Received: June 6, 2018

International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

Low Cost FPGA Implementation of RFIR Filter Based on a Radix-4 Algorithm

Chetti Venkateswarlu[1]* **Tipparti Anil Kumar[2]**

[1]*Electronics and Communication Engineering, Narsimha Reddy Engineering College, India*
[2]*Electronics and Communication Engineering, CMR Institute of Technology, India*
* Corresponding author's Email: venkatchece@gmail.com

Abstract: The Reconfigurable Finite Impulse Response (RFIR) filter design is a significant operation in Digital Signal Processing (DSP). The RFIR designs often implemented to evaluate the system performance and hardware utilization. The traditional RFIR filter design depends on more sub module such as subtractions, adders, and shifters, which occupies more area and increase the system complexity. To overcome this problem, a Low Cost -Radix4- RFIR (LC-R4-RFIR) filter design is introduced. This research work designed an efficient RFIR filter with the help of Radix 4 approach, which reduced the filter area and hardware utilization. The RFIR filter was designed by using R4 approach for multiplication operation. Multiplication is the one of the main process of adding a number of Partial Products (PPs) Hence, integer multiplication is implemented in serial parallel mode by employing an accumulator to add these PPs. Using R4, the multiplication operation performed which mitigated the area and hardware utilization of the RFIR design. In Field Programmable Gate Array (FPGA) implementation, the number of Look Up Table (LUT), Slice, flip-flop, area, and frequency calculated for different Virtex devices such as Virtex-6, Virtex-6 Low Power (LP) and Virtex-7. This FPGA experimental results showed that the LC-R4-RFIR filter design performed better compared to conventional FIR filter designs.

Keywords: Field programmable gate array, Reconfigurable finite impulse response, Low power, Radix-4 algorithm.

1. Introduction

The FIR digital filter is the basic component of DSP systems. Generally, the FIR filters employed in mobile communication devices and multi-media applications such as matched filtering, video conventional functions, signal pre-conditioning and channelization [1]. The FIR filters provide several advantages like computational efficiency in multi-rate applications, Attainable Linear-Phase Response (ALPR) and desirable numerical property employed for finite precision, and fractional arithmetic [2, 3]. In the present days, the RFIR filters are easily reconfigurable based on input tabs, are used in digital communication systems. The RFIR filter coefficients change dynamically when runtime plays a vital PART in the Software Defined Radio (SDR), digital up or down converters and so on [4, 5]. Compared to the existing non-RFIR filter designs with reconfigurable and without reconfigurable, that RFIR filters consumeless power [6]. In recent years, the different implementation techniques and system architectures have been proposed to increase the performance of the RFIR filter in terms of reducing system complexity and high-system performance [7].

The RFIR filter design is implemented based on Statistics Center Reconfigurable (SCR) technique. That filter architecture design achieved a low-area and power consumption. But drawback of this technique is not discussing the dynamically reconfigurable mechanism [8]. The pipelined modified booth multiplier technique employed for RFIR filter architecture, which is the order of the filter to improve low-power consumption than existing architectures, but this strategy is not possible for the more-power applications [9, 10]. A low-power 8-bit RFIR with minimum power consumption that improved system efficiency, but the drawback of this technique only matched for 8-bit data [11]. In existing work, multiplier and Shift and add method has been used to perform the multiplication operation. But, these methods occupy more area more

computation time. To overcome above mentioned problem, the LC-R4-RFIR filter design is implemented in this paper, here the R4 employed for arithmetic operations because this method occupy less area in the RFIR filter. Generally, the complexity of the RFIR filter design is dominated by multiplier approach. The reduction of area and hardware utilization can be achieved by reducing PPs of the multiplier. In FPGA implementation, the number of LUTs, slice and flip-flop reduced in LC-R4-RFIR for various types of devices like Virtex-6, Virtex-6 LP, and Virtex-7 compared to conventional techniques.

This paper is recognized as follows. In section 2, described some previous related work. In Section 3, shows LC-R4-RFIR architecture design. In Section 4, mentioned experimental setup and results and discussion. The conclusion is made in Section 5.

2. Related work

B.K. Mohanty, P.K. Meher, S.K. Singhal, and M.N.S. Swamy [12] implemented the VLSI architecture employed for RFIR filter based on Distributed Arithmetic (DA) approach. In this work, an analyzed the register complexity of direct-form and transform-form structure of the filter. The direct-form structure involves fewer numbers of registers then the transpose-form structure. The advantage of this technique is, the less complexity of the large filter lengths. But, this proposed method was increased the hardware compexity.

Sriram. N, and J. Selvakumar [13] have illustrated the Pipeline Modify Booth Multiplier (PMBM) method used for implementing less power RFIR filter structure. The pipeline method was significantly utilized to increase the performance of Digital Circuits. In this paper, the delay value was high, because of automatically reduced the system speed and throughput value.

S. Ramanathan, G. Anand, P. Reddy, and S.A. Sridevi [14] proposed a low power adaptive FIR filter design based on DA method with low-area and power consumption. The Least Mean Square (LMS) process was used to update the weight and reduce the Mean Square Error (MSE) between the current filter outcome and the desired response. The pipelined DA table reduced switching activity and power consumption. This research work used for carry save accumulator for FIR filter design, which occupy more area in the FIR filter architecture.

M. Pristach, V. Dvorak, and L. Fujcik. [15] proposed the FIR filter architecture with the support of block memories. The architecture has Random Access Memory (RAM) to store the data, and one Multiply-Accumulate (MAC) unit for the multiple and the accumulation process purpose. That design performed one by one computation to reach the fewer requirement of hardware. The proposed technique achieved high operating frequency, and low-power consumption. The RAM was employed for the data storage purpose in the research method, if the data increase its not suitable for FIR filter design due to space of RAM is limited.

S. Bhattacharjee, S. Sil, and A. Chakrabarti [16] proposed low-power FIR filter design implementation for DSP applications based on FPGA with the support of Xilinx 6V1X130T1FF1156. In this paper, many forms of the structure were observed and analyzed and they found out that FIR structure took a number of registers and it reduced power consumption. But this technique is only suitable for high-speed DSP application.

All these related works contain several problems like more area, power, high critical path, and FPGA utilization. To conquer this problem, the LC-R4-RFIR method improves the FPGA implementation results like LUT, slice, and flip-flop.

