Techniques to Optimize 32 Bit Wallace Tree Multiplier

A. Radhika
M.Tech., (Ph.D)
Department of Electronics and Communication Engineering
Anurag College of Engineering, Ghatkesar, Telangana, India

D. Nandini
B.Tech Student
Department of Electronics and Communication Engineering
Anurag College of Engineering, Ghatkesar, Telangana, India

M.Harish
B.Tech Student
Department of Electronics and Communication Engineering
Anurag College of Engineering, Ghatkesar, Telangana, India

T.Sri Sadhana
B.Tech Student
Department of Electronics and Communication Engineering
Anurag College of Engineering, Ghatkesar, Telangana, India

Abstract- Multipliers play an important role in today's digital signal processing, micro processors and various other applications. The process used by multiplier is Shift-add algorithm. The three major parameters that are to be considered while designing a multiplier using VLSI design are speed, area and power consumption. Wallace tree multiplier is a combinational circuit used to multiply two numbers. Instead of performing additions using standard parallel adders, Wallace tree uses carry save adders and only one parallel adder. So, it produces a product in far less time. Advantage of a Wallace tree multiplier is, it requires less power for bit widths between 8 and 32. This paper discusses two techniques to optimize multiplication of two 32 bit numbers in terms of speed and area. In the first technique, instead of using parallel adder, carry look ahead adder is used. In the second technique, instead of using 64-bit CSAs, CSAs are built according to the number of shifts in the partial products. Therefore, the total requirement of hardware decreases making this Wallace Tree multiplier, area efficient. The entire design is coded in Verilog HDL, simulated with ModelSim and synthesized using Xilinx Basys 3 device. The result shows that the proposed design takes very less time and hardware for multiplication of two 32 bit numbers.

Keywords – Carry save adder, Basys 3 board, ModelSim simulator, Wallace tree multiplier

I.INTRODUCTION

The basic arithmetic operation which is widely used during computation is multiplication. Most of the arithmetic operations require the use of multiplication and it plays a very important role in today's digital signal processing. Many DSP applications demand high throughput and real-time response, performance constraints that often dictate unique architectures with high levels of concurrency. The present development in the processors aims at high speed. So the requirement of high speed multipliers increased. Therefore, a designer has to concentrate more on high speed circuit design. Generally, the performance of any DSP processor is affected by the type of multiplier used.

Carry Save Adder Concept:

A carry save adder can add three numbers instead of just two numbers. But it doesn't outputs a single value, instead gives two, a sum and a carry. A 1-bit carry save adder can be considered as a 1-bit full adder with the signals renamed as shown in figure 1. The Cin of full adder is renamed as Z. The original answer Z is renamed as S and Cout as C.

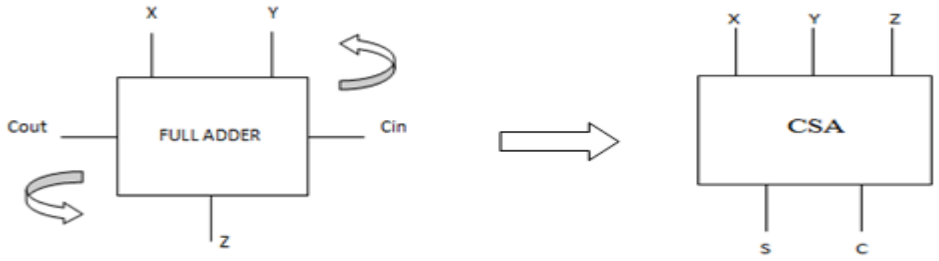

Figure 1: The carry save adder block resembles full adder circuit

The computation of sum s and carry c is as shown in the figure 2.

```
X =     10101010
Y =     10101010
Z =     10101010
-------------------
S =     10101010
C =    10101010
-------------------
SUM =  111111110
-------------------
```

