International Journal of Multidisciplinary Engineering in Current Research
ISSN: 2456-4265, Volume 2, Issue 12, December 2017, http://ijmec.com/

High Speed Modified Booth Encoder for Signed and Unsigned Numbers

SIMHADRI LAKSHMI LAVANYA, P.SRIDHAR REDDY
M. Tech Scholar, Associate Professor
Department of ECE
ISTS Women's Engineering College, Rajanagaram, Rajamahendravaram, A.P, India.

Abstract: - **Emerging non volatile memories (NVMs), such as PRAM, and RRAM, have been widely investigated to replace MRAM as the configuration bits in field-programmable gate arrays (FPGAs) for high security and instant power ON. However, the variations inherent in NVMs and advanced logic process bring reliability issue to FPGAs. This brief introduces a low-power variation-tolerant non volatile lookup table circuit to overcome the reliability issue. Because of large *R*OFF/*R*ON, 1T1R RRAM cell provides sufficient sense margin as a configuration bit and a reference resistor. A single-stage sense amplifier with voltage clamp is employed to reduce the power and area without impairing the reliability. Matched reference path is proposed to reduce the parasitic *RC* mismatch for reliable sensing. Evaluation shows that 22% reduction in delay, 38% reduction in power, and the tolerance of variations of 2.5× typical *R*ON or *R*OFF in reliability are achieved for proposed non volatile LUT with six inputs.**

Keywords: **NVM, FPGA, LUT, PRAM, RRAM**

1. INTRODUCTION

1.1 Motivation

Power dissipation is recognized as a critical parameter in modern VLSI design field. To satisfy MOORE'S law and to produce consumer electronics goods with more backup and less weight, low power VLSI design is necessary.

Fast multipliers are essential parts of digital signal processing systems. The speed of multiply operation is of great importance in digital signal processing as well as in the general purpose processors today, especially since the media processing took off. In the past multiplication was generally implemented via a sequence of addition, subtraction, and shift operations. Multiplication can be considered as a series of repeated additions. The number to be added is the multiplicand, the number of times that it is added is the multiplier, and the result is the product. Each step of addition generates a partial product. In most computers, the operand usually contains the same number of bits. When the operands are interpreted as integers, the product is generally twice the length of operands in order to preserve the information content. This repeated addition method that is suggested by the arithmetic definition is slow that it is almost always replaced by an algorithm that makes use of positional representation. It is possible to decompose multipliers into two parts. The first part is dedicated to the generation of partial products, and the second one collects and adds them.

The basic multiplication principle is two fold i.e. evaluation of partial products and accumulation of the shifted partial products. It is performed by the successive additions of the columns of the shifted partial product matrix. The 'multiplier' is successfully shifted and gates the appropriate bit of the 'multiplicand'. The delayed, gated instance of the multiplicand must all be in the same column of the shifted partial product matrix. They are then added to form the product bit for the particular form. Multiplication is therefore a multi operand operation. To extend the multiplication to both signed and unsigned numbers, a convenient number system would be the representation of numbers in two's complement format.

The MAC (Multiplier and Accumulator Unit) is used for image processing and digital signal processing (DSP) in a DSP processor. Algorithm of MAC is Booth's radix-2 algorithm, Modified Booth Multiplier; 17-bit SPST adder improves speed and reduces the power.

1.2 Speed and Size

In this, when performance of circuits is compared, it is always done in terms of circuit speed, size and power. A good estimation of the circuit's size is to count the total number of gates used. The actual chip size of a circuit also depends on how the gates are placed on the chip – the circuit's layout. Since we do not deal with layout in this report, the only thing we can say about this is that regular circuits are usually smaller than non-regular ones (for the same number of gates), because regularity allows more compact layout. The physical delay of circuits originates from the small delays in single gates, and from the wiring between them. The delay of a wire depends on how long it is. Therefore, it is difficult to model the wiring delay; it requires knowledge about the circuit's layout on the chip. The gate delay, however, can easily be modeled by saying that the output is delayed a constant amount of time from the latest input. What we can say about the wiring delay is that larger circuits have longer wires, and hence more wiring delay. It follows that a circuit with a regular layout usually has shorter wires and hence less wiring delay than a non-regular circuit.

Therefore, if circuit delay is estimated as the total gate delay, one should also have in minded the circuit's size and amount of regularity, when comparing it to other circuits. "Delay" usually refers to the "worst-case delay". That is, if the delay of the output is dependent on the inputs given, it is always the largest possible output delay that sets the speed. Furthermore, if different bits in the output have different worst-case delays, it is always the slowest bit that sets the delay for the whole output. The slowest path between any input bit and any output bit is called the “critical path”. If a circuit is to be speed up, it is always the critical path that should be attacked in the first place.

