Abstract

A Finite Impulse Response (FIR) filter is a type of digital filter used in signal processing. Unlike Infinite Impulse Response (IIR) filters, FIR filters have a finite impulse response, meaning their output is based only on a finite number of previous input samples. FIR filters are characterized by their impulse response, which is the filter's output when the input is an impulse signal (a single sample followed by zeros).

This impulse response is determined by the filter's coefficients, which are the values applied to each input sample to produce the output. FIR filters are typically implemented using convolution. The output of the filter at any given time is calculated by convolving the input signal with the filter's impulse response

FIR filters have several advantages, including stability, linear phase response, and ease of design. They are commonly used in applications such as audio and image processing, communications, and control systems.

The Bartlett window, also known as the triangular window, is a type of window function used in signal processing and spectral analysis. Window functions are used to taper the data before applying Fourier transform or other spectral analysis techniques to minimize spectral leakage and artifacts caused by abrupt truncation of the data.

Table of Contents

Problem Statement

EE521: Digital Signal Processors Lab

Assignment Question: Set No. - 5

---

1. Design a high-pass digital FIR filter with a sampling frequency of 800 Hz and a cutoff frequency of 550 Hz, employing a Bartlett window of length N = 5. Generate a random composite signal of 450 Hz and 650 Hz, whose data sampling frequency is 800 Hz. Use this designed filter to process the generated signal.

Instructions:

- Design the filter manually and calculate all the filter coefficients.
- Using **Python**, plot the magnitude and phase responses of the designed filter. Do not use any in-built functions/libraries (Write all the codes from scratch).
- Pass the generated signal to the designed filter to get the outputs.
- Plot the time and frequency domain representations for both the input and the filtered output signals.
- Also, compare the results with the in-built **Python** functions/libraries.
- Submit the **Python** code in a single iPython notebook (.ipynb) file.
- Then, implement the filter using **Verilog**.
- Use the previously calculated filter coefficients in the testbench to verify the designed model.
- Submit the **Verilog** design source code along with the testbench code and the screenshot of the simulated results in a single .doc file or pdf file.
- Prepare a detailed assignment report explaining all the questions asked, different steps, derivations and calculations, algorithms and the generated outputs and plots in a single **zip file**.

Best Wishes

Main Content

1 Introduction

1.1 Finite Impulse Response Filter

In signal processing, a finite impulse response (FIR) filter is a filter whose impulse response (or response to any finite length input) is of finite duration, because it settles to zero in finite time. This contrasts with infinite impulse response (IIR) filters, which may have internal feedback and may continue to respond indefinitely (usually decaying).

The impulse response (that is, the output in response to a Kronecker delta input) of an Nth-order discrete-time FIR filter lasts exactly N+1 samples (from first nonzero element through last nonzero element) before it then settles to zero.

1.2 Bartlett Window

The Bartlett window, also known as the triangular window, is a type of window function used in signal processing and spectral analysis. It is named after Maurice Stevenson Bartlett, who introduced it in 1950.

A window function, in this context, is a mathematical function applied to a signal to modify it before performing some analysis. Windowing is often used to minimize the effects of spectral leakage when analyzing a finite segment of a signal. The Bartlett window is designed to taper the ends of a signal segment smoothly to zero, making it suitable for reducing spectral leakage in situations where the signal's frequency content changes gradually.

The Bartlett window is defined as a triangular function, with its values ramping linearly from zero at the beginning and end of the window to a maximum value in the middle. This triangular shape helps to suppress spectral leakage by smoothly tapering the signal's edges. Mathematically, the Bartlett window $w(n)$ of length N is defined as:

$$w(n) = \begin{cases} \frac{2n}{N-1} & \text{if } 0 \leq n \leq \frac{N-1}{2} \\ 2 - \frac{2n}{N-1} & \text{if } \frac{N+1}{2} \leq n \leq N-1 \end{cases}$$

