Indian Institute of Technology Palakkad
भारतीय प्रौद्योगिकी संस्थान पालक्काड
Under Ministry of Human Resource Development, Govt. of India
मानव संसाधन विकास मंत्रालय के अधीन, भारत सरकार

Final Year Project Phase 2, Mid-Term Report

Electrical Engineering

IIT Palakkad

March 11, 2022

Developing a Radar Signal Processor for the Detection of Targets

Mentors: Dr. Swaroop Sahoo & Dr. Subrahmanyam Mula

Name	: Aswin Raj K
Roll Number	: 121801008
Department	: Electrical Engineering

Developing a Radar Signal Processor for the Detection of Target

Abstract

In this report, the work done as a part of developing the radar signal processor for the detection of drones is discussed. The signal processing part is designed to be implemented on an FPGA (ZC702) board. It is designed to produce pulses with monotonous frequency as well as chirp [1, p. 420]. The required RF pulse is generated at baseband that will be upconverted to X-band on the SDR and transmitted out. The FPGA is programmed to store the sampled data coming from the SDR in the form of a data matrix. The data stored in the data matrix is used to extract the doppler [2, p. 92] and range information. Before its implementation onto an actual FPGA board the radar is simulated using MATLAB to find the best parameters required for the radar to operate.

1 Introduction

FPGA is extensively used for high computing applications. It provides us with the flexibility of massive parallel data processing, better performance, efficiency, and faster prototyping. It can be configured to run several digital signal processing algorithms within in short period. FPGA, when compared to equivalent discrete circuits, occupies less space. All these capabilities of FPGA make it the perfect platform for the implementation of a radar signal processor. In traditional radar signal processors, slow processing and low resolution have limited it's working. The implementation of a radar signal processor on an FPGA has increased the computation speed together with the improvement in the accuracy and precision of the results.

In this report, the implementation of a radar signal processor based on FPGA is discussed. The implementation of Hamming window [4], FFT algorithm, Double delay-line canceler [1, p. 107], and Matched filtering [3, p. 161] is thoroughly explained in this report. The FPGA simulation result is compared with the reference done in MATLAB for verification.

2 Project Overview

Project Objectives

1. **Simulating the radar using Matlab.**
   - Before implementing the radar onto an actual FPGA, we need to find the right parameters to ensure it is working as expected.
   - A matlab simulation of the radar model is done to finalize the required parameters which meets our need.
2. **Implementing the transmitter on the FPGA.**
   - Chirp signal generator is to be implemented on the FPGA in digital domain.
   - This generated digital signal needs to be sent to the SDR for transmitting it through the antenna.
3. **Ensuring that the chirp pulses transmitted from the FPGA is as expected.**
   - This can be done using another receiver and observing the transmitted signal transmitted from the FPGA.
4. **Implementing the receiver on the FPGA.**
   - Storing the echo data received by the SDR onto the memory block of FPGA.
   - Processing the digitized data on the FPGA itself.
   - Blocks for Hamming window, 3 pulse canceler, Matched filtering, Threshold detection, and FFT is to be designed for processing the data.

⑤ **Exporting the processed data for visualization.**

- Export the final processed data from the FPGA to MATLAB for easy visualization.
- Plotting the result in MATLAB to ensure that the radar is working as expected.

