Design and Implementation of Binary Tree architecture for Fast Multiplication

Mr.S.B.Jadhav[#1], Mrs.J.K.Patil[*2], Mr.Y.S.Jagdale[#3]

[#]*Department of Electronics& Telecommunication Engineering,*
Bharati Vidyapeeth's College of Engineering Kolhapur, M.S India
[1]sachin30_2k6@rediffmail.com
[3] ysj@ rediffmail.com
Bharati Vidyapeeth's College of Engineering Kolhapur, M.S India
[2] *jayamala.p@ rediffmail.com*

***Abstract*— This paper presents the details of hardware implementation of modified partial product reduction tree using 4:2 and 5:2 compressors. Speed of multiplication operation is improved by using higher compressors .In order to improve the speed of the multiplication process within the computational unit; there is a major bottleneck that is needed to be considered that is the partial products reduction network which is used in the multiplication block .For implementation of this stage require addition of large operands that involve long paths for carry propagation. The proposed architecture is based on binary tree constructed using modified 4:2 and 5:2 compressor circuits. Increasing the speed of operation is achieved by using higher modified compressors in critical path. Our objective of work is, to increase the speed of multiplication operation by minimizing the number of combinational gates using higher n: 2 compressors. The experimental test of the proposed modified compressor is done using Spartan-3FPGA device (XC3S400 PQ-208).Using tree architectures for the partial products reduction network represent an attractive solution that is frequently applied to speed up the multiplication process. The simulation result shows 4:2 and 5:2 compressor output which is done using Questa Sim 6.4c Mentor Graphics tool.**

Keywords— **Binary tree (4:2, 5:2), Field Programmable Array (FPGA), Digital Signal Processing (DSP), Application Specific Integration Circuits (ASCIs)**

I. INTRODUCTION

Multiplication is the key arithmetic operation which is widely used in many microprocessors and digital signal Processing applications. Microprocessors use multipliers within their arithmetic logic units, and digital signal processing systems require multipliers to implement DSP algorithms such as convolution and filtering. Since the multiplier lies directly within the critical path in most systems, the demand for high speed multiplier is continuously increasing [1]. However, with the fast growing of portable computing devices, the power consumption of the multiplier has become equally important. All this has resulted in the pursuit of high speed low power multiplier design techniques.

A multiplication process essentially consists of generating the partial product's matrix, reducing the matrix to two rows followed by the final carry propagation addition. As the main determining factor with regards to the performance characteristics of a multiplier, the most study aspect of digital multiplication is the partial product reduction circuitry [2]. Traditionally, partial product reduction has been carried out through the use of carry-save adders consisting of rows of 3:2 counters otherwise known as full-adders. In the past few years, a focus has been put on higher-order reduction scheme mainly through the use of 4:2 compressors. As an alternative to 3:2 counters, the 4:2 digital compressors were first proposed by Weinberger [3] in 1981. Similar to counter structures, digital compressors reduce a given set of inputs to a vector output. But it differs itself by introducing the notion of horizontal data paths within stages of reduction, which has transformed the standard frame of mind for counter based partial product reduction schemes into compressor based schemes [4]. There are two well-known kinds of parallel multiplication algorithms, array multiplication algorithms and Dadda multiplication algorithms. In array multiplication algorithms, cells, which consist of an AND-gate computing inner products and a counter, are put in a network pattern like an array. Dadda multiplication algorithms use AND-gates, carry save counters and a carry propagate adder (CPA) [5, 6].Dadda multiplication algorithms have a tree structure.

This paper presents binary tree architecture suited for increasing the speed multiplication and addition operation using 4:2 and 5:2 higher compressors. Multiplication using binary tree structure, cells, which consist of an AND-gate computing inner products and a counter, are put in a network pattern like a tree structure. The implementation efficiency is a result of improving overall propagation delay in the adder stage operation which mainly cause for speed up the operation as compare to array multiplier architecture

II. LOGICAL DECOMPOSITION OF BINARY TREE

A. 4:2 Compressor using Full Adder

In general, compressors reduce N-input bits to a single sum bit of equal weight to that of the inputs but unlike counters, the remaining output bits are all of equal weight: one bit position greater than that of the inputs. Although the 4:2 compressors is unlike defined as a counter, the primitive configuration of

