International
Journal
Of Advanced Research in Engineering & Management (IJAREM)

DESIGN OF LOW POWER MULTIPLIERS

GowthamPavanaskar, RakeshKamath.R, Rashmi, Naveena
Guided by:
DivyeshDivakar
AssistantProfessor
EEE department
Canaraengineering college, Mangalore

Abstract: With advances in technology, the need for designing the multipliers which offer high speed, low power consumption, less area or even combination of them. We try to find the best solution among speed and area in this work.In this paper we have first tried to design different adders and compare their speed and complexity of circuit. And then we have designed Booth multiplier then followed by Wallace Tree multiplier and have compared the speed and Power consumption in them. After designing and comparing the adders we turned to multipliers. Initially we went for Booth's Multiplier and designed Radix-4 modified booth multiplier and analysed the performance. Then we turned to Wallace Tree Multiplier.The result of thiswork helps us to make a proper choice of different multipliers in fabricating different arithmetic units as well as making a choice among different adders in different digital applications according to requirements.

1. INTRODUCTION

A multiplier is one of the key hardware blocks in most digital signal processing (DSP) systems. Typical DSP applications where a multiplier plays an important role include digital filtering, digital communications and spectral analysis. Many current DSP applications are targeted at portable,
battery-operated systems, so that power dissipation becomes one of the primary design constraints. Since
multipliers are complex circuits, they must operate at a high system clock rate. Thus reducing the delay of a multiplier is an essential part of satisfying the overall design. First, with the steady growth of operating frequency and processing capacity per chip, large currents have to be delivered and the heat due to large power consumption must be removed by proper cooling techniques. Second, battery life in portable electronic devices is limited. Low power design directly leads to prolonged operation time in these portable devices.

Multipliers have large area, long latency and consume considerable power. Therefore low-power multiplier design has been an important part in low- power VLSI system design. There has been extensive work on low-power multipliers at technology, physical, circuit and logic levels. A system's performance is generally determined by the performance of the multiplier because the multiplier is generally the slowest element in the system. Furthermore, it is generally the most area consuming. Hence, optimizing the speed and area of the multiplier is a major design issue. However, area and speed are usually conflicting constraints so that improving speed results mostly in larger areas.A fast and accurate operation of a digital system is greatly influenced by the performance of the adders because of their extensive use in multiplication[1].

2. BINARY ADDERS

Addition is the most common and often used arithmetic operation on microprocessor, digital signal processor, especially digital computers. Also, it serves as a building block for synthesis of all other arithmetic operations. Therefore, regarding the efficient implementation of an arithmetic unit, the binary adder structures become a very critical hardware unit.

Although many researches dealing with the binary adder structures have been done, the studies based on their comparative performance analysis are only a few. In this work, qualitative evaluations of the classified binary adder architectures are given. Among the huge member of the adders VHSIC HDL (Hardware Description

Language) code has been written for Ripple-carry, Carry-select and Carry-look ahead adder to emphasize their performance.

2.1 DESIGN OF ADDERS

The addition operations will result in sum value and carry value. All complex adder architectures are constructed from its basic building blocks such as Half Adder (HA) and Full Adder (FA).

With the advances in technology, researchers have tried and are trying to design adders which offer either high speed, low power consumption, less area or the combination of them. The most commonly used adders such as Ripple Carry Adder (RCA), Carry Select Adder (CSlA), Carry Look Ahead Adder (CLA), are discussed and the performance parameters of adders such as area and delay are determined and compared. The each and every adder is named based on the propagation of carry between the stages[2,3].

2.1.1 RIPPLE CARRY ADDERS (RCA)

The architecture of ripple carry adder is composed of cascaded full adders for n-bit adder, as shown in Figure 2.1 which is a 4-bit ripple carry adder. In figure 2.1 input is from the right side because the first cell traditionally represents the least significant bit (LSB). It is constructed by cascading full adder blocks in series. The carry out of one stage is fed directly to the carry-in of the next stage. For an n-bit parallel adder it requires n full adders. Bits X_0 and Y_0 in the figure represent the least significant bits of the numbers to be added. The sum output is represented by thebits S_0, S_1, S_2 and S_3 [4].

