International Journal of Current Engineering and Technology
E-ISSN 2277 – 4106, P-ISSN 2347 – 5161

Available at http://inpressco.com/category/ijcet

Research Article

Wallace Tree Multiplier using Compressor

Jinimol P George†* and Ramesh P†

†Department of ECE, Cochin University for Science and Technology, Kerala, India

Accepted 31 May 2015, Available online 05 June 2015, **Vol.5, No.3 (June 2015)**

Abstract

Multiplier is an important block in most of digital and high performance systems So the performance of such system can be improved by implementing high speed multiplier. Varieties of multipliers are available, in that a fast multiplier based on booth encoded Wallace tree is discussed in this research. The conventional Wallace tree multiplier is based on carry save adder. Here the speed of the multiplier is improved by introducing compressors instead of the carry save adder. 3-2 compressor, 4-2 compressor, 5-2 compressors and 7-2 compressors are used with Wallace tree multiplier. Higher order compressors have better performance compared with 3-2 compressor. So the speed of the multiplier can be improved by introducing the higher order compressors. The coding is done on Verilog HDL and synthesis is done by using Xilinx ISE 14.7. Further analysis is done by using Cadence Encounter tool. Various design parameters like delay, area, power of Wallace booth multiplier with various compressors and different radix are analyzed.

Keywords: *Booth multiplier, Wallace tee, Compressor, Radix.*

1. Introduction

With the rapid advances in multimedia and communication systems, real-time signal processing and large capacity data processing are increasingly being demanded. Multiplication is a one of the most important operation in digital signal processing and DSP system. In high performance systems such as microprocessor, DSP etc addition and multiplication of two binary numbers is fundamental and most often used arithmetic operations. As they are basically accomplished by repetitive application of multiplication and addition, their speed becomes a major factor. The speed performance of multiplication influents the overall performance of digital signal processing as well as in digital computer systems. Many high performance algorithms and architectures have been proposed to improve and accelerate multiplication operations. The demand of high speed processing has been increasing as a result of expanding computer and signal processing applications. Since the multiplier requires the longest delay among the basic operational blocks in digital system, the critical path is determined more by the multiplier.

Low power consumption is also an important issue in multiplier design. To reduce significant power consumption it is good to reduce the number of operation thereby reducing dynamic power which is a major part of total power consumption. So the need of high speed and low power multiplier has increased. Designer mainly concentrates on high speed and low power efficient circuit design. The objective of a good multiplier is to provide a physically packed together, high speed and low power consumption unit. Furthermore, multiplier consumes much area and dissipates more power. Hence designing multipliers which offer either of the following design targets – high speed, low power consumption, less area or even a combination of them. However, the fact remains that the area and speed are two conflicting performance constraints. Hence, innovating increased speed always results in larger area. In this project, a better trade-off between the two is achieved, by realizing a marginally increased speed performance through a small rise in the number of transistors.

The multiplication operation involves generation of partial products and their accumulation. The speed of multiplication can be increased by reducing the number of partial products and/or accelerating the accumulation of partial products. Among the many methods of implementing high speed parallel multipliers, there are two basic approaches namely Booth algorithm and Wallace Tree compressors. The use of booth multipliers is attractive depending upon the bit pattern of the operands. When implemented, choice of radix of encoding determines the overall performance of these multipliers for different operand lengths. In parallel booth multiplier architectures, the choice of adder structure for adding effects area delay and power dissipation. Ripple carry adder optimizes

*Corresponding author: **Jinimol P George**

area at the expense of speed while carry save adders trade-off area for speed. Tree adder structures based on carry save addition provide regularity of layout for path-delay balancing, which minimizes glitches. In high-speed designs the Wallace tree method is usually used to add the partial products. In this method all the bits in each column are selected at a time and compress them into two or three bits. Adders and compressors can be used to vertical bits compression. An adder itself is a compressor, it can compress three bits into two bits. Hence it is a 3:2 compressor. For higher order multiplication, higher order compressors can be used to compress the bits.

The rest of the paper is organized as follows: In section 2 ,3, 4, and 5 the architectures of 3-2, 4-2, 5-2 and 7:2 compressors are presented. The section 6 is deals with Wallace tree multiplier.

2. 3-2 Multiplier

A 3-2 compressor takes 3 inputs A, B, C and generates 2 outputs, the sum bit, and the carry bit. The 3-2 compressor can also be employed as a full adder cell when the third input is considered as the carry input from the previous compressor block or C = C_{in}.