3. LC-R4-RFIR Methodology

In an existing FIR filter architecture, the (DA) structure contains N-number of bit shift register, LUT, scalable accumulator which contains subtraction, adder unit, and registers. When DA procedure is directly applied to the RFIR filter, the complex multiplication accumulation operation converter into the adding and shifting operation. A designing of the FIR filter design requires filter coefficients, multipliers and adder, which increases the size of the filter. Hence, the R4 multiplication used for LC-RFIR filter design, which reduce area size and increase system speed.

3.1 The Radix-4 based RFIR filter design

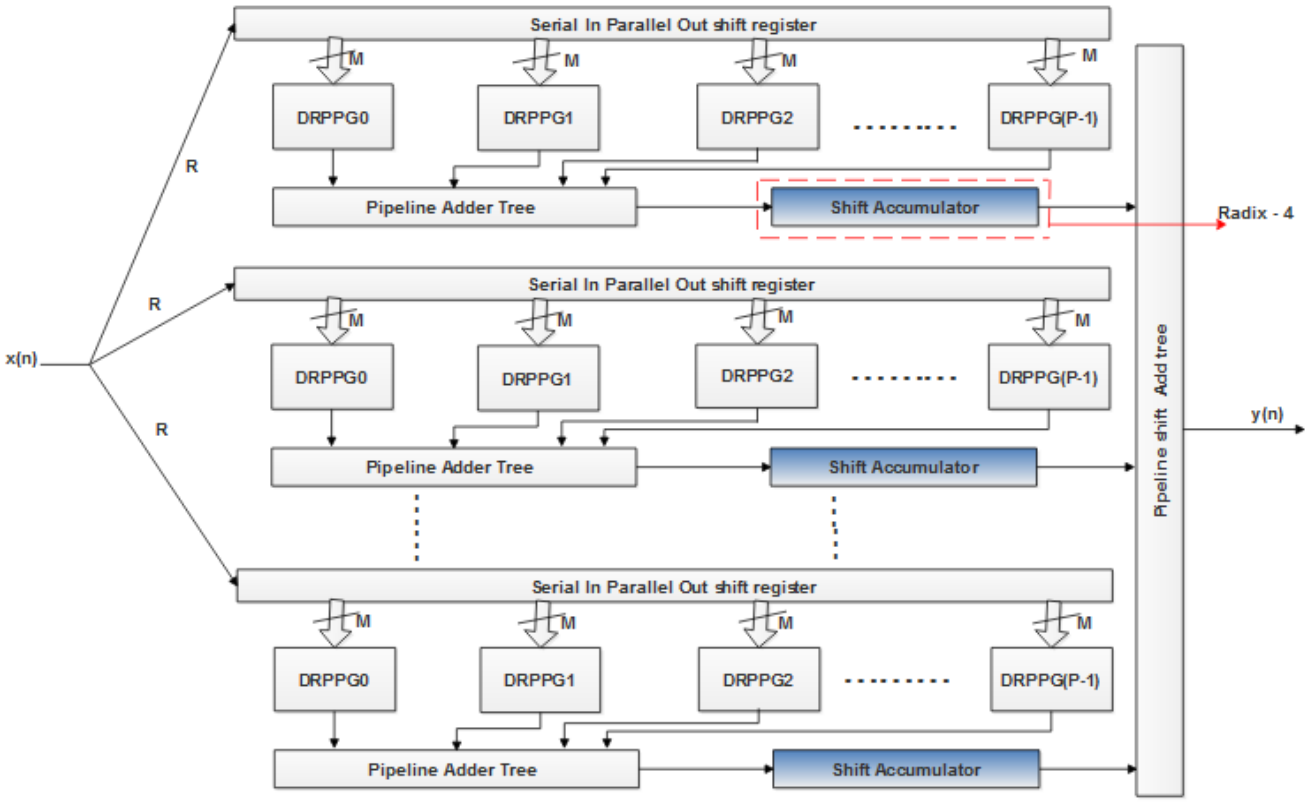

Figure. 1 Architecture design of the LC- R4-RFIR filter design

The fig. 1 shows the architecture of the LC-R4-RFIR filter design, here M represents parameters. A number of registers required to implement LUT based RFIR filters. But, the registers have limited resources in FPGA implementation. Each LUT consists of only 2-bits of the registers in the FPGA devices. A numbers of partial-inner-product S_l, p cannot be recovered from the Distributed RAM (DRAM) simultaneously as only one LUT value can be read from the DRAM per cycle. Furthermore, if L is the bit width of the input, the duration of sample period of the design is L times the operation clock period, it may not be suitable for the application that require more throughput. Employing a DRAM to improve LUT for every bit slice will lead to resource consumption. Hence, this LC-R4-RFIR method decomposes the partial inner-product generator into Q -parallel sections, each section has R time multiplexed operations related to R bit slices, where $L$is a composite number given by$L = RQ(R$and Q two positive integer). Index $l$in Eq. (1) is mapped with $r + qp$ used for $r = 0,1,2,3,\ldots\ldots Q - 1$ to modify in Eq. (2).

$$y = \sum_{l=0}^{L-1} 2^{-1} \left(\sum_{p=0}^{P-1} S_{l,p}\right) \quad (1)$$

$$S_{l,p=\sum_{m-0}^{m-1} h(m+pM)[S(M+pM)]} \quad (2)$$

Here, $l = 0,1,2,\ldots.,L-1$ and $p = 0,1,2,\ldots\ldots,P-1$since the sum of partial product is $S_{l,p}$ of the M samples.

$$\sum_{q-1}^{Q-1} 2^{-RP}[\sum_{r-0}^{R-1} 2^{-1}(\sum_{P-0}^{P-1} r + q, R, P)] \quad (3)$$

In the Eq. (3), $q-$represents as index and r represents time index. The LC-R4-RFIR architecture has Q portions and every portion contains of P DRAM based on Reconfigurable Partial Product Generators (DRPPG) and pipeline adder trees to calculate the rightmost summation followed by Shift Accumulate (SA), which performs based on Rcycles according to the next summation (second summation). Block diagram of the DRPPG structure shows in the Fig. 2.

