FPGA Implementation of Wallace Tree Multiplier using CSLA / CLA

Shruti Dixit[1], Praveen Kumar Pandey[2]

[1]Suresh Gyan Vihar University, Mahaljagtapura, Jaipur, Rajasthan, India
[2]Suresh Gyan Vihar University, Mahaljagtapura, Jaipur, Rajasthan, India

Abstract: ***This paper discusses two different architectures of Wallace Tree Multiplier and analyses which one is better in respect of area and power consumption in order to fortify the multiplier design. In one of the architecture the final addition is performed using Carry look-ahead adder whereas in the other architecture the final addition is performed using Carry select adder. The proposed design in this paper is implemented on FPGA Spartan 3e and simulation tools used are Xilinx and Modelsim.***

Keywords: Wallace tree multiplier, CSLA, CSA, Xilinx, FPGA

1. Introduction

The thirst of taking the technology to more and more heights has inculcated numerous techniques and designs which are very fast and compact than the previous technology. As the VLSI field is moving towards more compact design with higher performance, the need to develop the components which are compatible with the advancement becomes mandatory. Multiplier is the logic device of great concern in terms of performance of a processor. In any system, the task of processor is very crucial. The task that took most of the processor's time is multiplication thus enhancing the performance of multiplier leads to better performance of processor especially in field of digital signal processing and data processing ASIC. This paper work presents two different form of Wallace tree multiplier using two different adder circuits namely carry look-ahead adder and carry select adder. After developing these two different forms of Wallace tree multiplier a comparative study is being carried out on the basis of area and power consumption by the two designs. As the component size on chip is reducing to offer mobility, the dissipation of power becomes a matter of concern. Multiplication at its basic can be defined as the process of adding a number to itself a number of times. Thus it can be interpreted from this fact that addition is a sub-process in multiplication criterion that has to be fulfilled. In the process the Wallace tree formation aligned the partial products in form of a tree and then with the help of fast adders final product is obtained. After the architecture design and comparison the final step is FPGA implementation

2. Adder

Adder can be defined as the functional block used for addition of numbers. The adders which will be discussed in this paper are binary adders which do the addition of binary numbers only. The basic adders are: half adder and full adder. With the help of these basic adders more complex adders are designed. The two fast adders used in this paper work are:

1) Carry Look-Ahead Adder.
2) Carry Select Adder.

2.1 Carry Look-Ahead Adder

Reduction in the computation time of the carry signal is the key to enhance the speed of the adder. Therefore, a phenomenon called 'Carry look-ahead' is employed to generate and propagate the carry signal fast. In Carry Look-ahead approach the next stage carry signal can be computed without waiting for the carry of the previous stage. This can be accomplished with the use of intermediate signals generate and propagate signal. The carry of any stage can be computed using these two signals and initial carry-in signal. The equations are as follows:

$C_{out} = x.y + y.c_{in} + C_{in}.x;$
$C_{out}= x.y + (x+y)\ C_{in};$
$C_{out} = G + P.C_{in};$

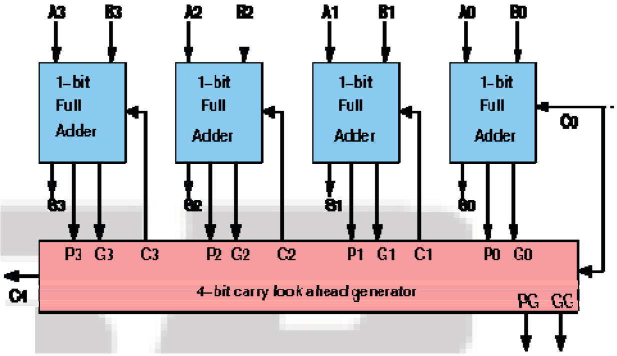


Figure 1: Hierarchical Carry Look-ahead Adder

2.2 Carry Select Adder

The practice of addition with carry select addition approach conventionally divides the adder into blocks. If the adder is an n-bit adder and we want to divide it into blocks having m-bits each then obviously the numbers of blocks thus formed are n/m blocks. These n/m blocks performs addition of m-bits using any of the adder algorithm such as ripple carry adder, carry skip adder, carry look-ahead adder etc. to produce the output sum and carry signals. Now the n/m blocks in the CSLA may be viewed as n/m stages consists of two m-bits adder and one m-bit 2-to-1 multiplexer each except the block in the initial stage contains the least significant bits of the two numbers to be added and performs addition with the use of single adder only.