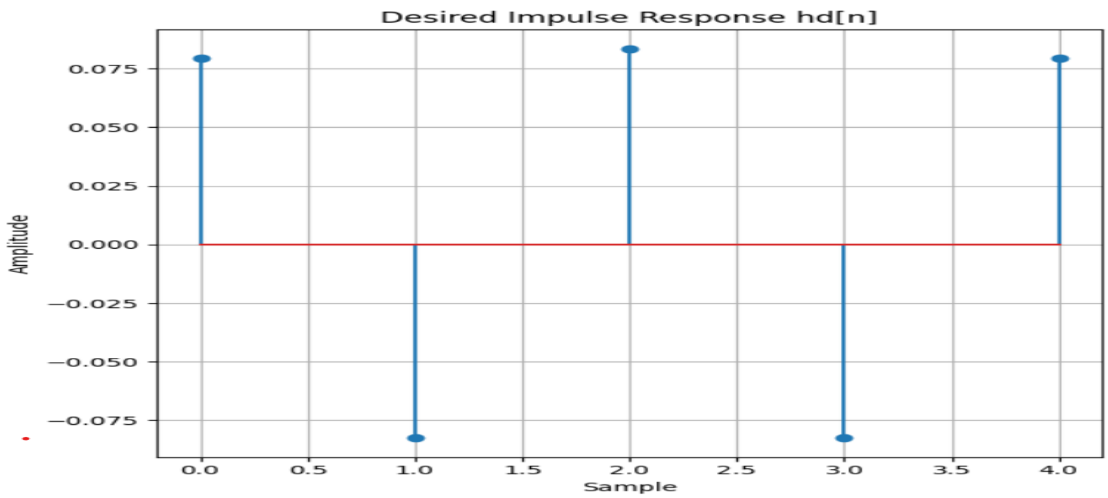
where n is the index of the sample in the window, and N is the total number of samples in the window. The Bartlett window is symmetric around its midpoint.

2 Implementation In Python

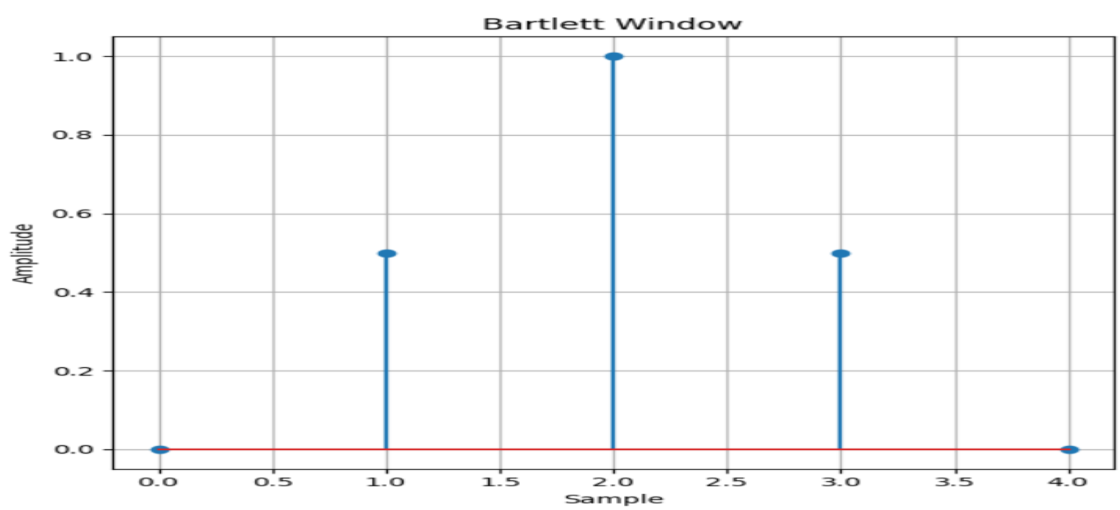
2.1 Parameters

Order:: 5
Digital Cutoff frequency:: $f_c = 550$ Hz
Sampling Frequency::f_s 1200 Hz
Analog Cutoff Frequency :: $w_c = \frac{2\pi * f_c}{f_s} = 2.88$ rad/s