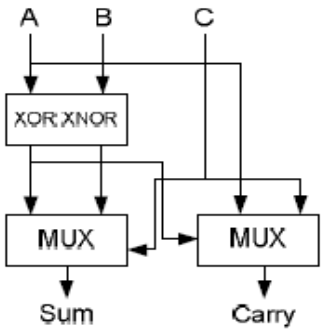

Fig.1 A 3-2 compressor

The architecture of the 3-2 compressor is shown in Fig.1. The equations governing the 3-2 compressor outputs are shown below:

$$Sum = (A \oplus B) \bullet \sim C + \sim(A \oplus B) \bullet C \quad (1)$$

$$Carry = (A \oplus B) \bullet \sim C + \sim(A \oplus B) \bullet A \quad (2)$$

It can be seen that in this implementation the overall delay is Δ-XOR +Δ-MUX (where Δ refers to delay).

3. 4-2 Multiplier

The 4-2 compressor has 4 inputs X1, X2, X3 and X4 and 2 outputs Sum and Carry along with a Carry-in (Cin) and a Carry-out (Cout) as shown in Fig.2. The input Cin is the output from the previous lower significant compressor. The Cout is the output to the compressor in the next significant stage.

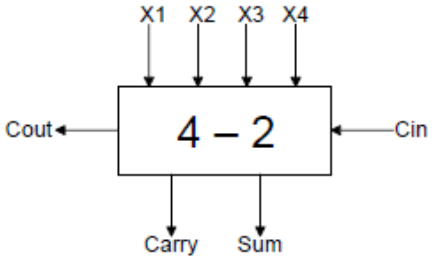

Fig.2 A 4-2 compressor block

The standard implementation of the 4-2 compressor is done using 2 Full Adder cells as shown in Fig. 3.

When the individual full Adders are broken into their constituent XOR blocks. And replacing some XOR blocks with multiplexers results in a significant improvement in delay. The 4-2 compressor architecture is shown in Fig.4. The critical path delay of the proposed implementation is Δ-XOR + 2*Δ-MUX.

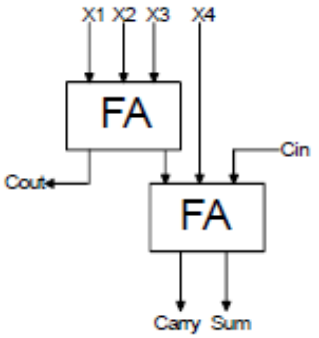

Fig.3 A 4-2 compressor implementation with full adders.

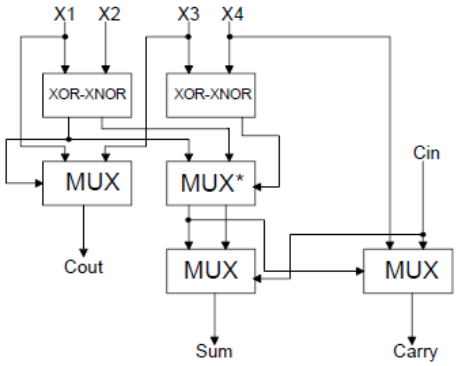

Fig.4 A 4-2 compressor

4. 5-2 Compressor

The 5-2 Compressor block has 5 inputs X1,X2,X3,X4,X5 and 2 outputs, Sum and Carry, along with 2 input carry bits (Cin1, Cin2) and 2 output carry bits (Cout1,Cout2) as shown in Fig.8a. The input carry bits are the outputs from the previous lesser significant compressor block and the output carry are passed on to the next higher significant compressor block.

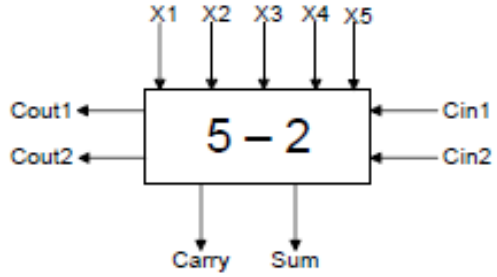

Fig.5 A 5-2 compressor block

The architecture and the basic equation that governs the function of the 5-2 compressor block is given below