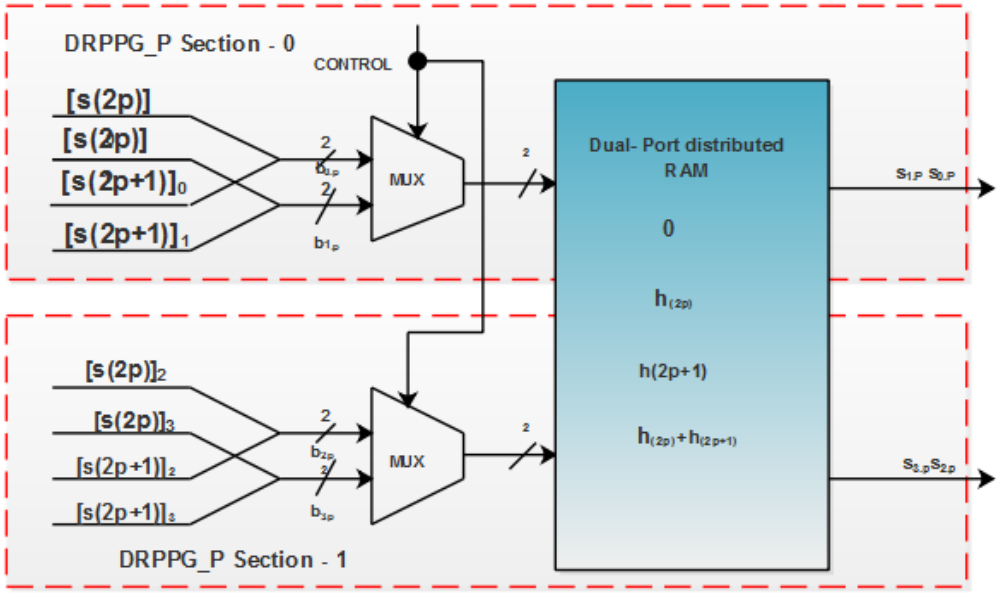

Figure. 2 Block diagram of the DRPPG structure

The LC-R4-RFIR architecture design can offer QPpartial inner products in an individual cycle and can generate LP inner product. In r^{th} cycle, P DRPPG in the i^{th} portion generate P-partial inner product $S_r + qR, p$ for $P-1$to be added by Pipeline Adder Tree (PAT). The PAT outputs are accumulated by SA, which is based on Rcycle presented in Fig. 2. The LC-R4-RFIR architecture has Q section and every section consists of the DRAM based on DRPPG and PAT to evaluate the rightmost result followed by SA which performs over R cycles according to the next (second) summation. The final stage of RFIR filter (pipeline shift and add tree) produces the shift output of the filter which employing the output from each R cycle. The accumulate value reset in each Rcycles by the control signal to keep the accumulator register ready to be employed for computing of the next output of the filter. If the increase operating clock period is fclk, LC-R4-RFIR structures support the input sample rate of fclk/R. However, it employs DRAM to minimize the total size of the LUTs by half. In the existing work, the multiplication process by using shifter occupy more area in the RFIR filter design. Hence, this paper used the R4 algorithm RFIR filter design, which occupy less area compared to existing method. Its described in the below section 3.2.

3.2 R4 multiplier using in RFIR

With the help of RFIR, the PP is reduced by half. PP plays an important role to perform the addition and multiplication in the filter design. In this paper RFIR filter design is implemented with the R4 booth recoding algorithm. The R4 multiplier technique increase speed by reducing the number of the PP by half. The basic idea of the R4 algorithm is instead of adding and shifting, every second column has been taken and multiply with 0, -1, 1, -2, and 2 to getting the same results. The steps of R4 algorithm presented below.

Step 1: Consider two inputs such as input and coefficient.

Step 2. Append zero to the Least Significant Bit (LSB) of the multiplier.

Step 3: Represent every group as PPs and to complete the set add necessary bits to coefficient.

Step 4: By applying R8 encoding on multiplicands (MDs), obtained PPs.

Step 5: Arrange the PPs like that PP2 and PP1 after leaving three p

Step 6: An Extant sign bits of all PPs according to MSB bits.

Step 7: Add entire PPs by employing high-performance adder. Table 1 shows the booth recoding for R4 algorithm.

The table 1 presented 8 different types of states. For example, from this table, MD is considered as the multiplicand.

Multiplicand (input) = 00001010
Multiplier (coefficient) = 00001001

The Fig. 3 shows the Multiplier bit pair forming, the fig.4 shows the R4 multiplication operation and fig.5 shows the 2's complement for multiplicand bits.

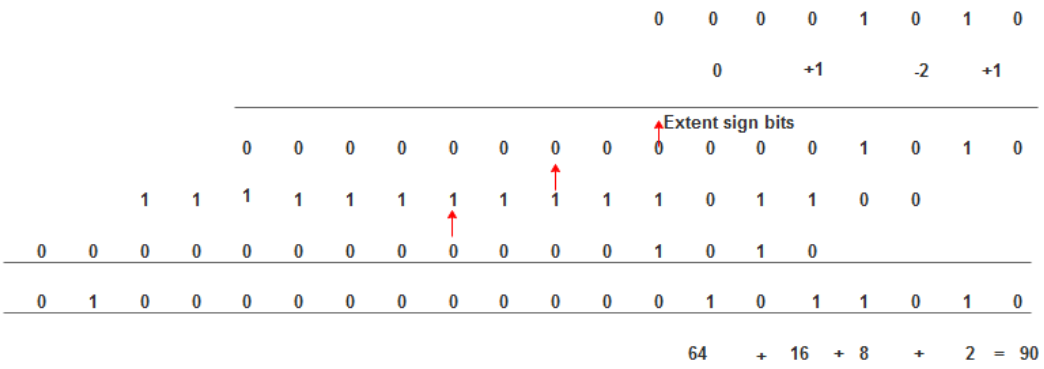

Figure. 3 Multiplier bit pair forming

Table 1. Booth recoding table for the R4

Multiplier bits block			**Recorded 1-bit pair**		**2-bit booth**	
i+1	i	i-1	i+1	i	Multiplier value	Partial Product (PP)
0	0	0	0	0	0	0xMD
0	0	1	0	1	1	1xMD
0	1	0	1	-1	1	1xMD
0	1	0	1	0	2	2xMD
1	0	0	-1	0	-2	-2xMD
1	0	1	-1	1	-1	-1xMD
1	1	0	0	-1	-1	-1xMD
1	1	0	0	0	0	0xMD

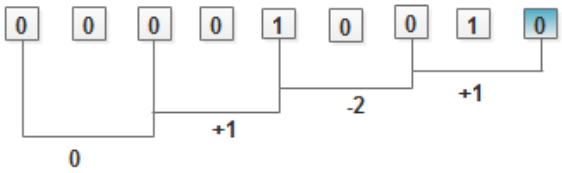

Figure. 4 R4 multiplication operation

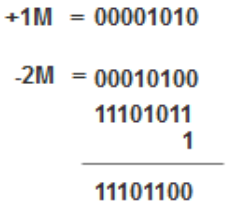

Figure. 5 Two's complement for multiplicand bits

From fig.5, -2 means left shift by one bit and compliment operation. This example shows that, the number of PPs has been minimized, and the multiplication process speed has been increased. Finally, a product of the multiplication is obtained by adding PPs. The main purpose of this algorithm is to mitigate the number of PPs, which is important in circuit design as it relates to the propagation delay in running of the circuit, power consumption and circuit complexity also mitigated in this paper.