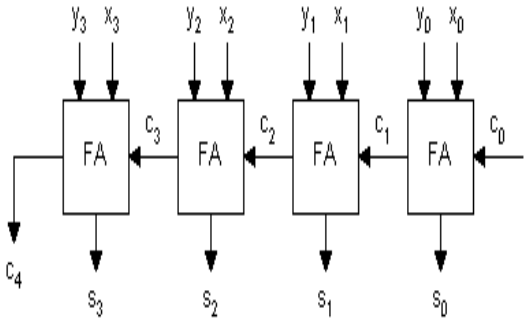


Figure 2.1 A 4-bit Ripple Carry Adder

2.1.2 CARRY SELECT ADDERS (CSlA)

The carry select adder comes in the category of conditional sum adder. Conditional sum adder works on some condition. Sum and carry are calculated by assuming input carry as 1 and 0.Blocks of bits are added in two ways: one assuming a carry-in of 0 and the other with a carry-in of 1. This results in two pre-computed sum and carry-out signal pairs.A 4-bit Carry Select Adder is as shown in the figure 2.2[5,6].

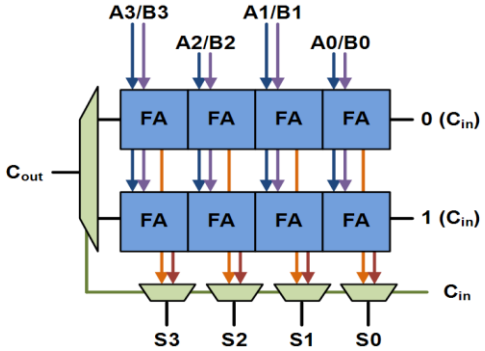


Figure 2.2 A 4-bit Carry Select Adder

2.1.3 CARRY LOOK AHEAD ADDERS (CLA)

Carry Look Ahead Adder can produce carry faster, as the carry bits are generated in parallel by an additional circuitry whenever inputs change. It is based on the fact that a carry signal will be generated in two cases:

- When both bits A_i and B_i are 1
- When one of the two bits is 1 and the carry-in is 1

This technique uses carry bypass logic to speed up the carry propagation

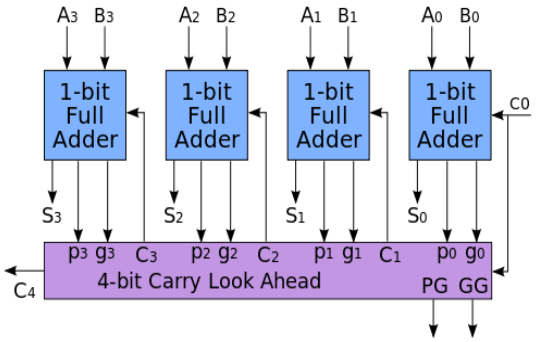


Figure 2.3 Block diagram of Carry look ahead adder

2.1.3.1 LOGIC EQUATIONS

The below equations can be written in terms of two new signals P_i and G_i which are shown in Figure 2.4

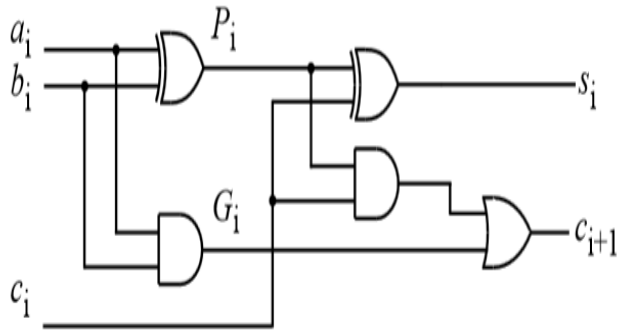


Figure 2.4 Full Adder at stage i with P_i and $G_i$shown.

Let A_i and B_i be the augends and addend inputs, $C_i$the carry input, S_i and C_{i+1}, the sum and carry-out to the i^{th}bit position. If the auxiliary functions, P_i and $G_i$called as propagate and generate signals, the sum output respectively are defined as follows.

$P_i = A_i + B_i$(2.9)

$G_i = A_iB_i$(2.10)

$S_i = A_i \text{ xor} B_i \text{xor} C_i$(2.11)

$C_{i+1} = G_i + P_iC_i$(2.12)

Let's apply this to a 4-bit adder to make it clear.