1.3 Objective

The main objective of this thesis is to design and implementation of a Multiplier and Accumulator. A multiplier which is a combination of Modified Booth and SPST (Spurious Power Suppression Technique) adder are designed taking into account the less area consumption of booth algorithm because of less number of partial products and more speedy accumulation of partial products and less power consumption of partial products addition using SPST adder approach.

1.4 Basics of Multiplier

Multiplication is a mathematical operation that at its simplest is an abbreviated process of adding an integer to itself a specified number of times. A number (multiplicand) is added to itself a number of times as specified by another number (multiplier) to form a result (product). In elementary school, students learn to multiply by placing the multiplicand on top of the multiplier. The multiplicand is then multiplied by each digit of the multiplier beginning with the rightmost, Least Significant Digit (LSD). Intermediate results (partial products) are placed one atop the other, offset by one digit to align digits of the same weight. The final product is determined by summation of all the partial-products. Although most people think of multiplication only in base 10, this technique applies equally to any base, including binary.

Here, we assume that MSB represent the sign of the digit. The operation of multiplication is rather simple in digital electronics. It has its origin from the classical algorithm for the product of two binary numbers. This algorithm uses addition and shift left operations to calculate the product of two numbers. Based upon the above procedure, we can deduce an algorithm for any kind of multiplication. We can check at the initial stage also that whether the product will be positive or negative or after getting the whole result, MSB of the results tells the sign of the product.

1.5 Tools used :

HDL: Verilog
Software Tools : Xilinx 12.1 , modelsim 6.5e

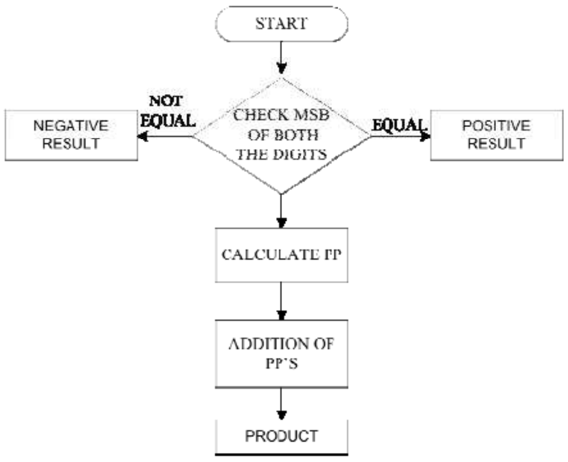


Figure 1.1 Signed Multiplication Algorithm

1.6 Binary Multiplication

In the binary number system the digits, called bits, are limited to the set [0, 1]. The result of multiplying any binary number by a single binary bit is either 0, or the original number. This makes forming the intermediate partial-products simple and efficient. Summing these partial-products is the time consuming task for binary multipliers. One logical approach is to form the partial-products one at a time and sum them as they are generated. Often implemented by software on processors that do not have a hardware multiplier, this technique works fine, but is slow because at least one machine cycle is required to sum each additional partial-product. For applications where this approach does not provide enough performance, multipliers can be implemented directly in hardware. The two main categories of binary multiplication include signed and unsigned numbers. Digit multiplication is a series of bit shifts and series of bit additions, where the two numbers, the multiplicand and the multiplier are combined into the result. Considering the bit representation of the multiplicand x = xn-1…..x1 x0 and the multiplier y = yn-1…..y1y0 in order to form the product up to n shifted copies of the multiplicand are to be added for unsigned multiplication. The entire process

1.7 Multiplication Process

The simplest multiplication operation is to directly calculate the product of two numbers by hand. This procedure can be divided into three steps: partial product generation, partial product reduction and the final addition. To further specify the operation process, let us calculate the product of 2 two’s complement numbers, for example, 1101_2 (-3_{10}) and 0101_2 (5_{10}), when computing the product by hand, which can be described according to figure 1.3.

International Journal of Multidisciplinary Engineering in Current Research
ISSN: 2456-4265, Volume 2, Issue 12, December 2017, http://ijmec.com/

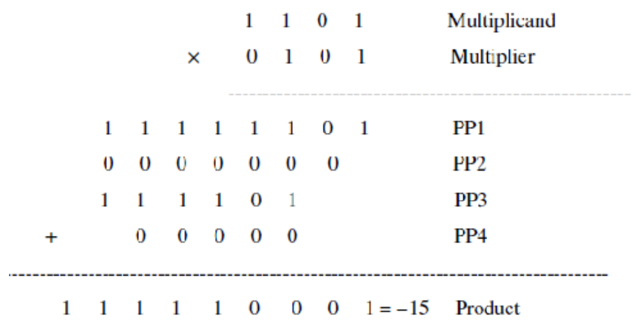


Figure 1.2 Multiplication calculations by hand

The bold italic digits are the sign extension bits of the partial products. The first operand is called the multiplicand and the second the multiplier. The intermediate products are called partial products and the final result is called the product. However, the multiplication process, when this method is directly mapped to hardware. As can been seen in the figures, the multiplication operation in hardware consists of PP generation, PP reduction and final addition steps. The two rows before the product are called sum and carry bits. The operation of this method is to take one of the multiplier bits at a time from right to left, multiplying the multiplicand by the single bit of the multiplier and shifting the intermediate product one position to the left of the earlier intermediate products.