2.2 Desired Impulse Response

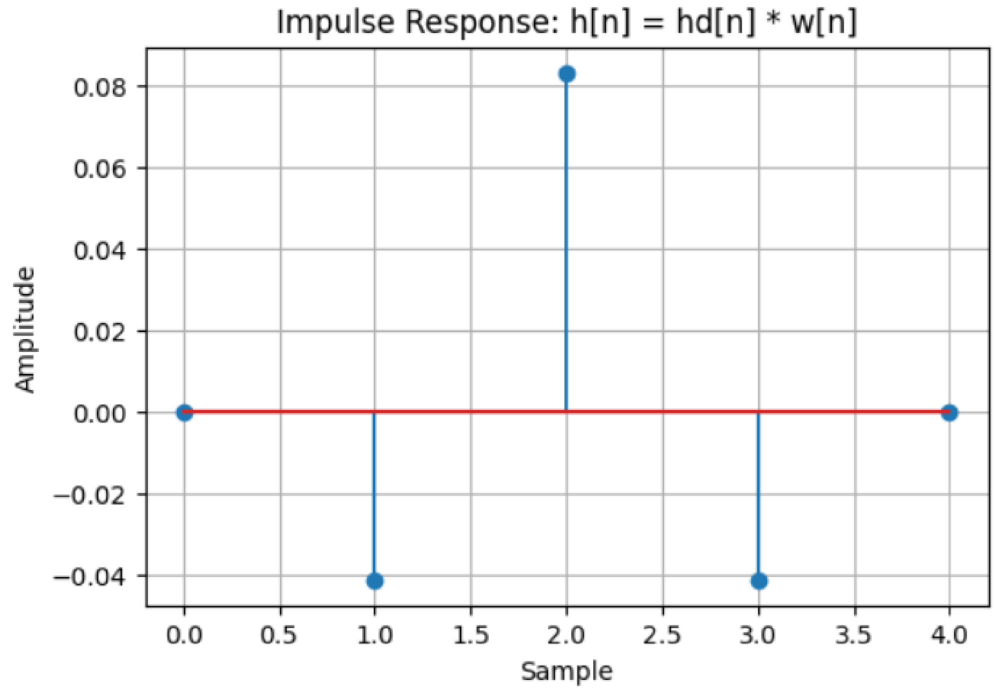


2.3 Bartlett Window

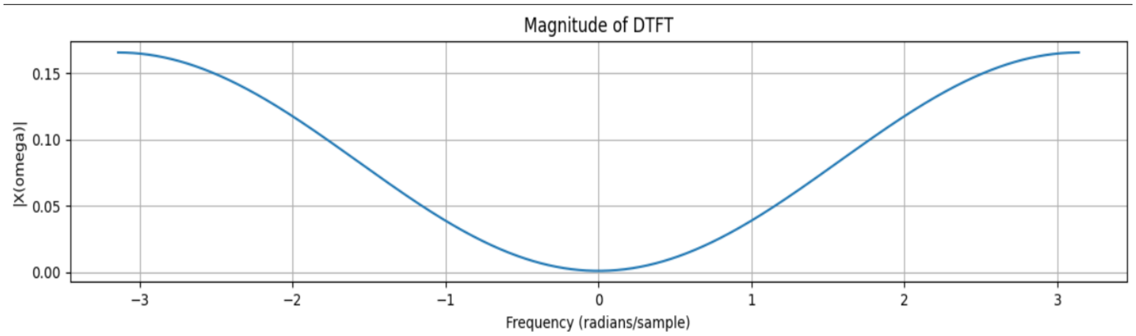


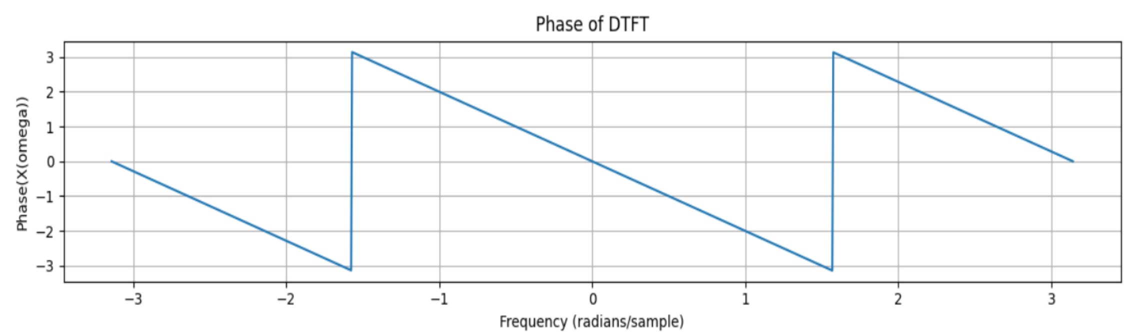
2.4 Coefficients of FIR Filter

h[n]=[0. -0.04119233 0.08333333 -0.04119233 0.]