The remaining blocks perform the addition two times, one with carry-in equals to 1 and other with the carry-in equals to 0. Then these pre-calculated values of sum bits with their respective pre-assumed carry-in values are stored in the block and when the true carry-in value becomes available, the correct pair is extracted. Now the task of the selection of the proper pair is achieved with the help of 2-to-1 MUX. The actual carrying signal acts as the select line of the MUX.

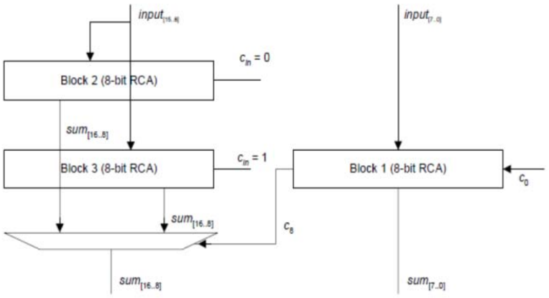


Figure 2: 16-bit CSLA with 8-bit RCA Blocks

3. Multiplier

Multiplier is the functional device which is used to multiply two numbers. In the process of multiplication each bit of multiplier is multiplied with each bit of multiplicand and partial products are obtained. Then these partial products are added to get the final product. There are mainly two types of multipliers: array multiplier and tree multiplier. In this paper one of the tree multiplier is discussed known as Wallace tree multiplier.

3.1 Wallace Tree Multiplier

This architecture helps in reducing the partial products with a rate of log3/2 N/2. The three main steps in the algorithm of Wallace tree multiplier design are:

1) The multiplication of the multiplier bits with the multiplicand generates a bit product stream.
2) The bit product matrix thus formed has been reduced into less number of rows with the help of half and full adders, this step persist till the last addition is done.
3) Last and final step is the final addition using adders and the final result can be obtained after this step.

R15	R14	R13	R12	R8	R4	R0
(b3a3)	(b2a3)	(b1a3)	(b0a3)	(b0a2)	(b0a1)	(b0a0)
	R11	R10	R9	R5	R1	
	(b3a2)	(b2a2)	(b1a2)	(b1a1)	(b1a0)	
		R7	R6	R2		
		(b3a1)	(b2a1)	(b2a0)		
			R3			
			(b3a0)			

Figure 3: Wallace Tree Formation with Partial Products

4. Proposed Methodology

The work presented in this dissertation report proposed the design of Wallace tree multiplier with two different architectures. In one of the architectures the Wallace tree is used with the carry select adder and in the other one it is implemented in combination of carry look-ahead adder. Then a comparative study is also shown in respect of area and power consumption. The final objective of the thesis is the hardware implementation of the proposed design. The hardware implementation is achieved on the FPGA Spartan 3e device with Xilinx atmosphere and the programming in VHDL is developed in MODELSIM.

The three main steps followed in the approach suggested in the thesis are:

- Generating the partial products.
- Wallace tree implementation using the partial products with half and full adders.
- Final summation using Carry select adder (CSLA) and Carry look-ahead adder (CLA).

When the proposed architectures are designed successfully then with the help of mapping report we can find out the area and power consumption and can compare which architecture provides favorable results.

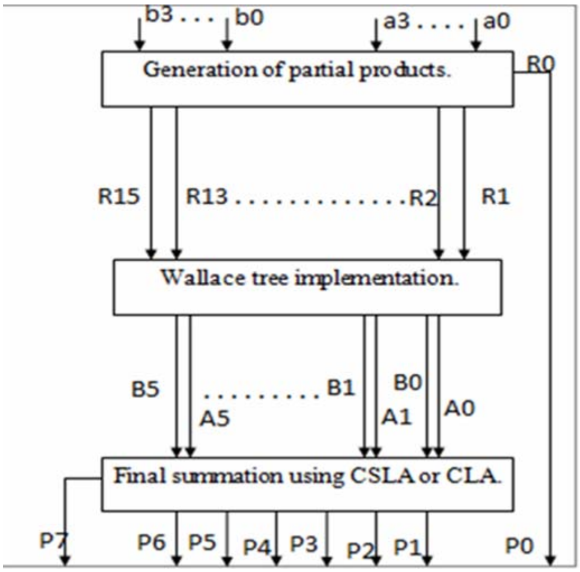

b3 . . . b0
a3 . . . . a0
Generation of partial products.
R0
R15
R13 . . . . .R2
R1
Wallace tree implementation.
B5
. . . . B1
A5
B0
A1
A0
Final summation using CSLA or CLA.
P7
P6
P5
P4
P3
P2
P1
P0

