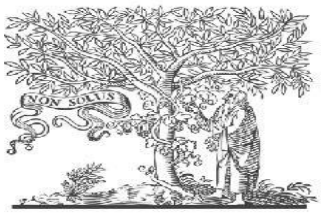
International Journal for Innovative Engineering and Management Research
A Peer Revieved Open Access International Journal
www.ijiemr.org

COPY RIGHT



ELSEVIER
SSRN

IJIEMR Transactions, online available on 26th Nov 2021. Link
:http://www.ijiemr.org/downloads.php?vol=Volume-10&issue=ISSUE-11

DOI: 10.48047/IJIEMR/V10/I11/48

Title **Improving the performance of FPGA-based FIR filters using DSP**

Volume 10, Issue 11, Pages: 294-298

Paper Authors
Chetti Venkateswarlu, Dr.T.Anil Kumar

USE THIS BARCODE TO ACCESS YOUR ONLINE PAPER

To Secure Your Paper As Per UGC Guidelines We Are Providing A Electronic Bar Code

International Journal for Innovative Engineering and Management Research
A Peer Revieved Open Access International Journal
www.ijiemr.org

Improving the performance of FPGA-based FIR filters using DSP

[1]Chetti Venkateswarlu, [2]Dr.T.Anil Kumar
[1]Assistant Professor, Department of Electronics and Communication Engineering,
Malla Reddy Engineering College for Women (Autonomous) Hyderabad
[2] Professor, Department of Electronics and Communication Engineering,
CMR Institute of Technology (Autonomous) Hyderabad
[1]venkatchece@gmail.com [2]tvakumar2000@yahoo.co.in

***Abstract* – Poor performance is typically the outcome of processor-based implementations of DSP algorithms since the architecture is compromised in order to gain programmability simplicity. Because of this, architects have turned to a variety of hardware platforms on which the architecture may be tailored to meet the necessary performance goals. DSP algorithms can now be implemented on FPGAs, thanks to the technology's recent appearance as a viable platform. With today's higher performance demands, classic LUT-based FPGAs are no longer sufficient. Because of this, the underlying FPGA fabric has seen an increase in both its quality and quantity from the manufacturers. Special macro blocks for digital signal processing have been added to current FPGAs. These aid in the quick and efficient completion of various mathematical operations, such as addition, multiplication, and so on. Direct and Transposed FIR filter topologies are discussed in this work utilizing these DSP blocks. For example, Xilinx Spartan-6 FPGAs exhibit an increase in performance relative to older implementations after extensive testing.**

Keywords –FPGA, FIR filters, LTI systems, DSP48A1, LUT

I. INTRODUCTION

The electronic industry's digital signal processing (DSP) sector is one of the most rapidly expanding in the world. Digital signal processing with a high-performance requirement is becoming more used [1-4]. As a result, DSP calculations need a platform capable of continually processing fresh data samples. Any number of sample rates from as little as a few hertz (Hz) up to hundreds of megahertz (MHz) may be used by the source. A DSP algorithm must be split up into many redundant architectures, each with its own set of performance characteristics to account for the wide range of possible sampling rates [7].

Intense mathematical computations are always at the heart of any DSP system. These may range from simple arithmetic operations like addition and multiplication to more complex operations such as convolution. Filtering is one such operation that is frequently used in many DSP applications. Traditionally, filtering algorithms are coded using some high-level language and then implemented using a general or a DSP processor. This, however, results in poor hardware efficiency as the hidden concurrencies in these algorithms remain unutilized [6]. Processor based solutions, therefore, limit the evolution of the architecture [8]. This calls for some hardware-oriented solutions where the notion is to develop the architecture that specifically meets the performance requirements of the application.