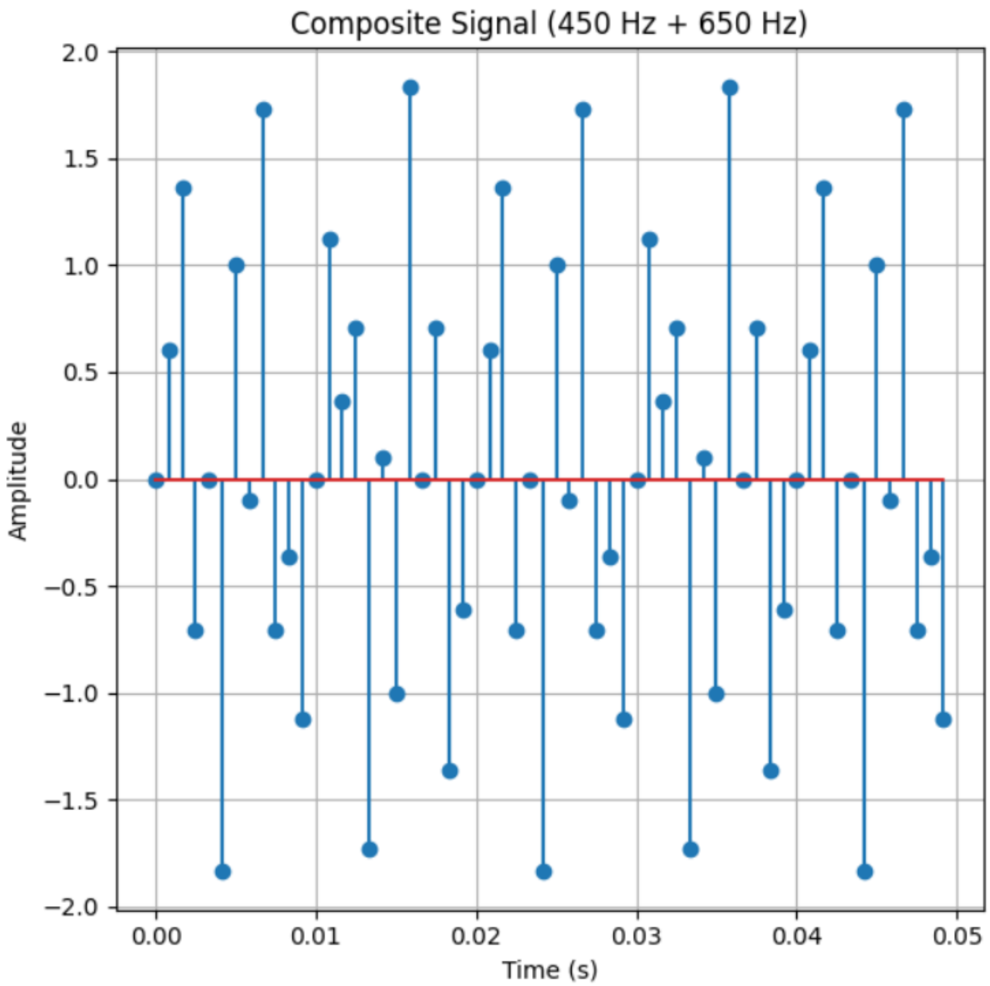


2.5 Frequency Response of FIR High Pass Filter

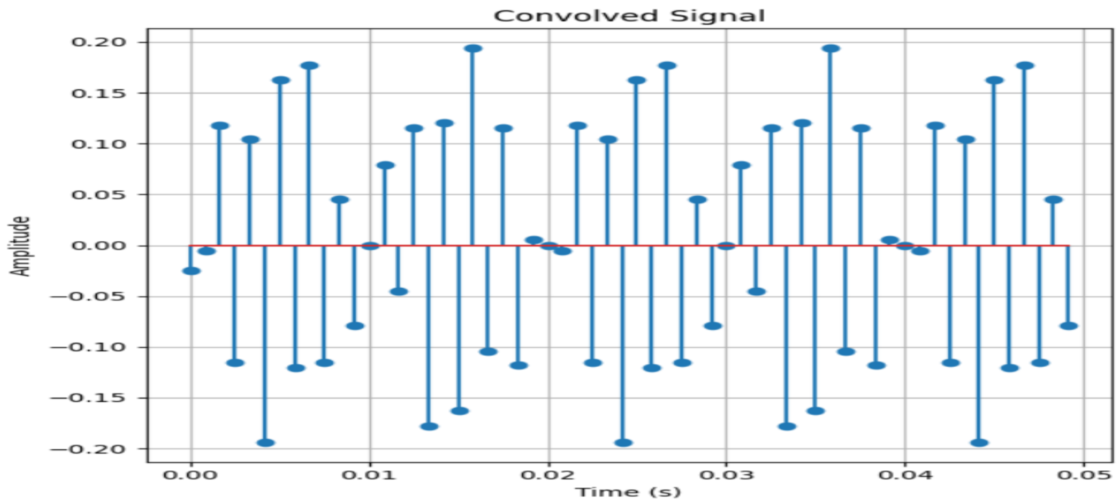




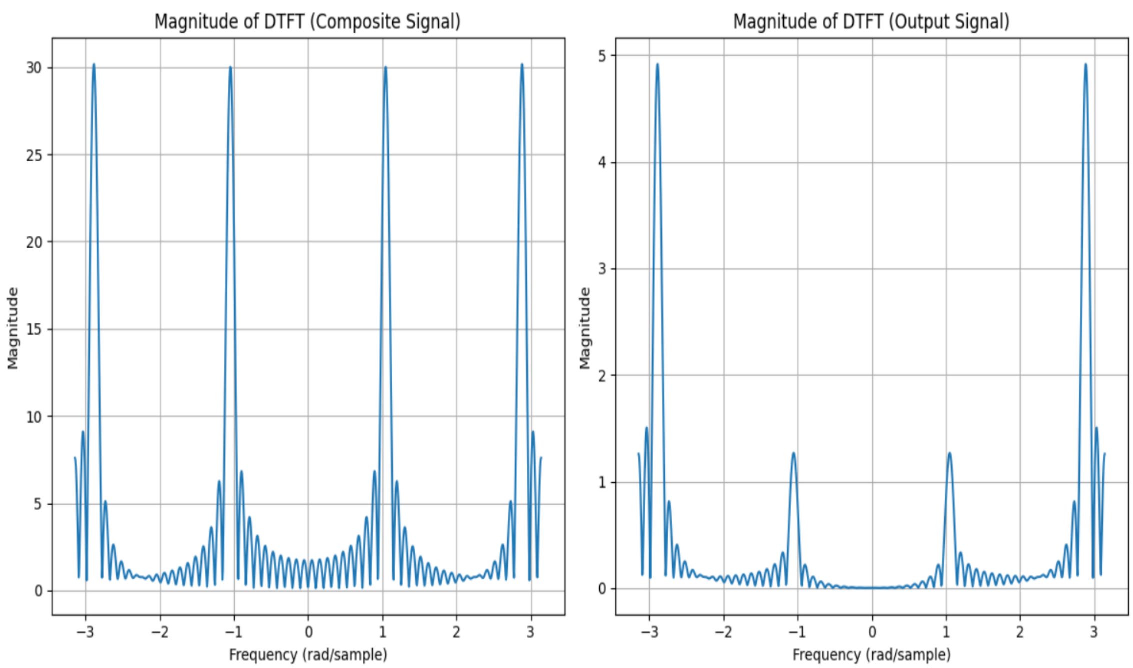
2.6 Composite signal



2.7 Filter Output



2.8 Frequency Response of Output Signal



3 Verilog Implementation

3.1 Illustration of 8 bit Baugh-wooley Multiplication

The 8-bit Baugh-Wooley multiplier is an efficient algorithm used for multiplying two 8-bit binary numbers. It optimizes the multiplication process by reducing the number of required additions and subtractions compared to traditional methods. The algorithm generates partial products by bitwise AND operations and then accumulates these partial products using optimized addition and subtraction operations. By exploiting the properties of two's complement arithmetic and streamlining the multiplication process, the Baugh-Wooley algorithm offers a faster and more resource-effective approach to binary multiplication, making it suitable for hardware implementation in digital circuits and arithmetic units.

