FPGA Implementation of Scalable Microprogrammed FIR Filter Architectures using Wallace Tree and Vedic Multipliers

Abdullah A. AlJuffri†
†National Center for Electronics and Photonics Research
King Abdulaziz City for Science and Technology
Riyadh 11442, Saudi Arabia
†aaljuffri@kacst.edu.sa

Aiman S. Badawi†, Mohammed S. BenSaleh
Abdulfattah M. Obeid, Syed Manzoor Qasim
Communications and Information Tech. Research Institute
King Abdulaziz City for Science and Technology
Riyadh 11442, Saudi Arabia

***Abstract*—Field programmable gate array (FPGA) is widely used for efficient hardware realization of digital signal processing (DSP) circuits and systems. Finite impulse response (FIR) filter is the core of any DSP and communication systems. To improve the performance of FIR filter, an efficient multiplier is required. Wallace tree and Vedic multipliers are used in this paper for the implementation of sequential and parallel microprogrammed FIR filter architectures. The designs are realized using Xilinx Virtex-5 FPGA. FPGA implementation results are presented and analyzed. Based on the implementation results, sequential FIR filter using Wallace tree multiplier/carry skip adder combination proves to be more efficient as compared to other multiplier/adder combinations.**

Keywords—FPGA; FIR Filter; microprogrammed; multiplier

I. INTRODUCTION

Filtering is one of the fundamental steps in many digital signal processing (DSP) applications such as video processing, image processing and wireless communication. Digital filters are normally used to filter out undesirable parts of the signal or to provide spectral shaping such as equalization in communication channels, signal detection or analysis in radar applications. Adders, multipliers and shift registers are the basic building blocks commonly used in the implementation of digital filters. These building blocks are arranged and interconnected in different ways according to the filter architecture. Different architecture of digital filters can be realized to achieve the same transfer function. The architectures possess different attributes in the form of speed, complexity and power dissipation [1].

Finite impulse response (FIR) and infinite impulse response (IIR) are two such filters used in different applications. FIR filters are the important building blocks for digital signal, video and image processing applications. Basically, FIR filter performs a convolution on a window of N data samples. A common implementation of the FIR filter is shown in fig. 1, which is also known as direct form FIR filter. As can be seen from the figure, N-tap or $(N\text{-}1)^{th}$ order FIR filter consist of N shift registers, N multipliers and N-1 adders. The impulse response of the FIR filter can be directly inferred from the tap coefficients (W). The multiplier is the fundamental component which decides the overall performance of the FIR filter [2].

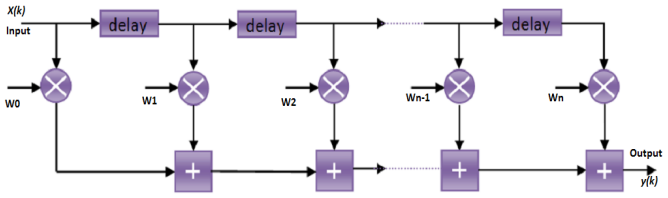

Fig. 1. Direct form FIR filter.

FPGAs have emerged as a platform of choice for faster and efficient realization of compute-intensive applications. It provides hardware speed, software flexibility and price/performance ratio much more favorable than application-specific integrated circuits (ASICs) [3], [4].

The objective of this paper is to extend the work originally proposed in [5-8] by implementing the sequential and parallel microprogrammed FIR filters using Wallace tree and Vedic multipliers in FPGA and evaluating their performance for different number of taps. The multipliers use different adder configurations such as carry skip and Kogge-Stone.

The rest of the paper is organized as follows. Section II presents an overview of microprogrammed FIR filters. Section III and IV describes the Wallace tree and Vedic multiplier designs respectively. Section V presents and analyzes the FPGA implementation results. Finally, concluding remarks are presented in section VI.

II. MICROPROGRAMMED FIR FILTER

The microprogrammed FIR filter consists of a datapath and a microprogram control unit (MCU). The most important advantage of the MCU is its flexibility. It consists of two main parts. The first part is designed for addressing the microinstructions stored in the control memory and the second part is designed to hold and generate microinstructions for the datapath unit [1], [5-7].

