ISSN (Online) 2321 – 2004
ISSN (Print) 2321 – 5526

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH IN ELECTRICAL, ELECTRONICS, INSTRUMENTATION AND CONTROL ENGINEERING
Vol. 1, Issue 5, August 2013

VLSI Implementation of Pipelined FIR Filter

Aarti Sharma[1], Sanjay Kumar[2]

Department of ECE, Thapar University, Patiala, Punjab, India[1]

Department of ECE, Thapar University, Patiala, Punjab, India[2]

Abstract: This paper proposes to optimize the system speed with minimal cost and hardware by making use of pipelining approach in the designing of FIR filter. The non- pipelined and pipelined FIR filter has been designed using Hardware Description Language (HDL) and a comparative study of both the filter designs using Radix-4 & Radix-8 has been done. The design synthesis and power analysis are carried out using Xilinx ISE 13.1and Synopsis tool, respectively. The concept of pipelining has been incorporated that results in reducing the delay of the FIR filter, thereby enhancing the speed and reducing the power dissipation as compared to the non-pipelined techniques. Simulation validates the results.

Keywords: FIR filter, Booth multiplier, Pipelining, Non-pipelining, Kaiser Window.

I. INTRODUCTION

In signal processing, the filter is used to remove some unwanted component or feature from a signal thereby improving the quality of signal. It alters the amplitude and/ or phase characteristics of a signal in a desired manner with respect to frequency. The primary function of filter are – to confine a signal into a prescribed frequency band, to decompose a signal into two or more sub-bands, to modify the frequency spectrum of a signal and to model the input-output relationship of a system. Filters are extensively used in signal processing and communication system in applications like noise reduction, echo cancellation, image enchancement, speech and waveform synthesis etc.

There are two main kind of filter: analog and digital filter. Analog filter has analog signal at both its input & output and are made up from components such as resistors, capacitors and op amps to produce the required filtering effect. Such filters are fast and simple to realize but are little stable, sensitive to temperature variations and expensive to realize in large amounts. Digital filter on the other hand uses digital processor to perform numerical calculations on sampled values of the signal and eliminate the problems associated with their classical analog counterparts, thus are preferably used in place of analog filter [1]. Broadly, digital filters are classified as: Finite Impulse Response (FIR) and Infinite Impulse Response (IIR) filter. FIR filters have linear phase, stability, fewer finite precision errors, and efficient implementation hence preferred over IIR filter [2]. This paper discusses the design and implementation of a non-pipelined and pipelined FIR filter using both the encoding schemes for multipliers – Radix-4 and Radix-8.

The remainder of the paper proceeds as follows: Section II describes the brief summary of FIR filter theory and section III presents the algorithm used to design the multiplier which is the basic block of FIR filter structure. Section IV briefly explains FIR filter design and Section V describes the technique used to optimize the FIR filters. Synthesis results and simulation results are presented in section VI and section VII respectively. Concluding remarks are given in the final section.

II. FINITE IMPULSE RESPONSE FILTER

Finite impulse response (FIR) filters are a class of digital filters that have a finite impulse response and are among one of the primary types of filter used in DSP and communication system [3]. They do not have any feedback and therefore if excited by impulse response, the ouput will invariably become zero. The input- output relationship of FIR filter is given by (1)

$$y(n) = \sum_{k=0}^{N-1} p(k)\, x(n-k) \qquad (1)$$

where, $p(k)$, k = 0,1,2,3……N-1 are the impulse response coefficients of the filter. N is the filter length that is number of coefficients.