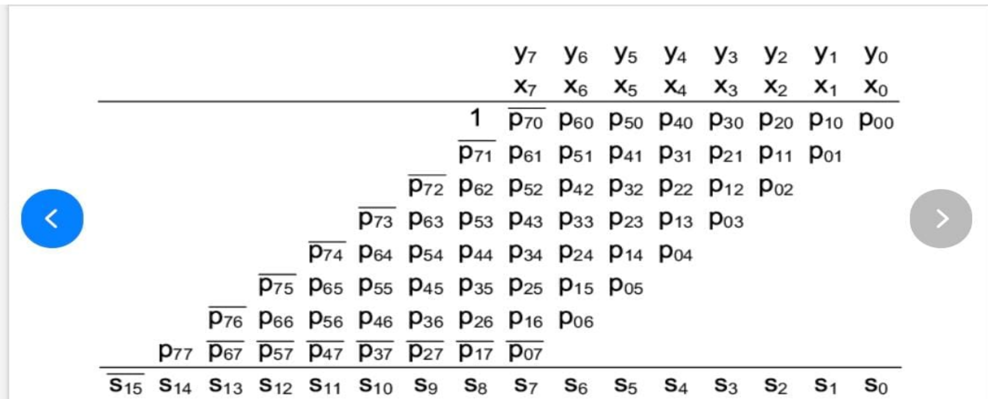
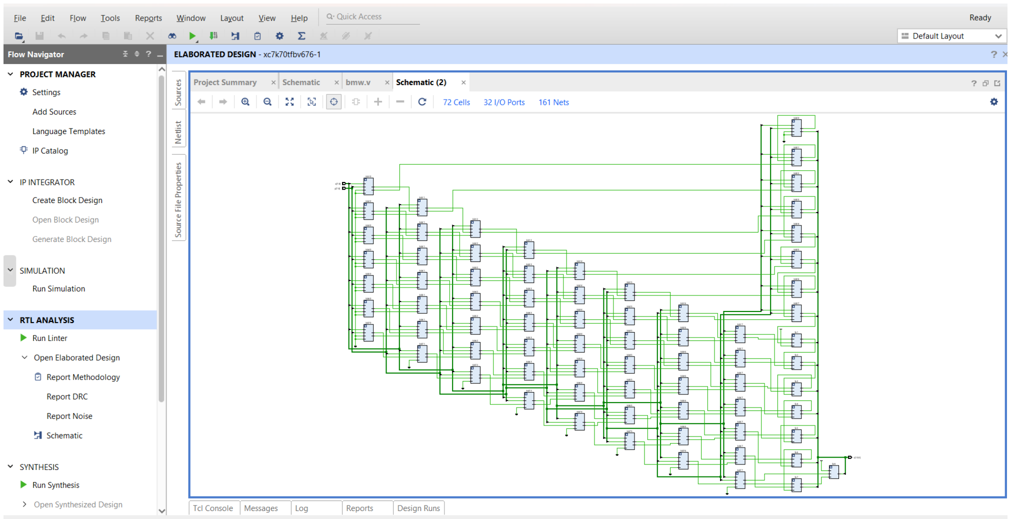
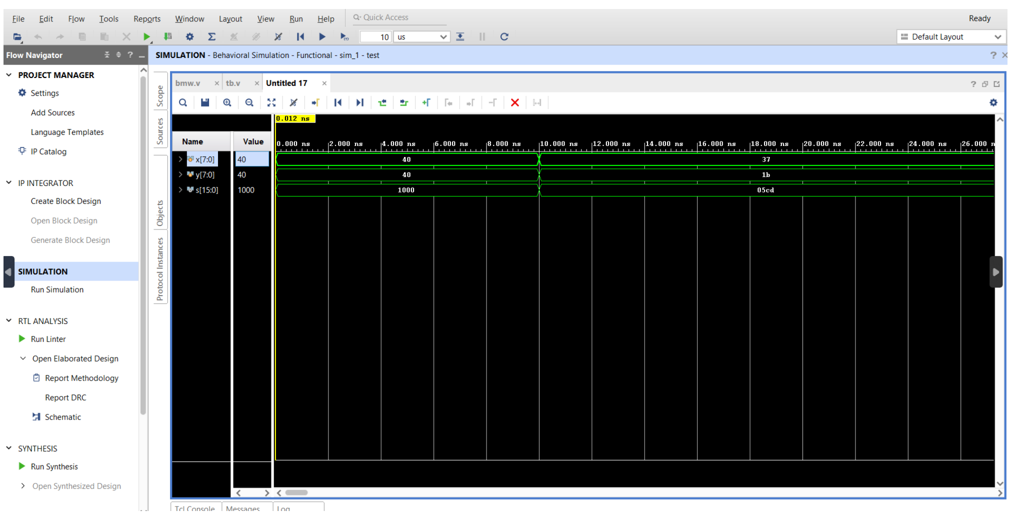


Illustration of an 8-bit Baugh-Wooley multiplication.

