IJournals: International Journal of Software & Hardware Research in Engineering
ISSN-2347-4890
Volume 4 Issue 2 February, 2016

OPTIMIZATION OF FPGA RESOURCE USAGE IN TODAYS DESIGN AND APPLICATION (AN FPGA IMPLEMENTATION)

Warsame H. Ali, Emmanuel S. Kolawole
Department of Electrical and Computer Engineering, Prairie View A&M University, Prairie View, Texas
Email: whali@pvamu.edu, ekolawole@student.pvamu.edu

ABSTRACT

In today's technological advancement, using the right tools to make a decision while designing and implementing is very important to the designers and decision makers. In this paper two methods, Serial and Parallel implementation, are used to validate the effective design and implementation in terms of speed, resource usage, time to completion and power consumption in the design and implementation of FPGA in today's design and applications. This paper presents the design and implementation of a Low-pass, High-pass and a Band-pass Finite Impulse Response (FIR) Filter using SPARTAN-6 Field Programmable Gate Array (FPGA) device. The Filter performance is tested using Filter design and Analysis (FDA) and FIR tools from Mathworks. System Generator ISE design suit 14.6i is used in synthesizing and co-simulation for FPGA Filter output verification. Finally, comparison is done between the results obtained from the software simulations and those from FPGA using hardware co-simulation. The simulation waveforms and synthesis reports verify the parallel implementation of FPGA which proves its effectiveness in terms of speed, resource usage and power consumption.

Keywords: *Digital Filters, FIR Filter, Matlab Simulink, FDATool, FIR Tool, Distributed Arithmetic, FPGA, Xilinx System Generator*

1. Introduction

In signal processing, the function of a filter is to remove unwanted parts of the signal, such as random noise, or to extract useful parts of the signal, such as the components lying within a certain frequency range[1]. The FIR Filter block diagram is as shown in **Figure 1**[10].

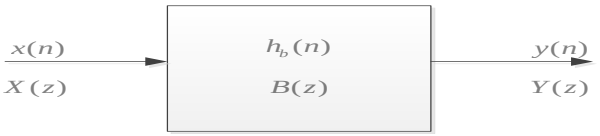

Figure 1. A block diagram of a FIR filter [10]

An ideal filter is a network that allows signals of only certain frequencies to pass while blocking all others. Depending on the region of frequencies that are allowed through or not, filters are characterized as low-pass, high-pass, band-pass, band-reject and all-pass. There are many needs for electric filters, some of the more common being those used in radio and television sets, which allow tuning into a certain channel by passing its band of frequencies while filtering out those of other channels [2].

2. Objective

- Improve overall system performance
- Provides less reasonable cost in implementation
- Increases the speed of operation
- Low power consumption

- Provides a more reliable and stable system
- Easy enhancement of Linear phase characteristic
- Provides low resource usage per design and implementation

3. FIR Filters Design

The main body of this paper is to discuss the following:

- **FDATool Filter Design**

The design of an FIR filter for a specific application includes calculating the coefficients according to different criteria like filter order, sampling frequency, pass-band and stop-band frequencies, etc. The coefficient calculating could be done by different software which makes it as simple task. There are different methods used in designing Digital FIR Filters, such as Equiripple, window, least-square, frequency sampling and interpolated FIR method. In this paper, I have chosen Equiripple method because of its advantages over other methods[3].

- It meets specifications with the least number of coefficients
- Uses less amount of resources on FPGA for implementation
- The weighted approx error between the desired and actual freq response is spread evenly across the passband and stopband of the Filter thereby minimizing error
- Passband and stopband deviations can be specified separately

Below is the Equirriple equation;

$$\varepsilon(j2\pi f) = \max_{f \in [-0.5, 0.5]} \left| Q(j2\pi f)[H_d(j2\pi f) - H(e^{j2\pi f})] \right| \quad (1)$$

where:

$H(e^{j2\pi f})$ is the best approximation frequency response

$H_d(j2\pi f)$ is the ideal frequency response

Q() is the weighting function

$\varepsilon(j2\pi f)$ is the Equiripple factor

Table 1 as shown below elaborates the specifications of Low-pass, High-pass and Band-pass Filters used in this paper;
Fs is the sampling frequency
F-stop is the stopband frequency

F-pass is the passband frequency

A-pass is the passband attenuation

A-stop is the stopband attenuation

Table 1 below shows the filters specifications used in this paper.