Figure 4: Design Flow of Proposed Work

In this arrangement the partial products with same weight are grouped together and written in a column. There are total six columns in the arrangement in accordance to the max weight carried by partial product term i.e. a3b3 has highest weight 6. Now the addition is applied in each column using half and full adder according to the need, if there are two numbers to be added then half adder is employed and if there are three numbers for addition then full adder has to be used. The main objective of Wallace tree implementation in this architecture is to generate the input terms for adders whether CSLA or CLA. As we know the first partial product a0b0 doesn't need any computation, it is directly taken as the LSB of the product. Therefore R0 is equals to P0. Now the task we have to do is to extract two final bits from each remaining column after excluding the first column from right as it contains R0. Now carefully examining the columns we can see the second column from right has two partial products P4 and P1, which means it has only two bits in all so they can be taken directly and termed as A0 and B0. Moving further to the next column we are having three terms with us P8, P5 and P2, so here we need to apply adders to get final two bits for CSLA/CLA. A half adder is applied with p8 and P5 as inputs which gives two outputs sum and carry, now from this column the sum obtained is taken as B1, the partial product term P2 as A1 and the carry so obtained as A2. Going ahead in this manner, we will get final six pairs of bits as A0B0, A1B1, A2B2, A3B3, A4B4, and A5B5. These numbers so obtained can be treated as two input numbers for the adder. The Wallace tree implementation and the adder stage can be shown in figure below.

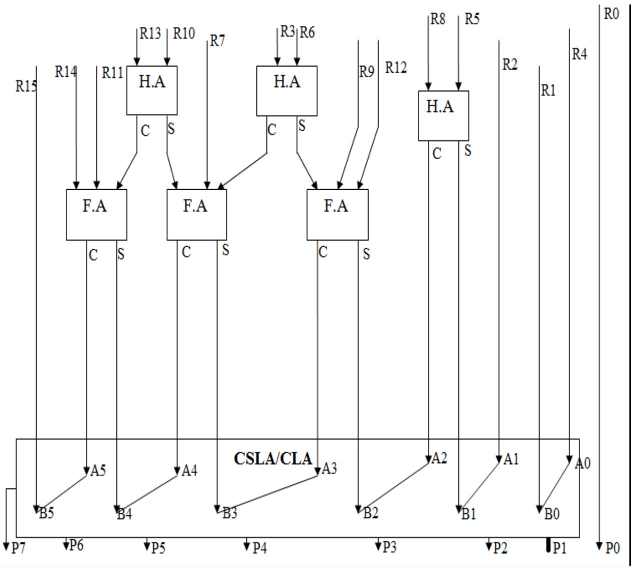


Figure 5: Wallace Tree Multiplier Using CSLA/CLA

After the Wallace tree implementation the last step remains is the addition of the bits obtained. Through Wallace tree computation process we obtain total 12 bits which can be divided into two numbers A (A0, A1, A2, A3, A4, A5) and B (B0, B1, B2, B3, B4, B5). Now the product of the numbers (a) and (b) can be obtained by adding numbers A and B. the LSB of the product is already known which nothing but R0 is. So P0 = R0. Now the summation of A and B will provide the remaining bits of the product. The sum bits at each stage are the product bits and the carry goes to the next stage. And the final carry out is the MSB of the product. Hence we get p1, p2, p3, P4, p5, P6 and p7 after the last step.

The final step of the presented work is the hardware implementation of the multiplier design proposed in this paper on FPGA Spartan 3E. The programming is done in VHDL and with the help of Xilinx tool the code is burnt into the FPGA kit. The Simulation is done in both Xilinx and Modelsim environment. The area and power analysis is conducted using FPGA implementation.

5. Results

The factors that are observed to compute the area consumption are:

- The numbers of LUTs (look up tables) generated.
- Number of slices used.
- Memory area occupied.
- Density of gates.