The concept of developing the architecture on demand requires some sort of underlying platform. Traditionally, Application Specific Integrated Circuits (ASICs) have been used as the implementation platform of choice [9]. However, the non-recurring engineering (NRE) costs associated with the design cycle of ASICs has limited their use to some specialized domains only. FPGAs provide an alternate platform for developing the architecture on demand. FPGAs have many advantages over ASICs like large-scale integration, lower NRE costs, re-configurable design approach [9-11] etc. thereby providing an attractive platform for rapid system prototyping [12-13]. Recently FPGA fabric has evolved enormously with state of art FPGAs supporting many specialized primitives, IPs and macro blocks in addition to the conventional LUT resources [14-16]. This has prompted designers to develop fully customized systems on these devices, thus, extending the domain of FPGAs beyond prototyping to low and medium volume productions [17-19].

The rest of the paper is organized as follows. Section II briefly discusses the FIR systems. Section III discusses the structures for FIR systems that have been considered for implementation in this work. Section IV discusses the DSP48A1 macro block that is an inherent element in modern FPGAs. Section V analyses the metrics obtained from synthesis and implementation. The paper concludes in section VI and references are listed at the end.

II. FIR SYSTEMS

A particularly important class of systems consists of those that are linear and time invariant. These two properties in combination lead to special kind of systems known as Linear Time Invariant (LTI) systems. LTI systems are characterized in time domain by their response to a unit sample sequence:

$$y[n] = \sum_{k=-\infty}^{\infty} x[k]h[n-k] \quad (1)$$

$$y[n] = \sum_{k=-\infty}^{-1} h[k]\,x[n-k] + \sum_{k=0}^{\infty} h[k]\,x[n-k] \quad (2)$$

$$= \{h[-\infty]x[n+\infty] + \cdots\cdots + h[-2]x[n+2] + h[-1]x[n+1]\} + \{h[0]x[n] + h[1]x[n-1] + \cdots\cdots\} \quad (3)$$

The class of LTI systems can be divided in to two types: Those having a finite duration impulse response (FIR) and those having an infinite duration impulse response (IIR).

International Journal for Innovative Engineering and Management Research
A Peer Revieved Open Access International Journal
www.ijiemr.org

Thus, an FIR system has an impulse response that is zero outside of some finite time interval:

$$y[n] = \sum_{K=0}^{M-1} h[k]\, x[n-k] \quad (4)$$

$$y[n] = y[0]x[n] + h[1]x[n-1] + \cdots\cdots + h[M-1]x[n-M+1] \quad (5)$$

An FIR system simply weights, by the value of the impulse response $h[k]$, $k = 0,1,2 \cdots M-1$, the most recent M signal samples and sums the resulting M products. In effect, the system acts as a window that views only the most recent M input signal samples in forming the output. Thus, we say that an FIR system has a finite memory of length M samples.

III. Structures For FIR Systems

An FIR system is mathematically represented as:

$$y[n] = \sum_{k=0}^{M} b_k\, x[n-k] \quad (6)$$

Taking Z transform

$$Y[z] = \sum_{k=0}^{M} b_k\, Z^{-k} X[z] \quad (7)$$

$$\frac{Y[z]}{X[z]} = \sum_{k=0}^{M} b_k\, Z^{-k} \quad (8)$$

$$H[z] = \sum_{k=0}^{M} b_k\, Z^{-k} \quad (9)$$

Taking inverse Z transform

$$h(n) = \begin{cases} b_k, & for\ 0 \le k < M \\ 0, & otherwise \end{cases} \quad (10)$$

Therefore,

$$y[n] = \sum_{k=0}^{M} b_k\, x[n-k] \quad (11)$$

$$y[n] = \sum_{k=0}^{M} b[k]x\,[n-k] \quad (12)$$

$$y[n] = b[0]x[n] + b[1]x[n-1] + b[2]x[n-2] + \cdots\cdots + b[M-1]x[n-M+1]b[M]x[n-M] \quad (13)$$

A. Direct Form Structure

The Direct form structure consists of a chain of delay elements and thus this structure is also referred to as a tapped delay line structure or a transversal filter structure. The signal sample at each tap along this chain is weighted by the appropriate coefficient (impulse response) and the resulting products are summed to form the output $y[n]$. The realization requires M multiplication and $M-1$ additions per sample. The critical path for such a realization would include a multiplier unit and $M-1$ adder units and is given by:

$$CP_{Direct} = T_{Mult} + (M-1)T_{Ædder} \quad (14)$$

Where T_{Mult} and T_{Adder} is the delay associated with multiplier and an adder respectively.

