Design of FIR Filter on FPGAs using IP cores

Apurva Singh Chauhan[1], Vipul Soni[2]

[1,2]Assistant Professor, Electronics & Communication Engineering Department
JECRC UDML College of Engineering, JECRC Foundation, Jaipur (Rajasthan)

apurvasinghchauhan.fet@gmail.com[1]
soni.vipul5@gmail.com[2]

Abstract

The paper describes the development of FIR filters on Field programmable gate array (FPGAs) using IP cores. FIR filter has been designed and realized by FPGA for filtering the digital signal. The implementation of FIR filter on a Xilinx XC3S400FPGA is considered and the coefficients are computed through the Hamming windowing technique. The model is capable of performing filtering operations like low pass, high pass, band pass and band stop based on selection that is embedded into the design. The most basic functions required for nearly any signal processor include addition, multiplication and delays. The filter is set to 16-bit signed data processing. IP Corse has been used to filter the input data. The design is coded through VHDL (hardware descriptive language). To verify the designed outputs simulation, compilation and synthesis have been done. To test the correctness of the design the observed output is compared with the calculated output results from MATLAB implementation that confirms the effectiveness of the design.

Keywords: *FIR filter, FPGA, VHDL code, MATLAB.*

1. Introduction

FIR filter has been designed and realized on FPGA for filtering the digital signal. This technique can be applied to any FPGAs. Signal processing is an important area where FPGAs have found many applications in recent years. FPGA contains over a million equivalent logic blocks (logic gates and tens of thousands of flip-flops) [1][2].This means that it is not possible to use traditional methods of logic design involving the drawing of logic diagrams when the digital circuit may contain thousands of gates. The reality is that today digital systems are designed by writing software in the form of hardware description languages (HDLs) [5]. Computer-aided design tools are used to both simulate VHDL design and to synthesize the design to actual hardware.

The Xilinx Floating-Point core is a function inbuilt in IP cores provides designers with the means to perform floating-point arithmetic on an FPGA. The core can be customized to allow optimization for operation, word length, latency, and interface. Available for Virtex™-II, Virtex-II Pro, Virtex-4, Virtex-5, Spartan™-II, Spartan-3, and Spartan-3E FPGA family members.The core employs a floating point representation that is a generalization to allow for non-standard sizes [1].When standard sizes are chosen, the format and special values.

The designing of an FIR filter in VHDL with MATLAB (for the generation of coefficients of filter) and programming it onto an FPGA is explained in this paper. Implementation of project onto an FPGA (including hardware and software parts) VHDL, MATLAB and basic digital filter concepts are used.

2. FIR Filters and its Characteristics

FIR filters are digital filters with finite impulse response. They are also known as non-recursive digital filters as they do not have the feedback (a recursive part of a filter) [3]. Finite Impulse Response (FIR) filters are defined by scaled and time-delayed versions of the filter input signal only, as given by the following difference equation:

$$y[n] = b_0\, x[n] + b_1\, x[n-1] + b_2\, x[n-2]..., \quad n=0,1,2 \qquad \text{.......... (1)}$$

where the input and output y[n]=0 for n < 0. An FIR filter can be represented by a block diagram as shown in Figure 1.

The z^{-1} terms represent unit delays while this representation for a delay element is common and widely accepted in the signal processing community, the specification of delay in terms of powers of z is a z-domain characterization while the block diagram itself is a time-domain representation.

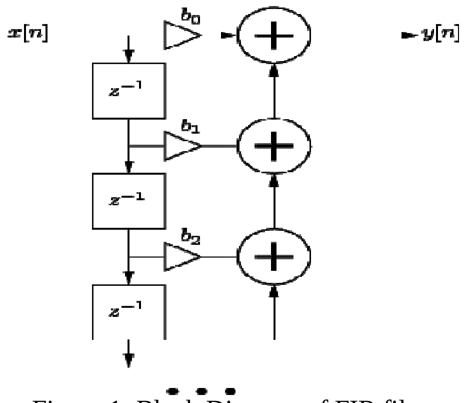

Figure 1: Block Diagram of FIR filter

With respect to the filter block diagram, FIR filters make use of feed forward terms only.

- The impulse response of an FIR filter is only as long as the maximum delayed input term in its difference equation.
- The maximum possible gain of an FIR filter is given by the sum of input terms, scaled by their coefficients, in its difference equation.
- The summation of feed forward input terms can result in destructive signal.
- The FIR filter is said to have an order equivalent to the number of unit delays in its difference equation interference, or cancellations, at certain frequency values.

