Ten Beam Receiver Subsystem (TBRS) Digital Receiver (DR) Simulation

Gary Stephens
Son Trinh

October 14, 2009

CONTROL AND SIGNAL PROCESSING UPGRADE (CSPU)
SYSTEM ENGINEERING AND SUSTAINMENT INTEGRATOR (SENSOR)
SUBCONTRACT NO. 40316, CLIN 0318

Prepared for:

SENSOR
ITT Corporation Systems Division
4450 East Fountain Boulevard
Colorado Springs, CO 80916-2153

Prepared by:

Technology Service Corporation
Phase IV Systems Operation
3405 Triana Boulevard
Huntsville, AL 35805-4695

Contents

List of Figures

List of Tables

1 Introduction

This report documents the simulation of the digital receiver (DR) of the ten-beam receiver subsystem (TBRS). This includes simulation of the TBRS intermediate frequency (IF) analog-to-digital converter (A/D) samples and simulatiuon of the processing of these samples.

Two separate software simulations were created. The first simulates the IF signal samples. The second simulates the digital down conversion (DDC) processing of the IF samples. Two simulations were generated since the simulated IF samples are generated with pseudo random noise (PRN) samples plus determistic signal samples. The simulated IF signal plus noise samples are output to data files that can be processed multiple times. This allows the same IF samples to be processed by both the processing simulation and the actual firmware in the TBRS. This also allows the same IF samples to be processed multiple times as the firmware and simulation software are developed and tested.

1.1 Document Overview

This document contains the following sections:

- This section provides an introduction.
- Section 2 on page 6 describes the simulation of the IF signal plus noise samples. This section also describes how to run the simulation and generate IF samples.
- Section 3 on page 17 describes the simulation of the TBRS DDC and how to run the simulation.

1.2 Abbreviations

The following abbreviations and terms are used in this document.

A/D Analog-to-digital converter.

BPF Bandpass filter.

CSPU Control and signal processor upgrade project.

DDC Digital down conversion.

DLPF Digital low pass filter.

DSPS Digital signal processing subsystem.

FIR Finite impulse response.

FFT Fast Fourier transform.

GCW Gated continuous wave.

HP and VP Horizontal and vertical polarization.

I and Q In-phase and quadrature-phase.

IF Intermediate frequency.

LFM Linear frequency modulation.

LPF Low pass filter.

LO Local oscillator.

MF Matched filter.

MP Multipulse.

PRN Pseudo-random number.

PSD Power spectral density.

QD Quadrature demodulator.

Radar The Eglin AN/FPS-85 phased array Radar.

SNR Signal-to-noise ratio.

SP Single pulse.

Specification DSPS specification.

TBRS Ten-beam receiver subsystem.

VM Vector modulator.

WSS Wide sense stationary.

2 Simulation of TBRS IF A/D Samples

This section describes the simulation of the TBRS IF A/D samples.

2.1 Simulation Overview

The analog inputs to the TBRS contain two narrowband (bandpass) signal plus noise components multiplexed in frequency. These are the horizontal polarization (HP) and vertical polarization (VP) returns from a radar target. Both components have a bandwidth of approximately 1 MHz. The HP component is centered at 20.5 MHz and extends from 20 to 21 MHz. The VP component is centered at 23.5 MHz and extends from 23 to 24 MHz. The overall HP/VP spectrum is centered at 22 MHz.

The TBRS samples each analog input with a 14-bit A/D at a 60 MHz sampling rate. Two software vector modulators (VM) are used to simulate the HP and VP bandpass signal plus noise components. A block diagram of the HP modulator is shown in Figure 1 on the following page. The VP modulator is identical except for the parameters.

A software Gaussian pseudo-random number (PRN) generator is used to generate the two independent, zero mean, white Gaussian noise vectors shown in the figure. The standard deviation σ of the noise samples in counts is calculated from the analog bandpass noise power in dBm input to the simulation.

The noise samples are added to baseband quadrature signal samples. These are the baseband in-phase and quadrature-phase (I and Q) components of the CSPU waveforms sampled at the TBRS A/D sampling rate $f_s = 60$ MHz. The sampling period T_s is the the inverse of the sampling frequency. This includes both linear frequency modulated (LFM) and gated continuous wave (GCW) signals. The LFM bandwidth is B_{lfm} and is zero for GCW waveforms. The pulse width is τ_p and the amplitude in counts is A_H for the HP modulator. The amplitude in counts is calculated from the signal power level in dBm input to the simulation.

The signal amplitude is scaled prior to the addition. This will be explained shortly. The combined signal plus noise samples are lowpass filtered with digital lowpass filters (DLPF) to set the noise bandwidth. This also limits the signal bandwidth. The DLPF has a noise gain of G_n and a signal gain of G_s. The output of the filter is scaled by $1/G_n$ so that the standard deviation of noise at the filter output is equal to the desired value σ. The signal gain through the filter is G_s, so the signal is pre-scaled by G_n/G_s so that the signal amplitude at the filter output is A_H.

The DLPF is a finite impulse response (FIR) filter with N impulse response samples w_i.

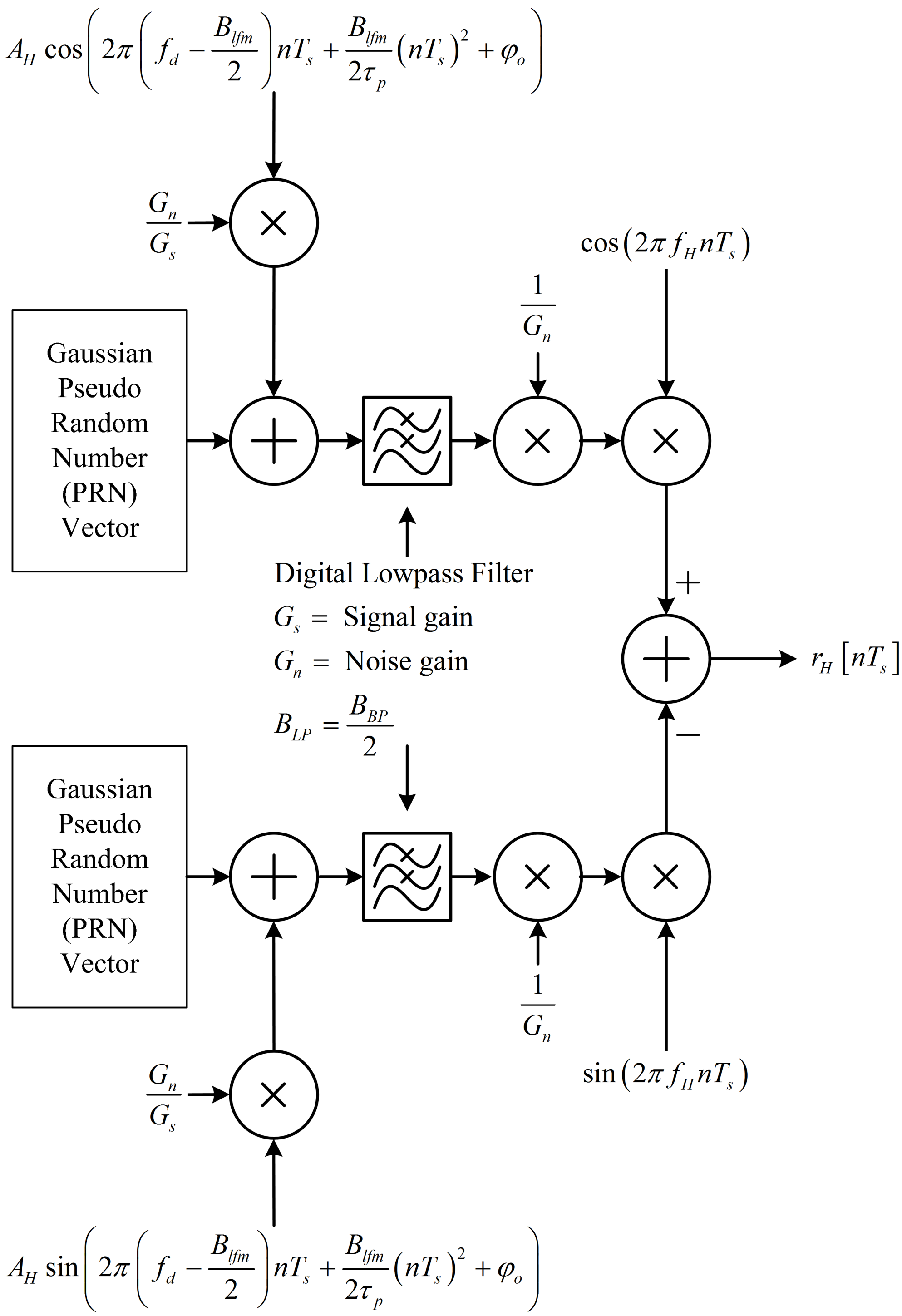

Figure 1: Digital Vector Modulator

The signal and noise gain through the filter are given as

$$
\begin{aligned}
G_s &= \sum_{i=0}^{N-1} w_i \\
G_n &= \sqrt{\sum_{i=0}^{N-1} w_i^2}
\end{aligned}
$$

