ANALYSIS AND DESIGN OF FIR FILTER USING MODIFIED CARRY LOOK AHEAD ADDER BASED ON DECIMAL FPGA LOGIC BLOCK ARCHITECTURES

[1]ADLENEPRIYATHARISINI J, [2]KAVITHA.S

[1]PG scholar, Department of Electronics and Communication Engineering, Nandha Engineering College (Autonomous)

[2]Professors, Department of Electronics and Communication Engineering, Nandha Engineering College (Autonomous)

Abstract -The dynamic growth in portable multimedia system devices and communication system has exaggerated the demand for space and power saving high speed Digital Signal processing (DSP) system. The Finite Impulse Response (FIR) Filter is the necessary part in designing an digital signal process system. Usage of digital Finite Impulse Response (FIR) filter is one of the prime block in DSP. Digital multipliers and adders are the foremost crucial arithmetic purposeful units in FIR filters and conjointly decide the performance of whole system. Thus, the low grid style has become a significant performance goal. This paper proposes an FIR filter that is designed using Carry-Look ahead adder and multiplier; where the multiplier is proposed by internal circuit of Modified Carry Look ahead Adder. CLA adder is employed for addition operation that uses quickest carry generation technique to increase the speed by reducing the time needed to repair carry bits and multiplier factor performs multiplication method in hierarchical manner. Thus, the planned methodology will minimize the active power and delay of the FIR filter. The tentative results shows that the FIR filter multiplier factor methodology achieves less quantity of delay and power reduction compared to traditional methodology. The planned FIR filter is programmed using Verilog code and was synthesized and enforced using Xilinx ISE 14.2 tool. and therefore the power is analyzed using power analyzer.

I. INTRODUCTION

1.1 Field-Programmable Gate Array (FPGA)

An FPGA is an integrated circuit that can be programmed after manufacturing to function as any digital circuit determined by the designer. The key difference between FPGA and Application Specific Integrated Circuit (ASIC) is that the ASIC can only be used for a certain application. The major building blocks in modern FPGAs consist of Configurable Logic Blocks (CLBs), Interconnections, Input/output Blocks (IOBs) and embedded blocks such as DSP blocks. An overview of the architecture of modern FPGAs is shown in figure 1.1.

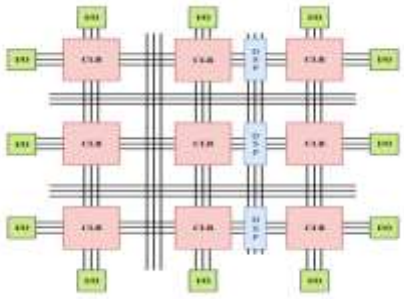


Figure.1.1. An overview of a typical FPGA architecture.

The red boxes indicate a number of CLB blocks distributed over an FPGA. The green boxes indicate input/output blocks of an FPGA. The blue boxes indicate DSP blocks where the focus of this work lies. The vertically and horizontally distributed lines indicate interconnections between the blocks.

1.1.1 Configurable Logic Block (CLB)

The CLB is used to implement logic functions and mathematical operations. Inside each CLB, are so called Configurable Logic Elements (CLEs) which consists of Look-Up Table(LUTs), D-flip flops and a 2-to-1 multiplexers, as illustrated in figure 1.2 .

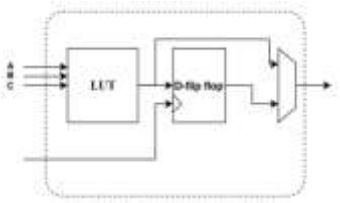


Figure 1.2. Simplified internal structure of a typical CLE.

The purpose of a LUT is to implement mathematical operations such as addition. The LUT and its three-inputs is shown in figure 1.2. The core of the FPGA consists of thousands of CLE copies connected with each other. The LUT is made of a series of cascaded multiplexers where the LUT inputs are used as the select lines and the inputs to multiplexers is a 1-bit SRAM memory that is set to either 0 or 1. The internal structure of a LUT is shown in figure 1.3. The three-inputs of a LUT control the selection of the 1/0 input bit of the multiplexers.

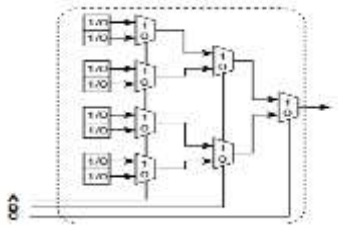


Figure.1.3. Simplified internal structure of a LUT.

Input/Output blocks are placed all around FPGA's edges, as shown in figure 1.1 where the IOBs function as a communication interface between the internal signals and the external pins.