All the bits of the partial products in each column are added to obtain two bits: sum and carry. Finally, the sum and carry bits in each column have to be summed. Similarly, for the multiplication of an *n*-bit multiplicand and an *m*-bit multiplier, a product with *n* + *m* bits long and *m* partial products can be generated. The method shown in figure 1.3 is also called a non-Booth encoding scheme.

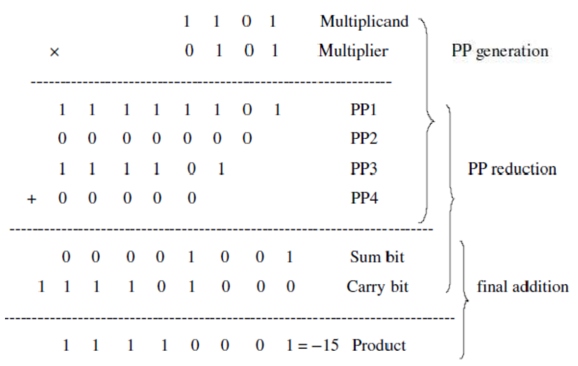


Fig 1.3: Multiplication Operation in hardware

2. LITERATURE SURVEY

2.1 Introducton:

The multiplication operation consists of producing partial products and then adding these partial products the final product is obtained. Thus the speed of the multiplier depends on the number of partial product and the speed of the adder. Since the multipliers have a significant impact on the performance of the entire system, many high performance algorithms and architectures have been proposed [1-23]. The very high speed and dedicated multipliers are used in pipeline and vector computers. The high speed Booth multipliers and pipelined Booth multipliers are used for digital signal processing (DSP) applications such as for multimedia and communication systems. High speed DSP computation applications such as Fast Fourier transform (FFT) require additions and multiplications.

The papers [1, 4] presents a design methodology for high speed Booth encoded parallel multiplier. For partial product generation, a new Modified Booth encoding (MBE) scheme is used to improve the performance of traditional MBE schemes. But this multiplier is only for signed number multiplication operation.

The conventional modified Booth encoding (MBE) generates an irregular partial product array because of the extra partial product bit at the least significant bit position of each partial product row. Therefore papers [2, 3] presents a simple approach to generate a regular partial product array with fewer partial product rows and negligible overhead, there by lowering the complexity of partial product reduction and reducing the area, delay, and power of MBE multipliers. But the drawback of this multiplier is that it function only for signed number operands.

The modified-Booth algorithm is extensively used for high-speed multiplier circuits. Once, when array multipliers were used, the reduced number of generated partial products significantly improved multiplier performance. In designs based on reduction trees with logarithmic logic depth, however, the reduced number of partial products has a limited impact on overall performance. The Baugh-Wooley algorithm [5, 8, 11] is a different scheme for signed multiplication, but is not so widely adopted because it may be complicated to deploy on irregular reduction trees. Again the Baugh-Wooley algorithm is for only signed number multiplication. The array multipliers [12] and Braun array multipliers [13] operates only on the unsigned numbers. Thus, the requirement of the modern computer system is a dedicated and very high speed multiplier unit that can perform multiplication operation on signed as well as unsigned numbers. In this paper we designed and implemented a dedicated multiplier unit that can perform multiplication operation on both signed and unsigned numbers, and this multiplier is called as SUMBE multiplier.

In general, a multiplier uses Booth's algorithm and array of full adders (FAs), or Wallace tree instead of the array of FA's., i.e., this multiplier mainly consists of the three parts: Booth encoder, a tree to compress the partial products such as Wallace tree, and final adder. Because Wallace tree is to add the partial products from encoder as parallel as possible, its operation time is proportional to, where is the number of inputs. It uses the fact that counting the number of 1's among the inputs reduces the number of outputs into. In real implementation, many (3:2) or (7:3) counters are used to reduce the number of outputs in each pipeline step. The most effective way to increase the speed of a multiplier is to reduce the number of the partial products because multiplication precedes a series of additions for the partial products. To reduce the number of calculation steps for the partial products, MBA algorithm has been applied mostly where Wallace tree has taken the role of increasing the speed to add the partial products. To increase the speed of the MBA algorithm, many parallel multiplication architectures have been researched .Among them, the architectures based on the Baugh–Wooley algorithm (BWA) have been developed and they have been applied to various digital filtering calculations.