Table 1. FIR Filters specifications

Options	Low-Pass Filter	High-Pass Filter	Band-Pass Filter
Design Method	FIR Equiripple	FIR Equiripple	FIR Equiripple
Frequency Specifications	Units: KHz Units: kHz Fs: 1500 F-stop: 300 F-pass: 270	Units: kHz Fs: 1500 F-stop: 450 F-pass: 480	Units: kHz Fs: 1500 F-stop1: 270 F-stop2: 480 F-pass1: 300 F-pass2: 450

Magnitude specification	Units: dB A-stop: 54 dB A-pass: 1dB	Units: dB A-stop: 54 dB A-pass: 1dB	Units: dB A-stop1: 54 dB A-stop2: 54 dB A-pass: 1dB

FIRs have the advantage of being much more realizable in hardware [7] because they avoid division and feedback paths. FIR filter response y(n) is computed by operation between filter coefficients b(k) and the data input x(n) as shown by the equation:

$$y[n] = \sum_{k=0}^{N-1} h_k . x[n-k] \quad (2)$$

where $x[n-k]$ = represents the filter input,

h_k = represents the filter coefficients,

$y[n]$ = represents the filter output

N = represents the number of filter coefficients (order of filter).

The value of the constant k is a minimum value for which the expression $N \le 2^k$ is valid. $k = \lfloor 1 + log_{(2)} N \rfloor$; where the operator $\lfloor \ \rfloor$ represents rounding down to a less value.

In FIR Filter design, Filter frequency response coefficients and the corresponding window type function must be known before Filter hardware realization. **Table 2** shows the transfer function equations used in different types of Filters.[8].

Table 2. The frequency responses of standard ideal Filters[8]

Type of Filter	**Frequency Response** $h_d[n]$
Low-pass Filter	$h_d[n] = \begin{cases} \dfrac{\sin\left[\omega_c (n-M)\right]}{\pi (n-M)}; & n \neq M \\ \dfrac{\omega_c}{\pi}, & n = M \end{cases}$
High-pass Filter	$h_d[n] = \begin{cases} -\dfrac{\sin\left(\omega_c (n-M)\right)}{\pi (n-M)}; & n = M \\ 1 - \dfrac{\omega_c}{\pi}; & n \neq M \end{cases}$
Band-pass Filter	$h_d[n] = \begin{cases} \dfrac{\sin\left(\omega_{c2} (n-M)\right)}{\pi (n-M)} - \dfrac{\sin\left(\omega_{c1} (n-M)\right)}{\pi (n-M)}; & n \neq M \\ \dfrac{\omega_{c2} - \omega_{c1}}{\pi}; & n = M \end{cases}$

The value of variable n ranges between 0 and N, where N is the filter order. A constant M can be expressed as M =N/2. Also M=2N. Since the variable n ranges from 0 and N, the ideal filter frequency response has N+1 sample. h_d [n] is the frequency response of each individual filter classification to be designed; low-pass, high-pass or band-pass filter.

The FIR filter coefficients are found using the following expression:

$$h[n] = w[n].h_d[n]; 0 \le n \le N-1 \quad (3)$$

For direct realization of FIR filter, it is based on the direct implementation of this expression;

$$y[n] = \sum_{k=o}^{N-1} h[k].x[n-k] \quad (4)$$

W[n] is the window function coefficients. In this design, Hamming window function is used based on the filter specification. For Hamming window, the expression for $w[n]$ is given as;

$$w[n] = 0.54 - 0.46\left(1 - \cos\left(\frac{2\pi n}{N-1}\right)\right); 0 \le n \le N-1 \quad (5)$$