III. BOOTH MULTIPLIER

Multipliers are the basic building block in DSP, microprocessors and other applications. The system's performance is entirely dependent upon the multipliers because they have large area, long latency and consume considerable power hence there is a need to design high speed, low power consumption, regular and less area multipliers. The speed of the multipliers can be increased by reducing the number of partial products. Parallel multipliers are fastest among all multipliers. Booth multipliers are the parallel multipliers that operate on signed operands in two's complement form and have high performance, low power consumption and does not suffer from bad regularity [5]. This paper presents an efficient implementation of high speed parallel multipliers using both the encoding schemes

ISSN (Online) 2321 – 2004
ISSN (Print) 2321 – 5526

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH IN ELECTRICAL, ELECTRONICS, INSTRUMENTATION AND CONTROL ENGINEERING
Vol. 1, Issue 5, August 2013

Radix–4 and Radix-8 which are further used in the designing of FIR filter. Radix-4 and Radix-8 reduces the number of partial products to *n*/2 and *n*/3 respectively where *n* is the length of the multiplier. The number of partial products can be further reduced by using higher radices but the disadvantage is that we need to generate more multiples of multiplicand. Radix-4 booth recoding encodes the multipliers bits into [-2, 2]. Multiplier bits are grouped into the block of three such that each block overlaps the previous block by one bit. The overlapping keeps track off what was happened in the last block as the MSB of the block act as sign bit. For each group of three bits a partial product is generated according to the recoding scheme as shown in Table I. The same procedure is true for radix-8 but the difference is here the multipliers bits are grouped into block of four bits and encodes the multiplier bits into [-4,4] according to the encoding table as shown in Table II. Finally a regular carry select adder has been used to add all the partial products [6].

Table I Radix-4 Recoding Scheme

Multiplier Bits			**Recoding Operation on Multiplicand, X**
Y_{i+1}	Y_i	Y_{i-1}	
0	0	0	0
0	0	1	+1X
0	1	0	+1X
0	1	1	+2X
1	0	0	-2X
1	0	1	-1X
1	1	0	-1X
1	1	1	0

Table II Radix-8 Recoding Scheme

Multiplier Bits				**Recoding Operation on Multiplicand, X**
Y_{i+2}	Y_{i+1}	Y_i	Y_{i-1}	
0	0	0	0	0
0	0	0	1	+1X
0	0	1	0	+1X
0	0	1	1	+2X
0	1	0	0	+2X
0	1	0	1	+3X
0	1	1	0	+3X
0	1	1	1	+4X
1	0	0	0	-4X
1	0	0	1	-3X
1	0	1	0	-3X
1	0	1	1	-2X
1	1	0	0	-2X
1	1	0	1	-1X
1	1	1	0	-1X
1	1	1	1	0

Multiply by zero means the multiplicand is multiplied by "0". Multiply by "1" means the product still remains the same as the multiplicand value. Multiply by "-1" means that the product is the two's complement form of the multiplicand. Multiply by "-2" is to shift left one bit the two's complement of the multiplicand value and multiply by "2" means just shift left the multiplicand by one place. Multiplying the multiplicand by "3" is equivalent to (2X+X) addition of multiplicand and left shifted multiplicand by one digit. Multiply by "-3" means addition of two's complement of multiplicand and shift left one bit the two's compliment of the multiplicand value. Multiply by "4" means shift left the multiplicand by two place. Multiply by "-4" means shift left the two's complement of multiplicand by two places.

IV. FIR FILTER DESIGN

In this paper, an FIR filter has been designed using Kaiser window technique with the help of MATLAB Filter Design and Analysis tool box (FDA) as shown in Fig. 1 and the coefficients are directly imported to the VHDL file. The input, output and coefficients are represented in fixed point notation and are quantized to 16 bits. The Kaiser window has a adjustable parameter α which optimize the main-lobe width and direct control over stop-band attenuation can be achieved thereby sustaining optimality and flexibility. The direct form realization of FIR filter and magnitude response are shown in Fig. 2 and Fig. 3 respectively. The design parameters or specification are as follows

1. Stop-band attenuation = 40 dB
2. Pass-band ripple = 0.01 dB
3. Transition width = 500 Hz
4. Sampling frequency = 10 kHz
5. Ideal cut-off frequency = 1200 Hz