Figure 2: Example

II. PROPOSED METHODOLOGY

2.1 Wallace tree multipication algorithm: The use of carry save adder, to perform multiplication, first calculates the partial products of the multiplication, and then input them to the carry save adder. For example, consider the mutiplication of two 8-bit binary values. It generates eight partial products as shown in the Figure 3.

```
X =        10101010
Y =        11111111
          10101010   ------> PP0
         10101010    ------> PP1
        10101010     ------> PP2
       10101010      ------> PP3
      10101010       ------> PP4
     10101010        ------> PP5
    10101010         ------> PP6
   10101010          ------> PP7
Z =  1010100101010110
```

Figure 3: Generation of Partial Products

2.2 Existing System: In the existing system, a ripple carry adder (parallel adder) is used to add the output of the final CSA, the sum and carry. The main disadvantage of this system is, propagation delay is more since the carry bit takes more time to ripple through the remaining full adders.

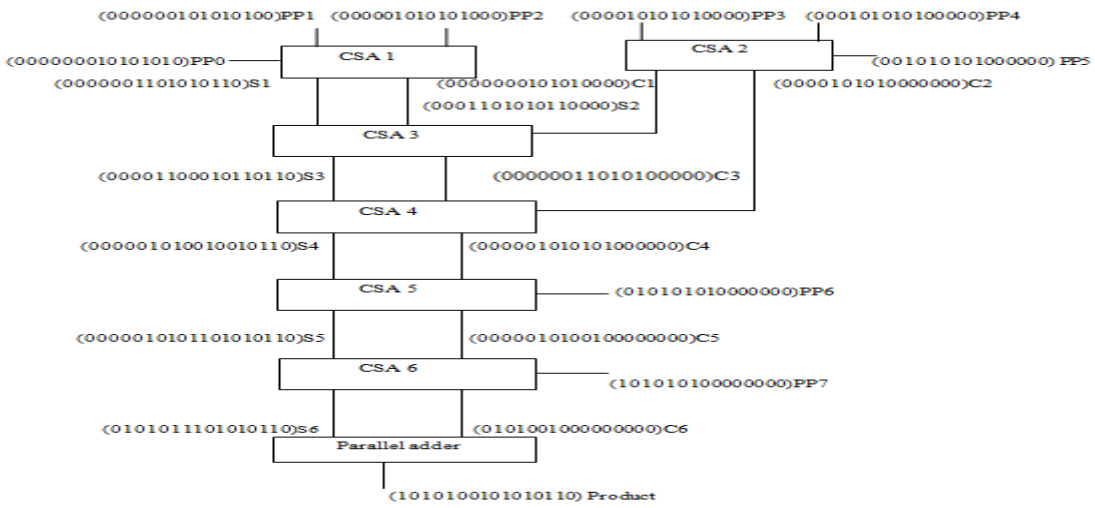

Figure 4: Existing Technique

2.3 Proposed Techniques:

2.3.1 Procedure:

Consider two 32 bit numbers. Partial products (P0-P31) are obtained for the numbers. Carry save adders are designed. The obtained 32 partial products are divided into groups, each containing three partial products. Carry look ahead adder is designed and the result is obtained.

2.3.2 Technique 1:

In this technique, the parallel adder is replaced by carry look ahead adder as shown in the figure. The advantage of using a carry look ahead adder is, the propagation delay is less compared to parallel adder since it doesn't wait for the previous adder's carry.