$$X1 + X2 + X3 + X4 + X5 + Cin1 + Cin2 = Sum + 2 * (Carry + Cout1 + Cout2$$

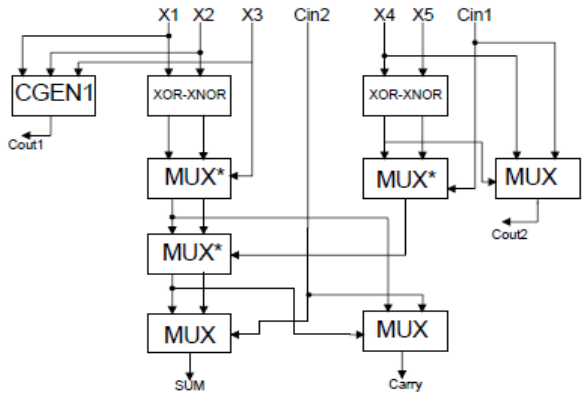

Fig.6 A 5-2 compressor

5. 7-2 Compressor

The 7-2 compressor reduces the 7 input bits to two outputs. The architecture of a 7-2 multiplier is shown below **Fig.7**.

In the **Fig.7**,

$$C1 = (x3 + x4) \bullet x2 + x3 \bullet x4$$
$$C2 = (X6 + X7) \bullet X5 + X6 \bullet X7$$
$$C3 = (S4 + X1) \bullet S3 + S4 \bullet X1$$

The 7-2 compressor provides the parallel addition of input bits. Hence speed of the addition operation can be improved.

6. Wallace tree Multiplier

A Wallace tree multiplier is an efficient hardware implementation of a digital circuit that multiplies two integers devised by an Australian computer scientist Chris Wallace in 1964. Conventional Wallace tree reduces the no. of partial products and use carry select adder for the addition of partial products.

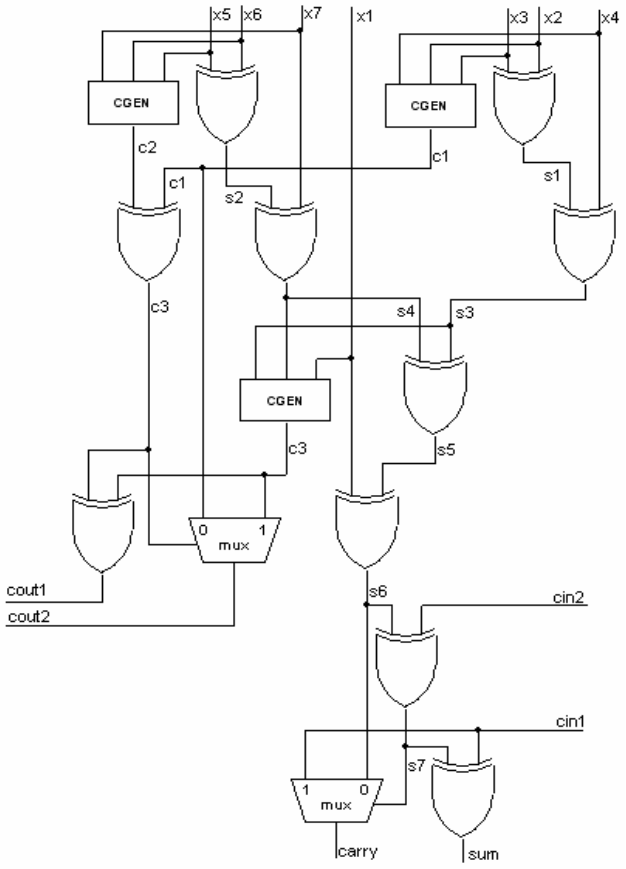

Fig.7 Structure of a 7-2 compressor

Wallace tree has three steps

1. Multiply each bit of multiplier with same bit position of multiplicand. Depending on the position of the multiplier bits generated partial products have different weights.
2. Reduce the number of partial products to two by using layers of full and half adders.
3. After second step we get two rows of sum and carry, add these rows with conventional adders.

Explanation of second step

As long as there are three or more rows with the same weight add a following layer:

- Take any three rows with the same weights and input them into a full adder. The result will be an output row of the same weight i.e sum and an output row with a higher weight for each three input wires i.e carry.
- If there are two rows of the same weight left, input them into a half adder.
- If there is just one row left, connect it to the next layer.

The advantage of the Wallace tree is that there are only O(log n) reduction layers (levels), and each layer has O(1) propagation delay. As making the partial products is O(1) and the final addition is O(log n), the multiplication is only O(log n), not much slower than addition (however, much more expensive in the gate count). For adding partial products with regular adders would require O(log n2) time.