One of the most advanced types of MAC for general-purpose digital signal processing has been proposed by Elguibaly. It is an architecture in which accumulation has been combined with the carry save adder (CSA) tree that compresses partial products. In the architecture proposed in the critical path was reduced by eliminating the adder for accumulation and decreasing the number of input bits in the final adder. While it has a better performance because of the reduced critical path compared to the previous MAC architectures, there is a need to improve the output rate due to the use of the final adder results for accumulation. An architecture to merge the adder block to the accumulator register in the MAC operator was proposed in to provide the possibility of using two separate /2-bit adders instead of one -bit adder to accumulate the bit MAC results. Recently, Zicari proposed an architecture that took a merging technique to fully utilize the 4–2 compressor. It also took this compressor as the basic building blocks for the multiplication circuit.

A new architecture for a high-speed MAC is proposed. In this MAC, the computations of multiplication and accumulation are combined and a hybrid-type CSA structure is proposed to reduce the critical path and improve the output rate. It uses MBA algorithm based on 1's complement number system. A modified array structure for the sign bits is used to increase the density of the operands. A carry look-ahead adder (CLA) is inserted in the CSA tree to reduce the number of bits in the final adder. In addition, in order to increase the output rate by optimizing the pipeline efficiency, intermediate calculation results are accumulated in the form of sum and carry instead of the final adder outputs.

2.1. Overview of MAC:

A multiplier can be divided into three operational steps. The first is radix-2 Booth encoding in which a partial product is generated from the multiplicand X and the multiplier Y. The second is adder array or partial product compression to add all partial products. The last is the final addition in which the process to accumulate the multiplied results is included.

The general hardware architecture of this MAC executes the multiplication operation by multiplying the input multiplier X and the multiplicand Y. This is added to the previous multiplication result Z as the accumulation step.

The N-bit 2's complement binary number can be expressed as

$$X = -2^{N-1}x_{N-1} + \sum_{i=0}^{N-2} x_i 2^i, \qquad x_i \in 0,1. \quad \text{........(1)}$$

If (1) is expressed in base-4 type redundant sign digit form in order to apply the radix-2 Booth's algorithm.

$$X = \sum_{i=0}^{N/2-1} d_i 4_i \quad \text{...........(2)}$$

$$d_i = -2x_{2i+1} + x_{2i} + x_{2i-1}. \quad \text{...(3)}$$

If (2) is used, multiplication can be expressed as

$$X \times Y = \sum_{i=0}^{N/2-1} d_i 2^{2i} Y. \quad \text{.........(4)}$$

If these equations are used, the afore-mentioned multiplication–accumulation results can be expressed as

$$P = X \times Y + Z = \sum_{i=0}^{N/2-1} d_i 2^i Y + \sum_{j=0}^{2N-1} z_i 2^i. \quad \text{.......(5)}$$

Each of the two terms on the right-hand side of (5) is calculated independently and the final result is produced by adding the two results. The MAC architecture implemented by (5) is called the standard design [6].

If -bit data are multiplied, the number of the generated partial products is proportional to N. In order to add them serially, the execution time is also proportional to N. The architecture of a multiplier, which is the fastest, uses radix-2 Booth encoding that generates partial products. If radix-2 Booth encoding is used, the number of partial products, is reduced to half, resulting in the decrease in Addition of Partial Products step. In addition, the signed multiplication based on 2's complement numbers is also possible. Due to these reasons, most current used multipliers adopt the Booth encoding.

3. ARCHITECTURE OF BOOTH ENCODER FOR DESIGN OF ADD MULTIPLY OPERATOR

3.1 Historical Perspective

The electronics industry has achieved a phenomenal growth over the last two decades, mainly due to the rapid advances in integration technologies, large-scale systems design - in short, due to the advent of VLSI. The number of applications of integrated circuits in high-performance computing, telecommunications, and consumer electronics has been rising steadily, and at a very fast pace. Typically, the required computational power (or, in other words, the intelligence) of these applications is the driving force for the fast development of this field. Figure 1.1 gives an overview of the prominent trends in information technologies over the next few decades. The current leading-edge technologies (such as low bit-rate video and cellular communications) already provide the end-users a certain amount of processing power and portability.

International Journal of Multidisciplinary Engineering in Current Research
ISSN: 2456-4265, Volume 2, Issue 12, December 2017, http://ijmec.com/

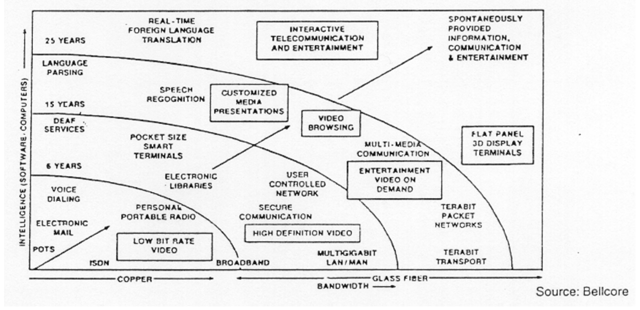


Figure 3.1. Trends in Information Technologies.

