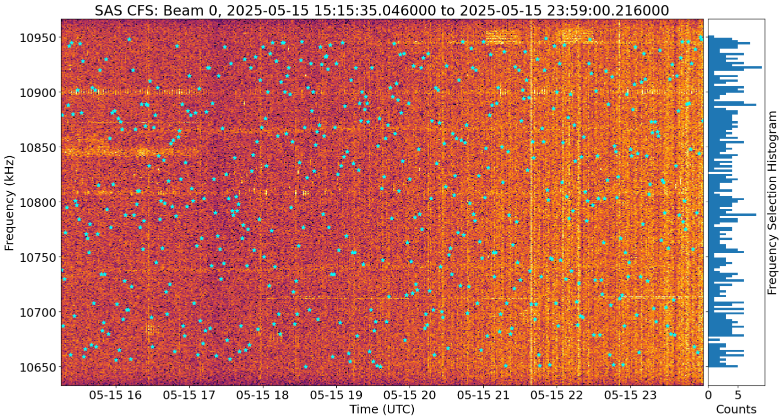
# ROS CFS Frequency Selection



Run parameters

- Range = 300 *kHz*
- Freq Res = 5 *kHz*

# Borealis CFS Frequency Selection



Run parameters

- Range = 300 *kHz*
- Duration = 90 *ms*
- Freq Res = 1 *kHz*

The first major release of Borealis comes with CFS, improved performance, and many other improvements!

For more information on Borealis see

`https://borealis.readthedocs.io/en/latest/`

`https://github.com/SuperDARNCanada/borealis`

# Protobuf Removal

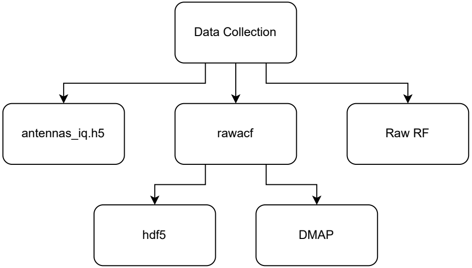
Protobuf is a platform neutral software package developed by google to enable data transfer between different programming languages

- Protobuf was used to support data messages between python and C++ processes
- Bulky program adding unneeded bloat to Borealis

Borealis messages are small and require little overhead. The largest data transfer is sending pulse information from python to C++ which was slowing operation during complicated experiments

- Redesigned message transfer to use shared memory
- Messages now built 20 times faster
- Allows more complicated pulse configurations and encoding without performance costs

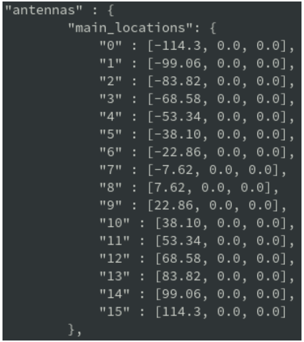
# File Generation



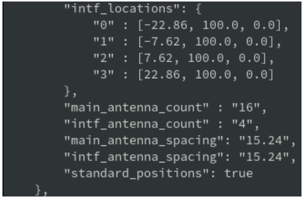
- Array style files no longer generated
- All files are generated in the site style and follow the hdf5 file format
- Borealis can write DMAP rawacfs in real-time
- File data is now indexed by the data type (ie: ’antennas_iq’ instead of a generic ’data’ label)

# Changes to Configuration Files

- Explicit antenna location is now required
- Antenna metadata saved in data files
- N200 indexing simplified
- Rx and Tx channel connections to antenna chains are explicitly defined now

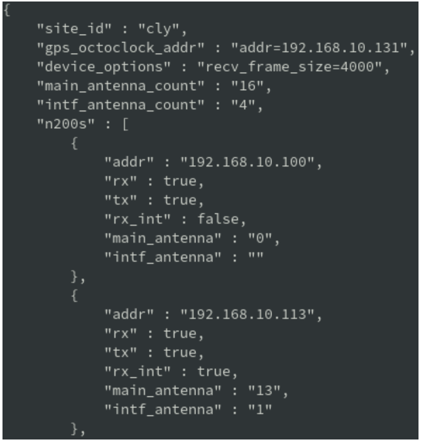


```
"antennas" : {
        "main_locations": {
            "0" : [-114.3, 0.0, 0.0],
            "1" : [-99.06, 0.0, 0.0],
            "2" : [-83.82, 0.0, 0.0],
            "3" : [-68.58, 0.0, 0.0],
            "4" : [-53.34, 0.0, 0.0],
            "5" : [-38.10, 0.0, 0.0],
            "6" : [-22.86, 0.0, 0.0],
            "7" : [-7.62, 0.0, 0.0],
            "8" : [7.62, 0.0, 0.0],
            "9" : [22.86, 0.0, 0.0],
            "10" : [38.10, 0.0, 0.0],
            "11" : [53.34, 0.0, 0.0],
            "12" : [68.58, 0.0, 0.0],
            "13" : [83.82, 0.0, 0.0],
            "14" : [99.06, 0.0, 0.0],
            "15" : [114.3, 0.0, 0.0]
        },
```



