DOI: 10.15415/jtmge.2010.11008

Journal of Technology Management for Growing Economies

Volume 1 Number 1 **April 2010**

FPGA Based Hardware Co-Simulation of an Area and Power Efficient FIR Filter for Wireless Communication Systems

Rajesh Kumar
Swapna Devi
S.S. Pattnaik
National Institute of Technical Teachers Training and Research, Chandigarh

CHITKARA UNIVERSITY

FPGA Based Hardware Co-Simulation of an Area and Power Efficient FIR Filter for Wireless Communication Systems

Rajesh Kumar
Swapna Devi
S.S. Pattnaik
National Institute of Technical Teachers Training and Research, Chandigarh

Abstract

In this paper FPGA based hardware co-simulation of an area and power efficient FIR filter for wireless communication systems is presented. The implementation is based on distributed arithmetic (DA) which substitutes multiply-and-accumulate operations with look up table (LUT) accesses. Parallel Distributed arithmetic (PDA) look up table approach is used to implement an FIR Filter taking optimal advantage of the look up table structure of FPGA using VHDL. The proposed design is hardware co-simulated using System Generator10.1, synthesized with Xilinx ISE 10.1 software, and implemented on Virtex-4 based xc4vlx25-10ff668 target device. Results show that the proposed design operates at 17.5 MHz throughput and consumes 0.468W power with considerable reduction in required resources to implement the design as compared to Coregen and add/shift based design styles. Due to this reduction in required resources the proposed design can also be implemented on Spartan-3 FPGA device to provide cost effective solution for DSP and wireless communication applications.

Keywords: FPGA, PDA, Simulation, Add/Shift, VHDL.

INTRODUCTION

Today's consumer electronics such as cellular phones and other multi-media and wireless devices often require digital signal processing (DSP) algorithms for several crucial operations (Allred et al., 2004). Due to a growing demand for such complex DSP applications, high performance, low-cost Soc implementations of DSP algorithms are receiving increased attention among researchers and design engineers. There is a constant requirement for efficient use of FPGA resources (Macpherson and Stewart, 2006) where occupying less hardware for a given system that can yield significant cost-related benefits:

(i) Reduced power consumption;
(ii) Area for additional application functionality;
(iii) Potential to use a smaller, cheaper FPGA.

Finite impulse response (FIR) digital filters are common DSP functions and are widely used in multiple applications like telecommunications, wireless/satellite communications, video and audio processing, biomedical signal processing and many others. On one hand, high development costs

Journal of Technology Management for Growing Economies
Vol. 1 No. 1, April 2010
pp.113-122

and time-to-market factors associated with ASICs can be prohibitive for certain applications while, on the other hand, programmable DSP processors can be unable to meet desired performance due to their sequential-execution architecture (Longa and Miri, 2006). In this context, reconfigurable FPGAs offer a very attractive solution that balance high flexibility, time-to-market, cost and performance. Therefore, in this paper, an important DSP function i.e. FIR filter is implemented on Virtex-4 FPGA. The impulse response of an FIR filter may be expressed as:

$$Y = \sum_{k=1}^{K} C_k x_k \tag{1.1}$$

where C_1,C_2…….C_K are fixed coefficients and the x_1, x_2…… x_K are the input data words. A typical digital implementation will require *K* multiply-and-accumulate (MAC) operations, which are expensive to compute in hardware due to logic complexity, area usage, and throughput (White, 1989). Alternatively, the MAC operations may be replaced by a series of look-up-table (LUT) accesses and summations. Such an implementation of the filter is known as distributed arithmetic (DA).

where C1,C2…….CK are fixed coefficients and the x1, x2…… xK are the input data words. A typical digital implementation will require K multiply-and-accumulate (MAC) operations, which are expensive to compute in hardware due to logic complexity, area usage, and throughput (White, 1989). Alternatively, the MAC operations may be replaced by a series of look-up-table (LUT) accesses and summations. Such an implementation of the filter is known as distributed arithmetic (DA).