B. Transposed Form Structure

The transposed form structure is obtained by applying the transposition theorem to the direct form. The realization again requires M multiplication and $M-1$ additions per sample. The critical path for such a realization would include a multiplier unit and an adder unit and is given by:

$$CP_{Transpsoed} = T_{Mult} + T_{Ædder} \quad (15)$$

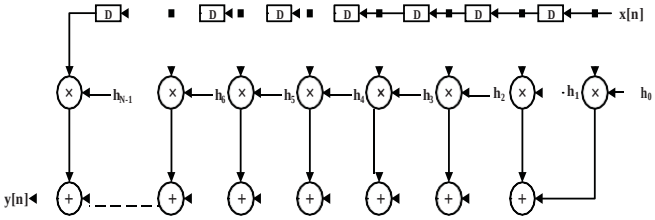


Fig. 1 Direct FIR Filter

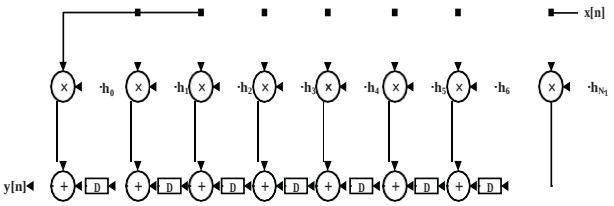


Fig. 2 Transposed FIR Filter

IV. DSP48A1 Block

The DSP48A1 is the advanced version of DSP48A that existed in the earlier Spartan-3A family [20-22]. It supports many arithmetic operations that are implemented with minimal use of general purpose FPGA fabric. This ensures high performance and efficient device utilization. The macro block supports an 18-bit pre-adder; an 18×18 multiplier and a 48-bit sign-extended adder cum subtractor cum accumulator. The entire DSP48A1 block is rigorously pipelined - a feature that enables the block to be clocked at high frequencies. One of the important features of these macro blocks is that they can be easily cascaded without the use of general routing. Architecture highlights of the DSP48A1 slices are:

1) Two-input pre-adder/subtractor.
2) Two-input, flexible 48-bit post-adder/subtractor.
3) Advanced carry management.
4) Performance enhancing pipeline options.
5) Ability to perform multiply-add operation.
6) Separate reset and clock enable for control and data registers, ensuring maximum clock performance and highest possible sample rates with no area cost.

V. Experimentation And Analysis

In this work, FPGAs that have specialized DSP48 blocks have been considered. Although DSP48 blocks were introduced in earlier Xilinx FPGAs, it is only recently that they have gained significance in terms of number, performance optimality and ease of use. Our work, specifically targets the DSP48 block of the Spartan-6 FPGA device. Extensive experimentation is done by carrying out implementations for varying filter orders (keeping operand word-length constant) and varying operand word-lengths (keeping filter order constant). For analysis resources utilized, performance (timing) and power have been considered.

Resources include both general FPGA fabric (LUTs, flip-flops etc.) and special FPGA primitives (carry-chains,

DSP48 blocks etc.). Performance is measured by carrying out the timing analysis of the realized filter structures. Such an analysis may be static or dynamic in nature. The former is mainly concerned with the combinational delays incurred along different paths from input to output. However, for practicality, we have reported the combinational delay along the critical path (largest combinational path) only. Further, the reported numbers are recorded after complete constrained placement and routing of the design. This is because post-synthesis timing analysis is often vulnerable to change, as the logic has not yet been mapped on to the FPGA device. Therefore, post place & route timing analysis has been done which is more accurate. Unlike static timing analysis, dynamic timing analysis, typically checks the functionality of the design in a real time constrained environment. A straightforward approach towards dynamic timing analysis is to design a test bench and apply numerous test vectors. Corresponding to each test vector an output vector is recorded and verified for correctness. If correct, the functionality of the design is confirmed.