4. Result and discussion

The RTL schematic was taken from Synplify pro tool. FPGA performance was analyzed for different devices of Virtex- 6, Virtex-6LP and Virtex-7 by using Xilinx 14.4 ISE tool.

4.1 LUT

A LUT stands for Lookup Table, in common terms a table determines what is the result for any given I/Ps. With regards to combinational logic, it is called as 'truth table'**.** This table effectively characterizes how your combinational logic behaves.

4.2 Flip-flop

Flip-flops are binary shift registers used to synchronize the logic and save logical states between clock cycles inside an FPGA circuit. On each clock edge, a flip-flop latch 1 or 0 esteem on it's I/P and holds that esteem consistent until the point that the following clock cycle.

4.3 Slices

Logic resources are resources on the FPGA that perform logic functions. Logic resources gathered in slices to make configurable logic squares. A slice contains an arrangement of LUTs, flip-flops, and multiplexers. A LUT is a collection of logic gates hard-wired on the FPGA.

4.4 Frequency

Frequency is defined as the rate at which something occurs over a particular period of time (or) given a sample.

Table 2. Comparison of different Xilinx FPGA devices for the Existing and LC-R4-RFIR method

Target FPGA	Circuit	LUT	Flip-flop	Slice	IOB	Frequency (MHz)
Virtex6 xc6vcx75t	Bonetti [7]	69/46560	38/91320	45/11640	18/240	158.743
	Jia [8]	55/46560	27/91320	40/11640	14/240	256.32
	Mohanty [12]	45/46560	21/93120	36/11640	11/240	310.214
	LC-R4-RFIR	26/46560	12/91320	10/11640	10/240	436.77
Virtex 6LP xc6vl75tl	Bonetti [7]	68/46560	37/91320	40/11640	16/240	127.945
	Jia [8]	51/46560	29/91320	35/11640	13/240	280.21
	Mohanty [12]	26/46560	20/93120	27/11640	11/240	350.214
	LC-R4-RFIR	26/46560	12/91320	11/11640	10/240	409.5
Virtex7XC 7X330t	Bonetti[7]	73/204000	39/408000	38/51000	19/600	155.07
	Jia [8]	66/204000	31/408000	35/51000	11/600	210.32
	Mohanty [12]	54/46560	29/93120	28/11640	10/240	422.211
	LC-R4-RFIR	26/204000	12/408000	10/51000	10/600	546.44

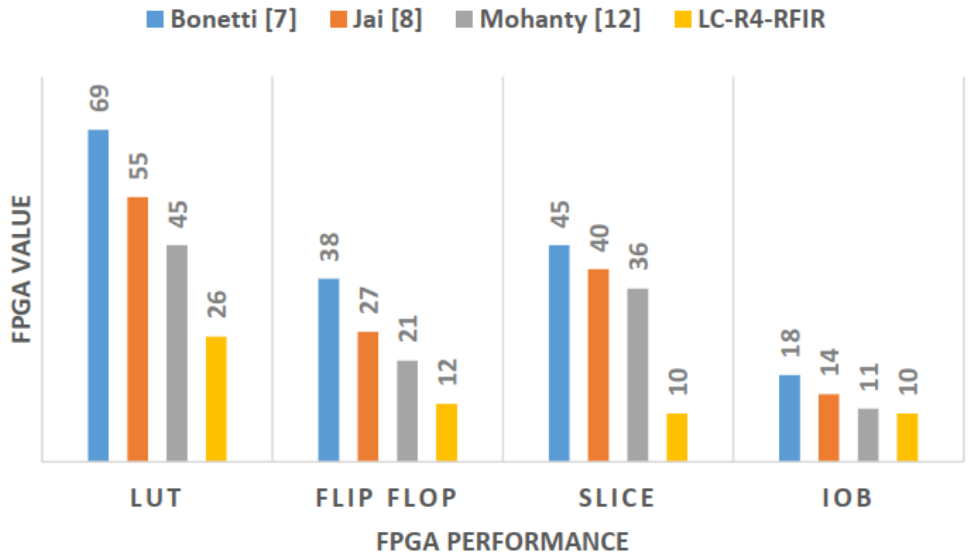

Figure. 6 Comparison of the FPGA performance virtex-6 for the Existing and LC-R4-RFIR method

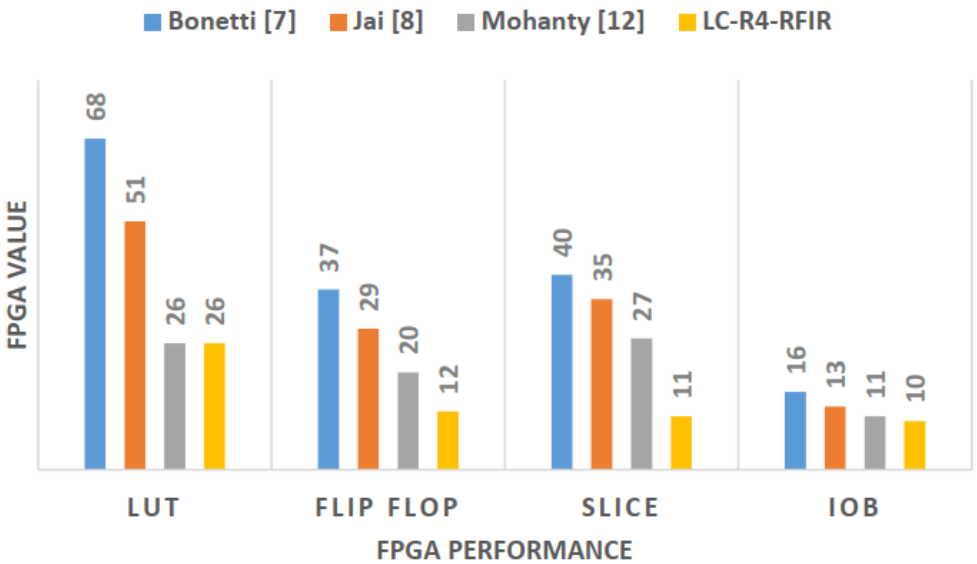

Figure. 7 Comparison of the FPGA performance virtex-6LP for the Existing and LC-R4-RFIR method