DISTRIBUTED ARITHMETIC

DISTRIBUTED ARITHMETIC (DA) is an efficient method for computing inner products when one of the input vectors is fixed. It uses look-up tables and accumulators instead of multipliers for computing inner products and has been widely used in many DSP applications such as DFT, DCT, convolution, and digital filters (White, 1989). The example of direct DA inner-product generation is shown in equation 1where ***xk*** is a 2's-complement binary number scaled such that $|x_k| < 1$. We may express each x_k as

$$x_k = -b_{k0} + \sum_{n=1}^{N-1} b_{kn} 2^{-n} \tag{2.1}$$

where the bkn are the bits, 0 or 1, b_{k0} is the sign bit. Now combining equation 1.1 and 2.1 in order to express *y* in terms of the bits of x_k then we get

$$Y = \sum_{k=1}^{K} C_k[-b_k + \sum_{n=1}^{N-1} b_{kn}2^{-n}] \quad (2.2)$$

The above equation 2.2 is the conventional form of expressing the inner product. Interchanging the order of the summations, gives us:

$$Y = \sum_{n=1}^{N-1}[\sum_{k=1}^{K} C_k b_{kn}]2^{-n} + \sum_{k=1}^{K} c_k(-b_{k0}) \quad (2.3)$$

The above equation 2.3 shows a DA computation where the bracketed term is given by

$$\sum_{k=1}^{K} C_k b_{kn} \quad (2.4)$$

Each b_{kn} can have values of 0 and 1 so equation 2.4 can have 2^K possible values. Rather than computing these values on line, we may pre-compute the values and store them in a ROM. The input data can be used to directly address the memory and the result. After *N* such cycles, the memory contains the result, *y*. As an example, let us consider *K* = 4, C_1 = 0.45, C_2 = -0.65, C_3 = 0.15, and C_4 = 0.55. The memory must contain all possible combinations (2^4 = 16 values) and their negatives in order to accommodate the term

$$\sum_{k=1}^{K} C_k b_{kn} \quad (2.5)$$

which occurs at the sign-bit time. As a consequence, 2 x 2K word ROM is needed. Figure 1 shows the simple structure that can be used to compute these equations. The S, signal is the sign-bit timing signal. The term x_k may be written as

$$x_k = \frac{1}{2}[x_k - (-x_k)] \quad (2.6)$$

and in 2's-complement notation the negative of x_k may be written as

$$-x_k = -\bar{b}_{k0} + \sum_{n=1}^{N-1} \bar{b}_{kn}2^{-n} + 2^{-(N-1)} \quad (2.7)$$

where the over score symbol indicates the complement of a bit. By substituting equation 2.1 & 2.7 into equation 2.6, we get

x

$$x_k = \frac{1}{2}[-(b_{k0} - \bar{b}_{k0}) + \sum_{n=1}^{N-1}(b_{kn} - \bar{b}_{kn})2^{-n} - 2^{-(N-1)} \quad (2.8)$$

In order to simplify the notation later, it is convenient to define the new variables as

$$a_{kn} = b_{kn} - \bar{b}_{kn} \qquad \text{for n} \neq 0 \tag{2.9}$$

and

$$a_{k0} = b_{k0} - \bar{b}_{k0} \tag{2.10}$$

where the possible values of the a_{kn}, including n=0, are ± 1. Then equation 2.8 may be written as

$$x_k = \frac{1}{2}[\sum_{n=0}^{N-1} a_{kn}2^{-n} - 2^{-(N-1)} \tag{2.11}$$

By substituting the value of x_k from equation 2.11 into equation 1.1, we obtain

$$Y = \frac{1}{2}\sum_{k-1}^{K} C_k[\sum_{n=0}^{N-1} a_{kn}2^{-n} - 2^{-(N-1)}]$$

$$Y = \sum_{n=0}^{N-1} Q(b_n)2^{-n} + 2^{-(N-1)}Q(0) \tag{2.12}$$

where

$$Q(b_n) = \sum_{k=1}^{K} \frac{C_k}{2a_{kn}} and Q(0) = \sum_{k=1}^{K} \frac{C_k}{2} \tag{2.13}$$

It may be seen that *Q(bn)* has only $2^{(K-1)}$ possible amplitude values with a sign that is given by the instantaneous combination of bits. The computation of ***y*** is obtained by using a $2^{(K-1)}$ word memory, a one-word initial condition register for Q(O), and a single parallel adder subtractor with the necessary control-logic gates.