$C_1 = G_0 + P_0.C_0$(2.13)

$C_2 = G_1 + P_1.G_0 + P_1.P_0.C_0$(2.14)

$C_3 = G_2 + P_2.G_1 + P_2.P_1.G_0 + P_2.P_1.P_0.C_0$(2.15)

$C_3 = G_3 + P_3.G_2 + P_3.P_2.G_1 + P_3.P_2.P_1.G_0 + P_3.P_2.P_1.P_0.C_0$(2.16)

2.1.3.2 COMPLEXITY AND DELAY FOR N-BIT CLA STRUCTURE

$A_{CLA} = O(n) = 14n$ (2.17)

$T_{CLA} = O(\log n) = 4 \log_2 n$ (2.18)

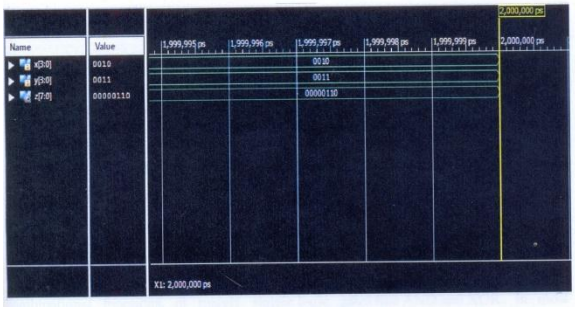
2.2 ANALYSIS OF ADDERS

Three-different adders Ripple Carry Adders, Carry Select Adders and the Carry Look Ahead Adders are compared. The basic purpose was to know the time and power trade-offs between different adders which will give a clear picture of which adder suits best in which type of situation during design process. Table 2.1 represents comparisons of all the three adders which were taken into consideration.

Table 2.1 Theoretical Area Delay Product

Adders	Delay for n-bit	Area for n-bit	Area Delay product
Ripple carry adder	2n	7n	$14n^2$
Carry select adder	$2.8(n)^{1/2}$	14n	$39.6(n)^{3/2}$
Carry look ahead adder	$4\log_2 n$	4n	$16n \log_2 n$

The three adders which were taken into consideration are simulated using VHDL and Xilinix ISE 12.1.Then the performance parameters of the various adders are calculated and compared.



2.5.Simulation result of Radix-2 Booth Multiplier.

So analyzing the above facts we reached at the following conclusions about different adders and intelligent use of them in different circumstances according to the space time trade-off. The results can be summarized as follows.

- Regarding the circuit area complexity in the adder architectures, the ripple-carry adder (RCA) is the most efficient one, but the carry select adder (CSlA) with highest complexity is the least efficient one.

- Considering the circuit delay time, carry select Adder (CSlA) is the fastest one for every n-bit length, so has the shortest delay. Otherwise, ripple carry adder (RCA) is the slowest one, due to the long carry propagation.

- The term area-delay product which gives the clear picture of the space-time trade off. It is worthy to note that while we consider all the adders discussed above ripple carry adders and carry select adders are the two sides of the spectrum. Because, while Ripple Carry Adders have a smaller area and lesser speed, in contrast

to which Carry Select adders have high speed (nearly twice the speed of ripple carry adders) and occupy a larger area. But Carry Look Ahead Adder (CLA) has a proper balance between both the Area occupied and Time required. Hence among the three, carry look ahead adder has the least area delay product.

Hence we should use Carry Look Ahead Adders when it comes to optimization with both Area and Time. For an instance, the last stage of the Booth multiplier is a Carry look Ahead Adder. After the design of adders, different types of multipliers are discussed.

3. MULTIPLIER DESIGN

Binary multiplication is usually performed in digital electronics by using an electronic circuit called as binary multiplier. These binary multipliers are implemented using different computer arithmetic techniques. Booth multiplier that works based on booth algorithm is one of the most frequently used binary multipliers.

3.1 TYPES OF MULTIPLIERS

The following are four important types of multipliers

- Booth's Multiplier
- Array Multiplier
- Baugh Wooley Multiplier
- Wallace Tree Multiplier

3.2.1 BOOTH MULTIPLIER

Booth multiplication algorithm or Booth algorithm was named after the inventor Andrew Donald Booth. It can be defined as an algorithm or method of multiplying binary numbers in two's complement notation. It is a simple method to multiply binary numbers in which multiplication is performed with repeated addition operations by following the booth algorithm. Booth multiplication algorithm consists of three major steps, that includes generation of partial product called as recoding, reducing the partial product in two rows, and addition that gives final product. Again this booth algorithm for multiplication operation is further modified and hence, named as modified booth algorithm.

