Case Study: ASIC Implementation of Digital Multipliers

Subha Sri. Thiruveedhi, G. Srikar Babu, Velaga Varun, K Viswanath Chowdary

Assistant Professor, Department of ECE, CVR College of Engineering, Hyderabad, Telangana,

Mail id: rupashubha@gmail.com, Mobile No: +91 9014481363

UG Student, Department of ECE, CVR College of Engineering, Hyderabad, Telangana,

Mail id: gillapally.srikar@gmail.com

UG Student, Department of ECE, CVR College of Engineering, Hyderabad, Telangana,

Mail id: varunvelaga@gmail.com

UG Student, Department of ECE, CVR College of Engineering, Hyderabad, Telangana,

Mail id: viswanath.katragadda@gmail.com

Abstract— Multiplication is an important fundamental function in arithmetic operations. In fact, multiplication based operations such as Multiply and Accumulate (MAC) and inner product are among some of the frequently used computation –intensive arithmetic functions currently implemented in many Digital signal processing (DSP) applications (such as convolution , Fast Fourier Transform (FFT), filtering and others. In this paper, we present the analysis, design and implementation of various Digital Multipliers such as serial multiplier, parallel multipliers and serial – parallel multipliers. In day to day life VLSI field being very prominent to acquire very low area, low power and high accuracy. To obtain these factors various digital multipliers are implemented in ASIC CADENCE by using 45 nano meter technology with subsequent TSMC libraries. By using CADENCE VIRTUOSO tool, can obtain schematics of design and its test bench, power analysis of the design and its simulation wave form.

Keywords— Multiplication, Digital Multiplier, Serial multiplier, Non –additive (Braun) Multiplier, Booth Multiplier, Baugh – Wooley Multiplier, Wallace Tree Multiplier, and Serial – Parallel Multiplier.

INTRODUCTION

Multiplication can be considered as a series of repeated additions. The number to be added is called the multiplicand, the number of times it is added is called the multiplier, and the result obtained is called the product. The basic operations involved in multiplication include generating and accumulating or adding the partial products. The multiplier architecture [11] can be generally classified into the following categories: serial, parallel and serial – parallel. The paper contains following topics (1) Serial Multiplier (2) Parallel Multipliers (3) Serial – Parallel Multiplier. The following figure 1.1 shows the various digital multiplier and its types.

(1) SERIAL MULTIPLIERS

It uses a successive addition algorithm. Simple in structure because both the operands are entered in a serial manner [12]. It requires less hardware and a minimum amount of chip area. The accuracy is poor due to operands being entered sequentially.

(2) PARALLEL MULTIPLIERS

Most advanced digital systems incorporate a parallel multiplication unit to carry out high speed mathematical operations. Today, high speed parallel multipliers with much larger areas and higher complexity are used extensively in RISC, DSP and graphics accelerators. Some examples of parallel multipliers are the array multipliers such as Braun multiplier, Baugh –Wooley multiplier and Tree multipliers like Wallace Tree multiplier etc., these multipliers have regular layout, although tree multipliers are generally faster. The drawback is relatively larger chip area consumption. It presents high speed performance, but it is expensive in term of silicon area and power consumption due to both operands are input to the multiplier in a parallel manner and it is more complex as compared to serial multipliers.

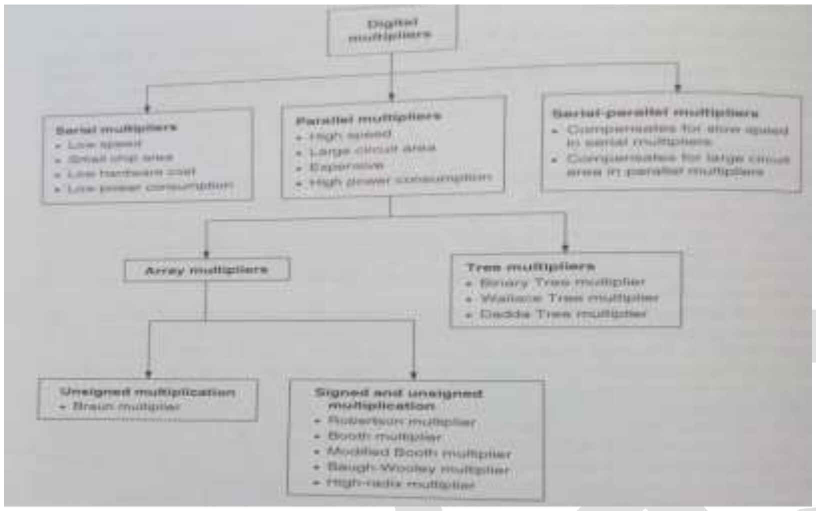


Figure 1.1: Classification of Digital Multiplier

(a) BRAUN MULTIPLIER

Braun Edward Louis first proposed the Braun multiplier in 1963 [1].It is commonly known as the Carry Save Array Multiplier and Non – Additive Multiplier. It consists of an array of AND gates and adders arranged in an iterative structure that does not require logic registers. An n*n Braun multiplier requires n (n-1) adders & n^2 AND gates [2,3]. The efficient implementation of multiplier is shown in the figure 1.2.