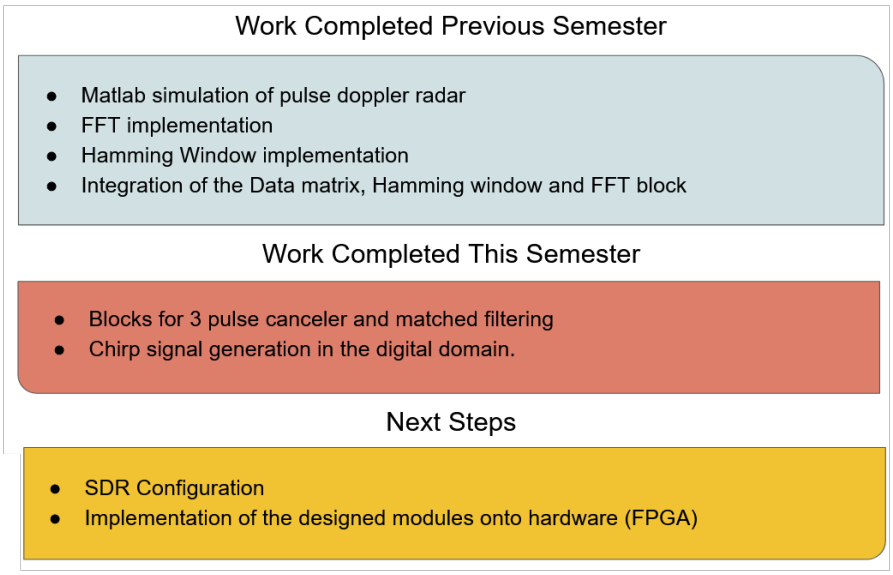


Figure 1: Completed and pending works

3 Hardware Implementation

The entire signal processor is implemented onto an FPGA board with an SDR attached to it. The entire process is summarized in the figure 2. An SDR is attached to the FPGA board. The SDR is configured to the required needs. The sampled data coming from the SDR is stored in block RAM for further processing. In figure 1, the doppler [2, p. 92] and range information at each stage of processing is plotted. The raw data coming from SDR contains noise as well as clutter [1, p. 470], which is removed in later steps. It is understood from figure 1 that after clutter cancellation the clutter is removed, adding up the high frequency noise. Matched filtering [2, p. 161] is then employed to reduce the effect of noise. The final processed data only contains the information of the targets. Chirp Signal Generator, Matched filter, 3 Pulse Clutter Canceler [1, p. 107], 32-point FFT and Hamming Window [8] is designed using Vivado and to be implemented onto an actual hardware.

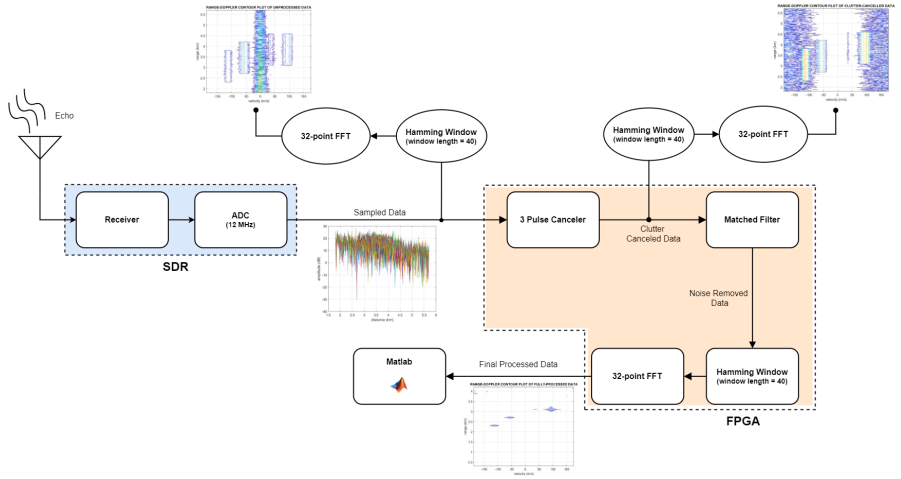


Figure 2: Block Diagram

The required pulse parameters are found out via simulation in MATLAB. The parameters used for the radar is as given in table 1. The FPGA is configured with the SDR such that it sent 40 pulses of width 10µs at intervals of 20µs, during which it listens for any incoming echoes.

Parameter	**Value**
Pulse width (T)	10μs
Pulse repetition frequency	25 kHz
Pulse repetition interval	40μs
R_{min}	1.8 km
R_{max}	5.7 km
Unambiguous range	3.9 km
Unambiguous velocity	375 m/s

Table 1: Radar parameters used

3.1 Pulse Generation

A typical radar pulse is as shown in figure 3. Each pulse contains only a single frequency component. Such a pulse won't give us the required performance. Thus, instead of sending normal pulses at regular intervals, chirp signal [1, p. 420] is used.