1.2 DSP Block

Eventhough a Configurable Logic Block is capable of performing computation operations and storing data, it would be too slow and utilizes a huge amount of resources because a CLB is designed to perform different types of functions. As the need for the FPGA is increasing, modern FPGAs provide embedded blocks to perform specific functions, some of which are memory blocks and DSP blocks. The introductions of these embedded blocks clearly reduce area utilization, power consumption and as a consequence results in better performance. Although the internal structure of DSP blocks and its related details are different depending upon manufacturer and device version, the overall architectures of most DSP blocks look alike. A DSP block consists of an adder and multiplier followed by an accumulator. The aims of introducing DSP blocks are to enhance performance of these computation operations. Also, there are specific connections in each DSP block that can be used to connect multiple DSP blocks to each other which in turn can be used to implement an efficient FIR filter.

1.3 Digital Filter

A digital filter is used to modify signal characteristics in the time and/or frequency domain. A digital signal is obtained by sampling the continuous signal in different time frames after which the signal is represented as a sequence of discrete values, unlike the analog signal which is continuous and represented as a function of time.

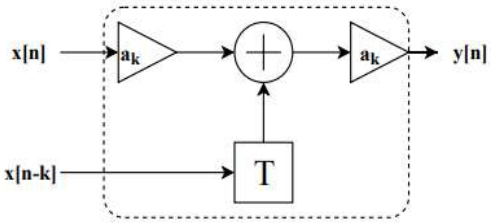


Figure.1.5 An example of a signal flow graph

The ak indicate FIR filter coefficients and x(n- k) indicate delayed versions of input signal x[n]. The fundamental components used in digital filters, as shown in figure 1.5, are multipliers, adders and delay elements.

1.3.1 Finite-length Impulse Response (FIR) Filter

LTI filters are usually divided into two types which are finite-length impulse response (FIR) and infinite-length impulse response (IIR). Finite-length impulse response means that the impulse response becomes zero after a finite number of samples. In contrast, the infinite length impulse response has an infinite number of sample values. The work presented in this thesis covers the FIR filter only.

II. LITERATURE SURVEY

M. Keerthi et.al proposed a modified mixed-grain architecture with the ALU-like functionality reduces the LUT memory size by a factor of 16 compared to commercial FPGAs, and mapped onto standard cells has a 1.9-3.3 times higher data path mapping efficiency. FPGA architectures may be an interesting alternative to the traditional general-purpose FPGA devices, especially if characteristics of a target application domain are known a prior. FPGA Implementation of Distributed Arithmetic for FIR Filter [1]. Sahin, Suhap & Kavak, Adnan & Becerkli, Yaşar & Demiray, Hilmi Implemented the floating point arithmetics using an FPGA. Floating point operations, which find their applications in vast areas such as many mathematical optimization methods, digital signal and image processing algorithms, and Artificial Neural Networks (ANNs), require much area and time for ordinary implementation on Field Programmable Gate Arrays (FPGAs). However, meaningful floating point arithmetic implementation on FPGAs is quite difficult with low level design specifications due to mapping difficulties [2]. Parandeh-Afshar et al implemented the compressor tree architecture technique and provide the hardened compressors to soft logic to boost up the speed of addition of multi inputs in the are of DSP and video processing applications [3]. Thomson designed an arithmetic logic unit made up of reversible gates for performing modular operations. Designed a reversible ALU for performing logic and arithmetic operations here in this paper we propose a new design of ALU made up of reversible gates with better power saving property [4]. An efficient operational unit could be designed with the help of simulation tool and the functional units with optimized count of slices, count of flip LUTs. The maximum frequency can be obtained from the analysis of timing view. Power count of the system should be proportional to the respective frequency. Time delay can be optimized using Vedic mathematics technique in implementation of multipliers as the speed of ALU depends prominently on the speed of multiplier. Further in the future work the speed of ALU can be increased by using various fast and efficient multipliers available in the literature [5]. Murray, Kevin & Luu has proposed fracturable LUT architectures which use two bits of hardened arithmetic achieve 25% better area-delay product than non-fracturable LUT architectures without hardened arithmetic.