CIRCUIT DESCRIPTION

The basic LUT-DA scheme on an FPGA would consist of three main components as shown in figure1. These are input registers, 4-input LUT unit and shifter/accumulator unit.

Input Registers: To reduce the consumption of logic elements, RAM resources are used to implement the shift registers (Allred et al., 2004)

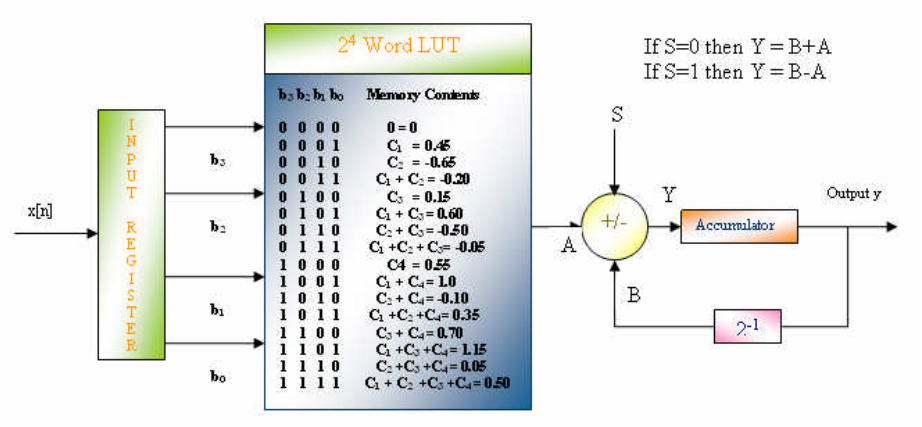

Figure 1: LUT based DA implementation of 4-tap filter
LUT Unit: To implement 4-input and 3-input LUT unit, an LUT table is used, which represent all the possible sum combinations of filter coefficients Figure 1.

Shifter and Accumulator Unit: It consists of an accumulator and a shifter.

PROPOSED WORK

In DA implementation as the filter size *K* increases, the memory requirements grow exponentially as *2K*. This problem is solved in this paper by breaking up the filter into smaller base DA filtering units that require less memory sizes and, less area. If the *K* tap filter is divided into *m* units of *k* tap base units *(K = m×k)*, then the total memory requirement would be $m \times 2^k$ memory words. The total number of clock cycles required for this implementation is B + [log2(m)]; the additional second term is the number of clock cycles required to implement an adder tree to calculate the sums of the units. Thus the decrease in throughput of this implementation is marginal. For instance, in this proposed design $K = 41$, instead of 2^{41} in a full LUT implementation, we have chosen 12 partitions with $k = 4$ for $m = 5$ and $k = 3$ for $m = 7$ which would only require 136 memory words.

In this proposed work a 41-tap low pass filter has been designed. The first step in design flow is to develop an optimized VHDL code using distributed Arithmetic Algorithm and implement it using black box of System generator to develop proposed model of design. Figure 2 shows the developed model of proposed design using various Simulink and System Generator blocks.

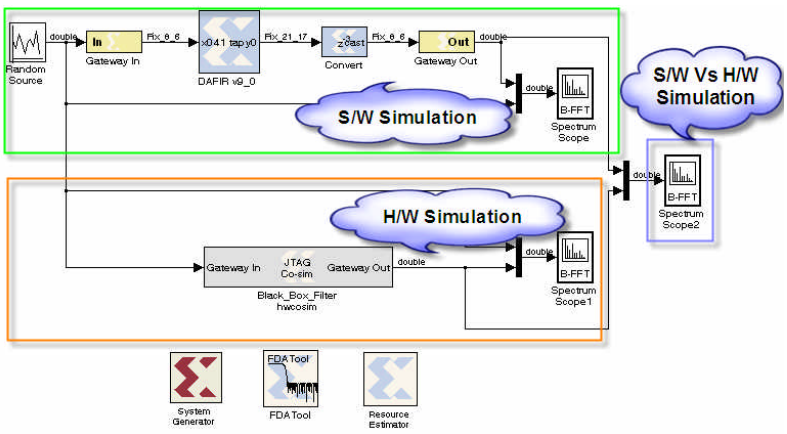

Figure2: Model for Hardware Co-simulation

The part of model enclosed in green boundary shows the software based simulation whose output can be seen in figure 3, part of model enclosed in orange boundary shows hardware based simulation whose output can be seen in figure 4 and spectrum scope in blue boundary shows the comparison between software and hardware based simulation whose output is shown in figure 5.