A. Sequential Architecture of Microprogrammed FIR Filter

The sequential architecture of N-tap microprogrammed FIR filter is shown in Fig. 2. It basically comprises of a MCU and a datapath unit. The MCU consists of a microprogram counter

ISBN: 978-1-4799-5679-1 ©2015 IEEE

and microprogram memory. The datapath unit comprises of 2*N* data (X) and coefficient (W) registers and *M*-to-*N* decoder (*M* = log^2N), two *N*-input multiplexers for selecting the data and coefficients, a multiplier and an adder, a two input multiplexer to control the flow of data from multiplier or accumulator, one 16-bit accumulator and a 16-bit register to latch the data [8].

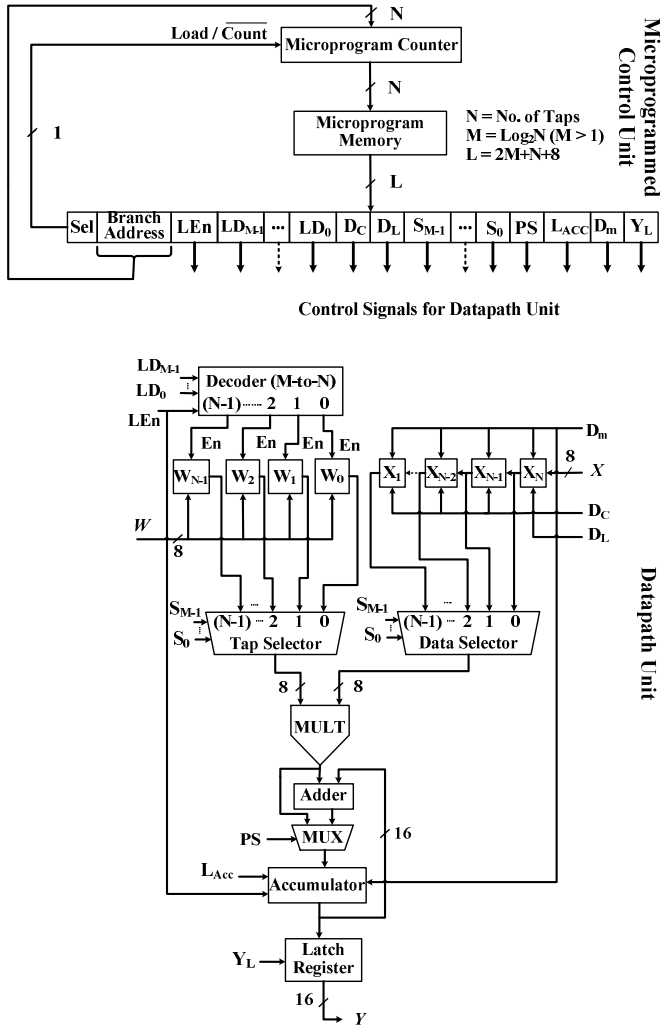

Fig. 2. Architecture of sequential microprogrammed FIR filter[8].

B. Parallel Architecture of Microprogrammed FIR Filter

The parallel architecture utilizes multiple adders and multipliers, based on the size of the FIR filter, in contrast to single adder and multiplier used in the sequential architecture design. Fig. 3 illustrates the parallel architecture of the microprogrammed FIR filter [8]. For example, the datapath microarchitecture of 4-tap parallel FIR filter consists of the following sub-modules:

- Four 8-bit data registers
- One 2-to-4 decoder
- Four 8-bit coefficient registers
- Four multipliers (8×8)
- Three 16-bit adders
- One 16-bit register for latching the output