This assumes that the input noise samples are independent and the noise spectrum is white.

The filter outputs are scaled and then modulated to the center frequency f_H. This converts the I and Q lowpass signal plus noise samples to bandpass signal plus noise processes at the desired center frequency and sampling rate. The resulting banpass samples are called $r_H[nT_s]$

The composite HP and VP bandpass signal plus noise samples are simulated with two software VM objects as shown in Figure 2 on the next page. The signal amplitudes A_H and A_V, and noise standard deviations σ_H and σ_V of each VM are set independently, though the noise standard deviations are typically set equal. The quadrature LO frequencies are f_H = 20.5 and f_V = 23.5 MHz. The pulse width and LFM bandwidth are set by the CSPU waveform being simulated and are common to both VMs. The 11 CSPU waveforms are listed in Table 1

Table 1: CSPU Waveforms

Pulse Type	τ_p (usec.)	B_{lfm} (MHz)
1	1	0
2	10	0
3	16	1
4	25	0
5	32	1
6	64	1
7	125	0
8	128	0.1
9	128	1
10	250	0.1
11	250	1

The outputs the vector modulators ($r_H[nT_s]$ and $r_V[nT_s]$) are summed and a constant μ is added to simulate the unwanted DC offset of the A/D. The processing up to this point is performed in floating point. The floating point samples are rounded to simulate the TBRS A/D quantization. The resulting samples are called $r[nT_s]$.

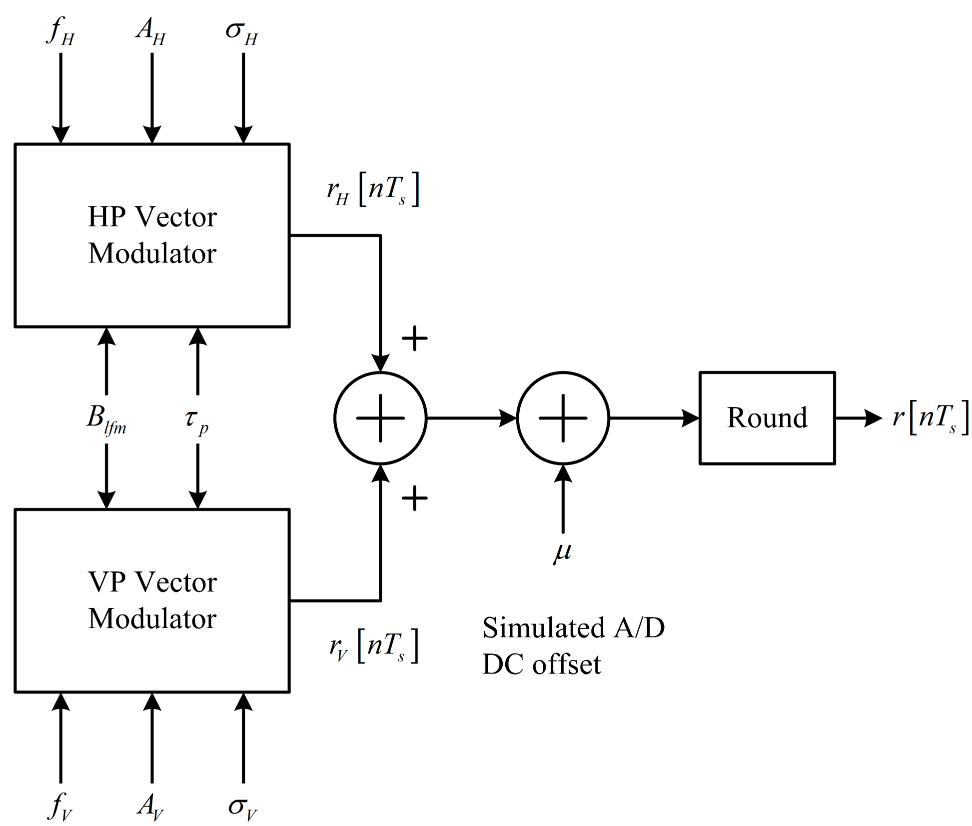

Figure 2: Dual Vector Modulators

2.2 Calculation of Signal and Noise Count Values

The TBRS A/D characteristics and the typical A/D input noise level must be considered to properly simulate the A/D output samples. The TBRS A/D is a 14-bit device. The output numeric format is two's complement, so the range is from -2^{13} to $+2^{13}-1$. The A/D reaches full scale output when the input is at +10 dBm into 50 ohms. Solving for the input full scale voltage gives

$$\begin{aligned} 10\log_{10}\left(\frac{A_{fs}^2}{2R}\right)+30 &= +10 \\ \log_{10}\left(\frac{A_{fs}^2}{100}\right) &= -2 \\ \frac{A_{fs}^2}{100} &= 10^{-2} \\ A_{fs}^2 &= 1 \\ A_{fs} &= 1 \text{ volt.} \end{aligned} \tag{1}$$

The shows that the full scale input is 1 volt. The voltage quantization is

$$v_q = \frac{1}{8192} = 122.07\times10^{-6} \text{ volts.} \tag{2}$$

The noise level into the A/D is nominally set so that the noise standard deviation in counts will be $\sigma = 3v_q$. This gives a noise power of

$$\begin{aligned} P_n &= 10\log_{10}\left(\frac{\sigma^2}{R}\right)+30 \\ &= 10\log_{10}\left(\frac{134.11\times10^{-9}}{50}\right)+30 \\ &= -85.72+30 \\ &= -55.72 \text{ dBm.} \end{aligned} \tag{3}$$

The HP and VP peak signal power and noise power in dBm are input to the simulation. These values must be converted to counts when generating the inputs to the dual VMs. The peak signal level in volts and counts is given as

$$\begin{aligned}
P_s &= 10\log_{10}\left(\frac{A_v^2}{2R}\right) + 30 \\
10^{\frac{P_s-30}{10}} &= \frac{A_v^2}{100} \\
A_v &= 100 \cdot 10^{\frac{P_s-30}{20}} \\
&= 10^{1+\frac{P_s-30}{20}} \\
A_c &= \frac{A_v}{v_q}. \qquad (4)
\end{aligned}$$

The nominal noise power is given by equation 3 on the previous page and the noise standard deviation in counts is 3 . If a lower or higher noise power is desired, the conversion from power in dBm to noise standard deviation in volts and counts is given as

$$\begin{aligned}
P_n &= 10\log_{10}\left(\frac{\sigma_v^2}{R}\right) + 30 \\
10^{\frac{P_n-30}{10}} &= \frac{\sigma_v^2}{50} \\
\sigma_v &= \sqrt{50} \cdot 10^{\frac{P_n-30}{20}} \\
\sigma_c &= \frac{\sigma_v}{v_q}. \qquad (5)
\end{aligned}$$

The amplitude and noise counts set in the equations above are for both the HP and VP narrowband processes. If the noise power is set to -55.72 dBm for both HP and VP, then each will have -55.72 dBm.