4:2 compressor is based on a two cascaded full adder structure, which has 4 inputs and 2 outputs as shown in Figure. 1. The four inputs *X1*, *X2*, *X3* and *X4*, and the output *Sum* have the same weight. The output *Carry* is weighted one binary bit order higher. The 4:2 compressors receives an input *CIN* from the preceding module of one binary bit order lower in significance, and produce an output *Cout* to the next compressor module of higher significance as shown in figure 2. Besides, to accelerate the carry save summation of the partial products, it is imperative that the output, *Cout* be independent of the input *CIN*. Different structures of 4:2 compressors exist and they all have to abide by the fundamental equation given as follows [5]:

$$S = X_1 \oplus X_2 \oplus X_3$$

$$\text{Sum} = S \oplus X_4 \oplus C_{IN} = X_1 \oplus X_2 \oplus X_3 \oplus C_{IN} \quad (1)$$

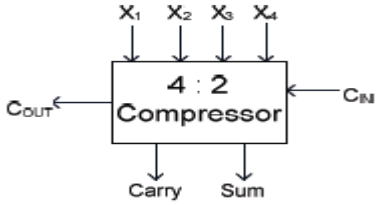

Figure 1. Symbol of 4: 2 counter

The most primitive implementation of 4:2 compressors is that of two cascaded full adders, as shown in Figure. 2. [1]. By increasing regularity, this configuration lends itself to gains at the architecture level of the multiplier. At gate level, 4:2 compressors are anatomized into XOR gates and carry generators, as shown in Figure. 3 a]. Therefore, different designs can be classified based on the critical path delay in terms of the number of primitive gates. Let ΔXOR denote the delay of an XOR gate and ΔCGEN denote the delay of a carry generator. A compressor is said to have a delay of (*m* ΔXOR + *n* ΔCGEN) if it's critical path consist of *m* XOR gates and *n* carry generators. Since the difference between the delays of widely used XOR gate and carry generator is trivial in an optimized design, the delay of the compressor is commonly specified as (m + n) Δ [5]. Therefore, the straightforward implementation of a 4:2 compressor of Figure. 2 have a long critical path delay of 4Δ.

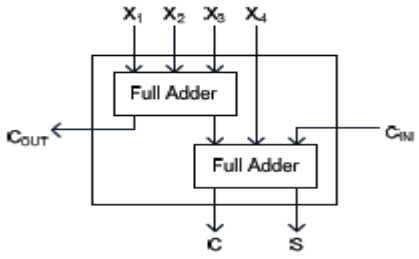

Figure 2. Architecture level representation of 4:2 compressors.

An alternative implementation of 4:2 compressors is derived from the modified equations for the functions of Figure. 2. The three outputs of the compressor are described as follows:

$$C_{OUT} = (X_1 \oplus X_2).X_3 + X_1 \bullet X_2 = (X_1 \oplus X_2) \bullet X_3 + \overline{(X_1 \oplus X_2).X_1} \quad (2)$$

$$S = X_1 \oplus X_2 \oplus X_3 \quad (3)$$

$$\text{Sum} = S \oplus X_4 \oplus C_{IN} = X_1 \oplus X_2 \oplus X_3 \oplus \text{CIN} \quad (4)$$

$$\text{Carry} = (s \oplus X_4) C_{IN} + S \bullet X_4 \quad (5)$$

$$= X_1 \oplus X_2 \oplus X_3 \oplus X_4 \bullet C_{IN} + \overline{(X_1 + X_2 + X_3 + X_4)} \bullet X_4$$

4:2 Compressor could be realized using by different combinations of X-OR Gates, AND gates and MUXs
The logic level decomposition of 4:2 compressors is as given below:

B. Binary Tree using Full adder & AND Gates

It is formed by using 3 input X-OR gates and 3 input AND gates. The critical path of compressor is 4 X-OR gates. Figure 3 b] shows RTL schematic of Primitive Implementation of 4:2 Compressor using logic gates. The objective of our work is, by using higher compressor minimize the propagation delay of the gates which is also called ripple carry propagation of the gates at the summation stage [1]

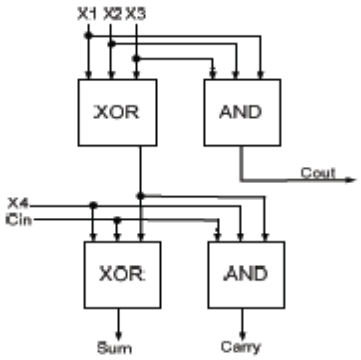

Figure 3 a) Binary Tree Implementation using logic gates.