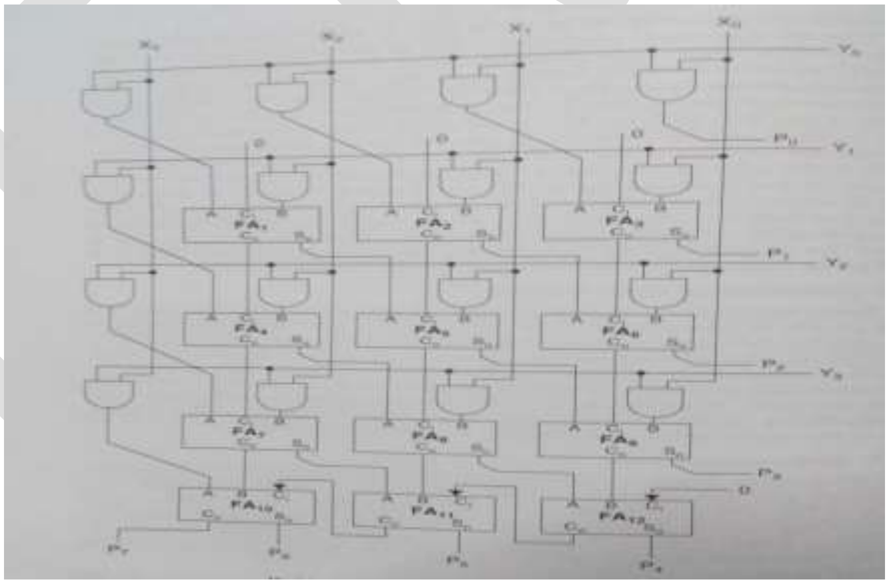


Figure 1.2: Schematic Diagram of a 4*4-bit Braun Multiplier

It makes ideal for VLSI and ASIC realization. Each of the X_iY_j product bits is generated in parallel with the AND gates [2]. Each partial product can be added to the previous sum of partial products by using a row of adders. The carry out signals are shifted 1 bit to the left & are then added to the sums of the first adder and the new partial product. The shifting of the carry out bits to the left is done by a CSA. Instead, the respective carry bit is saved for the subsequent adder stage. RCA are used at the final stage of the array to output the final result. The worst-case multiplication time of a Braun Multiplier can be expressed as [3]

$$t_{Braun} = (n-1)\, t_{carry\text{-}save} + t_{AND} + (n-1)\, t_{ripple\text{-}carry}$$

$t_{carry\text{-}save}$ = time required to generate Carry – out (C_{out})

$t_{ripple\text{-}carry}$ = time taken for the Carry –out (C_{out})

t_{AND} = delay of an AND gate.

It well performs for unsigned operands that are less than 16 bits, in terms of speed, power and area. Replacing the full adders FA1, FA2, FA3… FA12 in figure 1.2 with half adders can enhance the performance of the multiplier. Each replacement will result in a savings of 3 logic gates, even though performance is improved. Optimizing the interconnections between the adders, so that the delay throughput each adders path s is approximately the obtained by optimally interconnecting 2 full adders with fast input and same, can enhance the performance of Braun Multiplier. Figure 1.3 shows a modified Braun multiplier [4] in which two optimally interconnected full adders.

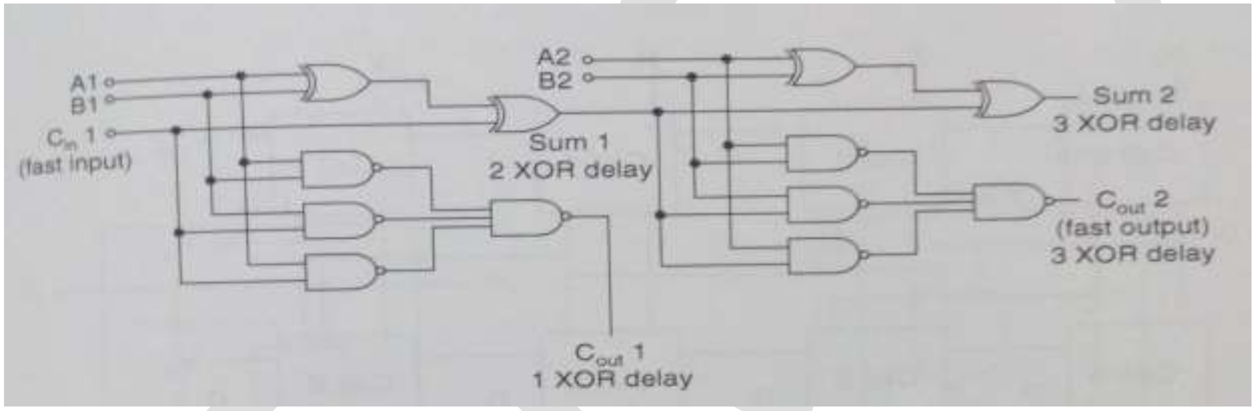


Figure 1.3: Two Optimally Interconnected Full Adders

(b) BAUGH WOOLEY MULTIPLIER

It is enhanced version of Braun multiplier. It is designed to cater to multiplication of both signed and unsigned operands [5], which are represented in the 2's complement number system. The algorithm is also based on the carry-save algorithm. The structure of a 4*4 – bit 2's complement multiplier is shown in the figure 1.4, with the cell number representing the type of the basic cell. The macroscopic views of these basic cells are shown in the figure 1.5. [2,6]