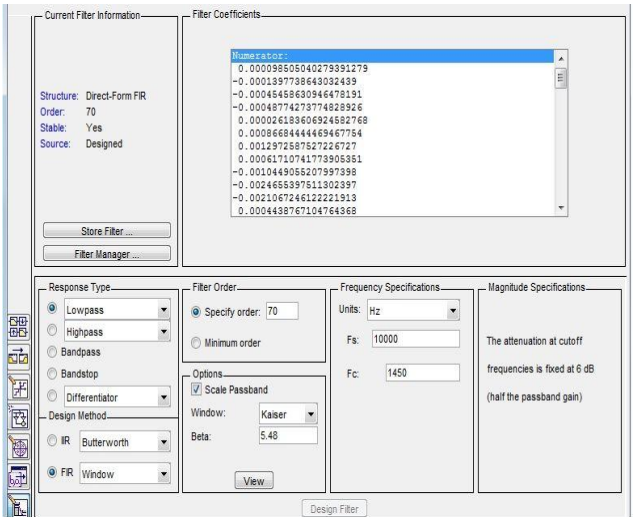
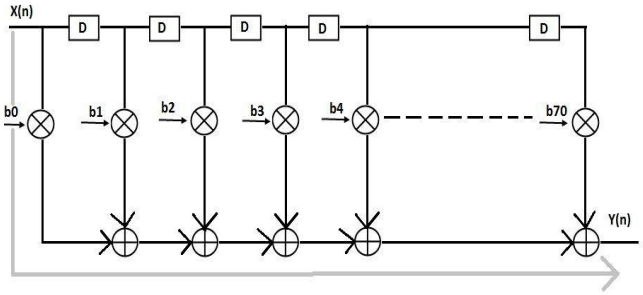


Fig. 1 FIR digital low-pass filter parameters.

ISSN (Online) 2321 – 2004
ISSN (Print) 2321 – 5526

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH IN ELECTRICAL, ELECTRONICS, INSTRUMENTATION AND CONTROL ENGINEERING
Vol. 1, Issue 5, August 2013



$$T_M + 70T_A$$

Fig. 2 Seventy-one tap non-pipelined FIR structure.

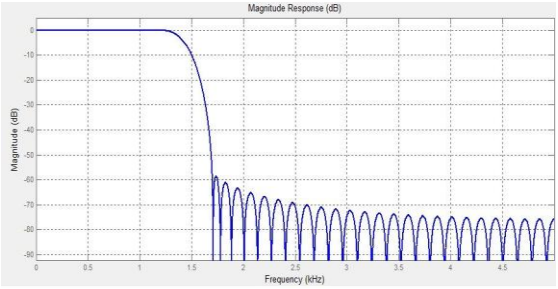
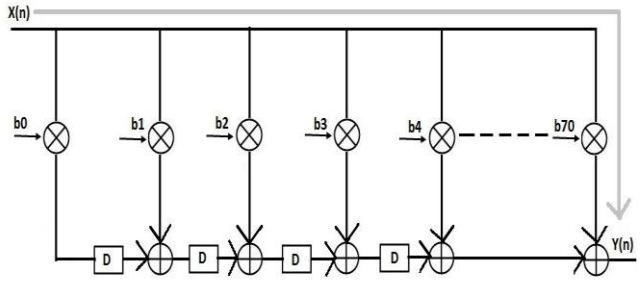


Fig. 3 FIR digital low-pass filter magnitude response

V. PIPELINING

Pipelining is an implementation technique in which multiple instructions are overlapped in execution and results in speed enhancement for the critical path in most of the DSP system, microprocessors etc. It can either increase the clock speed or reduce the power consumption at the same speed in a DSP system. The total execution time for each individual instruction is not altered by pipelining. It does not accelerate instruction execution time but it does accelerate program execution time by increasing the number of instruction finished per cycle [2].

In pipelining any operation along the critical path is broken into smaller and quicker operation with registers between the levels in order to get a smaller critical path or increase in the operating frequency which leads to higher throughput [7]. The pipelined structure of the proposed FIR filter design is shown in Fig. 4.



$$T_M + T_A$$

Fig. 4 Seventy-one tap pipelined FIR structure.

Due to pipelining the critical path has been reduced from $T_M + 70T_A$ to $T_M + T_A$ where, T_M is the time required for multiplication operation and T_A is the time required for addition operation.

VI. SYNTHESIS RESULTS

Table III Synthesis Report of Non-pipelined FIR Filter.