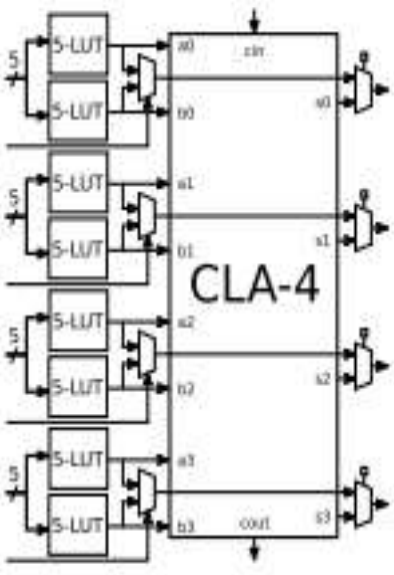
III. EXISITNG CLA ADDER DESIGN

Carry look Ahead adder: A carry look-ahead adder improves speed by decreasing the measure of time required to decide carry bits. It tends to be stood out from the more straightforward, yet normally slower, carry adder for which the carry bit is determined close by the total piece, and each piece must hold up until the past carry has been determined to start ascertaining its very own outcome and carry bits.

$Pi = Ai \oplus Bi$
$Gi = Ai*Bi.$
The output sum and carry can be defined as:
$S = Pi \oplus Ci.$
$Ci +1 = Gi + PiCi$

Gi is known as the carry Generate signal since a carry (Ci+1) is generated whenever Gi =1, regardless of the input carry (Ci). Pi is known as the carry propagate signal since whenever Pi =1, the input carry is propagated to the output carry, i.e., Ci+1. = Ci.

The BLE consists of a non-fracturable six-input LUT with an choose able registered output pin. There are cin and cout pins into and out of the BLE, respectively, to drive a hard adder. More industry trends on using larger LUTs has interesting implications in terms of the efficiency of addition. When implementing arithmetic using only 4-LUTs, every bit of addition requires one LUT for the sum and another LUT for the carry. With 5-LUTs and larger, a soft implementation of arithmetic can be more efficient. Shows how three LUTs can implement two bits of addition. With fracturable 6-LUTs, this benefit grows even larger as fracturing into the two 5-LUTs mode allows implementation of both the 2-bit carry and a sum operation in a single fracturable 6-LUT. Figure 3.1 shows balanced LUT interaction using four bit CLA.



.

Figure 3.1 Four-bit CLA with *balanced* LUT interaction.

IV. A New Logical Full Adder

The truth table for addition 2 bits with carry flag is presented as follow:

Table 4. 1: Truth table Full-Adder

C_{in}	x	y	Adder	
			Sum	C_{out}
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

From table 4.1, a conclusion reached regarding output values: Sum values and Carry-out flags. This conclusion derives from the idea of the operating current (electronic), by switching on/off the Current to obtain the expected logic value:

$$C_{in}=0\rightarrow\begin{cases}Sum\begin{cases}Sum=x\ xor\ y\\ x=0\rightarrow Sum=y\\ x=1\rightarrow Sum=\overline{y}\end{cases}\\ C_{out}\begin{cases}C_{out}=x\ and\ y\\ x=0\rightarrow C_{out}=0\\ x=1\rightarrow C_{out}=y\end{cases}\end{cases}$$

$$C_{in}=1\rightarrow\begin{cases}Sum\begin{cases}Sum=x\ nxor\ y\\ x=0\rightarrow Sum=\overline{y}\\ x=1\rightarrow Sum=y\end{cases}\\ C_{out}\begin{cases}C_{out}=x\ or\ y\\ x=0\rightarrow C_{out}=y\\ x=1\rightarrow C_{out}=1\end{cases}\end{cases}$$

Two circuits are built for sum and carry-out based on MUX2-1. Those use MUX as a switch which is controlled to allow expected values to pass through.

The circuit for sum value: A circuit is built through a control of values including input x or input y that are shown in Table 4.2 and Figure 4.1

Table 4.2: Truth table Sum value

Controlled by Cin and y			Controlled by Cin and x		
Cin	**y**	**Sum**	**Cin**	**x**	**Sum**
0	**0**	**x**	**0**	**0**	**Y**
0	**1**	$\overline{x}$	**0**	**1**	$\overline{y}$
1	**0**	$\overline{x}$	**1**	**0**	$\overline{y}$
1	**1**	**x**	**1**	**1**	**Y**

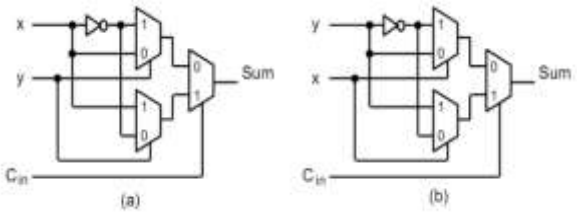


Figure 4.1: Circuit for Sum value: (a) Controlled by Cin & y ; (b) Controlled by Cin & x