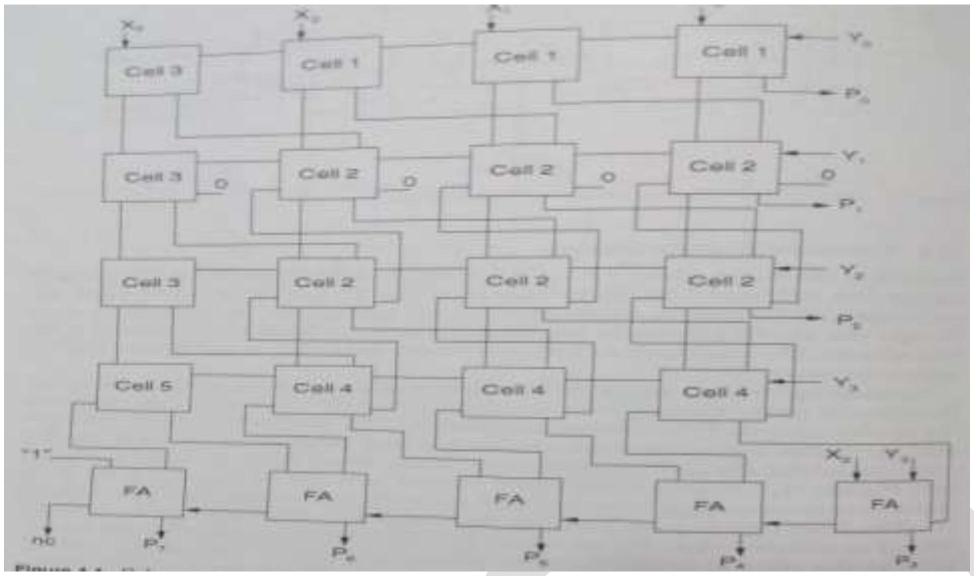


Figure 1.4: Schematic Circuit of a 4*4-bit Baugh-Wooley Multiplier

It operates on signed operands with 2's complement representation to make sure that the signs of all the 2's complement numbers X and Y. The product of X and Y is expressed as P=XY.

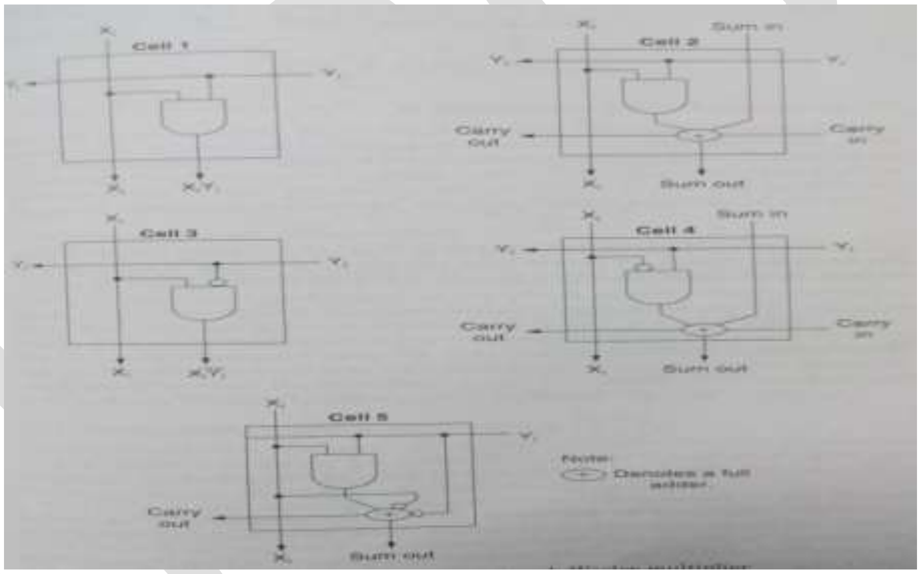


Figure 1.5: The Five Basic Building of the Baugh-Wooley Multiplier

Using a step by step approach, this 2's complement multiplication algorithm can be converted into an equivalent parallel array expression, as adopted by the Baugh – Wooley multiplier. Each partial product bit is the result of an AND gate of a multiplier. Each column represents the addition in accordance with the respective weight of the product term. In this scheme, a total of n (n-1) +3 full adders are required. Hence, for the case of n=4, the array requires 15 adders. Signed multiplicands must first be converted into their 2's complement representation before multiplication. A 2's complement generator is shown in the figure 1.6.