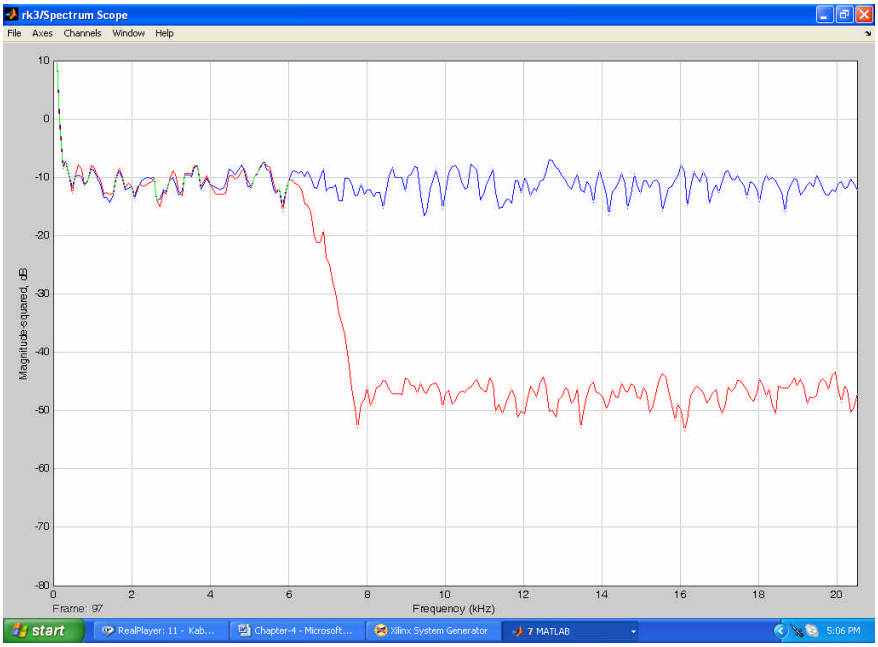

Figure 3: Software Based Simulation

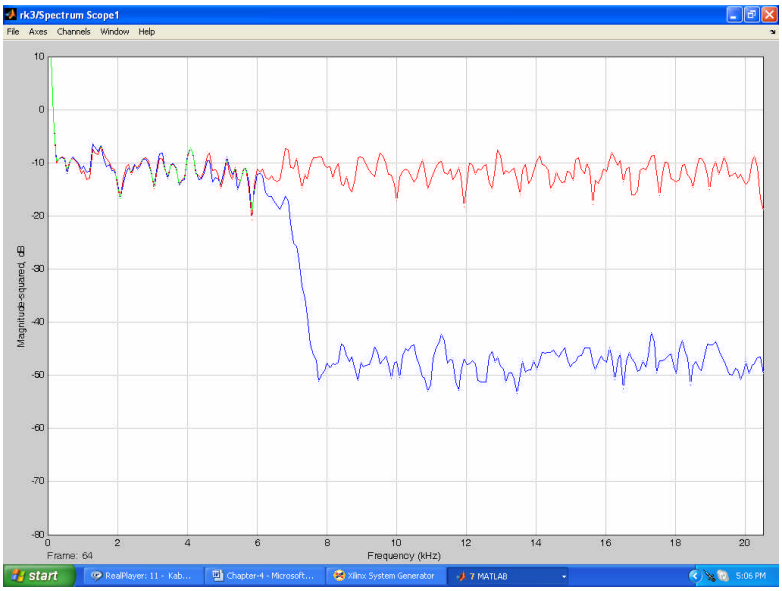

Figure 4: Hardware Based Simulation

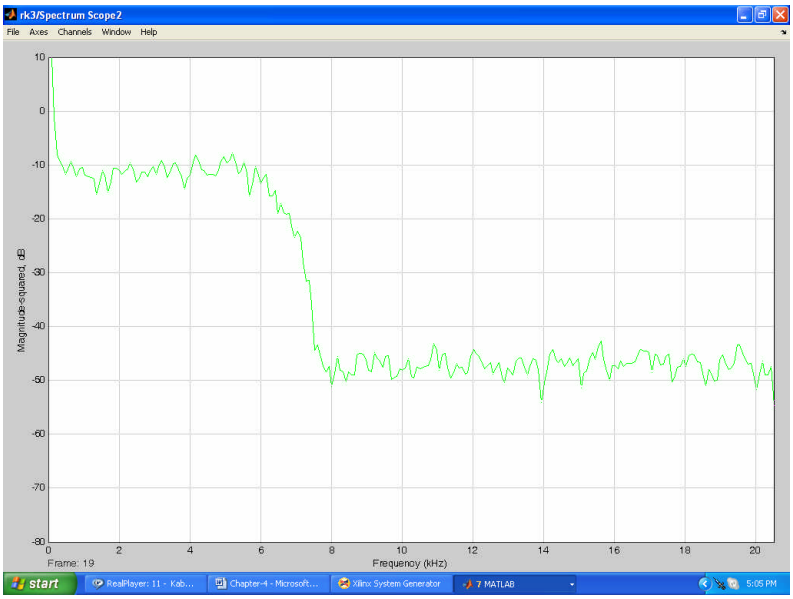

Figure 5: S/W & H/W Based Simulation Comparison

The output wave form with green color in figure 5 means complete matching of software based simulation with hard ware based simulation without errors.

RESULTS

The proposed design is implemented on Virtex-4 based xc4vlx25-10ff668 target FPGA.

Table 1 shows the comparison of proposed PDA design with the published add-shift and coregen based PDA (Mirzaei et al., 2006) implemented on Virtex-4 device. It can be seen from the table that the throughput and performance of the proposed design are 17.50 MHz and 210 Msps respectively which are almost equal to other compared designs.

Table 1: Virtex-4 Based Comparison PDA, Coregen & Add/Shift

Design Style	Slices	LUTs	FFs	Throughput (MHz)	Performance (Msps)
Add-Shift	2154	1719	4161	18.58	223
Coregen PDA	2475	3642	4748	18.50	222
