

Design and Implementation of an Adaptive 32 order FIR Filter using FPGA

Madhuri Jadhav¹, Tanvi Thakur², Sheetal vatari³

¹ Assistant Professor, ² Lecturer, ³ Lecturer

¹ *Electronics and telecommunication, SLRTCE, Mira road,,jadhavmadhuri412@gmail.com*

² *Electronics, SLRTCE, Mira road,thakurtanvi91@gmail.com*

³ *Electronics and telecommunication, Universal college,Thane,sona.vatari@ymail.com*

Abstract— Adaptive FIR Filter are used in many algorithms in which the response of the Filter has to be changed with respect to the need of the signal. As speech signal is a quasi-periodic signal the period of speech changes with time. For cancelling noise in speech signals the adopted algorithms such as LMS algorithm uses adaptive filter coefficients to change the output response so as to minimize the mean square error. The current IP cores in Xilinx helps implement FIR filter but the filter coefficients cannot be changed and there is a need for an architecture where filter coefficients can be changed. Here adaptive FIR filter has been designed and implemented.

Keywords — *FPGA, Adaptive FIR, LMS, Xilinx.*

I. INTRODUCTION

Speech signals convey added information than verbal. The information conveyed by speech is complicated and it includes time frequency modulation of carrier's information such as formants and pitch intonation. Speech is an immensely information-rich signal exploiting frequency-modulated, amplitude-modulated and time-modulated carriers to convey information about words, speaker identity, accent, expression, style of speech, emotion and the state of health of the speaker.

Finite impulse response (FIR) digital filters are widely used due to their crucial role in various digital signal processing (DSP) applications. FIR filter has been designed and realized on FPGA for filtering the digital signal. This technique can be applied to any FPGAs. Finite impulse response (FIR) filters are digital filters whose response to a unit impulse (unit sample function) is finite in duration. The impulse response is "finite" because there is not at all feedback in the filter [1]. If the input is an impulse (that is, a single "1" sample followed by out after the "1" sample has made its way in the delay line past all the coefficients. FIR (Finite Impulse Response) filters are implemented using a finite number "n" delay taps on a delay line and "n" computation coefficients to compute the algorithm (filter) function [2]. This is in contrast to infinite impulse response (IIR) filters whose response to a unit impulse (unit sample function) is infinite in duration. FIR filters can be implemented using either recursive or non-recursive techniques, but usually non-recursive techniques are used. FIR filters are broadly used in digital signal processing (DSP) systems that are characterized by the extensive sequence of multiplication operations. With the introduction of Field Programming Gate Arrays (FPGA), it is feasible to provide custom hardware for application specific computation design.

The changes in designs in FPGA can be accomplished within few hours and thus result in significant savings in cost and design cycle. FPGAs offer speed comparable to dedicated and fixed hardware systems for parallel algorithm acceleration. Floating point implementation on FPGA is a challenging problem because floating point numbers require more fields than fixed points numbers and availability of physical resources on FPGAs is limited [3]. Floating point helps engineers and scientist carry out complex mathematical calculation easily and more accurately. Accuracy in most of the processors is limited due to the usage of integer format and since the frequency response can be very minutely changed, the need of accuracy arrives. If floating point is used on the processor, the time taken increases as it needs a floating point hardware to carry out its calculations [1] [2]. Sequential FIR filter implementation has been given wide consideration, very minute work has been done to reduce the hardware complexity or power consumption of parallel FIR filters.

In this paper, section I is an overview of Finite impulse response, which spans a wide variety of application areas, section II describes FIR Filters and its Characteristics, section III describes the Types of Linear-Phase Fir Filters, section IV describes the Convolution and section V is the Flow Chart for the VHDL Coding that is being implemented, section VI describes the obtained results and section VII is conclusion.

II. FIR FILTERS AND ITS CHARACTERISTICS

FIR filters are type of digital filters having finite impulse response. It is also called as non-recursive digital filters since they do not have the feedback [4]. Finite Impulse Response (FIR) filters are well-defined by scaled and time-delayed

versions of the filter input signal only, as shown by the difference equation:

$$y[n] = b_0 x[n] + b_1 x[n-1] + b_2 x[n-2] \dots, n=0, 1, 2 \dots \dots \dots (1)$$

Where the input and output $y[n] = 0$ for $n < 0$

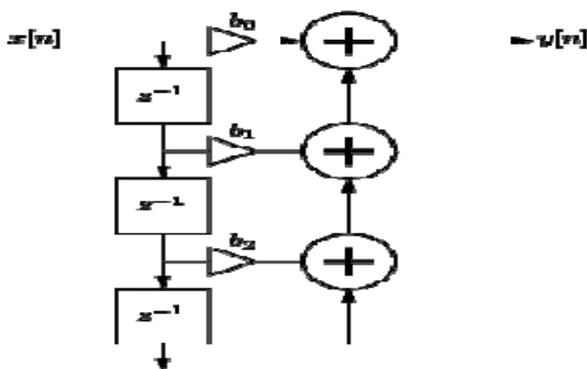


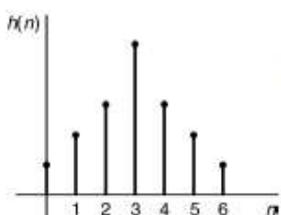
Fig 1: Block diagram representation of FIR filter