Dynamic timing analysis also enables the designer to capture a realistic picture of the switching activity that is occurring within a routed design. The same is used to assess the power dissipation (dynamic) of an implemented design. Generally, a value change dump (VCD) or a simulation activity interchange format (SAIF) file captures the switching activity of a design, which in combination with the design netlist and power constraint file is used to generate a detailed power report. Power dissipation involves both static and dynamic components. While static power is device specific, dynamic power depends on the complexity of the routed design. The amount of mapped logic, clock frequency, density of interconnects and the toggle rate of signals along nodes are some of the factors that will affect the reported power metrics from the synthesizer database.

Xilinx ISE 14.1 has been used to carry out the synthesis, simulation and implementation of different filter structures. For power estimation, Xpower analyzer has been used. The entire analysis has been done on a comparative basis and some frequently used filter designs have been considered. Such an analysis gives a good measure of the achievable performance speed-up using DSP48 blocks. Filter realizations reported in [23-24] have been considered. However, the filters in [23-24] have been implemented using Virtex-5 FPGAs. Since our work focusses on Spartan-6 FPGAs, the filter designs presented in [23-24] are re-implemented using Spartan-6 FPGAs.

A. Resource Analysis

A comparison of different FPGA resources utilized by different FIR filter realizations proposed in this work and those reported in [23] and [24] is presented in table 1. The analysis considers an input operand word-length of 16 bits. It is observed that filter realizations based on DSP48 macro blocks utilize the underlying FPGA device efficiently compared to other realizations. Further, Transposed form realizations show a higher occupied slice count than the Direct form filters. This is due to the extra registers utilized by the transposed form structures. Table 2 provides a comparison of various resources utilized by different filter realizations for an operand word-length of 16 bits. Further analysis plots different utilized resources as a function of operand word-length (for constant tap length of 16) and tap length (for constant operand word-length of 16 bits). The results are shown in figures 3 and 4 respectively.

TABLE 1. OCCUPIED SLICES FOR DIFFERENT FIR FILTERS

Design	Order	Occupied Slices
6:3 Compressor based CSD**[23]**	8	452
Systolic (transposed form)**[23]**	8	120
MAC based (2-parallel/unfold)**[23]**	8	**17**
MATLAB based (Pipelined) **[23]**	16	613
IP based MAC (Systolic) **[23]**	16	511
Canonic sign. Digit (transposed) **[23]**	16	1011
Direct Form **[24]**	8	169
Transposed Form **[24]**	8	140
Direct Form **[24]**	16	302
Transposed Form **[24]**	16	219
Direct Form **[DSP48]**	8	**16**
Transposed Form **[DSP48]**	8	38
Direct Form **[DSP48]**	16	**32**
Transposed Form **[DSP48]**	16	61

TABLE 2. LUTS, REGISTERS, SLICES, DSP48S FOR DIFFERENT PROPOSED FIR FILTERS

Design	Order	LUTs	Registers	Slices	DSP48s
Direct Form	8	**53**	**112**	**16**	15
Transposed Form	8	81	224	37	15

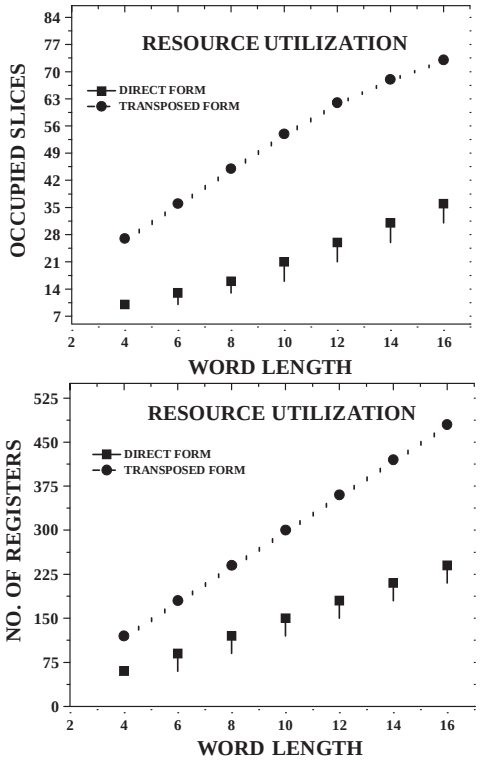


Fig. 3 Slice and registers versus word length for 16 bit operands.