Also, **Figure 2** below shows a low-pass filter design using FDA Tool

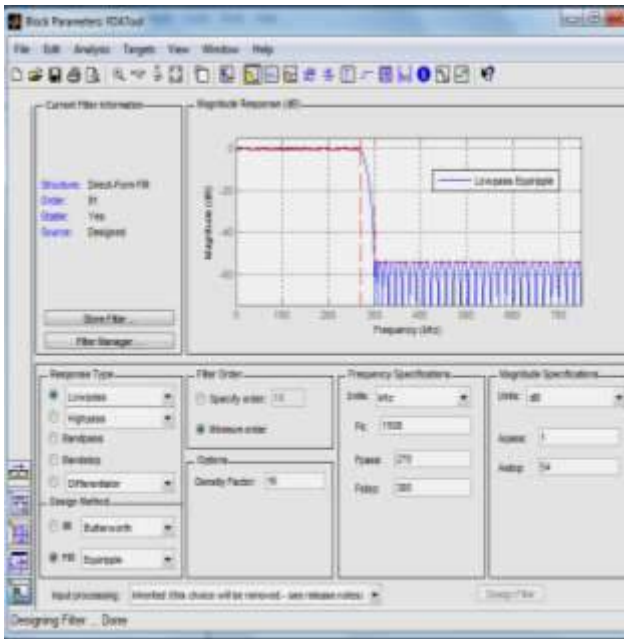

Figure 2. Low-pass Filter using FDATool

- **FIR Compiler & Simulink Model**

In this paper FDATool from Matlab mathematical computational package with digital signal processing toolboxes is used to design filter response and generate coefficients tables. In the proposed approach, FIR tool utilizes distributed arithmetic (DA) as shown in **Figure 3** which actually uses lookup table for storing constant coefficients

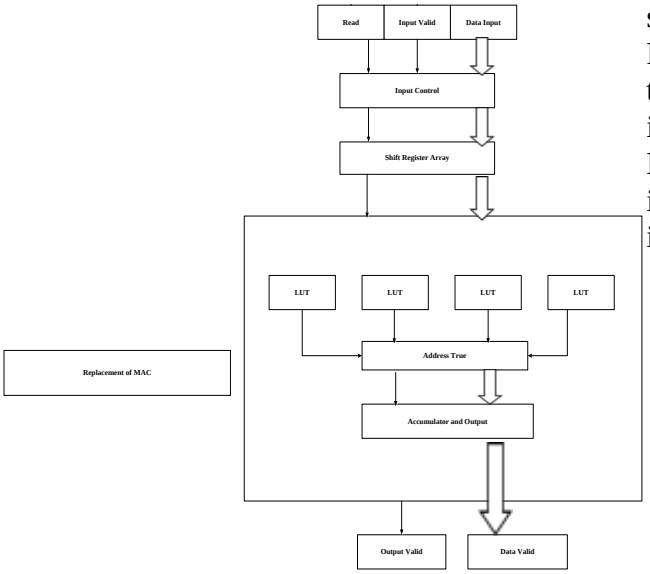

Figure 3. Implementation of FIR Filter using DA [6]

- **Hardware Co-Simulation/System Generator**

This defines the interaction between the hardware and software during the implementation of the filters. Example of co-simulation JTAG generated during the parallel implementation of all the three filters in this paper as shown in **Figure 4** below

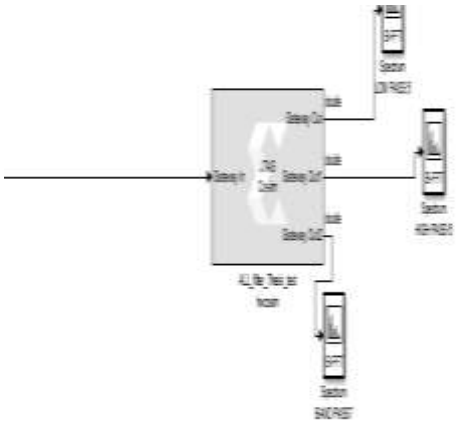

Figure 4. Co-simulation Jtag output of parallel implementation

- **FIR Filter implementation on FPGA**

After designing the filters based on their specifications from Matlab, the Xilinx software package provided by Spatran-6 FPGA boar, System Generator is then used for the appropriate FIR FPGA filter implementation for Low-pass, High-pass, Band-pass filter and the parallel implementation of all the three filters as shown in **Figure 5**, **6, 7 and 8** below.