2.3 Executing the Simulation

The simulation was generated using the open source Python programming language with additional open source scientific and graphics libraries. This includes the following:

- Python version 2.5.4. This is the core Python language.
- NumPy multi-dimensional array library version 1.3.0.
- Matplotlib plotting library version 0.99.0.
- wxPython graphical user interface library version 2.8.10.1.
- SciPy advanced math and signal processing library version 0.7.1.

These items can all be obtained separately, but it is much easier to install Python and the required libraries using the integrated installer available at www.pythonxy.com. This site maintains the required packages and libraries and includes a "one-click" installer for the Windows platform.

The IF simulation main source file is named `TbrsIfSim.py`. Python is an interpreted language and no compilation to an executable file is required. `TbrsIfSim.py` requires the following additional files:

- `fir.py`. This file implements the FIR software objects for the DLPFs in the VMs.
- `util.py`. This file contains utility functions used by `TbrsIfSim.py`.
- `h767_Dec48_100.csv`. This "comma separated value" file contains the impulse response samples for the DLPFs in the VMs.
- Input parameter files for each pulse type to be simulated.

The input parameter files set the parameters used to simulate the IF samples. An example for pulse type 3 is shown below. The input file name is `pt03.inp`. The file name must begin with "pt" and have extension ".inp". The input file should include all 10 values as shown.

```
pulseType          = 3            # CSPU pulse type.
beamID             = 5            # NOVEM antenna beam number (1..10)
numBasebandSamples = 250          # Number of output baseband samples to generate.
Ps_H               = 0.0          # HP peak signal power (dBm).
Ps_V               = -10.0        # VP peak signal power (dBm).
Pn_H               = -55.72       # HP noise power (dBm).
Pn_V               = -55.72       # VP noise power (dBm).
tau_d              = 0.550205775  # Target delay (fraction of 60 MHz IF sample window).
f_d                = 0.0          # Target Doppler frequency offset (Hz).
muIF               = 0.0          # Simulated A/D dc offset (counts).
```

The simulation of the TBRS DDC processing is named `TbrsSimDdc.py` and is discussed in Section 3 on page 17. Both `TbrsIfSim.py` and `TbrsSimDdc.py` are designed to be executed from a common folder. The simulated IF samples generated by `TbrsIfSim.py` are stored in a sub-folder named `IFdata` and are read by `TbrsSimDdc.py`. The sub-folder is created by `TbrsIfSim.py` if it does not exist.

`TbrsIfSim.py` generates the IF data file name based on the current date and time, and the pulse type. An example file name is `IF_Y2009M10D13H13M31S33PT03.dat`. The components of this file name are:

- Year 2009.
- Month 10 (October).

- Day 13 (the thirteenth day of October).
- Hour 13 (1:00 PM).
- Minute 31.
- Second 33.
- Pulse type 3.
- File extension ".dat".

`TbrsIfSim.py` generates a second file with the same base name, but with file extension ".txt". This file repeats the ".inp" input parameter file contents used to generate the IF data samples. `TbrsSimDdc.py` reads the ".dat" and ".txt" files when processing the data.

`TbrsIfSim.py` can be executed by double-clicking its name/icon from Windows Explorer or by typing `TbrsIfSim.py <CR>` from a command prompt opened to the simulation folder. The command line method is preferred as status information is printed that will be "lost" if the the program is executed from its icon.

When `TbrsIfSim.py` executes it builds a list of available input file and displays a dialog box to allow the user to select the desired pulse type to simulate. This is shown in Figure 3. The files are named with 2-digit pulse type numbers (i.e. `pt03.inp` instead of `pt3.inp`) so that the file list is sorted in the correct order.

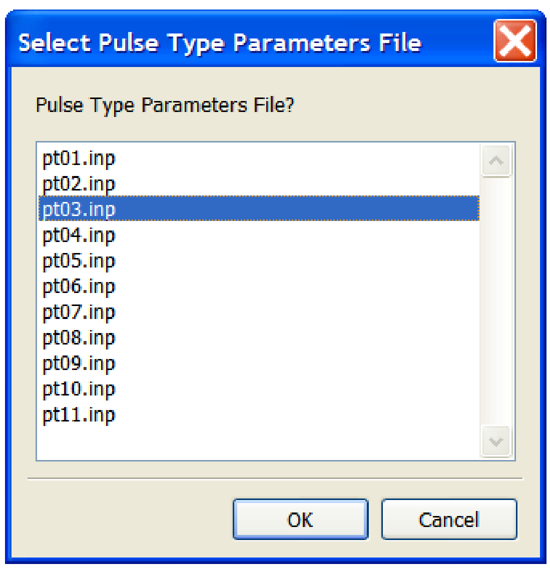

Figure 3: Input Parameter File Selection Dialog

Next `TbrsIfSim.py` prompts the user whether to save spectral plots as portable network graphics (.png) files as shown in in Figure 4 on the following page. If the user enters 1 as shown the plot files are saved in the `IFdata` folder along with the ".dat" and ".txt" files.

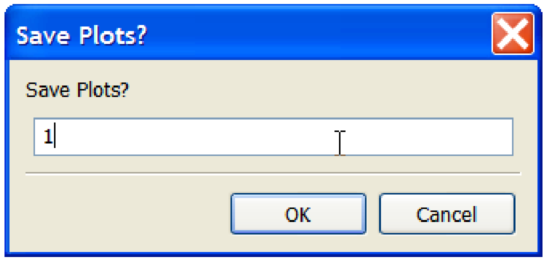

Figure 4: Save Plot Files Dialog

The data is then generated and two spectral plots are displayed on the screen (and optionally saved as ".png" files). The first plot shows the entire 60 MHz (±30 MHz) sampled spectrum. The second plot shows a zoom in view of the positive frequency portion of the spectrum. The output plots for the example pulse type 3 input file are shown in Figure 5 and Figure 6 on the next page.

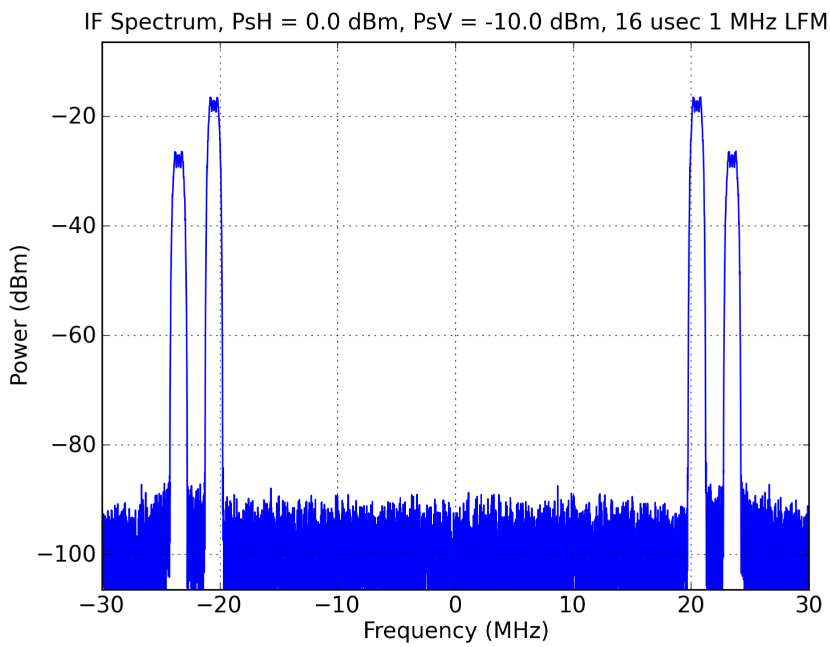

Figure 5: Full 60 MHz IF Spectrum

Figure 7 on page 16 shows the command line execution and the status information printed to the command prompt. This status information is "lost" if `TbrsIfSim.py` is executed by double-clicking its name/icon.