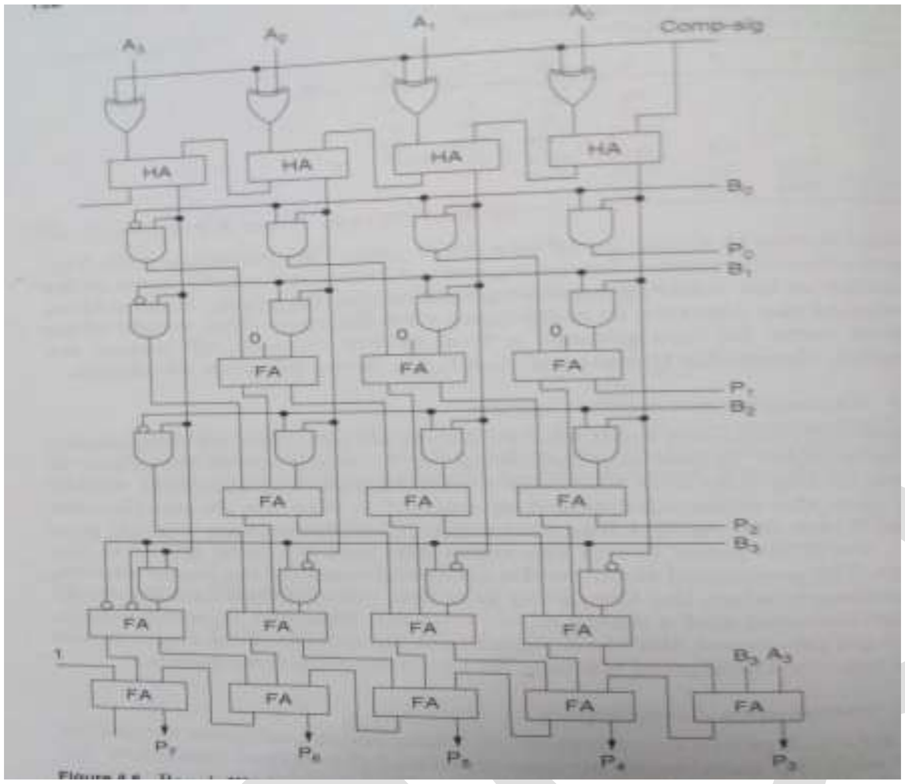


Figure 1.6: Baugh –Wooley Multiplier with 2's Complement Generator

It is basically a combinational circuit that will either pass the input unchanged or convert it into 2's complement form. When the control line comp-sig goes high, the XOR gates invert the input bits and a 1 gets added to the result. The generated result is the 2's complement of the input bits. When the comp-sig goes low, the multiplicand inputs do not get inverted and a 0 gets added to them. The area & power consumption of a number of multiplier structures vary with the number of bit operands and the layout [7] strategies. Since the Baugh-Wooley multiplier is an evolvement of the Braun multiplier, its performance can also be improved by using the earliest mentioned optimized interconnections as shown in the figure 1.6.

(c) BOOTH MULTIPLIER

Area efficient and fast multipliers are the essential blocks for high performance computing. Therefore, multipliers should be small enough so that a larger number of them may be integrated on a single chip. Conventional array multipliers, like the Braun multiplier and the Baugh –Wooley multiplier, achieve comparatively good performance [8] , but they require large areas of silicon, unlike the add shift algorithms, which require less hardware and exhibit poorer performance. This algorithm makes use of the booth encoding algorithm in order to reduce the number of partial products by considering 2 bits of the multiplier at a time, thereby achieving a speed advantage over other bit. The block diagram of an n*n bit modified booth multiplier is shown in the figure 1.7.

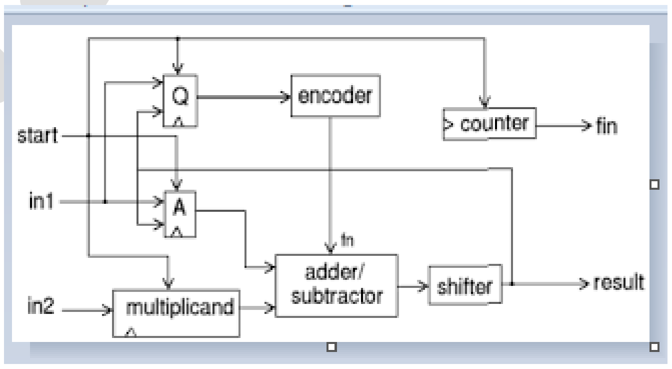


Figure 1.7: Block Diagram of an n*n –bit modified Booth Multiplier

The Booth encoder implements Booth encoding of the three multiplier bits and also handles the sign extension logic. Each encoder is dedicated to one partial product in the array. Since there is a circuit for each of the five possible generated partial product signals, one and only one signal is high during the steady state operation. The carry propagation circuits [4] are independent of the partial product circuits and they do not share any inputs. The structure of the Booth encoder is shown in figure 1.8.