This trend is expected to continue, with very important implications on VLSI and systems design. One of the most important characteristics of information services is their increasing need for very high processing power and bandwidth (in order to handle real-time video, for example). The other important characteristic is that the information services tend to become more and more personalized (as opposed to collective services such as broadcasting), which means that the devices must be more intelligent to answer individual demands, and at the same time they must be portable to allow more flexibility/mobility

As more and more complex functions are required in various data processing and telecommunications devices, the need to integrate these functions in a small system/package is also increasing. The level of integration as measured by the number of logic gates in a monolithic chip has been steadily rising for almost three decades, mainly due to the rapid progress in processing technology and interconnect technology. Table 1.1 shows the evolution of logic complexity in integrated circuits over the last three decades, and marks the milestones of each era. Here, the numbers for circuit complexity should be interpreted only as representative examples to show the order-of-magnitude. A logic block can contain anywhere from 10 to 100 transistors, depending on the function. State-of-the-art examples of ULSI chips, such as the DEC Alpha or the INTEL Pentium contain 3 to 6 million transistors.

Table-3.1: Evolution of logic complexity in integrated circuits.

ERA	**DATE**	**COMPLEXITY**
(number of logic blocks per chip)		
Single transistor	1959	less than 1
Unit logic (one gate)	1960	1
Multi-function	1962	2 - 4
Complex function	1964	5 - 20
Medium Scale Integration	1967	20 - 200 MSI)
Large Scale Integration	1972	200 - 2000 (LSI)
Very Large Scale Integration	1978	2000 - 20000(VLSI)
Ultra Large Scale Integration	1989	20000 - ?(ULSI)

The most important message here is that the logic complexity per chip has been (and still is) increasing exponentially. The monolithic integration of a large number of functions on a single chip usually provides:

- Less area/volume and therefore, compactness
- Less power consumption
- Less testing requirements at system level
- Higher reliability, mainly due to improved on-chip interconnects
- Higher speed, due to significantly reduced interconnection length
- Significant cost savings

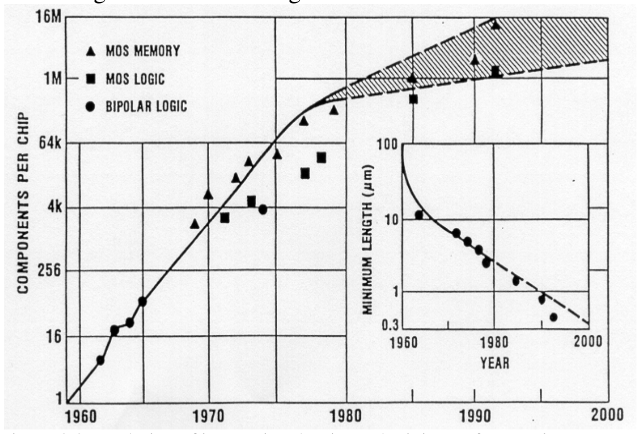


Figure-3.2: Evolution of integration density and minimum feature size, as seen in the early 1980s.

Therefore, the current trend of integration will also continue in the foreseeable future. Advances in device manufacturing technology, and especially the steady reduction of minimum feature size (minimum length of a transistor or an interconnect realizable on chip) support this trend. Figure 3.2 shows the history and forecast of chip complexity - and minimum feature size - over time, as seen in the early 1980s. At that time, a minimum feature size of 0.3 microns was expected around the year 2000. The actual development of the technology, however, has far exceeded these expectations. A minimum size of 0.25 microns was readily achievable by the year 1995. As a direct result of this, the integration density has also exceeded previous expectations - the first 64 Mbit DRAM, and the INTEL Pentium microprocessor chip containing more than 3 million transistors were already available by 1994, pushing the envelope of integration density.

Generally speaking, logic chips such as microprocessor chips and digital signal processing (DSP) chips contain not only large arrays of memory (SRAM) cells, but also many different functional units. As a result, their

International Journal of Multidisciplinary Engineering in Current Research
ISSN: 2456-4265, Volume 2, Issue 12, December 2017, http://ijmec.com/

design complexity is considered much higher than that of memory chips, although advanced memory chips contain some sophisticated logic functions. The design complexity of logic chips increases almost exponentially with the number of transistors to be integrated. This is translated into the increase in the design cycle time, which is the time period from the start of the chip development until the mask-tape delivery time. However, in order to make the best use of the current technology, the chip development time has to be short enough to allow the maturing of chip manufacturing and timely delivery to customers. As a result, the level of actual logic integration tends to fall short of the integration level achievable with the current processing technology. Sophisticated computer-aided design (CAD) tools and methodologies are developed and applied in order to manage the rapidly increasing design complexity.