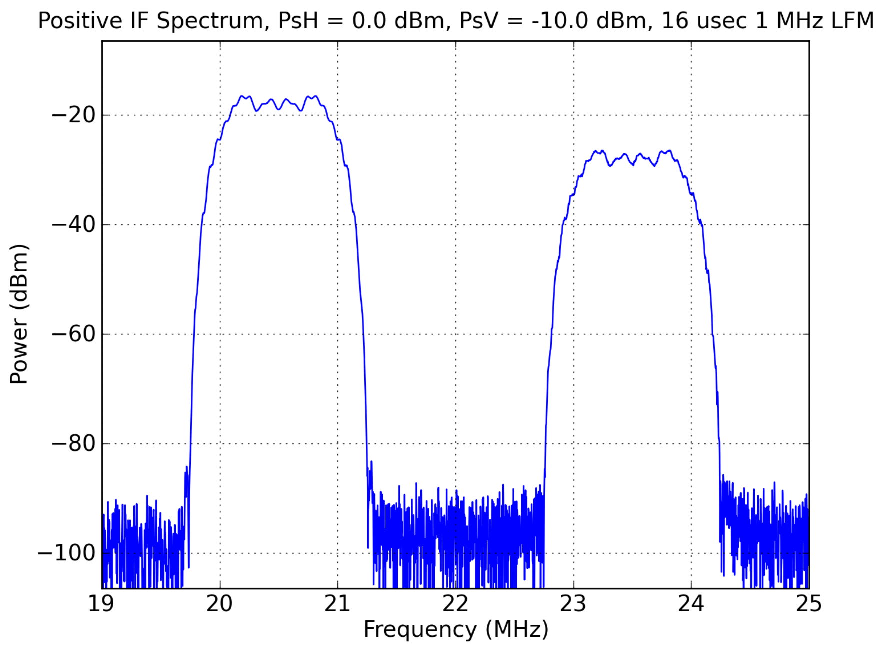

Figure 6: Zoom In View of Positive IF Spectrum

```
Command Window
C:\Users\Gary\SimTBRS>TbrsIfSim.py


Simulation input parameters
  Pulse type = 3
  Beam ID = 5
  Number of output baseband samples = 250
  HP peak signal power (dBm) = 0.000000
  VP peak signal power (dBm) = -10.000000
  HP noise power (dBm) = -55.720000
  VP noise power (dBm) = -55.720000
  Target delay = 0.550206
  Target Doppler frequency offset = 0.000000 Hz
  Simulated A/D dc offset = 0.000000 counts




Calculated from input parameters
  sigma_H = 2.998299 counts
  sigma_V = 2.998299 counts
  A_H     = 2590.537859 counts
  A_V     = 819.200000 counts

C:\Users\Gary\SimTBRS>
```

Figure 7: Command Line Execution and Status

3 Simulation of the TBRS Digital Down Conversion

This section describes the simulation of the TBRS DDC processing.

3.1 Simulation Overview

The TBRS digital down conversion (DDC) includes dual DDC circuits for the HP and VP signal components. A top-level block diagram of the TBRS DDC is shown in Figure 8 on the next page. The operation of the DDCs is identical and will be described for the HP DDC.

The simulated IF samples are read from the data file generated by `TbrsIfSim.py` and input to the quadrature demodulator and decimation by 12 block. A detailed diagram of this block is shown in Figure 9 on page 19. This is done for all 11 CSPU waveforms.

The input samples are split into two paths and multiplied by the cosine and sine local oscillators (LOs). These LOs "mix" the input sequence to baseband. The frequency of the LOs is equal to the HP IF frequency (20.5 MHz) plus the estimated target Doppler frequency. The TBRS generates these LOs using a numerically controlled oscillator (NCO) VHDL module.

The multiplier outputs are fed into DLPFs. These DLPFs are used to reduce the noise bandwidth by a factor of 12 so that the sampling rate can be reduced by a factor of 12 (decimated by 12). The simulation uses the 16-bit integer coefficients used by the TBRS VHDL code to implement this FIR. The number of coefficients for this filter is 83 and is called an 83-tap FIR.

The sampling rate is reduced from 60 MHz to 5 MHz by the decimation by 12. The processing that follows depends on the CSPU waveform being simulated. The multiple pulse (MP) waveforms (pulse types 4 and 7) are processed using the Matched Filter (MF) processing blocks. The single pulse (SP) waveforms are filtered and decimated by 4 to reduce the sampling rate from 5 to 1.25 MHz. The DLPF used to decimate by 4 uses the same 87-tap, 16-bit integer FIR coefficients used by the TBRS firmware for this filter.

This is the final DDC processing for pulse type 1 and the 1 MHz LFM pulse types (pulse types 3, 5, 6, 9, and 11). These pulse types all have a signal bandwidth of 1 MHz. Pulse type 1 is a GCW pulse type with the 1 MHz bandwidth set by the 1 microsecond pulse width.

The 100 kHz bandwidth pulse types (2, 8, and 10) are filtered again and their sampling rate is decimated by 10 to 125 kHz. A FIR with 167 taps is used for the filtering. This filter also has 16-bit integer coefficients.

The MP waveforms are processed in the MF processing blocks. The MF is a FIR filter with unity coefficients. The number of taps is set equal to the number of pulse samples N_p across the pulse at the 5 MHz input sampling rate. This is 125 taps for waveform 4 (25