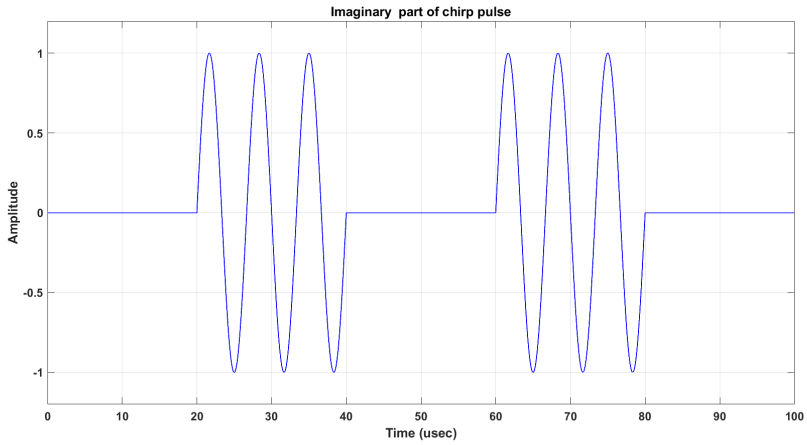
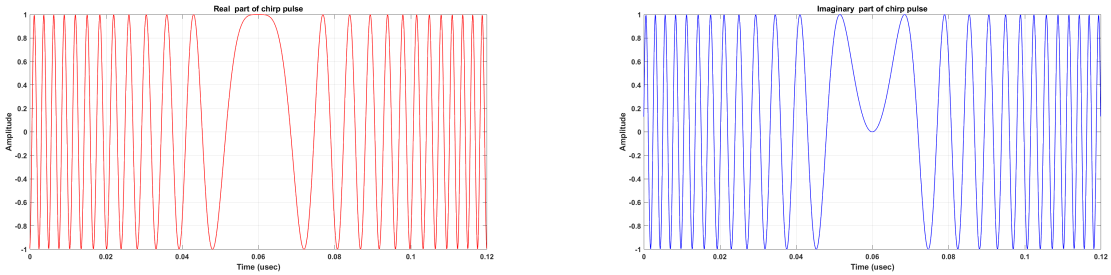


Figure 3: Normal pulse (pulse width = 20µs, PRF = 25kHz)

Chirp is a signal in which the frequency is varied. It is a pulse compression [1, p. 420] technique which allows a radar to radiate a large amount of energy but can simultaneously obtain the range resolution [2, p. 21] of a small pulse. Long pulse gives more range whereas the chirp signal within the pulse allows achieving range resolution of a small pulse. Equation 1 gives the value of the chirp signal for the nth sample.

$$S(n) = e^{\frac{j\pi\omega}{T}\left(nT_s \frac{(N-1)f_s}{2}\right)^2} \tag{1}$$

The real and imaginary part of signal is as shown in figure 4. The signal is sampled at f_s =12MHz and the data is stored inside block memory. The digitized data is to be sent to the SDR attached to the FPGA. The SDR converts the digital baseband signal to passband signal and is radiated at regular intervals.



(a) Real part of chirp pulse

(b) Imaginary part of chirp pulse

Figure 4: Three simple graphs

3.2 3 Pulse Canceler (Double delay-line canceler)

The delay-line canceler [1, p. 107] acts as a filter which rejects the d-c component of clutter [1, p. 470]. Because of its periodic nature it rejects energy in the vicinity of the pulse repetition frequency and its harmonics. The frequency response of a single-delay-line canceler does not always have a broad clutter rejection property. The clutter rejection notch can be widened by passing the output of the single delay-line canceler through one more delay-line canceler. This configuration is called double-delay line canceler.