Proposed PDA	1840	3467	2985	17.50	210

Figure 6 shows the comparison of area utilization between add/Shift, PDA (coregen) and proposed PDA (PPDA) for 41 tap filter designs. It can be observed that the PPDA uses considerably less amount of resources on the target device as compared to other compared designs. Due to this reduction in required resources the proposed design can be implemented on Spartan-3 FPGA as shown in table2.

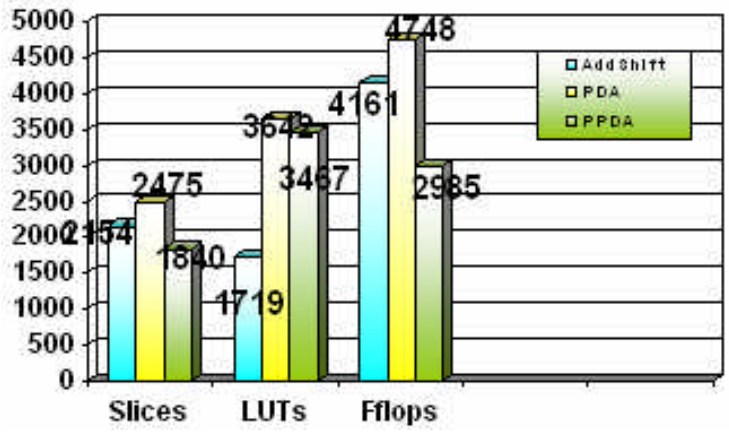

Figure 6: Area Comparison of Add Shift and PDA with PPDA

Table 2: Spartan-3 Based Implementation

Design Style	Slices (R/A)*	LUTs (R/A)*	FFs (R/A)*	Throughput (MHz)	Performance (Msps)
Proposed PDA	1840/1920	3467/3840	2985/3840	8.77	105.22
MAC Parallel	2046/1920	3193/3840	1323/3840	-	-
Add-Shift	2154/1920	1719/3840	4161/3840	-	-
Coregen PDA	2475/1920	3642/3840	4748/3840	-	-

(R/A)*: Resources required / Resources available on target FPGA

Table 3 Shows that the Proposed Design Consumes Total Power of 0.468W at 31.3 Degrees C Junction Temperature.