LOGIC ULTILISATION	NON-PIPELINED FIR FILTER	PIPELINED FIR FILTER
Number of Slice Flip Flop	1,186/29,504	2,298/29,504
Four Input LUTs	14,665/29,504	8,698/29,504
Number of Occupied Slices	8,393/14,752	5.004/14,752
Total Number of Four Input Slices	14,917/29504	8,906/29,504
Number Of Bonded IOBs	51/250	51/250
Average Fanout	2.55	2.75
Memory Usuage (Kb)	891980	900608

Table IV Synthesis report of pipelined FIR filter

LOGIC ULTILISATION	NON-PIPELINED FIR FILTER	PIPELINED FIR FILTER
Number of Slice Flip Flop	1,186/29,504	2,299/29,504
Four Input LUTs	19,687/29,504	11,412/29,504
Number of Occupied Slices	11.304/14,752	6,714/14,752
Total Number of Four Input Slices	20,037/29504	11,716/29,504
Number of Bonded IOBs	51/250	51/250
Average Fanout	2.65	2.78
Memory Usuage(Kb)	1076364	1080268

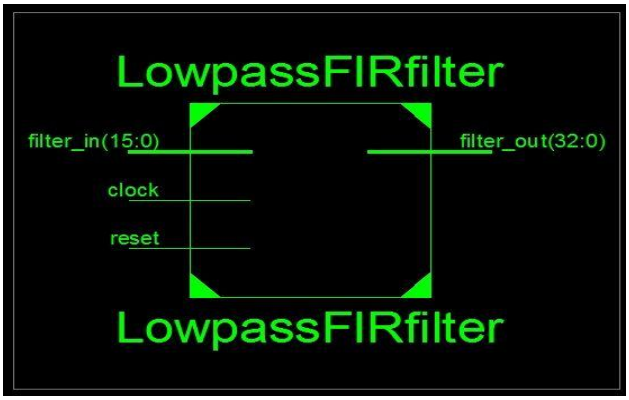


Fig.5 Top Level Circuit Diagram of Low-pass Filter

ISSN (Online) 2321 – 2004
ISSN (Print) 2321 – 5526

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH IN ELECTRICAL, ELECTRONICS, INSTRUMENTATION AND CONTROL ENGINEERING
Vol. 1, Issue 5, August 2013

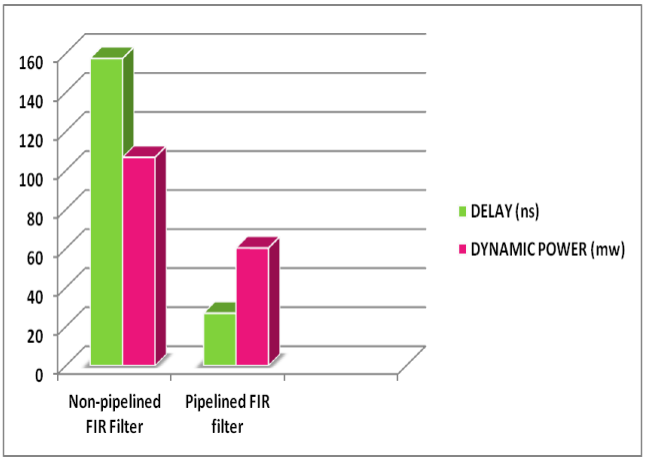


Fig. 6 Delay and Power analysis of FIR filter using Radix-4 multiplier.

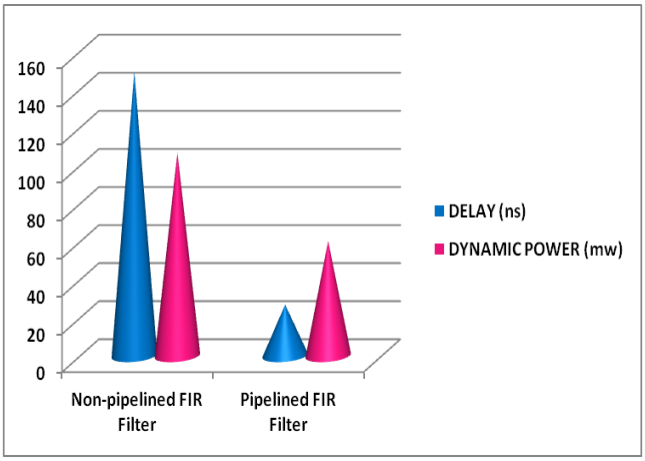


Fig. 7 Delay and Power analysis of FIR filter using Radix-8 multiplier.