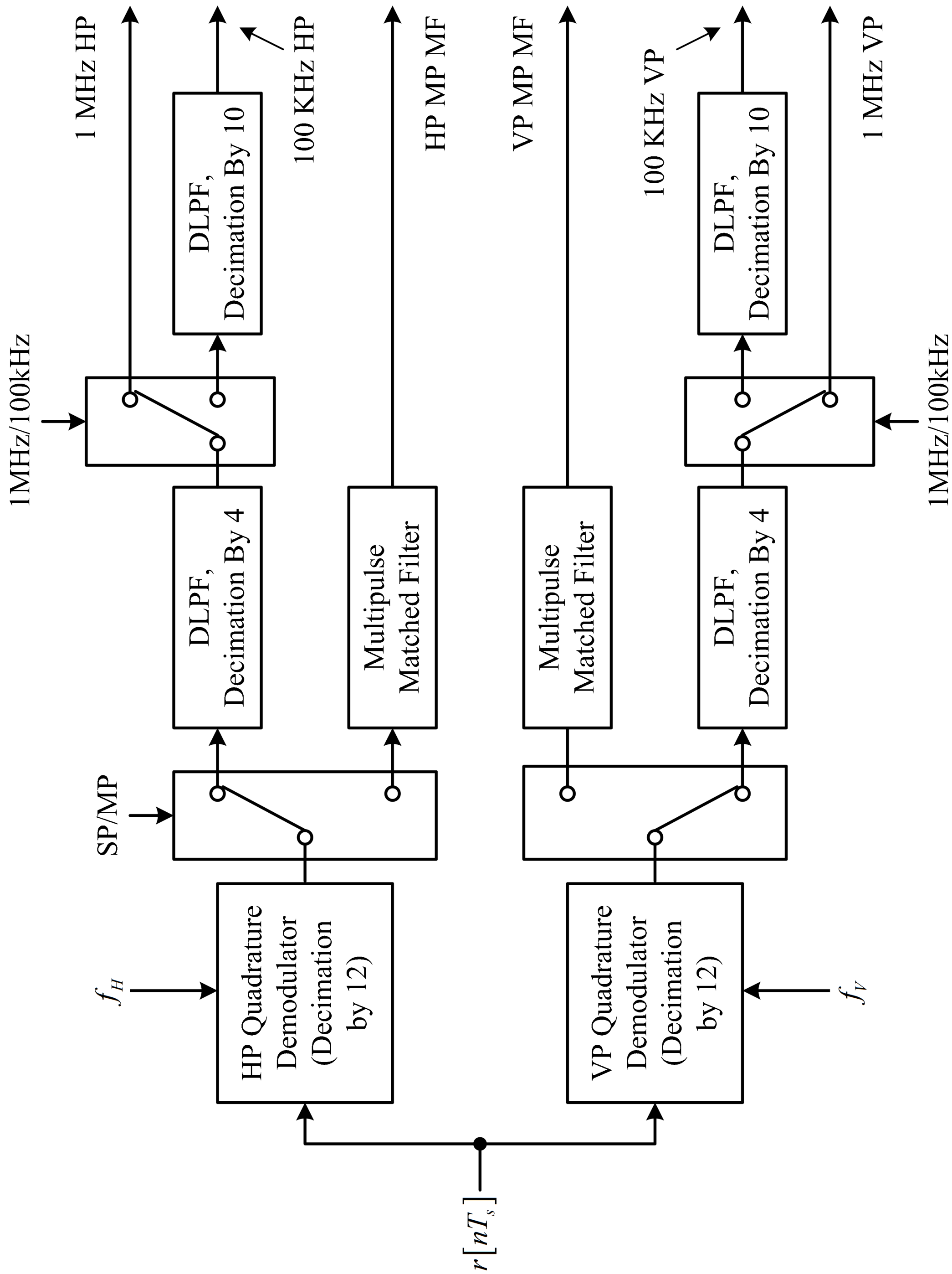

Figure 8: TBRS Digital Down Conversion

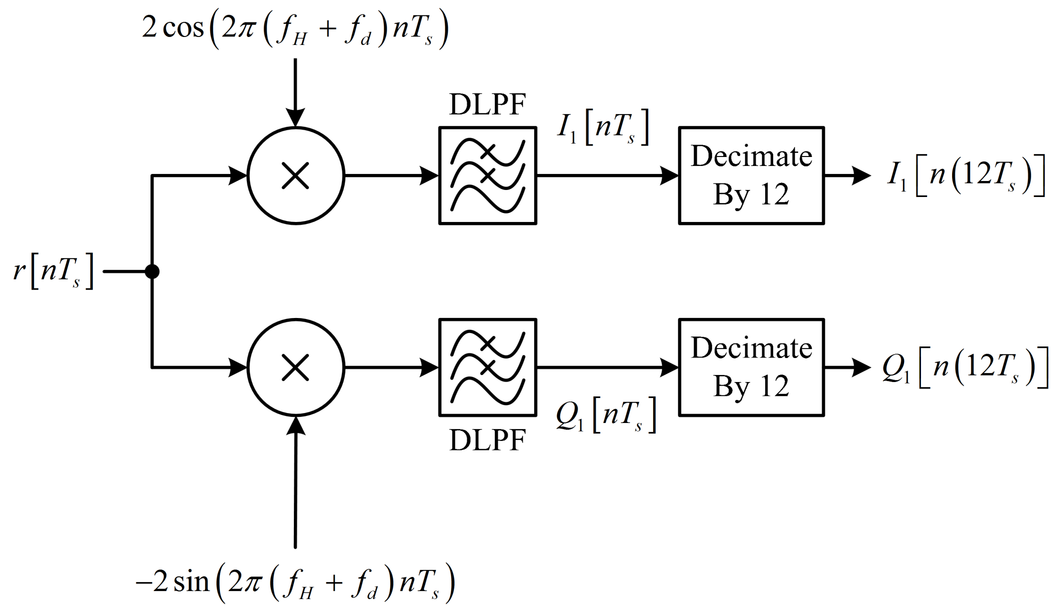

Figure 9: Quadrature Demodulator

microsecond pulse width) and 625 for waveform 7 (125 microsecnd pulse width). The MF FIR is followed by dual decimators. A block diagram of the waveform 4 MF is shown in Figure 10 on the next page.

The dual decimators each decimate by a factor of 250 from 5 MHz to 20 kHz for waveform 4. The "early gate" decimator outputs precede the "normal gate" outputs by 62 5 MHz samples. For instance, if the a normal gate decimator output is formed by summing input 5 MHz samples 250 to 499, the corresponding early gate output sample would be formed by summing input samples 188 to 437.

The dual decimators for waveform 7 are identical except the decimation factor is 750 instead of 250, and the early gate samples precede the normal gate samples by 312 5 MHz samples instead of 62.

`TbrsDdcSim.py` also computes the MF convolution for LFM pulse types. This is performed in the CSPU system by the digital signal processing subsystem (DSPS), but is included in `TbrsDdcSim.py` for completeness.

A block diagram of the processing is shown in Figure 11 on page 21 for the 1 MHz LFM waveforms. The input sampling rate is 1.25 MHz. The impulse response samples h_i and h_q are just the complex conjugate of the ideal LFM pulse samples. The impulse samples are multiplied by Taylor range sidelobe reduction weights w_{sl} and zero-padded to the power of 2 fast Fourier Transform (FFT) size that will be used to perform the convolution in the frequency domain.

The FFT size N_{fft} is determined by the number of baseband "range window" samples at

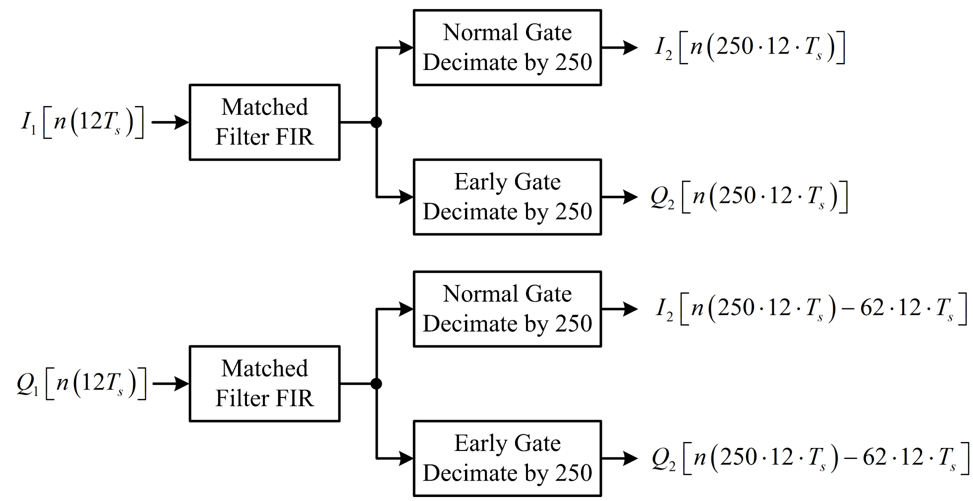

Figure 10: Multiple Pulse Matched Filter

the output of the TBRS DDC decimation by 4 (N_{rw}) and the number of pulse samples N_p. The FFT size is given as the smallest power of 2 $\left(2^k\right)$ such that $2^k \geq \left(N_p + N_{rw}\right)$.

The TBRS samples and the weighted impulse response sampes are zero-padded to N_{fft} and FFT'd. The FFT outputs are point-by-point vector multiplied and the product is inverse FFT'd to generate the time domain MF output.

3.2 Executing the Simulation

The simulation main source file is named `TbrsDdcSim.py` as mentioned above and requires the following additional files:

- `fir.py`. This file implements the FIR software objects for the DLPFs in the DDC.
- `util.py`. This file contains utility functions used by `TbrsDdcSim.py`.
- `lpf1.csv`. This file contains the 83-tap decimation by 12 integer coefficients.
- `lpf2.csv`. This file contains the 87-tap decimation by 4 integer coefficients.
- `lpf3.csv`. This file contains the 167-tap decimation by 10 integer coefficients.
- `IFdata`. This is the folder containing the simulated ".dat" and ".txt" files generated by `TbrsIfSim.py`.

`TbrsDdcSim.py` should be located in the same folder as `TbrsIfSim.py` and is executed the same way. Again, it is recommended to run the program from a command prompt.