Table 3: Power Consumption

Name	Value	Used	Total Available	Utilization (%)
Clocks	0.04479 (W)	1	---	---
Logic	0.00000 (W)	3477	21504	16.2
Signals	0.00000 (W)	5239	---	---
IOs	0.00000 (W)	27	450	6.0
DCMs	0.00000 (W)	0	8	0.0
Total Quiescent Power	**0.42292 (W)**			
Total Dynamic Power	**0.04479 (W)**			
Total Power	**0.46772 (W)**			
Junction Temp	**31.3 (degrees C)**			

CONCLUSIONS

In this paper, a Parallel Distributed Arithmetic algorithm for high performance reconfigurable FIR filter is presented to enhance the area & power efficiency. The proposed design is taking optimal advantage of look up table structure of target FPGA. The throughput and performance of the proposed design are 17.5 MHz and 210 Msps respectively with considerable amount of reduction in used resources. Due to this reduction in required resources the proposed design can be implemented on Spartan-3 FPGA device to provide cost effective solution for DSP and wireless communication applications.

REFERENCES

Allred, D.J. Yoo, H., Krishnan, V. Huang, W. and Anderson, D. (2004) 'A Novel High Performance Distributed Arithmetic Adaptive Filter Implementation on an FPGA', *Proceedings IEEE International. Conference on Acoustics, Speech, and Signal Processing (ICASSP'04)*, 5, 161-164.

Banerjee, P., Haldar, M., Zaretsky, D. and Robert Anderson (2004) 'Overview of a compiler for synthesizing Matlab programs onto FPGAs', *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 12: 3, 312-324.

Longa, P. and Miri, A. (2006) 'Area-Efficient FIR Filter Design on FPGAs using Distributed Arithmetic', *IEEE International Symposium on Signal Processings and Information Technology*, p. 248-252.

Macpherson, K.N. and Stewart, R.W. (2006) 'Area efficient FIR filters for high speed FPGA Implementation', *IEE Proceedings-Visual Image Signal Process*, 153: 6, 711-720.

Mirzaei, S., Hosangadi, A. and Kastner, R. (2006) 'FPGA Implementation of High Speed FIR Filters Using Add and Shift Method', *paper presented at the IEEE International Conference on Computer Design, ICCD*.

Ownby, M. and Mahmoud, W.H. (2002) 'A Design methodology for implementing DSP with Xilinx System Generator for Matlab', *IEEE*, p. 404-408.

White, S. A. (1989) 'Applications of distributed arithmetic to digital signal processing: A tutorial review', *IEEE ASSP Magazine*, 6, 4–19.

Yoo, H. and Anderson, D.V. (2005) 'Hardware-Efficient Distributed Arithmetic Architecture for High-Order Digital Filters', *Proceedings IEEE, ICASSP*, pp. V125-128.

Zack, S. and Dhanani, S. (2004) 'DSP Co-Processing in FPGAs Embedding High Performance, Low-Cost DSP Functions', WP212 (v1.0).

Rajesh Kumar is Assistant Professor in ECE Department, National Institute of Technical Teachers Training and Research, Chandigarh, India.

Dr. Swapna Devi is Assistant Professor in ECE Department, National Institute of Technical Teachers Training and Research, Chandigarh, India.

Dr. S.S.Pattnaik is Professor and Head in ETV, National Institute of Technical Teachers Training and Research, Chandigarh, India.

Chitkara University
Saraswati Kendra, Plot 11-12, Dainik Bhaskar Building
Sector 25-D, Chandigarh-160014, India
Email: journal@chitkarauniversity.edu.in
Website: www.chitkara.edu.in/journal/index.php