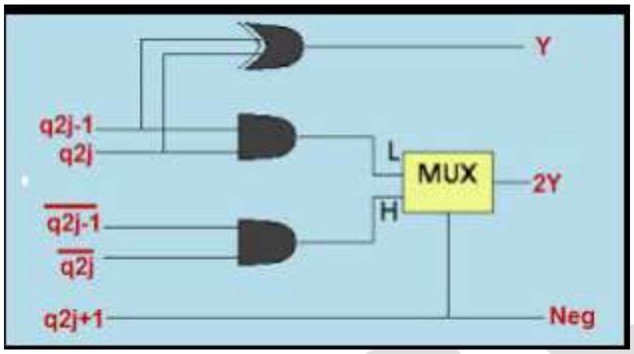


Figure 1.8: Structure of Booth Encoder

An extra output signal PL_j is also provided to represent the selection of the positive partial product. The sign – select Booth encoder is given in figure. Using this Booth encoder, the partial product generator can also be simplified, as shown in the figure 1.9. The simplicity of this circuit leads to smaller area usage and low-power consumption.

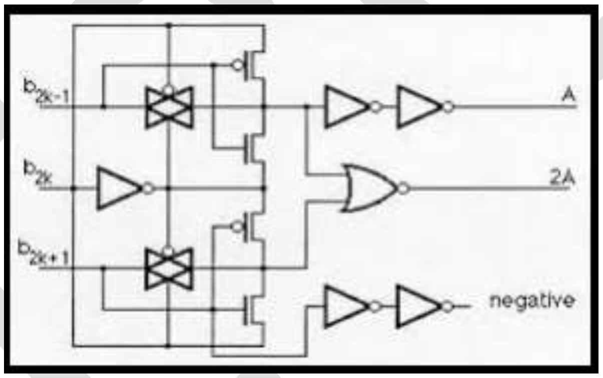


Figure 1.9: Sign-Select Booth encoder

(d) WALLACE TREE MULTIPLIER

Booth’s algorithm effectively reduces the number of partial products by half. The Wallace tree multiplication algorithm, however, can reduce the number of partial products by employing multiple input compressors capable of accumulating several partial products concurrently. In 1964, C.S. Wallace proposed the multiplier [9], which can handle the multiplication process for large operands. This is achieved by minimizing the number of partial product bits in a fast and efficient way by means of a CSA tree constructed from 1-bit full adders. This algorithm uses pseudo adders to add 3 inputs and then produces 2 outputs whose sum is equivalent to the sum of the 3 inputs. The main advantage of using these pseudo adders comes from having the ability to avoid carry propagation, which then increases the speed of multiplication. Using several pseudo adders concurrently can reduce the number of partial products, in parallel, with a resulting overall delay proportional to log3/2 n [10], for n number of rows. The main disadvantage of the Wallace tree algorithm is that the architecture exhibits some irregularities in the layout because it has a relatively complicated interconnection scheme. In general, its multiplication process can be summarized as follows:

a) After generating the partial products, a set of counters reduces the partial product matrix but it does not propagate the carries.

b) The resulting matrix is composed of the sums and carries of the counters.

c) Another set of counters then reduces this matrix and the whole process continues until a two row matrix is generated.

The two rows get summed up with a final adder, preferably by a carry propagate adder. This method takes advantage of the carry save architecture in order to avoid the carry propagation until the final adder. In this scheme, the number of levels is crucial since they determine the speed of the multiplier. The conventional Wallace Tree algorithm reduces the propagation stages by incorporating 3:2 compressors [9]. The following figure 1.10 shows the Schematic of Wallace tree multiplier.

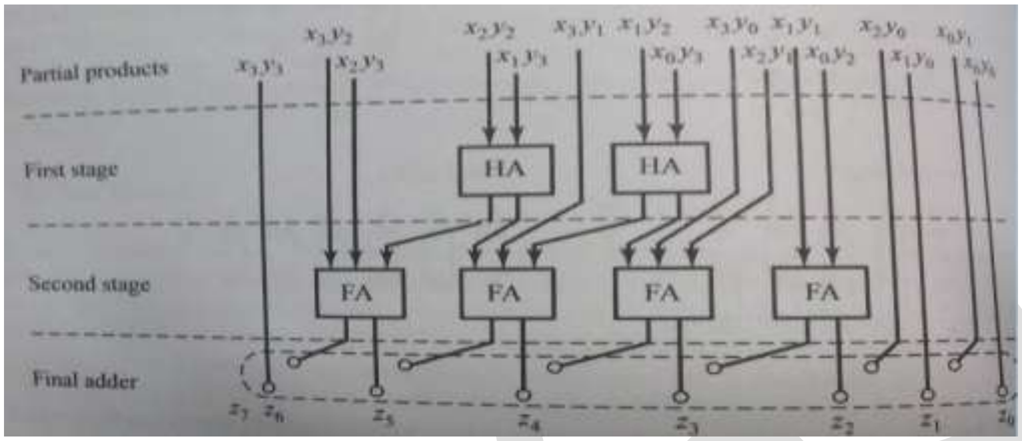


Figure 1.10: Schematic of 4-bit Wallace tree Multiplier

3) SERIAL PARALLEL MULTIPLIERS

It serves as a good tradeoff between the time consuming serial multiplier and the area consuming parallel multipliers. In a device using the serial – parallel multiplier, one operand is entered serially and the other is stored in parallel with a fixed number of bits. The resultant enhancement in the processing speed and the chip area will become more significant when a large number of independent operations are performed.