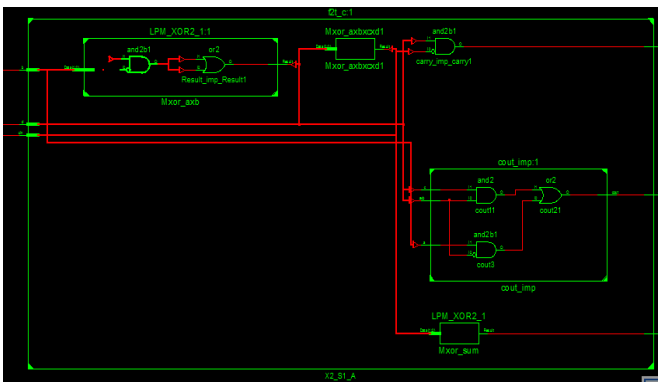

Figure 3 b) RTL schematic of Binary tree using logic gates.

III. CIRCUIT LEVEL OPTIMIZATION

Since full adders are building blocks of 4:2 compressors, we could improve the performance of 4:2 compressors by Optimizing full adder blocks. Considering a full adder which takes three equally weighted bits (X_1,X_2, X_3) and produces a sum-bit (*Sum*) as well as a carry-bit (*Carry*), outputs of the full adder can be described as follows:

$$\text{Sum} = X_1 \oplus X_2 \oplus X_3$$

$$= X_1X_2X_3 + \overline{X_1X_2}X_3 + \overline{X_1X_2X_3} + \overline{X_1}X_2\overline{X_3} \quad (6)$$

$$\text{Carry} = X_1X_2 + X_2X_3 + X_1X_3 \quad (7)$$

By taking the NOT of *Carry* signal, we obtain:

$$\overline{carry} = \overline{X_1X_2 + X_2X_3 + X_1X_3} \quad (8)$$

Comparing equation (6) with (8), we could find that *Sum* and *Carry* have some parts in common, which are X1X2, X_2X_3 and X_1X_3.Therefore, we could use part of the circuit, which is used to generate *Sum* signal, to generate *Carry* signal as well. Figure 5 a) shows block schematic of full adder with input a, b, and $C_{in\ and}$ output Sum and Carry. Figure 5 b) shows RTL schematic of full adder using logic gate.

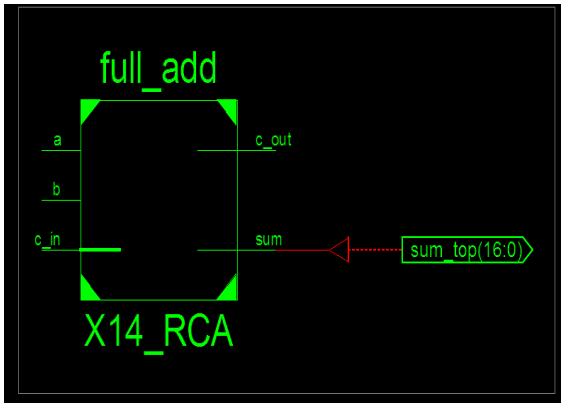

Figure 5 a) Block Schematic of Full adder (RCA).

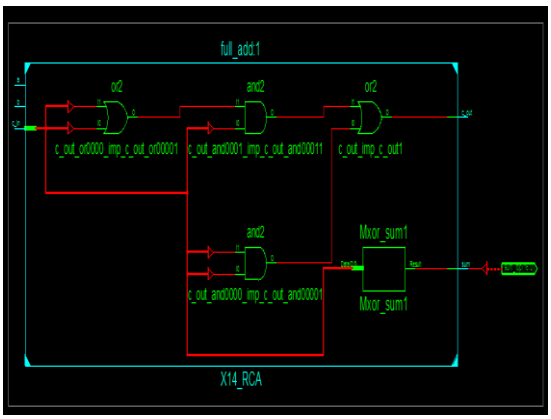

Figure 5 b) RTL schematic of Full adder using logic gates

IV. SIMULATION RESULT

The simulations are performed by using Modelsim in Mentor Graphics design tool. All the circuits are targeted for FPGA Spartan-3 (XC3S400 PQ-208). Therefore, the circuits are designed and optimized based on this process model. All the FPGA Circuitry is sized to achieve the fastest possible operation frequency as well as the proper functionality. In the test bench, each input is driven by buffered signals and each output is loaded with buffers, which offer a realistic simulation environment reflecting the compressor operation in actual applications.