3.1.1. MODIFIED BOOTH ALGORITHM

Modified BoothMultiplier is more efficient multiplier as compared to other multipliers based on certain parameters like number of input output blocks used, number look up tables used etc. For a better understanding of modified booth algorithm & for multiplication, we must know about each block of booth algorithm for multiplication process.

3.3 ANALYSIS OF MULTIPLIER

We have seen different commonly used multiplier such as Array Multiplier, Baugh Wooley Multiplier, Wallace Tree Multiplier and Modified BoothMultiplier. Hence by literature survey we found that Modified BoothMultiplier is more efficient multiplier as compared to other multipliers based on certain parameters like number of input output blocks used, number look up tables used etc. For a better understanding of modified booth algorithm & for multiplication, we must know about each block of booth algorithm for multiplication process.

4. RADIX-2 BOOTH MULTIPLIER

For unsigned multiplication there is no need to take the sign of the number into consideration. However in signed multiplication the same process cannot be applied because the signed number is in a 2's compliment form which would yield an incorrect result if multiplied in a similar fashion to unsigned multiplication. That's where Booth's algorithm comes in. This multiplier is of the iterative Radix-2 Booth Multiplier type, implemented using asynchronous circuits. Booth algorithm gives a procedure for multiplying binary integers in signed –2's complement representation. Booth's algorithm preserves the sign of the result[7] .

Booth algorithm requires examination of the multiplier bits, and shifting of the partial product. Prior to the shifting, the multiplicand may be added to partial product, subtracted from the partial product, or left unchanged according to the rules as shown in the Table 4.1.

Table 4.1 Rules for Radix-2 multiplier

X(i)	E	Operation
0	0	Right shift
0	1	Add and shift
1	0	Subtract and shift
1	1	Right shift

4.2 STEPS TO CARRY OUT RADIX-2 METHOD

Step 1:

Convert the two operands into binary. Then determine which operand has the least transitions between bits and set X equal to that operand and Y equal to the other operand.

Calculate –Y using 2's complement, to subtract Y from the value in Z which is the same as adding –Y to the value in Z.

Step 2:

Set up 4 columns as follows:

1st Column is X(i) : This holds X operand. This will show each RSA step.

2nd Column is E : This holds the least significant bit from X before RSA.
Initially set this to 0.

3rd Column is Z : This column holds the results from each step in the algorithm

4th Column is V : This column holds the overflow from Z when right-shifting

Step 3:

Analyze the least significant bit of X and the bit E. From that string take the action according to Table 5.1

Step 4:

Right shift X. Go to step 3 and repeat the process until all the bits of the multiplier X has been considered