4.2 Implementing with LUTs

Two LUT6-1's are used to establish the FA. LUT6-1 is a basis LUT inside the slice. Because the sum value and carry-out flag above can be executed at the same time, two basic LUT6-1's used: one for Sum circuit and one for Carry-out circuit. From the observation the two truth tables of addition as well as circuits of Carry-out flag for addition, it is learnt that it is possible combine them to have FA with control=0: following as Figure.4.2

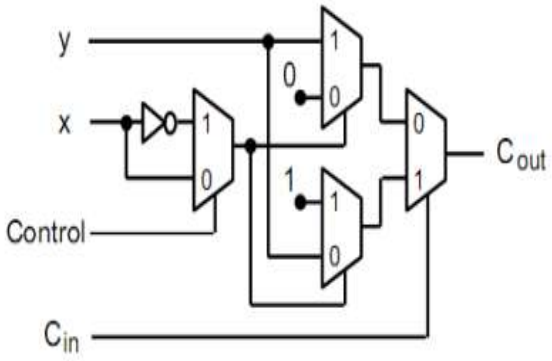


Figure 4.2 Diagram Cout with a control pin

Moreover, the final circuit for FA is obtained by using one LUT (6-input, 2-output) that is divided into 2 small LUT's, one for Sum and one for Carry-out flag .When Carry-out flag is equal to 1, addition shows 'overload'.figure 4.3 shows new full adder using one LUT (6-2)

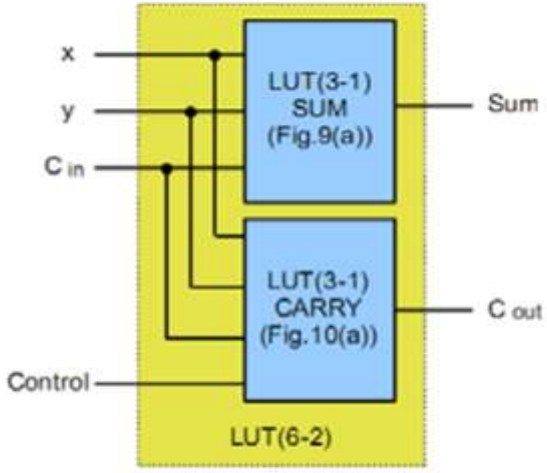


Figure 4.3 : New Full Adder (n_FA)

V. PROPOSED CLA ADDER USING PROPOSED FULL ADDER

5.1 Logical Structure of the Hierarchical Carry Look ahead Adder

In contrast to the well known ripple-carry design which has a total time delay that is proportional to the length N of the adder, the carry look ahead design is suited to generate all carries

simultaneously by additional logic circuitry. This results in a constant addition time, independent of the length of the adder. For a good understanding we give a short overview of the logical structure of this adder type. A more detailed description can be found in. Let Ci−1 be the carry input to the ith bit position and C−1 the carry input to the least significant position. Let Si and Ci be the sum and carry outputs of the ith stage. Then, the sum and carry bit can be described as

$$S_i = (A_i \oplus B_i) \oplus C_{i-1} = P_i \oplus C_{i-1} \quad \text{and} \quad C_i = A_i \cdot B_i + (A_i \oplus B_i) \cdot C_{i-1} = G_i + P_i \cdot C_{i-1},$$

where Pi = Ai⊕ Bi and Gi = AiBi· are defined as PROPAGATE- and GENERATE variables, respectively, which can be generated simultaneously from the external inputs Ai and Bi within the GP-level as illustrated in Figure 3. If we recursively apply the carry formula given above we obtain the following set of carry equations:

$$C_0 = C_{-1} \cdot P_0 + G_0$$
$$C_1 = C_{-1} \cdot P_0 \cdot P_1 + G_0 \cdot P_1 + G_1$$
$$C_2 = C_{-1} \cdot P_0 \cdot P_1 \cdot P_2 + G_0 \cdot P_1 \cdot P_2 + G_1 \cdot P_2 + G_2$$
$$\ldots$$
$$C_{n-1} = C_{-1} \cdot P_0 \cdot P_1 \cdot \ldots \cdot P_{n-1} + \ \ldots \ + G_{n-2} \cdot P_{n-1} + G_{n-1}$$

These equations show that all carry inputs Ci are available simultaneously and all sum bits Si can be generated in parallel as illustrated in Figure 2 for a 4-bit carry look ahead adder. As also shown in Figure 5.1 distinguish between structure levels GP, CLA and SUM according to the generated signals within the levels.