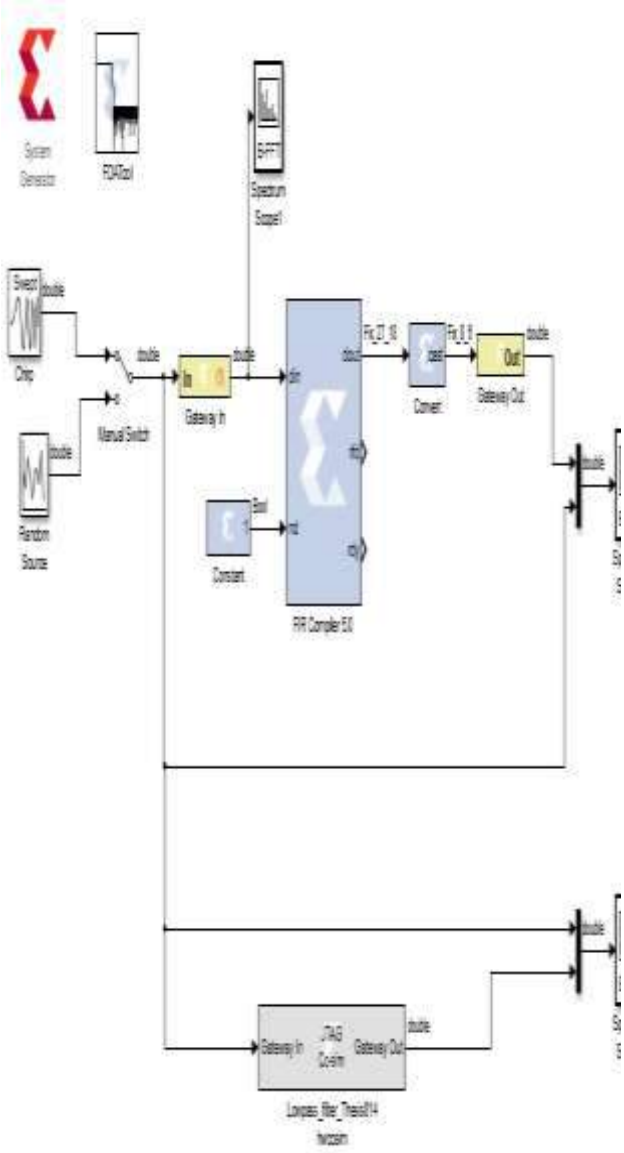

Figure 5. Low-pass FIR Filter

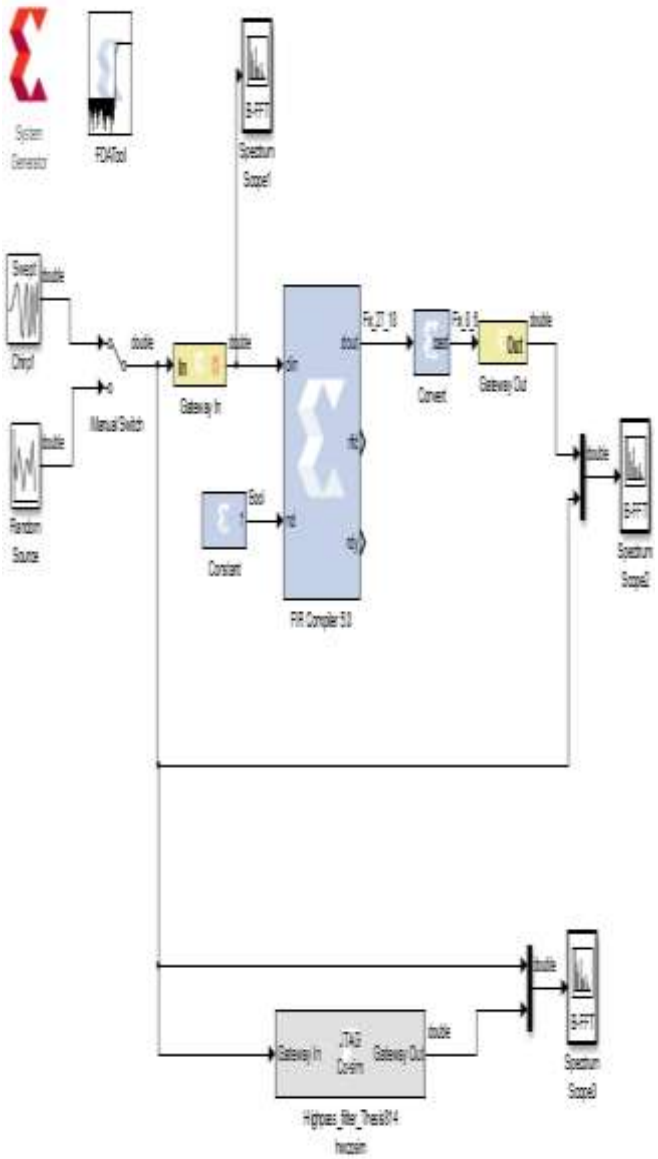

Figure 6. High-pass FIR Filter

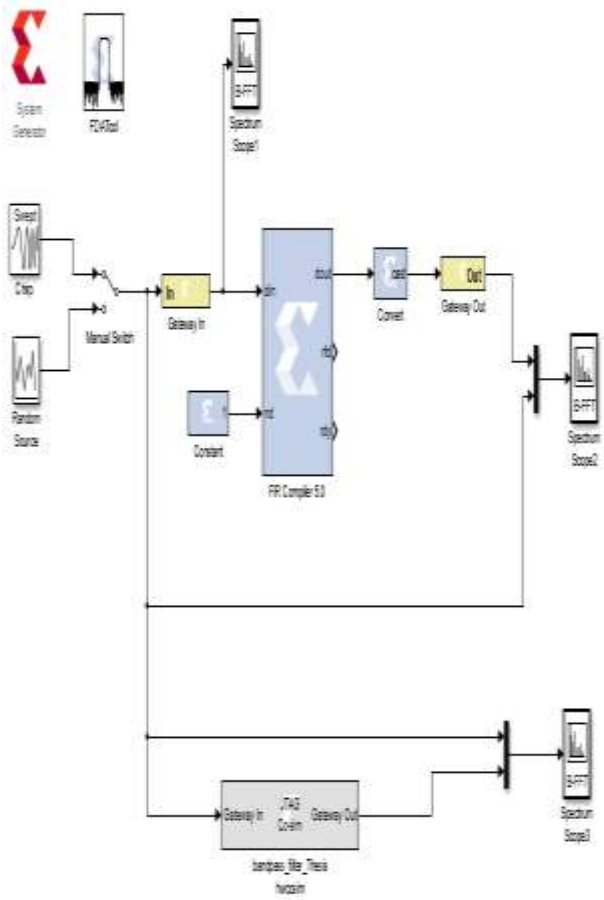

Figure 7. Band-pass FIR Filter

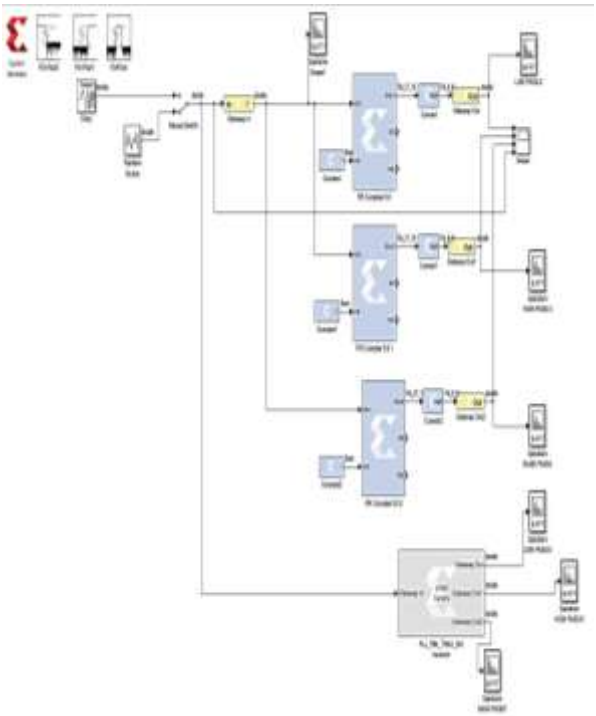

Figure 8. Parallel implementation of FIR Filters

- **Simulation/FPGA implementation comparison**

This shows the simulation results among Low-pass, Band-pass and High-pass filters used in this paper and their comparison. The graphs below shows simulation outputs for Low-pass filter, Band-pass and High will be shown in the final paper. **Figures 9b and 9c** verified the comparison between the Low-pass Filter simulation from Matlab and FPGA implementation. **Figure 9a** is the Input signal below (F_C) using chirp source. Where (F_C) is the cut-off frequency.