The power is consumed in two forms:

- Static power consumption
- Dynamic power consumption

5.1 RTL Schematic of the Designs

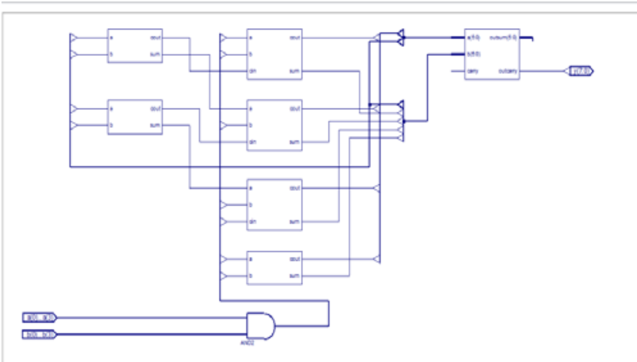


Figure 6: RTL Schematic of Wallace tree multiplier using CSLA

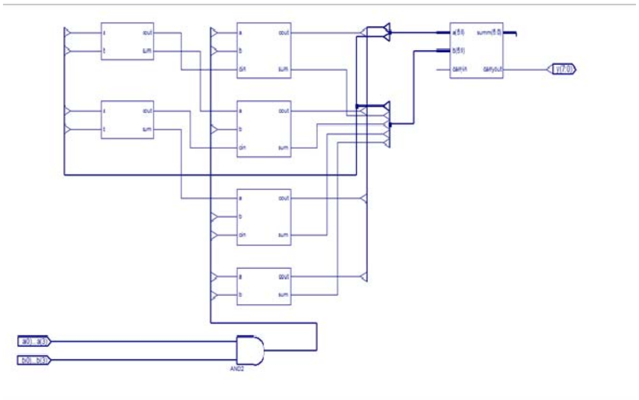
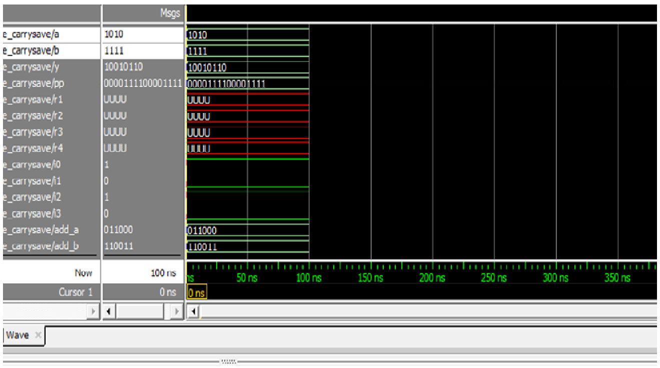


Figure 7: RTL Schematic of Wallace tree multiplier using CLA

5.2 Output Waveforms (using ModelSim)



	Msgs	
e_carrysave/a	1010	1010
e_carrysave/b	1111	1111
e_carrysave/y	10010110	10010110
e_carrysave/pp	0000111100001111	0000111100001111
e_carrysave/r1	UUUU	UUUU
e_carrysave/r2	UUUU	UUUU
e_carrysave/r3	UUUU	UUUU
e_carrysave/r4	UUUU	UUUU
e_carrysave/i0	1	
e_carrysave/i1	0	
e_carrysave/i2	1	
e_carrysave/i3	0	
e_carrysave/add_a	011000	011000
e_carrysave/add_b	110011	110011
Now	100 ns	
Cursor 1	0 ns	0 ns

Figure 8: Output waveform for Wallace tree multiplier using CSLA

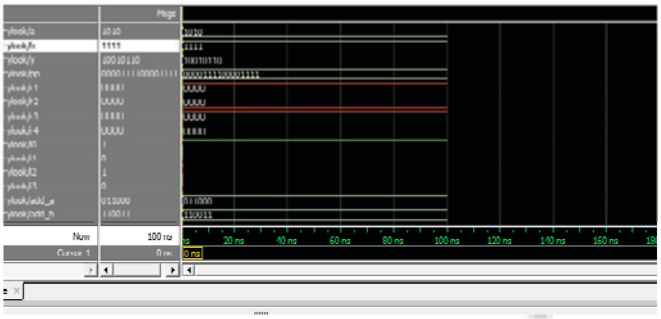


Figure 9: Output waveform for Wallace tree multiplier using CLA

5.3 Area Performance of Wallace Tree Multiplier

Table 1: Comparison on Basis of Area Consumption

Architecture	*XC3S200-4pq208*			
	LUT	*Slices*	*Memory*	*Gate Density*
CSLA	27	15	200688 KB	174
CLA	30	17	200112 KB	162