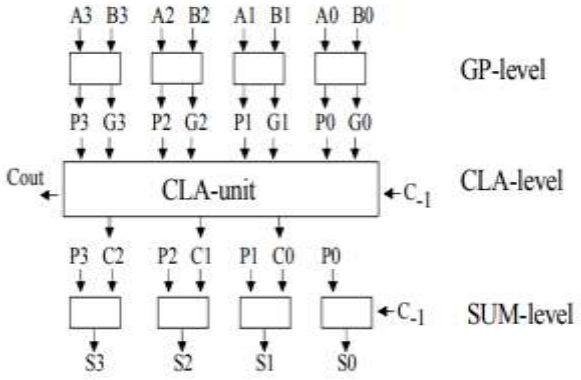


Figure. 5.1 Structure of a 4-bit carry Look ahead adder

Theoretically, it is possible to build adders of any word length if the CLA unit can be freely expanded. However, in practice the complexity of the CLA unit is limited. This leads to a hierarchical structure based on Block-Carry-Look ahead (BCLA) units. Each of them generates only a limited number of carries and, additionally, BLOCK-PROPAGATE (P0*) and BLOCK-GENERATE (G0*) signals which are used to evaluate carries of the following BCLA level. Figure 4 shows a two-level CLA unit of a 16-bit CLA adder in a 4-bit BCLA unit configuration. Each of the 4- bit BCLA units is suited to evaluate three carries.

The most significant carry signal Cout can be written as C C out = −1 · P + G 0 0 max max , where max refers to the variables of the most significant BCLA unit. The number of BCLA levels is given by

$$L=[^{\log}\text{BCLAbitwith}^{N}]$$

The implementation of hierarchical CLA adders is done in four steps: Adaptive structure generation, technology mapping, partitioning and placement. We present a method for realizing all these steps for any SRAM-based FPGA which can be described by the generic models defined in Section 2. During each design step we will efficiently use knowledge about the logical structure. As described, the length of a BCLA unit is not

bound to a special value. Therefore, we can write the equations for BLOCK-PPOPAGATE, -GENERATE and CARRIES

Since the carry look ahead adder is chosen for high-performance addition the aim of our logic adaption step is to reduce the BCLA level count to a minimum. According to Eq.(2) this is accomplished by maximizing the bitwidth of the BCLA units which increases the number of carry signals evaluated within a unit. As shown in Table 2 the complexity of a BCLA is limited by the implementation of the BLOCKGENERATE signal which needs a signal input of 2*BCLA bit width -1 and a number of sum terms which equals BCLA bit width. Since this is also valid for the most complex CARRY signal we can use the same implementation method for both. Therefore, our aim is to determine that configuration of a FPGA device which is suitable to implement the most complex BLOCK-GENERATE signal within a CLB.

VI. FIR FILTER DESIGN

The FIR filter consists of three main components: A D-FF to implement a simple delay. A Multiplier is used to implement the filter coefficients. An Adder to sum the nodes at the end of each tap. Functional verification of all the adders and multiplier are performed and these modified architectures are applied in 4-tap FIR filter finally results are summarized.

Here X(n) is input filter coefficients and B0, B1, B2 and Bn are transfer function coefficients and Y(n) is output filter coefficient. Multipliers can be replaced with shift and add operation, that is MCM (Multiple Constant Multiplication) in which a set of constants (here h0, h1…, is multiplied with a variable (here x(n)).

In FIR filter the result of the delay operates on the process of input samples. The values of hn are the coefficients in which they are used for multiplication operation. So that the output reaches at time and the summation operation is delayed for all the other samples that are multiplied by the appropriate coefficients. In CLA every bit in a binary sequence that to be added to the operation, the carry-look ahead logic will determine whether the bit pair will generate a carry or propagate a carry.

VII. EXISTING FIR FILTER DESIGN

Carry Look Ahead Adder is a kind of fast adder used in digital logic circuits. The CLA computes all the carry bits before the sum, which decreases the delay time to calculate the result of the larger value bits. The CLA solves the carry delay problem by predetermination of the carry signals in advance to the basis of the input signals. This technique was contrasted with the slow adder like RCA, where the the multiplier is proposed by internal circuit of Modified Carry Look ahead Adder which consist of four partial full adder with XOR gates and NAND gate. In Figure it shows the partial products has been taken to generate and propagate signal. These generate and propagate signal has been used in look ahead section to generate carry.

Here the carry is eliminated from first section and directly utilizing the propagate operation and generate to the input block of look ahead section. The FIR filter consists of three main components: A D-FF to implement a simple delay. A Multiplier is used to implement the filter coefficients. An Adder to sum the nodes at the end of each tap. Functional verification of all the adders and multiplier are performed and these modified architectures are applied in 4-tap FIR filter finally results are summarized.