4. RESULTS AND DISCUSSION

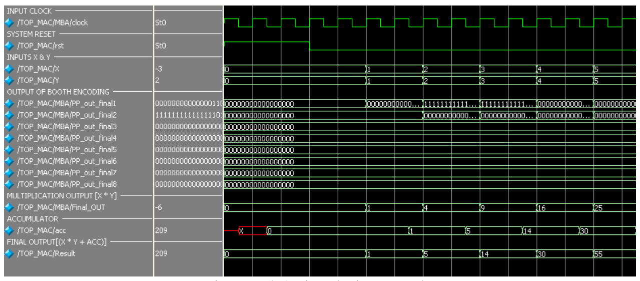
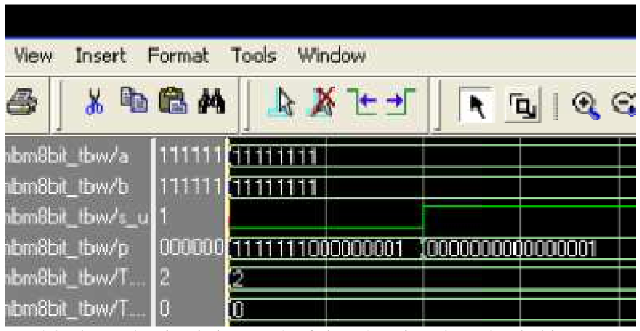
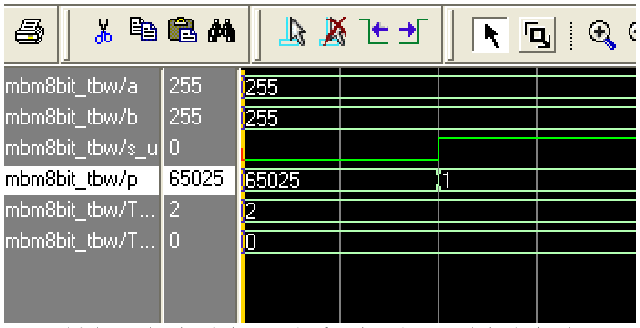
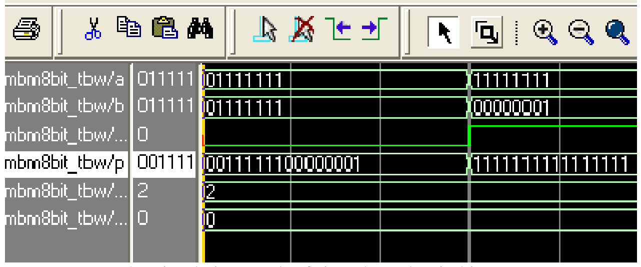


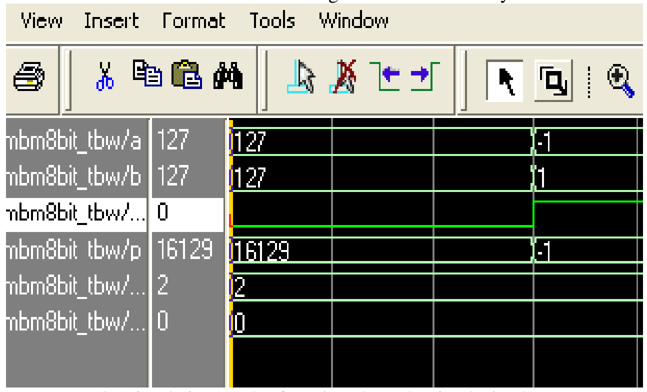
Figure: 4.1 simulation result


View Insert Format Tools Window
mbm8bit_tbw/a 111111 11111111
mbm8bit_tbw/b 111111 11111111
mbm8bit_tbw/s_u 1
mbm8bit_tbw/p 000000 1111111000000001 0000000000000001
mbm8bit_tbw/T... 2 2
mbm8bit_tbw/T... 0 0
4.2 shows the simulation result of signed unsigned number in binary.