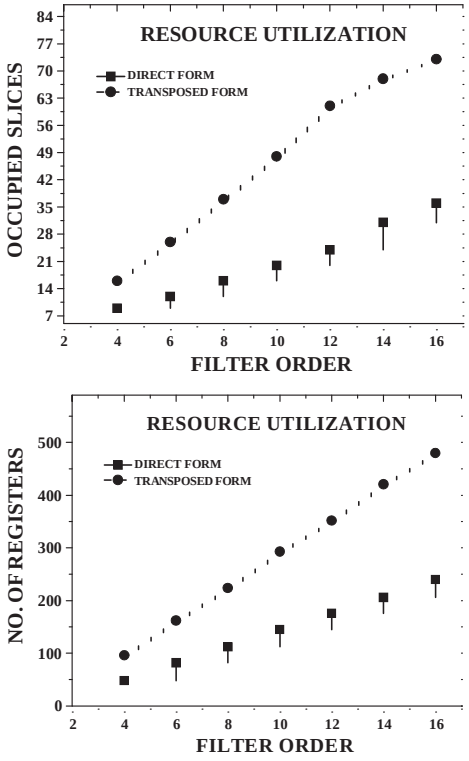


Fig. 4 Resource utilization versus filter order for 16 bit operands.

B. *Performance (Timing Analysis)*

Table 3 provides a comparison of the maximum achievable clock frequency by different FIR filter realizations proposed in this work and those reported in [23] and [24]. The analysis is done for an operand word-length of 16 bits. It is observed that filter realizations based on DSP48 macro blocks have high operating frequencies compared to other realizations. This is because structures based on DSP48 are implemented with a high degree of optimality. There are two reasons for this. First, the adder and multiplier units are implemented using the internal fabric of DSP48 block. Therefore, general FPGA routing, which incurs a lot of delay is avoided. Second, the DSP48 macro block is internally rigorously pipelined as the data transfer within the block is done via the registers. This breaks the critical paths and enables the filter to be operated at high frequencies. A comparison of clock frequency for different filter realizations is provided in table 4. Further analysis plots clock frequency as a function of operand word-length (for constant tap length of 16) and tap length (for constant operand word-length of 16 bits). The results are shown in figures 5 and 6 respectively.

C. *Power Analysis*

The use of DSP48 macro blocks results in a reduction in dynamic power dissipation. This is achieved by reducing the amount of mapped logic and switching activity in the filter structure. Since general LUT fabric is not used there is a reduction in the power dissipated in logic. The switching activity is reduced in two ways. First, the general FPGA routing is avoided as the adder and multiplier units are implemented using DSP48 blocks. This eliminates the switching along the general FPGA interconnects. Second, the blocks are rigorously pipelined by placing registers along different computational nodes. This also helps in relaxing the switching along these high activity nodes. A comparison of power dissipation for different FIR filters is provided in figure 5. Further analysis plots dynamic power dissipation as a function of operand word-length (for constant tap length of 16) and tap length (for constant operand word-length of 16 bits). The results are shown in figures 7 and 8 respectively.

TABLE 3. TIMING ANALYSIS FOR DIFFERENT FIR FILTERS

Design	Order	Throughput (MHz)
6:3 Compressor based CSD**[23]**	8	85.32
Systolic (transposed form)**[23]**	8	185.4
MAC based (2-parallel/unfold)**[23]**	8	212.18
MATLAB based (Pipelined) **[23]**	16	183.32
IP based MAC (Systolic) **[23]**	16	203.64
Canonic sign. Digit (transposed) **[23]**	16	220.31
Direct Form **[24]**	8	217.14
Transposed Form **[24]**	8	246.22
Direct Form **[24]**	16	190.01
Transposed Form **[24]**	16	233.12
Direct Form **[DSP48]**	8	224.4
Transposed Form **[DSP48]**	8	**271**
Direct Form **[DSP48]**	16	191.54
Transposed Form **[DSP48]**	16	**245.71**