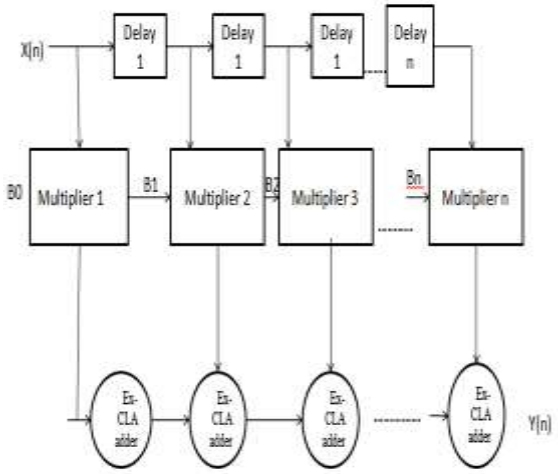


Figure. 7.1. Existing Fir Filter using CLA adder design

VIII. PROPOSED FIR FILTER DESIGN

Carry Look Ahead Adder is a kind of fast adder used in digital logic circuits. The CLA computes all the carry bits before the sum, which decreases the delay time to calculate the result of the larger value bits. The CLA solves the carry delay problem by predetermination of the carry signals in advance to the basis of the input signals. This technique was contrasted with the slow adder like RCA, where the adder should wait for the pervious carry to generate the specific result and carry bits. The CLA uses additional circuitry to generate carry bits in parallel, which eliminates time required to calculate the larger value bit result. The architecture of 4-bit CLA is shown in Fig. 5.1 CLA can be separated as two segments: Partial Full Adder (PFA) and the Carry Look-ahead Logic. The PFA produce propagate signals pi, generate signals gi and the sum output si and the carry-out bits ci +1 was generated by the look ahead logic circuit. The structure shows that Propagate P and generate G in a partial full-adder depends on the input bits only, the carry generator also not depends on its previous carry-in. As a result, once $C0$ is computed, $C4$ can reach stable state it does not want to wait for $C3$ to propagate.The FIR filter consists of three main components: A D-FF to implement a simple delay. A Multiplier is used to implement the filter coefficients. An Adder to sum the nodes at the end of each tap. Functional verification of all the adders and multiplier are performed and these modified architectures are applied in 4-tap FIR filter finally results are summarized.

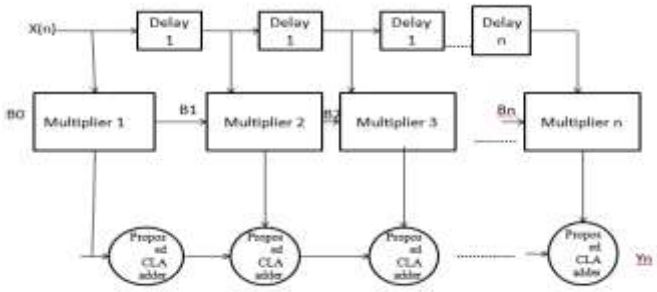


Fig.8.1. FIR Filter based on proposed CLA adder

Here X(n) is input filter coefficients and B0, B1, B2 and Bn are transfer function coefficients and Y(n) is output filter coefficient. Multipliers can be replaced with shift and add operation, that is MCM (Multiple Constant Multiplication) in which a set of constants (here h0, h1, is multiplied with a variable (here x(n)).In FIR filter the result of the delay operates on the process of input samples. The values of hn are the coefficients in which they are used for multiplication operation. So that the output reaches at time and the summation operation is delayed for all the other samples that are multiplied by the appropriate coefficients. In CLA every bit in a binary sequence that to be added to the operation, the carry-look ahead logic will determine whether the bit pair will generate a carry or propagate a carry.

IX. RESULTS AND DISCUSSION

Focus on evaluating the different hard adder variants using full application benchmarks. These allow us to evaluate the overall quality of the different implementation approaches, considering overall delay and area. Architectures which minimize the area–delay product are the most efficient. The benchmarks are full application

circuits, which include a variety of arithmetic operations including addition, subtraction and multiplication which can make use of hardened adders. The best hard adder architectures with soft LUT and fLUT architectures.

For the soft adder architectures, the critical path delay is similar, while both hard adder architectures reduce the critical path delay by 15%–16%. As a result, the combination of fLUTs with two bits of hardened addition per LUT improves the area–delay product by 25% compared to a soft (non-fracturable) LUT architecture. Table 9.1 shows the comparision between exisiting and proposed method.