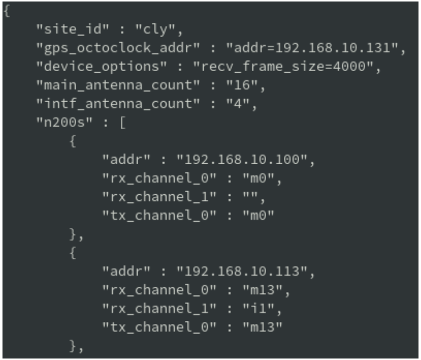
```
        "intf_locations": {
            "0" : [-22.86, 100.0, 0.0],
            "1" : [-7.62, 100.0, 0.0],
            "2" : [7.62, 100.0, 0.0],
            "3" : [22.86, 100.0, 0.0]
        },
        "main_antenna_count" : "16",
        "intf_antenna_count" : "4",
        "main_antenna_spacing": "15.24",
        "intf_antenna_spacing": "15.24",
        "standard_positions": true
    },
```

# USRP Channel Configuration



```
{
    "site_id" : "cly",
    "gps_octoclock_addr" : "addr=192.168.10.131",
    "device_options" : "recv_frame_size=4000",
    "main_antenna_count" : "16",
    "intf_antenna_count" : "4",
    "n200s" : [
        {
            "addr" : "192.168.10.100",
            "rx" : true,
            "tx" : true,
            "rx_int" : false,
            "main_antenna" : "0",
            "intf_antenna" : ""
        },
        {
            "addr" : "192.168.10.113",
            "rx" : true,
            "tx" : true,
            "rx_int" : true,
            "main_antenna" : "13",
            "intf_antenna" : "1"
        },
```

→

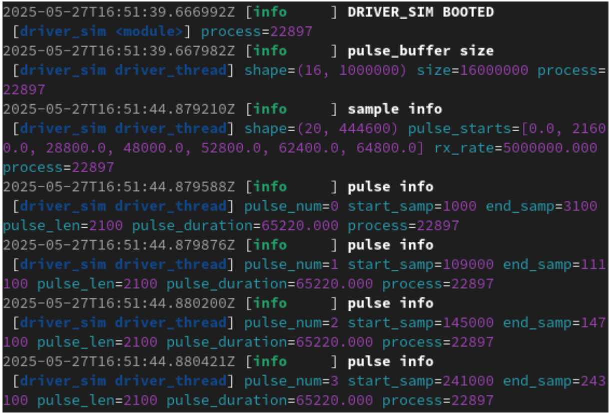


```
{
    "site_id" : "cly",
    "gps_octoclock_addr" : "addr=192.168.10.131",
    "device_options" : "recv_frame_size=4000",
    "main_antenna_count" : "16",
    "intf_antenna_count" : "4",
    "n200s" : [
        {
            "addr" : "192.168.10.100",
            "rx_channel_0" : "m0",
            "rx_channel_1" : "",
            "tx_channel_0" : "m0"
        },
        {
            "addr" : "192.168.10.113",
            "rx_channel_0" : "m13",
            "rx_channel_1" : "i1",
            "tx_channel_0" : "m13"
        },
```

# Experiment Configuration Changes

- Tx and Rx USRP center frequency can now be automatically determined
- Backend parameter type and value enforcement using updated pydantic vesion
- Clear Frequency Search mode can be used in experiments

# USRP Simulation & Filter Notebook



```
2025-05-27T16:51:39.666992Z [info     ] DRIVER_SIM BOOTED
 [driver_sim <module>] process=22897
2025-05-27T16:51:39.667982Z [info     ] pulse_buffer size
 [driver_sim driver_thread] shape=(16, 1000000) size=16000000 process=
22897
2025-05-27T16:51:44.879210Z [info     ] sample info
 [driver_sim driver_thread] shape=(20, 444600) pulse_starts=[0.0, 2160
0.0, 28800.0, 48000.0, 52800.0, 62400.0, 64800.0] rx_rate=5000000.000
process=22897
2025-05-27T16:51:44.879588Z [info     ] pulse info
 [driver_sim driver_thread] pulse_num=0 start_samp=1000 end_samp=3100
pulse_len=2100 pulse_duration=65220.000 process=22897
2025-05-27T16:51:44.879876Z [info     ] pulse info
 [driver_sim driver_thread] pulse_num=1 start_samp=109000 end_samp=111
100 pulse_len=2100 pulse_duration=65220.000 process=22897
2025-05-27T16:51:44.880200Z [info     ] pulse info
 [driver_sim driver_thread] pulse_num=2 start_samp=145000 end_samp=147
100 pulse_len=2100 pulse_duration=65220.000 process=22897
2025-05-27T16:51:44.880421Z [info     ] pulse info
 [driver_sim driver_thread] pulse_num=3 start_samp=241000 end_samp=243
100 pulse_len=2100 pulse_duration=65220.000 process=22897
```

- ▶ Improved simulation and testing
- ▶ USRP emulator for testing
- ▶ Filter notebook for developing and understanding decimation schemes
- ▶ Updated decimation scheme

## v1.0 Release Notes Summary

Borealis code: `https://github.com/SuperDARNCanada/borealis`
The features of the v1.0 release are:

- Protobuf dependency removed
- xarray-compatible data files
- Improved testing and simulation scripts
- apcupsd scripts for radar shutdown/startup on power loss/restoration
- daemon created for local_scd_server.py automation
- txdata file format removed
- ruff used for code linting and formatting
- Config file changes
- Pydantic dependency updated to v2
- Millisecond boundary sequence start
- Automatic RX/TX center freq selection
- TX samples sent to usrp_driver via shared memory
- Bug fixed that affecting frequencies that were not multiples of 10 kHz
- Improved default filter scheme
- Clear frequency search