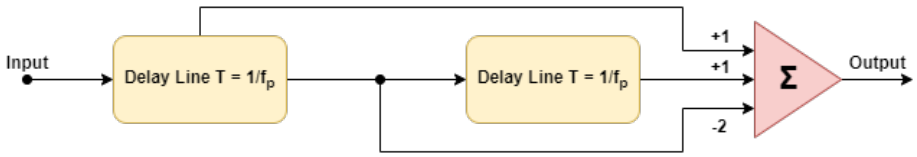


Figure 5: Double delay-line canceler block diagram

The data matrix is multiplied with the coefficient matrix to get the output. The dimension of the data matrix is 313 x 40 and that of the coefficient matrix is 40 x 39. Therefore, the output matrix would be 313 x 39. The matrix multiplication algorithm is designed using Xilinx Vitis and then exported as an IP to Xilinx Vivado design suite for simulation. The block diagram is as shown in figure 6. The 'datasrc' IP block supplies the data to the 'PulseCanceler' IP block. The IP block stores the data in memory and after all the data is transferred to the 'PulseCanceler', it then performs the necessary multiplication and outputs through the 'data_OUT' port. The 'dataWrite' IP block displays the output coming from the 'data_OUT' port in the console window of Vivado which is then plotted using MATLAB for verification.

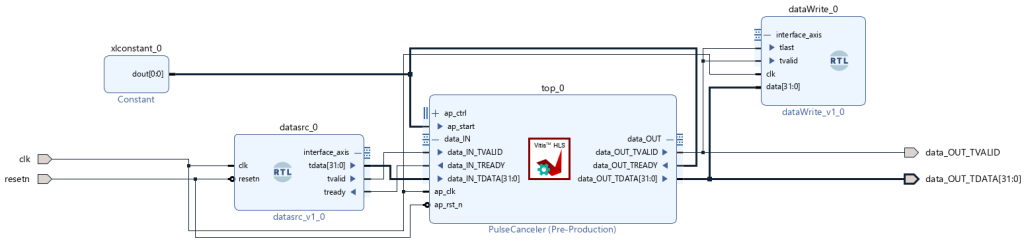


Figure 6: Double delay-line canceler IP block diagram

3.3 Matched Filtering

After clutter cancellation, the SNR ratio is improved using matched filtering [3, p. 161]. Matched filtering is performed on the data matrix along the fast time samples. For performing convolution, the FIR compiler [8] provided by the Xilinx is utilized. The filter is time delayed function of the chirp signal. The chirp signal is delayed by N/2 samples. The filter coefficient is generated using MATLAB to be used in the FIR compiler. Since the FIR compiler can have only real coefficients, the entire convolution

process is splitted into 4 convolution operations. If $s(t) = s_I(t) + js_Q(t)$ is the complex signal and $h(t) = h_I(t) + jh_Q(t)$ is the filter's complex impulse response then,

$$s(t) \circledast h(t) = s_I(t) \circledast h_I(t) - s_Q(t) \circledast h_Q(t) + j(s_I(t) \circledast h_Q(t) + s_Q(t) \circledast h_I(t)) \tag{2}$$

Where all the filter involved are real. For each convolution operation we use FIR compilers [8]. The IP block diagram is as shown in figure 7.