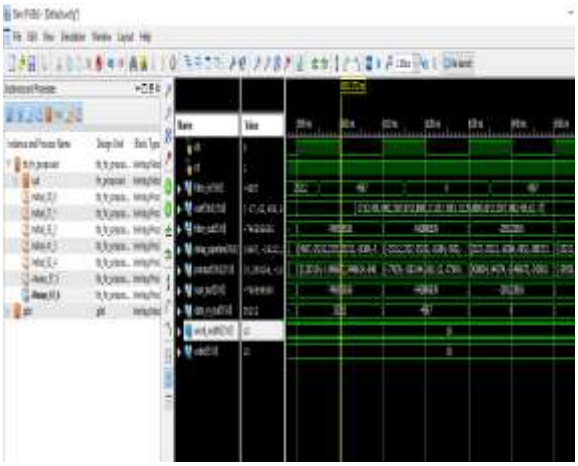
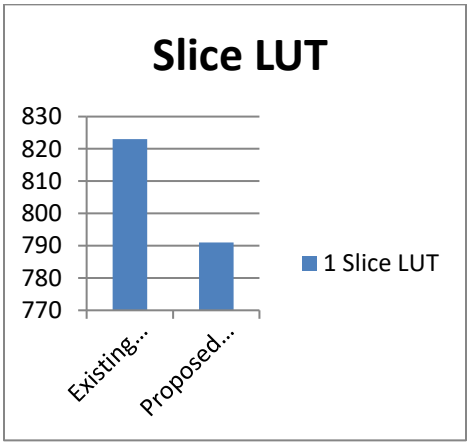
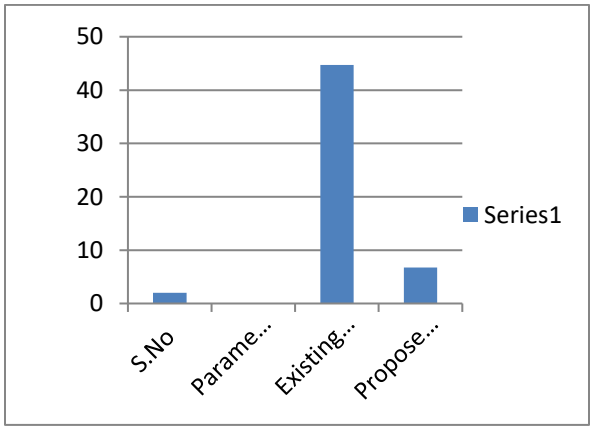


Figure 9.1 Output Wave of Proposed FIR filter design

Table 9.1 Parameters Comparison Summary for 16 Tab Fir filter design

S.No	Parameters	Existing Method	Proposed Method
1.	Slice LUT	823	791
2.	Combinational Delay in ns	44.731	6.733
3.	Power in Watts	0.179	0.126
4.	Maximum Frequency in Mhz	22.356	148.518





There is an important caveate in this comparison is not completely fair because the number of inputs to the CLBs are kept constant when in reality the ideal number of inputs to a CLB

with non-f LUTs should be lower than that for fracturable even if the underlying logic elements themselves have the same number of inputs. Hence, our area results for the non-f LUT architectures could likely be somewhat improved with cluster input count retuning. Overall, however, the case for an fLUT architecture with two bits of hard arithmetic per LUT is compelling.

X. CONCLUSION

The purpose of this paper is to propose a new FA architecture on an FPGA platform with two optimization goals. The first is to improve the FA/S ratio in order to increase speed. The second is to create a new circuit for FA/S to allow for production with fewer gates. As a result, the proposed design is based on the multiplexer and contains only two types of components: NOT gate and multiplexer, allowing the design to be easily implemented on an FPGA chip. Both designs are subjected to detailed testing using different FPGA series. The experiment revealed that the proposed n FA performed 20% to 30% faster than the standard FA using the same area resources, and the proposed n FA performed 28% to 40% faster than the standard FA using only half the resources.

The proposed design of digital FIR filter using modified multiplier based CLA is discussed. A basic analysis was performed in terms of area, power and delay. This multiplier based on CLA adder proved to be more efficient in terms of speed of operation compare to conventional multipliers. Based on the proposed multiplier, the architecture of direct form realization of digital FIR filter have been derived. The synthesis results show that the proposed FIR filter using modified multiplier based on carry look ahead adders achieves high speed and reduces the hardware cost with lesser power consumption related to conventional FIR filter with other multipliers. In future, the pipelining concept can also be extended to adder unit present in digital FIR filter to achieve better power and area reduction. Further the design of 16-tap FIR filter can be prolonged to n-tap that to be used in real time applications.