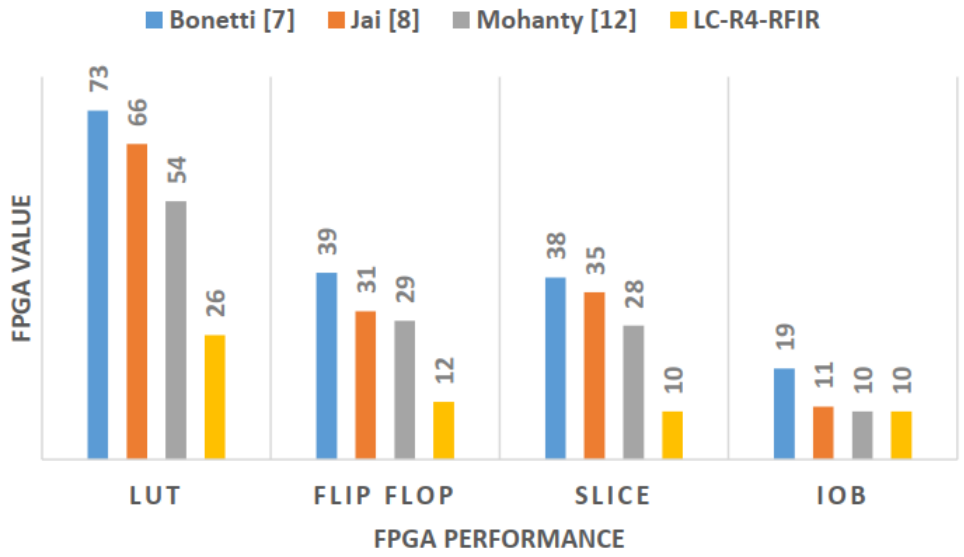

Figure. 8 Comparison of the FPGA performance virtex-7 for the Existing and LC-R4-RFIR method

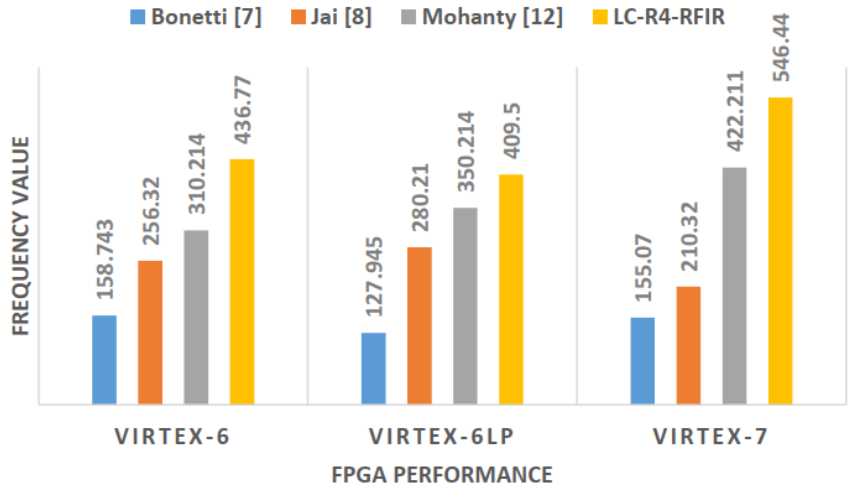

Figure. 9 Comparison of the frequency for Virtex-6, Virtex-6 LP, and Virtex7

In this research work, the LC-R4-RFIR and existing methods are implemented by using Xilinx tool that results are tabulated, which is shown in the tab.2. Table 2 shows the implementation of different Xilinx FPGA devices for existing and LC-R4-RFIR methods, which used for analyzing the performance parameters like LUTs, the number of flip-flops, slices, Input Output Block (IOB) and operating frequency for Virtex-6, Virtex-6LP, Virtex-7. In existing method [7] [8], shifter and adder has been used to perform the multiplication operation. That method requires more area and computation time, and hardware utilization. To overcome that problem, R4 algorithm is introduced in this paper to improve the FPAG performances. These four methods have been implemented and tabulated. From the tab.2, it is clear that the LUT, flip-flop, slices reduced and operating frequency increased in LC-R4-RFIR method compared to the existing methods. Due to the reduction of those parameters, the area minimized in RFIR architecture. These FPGA results have been taken from Xilinx software. In the fig.6, fig.7 and fig.8 shows the performance of virtex-6, virtex-6LP, and virtex-7 for existing method and LC-R4-RFIR method. In the fig.9 shows the comparison of the frequency for Virtex-6, Virtex-6LP, and Virtex-7.