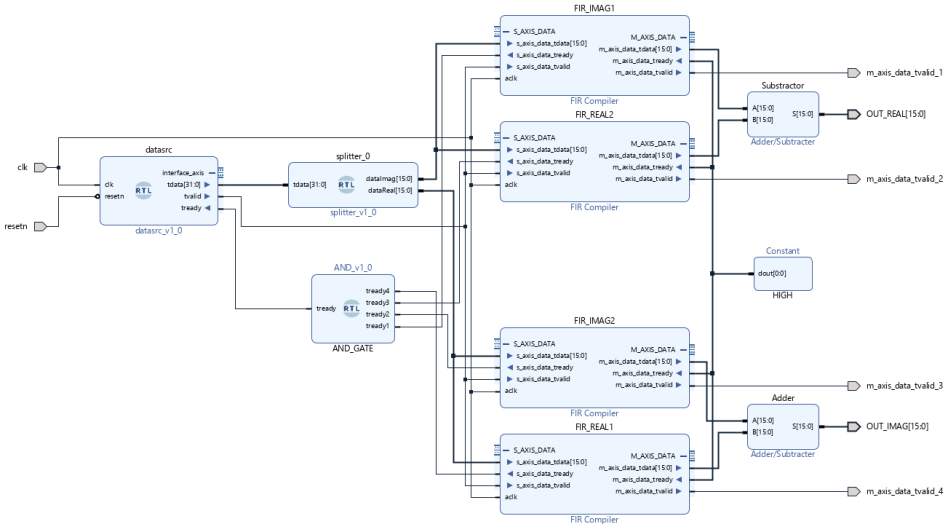


Figure 7: Matched filter IP block diagram

The 'datasrc' IP block supplies the data. 'splitter' splits the 32-bit data coming from 'datasrc' into its imaginary and real parts and is then supplied to each of the FIR compiler for performing convolution. Subtractor and an Adder block is used to perform complex subtraction and addition shown in equation 2, to get the final result. The final output is available the 'OUT_IMAG' and 'OUT_REAL' port of the subtractor and adder IP block respectively.

4 Results

The output of the 3-pulse canceler [1, p. 107] designed using Vitis is compared with that of the output from MATLAB.The output of the designed module is plotted along side MATLAB output. It is clear from figure 8 that the output of the designed 3-pulse canceler is same as that of the MATLAB output.

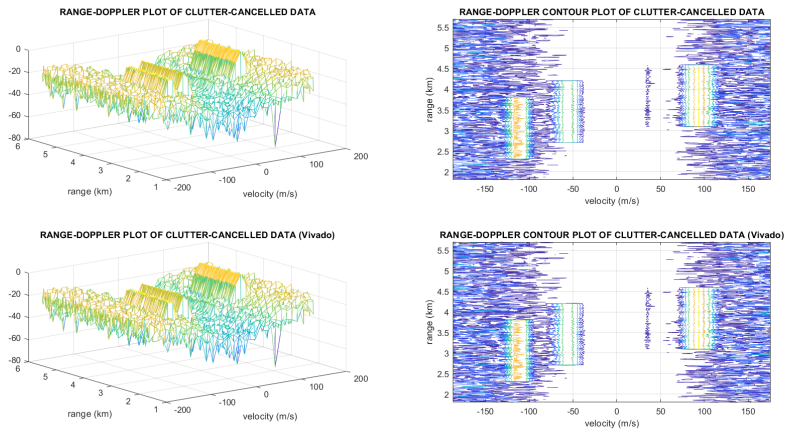


Figure 8: 3 pulse canceler output using MATLAB and the designed module in Vivado

5 Conclusion and Future Work

Matlab simulation of the radar model is successfully completed. The target parameters were successfully extracted and it matches with the actual target parameters. Modules for performing Hamming window, FFT, 3 pulse canceler, Matched filtering and base band signal generation is completed. The working of each of these designed modules is verified by comparing with the reference done in MATLAB. Some more testing needs to be done on Matched Filtering. Now all these designed modules needs to be implemented onto an actual FPGA.

References

[1] Merrill I. Skolnik, "Introduction To Radar Systems (2nd ed.)", Tata McGraw-Hill Edition, 1980.

[2] Mark A Richards, "Fundamentals of Radar Signal Processing (1st ed.)", Tata McGraw-Hill Edition, 2005.

[3] Robert Fisher; Simon Perkins; Ashley Walker; Erik Wolfart. "Image Synthesis — Noise Generation". Retrieved 11 October 2013.

[4] Smith, Julius O., "Pectral Audio Signal Processing", W3K Publishing, December 2011.

[5] Xilinx User Guide for ZC702 Evaluation Board for the Zynq-7000 XC7Z020 SoC.

[6] Block Memory Generator v8.3 LogiCORE IP Product Guide.

[7] Fast Fourier Transform v9.1 LogiCORE IP Product Guide

[8] FIR Compiler v7.2 LogiCORE IP Product Guide

Note: Link to the github repo is attached here.