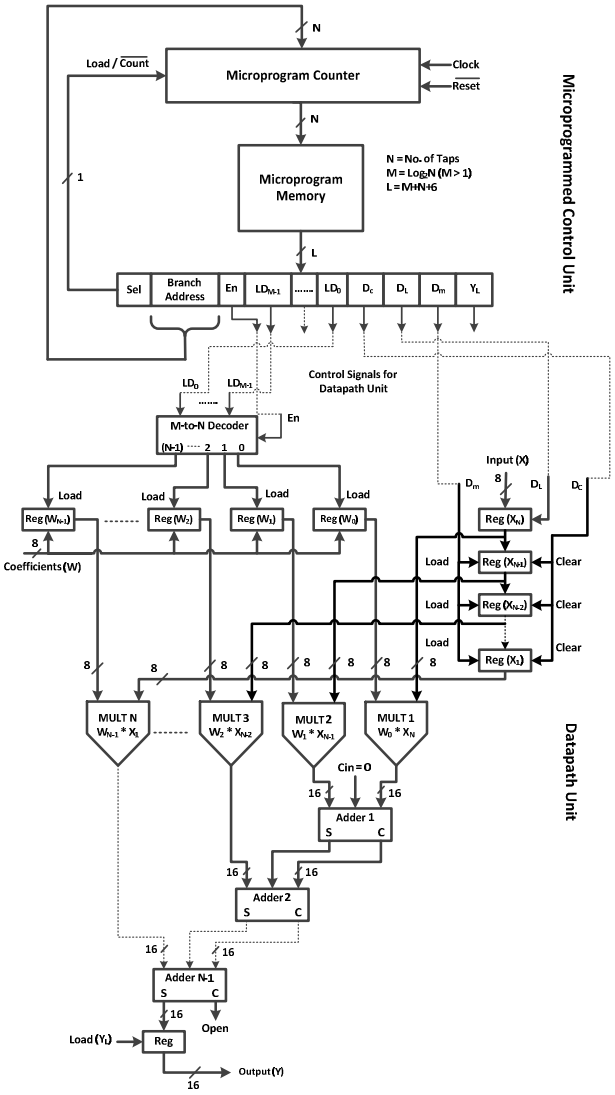

Fig. 3. Architecture of parallel microprogrammed FIR filter[8].

III. WALLACE TREE MULTIPLIER DESIGN

A method for fast multiplication was originally proposed by Wallace [9]. Wallace tree is an efficient hardware implementation of a digital circuit that multiplies two integers. Using this method, a three step process is employed to multiply two integer numbers. The first step is to multiply each bit of one of the arguments, by each bit of the other, yielding n^2 results. Based on the position of the multiplied bits, the wires carry different weights. The second step is to reduce the number of partial products to two by layers of full and half adders. The third step is to group the wires in two and then add them using conventional adder [10]. In this paper, two different architectures of Wallace tree multiplier are presented. First one is designed using only half adder and full adder, while the second one uses a more sophisticated carry skip adder (CSA) [11].

A. Wallace Tree Multiplier using Full and Half Adders

The Wallace tree method reduces the number of adders by minimizing the number of half adders in any multiplier. In 8×8 multiplier, the first partial product is the least significant bit in the output of the multiplier result. After that, moving to the next column of the partial product if there are any adders from the previous product, the full adder is used otherwise a half adder is used and so on. Fig. 4 shows how the algorithm is implemented [10].

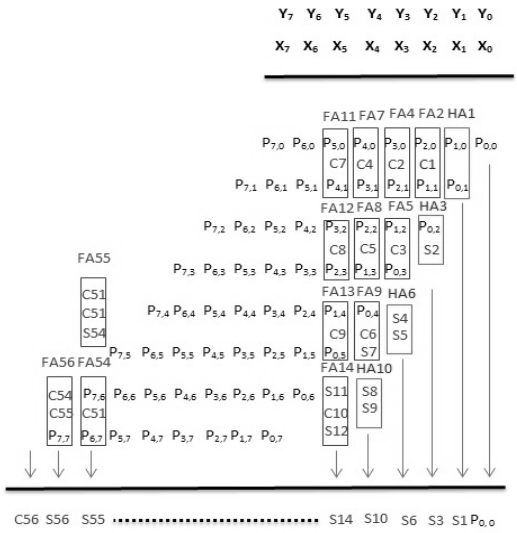

Fig. 4. Wallace Tree partial product addition using half and full adders.