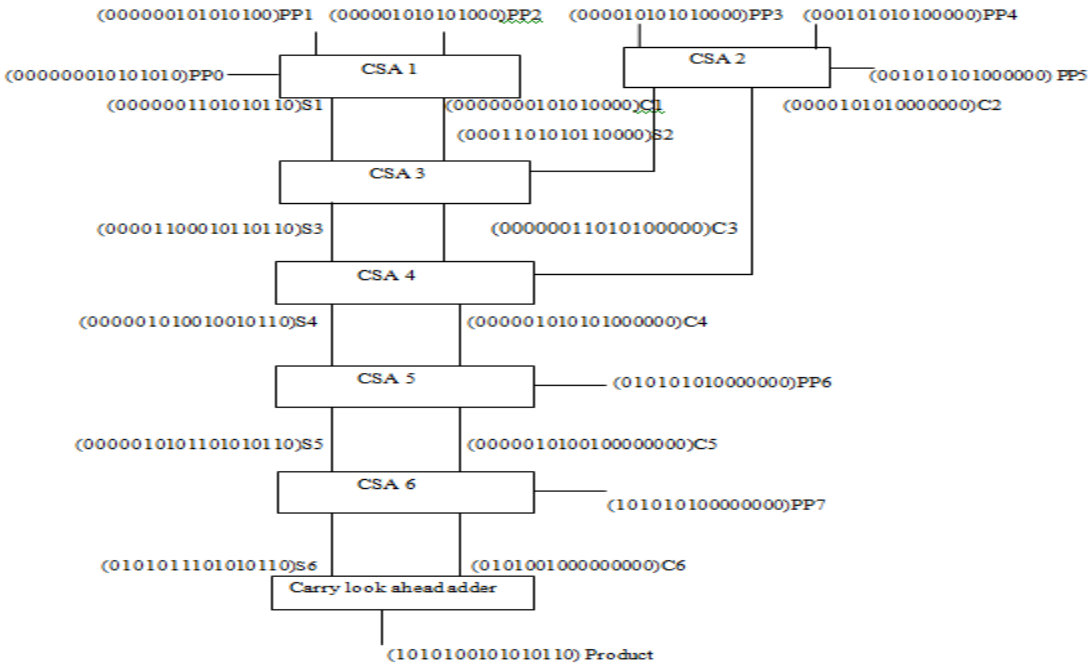

Figure 5: Technique 1

2.3.3 Technique 2:

In this technique, the size of the CSA is chosen according to the number of shifts in the partial products as shown in the figure below.