The z^{-1} terms represent unit delays in terms of powers of z , the specification of delay is a z -domain characterization since the block diagram itself is a time domain representation [5]. Referring to the block diagram of FIR filter, FIR filters uses feed forward terms only.

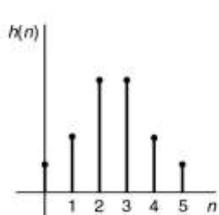
- The impulse response of an FIR filter is only as long as the maximum delayed input term in its difference equation.
- The sum of input terms, scaled by their coefficients, in its difference equation is given by the maximum possible gain of an FIR filter
- The summation of feed forward input terms can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

III. FOUR TYPES OF LINEAR-PHASE FIR FILTERS

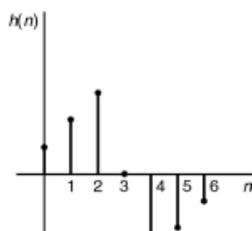
| Type | impulse response | |
|------|------------------|----------------|
| I | symmetric | length is odd |
| II | symmetric | length is even |
| III | anti-symmetric | length is odd |
| IV | anti-symmetric | length is even |



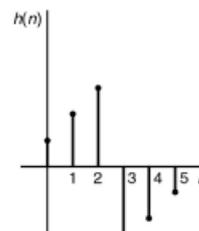
FIR I: odd length, symmetric



FIR II: even length, symmetric



FIR III: odd length, antisymmetric



FIR IV: even length, antisymmetric

Fig 2: Types of Linear-Phase Fir Filters

IV. CONVOLUTION

Convolution is the calculated process that relates the output, $y(t)$ of a linear, time-invariant system to its input, $x(t)$ and impulse response $h(t)$. That is,

$$y(t) = x(t) * h(t) = h(t) * x(t) \quad (1)$$

Here "*" represents the commutative convolution operation. The continuous time relationship is defined by the convolution integral.

$$y(t) = \int_{-\infty}^{\infty} x(\tau) h(t-\tau) d\tau \quad (2)$$

For infinite discrete sequences, $x(n)$, $y(n)$, and $h(n)$, reduces to another familiar expression, the convolution sum.

$$y(n) = \sum_{k=-\infty}^{\infty} h(k) \cdot x(n-k) \quad (3)$$

For two finite discrete sequences of length N_x and N_h , the linear or aperiodic convolution sum takes a marginally different form.

$$y(n) = \sum_{k=0}^n h(k) \cdot x(n-k) \quad (4)$$

where $h(k)$ and $x(n-k)$ are zero outside their suitably defined intervals. For $N_x > N_h$, each summation need only be calculated for the $0 \leq k \leq N_h-1$ terms. The output, $y(n)$, will have length N_x+N_h-1 . Convolution processing is notoriously computationally intensive [2]. Other convolution methods aim to reduce the number of required computations and reduce the amount of input to output latency without altering the expected

output. Some of these techniques include block convolution and hybrid time/frequency processing [5].

V. IMPLEMENTATION

1. Design Flow.

The coefficient values and the input speech signal values is generated in MATLAB. The values are in fixed point format. Figure 3 shows the design flow of the entire process of Configurable FIR filter implementation on FPGA through VHDL coding done in Xilinx ISE design suit 14.5 version.

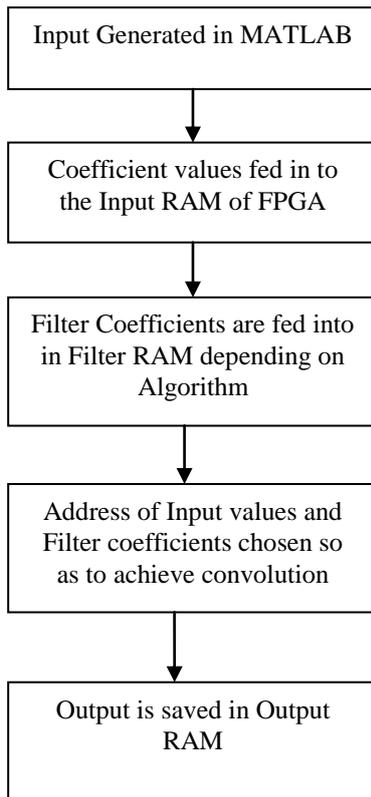


Fig 3: Flow Chart for the VHDL Coding

The Input values are first fed in the Input Data RAM of the FPGA. When the input RAM is filled with 256 values of data from the input, the filter coefficients are filled in the FILTER RAM of the FPGA [6]. The FIR filter is of the order of 32, hence 32 values are fed in the Filter RAM. Once the filter coefficients are adjusted the convolution process is carried out and the output is saved in the OUTPUT RAM [3].