2.1 Filter Specification

The mainly filter specifications includes

(i) Signal characteristics.
(ii) The characteristics of the filter.
(iii) The manner of implementation.

Although the above specification is application dependent it will be helpful to devote some time on the characteristics of the filter. The characteristics of digital filters are often in specified in the frequency domain. For frequency selective filters, such as low-pass and band-pass filters, the specifications are often in the form of tolerance. Figure 2 shows the magnitude frequency response specifications for a low-pass filter.

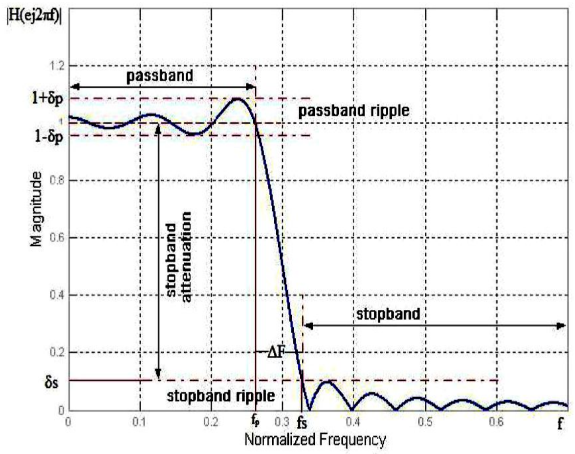

Figure 2: Magnitude frequency response specifications for a low-pass filter

In the pass-band, the magnitude response has a peak deviation of ap and in the stop-band; it has a maximum deviation of as [3] [7]. The width of transition band determines how sharp the filter is. The magnitude response decreases monotonically from the pass-band to stop-band in this region.

The following are the key parameters of interest:

- ap peak pass-band deviation(or ripples)
- as stop- band deviation.
- fs stop- band edge frequency.
- fp pass- band edge frequency.
- Fs sampling frequency.

The edge frequencies are often given n the normalized form, that is as the fraction of the sampling frequency (f/Fs). Pass-band and stop-band deviation may be expressed in decibels. When they specify the pass-band ripples and minimum stop-band attenuation respectively. Thus the minimum stop-band attenuation, As and the peak pass-band ripple,

- Ap, in decibels are given as
- As (stop-band attenuation) = -20log10 as
- Ap (pass-band ripple) = 20 log10 (1+ap)

The difference between fs and fp gives the transition width of the filter. Another important parameter is the filter length, N, which defines the number of filter.

3. Filter Coefficient Design

The window, optimal and frequency sampling method are the most commonly used for designing filter coefficient [8] [9]. In this paper the window method is used for designing the FIR filter. Hamming window method is considered here for designing purpose. Flow chart in the Figure 3 shows the generation of the coefficient of the FIR filter generated through MATLAB software. As seen in the flow chart, coefficients generation requires three inputs, first is the starting and ends of the band, second is the input sampling frequency and finally the order of the filter. The coefficient generated for the eight stages FIR filter so the order of the filter is eight.

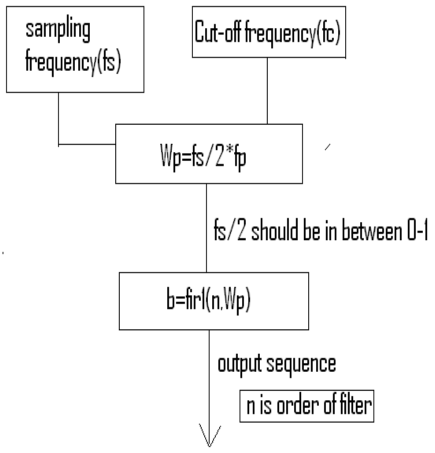

Figure 3: Flow Chart for Generation of Coefficient

3.1 Window Method

In this method the frequency response of a filter and the corresponding impulse, are related by the Fourier transform

$$h_D(n) = \frac{1}{2\pi}\int_{-\pi}^{\pi} H_D(\omega)e^{j\omega n}d\omega \qquad \text{………… (2)}$$

The ideal low pass frequency response is shown in Figure 4, where ω_c is the cutoff frequency and the frequency scale is normalized: T=1 the responses from. $-\omega_c$ to $+\omega_c$.

$$\begin{aligned} h_D(n) &= \frac{1}{2\pi}\int_{-\pi}^{\pi} 1 * e^{j\omega n}d\omega = \frac{1}{2\pi}\int_{-\omega_c}^{\omega_c} e^{j\omega n}d\omega \qquad \text{……….. (3)} \\ &= \frac{2f_c}{n\omega_c}\sin(n\omega_c) \quad , n \neq 0, -\infty \leq n \leq \infty \\ &= 2f \quad , n = 0 \end{aligned}$$