The flow chart for the Radix-2 Booth multiplier is as shown in the Figure 5.1

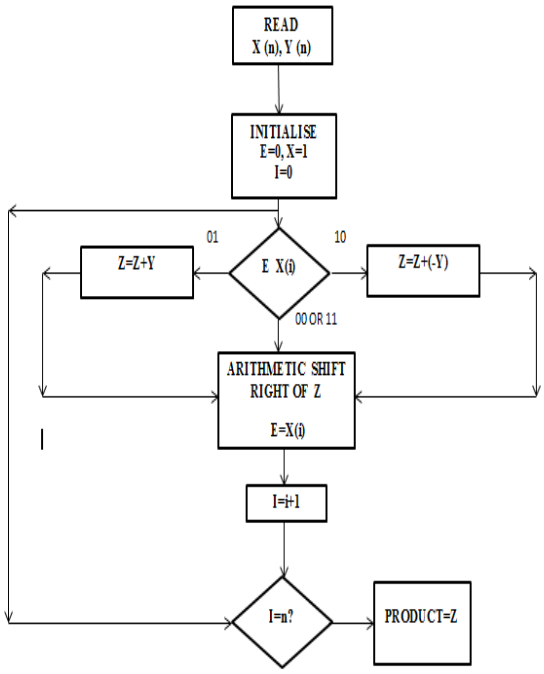


Figure 5.1 Flow chart of Radix-2 multiplier

The Radix-2 Booth multiplier has drawbacks

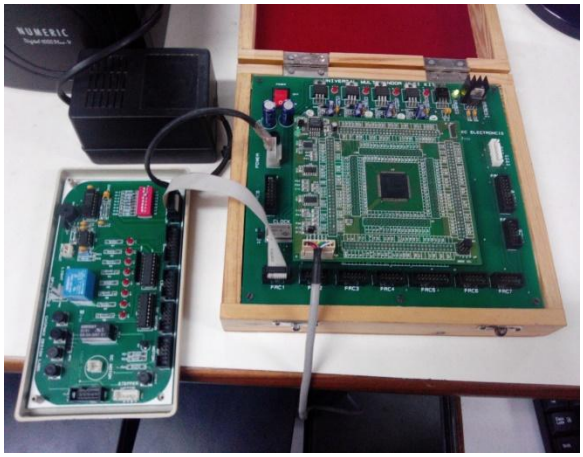
- The number of add or subtract operation became variable and hence became inconvenient while designing Parallel multipliers
- The Algorithm becomes inefficient when there are isolated 1's

5.IMPLEMENTATION OF BOOTH MULTIPLIER

5.1 FPGA IMPLEMENTATION

We have used Spartan-3 FPGA kit to show the outputs of the Booth Radix-2 Multiplier. FPGAstands for Field-Programmable Gate Array. It is an integrated circuit designed to be configured by a customer or a designer after manufacturing, hence "field-programmable". FPGAs contain an array of programmable logic blocks, and a hierarchy of reconfigurable interconnects that allows many logic gates that can be inter-wired in different configurations. Logic blocks can be configured to perform complex combinational functions, or merely simple logic gates like AND and XOR. In most FPGAs, logic blocks also include memory elements, which may be simple flip-flops or more complete blocks of memory.Specific applications of FPGAs include digital signal processing, software-defined
radio, ASIC prototyping, medicalimaging, computervision, speechrecognition, cryptography,
bioinformatics, computer hardware emulation, radio astronomy, metal detection and a growing range of other areas.

FPGAs avoid the high initial cost, the lengthy development cycles, and the inherent inflexibility of conventional ASICs (Application Specific Integrated Circuit). Also, FPGA programmability permits design upgrades in the field with no hardware replacement necessary, an impossibility with ASICs[8].



6. CONCLUSION

We studied Booth Multipliers and Radix-2, comparing Radix-2 and Radix-4 booth multipliers we found that radix-4 consumes less power than radix-2, because radix-4 uses almost half number of iterations than radix-2.We saw Wallace tree having nearly same delay as of radix-4 multipliers whereas consuming a little more power than the former.These two techniques are employed to speed up the multiplication process as their capability to reduce partial products generation and compress partial product term. The system has been designed efficiently using VHDL codes and successfully simulated and synthesized using Xilinx.

We have only considered radix-4 recoding as it is simple and consumes less power. Higher-radix recoding further reduces the number of partial products and thus has the potential of power saving. It can be extended to radix-8, in which partial products will be reduced but the complexity associated is high.

References:

[1]. S.TahmasbiOskuii, P.G.Kjeldsberg, and O.Gustafsson, "Transition activity aware design of reduction-stages for parallel multipliers", in Proc. 17th Great Lakes Symp. On VLSI, March 2007,pp.120-125.
[2]. “Design and Performance analysis of Various Adders using Verilog" IJCSMC, Vol.2, Issue.9, September 2013, Maroju SaiKumar1, Dr.P.Samundiswary.
[3]. Padma Devi, AshimaGirdher, and Balwinder Singh, "Improved Carry Select Adder with Reduced Area and Low Power Consumption", International Journal of Computer Applications, vol.3, no.4, pp-14-18, June 2010.
[4]. Sarabdeep Singh, Dilip Kumar, "Design of Area an Power Efficient Modified Carry Select Adder", International Journal of Computer Applications,vol.33,no.3, pp.14-18, Nov 2011.
[5]. K.H.Tsoi, P.H.W.Leong, "Mullet -a parallel multiplier generator," fpl,pp.691-694, International conference on Field Programmable Logic and Applications, 2005
[6]. Z.Huang, "High Level Optimizaton for Low-Power Multiplier Design,"Phd dissertation, Univ. Of California, Los Angles, June 2003.
[7]. "HDL Programming Fundamentals, VHDL and Verilog", by NazeihM.Botros, Charles River Media, Har/Cdr Publisher, 18 December 2005.
[8]. “Initail Design For spartan-3E Starter Kit(LCD Display Control)", Ken Chapman Xilinx Ltd 16th February 2006.