B. Wallace Tree Multiplier using Carry Skip Adder

A carry-skip adder consists of a simple ripple carry-adder (RSA) with a special speed up carry chain called a skip chain [10]. The purpose of using the CSA is to improve the worst case path delay. In this work, we used a 4-bit CSA for implementing the Wallace tree multiplier.

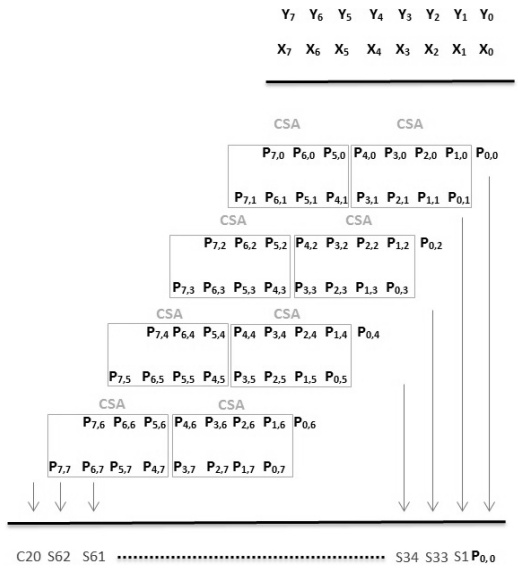

Fig. 5. Wallace tree partial product addition using carry skip adder.

In this design, 4-bit carry skip is used for the addition of partial products. Four bits from one row is being added with the next row as shown in Fig. 5 and the carry output from the first addition is the carry input for the second addition. The main advantage of using CSA is to improve the speed.

IV. Vedic Multiplier Design

Vedic mathematics is an ancient form of mathematics which was developed in India by *Sri Bharati Krishna Tirthaji*, a renowned Sanskrit scholar and mathematician of his times. It is based on sixteen Sutras or algorithms [12]. *Urdhva Tiryakbhyam Sutra* (vertically and crosswise algorithm) is used for efficient digital multiplication. Its calculation is defined by vertical and crosswise product that gives advantage over the normal conventional horizontal multiplication. For binary number, the multiplication operation is reduced to bitwise "AND" operation and the addition operation use full or half adders.

The Vedic mathematics concept is applied to develop modular RTL Verilog code for 2×2 multiplier which can be used as a building block to develop 4×4 multiplier. An 8×8 multiplier can be further designed using the 4×4 multiplier and so on. The 4-bit and 8-bit multipliers used conventional half and full adders for the proposed design. The same Vedic multiplier design is realized using Kogge-Stone adder (KSA). KSA is a parallel prefix form of carry look-ahead adder. It generates the carry signals in $O(log_2N)$ time, and is thus widely considered as the fastest adder design possible [13-14].

V. FPGA Implementation Results

The FIR filter designs are coded in Verilog hardware description language (HDL) and implemented in FPGA using Xilinx Virtex-5 (xc5vlx50t-1ff1136) as the target device. The Wallace tree and Vedic multipliers are used in sequential and parallel architecture of microprogrammed FIR filters. Synplify pro tool is used for the synthesis, translation, mapping and place-and-route process. Different reports are generated by the CAD tool which are summarized in the tables below. The iteration of synthesis has been done for different number of taps of FIR filter (N = 4, 8, 16, 32 and 64). The FPGA resource utilization table includes Virtex-5 slice look-up tables (LUTs), minimum period and maximum clock frequency. Table I and table II summarizes the implementation results of the sequential FIR filter using Wallace tree and Vedic multipliers respectively. Table III and table IV show the implementation results of the parallel FIR filter using Wallace tree and Vedic multipliers respectively.