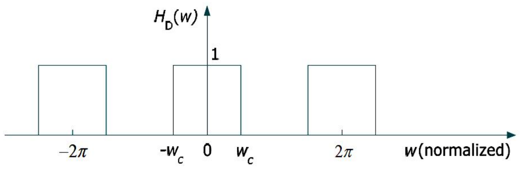

Figure 4: Ideal frequency response of a low pass filter

The ideal frequency response obtained in prior experiments is shown in the given Table 1.

Table 1: Summary of ideal impulse responses

Filter type	$h_D(n), n \neq 0$	$h_D(0)$
Lowpass	$2f_c \frac{\sin(n\omega_c)}{n\omega_c}$	$2f_c$
Highpass	$-2f_c \frac{\sin(n\omega_c)}{n\omega_c}$	$1-2f_c$
Bandpass	$2f_2 \frac{\sin(n\omega_2)}{n\omega_2} - 2f_1 \frac{\sin(n\omega_1)}{n\omega_1}$	$2(f_2 - f_1)$
Bandstop	$2f_1 \frac{\sin(n\omega_1)}{n\omega_1} - 2f_2 \frac{\sin(n\omega_2)}{n\omega_2}$	$1-2(f_2 - f_1)$

Here the multiplication of the ideal frequency response with a window function is takes place. When this window is multiplied by the ideal transfer function then all the coefficients within the window are retained and all that are outside the window are discarded [4].

3.2 Hamming Window

w = hamming (L) returns an L-point symmetric Hamming window in the column vector w. L should be a positive integer. The coefficients of a Hamming window are computed from the following equation.

$$w(n) = 0.54 - 0.46\cos\left(2\pi\frac{n}{N}\right), \quad 0 \le n \le N \qquad \ldots\ldots\ldots\ldots\ldots.. (4)$$

The window length is L=N+1 w = hamming (L,*'sflag'*) returns an L-point Hamming window using the window sampling specified by *'sflag'*, which can be either 'periodic' or 'symmetric' (the default). The periodic flag is useful for DFT/FFT purposes, such as in spectral analysis. The DFT/FFT contains an implicit periodic extension and the periodic flag enables a signal windowed with a periodic window to have perfect periodic extension. When 'periodic' is specified, hamming computes a length L+1 window and returns the first L points. When using windows for filter design, the 'symmetric' flag should be used.
For the sake of implementation the following specifications are considered for FIR filter

- Cut off frequency =70 Hz
- Sampling frequency =1.6 kHz
- Order of the filter =8

The Figure 5 shows the sinusoidal input of 70Hz wave which is given to filter that is digitized in the MATLAB and generated the equivalent hexadecimal values which are used in VHDL coding for filter design. There are basically two inputs should be here in the digital form for the digital FIR filter designing. Table 2 shows the hexadecimal values of input sinusoidal wave. The graph is showing the input plot of the input wave.

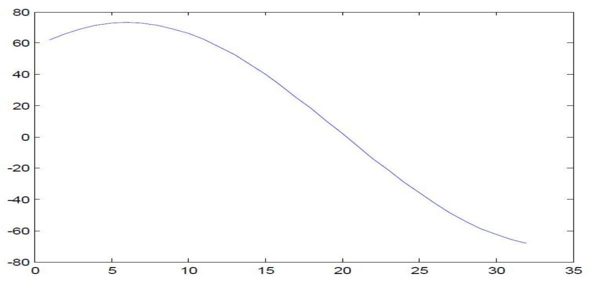

Figure 5: Plot of the Input Wave

Table 2: Inputs for 70 Hz wave

Inputs	Decimal	Hexadecimal
X1 (n)	0.0037	46
X2 (n)	0.0317	46
X3 (n)	0.1255	45
X4 (n)	0.2498	44
X5 (n)	0.3084	41
X6 (n)	0.2498	3E
X7 (n)	0.1255	3A
X8 (n)	0.0317	35

Table 3 given below shows the digitized form of coefficients for eight stage FIR filter.

Table 3: Coefficients for 70 Hz wave

Coefficients	Decimal	Hexadecimal
H1 (n)	0.0037	17
H2 (n)	0.0317	19
H3 (n)	0.1255	26
H4 (n)	0.2498	3B
H5 (n)	0.3084	46
H6 (n)	0.2498	3B
H7 (n)	0.1255	26
H8 (n)	0.0317	19

The given Figure 6 shows the magnitude and phase response of filter.