TABLE 4. TIMING ANALYSIS FOR DIFFERENT PROPOSED FIR FILTERS

Design	Order	Throughput (MHz)
Direct Form	8	230
Transposed Form	8	278

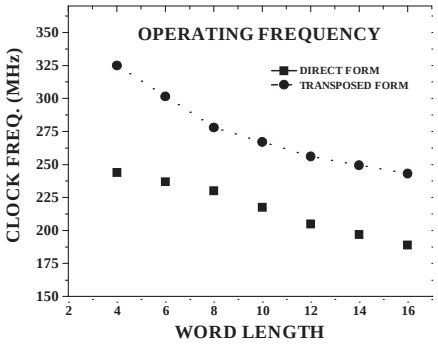


Fig. 5 Operating Frequency versus word length for 16 tap filters

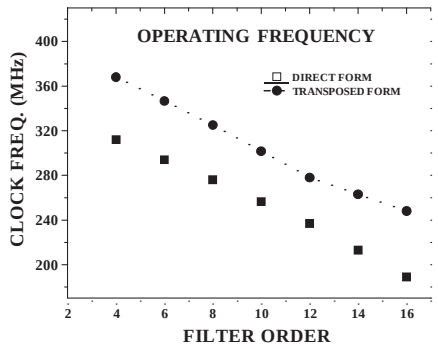


Fig. 6 Operating Frequency versus filter order for 16 bit operands.

TABLE 5. POWER ANALYSIS FOR DIFFERENT PROPOSED FIR FILTERS

Design	Order	Dynamic Power (mW)
Direct Form	8	3.01
Transposed Form	8	4.19

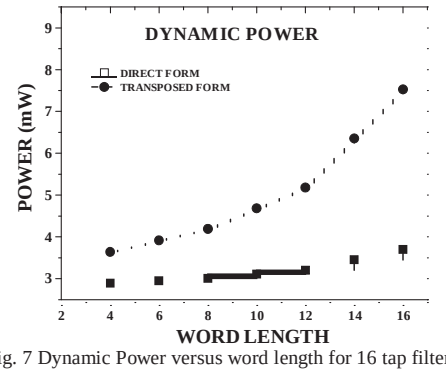


Fig. 7 Dynamic Power versus word length for 16 tap filters

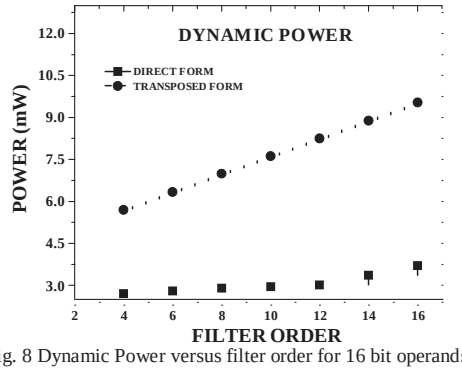


Fig. 8 Dynamic Power versus filter order for 16 bit operands

VI. CONCLUSION

This work implemented FIR filters using adder and multiplier units based on DSP48 macro blocks. The analysis done in this work showed that involving DSP48 macro blocks in the synthesis process can directly impact the resources, performance and power requirements of the design. Different filter realizations were considered, and it was shown that the realizations based on DSP48 will always outperform the other traditional approaches. However, a major limitation in using DSP48 macro blocks is their limited number which kind of hinders their usage in DSP applications. Thus, the complexity of the intended design and the capacity of the target FPGA device is an important consideration while using these macroblocks.

.

REFERENCES

[1] P. Lapsley. J. Bier, A. Shoham and E. A. Lee, DSP Processor Fundamentals: Architecture and Features, Elsevier, 1993.

[2] N. Singh, S. Ayub and J. P. Saini, "Design of Digital IIR Filter for Noise Reduction in ECG Signal," in Proceedings of the 5th International Conference on Computational Intelligence and Communication Network, 2013.

[3] A. V. Oppenhiem and R. W. Schafer, Discrete-Time Signal Processing, Prentice Hall, 1989

[4] S. Haykin, Adaptive Filter Theory, Prentice Hall, 1991.