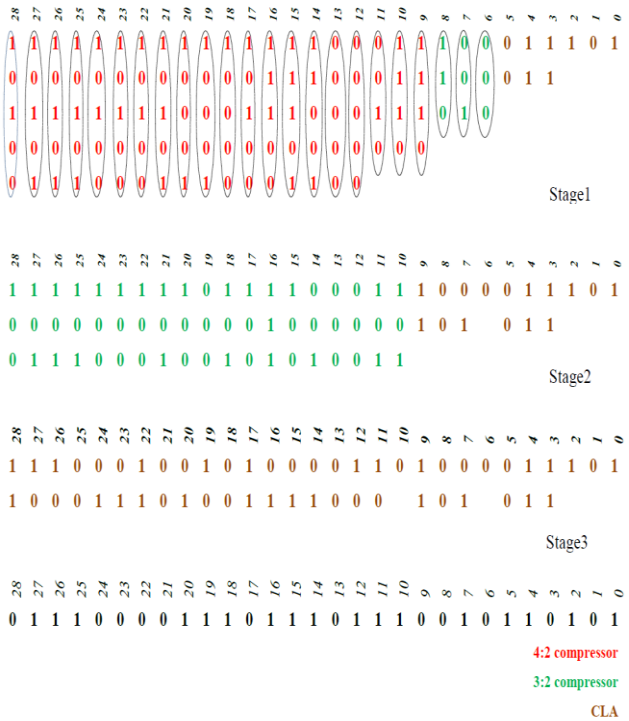

Fig.8 Example for the application of 3-2 and 4-2 compressor in a Wallace tree

7. Result

The main block of a conventional Wallace tree multiplier is the tree structure of carry save adder. Carry save adder is the combination of full adder and half adder. Here the carry save adder is replaced with the tree structure of appropriate compressor. The higher order compressors have better performance.