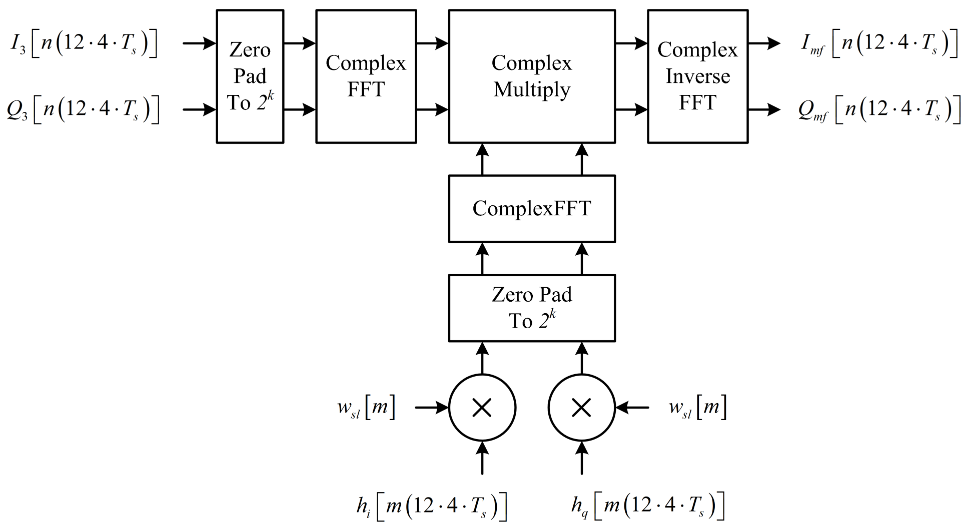

Figure 11: LFM Matched Filter

When `TbrsDdcSim.py` executes it displays a dialog box that prompts the user to select the pulse type to process (see Figure 12 on the next page).

The `IFdata` folder is then searched for ".dat" files of the selected pulse type. If pulse type 3 is selected as shown, files ending in the pattern "PT03.dat" are selected and presented in a dialog box so the user can select the specific file to process. This is shown in Figure 13 on the following page

If no files for the selected pulse type are located the program exits and prints an error message to the command line.

If an appropriate file is selected the program then prompts the user whether to save plot files (see Figure 14 on page 23). Simulation plots are generated and displayed to the screen regardless of the selection. If the user slects to save plot files, they are saved in the `plots` sub-folder.

Figure 15 on page 23 shows the status information printed to the console window after `TbrsDdcSim.py` completes execution.

`TbrsDdcSim.py` writes output data files to the following sub-folders:

- `5Mdata`. The 5 MHz samples at the output of the quadrature demodulator and decimation by 12 processing are written to this folder.
- `1_25Mdata`. The 1.25 MHz samples after decimation by 4 are written to this folder.
- `125kdata`. The 125 kHz samples after decimation by 10 are written to this folder.
- `plots`. Time domain and spectral plots are written to this folder.

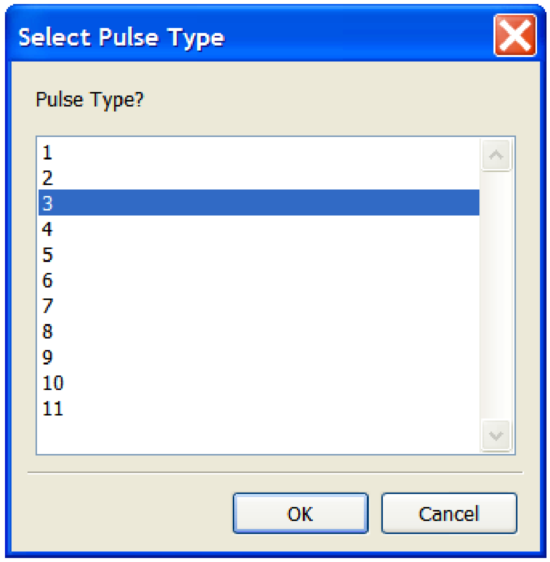

Figure 12: Input Pulse Type Selection Dialog

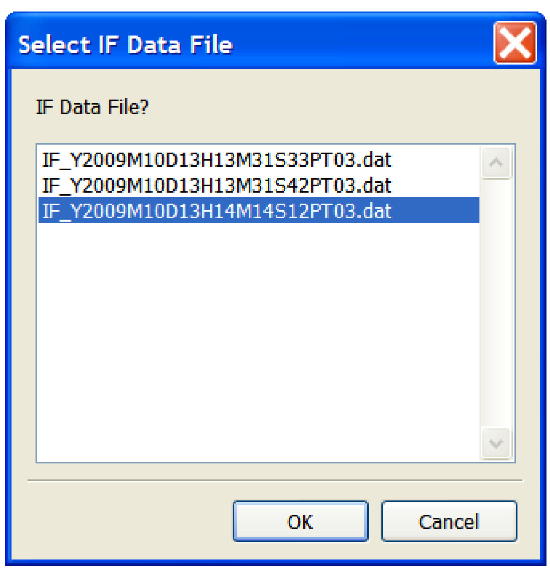

Figure 13: IF Data File Selection Dialog

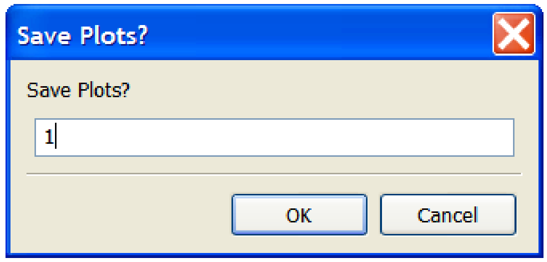

Figure 14: Save Plot Files Selection Dialog

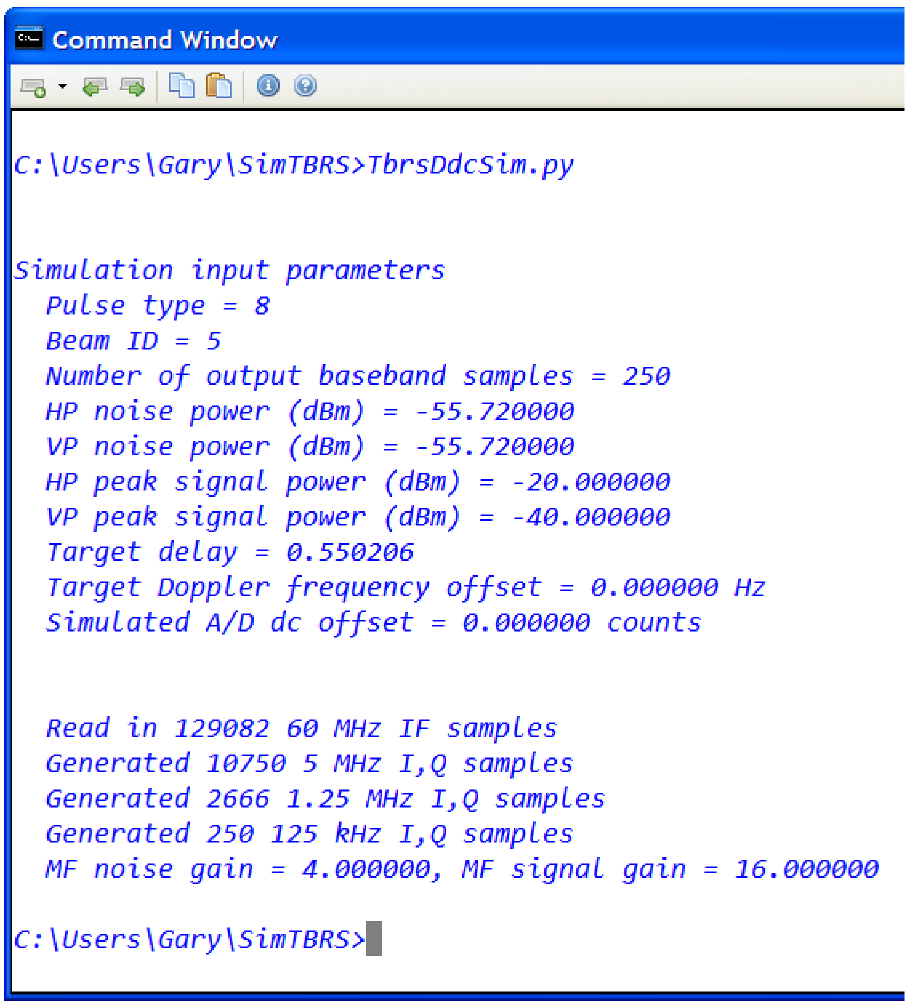

Figure 15: Console Window Status Information

`TbrsDdcSim.py` generates time domain and frequency domain spectral plots at the output of each stage of processing. Example plots are included here for pulse type 8 (128 microsecond pulse width, 100 kHz LFM bandwidth).