[5] K. K. Parhi, VLSI Digital Signal Processing Systems: Design and Implementation, Wiley, 1999.

[6] R. Woods, J. McAllister, G. Lightbody and Y. Yi, FPGA-based Implementation of Signal Processing Systems, Wiley, 2008.

[7] B. Khurshid and R. Naaz, "Cost Effective Implementation of Fixed-Point Adders for LUT based FPGAs using Technology Dependent Optimizations," Electronics Journal, Vol. 19 No. 1, June 2015.

[8] G. A. Constantinides, P. Y. K Cheung and W. Luk, Synthesis and Implementation of DSP Algorithms, Kluwer Academic Publishers, 2004.

[9] T. J. Todman, G.A. Constantinides, S. J. E. Wilton, O. Mencer, W. Luk and P. Y. K. Cheung, "Reconfigurable computing: architectures and design methods," in IEE proceedings Computers and Digital Techniques, Vol. 152, No. 2, March 2005.

[10] Z. Guo, W. Najjar, F. Vahid and K. Vissers, "A Quantitative Analysis of the Spped-up factors of FPGAs over Processors," in Proceedings of 12th International Symposium on FPGAs, pp. 162-170, ACM Press, February 2004.

[11] R. Tessier and W. Burleson, "Reconfigurable Computing for DSP: A Survey," Journal of VLSI Signal Processing, Vol. 28, pp. 7-27, Kluwer Academic Publisher, 2001.

[12] S. D. Brown, J. Rose R. J. Francis and Z. G. Vranesic, Field Programmable Gate Arrays, Kluwer Academic Publisher, 2001.

[13] R. Naseer, M. Balakrishnan and A. Kumar, "Direct Mapping of RTL Structures onto LUT-based FPGAs," IEEE Transactions on Computer Aided Design on Integrated Circuits and Systems, Vol. 17, No. 7, July 1998.

[14] G. C. Cardarilli, S. Pontarelli, M. Re and A. Salsano, "On the use of Signed Digit Arithmetic for the new 6-input LUT based FPGAs," in Proceedings of the 15th IEEE International Conference on Electronics, Circuits and Systems, September 2008.

[15] G. Zhou, L. Li and H. Michalik, "Area Optimization of Bit-Parallel Finite Field Multipliers with Fast Carry Logic on FPGAs," in Proceedings of the International Conference on Field Programmable Logic and Applications, September 2008.

[16] S. Gao, D. A. Khalili and N. Chabbini, "Optimized Realization of Large-Size Two's Complement Multipliers on FPGAs," IEEE North-East Workshop on Circuits and Systems, August 2007.

[17] K. Compton and S. Hauck, "Reconfigurable Computing: A Survey of Systems and Software," ACM Computing Surveys, Vol. 34, No. 2, pp. 171-210, June 2002.

[18] S. Hauck and A. Dehon, "Reconfigurable Computing: The Theory and Practice of FPGA based Computation," Morgan Kaufmann Publisher, November 2007.

[19] Altera Corporation, "Stratix III Device Handbook," V.1, November 2006.

[20] Xlinx DSP Design Considerations, Xtreme DSP for Virtex-4 FPGAs, UG073, (v 2.2), July 2006.

[21] Virtex-5 Family Overview LX, LXT, and SXT Platforms, Xilinx, Inc., San Jose, CA, 2010.

[22] L. Deng, K. Sobti, Y. Zhang and C. Chakarbati, "Accurate Area, Time and Power models for FPGA based Implementations," Journal of Signal Processing Systems, Springer, 2011.

[23] H. M. Kamboh, S.A. Khan, "An Algorithmic Transformation for FPGA Implementation of High Throughput Filters," in Proceedings of the 7th International Conference on Emerging Technologies, (2011).

[24] B. Khurshid and R. Naaz, "Technology-Dependent Optimization of FIR Filters based on Carry-Save Multiplier and 4:2 Compressor unit," Electronics Journal, Vol. 20, No. 2, December 2016.