7.1 Simulation Result of Compressors

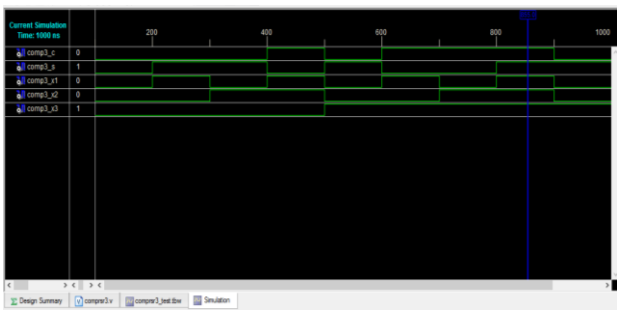

Fig.9 Simulation result of 3-2 multiplier

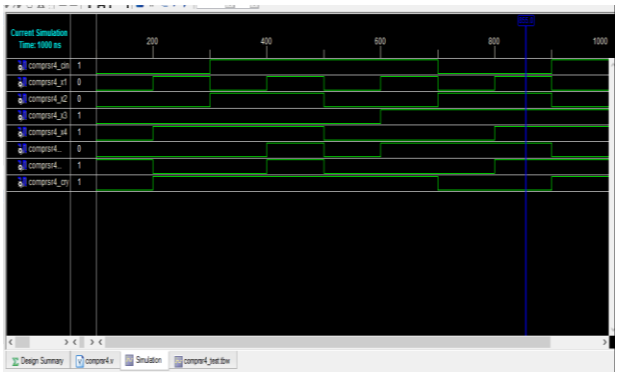

Fig.10 Simulation result of 4-2 multiplier

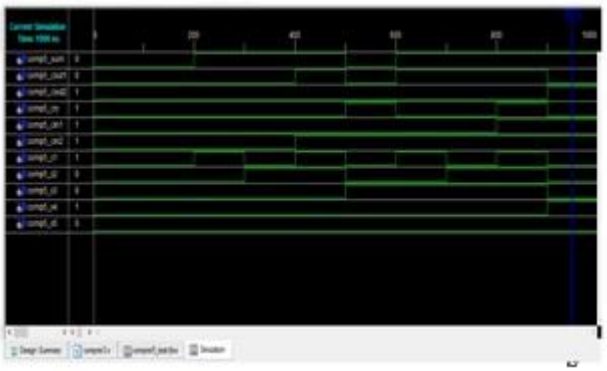

Fig.11 Simulation result of 5-2 multiplier

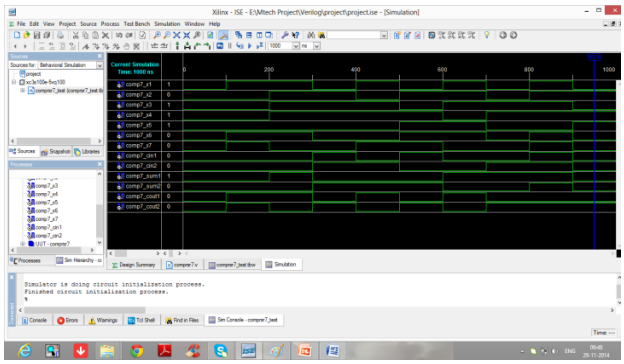

Fig.12 Simulation result of 7-2 multiplier

Table 1 Comparison of compressors

	Delay (ps)	Cell Area	Power (nW)
3:2 Compressor	283	17.64	343.58
4:2 Compressor	454.5	35.28	1052.25
5:2 Compressor	565.5	54.33	1703.65
7:2 Compressor	767.3	65.4	2010.33

The table shows the delay, cell area and power dissipation for each compressor. A simple 4-2 compressor requires two 3-2 compressor. So in conventional method 4-2 compressor need 566ps but the simple 4-2 compressor having only 454.5ps. So by introducing higher order compressor we can improve the performance of the Wallace tree multiplier.

Conclusions

In this research an efficient implementation of a high speed parallel multiplier using both these approaches is described. Here multipliers are designed by using the Radix-8 Booth Algorithm and the Radix-32 Booth algorithm with various compressors [15]. The number of partial products is n/3 or [n/3+1] in Radix-8 Booth algorithm while it gets reduced to n/5 or [n/5+1] in Radix-32 Booth algorithm, where n is the bit-width of the multiplier. This reduces the number of partial products that are to be added there by increasing the speed of operation. The Wallace tree uses Carry Save Adders (CSA) to accumulate the partial products. This reduces the time as well as the chip area. To further enhance the speed of operation, carry-look-ahead (CLA) adder is used as the final adder.

References

P. Aparna and N. Thomas (2012), Design and implementation of a high performance multiplier using hdl, *Computing, Communication and Applications (ICCCA),* 2012 International Conference on. IEEE, pp. 1–5.

M. Bansal, S. Nakhate, and A. Somkuwar (2011), High performance pipelined signed 64x64- bit multiplier using radix-32 modified booth algorithm and wallace structure, *Computational Intelligence and Communication Networks (CICN),* 2011 International Conference on. IEEE, pp. 411–415.

D. Das and H. Rahaman (2010), A novel signed array multiplier, *Advances in Computer Engineering (ACE),* 2010 International Conference on. IEEE, pp. 19–2

N. K. Gahlan, P. Shukla and J. Kaur (2012), Implementation of wallace tree multiplier using compressor, *International Journal of Computer Technology & Applications*, vol. 3, no.

A. Habibi and P. A.Wintz (1970), Fast multipliers, *IEEE Transactions on Computers*, vol. 19, no. 2, pp. 153–157

R. Hussin, A. Y. M. Shakaff, N. Idris, Z. Sauli, R. C. Ismail, and A. Kamarudin (2008), An efficient modified booth multiplier architecture, *Electronic Design*, ICED 2008 International Conference on. IEEE, pp. 1–4

M. Jagadeshwar Rao and S. Dubey, A high speed wallace tree multiplier using modified booth algorithm for fast arithmetic circuits, *IOSR Journal of Electronics and Communication Engineering (IOSRJECE),* vol. 3, pp. 07–11

K. G. Krishna, B. Santhosh, and V. Sridhar, Design of wallace tree multiplier using compressors

B. Likhar et al. (2013), Design and comparison of regularize modified booth multiplier using different adders, *Machine Intelligence and Research Advancement (ICMIRA),* International Conference on. IEEE, pp. 387–391

N. V. Maunika and M. V. Devi, A dwindled power and delay of Wallace tree multiplier, *International Journal of Engineering and Innovative Technology* (IJEIT) Volume, vol.

M. Othman, M. A. M. Ali et al. (2002), High performance parallel multiplier using Wallace booth algorithm, *Semiconductor Electronics*, ICSE 2002, IEEE International Conference on. IEEE, 2002, pp. 433–436.