The input parameters used to generate the IF data are:

```
pulseType          =            8  # CSPU pulse type.
beamID             =            5  # NOVEM antenna beam number (1..10)
numBasebandSamples =          250  # Number of output baseband samples to generate.
Pn_H               =   -55.720000  # HP noise power (dBm).
Pn_V               =   -55.720000  # VP noise power (dBm).
Ps_H               =   -20.000000  # HP peak signal power (dBm).
Ps_V               =   -40.000000  # VP peak signal power (dBm).
tau_d              =     0.550206  # Target delay (fraction of 60 MHz IF sample window).
f_d                =     0.000000  # Target Doppler frequency offset (Hz).
muIF               =     0.000000  # Simulated A/D dc offset (counts).
```

The calculated signal and noise counts are:

```
A_H                =   259.053786  # HP Amplitude (counts).
A_V                =    25.905379  # VP Amplitude (counts).
sigma_H            =     2.998299  # HP Sigma (counts).
sigma_V            =     2.998299  # VP Sigma (counts).
```

The following plots were generated when processing the IF data file:

- Figure 16 on the next page. These are the I and Q time domain 5 MHz sample plots for both HP and VP.
- Figure 17 on the following page. These are HP and VP spectral plots of the 5 MHz samples.
- Figure 18 on page 26. These are the I and Q time domain 1.25 MHz sample plots after decimation by 4.
- Figure 19 on page 26. These are HP and VP spectral plots of the 1.25 MHz samples.
- Figure 20 on page 27. These are the I and Q time domain 125 kHz sample plots after decimation by 10.
- Figure 21 on page 27. These are HP and VP spectral plots of the 125 kHz samples.
- Figure 22 on page 28. These are the I and Q time domain MF sample plots.
- Figure 23 on page 28. These are the log magnitude time domain MF sample plots.