REFERENCES

[1] M. Agarwal, R. Barsainya, and T. K. Rawat, “Implementation of low power reconfigurable parametric equalizer with row bypassing multiplier on fpga,” in Computing, Communication and Automation (ICCCA), 2016 International Conference on. IEEE, 2016, pp. 1352–1357.

[2] S. Dhanasekaran, T. Thamaraimanalan, R. Sudha Anandhi and A. Mohanapriya, “Comparative Analysis of Low Power and High Speed Performance in 8-bit Different Multipliers,” Journal of Advanced Research in Dynamical & Control Systems, Vol. 10, 04-Special Issue, 2018 , pp. 89-93.

[3] V. N. Kumar, K. R. Nalluri, and G. Lakshminarayanan, “Design of area and power efficient digital fir filter using modified mac unit,” in Electronics and Communication Systems (ICECS), 2015 2nd International Conference on. IEEE, 2015, pp. 884–887.

[4] E. Prabhu, H. Mangalam, and K. Saranya, “Design of low power digital fir filter based on bypassing multiplier,” International Journal of Computer Applications, vol. 70, no. 9, 2013.

[5] M. M. Mahmoud, D. A. El-Dib, and H. A. Fahmy, “Low energy pipelined dual base (decimal/binary) multiplier, dbm, design,” Microelectronics Journal, vol. 65, pp. 11– 20, 2017.

[6] Z. Abid, H. El-Razouk and D.A. El-Dib, “Low power multipliers based on new hybrid full adders”, Microelectronics Journal, Volume 39, Issue 12, Pages 1509-1515, 2008

[7] Hasan Krad and Aws Yousif Al-Taie, “Performance Analysis of a 32-Bit Multiplier with a Carry-Look-Ahead Adder and a 32-bit Multiplier

with a Ripple Adder using VHDL", Journal of Computer Science 4 (4): 305-308, 2008

[8] M. Keerthi, "FPGA Implementation of Distributed Arithmetic for FIR filter", IJERT Vol. 1, Issue 9, November- 2012.

[9] Bahram Rashidi, "Low Power FPGA Implementation of Digital FIR Filter based on Low Power Multiplexer Base Shift/Add Multiplier", International Journal of Computer Theory and Engineering, Vol. 5, No. 2, April 2013.

[10] Mengxue Lei, "Design of High Speed FIR filter with Distributed Parallel Structure", IEEE Information Technology, Networking, Electronic and Automation Control Conference- May 2016.

[11] Mohanty BK, Meher PK. A high-performance FIR filter architecture for fixed and reconfigurable applications. IEEE transactions on very large scale integration (vlsi) systems. 2016 Feb; 24(2): 444–-52.

[12] R. M. Deshmukh and R. Keote, "Design of polyphase fir filter using bypass feed direct multiplier," in Communications and Signal Processing (ICCSP), 2015 International Conference on. IEEE, 2015, pp. 1640– 1643

[13] Vishwanath BR, Theerthesha TS. Multiplier using canonical signed digit code. International Journal for Research in Applied Science & Engineering Technology (IJRASET). 2015; 3(5): 415–20.

[14] Kamble P, Deote N, Wanjari N, Gaikwad S. Implementation of FIR filter structure for audio application using XilinxSystem generator. International Journal of Advanced Research in Computer Science and Software Engineering. 2015; 5(1): 762–5.

[15] Yan and Chen, "Low-cost low-power bypassing-based multiplier design," in Circuits and Systems (ISCAS), Proceedings of 2010 IEEE International Symposium on. IEEE, 2010, pp. 2338–2341.

[16] N. T. A. El-Kheir, M. S. El-Kharashi, and M. A. ElMoursy, "A low power programmable fir filter using sharing multiplication technique," in IC Design & Technology (ICICDT), 2012 IEEE International Conference on. IEEE, 2012, pp. 1–4.

[17] Asadi, P. and K. Navi "A novel high-speed 54-54-bit multiplier", Am. J. Applied Sci., 4 (9): 666-672, 2007

[18] Wakerly, J.F., 2006. Digital Design-Principles and Practices. 4th Edn. Pearson Prentice Hall, USA.ISBN: 0131733494.

[19] B. Ramkumar and Harish M Kittur, "Low power and area efficient carry select adder", IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol 20, no. 2,pp. 371-375,Feb 2012

[20] S. Murugeswari and S.K. Mohideen, "Design of Area Efficient and Low Power Multipliers using Multiplexer based Full Adder", Proceedings of 2nd International Conference on Current Trends in Engineering and Technology, pp. 388-392, 2014.