Figure 9a. Input signal below (F_C) using chirp source

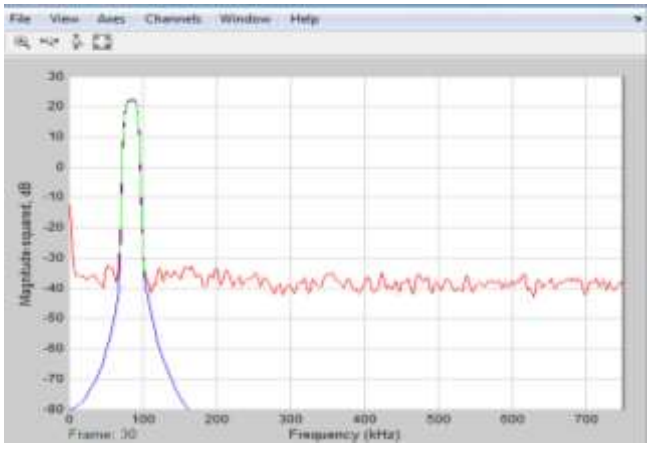

Figure 9b. Output waveform from Low-pass FIR filter using Matlab

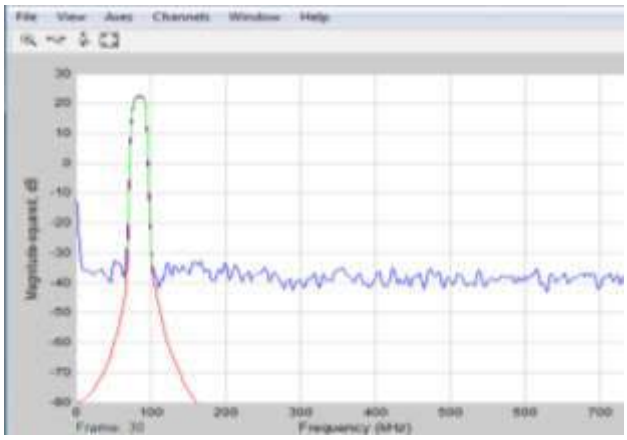

Figure 9c. Output waveform from Low-pass FIR filter using FPGA

Figures 9d. 9e and 9f shown below analyzed when the input signal is above the cut-off frequency and no output waveforms for Low-pass frequency.