TABLE I. FPGA Resource Utilization for Sequential FIR Filter Using Wallace Tree Multipliers

Taps	Wallace Tree			Wallace Tree using CSA		
	Slice LUTs	Min. Period (ns)	Max. Freq. (MHz)	Slice LUTs	Min. Period (ns)	Max. Freq. (MHz)
4	147	12.093	82.7	142	10.143	98.6
8	181	12.479	80.1	147	11.448	87.4
16	184	12.166	82.2	180	10.491	85.3
32	212	12.616	79.3	155	10.300	97.1
64	209	12.319	81.2	154	11.044	90.5

ISBN: 978-1-4799-5679-1 ©2015 IEEE

TABLE II. FPGA RESOURCE UTILIZATION FOR SEQUENTIAL FIR FILTER USING VEDIC MULTIPLIERS

Taps	Vedic			Vedic using KSA		
	Slice LUTs	Min. Period (ns)	Max. Freq. (MHz)	Slice LUTs	Min. Period (ns)	Max. Freq. (MHz)
4	195	8.557	116.9	217	9.515	105.1
8	204	9.177	109.0	230	10.660	93.8
16	233	9.180	108.9	246	11.000	90.9
32	219	9.386	106.5	231	11.650	85.8
64	214	9.538	104.8	230	10.639	94.0

TABLE III. FPGA RESOURCE UTILIZATION FOR PARALLEL FIR FILTER USING WALLACE TREE MULTIPLIERS

Taps	Wallace Tree			Wallace Tree using CSA		
	Slice LUTs	Min. Period (ns)	Max. Freq. (MHz)	Slice LUTs	Min. Period (ns)	Max. Freq. (MHz)
4	278	13.876	72.1	284	11.026	90.7
8	687	19.756	50.6	699	17.552	57.0
16	1268	31.662	31.6	1371	29.150	34.3
32	2304	54.787	18.3	1670	52.377	19.1
64	4352	101.39	9.9	3319	98.678	10.1

TABLE IV. FPGA RESOURCE UTILIZATION FOR PARALLEL FIR FILTER USING VEDIC MULTIPLIERS

Taps	Vedic			Vedic using KSA		
	Slice LUTs	Min. Period (ns)	Max. Freq. (MHz)	Slice LUTs	Min. Period (ns)	Max. Freq. (MHz)
4	429	11.557	86.5	488	11.695	85.5
8	1097	17.618	56.8	1217	17.680	56.6
16	2177	28.880	34.6	2490	29.307	34.1
32	3356	52.314	19.1	3909	52.649	19.0
64	6613	98.267	10.2	7553	98.979	10.1

It can be concluded from these tables that the parallel FIR filter architecture using Vedic multiplier/KSA combination consumes maximum FPGA resource. However, the sequential FIR filter using Wallace tree multiplier/CSA combination gave more optimal results. A closer look at the results also reveal that by increasing the number of taps, the FPGA resources used increase gradually in case of sequential architecture as compared to parallel architecture where the utilization of FPGA resources are more rapid. A similar trend is observed when the same designs are realized in ASIC [10].

VI. CONCLUSION

Digital filters are one of the main elements of DSP. FIR filter which mainly comprises of multiply-accumulate structure is the most commonly used digital filter. Since the performance of FIR Filter mostly depends on the multiplier used, an enhanced and improved multiplier will ameliorate the overall system performance. In this paper, we designed and implemented microprogrammed sequential and parallel FIR filter architectures in Xilinx Virtex-5 FPGA using Wallace tree multiplier/conventional adder, Wallace tree multiplier/Carry skip adder, Vedic multiplier/conventional adder and Vedic multiplier/Kogge-Stone adder combinations respectively. The synthesis results generated by Synplify pro tool is used to analyze the performance of both FIR filters. Based on the FPGA implementation results, the sequential FIR filter architecture designed using Wallace tree multiplier/Carry skip adder combination seems to be more efficient as compared to other multiplier/adder combinations studied in this paper.

ACKNOWLEDGMENT

The authors acknowledge the support provided by King Abdulaziz City for Science and Technology (KACST).