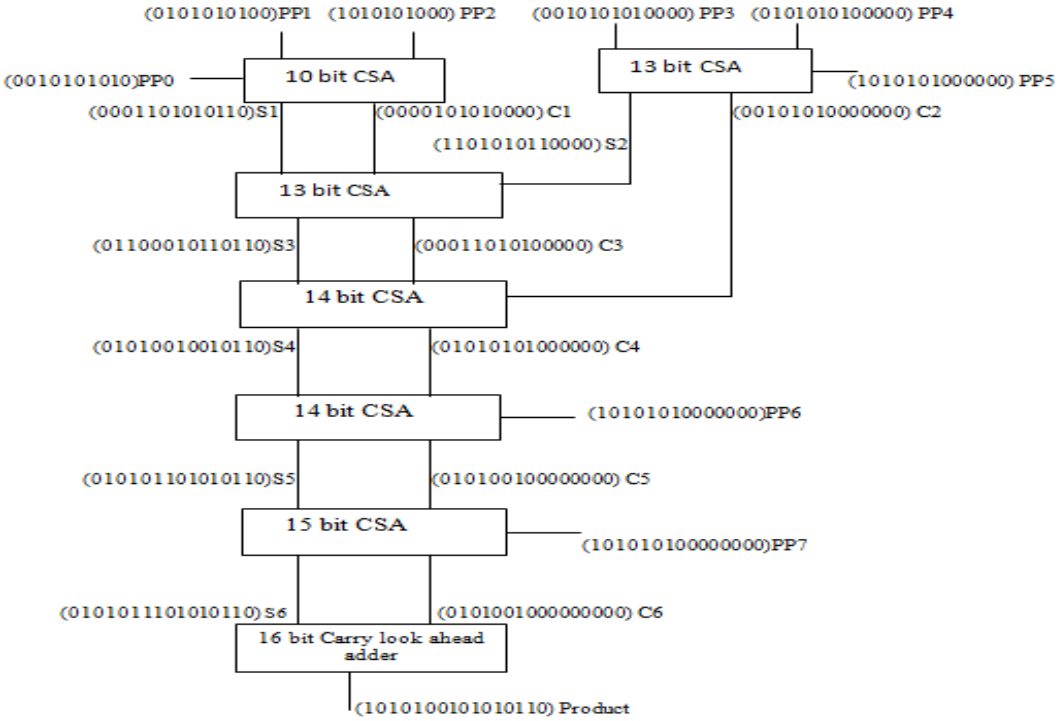

Figure 6: Technique 2

III EXPERIMENT AND RESULT

The proposed system uses carry save adders with their width proportional to the number of bits which are given as inputs to them instead of using all 64 bit CSAs. Even though technique 1 has the same delay as the existing system, the LUTs used are reduced. In the 2nd technique, both the delay and LUTs used are reduced.

Table 1: Area Comparison

S.No	Method	Number of bits	LUTs used
1	Existing method	32	2044
2	Technique 1	32	1674
3	Technique 2	32	1639

Table 2: Delay Comparison

S.No	Method	Number of bits	Delay (ns)
1	Existing method	32	5ns
2	Technique 1	32	5ns
3	Technique 2	32	4ns

Device Utilization Summary (estimated values)			[-]
Logic Utilization	**Used**	**Available**	**Utilization**
Number of Slices	1674	4656	35%
Number of 4 input LUTs	2929	9312	31%
Number of bonded IOBs	129	232	55%
Number of GCLKs	1	24	4%

Figure 7: Device Utilization Summary for Technique 1

Device Utilization Summary (estimated values)			[-]
Logic Utilization	**Used**	**Available**	**Utilization**
Number of Slices	1639	4656	35%
Number of 4 input LUTs	2864	9312	30%
Number of bonded IOBs	129	232	55%
Number of GCLKs	1	24	4%