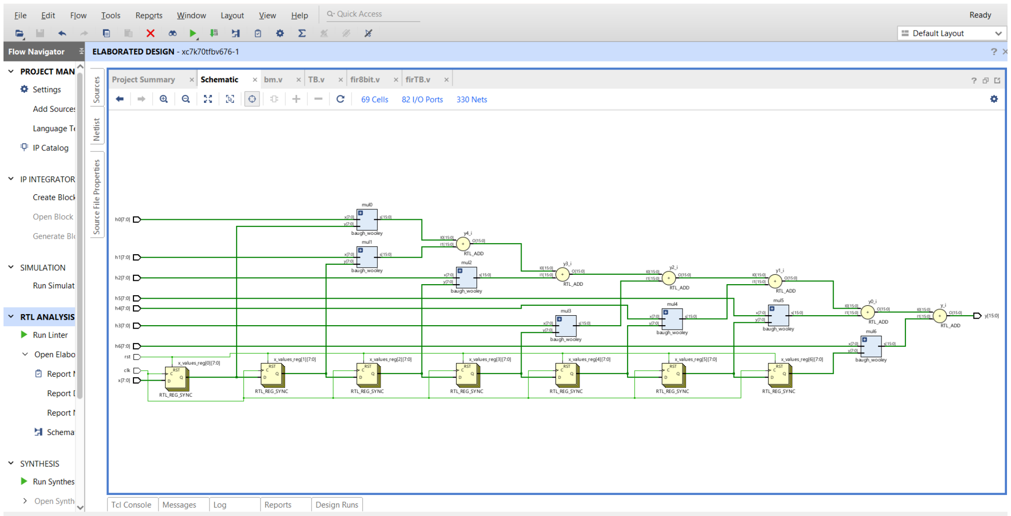
3.2 Schematic of 8-bit Baugh-Wooley Multiplier



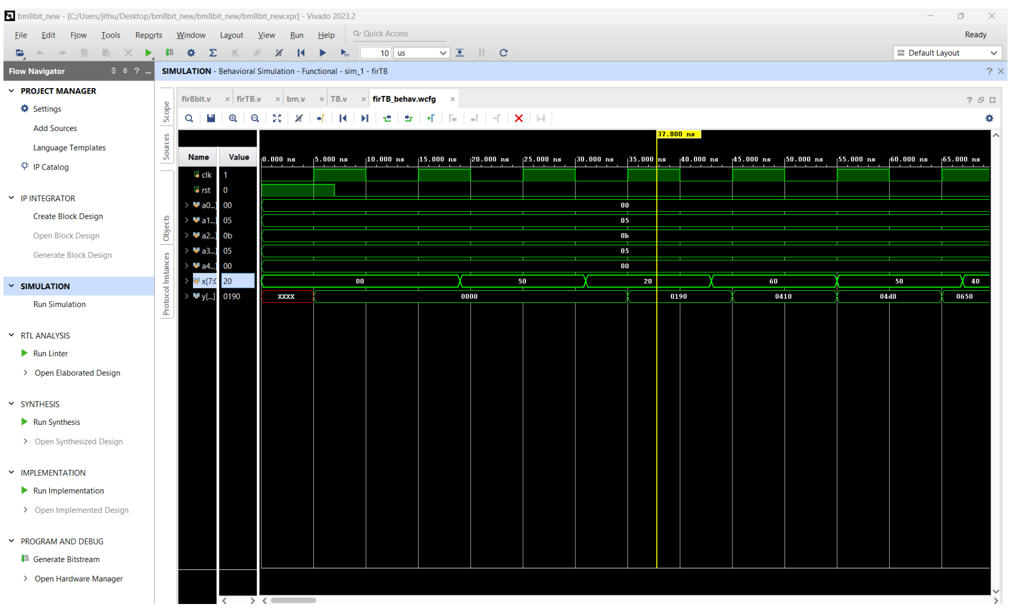
3.3 Simulation of 8-bit Baugh-Wooley Multiplier



3.4 Schematic of FIR High Pass Filter



3.5 Simulation of FIR High Pass Filter



4 Conclusion

In this project, we successfully designed and implemented a Finite Impulse Response (FIR) highpass filter using the Bartlett window in Python for coefficient generation, Using the coefficients generated the FIR Highpass Filter was simulated in verilog. Through meticulous parameter selection and optimization, the filter exhibited desirable frequency response characteristics, effectively attenuating low-frequency components while preserving higher frequencies. This project exemplifies the seamless integration of theoretical concepts into practical applications, offering a pathway for innovative solutions in digital signal processing.