REFERENCES

[1] S. M. Qasim, M. S. BenSaleh and A. M. Obeid, "Efficient FPGA implementation of microprogram control unit based FIR filter using Xilinx and Synopsys tools," *Proc. of Synopsys Users Group Conference (SNUG)*, Silicon Valley, USA, pp. 1-14, March 2012.

[2] M. A. Ashour and H. I. Saleh, "An FPGA implementation guide for some different types of serial-parallel multiplier structures," *Microelectroncis J.*, Vol. 31, PP. 161-168, 2000.

[3] S. M. Qasim, A. A. Telba and A. Y. AlMazroo, "FPGA design and implementation of matrix multiplier architectures for image and signal processing applications," *Int. J. Comp. Sci. Network Security*, Vol. 10, No. 2, pp. 168-176, Feb. 2010.

[4] A. M. Obeid, S. M. Qasim, M. S. BenSaleh, Z. Marrakchi, H. Mehrez, H. Ghariani and M. Abid, "Flexible reconfigurable architecture for DSP applications," *Proc. of 27th IEEE Intl. System-on-Chip Conf. (SOCC),* pp. 204-209, Sept. 2014.

[5] S. M. Qasim, M. S. BenSaleh, M. Bahaidarah, H. AlObaisi, T. AlSharif, M. Alzahrani and H. AlOnazi, "Design and FPGA implementation of sequential digital FIR filter using microprogrammed controller," *Proc. of 4th Intl. Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT),* pp. 1002-1005, Oct. 2012.

[6] M. S. BenSaleh, S. M. Qasim, M. Bahaidarah, H. AlObaisi, T. AlSharif, M. Alzahrani and H. AlOnazi, "Field programmable gate array realization of microprogrammed controller based parallel digital FIR filter architecture," *Proc. of World Congress on Engineering and Computer Science (WCECS)*, pp 828–831, Oct. 2012.

[7] S. M. Qasim and M. S. BenSaleh, "Hardware implementation of microprogrammed controller based digital FIR filter," *IAENG Trans. Engg. Tech.*, Vol. 247, pp 29-40, 2014.

[8] M. S. BenSaleh, S. M. Qasim, A. A. AlJuffri and A. M. Obeid, "Scalable design of microprogrammed digital FIR filter for sensor processing subsystem," *IEICE Electronic Express*, Vol. 11, No. 14, pp. 1-7, Aug. 2014.

[9] W. J. Townsend, E. E. Swartzlander and J. A. Abraham, "A comparison of Dadda and Wallace multiplier delays," *Proc. SPIE*, Vol. 5205, 2003.

[10] A. A. Aljuffri. M. M. AlNahdi, A. A. Hemaid, O. A. AlShaalan, M. S. BenSaleh, A. M. Obeid and S. M. Qasim, "ASIC realization and performance evaluation of scalable microprogrammed FIR filters using Wallace tree and Vedic multipliers," *Proc. of 15th IEEE Intl. Conf. on Environmental and Electrical Engineering (EEEIC)*, pp.1-4, June 2015, Accepted.

[11] R. Uma,V. Vijayan, M. Mohanapriya and S. Paul, "Area delay and power comparison of adder topologies," *Int. J. VLSI Design Comm. Syst.,* Vol. 3, No. 1, pp.153-168, Feb. 2012.

[12] H. D. Tiwari, G. Gankhuyag, C. M. Kim and Y. B. Cho, "Multiplier design based on ancient Indian Vedic mathematics," *Proc. of IEEE Intl. SoC Design Conference (ISOCC)*, pp. II-65-II-68, Nov. 2008.

[13] R. Anjana, B. Abishna, M.S. Harshitha, E. Abhishek, V. Ravichandra and M. S. Suma, "Implementation of vedic multiplier using Kogge-stone adder," *Proc. of Int. Conf. on Embedded Systems (ICES)*, pp. 28-31, 2014.

[14] D. Yagain, A. V. Krishna and A. Baliga, "Design of high-speed adders for efficient digital design blocks," *ISRN Electronics*, Vol. 2012, pp. 1-9, 2012.