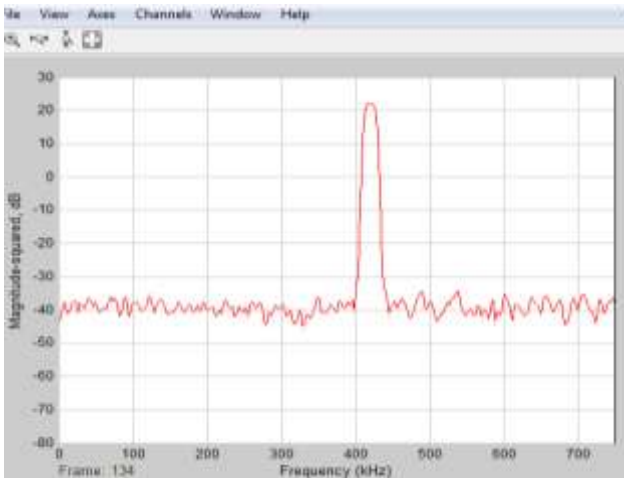

Figure 9d. Input signal above (F_C) using chirp source

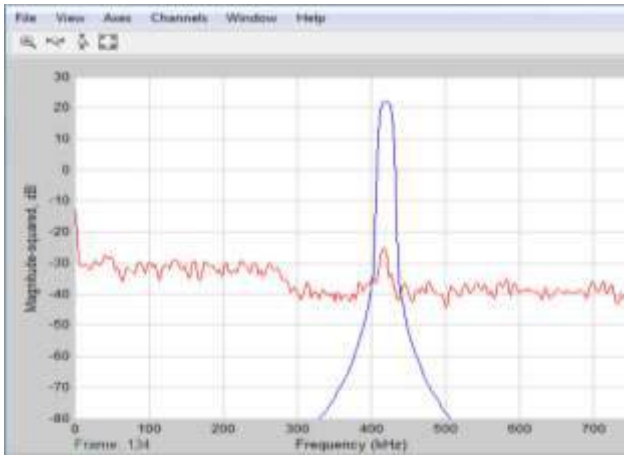

Figure 9e. Output waveform from low-pass FIR filter using Matlab

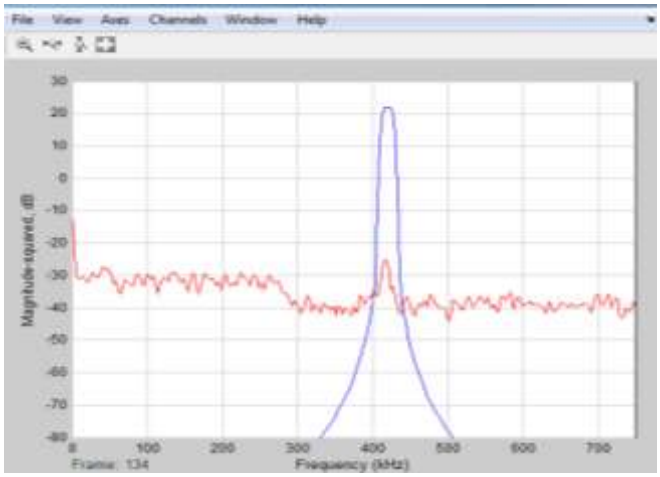

Figure 9f. Output waveform from low-pass filter using FPGA

4. Discussions

In this paper, a ninety two-order low-pass and band-pass and ninety nine-order high-pass FIR filter have been implemented in Spartan-6 LX75T-3FGG676C FPGA board using Xilinx Integrated Software Environment (ISE). The propose idea gives an efficient approach towards the implementation of FIR filter using Simulink and subsequent synthesis on FPGA that suited different applications unlike most previous approaches on a penalty of reducing computation speed. **Table 3** shows the comparison among the Filters and their parallel implementation as drafted from the synthesis reports.

Table 3. Comparison among Filters

Categories	Low-pass Filter	High-pass Filter	Band-pass Filter	Parallel implementation on FPGA
Minimum period(ns)	10.57	11.16	10.14	**15.03**
Peak memory	294	300	294	**410**

usage(MB)				
Power consumption (mW)	119	127	118	**273**

5. Conclusion

This paper discusses the implementation of Low-pass, High-pass and Band-pass FIR Digital Filters on an FPGA. The simulation results show that the output waveforms obtained from the software simulations correspond with those from FPGA implementation respectively using hardware co-simulations. Also, the parallel implementation proves that performance of a low-pass filter is not affected by both high-pass or band-pass filter and vice-versa, and the synthesis report equally shows less resource usage, speed increase, cost effectiveness, more flexibility and low power consumption which validate the parallelism nature of FPGA.

REFERENCES

[1] Ruan,A.W., Liao, Y.B., Li,J.X., "An ALU-Based Universal Architecture for FIR Filters" *IEEE in proc. International Conference on communications, Circuits and Systems*, pp. 1070-1073, 2009.

[2] P. Panayotatos, "Frequency Response of Filters", © Rutgers University, December 2005.

[3] Abdul Quayyum & Moona Mazher "Design of Programmable, Efficient Finite Impulse Response Filter Based on Distributive Arithmetic Algorithm," *International Journal of Information Technology and Electrical Engineering*, vol. 1, issue 1, Dec. 2012.

[4] Monmasson, E et al. "FPGAs in Industrial Control Applications," *IEEE Transactions on Industrial informatics*, vol. 7, issue 2, pp. 224-243, May 2011.

[5] Wenjing,H., Guoyun,Z., Waiyun,L., "Self-Programmable Multipurpose Digital Filter Design Based On FPGA" in proc. *IEEE International Conference on Internet Technology and Applications (iTAP)*, ,PP. 1-5, 2011.

[6] S. Mohanakrishnan, B.E joseph, "Automatic Implementation of FIR Filters on Field Programmable Gate Arrays," *IEEE Signal Processing Letters*, vol.2 issue 3, March 1995.

[7] Dr. H.J. Manal, H.S. Asaad, "High-Pass Digital Filter Implementation Using FPGA,"*IEEE International Journal of Advanced Computer Science and Applications (IJACSA)*, vol. 13 issue 3, June 2013.

[8] "Filter design Methods for FPGAs", Accel Chip, 2006.

[9] Xilinx white paper number 6984, www.xilinx.com

[10] Louis Litwin, FIR and IIR digital filters, *IEEE potentials* 2002.