SIMULATION RESULTS

As mentioned above all the various digital multipliers such as serial multiplier, parallel multiplier (Braun multiplier, Baugh –Wooley multiplier, Wallace tree multiplier), serial parallel multiplier. Here TSMC 45 micro meter technology libraries are used for gates creations, VIRTUOSO schematic editor is used for drawing of schematics and VIRTUOSO symbol editor is used for drawing of symbol. The transient analyses with the time period of 100ns are taken for various digital multipliers. The following figure 1.11 shows the Schematics of digital multipliers.

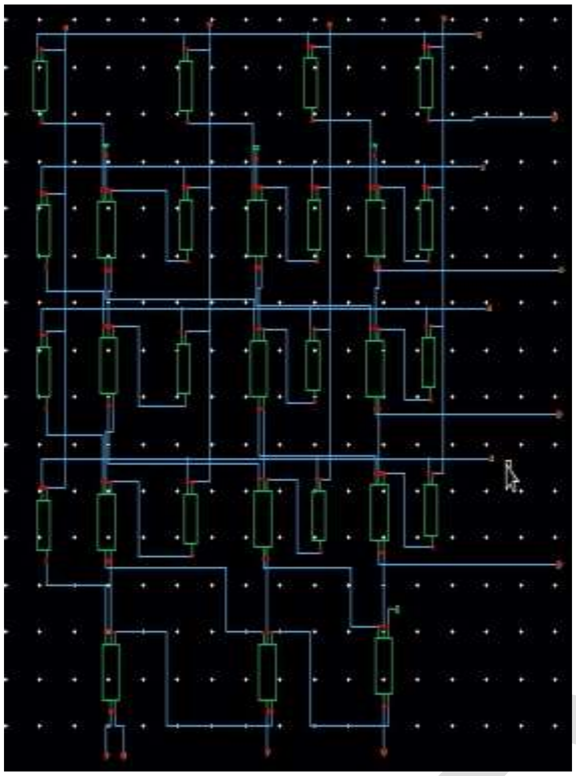


Figure a) Schematic of Braun Multiplier

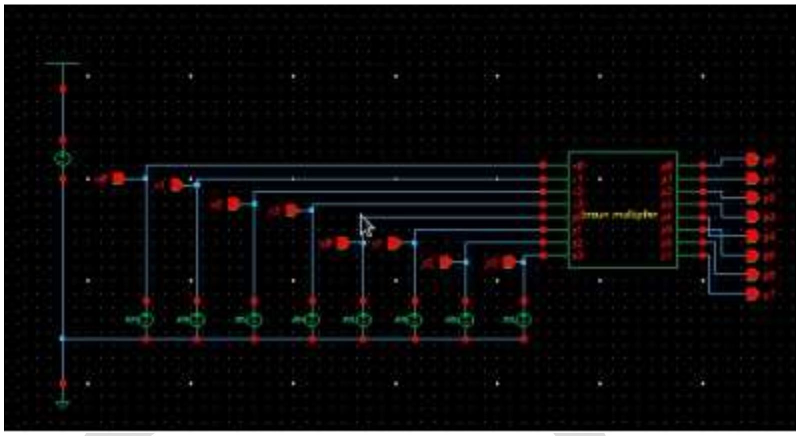


Figure b) Test Bench Schematic of Braun Multiplier

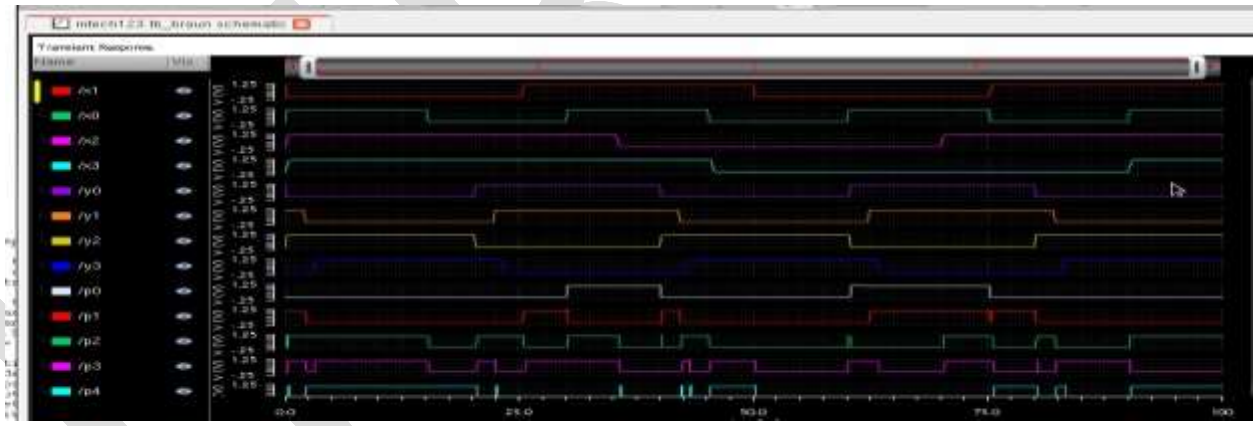


Figure c) Simulation Wave form of Braun Multiplier

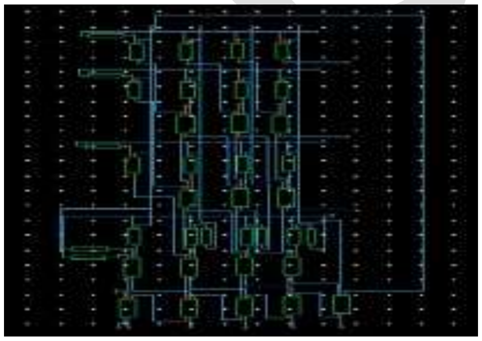


Figure d) Schematic of Baugh-Wooley Multiplier

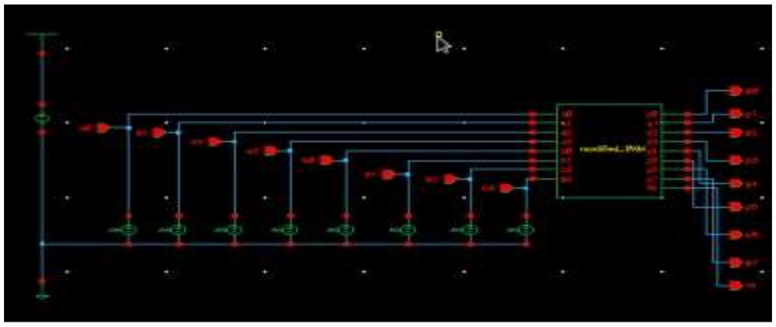


Figure e) Test Bench Schematic of Baugh-Wooley Multiplier

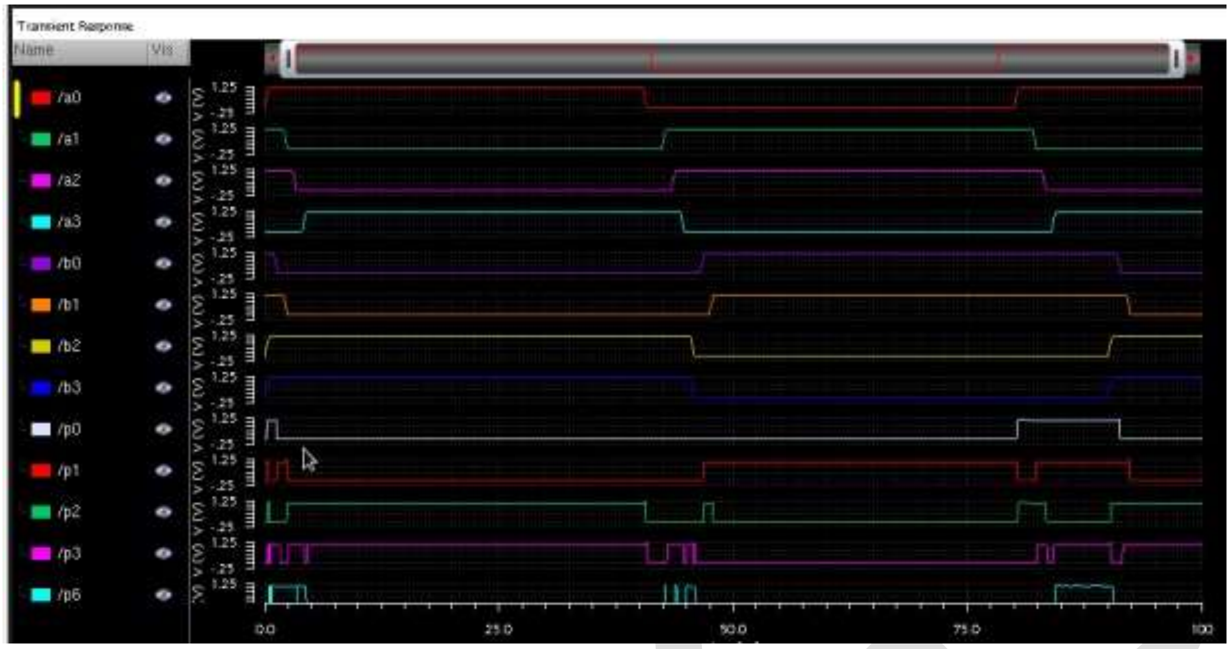
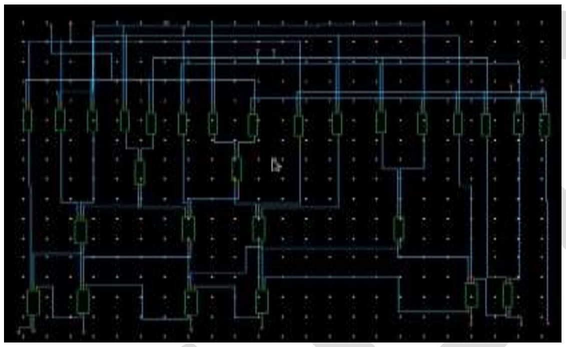