VII. Simulation Results

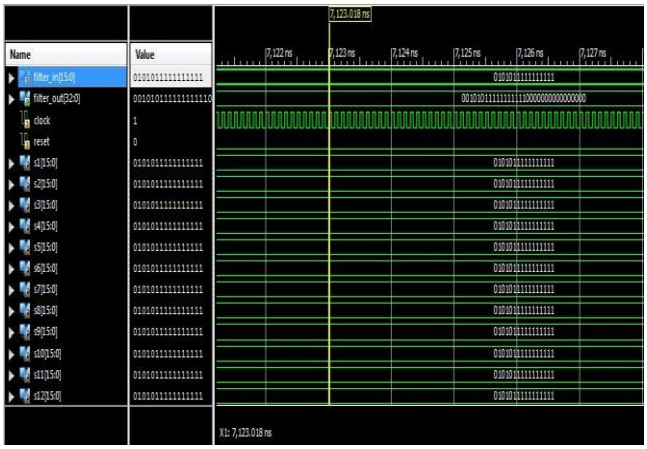


Fig. 8 Simulation of non-pipelined FIR filter using Radix-4 multiplier.

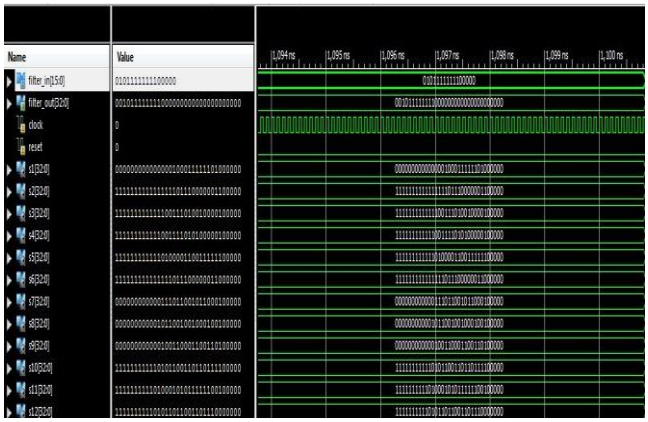


Fig. 9 Simulation of pipelined FIR filter using Radix-4 multiplier.