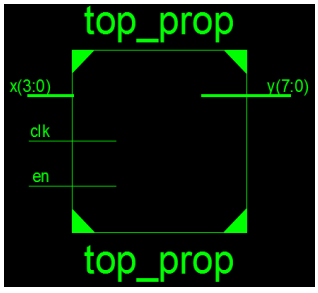

Figure. 10 RTL schematic of the top module for Virtex 6 LP device

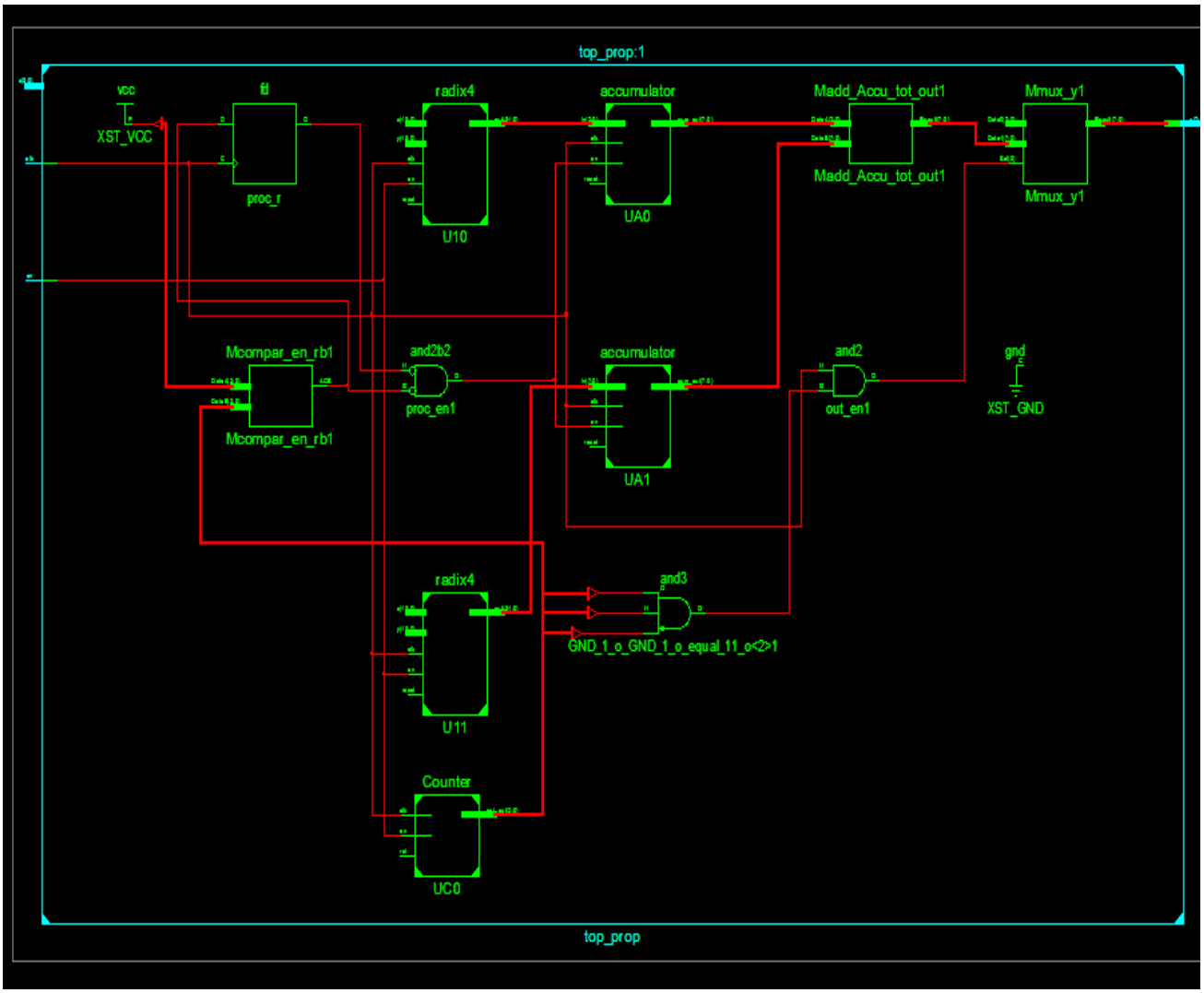

Figure. 11 RTL schematic of internal blocks for Virtex – 6 LP

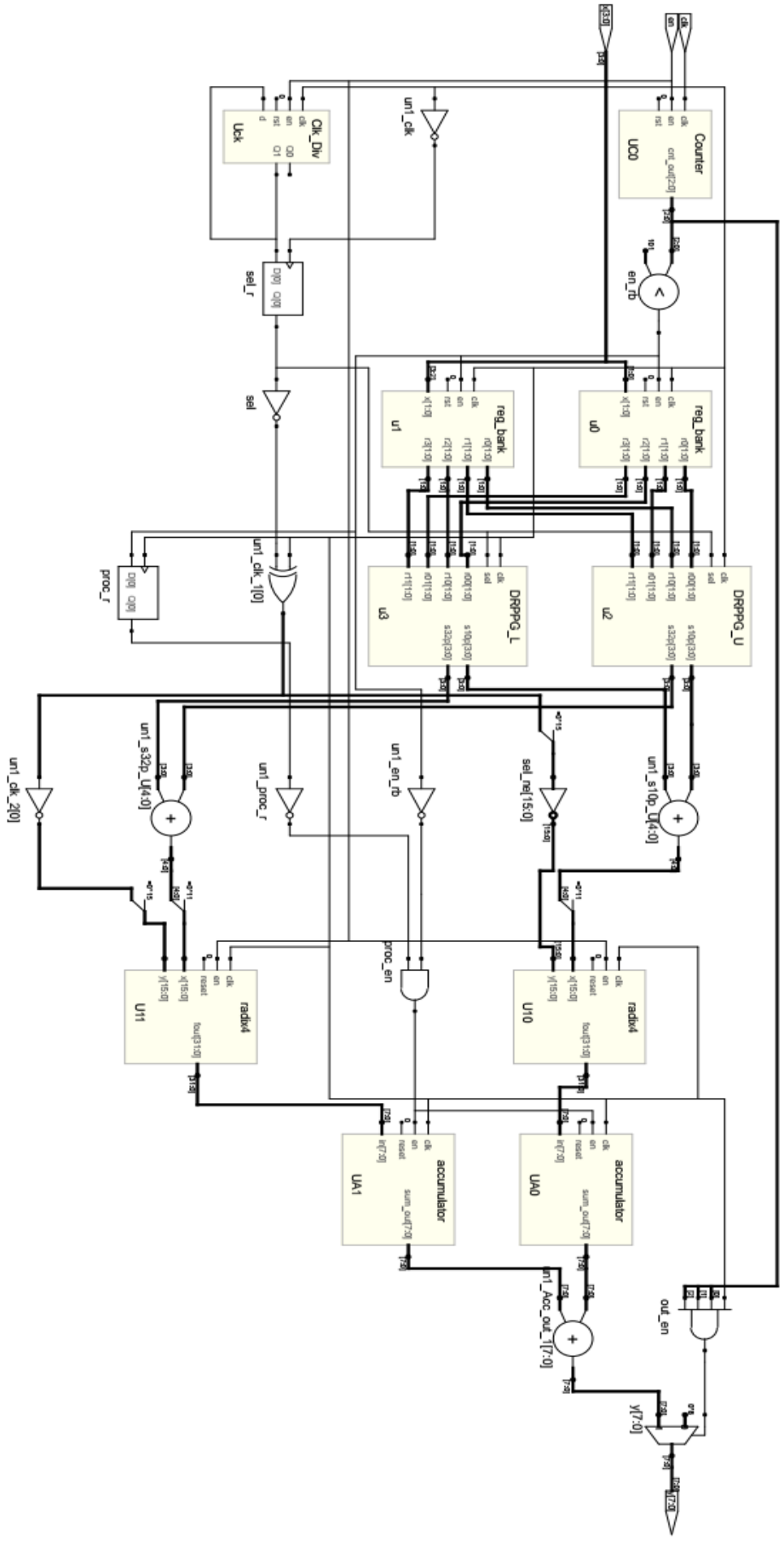

Figure.12 Block diagram of the RTL schematic for LC-R4-RFIR

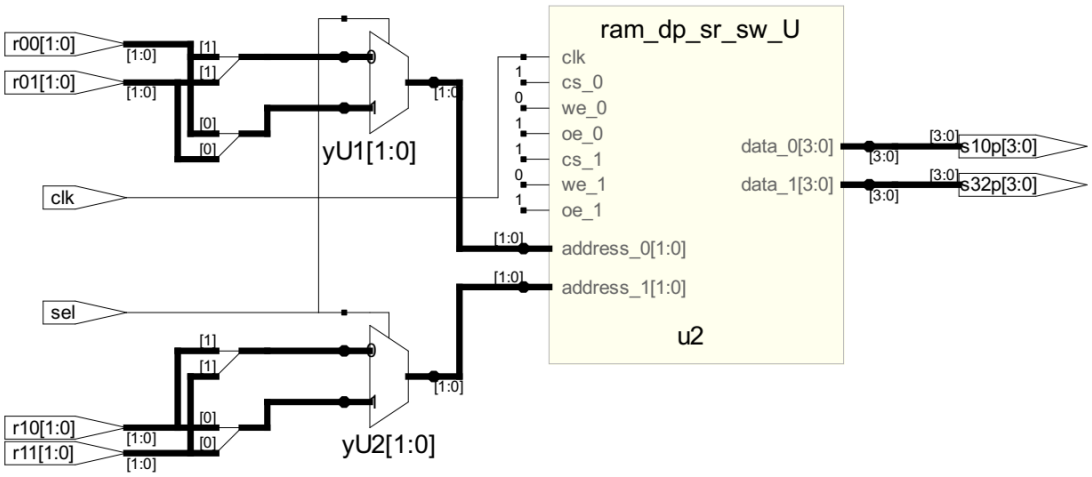

Figure. 13 RTL schematic of DRPPG module

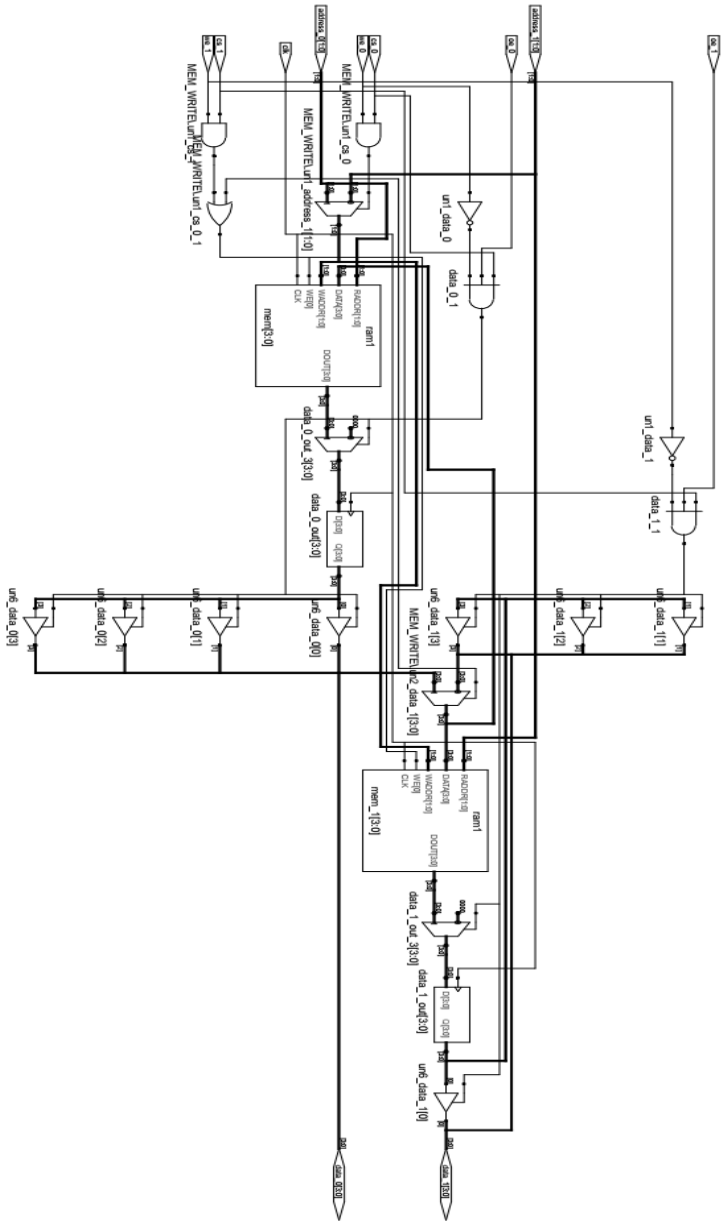

Figure. 14 RTL schematic of DROM module

Project File:	Xilinx_prop.xise	**Parser Errors:**	No Errors
Module Name:	top_prop	**Implementation State:**	Placed and Routed
Target Device:	xc6vlx75tl-1Lff484	**•Errors:**	
Product Version:	ISE 14.4	**•Warnings:**	
Design Goal:	Balanced	**•Routing Results:**	All Signals Completely Routed
Design Strategy:	Xilinx Default (unlocked)	**•Timing Constraints:**	All Constraints Met
Environment:	System Settings	**•Final Timing Score:**	0 (Timing Report)