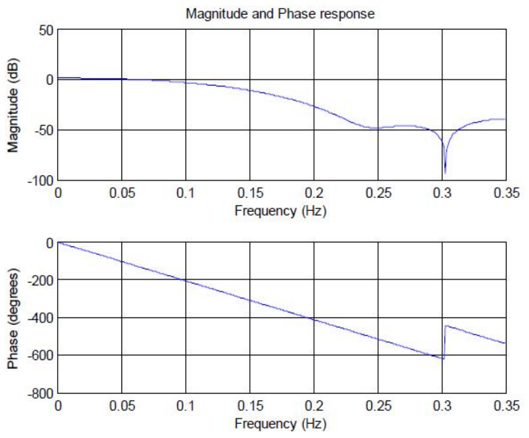

Figure 6: Magnitude and Phase response of Filter

4. FIR Filter Implementation on FPGAs

The coefficient and input sinusoidal signal is generated in the MATLAB and it converted into hexadecimal values by digitization. The hexadecimal values of coefficient and signal are proceeded to obtain the desire output through the multiplication and addition is done with IP cores which is inbuilt in Xilinx 10.1 software. Figure 7 shows the design flow of the entire process of FIR filter implementation on FPGA through VHDL coding done in Xilinx ISE design suit 10.1 versions.

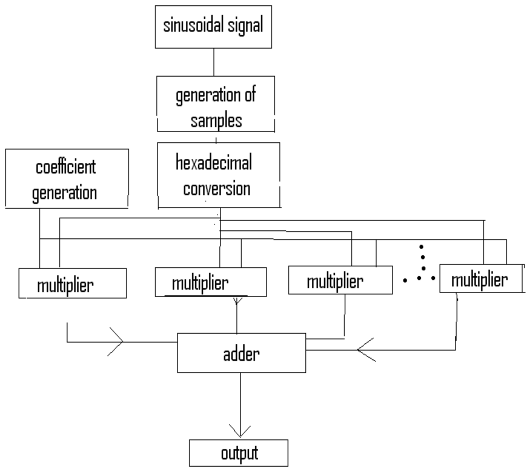

Figure 7: Flow Chart for the VHDL Coding

A basic step involves designing digital filter:

1. Determine a desired response or a set of desired response.
2. Select a class of filter (in this case filter is FIR) for approximating the desired response.
3. Establish a criterion for the response of a filter in the selected class compared to the desired response.
4. Synthesize the filter using a proper structure and a proper implementation form.
5. Analyse the filter performance.

The VHDL code (for FIR filter on FPGAs) is implemented on Spartan-3 Starter FPGA Kit board and Snap shot of Spartan-3 [11]. Figure 8 shows FPGA Kit in which filter is implemented and it includes the following components and features:

- 200,000-gate Xilinx Spartan-3 XC3S400 FPGA in a 256-ball thin Ball Grid Array
- Package (XC3S400FT256)
- 4,320 logic cell equivalents
- 9-pin RS-232 Serial Port
- DB9 9-pin female connector (DCE connector)
  RS-232 transceiver/level translator

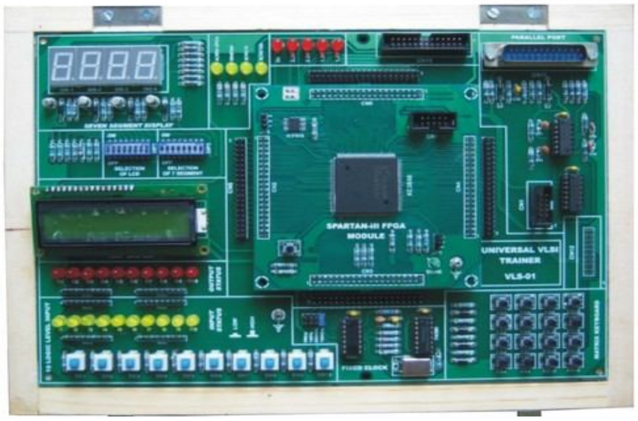

Figure 8: Snap shot of Spartan-3 FPGA Kit

This section shows the simulation results generated through VHDL code in Xilinx design suit. The Figure 9 and 10 shows input and output simulated result of FIR filter, it passes all the input which has given below the cut-off frequency.