5.3 Power Performance Reports for Wallace Tree Multiplier

Table 2: Comparison on basis of Power Consumption

Wallace tree multiplier	*Static power dissipation(mW)*	*Dynamic power dissipation(mW)*	*Total power dissipation (mW)*
Using CSLA	103.67	60.98	164.65
Using CLA	107.36	60.98	168.34

5.4 FPGA Implementation Result

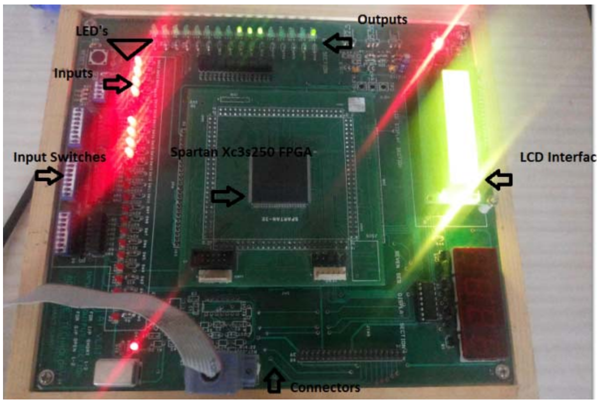


Figure 10: FPGA output for inputs (1111) X (1111)

6. Analysis and Future Scope

There is no doubt left in the fact that multipliers can be considered as the backbone of the processors especially in digital signal processing. As we have passed through the entire thesis, we are now capable of seeing the scope and implementation of the Digital Multipliers especially Wallace tree multiplier. The thesis elaborates all the vital parts of the design such as adders both conventional and fast adders and multipliers with all their types. This finally makes the ground for our thesis and also makes its understanding simpler. The need to develop better and optimized architectures of multipliers has led the journey of multiplier development further and further towards more profitable designs.

The design propounded in this thesis is developed keeping in mind all the constraints present in today's era of technology. The main focus is given to generate the design which is hardware implemental and also consumes less area and power. As area and power factors are the burning demands for the upcoming designs, they have to be considered and analyzed. Therefore we can conclude by looking upon the area and power report so obtained that which architecture is better among the two suggested architectures which come out to be the architecture of Wallace tree multiplier using carry select adder.

7. Conclusion

The analysis of the multiplier architectures designed in this paper work on the basis of the area and power consumption reveals the fact that the Wallace tree multiplier using carry select adder is better in terms of both area and power consumption, which can easily be seen in the comparison tables. The area report of Wallace tree multiplier with CSLA contains 27 LUTs, 15 slices and 162 gate density whereas the Wallace tree multiplier using CLA contains 30 LUTs, 17 slices and 174 gate density. Hence in area optimization aspect the architecture using CSLA for final addition provides better area usage. The total power consumed by CSLA is 164.65 mW whereas that of CLA is 168.34mW.

In future this architecture may be optimized further for more fortified multiplier architecture. The designs of the Wallace tree multiplier are synthesizable, which is verified by the hardware implementation of these designs on FPGA Spartan 3E.

References

[1] OklabdziaVojn G. and Barnes Earl P. "some optimal schemes for ALU implementation in VLSI technology", in proceedings 7th symposium on computer Arithmetic, pages 2-8, June 1985.

[2] "High speed binary adder", IBM journal of research and development 25, may, 1981

[3] "Computer arithmetic algorithm and hardware design" by Behroozparhami

[4] "Design and implementation of faster and low power multiplier" by Parth Sarathy Mohanty, Dept. of E.C.E, NIT Rourkela

[5] "Analysis and implementation of various adders and multipliers with power and Performance perspectives" by Puneet Bhardwaj, Dept of ECE, Thapar University, Patiala (Punjab) , July-2011

[6] "Fast VLSI binary addition" by Keshah K. Parhi, Dept. of ECE, University of Minnesota, MN55455, USA

Author's Profile

Shruti Dixit, Department of Electronics & Communication, Suresh Gyan Vihar University, Jaipur, Rajasthan, India

Praveen Kumar Pandey, Department of Electronics & Communication, Suresh Gyan Vihar University, Jaipur, Rajasthan, India