The average delay is the average of delays of all input data. The worst case delay is the largest delay among all input data. The designed 4:2 compressor is implemented using gate logic and circuits are simulated. The simulation results' using implemented technique is as shown in Table. I. As shown in tables, all the compressor circuits are simulated in terms of delay and power consumption .The operation frequency is maximum working frequency for each circuit, which is 95.767 MHz Simulation result indicates that 4:2 compressor designed in gate has the best performance. Simulation result indicates that 4:2 compressors designed in using gate logic, it reduces the average delay and worst case delay by 44.6% logic and 55.4% route.

TABLE I
SIMULATION RESULTS FOR 4:2 BINARY TREE.

Cell Name	Average Delay	Gate Delay	Net Delay	Operation Frequency (MHz)
Compressor f2c [Fig.3 a]	10.42 ns	4.659 ns	5.783 ns	95.767 MHz

The primitive 4:2 compressor using gate logic is implemented on a Field Programmable Gate Array (FPGA).Two major CAD software tools were used; Mentor Graphics and Xilinx 11.1 ISE tools. ModelSim is used for simulation; Fig.6 a) shows the simulation result of primitive 4:2 compressor using gate logic. The simulation result shows output of the proposed design with input sequence (a=0, b=0, c=1, d=1, and C_{in}=1) and output will give 4:2 compressed output (C_{out} =0, and Sum=1) as shown in Fig 6a).Figure 6 b) shows data flow trough 4:2 compressor.

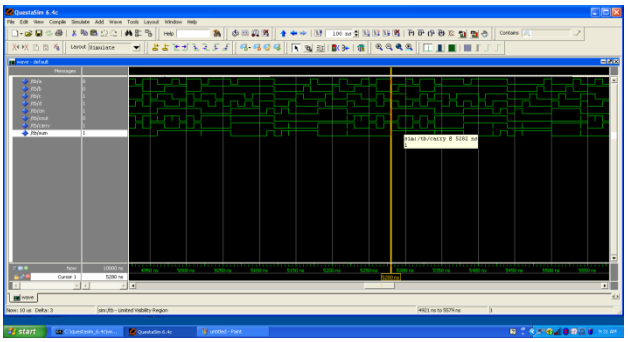

Figure 6. a) Simulation result of 4:2 compressors.

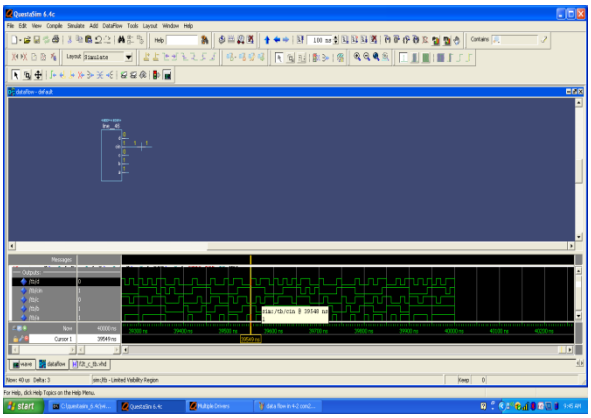

Figure 6 b) Data Flow trough 4:2 compressors.

V. CONCLUSION

This paper has described various designs of 4:2 compressors at both logic level and circuit level. The 4:2 Compressor is implemented in domino logic and followed by the simulation results of these circuits. The 4:2 compressors, which are designed in mux gate domino logic, have the best performance in terms of delay, and operation frequency compared with the gate logic designs. By exploiting this architecture using higher compressor for computation addition operation, the constraints on the coefficient values, this architecture yields extremely efficient and high speed programmable and custom implementations.

REFERENCES

[1] K. Prasad, and K.K. Parhi, “Low-power 4-2 and 5-2 compressors,” in Proc. of the 35th Asilomar Conference on Signals, Systems and Computers, vol.I, pp. 129-133, 2001.

[2] Peng Chang Majid Ahmadi, “High speed low power 4:2 compressor cell design,”

[3] A. Weinberger, "4:2 carry-save adder module," IBM Technical Disclosure Bulletin. vol.23. Jan.1981

[4] A. Abedelgwad, “High speed and area efficient multiply Accumulate (MAC) Unit for Digital signal Processing applications,” *IEEE International Symposium on Circuits and Systems*, ISCAS 2007.

[5] Ayaman.A.Fayed, “A merged Multiplier Accumulator for High Speed signal processing Applications,” *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2002.

[6] P.asadee, “A High-Speed Multiplication Algorithm Using Modified Partial Product Reduction Tree.

[8] Wayne Wolf, "Modern VLSI Designer", Prentice-Hall, Upper Saddle River, 2002.

[9] Douglas Perry, “VHDL programming by examples” McGraw-Hill Publication.