Device Utilization Summary				[-]
Slice Logic Utilization	**Used**	**Available**	**Utilization**	**Note(s)**
Number of Slice Registers	12	93,120	1%	
Number used as Flip Flops	12			
Number used as Latches	0			
Number used as Latch-thrus	0			
Number used as AND/OR logics	0			
Number of Slice LUTs	26	46,560	1%	
Number used as logic	26	46,560	1%	
Number using O6 output only	22			
Number using O5 output only	0			
Number using O5 and O6	4			
Number used as ROM	0			
Number used as Memory	0	16,720	0%	
Number used exclusively as route-thrus	0			
Number of occupied Slices	11	11,640	1%	
Number of LUT Flip Flop pairs used	26			
Number with an unused Flip Flop	14	26	53%	
Number with an unused LUT	0	26	0%	
Number of fully used LUT-FF pairs	12	26	46%	
Number of unique control sets	3			
Number of slice register sites lost to control set restrictions	12	93,120	1%	

Figure. 15 FPGA results for proposed Virtex - 6 LP

The fig. 10 shows the RTL schematic of the top module for Virtex 6 LP. It is taken from the Xilinx tool. The RTL schematic internal block presented in the fig. 11. The fig. 12, fig.13, and fig.14 shows the RTL schematic of entire block, DRPPG, and Distributed Read Only Memory (DROM) which is taken from the Synplify pro tool. The Virtex - 6LP device output is shown in fig. 15. These results are obtained from the Xilinx tool, which is shown in the screenshot for verification purpose. From this FPGA result screenshot, it's clear that FPGA performance has been improved in LC-R4-RFIR method compared to conventional methods.

5. Conclusion

In this paper, LC-R4-RFIR filter design has been implemented in Xilinx tool by using Verilog code. In this filter design, RFIR filter has been implemented by using R4 algorithm, which takes less area and less hardware utilization compared to the existing methods. A number of LUT, Flip-flop, Slice, IOB were reduced and the frequency range increased for three types FPGA devices like Virtex-6, Virtex-6 LP, Virtex-7 by using Xilinx tool compared to the conventional FIR filter designs. For example, Virtx-6 results, average reduction 42.22% of LUT, 45.85%

of flip flop, 72.22% of slice, 9.09% of IOB compared to the existing method Mohanty [12]. In future work, this RFIR filter design will be performed based on optimal adders to further mitigate hardware utilization.

References

[1] J.L.M. Iqbal and S. Varadarajan, "High-Performance Reconfigurable FIR Filter Architecture Using Optimized Multiplier", *Circuits, Systems, and Signal Processing*, Vol.32, No.2, pp.663-682, 2013.

[2] J. Chen, J. Tan, C.H. Chang, and F. Feng, "A new cost-aware sensitivity-driven algorithm for the design of FIR filters", *IEEE Transactions on Circuits and Systems I: Regular Papers*, Vol.64, No.6, pp.1588-1598, 2017.

[3] C.Y. Yao, W.C. Hsia, and Y.H. Ho, "Designing hardware-efficient fixed-point FIR filters in an expanding sub-expression space", *IEEE Transactions on Circuits and Systems I: Regular Papers*, Vol.61, No.1, pp.202-212, 2014.

[4] S.Y. Park and P.K. Meher, "Efficient FPGA and ASIC realizations of a DA-based reconfigurable FIR digital filter", *IEEE Transactions on Circuits and Systems II: Express Briefs*, Vol.61, No.7, pp.511-515, 2014.

[5] N. Bhagyalakshmi, K.R. Rekha, and K.R. Nataraj, "Design and implementation of DA-based reconfigurable FIR digital filter on FPGA", In: *Proc. of International Conf. on Emerging Research in Electronics, Computer Science and Technology*, pp. 214-217, 2015.

[6] A. Liacha, A.K. Oudjida, F. Ferguene, M. Bakiri, and M.L. Berrandjia, "Design of high-speed, low-power, and area-efficient FIR filters", *IET Circuits, Devices & Systems*, Vol.12, No.1, pp.1-11, 2017.

[7] A. Bonetti, A. Teman, P. Flatresse, and A. Burg, "Multipliers-driven perturbation of coefficients for low-power operation in reconfigurable FIR filters", *IEEE Transactions on Circuits and Systems I: Regular Papers*, Vol.64, No.9, pp.2388-2400, 2017.

[8] R. Jia, H.G. Yang, C.Y. Lin, R. Chen, X.G. Wang, and Z.H. Guo, "A computationally efficient reconfigurable FIR filter architecture based on coefficient occurrence probability", *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, Vol.35, No.8, pp.1297-1308, 2016.

[9] S.J. Lee, J.W. Choi, S.W. Kim, and J. Park, "A reconfigurable FIR filter architecture to trade off filter performance for dynamic power consumption", *IEEE Transactions on Very Large Scale Integration Systems*, Vol.19, No.12, pp. 2221-2228, 2011.

[10] H. Jiang, J. Han, F. Qiao, and F. Lombardi, "Approximate radix-8 booth multipliers for low-power and high-performance operation", *IEEE Transactions on Computers*, Vol.65, No.8, pp.2638-2644, 2016.

[11] K.H. Chen and T.D. Chiueh, "A low-power digit-based reconfigurable FIR filter", *IEEE Transactions on Circuits and Systems II: Express Briefs*, Vol.53, No.8, pp.617-621, 2006.

[12] B.K. Mohanty, P.K. Meher, S.K. Singhal, and M.N.S. Swamy, "A high-performance VLSI architecture for reconfigurable FIR using distributed arithmetic", *Integration, the VLSI Journal*, Vol.54, pp.37-46, 2016.

[13] N. Sriram and J. Selvakumar, "A Reconfigurable FIR Filter Architecture to Trade Off Filter Performance for Dynamic Power Consumption", *Int. J. Adv. Comput. Theor. Eng.*, Vol.2, No.1, 2013.

[14] S. Ramanathan, G. Anand, P. Reddy, and S.A. Sridevi, "Low Power Adaptive FIR Filter Based on Distributed Arithmetic", *Int. Journal of Engineering Research and Applications*, Vol.6, No.5, pp.47-51, 2016.

[15] M. Pristach, V. Dvorak, and L. Fujcik, "Enhanced Architecture of FIR Filters Using Block Memories", *IFAC-PapersOnLine*, Vol.48, No.4, pp.306-311, 2015.

[16] S. Bhattacharjee, S. Sil, and A. Chakrabarti, "Evaluation of Power Efficient FIR Filter for FPGA based DSP Applications", *Procedia Technology*, Vol.10, pp.856-865, 2013.