Figure f) Simulation Wave form of Baugh-Wooley Multiplier



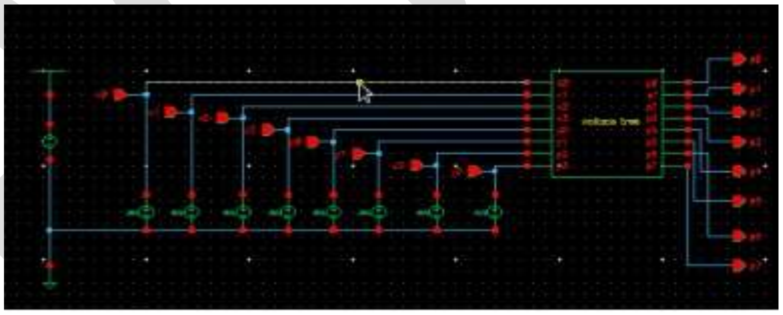


Figure g) Schematic of Wallace Tree Multiplier

Figure h) Test Bench Schematic of Wallace Tree Multiplier

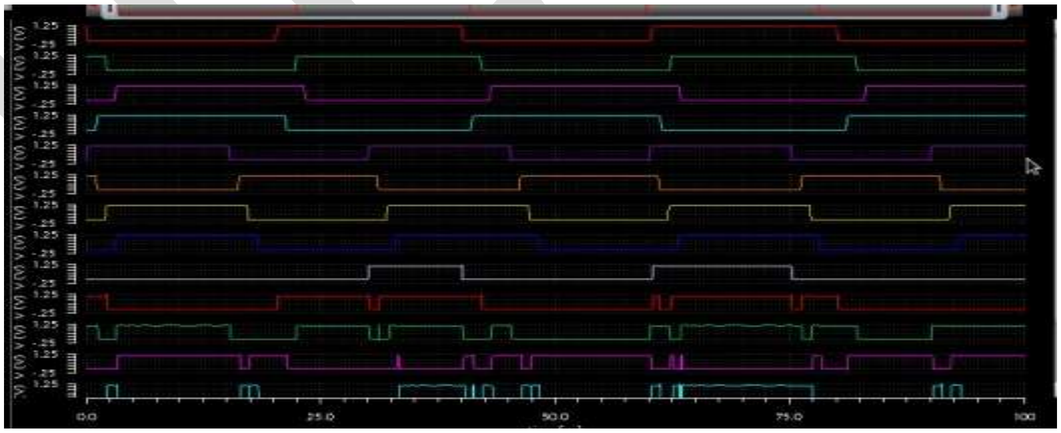


Figure i) Simulation Wave form of Wallace Tree Multiplier

CONCLUSION

This paper gives a glimpse of the multiplication techniques, types of multiplier architectures and a wide ranging review of the different multiplication algorithms. A myriad of eminent architectures of multipliers, including Braun multiplier, Baugh Wooley multiplier, Booth's multiplier and Wallace Tree multiplier have been covered. This paper presents all the schematics and simulation of various digital filters and in all the multipliers Braun multiplier will consumes less power.

REFERENCES:

[1] E.L.Braun, Digital Computer Design, Logic Circuitry, Synthesis, Academic Press, New York, 1963.
[2] A.Bellaaouar and M.I.Elmasry, Low-Power Digital VLSI Design: Circuits and Systems, Kluwer Academic Publishers, 1995.
[3] K.Hwang, Computer Arithmetic: Principles, Architecture and Design, Wiley, 1979.
[4] P.E.Madrid, B.Miller, and E.E Swartzlander, "Modified Booth Algorithm for High Radix Fixed Point Multiplication", IEEE Transaction on VLSI Systems, Vol 1, No 2, June 1993.
[5] C.R.Baugh and B.A.Wooley,"A two's Complement Parallel Array Multiplication Algorithm," IEEE Transactions on computers, Vol 22, No. 12, Dececember 1973.
[6] M.S.Elrabaa, I.S.Abu-Khaterand, and M.I.Elmasry, Advanced Low Power Digital Circuit Techniques, Kluwer Academic Publishers,1997.
[7] D.A.Pucknell and K.Eshranghin, Basic VLSI Design, 3rd ed., Prentice Hall, 1994.
[8] J.M.Rabaey and M.Pedram, Low Power Design Methodologies, Kluwer Academic Publishers,1996.
[9] C.F.Law, S.S.Rofail, and K.S.Yeo,"Low Power Circuit Implementation for Partial Product Addition Using Pass Transistor Logic," IEEE Proceedings Circuits Devices Systems, Vol 146, No.3, June 1999.
[10] A.D.Booth,"A Signed Binary Multiplication Technque,"Quarterly J.Mechanics and Applied Mathematics," Vol.4, Pt 2, June 1951.
[11] B.Parhami,Compter Arithmetic: Algorithms and Hardware Design, Oxford University press, New York,2000
[12] A. Clements, The Principles of Computer Hardware, 3rd ed., Oxford University press, New York, 2000.