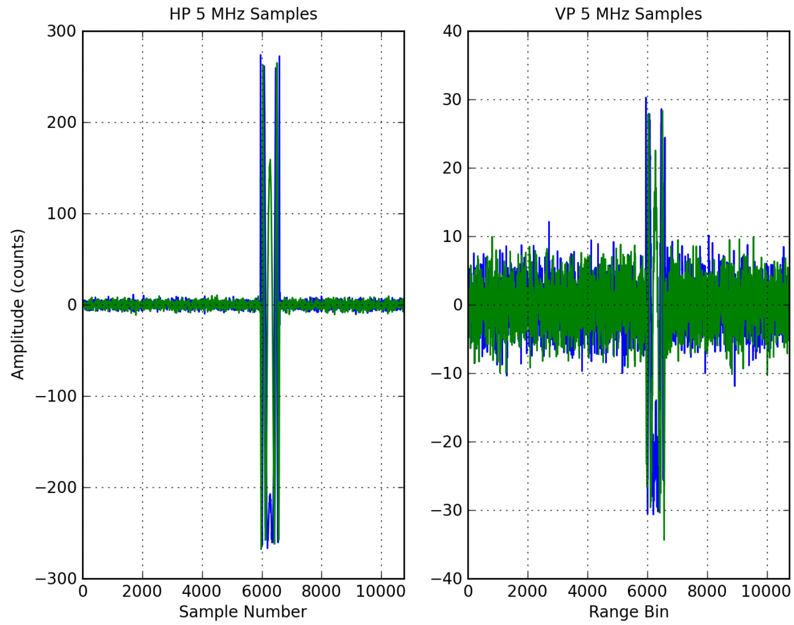

Figure 16: 5 MHz I and Q Time Domain Samples

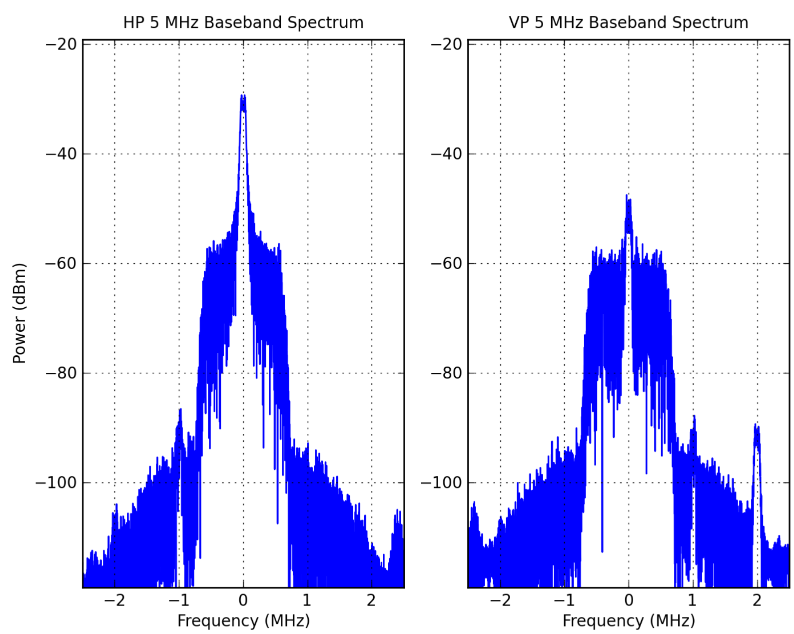

Figure 17: 5 MHz Spectral Plots

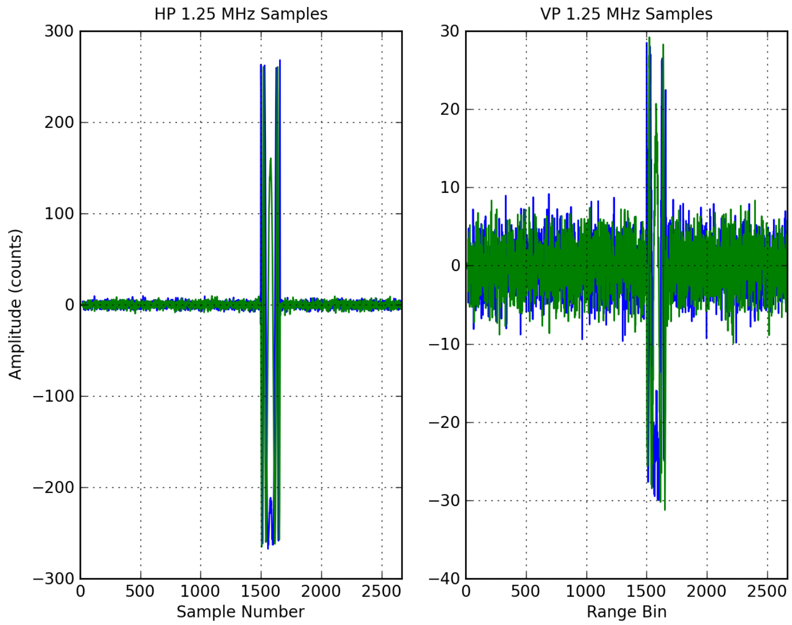

Figure 18: 1.25 MHz I and Q Time Domain Samples

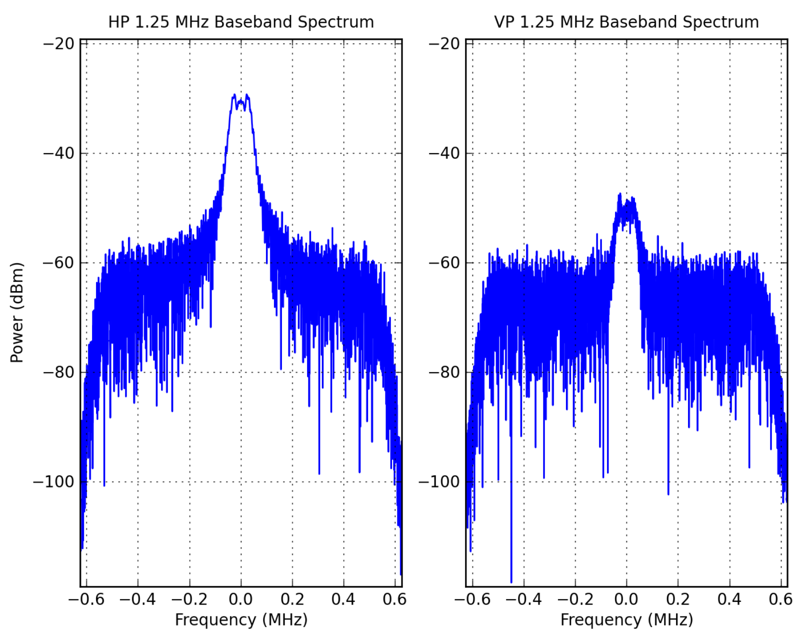

Figure 19: 1.25 MHz Spectral Plots

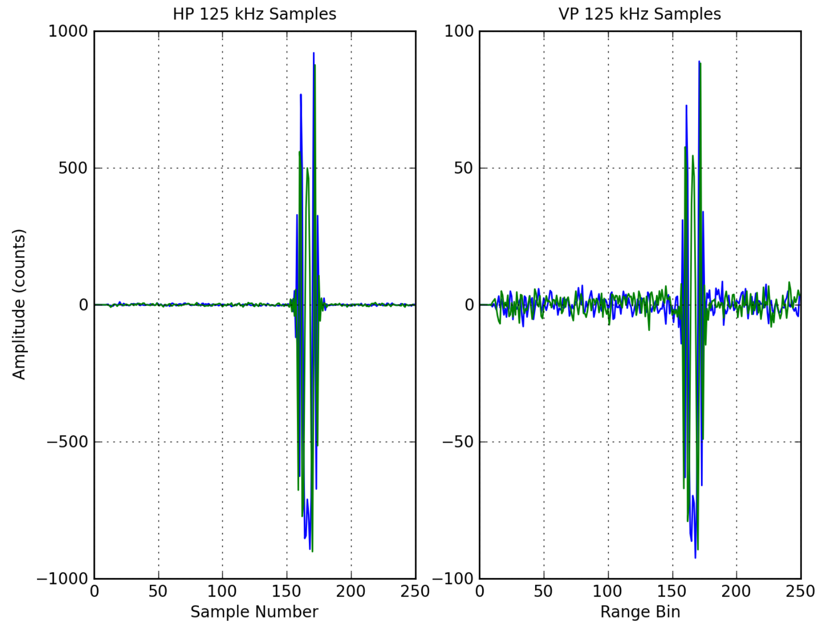

Figure 20: 125 kHz I and Q Time Domain Samples

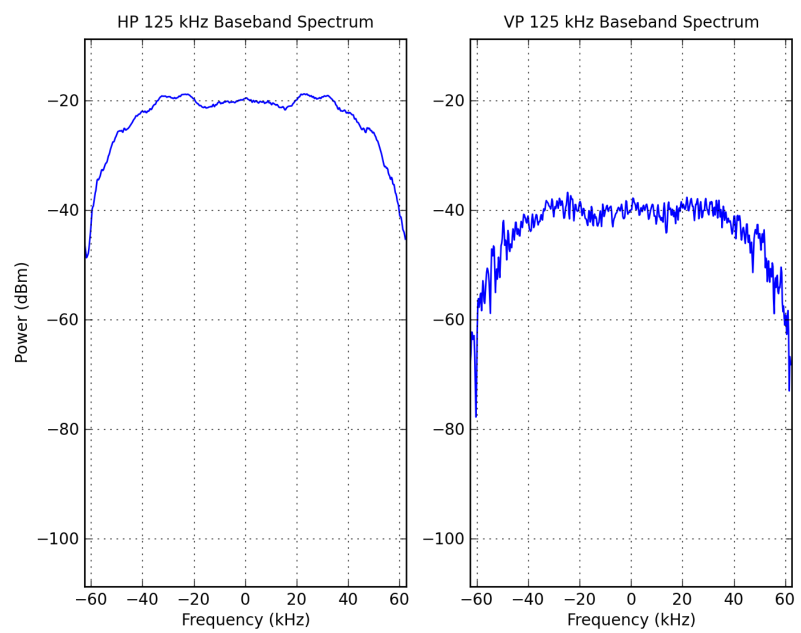

Figure 21: 125 kHz Spectral Plots

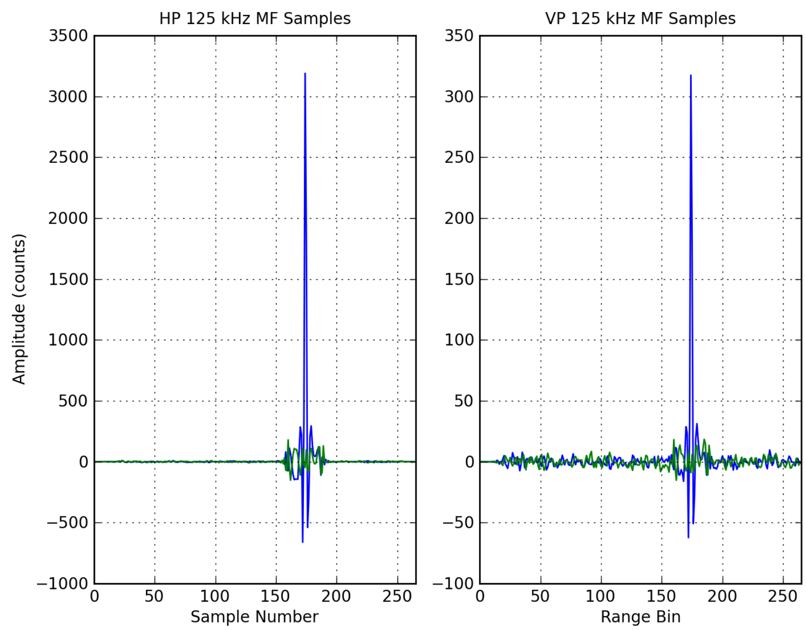

Figure 22: Matched Filter I and Q Time Domain Samples

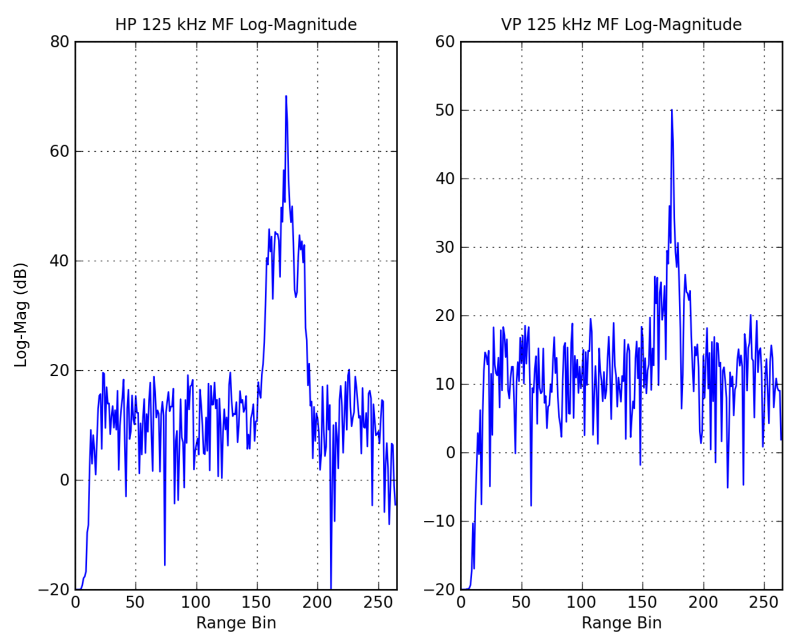

Figure 23: Matched Filter Log Magnitude Time Domain Samples