2. FIR FILTER IMPLEMENTATION

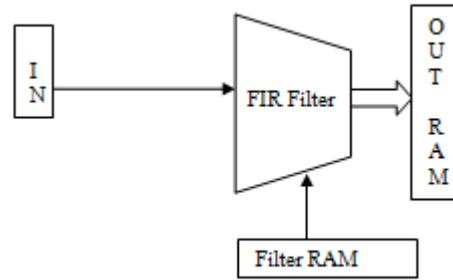


Fig 4: Block Diagram of FIR Filter Implementation

As shown in Fig. is the primary block diagram of the 32 order FIR filter on FPGA. The Filter RAM can be changed with each cycle of Filtering making it idle for implementing Least Mean Square Algorithm. The multiplier makes use of the DSP48 block in the SPARTAN 6 chip for making the process fast [6] [7].

The FIR Filter block consists of MAC units. The input is first addressed from the input RAM. According to the output value to be found the appropriate filter coefficients is first multiplied and saved in a temporary Register and is then added with the next multiplied block.

VI. RESULTS AND DISCUSSION

The FIR filter was implemented on the Xilinx 14.5 version. The VHDL was simulated the result of which is shown in Fig 5.

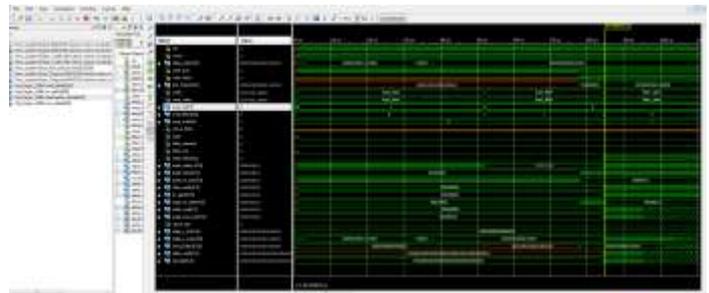


Fig 5: Simulation Results

The VHDL was synthesized, the RTL schematic of which is shown in Fig 6.



Fig 6: RTL schematic

VII. CONCLUSION

LMS adaptive filter has been implemented on Xilinx zync 7 FPGA with floating point data representations using with Xilinx ISE 14.5. This design is useful in implementing adaptive filtering algorithm such as LMS and RLS.

REFERENCES

- [1] Rafael Merredin Alves Falcão, "ADAPTIVE FILTERING ALGORITHMS FOR NOISE CANCELLATION," Faculdade de Engenharia da Universidade do Porto 2012.
- [2] Joseph G. Petrone, "Adaptive Filter Architectures For FPGA Implementation," A Thesis from THE FLORIDA STATE UNIVERSITY COLLEGE OF ENGINEERING 6-29-2004.
- [3] Kartheek.B.V, Manojkumar.S.B, M.B.Anandaraju, "Design and Implementation of Modified Adaptive Filtering Algorithm for Noise Cancellation in Speech Signal on FPGA for Minimum Resource Usage," International Journal of Inventive Engineering and Sciences (IJIES) ISSN: 2319-9598, Volume-1, Issue-7, June 2013. K. Elissa, "Title of paper if known," unpublished.
- [4] Rakhi Thakur , Kavita Khare, "High Speed FPGA Implementation of FIR Filter for DSP Applications," High Speed FPGA Implementation of FIR Filter for DSP Applications. International Journal of Modeling and Optimization, Vol. 3, No. 1, February 2013.
- [5] Shashikala Prakash , Renjith Kumar T.G , Subramani H, "AN FPGA IMPLEMENTATION OF THE LMS ADAPTIVE FILTER FOR ACTIVE VIBRATION CONTROL," International Journal of Research in Engineering and Technology eISSN: 2319-1163 | pISSN: 2321-7308.
- [6] Tian Lan, Jinlin Zhang, "FPGA Implementation of an Adaptive Noise Canceller," 2008 International Symposiums on Information Processing 23-25 May 2008 Print ISBN: 978-0-7695-3151-9.
- [7] Gyanendra Singh , Kiran Savita , Shivkumar Yadav, "DESIGN OF ADAPTIVE NOISE CANCELLER USING LMS ALGORITHM," International Journal of Advanced Technology & Engineering Research (IJATER) www.ijater.com ISSN No: 2250-3536 Volume 3, Issue 3, May 2013.