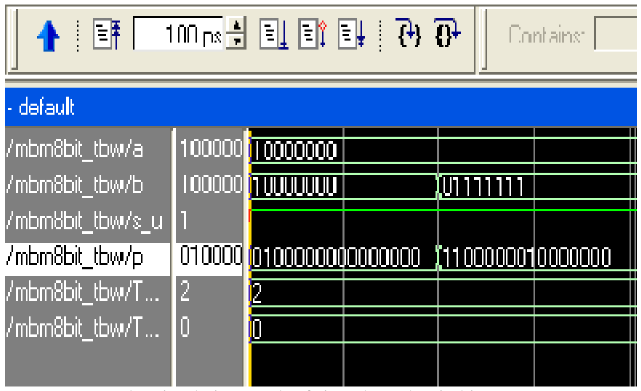


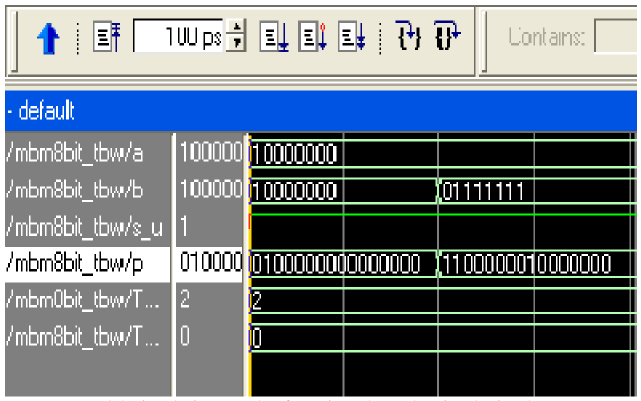
4.3shows the simulation result of unsigned operands in decimal


mbm8bit_tbw/a 011111 01111111 11111111
mbm8bit_tbw/b 011111 01111111 00000001
mbm8bit_tbw/... 0
mbm8bit_tbw/p 001111 0011111100000001 1111111111111111
mbm8bit_tbw/... 2 2
mbm8bit_tbw/... 0 0
4.5 simulation result of signed number in binary

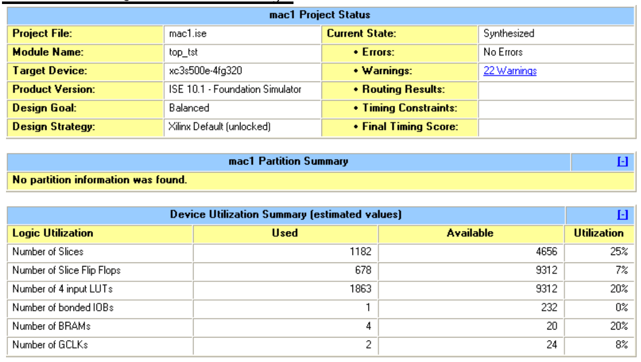


4.6 simulation result of unsigned number in decimal .


100 ps
Contains:
default
/mbm8bit_tbw/a 100000 10000000
/mbm8bit_tbw/b 100000 10000000 01111111
/mbm8bit_tbw/s_u 1
/mbm8bit_tbw/p 010000 0100000000000000 1100000010000000
/mbm8bit_tbw/T... 2 2
/mbm8bit_tbw/T... 0 0
4.7 simulation result of signed number in binary


100 ps
Contains:
default
/mbm8bit_tbw/a 100000 10000000
/mbm8bit_tbw/b 100000 10000000 01111111
/mbm8bit_tbw/s_u 1
/mbm8bit_tbw/p 010000 0100000000000000 1100000010000000
/mbm8bit_tbw/T... 2 2
/mbm8bit_tbw/T... 0 0
4.8 simulation result of unsigned number in decimal

Device utilization summary:



mac1 Project Status			
Project File:	mac1.ise	**Current State:**	Synthesized
Module Name:	top_tst	**• Errors:**	No Errors
Target Device:	xc3s500e-4fg320	**• Warnings:**	22 Warnings
Product Version:	ISE 10.1 - Foundation Simulator	**• Routing Results:**	
Design Goal:	Balanced	**• Timing Constraints:**	
Design Strategy:	Xilinx Default (unlocked)	**• Final Timing Score:**	

mac1 Partition Summary
No partition information was found.

Device Utilization Summary (estimated values)			
Logic Utilization	**Used**	**Available**	**Utilization**
Number of Slices	1182	4656	25%
Number of Slice Flip Flops	678	9312	7%
Number of 4 input LUTs	1863	9312	20%
Number of bonded IOBs	1	232	0%
Number of BRAMs	4	20	20%
Number of GCLKs	2	24	8%

This device utilization includes the following.

- Logic Utilization
- Logic Distribution
- Total Gate count for the Design

The device utilization summery is shown above in which its gives the details of number of devices used from the available devices and also represented in %. Hence as the result of the synthesis process, the device utilization in the used device and package is shown above.

Timing Summary:

Speed Grade: -4

Minimum period: 11.496ns (Maximum Frequency: 86.987MHz)

Minimum input arrival time before clock: 6.892ns

Maximum output required time after clock: 4.394ns

Maximum combinational path delay: No path found

In timing summery, details regarding time period and frequency is shown are approximate while synthesize. After place and routing is over, we get the exact timing summery. Hence the maximum operating frequency of this synthesized design is given as 86.987 MHz and the minimum period as 11.496 ns. Here, OFFSET IN is the minimum input arrival time before clock and OFFSET OUT is maximum output required time after clock.

RTL Schematic

The RTL (Register Transfer Logic) can be viewed as black box after synthesize of design is made. It shows the inputs and outputs of the system. By double-clicking on the diagram we can see gates, flip-flops and MUX.