Figure 8: Device Utilization Summary for Technique 2

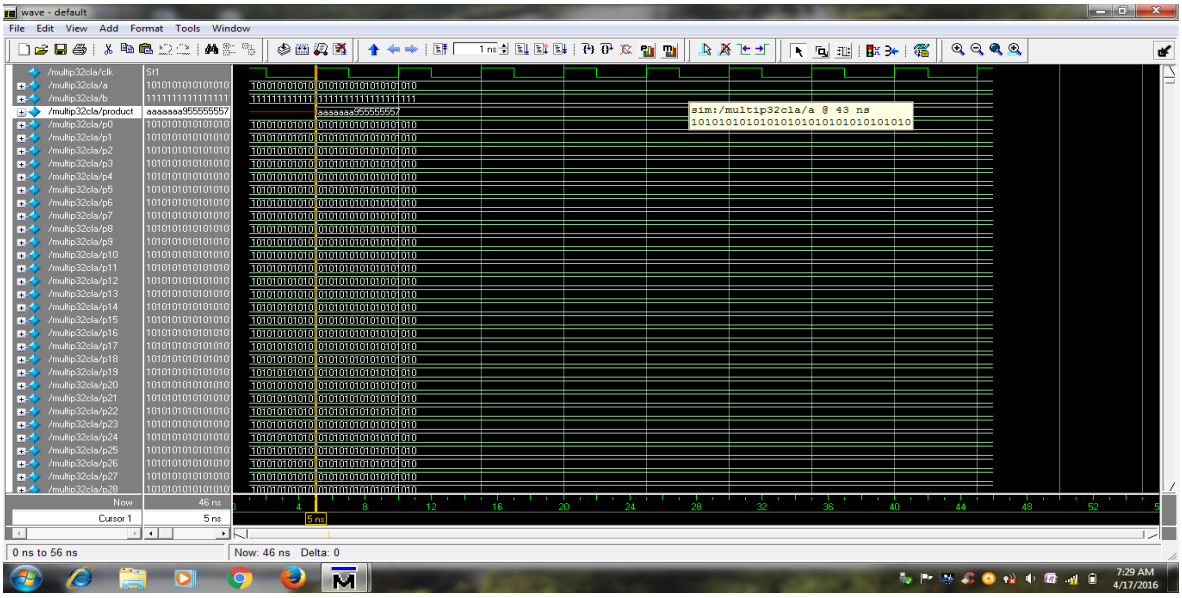

Figure 9: Simulation result for Technique 1

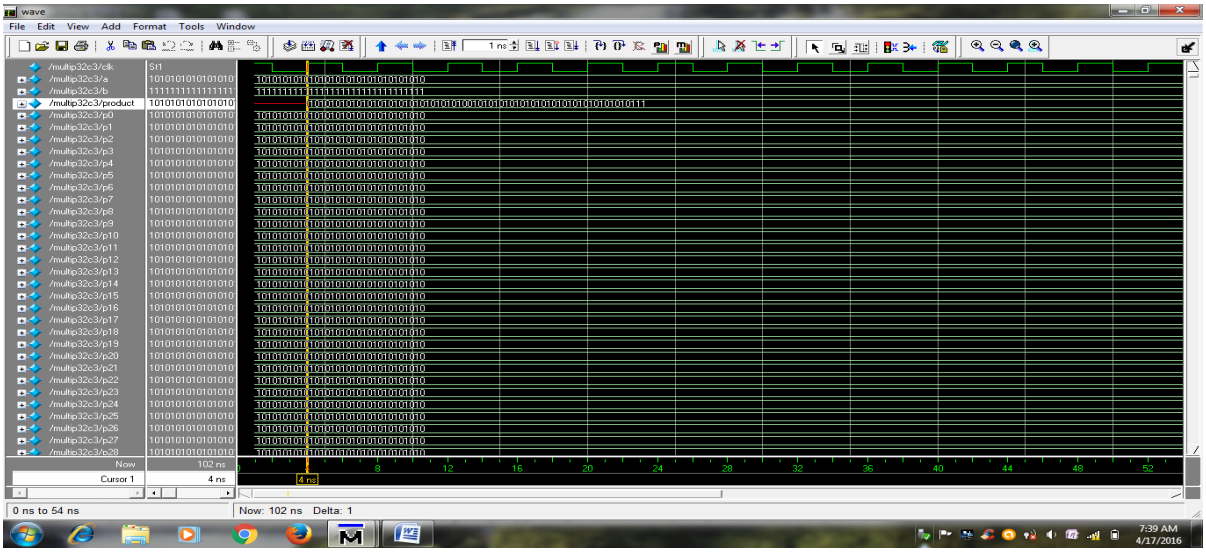

Figure 10: Simuation result for Technique 2

REFERENCES

[1] Keshaveni .N, "High speed area efficient 32 bit wallace tree multiplier", International Journal of Computer Applications (0975-8887), vol. 124- No 13, August 2015.

[2] Prof Loh, CS3220, Processor design, Spring 2005, February 2, 2005.

[3] Kartikeya Bhardwaj, Praveen S, IEEE 2014, Power and area efficient approximate wallace tree multiplier for error resilient systems.

[4] Chepuri Satish, Panem Charan Arur, G.Kishore and G.Mamatha, An efficient high speed wallace tree multiplier, International Journal of Emerging Trends in Electrical and Electronics (IJETEE – ISSN:2320-9569), vol 10, Issue 4, May 2014.

[5] Vijaya Chandra Kurapati, Analysis of IP based implementations of adders and multipliers in submicron and deep submicron technologies, JNTU, Hyderabad, 2006.

[6] Sutherland "Array Multiplier" second pages 1999/2/24.

[7] Simulation and Comparative Analysis of different types of multipliers, IJAICT vol 1, Issue 7, Novemeber 2014

[8] Basys 3 board reference, www.digilentinc.com