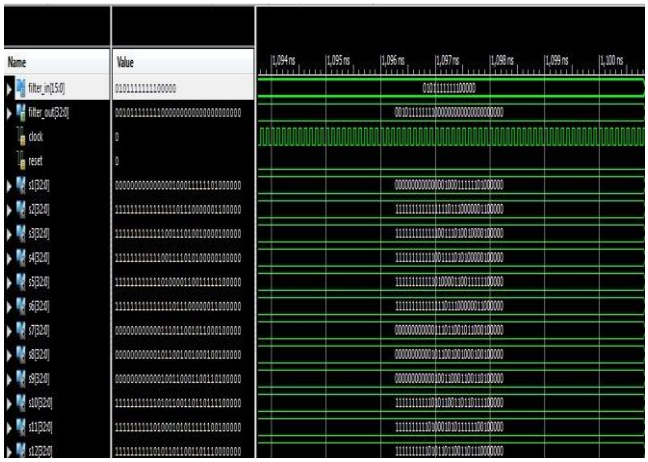


Fig. 10 Simulation of non-pipelined FIR filter using Radix-8

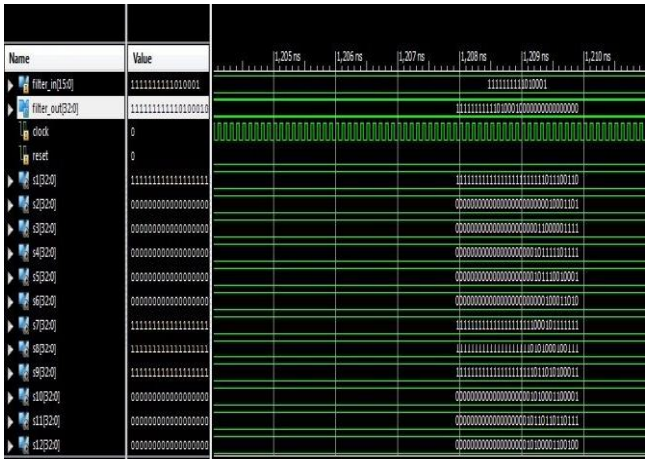


Fig. 11 Simulation of Pipelined FIR Filter Using Radix-8

VIII. CONCLUSIONS

The design of non-pipelined and pipelined FIR filter using both the encoding schemes – Radix-4 and Radix-8 has been accomplished via Hardware Description Language and

ISSN (Online) 2321 – 2004
ISSN (Print) 2321 – 5526

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH IN ELECTRICAL, ELECTRONICS, INSTRUMENTATION AND CONTROL ENGINEERING
Vol. 1, Issue 5, August 2013

synthesized on XILINX ISE Software (Xilinx ISE 13.1 version) for Spartan 3E FGPA (Field Gate Programmable Array) family (XC3S1600E). Synthesis reports are shown in Table III and Table IV. The Fig. 6 demonstrates that the pipelined technique enhances the speed and reduces delay from 157.413 ns to 26.892 ns i.e., 82.9 % in pipelined FIR filter designed using Radix-4 multiplier as compared to non-pipelined technique. On the other hand, delay has been reduced from 149.386 ns to 26.934 ns i.e. 81.97 % in pipelined FIR filter designed using Radix-8 multiplier as compared to non-pipelined technique as shown in Fig. 7. Power analysis shows that the dynamic power has been reduced by the same amount from 106.7415 mW to 60.3062 mW i.e., 43.50 % due to pipelining technique in both the designing of FIR filters as shown in Fig. 6 and Fig. 7. Simulation waveforms are shown in Fig. 8 – Fig. 11. The top RTL level symbol of proposed FIR filter is shown in Fig. 5. As all the structures are implemented using VHDL language, they can be ported to any FPGA family.

ACKNOWLEDGEMENT

The author would like to thank Mr. Sanjay Kumar for his complete guidance and entire VLSI department for their support.

REFERENCES

[1]. N. Parijatha, K.R.A. Hinduja and A.C. Shaker, "FPGA Optimized Low Power and High Speed FIR Filter Structures For DSP Applications", *International Journal of Engineering Research & Technology*, vol. 2, pp. 1-5, Mar. 2013.

[2]. R. Kaur, A. Raman, Member, IACSIT, H. Singh and J. Malhotra, "Design and Implementation of High Speed IIR and FIR Filter using Pipelining", *International Journal of Computer Theory and Engineering*, vol. 3, pp. 292-295, Apr. 2011.

[3]. B. Rashidi, B. Rashidi and M. Pourormazd, "Design and Implementation of Low Power Digital FIR Filter based on low power multipliers and adders on Xilinx FPGA", *International Conference on Electronics Computer Technology*, 2011, pp.18-22.

[4]. E. Ifeachor and B. Jervis, "Finite impulse response (FIR) filter design" in *Digital Signal Processing: A Practical Approach*, 2nd ed., D. Kindersley, Ed. South Asia: Pearson Education, 2002, pp. 342-440.

[5]. V.V. Haibatpure, P.S. Kasliwal and B.P. Patil, "Performance Evaluation of Proposed Vedic Multiplier In Microwind", *International Journal of Communication Engineering Applications*, vol. 03, pp. 498-502, Jul. 2012.

[6]. P. R. Aparna and N. Thomas, "Design and Implementation of a High Performance Multiplier using HDL", *International Conference on Computing, Communication and Application*, 2012, pp. 1-5.