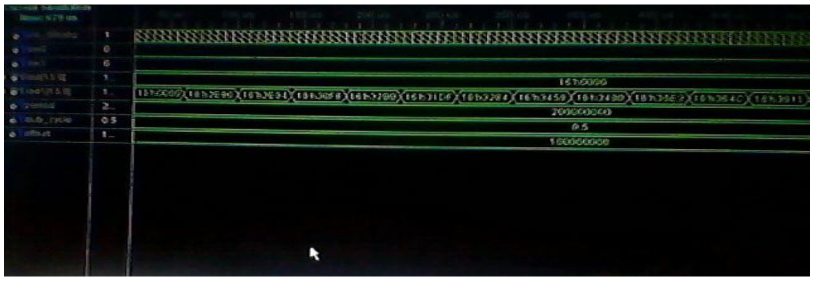

Figure 9: Input waveform of Low Pass FIR Filter

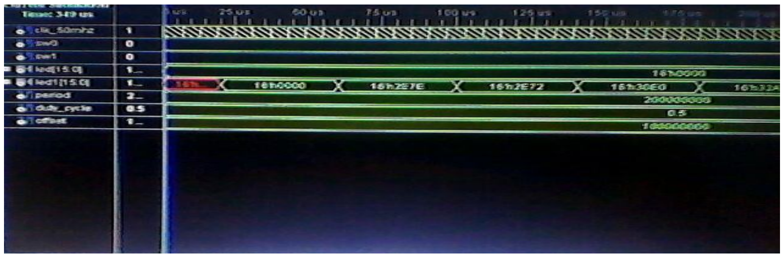

Figure 10: Output waveform of Low Pass FIR Filter

5. Result and Conclusion

The FIR filters are widely used in signal processing and can be implemented using programmable digital processors. But in the realization of large order filters the speed, cost, and flexibility is affected because of complex computations and the development of FIR filters on FPGAs enhanced speed of signal processing. This is due to the fact that the hardware implementation of a lot of multipliers can be done on FPGA which are limited in case of programmable digital processors.

The transposed form structure of FIR filter is implemented on FPGA. This paper mainly describes the design and simulation of FIR filter which is based on FPGA, Xilinx tools and MATLAB. By using these tools time required to get desired results has become less. FIR filter coefficients design has been performed by using MATLAB. VHDL has been used to enter hardware description. To test the correctness of the design the observed output is compared with the calculated output results from MATLAB implementation that confirms the effectiveness of the design. VHDL codes have been written, synthesized, mapped then successfully configured and prototyped. FIR filter designed fully complies with design requirements.

References

[1] D. E. Borth, I. A. Gerson, J. R. Haug, and C. D. Thompson "A flexible adaptive FIR filter VLSI IC". IEEE Journ. Select. Areas Commun. SAC-6(3):494–503, (Apr 1988).
[2] P. R. Cappello, "VLSI Signal Processing". IEEE Press, editor 1984.
[3] Prabhat Ranjan "Implementation of FIR Filter on FPGA" Department of Electronics and Communication Thapar University, Punjab (July 2008).
[4] J. B. Evans,Y. C. Lim, and B. Liu. "A high speed programmable digital FIR filter". In IEEE Int. Conf. Acoust., Speech, Signal Processing, (Apr 1990)
[5] R. Hartley, P. Corbett, P. Jacob, and S. Karr. , "A high speed FIR filter designed by compiler". In IEEE Cust. IC Conf., pages 20.2.1–20.2.4. May 1989.
[6] Ray Goslin, Dr. San Jose, CA 95124 "A Guide to Using Field Programmable Gate Arrays (FPGAs) for Application-Specific Digital Signal Processing Performance Gregory Digital Signal Processing" Program Manager Xilinx, Inc. 2100 Logic .
[7] Joseph B. Evans "An Efficient FIR Filter Architecture" Telecommunications & Information Sciences Laboratory Department of Electrical & Computer Engineering University of Kansas Lawrence, KS 66045-2228
[8] Mark S. Manalo and Ashkan Ashrafi "Implementing Filters on FPGAs" Department of Electrical and Computer Engineering Real-Time DSP and FPGA Development Lab
[9] R.Boite and H. Leich,comments in (1984.) on "A fast procedure to design equirriple minimum phase Fir filters".'IEEE Trans. Circuits Syst.CAS-31,503-504
[10] R. Rabiner and Ronald W. Schafer "Introduction to Digital Speech Processing" Lawrence Rutgers University and University of California, Santa Barbara, USA, Hewlett-Packard Laboratories, Palo Alto, CA, USA
[11] Spartan-3 Starter Kit Board User Guide UG130 (v1.1) May 13, 2005