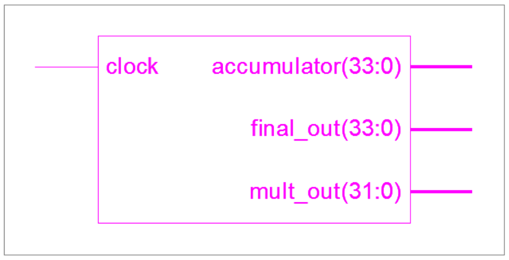
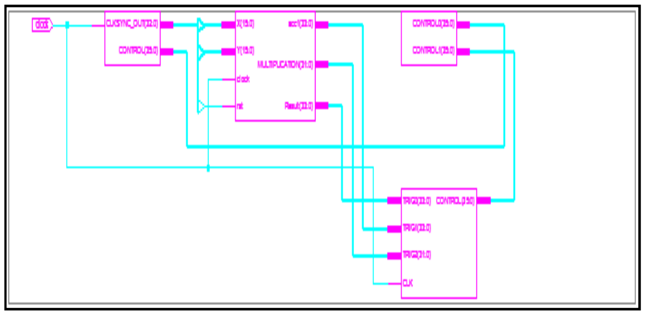


Figure 4.9 Schematic with Basic Inputs and Output

6.2 Summary

- The developed MAC design is modelled and is simulated using the Modelsim tool.
- The simulation results are discussed by considering different cases.
- The RTL model is synthesized using the Xilinx tool in Spartan 3E and their synthesis results were discussed with the help of generated reports.



4.10 MAC Design

5. CONCLUSION

In all multiplication operation product is obtained by adding partial products. Thus the final speed of the multiplier circuit depends on the speed of the adder circuit and the number of partial products generated. If radix 8 Booth encodind technique is used then there are only 3 partial products and for that only one CSA and a CLA is required to produce the final product.

REFERENCES

[1]. W. –C. Yeh and C. –W. Jen, "High Speed Booth encoded Parallel Multiplier Design," IEEE transactions on computers, vol. 49, no. 7, pp. 692-701, July 2000.

[2]. Shiann-Rong Kuang, Jiun-Ping Wang, and Cang-Yuan Guo, "Modified Booth multipliers with a Regular Partial Product Array,"IEEE Transactions on circuits and systems-II, vol 56, No 5, May 2009.

[3]. Li-Rong Wang, Shyh-Jye Jou and Chung-Len Lee, "A well-tructured Modified Booth Multiplier Design" 978-1-4244-1617-2/08/$25.00©2008 IEEE.

[4]. Soojin Kim and Kyeongsoon Cho "Design of High-speed Modified Booth Multipliers Operating at GHz Ranges" World Academy ofScience, Engineering and Technology 61 2010.

[5]. Magnus Sjalander and Per Larson-Edefors. "The Case for HPM-Based Baugh-Wooley Multipliers," Chalmers University of Technology,Sweden, March 2008.

[6]. Z Haung and M D Ercegovac, "High performance Low Power left to right array multiplier design" IEEE trans.Computer, vol 54 no3, page 272-283 Mar 2005.

[7]. Hsing-Chung Liang and Pao-Hsin Huang, "Testing Transition Delay Faults in Modified BoothMultipliers by Using C-testable and SIC Patterns"IEEE2007, 1-4244-1272-2/07.

[8]. Aswathy Sudhakar, and D. Gokila, "Run-Time Reconfigurable Pipelined Modified Baugh-Wooley Multipliers," Advances in Computational Sciences and Technology ISSN 0973-6107 Volume 3 Number 2 (2010) pp. 223–235.

[9]. M D Ercegovac, and T Lang, " Fast multiplication without carry propagate addition" IEEE Transaction on computers, vol.39,

[10]. P.E. Blankenship, " A comment on a two's complement parallel array multiplication algorithm," IEEE Tranaction on computers, vol. 23, no. 12, pp.1327, Dec 1974.

[11]. A. D. Booth, "A Signed Binary Multiplication Technique" Quartely Journal of Mechanics and Applied Mathematics. Vol. 4, no.2pp. 236-240, 1951.

[12]. O. L. MacSorley, "High Speed Arithmetic in Binary Computers," in Proceedings of the IRE, vol. 49, no. 1, January 1961, pp. 67-97.

[13]. J. Liu, S. Zhou, H.Zhu, and C. –K. Cheng. "An Algorithmic Approach for Generic Parallel Adders," in IEEE International Conference on Computer Aided Design, December 2003, pp. 734-740.

[14]. J. Fadavi-Ardekani, ªM×N Booth Encoded Multiplier Generator Using Optimized Wallace Trees,º IEEE Trans. VLSI Systems,vol. 1,no. 2, June 1993.

[15]. A.A. Farooqui et al., ªGeneral Data-Path Organization of a MACUnit for VLSI Implementation of DSP Processors,º Proc. 1998IEEEInt'l Symp. Circuits and Systems, vol. 2, pp. 260-263, 1998.